University of Warsaw
Faculty of Mathematics, Informatics and Mechanics

Sebastian Stawicki

Ensembles of Classifiers Based on Decision Bireducts

PhD dissertation

Supervisor

Prof. dr hab. Dominik Ślęzak

Institute of Informatics
University of Warsaw

May 2023

Author’s declaration:
aware of legal responsibility I hereby declare that I have written this dissertation myself and all the contents of the dissertation have been obtained by legal means.

May 15, 2023	....
date	*Sebastian Stawicki*

Supervisor’s declaration:
the dissertation is ready to be reviewed

May 15, 2023	....
date	*Prof. dr hab. Dominik Ślęzak*

Abstract

In this dissertation, we present decision bireducts, an extension of decision reducts in the theory of rough sets, the emphasis of which is on both a subset of attributes which describes decisions and a subset of objects for which that description is valid. We investigate their relationship with approximate decision reducts which were developed to handle large and noisy data. Additionally, we demonstrate how decision bireducts can be used as rule-based classifiers that provide greater flexibility in assigning decision values to objects when compared to approximate decision reducts. Moreover, we present theoretical results on the properties of decision bireducts as well as algorithms for their effective computation. Furthermore, we show that obtaining optimal bireducts with respect to given criteria is an NP-hard task.

In the dissertation we also investigate an important aspect of creating ensembles of decision bireducts. Ensembles of classifiers based on different approximate decision reducts can repeatedly misclassify the same data instances. In contrast, decision bireducts provide flexibility in selecting objects whose decision values are accurately described by the given set of attributes. This flexibility enables us to verify and potentially influence the creation of the ensemble, that assures avoidance of repeating errors on the same areas of the training data.

One of the aspects of interpretability in machine learning, which is often required in practical applications, is the ability to report the importance of particular attributes used in the prepared models. Therefore we present decision bireducts ensembles ability to provide feature importance scores. Moreover, to further increase the usefulness of such results, we introduce an approach to the evaluation of attribute scores produced by any machine learning method.

The results presented in the dissertation are supported by examples, while the practical usefulness of decision bireducts is demonstrated by the results of prepared experiments on both benchmark and real-world data. We also comprehensively present a case study demonstrating the application of decision bireducts ensembles to a decision problem encountered while developing a solution for an HR company specializing in the recruitment of IT professionals.

Streszczenie

W niniejszej rozprawie przedstawiamy pojęcie bireduktów decyzyjnych, które są rozszerzeniem reduktów decyzyjnych z teorii zbiorów przybliżonych, gdzie nacisk położony jest zarówno na podzbiór atrybutów opisujących decyzje jak i podzbiór obiektów, dla których ten opis jest prawidłowy. Badamy związek bireduktów decyzyjnych z przybliżonymi reduktami decyzyjnymi, które zostały opracowane na potrzeby przetwarzania dużych i zaszumionych danych. Pokazujemy, jak biredukty decyzyjne mogą być użyte jako klasyfikatory regułowe, które zapewniają większą elastyczność w przypisywaniu wartości decyzyjnych obiektom w porównaniu do metod bazujących na reduktach przybliżonych. Przedstawiamy teoretyczne wyniki dotyczące własności bireduktów decyzyjnych, jak również algorytmy służące ich efektywnemu obliczaniu. W szczególności pokazujemy, że zadanie szukania optymalnych bireduktów przy zadanych kryteriach jest zadaniem NP-trudnym.

W rozprawie omawiamy także kwestię tworzenia zespołów klasyfikatorów wykorzystujących biredukty decyzyjne. Przybliżone redukty decyzyjne wykorzystywane w zespole klasyfikatorów wspólnie mogą błędnie klasyfikować pewne obiekty. Tymczasem, dzięki elastyczności bireduktów decyzyjnych w kontekście wyboru obiektów, których wartości decyzyjne są prawidłowo opisywane przez dany zbiór atrybutów, w trakcie tworzenia zespołu klasyfikatorów możemy unikać powtarzania błędów popełnianych przez poszczególne klasyfikatory na tych samych obszarach danych treningowych.

Zespoły bireduktów decyzyjnych należy traktować jako zespoły klasyfikatorów interpretowalnych, gdzie każdy biredukt utożsamia się ze zbiorem prostych reguł decyzyjnych wyznaczanych przez dany podzbiór atrybutów. Jednym z istotnych aspektów interpretowalności w uczeniu maszynowym, który jest obecnie często wymagany w praktycznych zastosowaniach, jest także możliwość określenia istotności poszczególnych atrybutów wykorzystanych w przygotowywanych modelach. W związku z tym, w rozprawie pokazujemy jak określić istotność atrybutów wykorzystywanych przez zespoły bireduktów decyzyjnych. Ponadto, wprowadzamy nowe podejście do oceny i porównywania metod pomiaru istotności atrybutów bazujących na dowolnych metodach uczenia maszynowego.

Dla rezultatów przedstawionych w rozprawie prezentujemy intuicyjne przykłady, a praktyczną użyteczność bireduktów decyzyjnych pokazujemy w oparciu o wyniki eksperymentów przygotowanych na danych referencyjnych, jak i danych rzeczywistych związanych z praktycznymi zastosowaniami. Pokazujemy również kompleksowo studium przypadku, w którym zastosowane zostały zespoły bireduktów decyzyjnych do rozwiązania problemu decyzyjnego napotkanego podczas realizacji projektu dla firmy HR specjalizującej się w rekrutacji ekspertów IT.

Contents

Chapter 1

Introduction

The notion of decision bireducts is the key concept of this dissertation. It is a new extension of the notion of decision reducts from the theory of rough sets proposed in [140] and further developed in [117, 118, 119, 142, 59]. The notion is developed in order to provide a simple and interpretable concept of data reduction that is in many ways more flexible and general than its classical counterparts. The goal of this dissertation is to present the entirety of the results on this topic in a coherent and clear way.

Nevertheless, we should start by showing the context from which decision bireducts originate. The rough set theory, which is the extension of the classical set theory, is suitable for describing concepts in presence of incomplete information [91]. The indiscernibility relation, which is an equivalence relation, is the fundamental concept of this theory, using which one can group objects that cannot be distinguished from each other based on a given subset of attributes. The relation partitions the universe of objects into equivalence classes which can be used to handle imperfect knowledge about objects and concepts using the notions of lower and upper approximations. In the theory the notion of decision reducts is regarded as one of the fundamental concepts, and it is defined as a subset of attributes that preserves the characteristic of the indiscernibility relation as it is available with respect to the full set of attributes [95]. The notions analogous to decision reducts occur in many areas of science, such as Markov blankets in probabilistic modeling or irreducible multi-valued dependencies in relational databases [137]. Decision reducts have a range of applications in different domains including attribute selection and knowledge discovery [20]. There is also a number of thorough theoretical investigations [82] and software implementations allowing to better understand and apply decision reducts in practice [53].

The concept of decision reduct is fundamental to the rough set theory and its applications. However, in practical applications the data sets often involve large and noisy data. To address such challenges, several generalizations of the notion of decision reducts have been proposed; among them the notions of dynamic reducts [4] and approximate decision reducts [131] are a few to name.

An approximate decision reduct can be defined as an irreducible subset of attributes that, under specified criterion, retains the decision-related information of a data table intact above a specified threshold. For example, such a function can reflect a chance that a classifier, constructed using a selected subset of attributes, does not misclassify considered objects. It is further expected that values of such functions do not increase for smaller subsets of attributes, as classifiers based on less information have limited possibilities to distinguish between objects supporting different decision classes. Using such an approach, one can obtain subsets of attributes that are moderately less accurate than standard decision reducts but could be preferred in real-world applications as they are usually more robust and contain less number of attributes [93].

In [140], it was discussed that ensembles of classifiers based on different approximate decision reducts can repeatedly misclassify the same data instances. This is because the above functions evaluate the subsets of attributes by means of an overall summary of the data - without focusing on the particular objects. To address this issue, one might think about combining a process of searching for ensembles of approximate decision reducts with popular machine learning techniques, such as boosting or bagging [28].

Alternatively, the notion of decision bireduct is proposed as a new extension of the concept of decision reduct having the focus on both the subset of attributes, which describes decisions, as well as the subset of objects, for which that description is valid. A decision bireduct is represented as a pair, where a subset of attributes can be evaluated by means of a subset of objects for which it assures good classification. A subset of objects provides far more explicit information about the corresponding subset of attributes and its abilities to construct a good classifier than any evaluation function. In particular, information about objects, for which decision values are validly described, allows to verify whether the classifiers designed using different selected subsets of attributes do not repeat mistakes on the same areas of the training data. Decision bireducts serve as a connection between several different views concerning the goals of knowledge discovery. Ensembles of information bireducts were studied in [140] as a counterpart to the notion of concepts studied in the formal concept analysis and item sets studied in the literature related to association rule mining. A concept in the formal concept analysis is defined as a non-extendable subset of objects that behaves in the same way with respect to a non-extendable subset of attributes [35], and consequently, concepts correspond to the most regular areas of the data. Similarly, the item sets aim at describing a maximum number of objects with the same values on a maximum number of attributes [87]. Contrary to that, information bireducts correspond to non-extendable subsets of objects that can be said to be different using irreducible subsets of attributes, thus, indicating the most diverse and, consequently, the most informative segments of the data. In the dissertation, we aim at drawing analogies between decision bireducts and other methods of representing dependencies in data. We compare decision bireducts with standard and approximate decision reducts. We show correspondences between those notions and provide interpretations, which work as a basis for algorithmic solutions to the problem of extracting the most interesting as well as feasible decision bireducts from a given data. We pay special attention to the interpretation of considered types of decision reducts and decision bireducts in terms of collections of decision rules which are able to neglect potentially noisy instances and, therefore, are more likely to remain robust when classifying new data.

In the dissertation, we focus on ensembles of classifiers, specifically exploring the cooperative capabilities of ensembles of decision bireducts in solving classification tasks. In general, there is a range of approaches based on ensembles of classifiers in the areas of knowledge discovery and data classification [28, 100]. Utilizing ensembles, we can count on the stability and robustness of the collective model. Furthermore, it is expected that each component of an ensemble can be simpler than a single, not ensemble-based decision model providing a similar level of accuracy. However, ensembles of models present inherent complexity, often making it difficult to establish effective cooperation among the ensembles' components. Moreover, a lot of computing power is needed to derive them from the data.

When it comes to methods for learning ensembles of classifiers, there are certain commonly accepted goals. Each single classifier is expected to make mistakes, but for each training case, a majority of models in the ensemble should be correct. Complementarity and diversification of components are another very desirable feature for ensembles of classifiers. In case of rough-set-inspired approaches to knowledge discovery, this refers to computation of diverse decision reducts. If we want to extend this idea in the context of diversification of objects which are classified correctly/wrongly by particular models, we can rely on the notion of ensembles of decision bireducts.

The rough set literature is a good reference point for considering formal optimization problems related to construction of decision models. Starting from fundamental results on NP-hardness of the problem of finding minimal decision reduct, a lot of attention is paid to develop mathematical and algorithmic methods for operating with the simplest yet sufficiently accurate classifiers [95, 96]. In the dissertation we want to extend these results with respect to ensembles. In particular, we propose how to define the optimization problem related to searching for the simplest possible ensembles of decision models that meet specific accuracy constraints. We are aware that there are numerous ways to express constraints related to various aspects of an ensemble, including both the constraints of the ensemble as a whole and those regarding its individual components. Similarly, there are multiple ways of understanding the simplicity of an ensemble. Nevertheless, we believe that the introduced formulation, which focuses on collections of decision bireducts that include the minimal amounts of attributes (i.e., the optimization goal) and at the same time appropriately cover all considered objects using the corresponding decision rules (i.e., the accuracy constraint), can be a good starting point for further investigations.

In contemporary machine learning applications, there is often a requirement to provide interpretable decision models [32]. One aspect of interpretability involves the utilization of attributes that are understandable by subject matter experts, and connecting those attributes in a comprehensible manner. Consequently, reporting the importance of attributes used in a model often becomes an essential aspect of interpretability. In some practical applications the ranking of attributes based on their importance may become sometimes even more useful than the associated decision models themselves. Therefore, by studying the classification capabilities of decision bireduct ensembles, we also strive for their interpretability by designing a procedure to assess the attributes used by the ensembles. To enhance the usefulness of such results even further, we also introduce an approach for evaluating the importance scores of attributes generated by any machine learning method.

To validate our methods, both related to classification and interpretability, we used an experimental approach on synthetic and benchmark data sets. Finally, we provide a comprehensive case study that illustrates the practical application of decision bireducts ensembles. The case study pertains to a decision problem encountered during the development of a solution for an HR company specializing in the recruitment of IT experts.

1.1 Plan of the Dissertation

The dissertation is divided into eight chapters with supplement materials included in the appendices. This introductory chapter provides an outline of the considered problem, presents the main contributions and aims to help the readers with navigation through the chapters. It also presents literature review of related works.

Chapter 2 introduces the general notions used in subsequent chapters of the dissertation and presents some basic relevant details about them. In Section 2.1, we present the standard notion of decision tables that will be used throughout the dissertation to represent tabular data. In Section 2.2, we define the fundamental concepts of classification task and classification model. Moreover, we briefly review a selected part of the state-of-the-art and classical models that will serve as reference points for the methods presented in the dissertation. In Section 2.3, we discuss the notion of performance of classifiers, and we recall some of commonly used metrics. Furthermore, we discuss different validation techniques for assessing classification models.

Chapter 3 aims to provide an introduction to the fundamental concepts of rough set theory. In Section 3.1, we introduce basic concepts and fundamental building block of rough sets. In Section 3.2, classical decision reducts with several variants are recalled. We present the correspondence between decision reducts and decision rules and show how the task of searching decision reducts can be expressed in terms of Boolean formulae and prime implicants. In Section 3.3, we introduce approximate decision reducts.

Chapter 4 presents the notion of decision bireducts and their variants, along with ensembles of decision bireducts and their properties. In Section 4.1, we introduce decision bireducts, discuss their main properties, and outline their utilization as rule-based classifiers. In Section 4.2, we introduce a variant of decision bireducts called γ-decision bireducts, which are associated with the concept of the positive region in rough set theory. We discuss the properties of γ-decision bireducts and present intuitive insights into their relationship with standard decision bireducts. In Section 4.3, we explore the classical rough set approach that involves defining a propositional formula, referred to as the "discernibility function". This formula represents the necessary conditions that must be satisfied for all pairs of discernible objects. Consequently, the collection of decision reducts corresponds to the set of all prime implicants of the discernibility function. In the referenced section, we adapt this general technique to develop suitable Boolean formulae that capture the essential constraints associated with decision bireducts and γ-decision bireducts. Furthermore, we demonstrate how various attribute reduction heuristics can be directly applied to the task of searching for decision bireducts, following an appropriate transformation of the input data. In Section 4.4, we discuss optimization criteria for decision bireducts. Additionally, we introduce the notions of decision ε-bireducts and γ-decision ε-bireducts, which impose constraints on the proportion of covered objects. Furthermore, we demonstrate the NP-hardness of the optimization problem of finding a minimal decision ε-bireduct. In Section 4.5, we delve into the concept of ensembles of decision bireducts, examining their properties, and providing illustrative examples. We also introduce the notion of a correct ensemble, which requires that each training object is correctly recognized by more than half of the classifiers in the ensemble. In Section 4.6, the concept of "simplicity" for ensembles of decision bireducts is discussed. Furthermore, the NP-hardness of the optimization problem is presented when it comes to finding the simplest correct ensemble of decision bireducts.

Chapter 5 shows a practical look at algorithms and heuristics for computing decision bireducts. In Sections 5.1, 5.2 and 5.3, several algorithms for computing decision bireducts and γ-decision bireducts are presented. In Section 5.4, the first experiments related to ensembles of decision bireducts on benchmark data sets from the UCI repository are presented. In Section 5.5, we introduce a special case of the notion of decision bireducts in the domain of data streams and investigate its utilization, focusing on scenarios where the complete data set is not available during the computation process. Instead, we consider situations where events are processed incrementally, with each event arriving one at a time. In Section 5.6, we present the ideas and assumptions of a software library that is released as a result of our research. In Section 5.7, we discuss a general solution implemented within our library that enables a structural way of defining algorithms, e.g., those related to computing decision bireducts.

Chapter 6 comprehensively presents a case study demonstrating the application of decision bireducts ensembles to a decision problem encountered while developing a solution for an HR company specializing in the recruitment of IT professionals.

Chapter 7 focuses on presenting the ability to provide feature importance scores by ensembles of decision reducts and decision bireducts. In Section 7.1, we introduce methods for generating feature importance scores using ensembles of approximate decision reducts. These importance scores are then utilized in an experimental assessment, where they serve as input for feature selection and are evaluated on both synthetic and microarray data sets. In Section 7.2, we introduce an approach to evaluate attribute scores produced by any machine learning method and utilize it to compare the attribute importance scores provided by ensembles of decision bireducts against the importance scorings provided by XGBoost and correlation-based reference model.

Chapter 8 concludes the dissertation. Section 8.1 presents a summary of the overall content of the dissertation and Section 8.2 lists a few directions for future research.

Supplementary materials consist of two types of resources. Appendix A includes comparisons and detailed feature importance profiling views for the data sets and algorithms used in the dissertation and Appendix B provides some examples of using the developed software library, given in a form of Jupyter Notebooks.

1.2 Main Contributions

Contributions made with regard to the notion of decision bireducts:

- Definition and Formalization: We defined decision bireduct as well as other variants, e.g., decision ε-bireducts; or γ-decision bireducts related to the concept of positive region in the rough set theory.

- Characterization and Properties: In the classical rough set approach, a propositional formula called the “discernibility function” [115, 95] is formulated to represent the necessary conditions that need to be satisfied for all pairs of discernible objects. As a result, the collection of decision reducts corresponds to the set of all prime implicants of the discernibility function. We adapted this general approach to develop suitable Boolean formulae that capture the essential constraints related to decision bireducts and γ-decision bireducts.

- Evaluation and Comparison: We explored the connections between decision bireducts and classical constructs, namely decision reducts and approximate decision reducts.

- Computational Complexity: We demonstrated the NP-hardness of the optimization problem of finding a minimal decision ε-bireduct.

- Algorithmic Approaches: We introduced a number of algorithms and heuristics that are designed to efficiently search for decision bireducts and their variants.

- Generalizations and Extensions: We introduced a special case of the notion of decision bireducts in the domain of data streams and investigated the utilization of it, focusing on such situations where the complete data set is not available during the process of computation. Instead, we considered scenarios where events are processed incrementally, arriving one at a time.

Contributions made with regard to ensembles of decision bireducts:

- Definition and Formalization: We explored the concept of ensembles of decision bireducts and examined their properties and characteristics.

- Computational Complexity: We demonstrated the NP-hardness of the optimization problem of finding the simplest correct ensemble of decision bireducts. The concept of “simplicity” was defined analogous to the approach presented in the study on generalized decision reducts [137], where simplicity was determined based on the maximum cardinality among all subsets of attributes involved.

- Evaluation and Comparison: We performed experiments on synthetic and benchmark data sets.

- Real-world Applications: We comprehensively present a case study demonstrating the application of ensembles of decision bireducts to a decision problem encountered while developing a solution for an HR company specializing in the recruitment of IT professionals.

- Interpretability: To enhance the interpretability of the model, we presented a method for assessing the attributes employed in decision bireduct ensembles.

General contributions:

- Evaluation and Comparison: We proposed a general procedure for evaluating attribute importance methods.

- Implementation: A Python software library *scikit-rough* (`https://github.com/sebov/scikit-rough`), hosted on GitHub platform, is released as a result of our research.

1.3 Related Work

The notion of decision bireducts, which is the main subject of this dissertation, is a relatively new concept; however, we can already observe that it is gaining interest in various fields of research. In [47] the authors applied bireducts in intuitionistic fuzzy framework that can be used for simultaneous reduction of instances and features and experimentally evaluated the approach in the challenging domain of cancer treatment. In [89, 27, 90] the notion of fuzzy-rough bireducts was introduced for simultaneous reduction of data size and dimensionality. The authors examined the usability of bireducts in data preprocessing stages and their performance as compact and robust classifiers. They developed a heuristic strategy for the identification of fuzzy-rough bireducts basing on a music-inspired global optimization algorithm, called harmony search, and they tested how bireducts perform on noisy data. In [7] the use of bireducts was investigated in the context of data size reduction and inconsistencies removal. In [24] the authors introduced decision reducts, decision bireducts and γ-decision bireducts in covering approximation spaces. In [75] a concept of min-max attribute-object bireducts was introduced and discussed. In [6, 8] the reducts and bireducts were studied in the classical environment of rough set theory considering tolerance/similarity relations.

When discussing feature selection, it is important to highlight the close relationship between this domain and rough set theory. Discovering redundancies and dependencies among attributes is one of the fundamental and challenging problems in the philosophy of rough sets [92]. Since the early days of rough sets, numerous researchers have investigated this problem [80, 135, 122, 144]. In a standard rough set approach, subsets of attributes are chosen based on the concept of reducts. In [91], it is demonstrated that a decision reduct can consist only of strongly and weakly relevant attributes (understood as in [62]) if the available data sufficiently cover the universe. The concept of approximate decision reducts, introduced in [131], is further expanded on this idea by considering reducts that may not necessarily discern all objects but exhibit reduced sensitivity to random data disturbances.

On the other hand, when it comes to the more general aspect of feature selection methods, they can be divided into two main categories, i.e., wrapper and filter methods [60, 62, 40, 13]. The first approach involves ranking subsets of features based on the performance of a predictive model built using those features. A higher score indicates a better quality for the subset of attributes. Because the number of all possible subsets of attributes is exponentially large, various heuristics are employed to explore the attribute space [123, 113, 46, 77]. While the wrapper approach generally produces superior results compared to the filter approach, its computational complexity makes it difficult to apply for extremely high dimensional data. Filter methods employ predefined scoring functions to generate rankings of individual features or feature subsets. The ranking algorithms within this category can be split into two groups: univariate and multivariate. The univariate rankers assess the importance of individual attributes, without considering the dependencies between features. Within this group, examples include simple correlation-based rankers [42], statistical tests [70], or rankers based on the mutual information measure [98]. The multivariate attribute rankers aim to assess the relevance of features in a broader context, taking into account their dependencies. They accomplish this by testing the usefulness of features in groups, such as the relief algorithm [61], or by explicitly measuring the relatedness of attribute pairs and applying the minimum-redundancy-maximum-relevance framework [98, 29]. Another noteworthy example is Breiman's relevance measure, which quantifies the average increase in classification error resulting from the randomization of attributes used in constructing trees by the Random Forest algorithm [14, 30].

A crucial aspect of feature selection using the filter approach is to determine the optimal number of attributes to choose. The commonly employed method involves selecting a predetermined number of the top-ranked attributes. However, in practical applications, even domain experts usually cannot accurately estimate the appropriate number of attributes to be selected. To address this issue, various techniques have been developed. One such method involves employing permutation tests [23] to estimate the distribution of scores assigned to irrelevant attributes by the ranking algorithm. This estimation enables to compute the probability that an attribute with a given score is indeed irrelevant. By utilizing this information, it becomes possible to determine a suitable value of the number of attributes based on a predefined risk threshold for selecting irrelevant features.

One of the most important directions in machine learning refers nowadays to the paradigms of explainability [2] and interpretability [38]. When it comes to explainability, the goal is to provide some insights on what can be expected from a decision model, even if the model itself may be considered a black box. On the other hand, interpretable models are those which are self-explanatory such that they do not need additional tools to understand how they work – this may include rule-based models [17] or the models relying on simple statistical methods [139]. Interpretability of models is emphasized in many practical applications, such as medicine and bioinformatics [36] or human resources [26]. There are many approaches to explainability, either providing method for particular category of models [73] or model-agnostic [109]. Moreover, the research often is extended toward the ensembles of decision models [63, 43]. When there is a requirement for decision transparency, even highly performing models like XGBoost [19] may need to be transformed into simpler and more transparent models that aim to approximate the predictive performance of the trained model [111].

Quality assessment of attributes is a must-have for interpretable decision systems. The methods for assessing the quality and importance of attributes can help in providing more insight into trained models and a better understanding of decision domains under consideration. With regard to this, multiple results based on various heuristics and quality measures are available [103, 126]. The ensemble approaches that combine the outputs from multiple attribute importance rankings are also adopted for the task of feature selection [112]. To verify the stability of such algorithms in the presence of noisy attributes, one of several rank correlation statistics may be used [107]. Typically, when comparing attribute importance ranking methods, wrapper approaches are commonly employed. These approaches involve assessing the overall quality of the method by evaluating the accuracy of models trained on subsets of top-ranked attributes [31, 108].

For the HR-related case study we present in the dissertation, the explainability of models is crucial for the adoption machine learning solutions. The explainability not only makes a given machine learning methodology more appealing to the recruiters but also may help in detecting and avoiding the prejudicial bias of the job candidate scoring models [39]. Historically, recruitment support systems originated from the area of recommender systems, where the explainability also plays an important role. Examples of recommender systems, that were designed for the purpose of human resource matching and online recruitment, can be found, e.g., in [79, 114, 127]. One more thing worth mentioning is that the entire HR industry has undergone significant changes due to COVID-19 [18]. In particular, many businesses have introduced or expanded the possibility of remote work [81]. This trend has also impacted recruitment companies, especially those that perform recruitment tasks for other companies. This is also the case for the company presented in the case study within the dissertation, where, prior to COVID-19, the vast majority of orders were related to relocating employees to another country where the job is offered. Therefore, the original goal discussed was to assess which candidates were most likely to move to another country for a new job. However, in order to adapt to the post-pandemic reality, the goals and assumptions may need to be revisited and stated differently; for example, in the new context the concern of assessment could be more general, such as whether potential candidates would be interested in changing their jobs.

The next important topic is feature engineering [12]. In the domain of machine learning, some researchers claim that a separate phase of manual feature engineering is not needed because modern approaches are able to learn efficient models directly from the raw data. However, when it comes to real-world applications – often referred to as the realm of data science [88] – the construction of attributes, using the involvement and knowledge of domain experts, may improve not only the interpretability of models but also increase their accuracy. When it comes to the construction of attributes in HR applications, it is worth mentioning numerous studies in the context of job recommendation processes [69, 129]. More generally, feature engineering corresponds to the problem of searching for optimal data representations, which may be especially useful for the data coming from different sources and having different modalities [50]. Learning meaningful representations is also important if the data items correspond to complex objects. In such a case, representations based on complex attributes and hierarchical modeling [120] can be used.

1.4 Acknowledgments

I would like to thank my supervisor Prof. Dominik Ślęzak for guidance, infinite patience, and support in writing this dissertation. I am immensely grateful for all the ideas, discussions, and visionary insights regarding relationships and synergies between different topics.

I am deeply grateful to Prof. Andrzej Skowron for playing a significant role in shaping my academic journey. From my student days to my current involvement in scientific research, his guidance and support have been invaluable.

I would like to acknowledge: Andrzej Janusz for his advices and consultations in conducting the experiments and many discussions held on the topics of decision reducts and bireducts – regarding ideas as well as algorithmic nuances; Soma Dutta and Błażej Stawicki for editorial consultations and linguistic corrections.

I would like to thank all my co-authors – Paweł Betliński, Agnieszka Chądzyńska-Krasowska, Krzysztof Ciebiera, Michał Drewniak, Paweł Gora, Marek Grzegorowski, Kamil Herba, Andrzej Janusz, Marcin Kowalski, Michał Kozielski, Adam Krasuski, Hung Son Nguyen, Sinh Hoa Nguyen, Tuan Trung Nguyen, Przemysław Wiktor Pardel, Mariusz Rosiak, Marek Sikora, Krzysztof Stencel, Marcin Szczuka, Dominik Ślęzak, Sebastian Widz, Mateusz Wnuk, Piotr Wojnarowski, Marcin Wojnarski, Piotr Wojtas, Łukasz Wróbel, Jakub Wróblewski – whose ideas and commitment have been a great motivation and inspiration to me.

My deepest gratitude goes to my family and friends. I want to thank my Brother for his sense of humor and for always being someone I can rely on. Last but certainly not least, I wish to thank my Mom and late Dad for their love and support throughout my entire life. None of the things that I achieved would be possible without you. Thank You!

The research was supported by the grants N N516 368334, N N516 077837, 2011/01/B/ST6/03867, and 2012/05/B/ST6/03215 from the Ministry of Science and Higher Education of the Republic of Poland; by the National Centre for Research and Development under the grants SP/I/1/77065/10 and PBS2/B9/20/2013; and by EU Smart Growth Operational Programme under the Innovation Voucher project POIR.02.03.02-14-0009/15.

Chapter 2

Preliminaries

This chapter serves the purpose of introducing the fundamental notions and concepts that will be referred to throughout the dissertation. We present the standard tabular data representation and revisit key concepts associated with classification tasks, classification models, and evaluation metrics used for their assessment. We also provide a brief overview of some commonly used algorithms and classification methods, which will be later utilized in subsequent chapters for comparative analysis of our proposed approaches.

2.1 Data Representation

In the dissertation, we use the standard representation of tabular data in a form of decision tables and the basic concepts from the rough set theory [96, 95].

Definition 1 (decision table)**.** *A decision table is a pair $\mathbb{A} = (U, A \cup \{d\})$ of non-empty sets U and $A \cup \{d\}$, where U is a universe of objects, and $A \cup \{d\}$ is a set consisting of attributes such that every $a \in A \cup \{d\}$ is a function $a : U \to V_a$, where V_a denotes a's codomain and is called the value set of a. The distinguished attribute d, such that $d \notin A$, is called a decision attribute and the elements of A are called conditional attributes.*

In practice, we usually deal with finite sets U of objects and $A \cup \{d\}$ of attributes. For the decision attribute d, the values $v_d \in V_d$ correspond to decision classes that we want to describe using the values of attributes in A, in order to utilize such descriptions in the process of classification of objects outside U. To shorten the notation we will be sometimes referring to the elements of U using their ordinal numbers $i = 1, \ldots, |U|$, where $|U|$ denotes the cardinality of U. The shorthand notation will also allow for expressing the ranges of objects, e.g., $[\![1, 4..6, 8, 9, 11..13]\!]$ will be considered as equivalent to the subset of objects $\{u_1, u_4, u_5, u_6, u_8, u_9, u_{11}, u_{12}, u_{13}\}$.

2.2 Classification Task

Definition 2 (classification task)**.** *In machine learning, a classification task is a type of problem where the goal is to assign a decision value (or label) to an input object based on its attributes. If there are two mutually exclusive labels from which a choice must be made then the task is called a binary classification task. Otherwise, if there are three or more labels available, we say it is a multi-class classification task. If the goal is to assign zero or more labels for each input object then we say that it is a multi-label classification task. Moreover, if we are interested in assigning a probability distribution over a set of decision values, rather than getting the exact labels, then we say that it is a probabilistic classification task.*

Definition 3 (classification model)**.** *A classification model is a type of machine learning model that predicts a decision value (or decisions values) of a given input object. In other words, it is an implementation (or realization) of a given classification task. It can be represented as a kind of a function that maps input data objects to predicted decisions $\hat{d}_{\mathcal{M}} : \mathcal{U} \to \mathcal{D}$, where $\mathcal{U}$ is a generally understood universe of objects and $\mathcal{D}$ is a set of available decision values.*

In general the input objects can take various forms, such as texts, images, video clips, numerical data, or even can have multimodal representations. However, the focus of the dissertation will be on tabular data. Therefore, for the remainder of the work, we assume that $\mathcal{U}$ is a universe of objects U represented by means of attributes A, as described in Definition 1. Whereas, considering the decisions $\mathcal{D}$ we assume that, depending on the context, it is either equal to V_d or it represents probability estimates of an input object belonging to each of possible decision classes. Many machine learning methods provide both options, i.e., in addition to returning the most likely decision class for a given object, they can deliver also a probability distribution over all possible decision values.

Throughout the dissertation we will be looking for analogies and comparing the experimental results of our methods to other solutions, including classical and state-of-the-art techniques. Let us recall the existing algorithms that will be used later in the dissertation.

Logistic regression [22] is a classical, relatively simple and interpretable model commonly used in machine learning. It is an algorithm used to model the probability of a binary decision based on a set of input attributes using a sigmoid function. k-NN [21] is another classical algorithm. It is a method where the classification procedure is based on plurality voting of the unseen object's neighbors. In other words, the algorithm chooses the most commonly represented decision value among the neighbors of the given object. Next, we use the term "decision trees" to refer to hierarchical classification models that make predictions by applying a set of simple decision rules, arranged in the form of a tree structure. There are many algorithms available that can induce decision trees from data, e.g., ID3 [104], C4.5 [105], and CART (Classification and Regression Trees) [15]. It is also important to mention Random Forest [14] – a classical algorithm that uses a classifier ensemble. Each of its component is a decision tree built using a random subset of objects and attributes. To make a prediction on it combines the predictions from all individual trees.

XGBoost[1], which stands for *Extreme Gradient Boosting* [19], is a notable algorithm representing the state-of-the-art in machine learning. It is an ensemble machine learning algorithm implemented within the gradient boosting framework. XGBoost is a very popular method that is widely used by the data science community. The algorithm is known for its high performance, scalability and flexibility. Like other algorithms in its category, it iteratively builds weak models and combines them to form a strong final model. It can use several boosting strategies and different weak learners. The gradient boosting is a sequential process in which every next weak learner is created depending on the performance of the earlier ones. Such learners are supposed to compensate for the errors made by the ensemble. XGBoost starts with a single leaf that represents an initial "guess" for the whole data set. Then, each consecutive learner is firstly trained on objects corresponding to the intermediate errors of the ensemble created up to the current point of time, and secondly, it is combined with that ensemble in a way that minimizes the loss function.

2.3 Evaluation Metrics for Classification Models

In the dissertation we will be comparing the created classification models and for this reason we need to define evaluation metrics that are commonly used to judge the performance of the model after the training. The following definition, however quite informal, expresses the idea of very general concept of evaluation metric family of functions:

[1] `https://xgboost.readthedocs.io/`

Definition 4 (evaluation metric)**.** *An evaluation metric eval* $: \overbrace{\square}^{\text{model}} \times \overbrace{\square}^{\text{data table}} \to \mathbb{R}$ *is a function that quantifies the performance of a machine learning model* $\mathcal{M}$ *on a given data table* $\mathbb{A} = (U, A \cup \{d\})$*. The application of the evaluation metric is constrained to the input arguments that are compatible with each other, i.e., the model* $\mathcal{M}$ *must be able to recognize representation of objects in* $\mathbb{A}$ *and make predictions on them.*

Definition 5 (model outcomes)**.** *Let a binary classification task be defined on* $\mathbb{A} = (U, A \cup \{d\})$ *and let a classification model* $\mathcal{M}$ *be given. We define the subset of objects in* $\mathbb{A}$ *with a distinguished decision value* $v_{d_{relevant}} \in V_d$ *in* $\mathbb{A}$ *to be "relevant objects" or "relevant elements". The whole set of objects* U *is then partitioned into four disjoint subsets based on both the actual decision values in* $\mathbb{A}$ *and decisions predicted by the model* $\mathcal{M}$*. As "true positives" (*$TP_{\mathcal{M},\mathbb{A}}$*) and "false negatives" (*$FN_{\mathcal{M},\mathbb{A}}$*) we consider objects among the relevant elements for which the model predicted correct and incorrect decision value, respectively. Analogously, "true negatives" (*$TN_{\mathcal{M},\mathbb{A}}$*) and "false positives" (*$FP_{\mathcal{M},\mathbb{A}}$*) mean the non-relevant objects correctly and incorrectly classified by the model, respectively. The above subsets can be formally defined as follows:*

$$TP_{\mathcal{M},\mathbb{A}} = \{u \in U : d(u) = v_{d_{relevant}} \wedge d(u) = \hat{d}_{\mathcal{M}}(u)\} \tag{2.1}$$

$$FN_{\mathcal{M},\mathbb{A}} = \{u \in U : d(u) = v_{d_{relevant}} \wedge d(u) \neq \hat{d}_{\mathcal{M}}(u)\} \tag{2.2}$$

$$TN_{\mathcal{M},\mathbb{A}} = \{u \in U : d(u) \neq v_{d_{relevant}} \wedge d(u) = \hat{d}_{\mathcal{M}}(u)\} \tag{2.3}$$

$$FP_{\mathcal{M},\mathbb{A}} = \{u \in U : d(u) \neq v_{d_{relevant}} \wedge d(u) \neq \hat{d}_{\mathcal{M}}(u)\} \tag{2.4}$$

If $\mathcal{M}$ *and* $\mathbb{A}$ *are known from the context, we will be using just* TP, FN, TN, FP *for short.*

Definition 6 (precision)**.** *Let a binary classification task be defined on* $\mathbb{A} = (U, A \cup \{d\})$ *and let a classification model* $\mathcal{M}$ *be given. We say that "precision" is an evaluation metric expressing the fraction of objects classified by the model* $\mathcal{M}$ *as relevant that are actually relevant in* $\mathbb{A}$*. It is defined as follows:*

$$precision(\mathcal{M}, \mathbb{A}) = \begin{cases} \frac{|TP|}{|TP|+|FP|} & \text{if } |TP| + |FP| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

Definition 7 (recall)**.** *Let a binary classification task be defined on* $\mathbb{A} = (U, A \cup \{d\})$ *and let a classification model* $\mathcal{M}$ *be given. We say that "recall" (or "sensitivity", or "true positive rate" –* TPR*) is an evaluation metric expressing the fraction of relevant objects correctly classified by the model* $\mathcal{M}$*. It is defined as follows:*

$$recall(\mathcal{M}, \mathbb{A}) = sensitivity(\mathcal{M}, \mathbb{A}) = TPR(\mathcal{M}, \mathbb{A}) = \begin{cases} \frac{|TP|}{|TP|+|FN|} & \text{if } |TP| + |FN| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

Definition 8 (F1 score)**.** *Let a binary classification task be defined on* $\mathbb{A} = (U, A \cup \{d\})$ *and let a classification model* $\mathcal{M}$ *be given. We say that "F1-score" is the harmonic mean of precision and recall values. It is defined as follows:*

$$\text{F1-score}(\mathcal{M}, \mathbb{A}) = \begin{cases} 2 \cdot \frac{precision(\mathcal{M},\mathbb{A}) \cdot recall(\mathcal{M},\mathbb{A})}{precision(\mathcal{M},\mathbb{A}) + recall(\mathcal{M},\mathbb{A})} & \text{if } precision(\mathcal{M}, \mathbb{A}) + recall(\mathcal{M}, \mathbb{A}) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

Definition 9 (specificity)**.** *Let a binary classification task be defined on* $\mathbb{A} = (U, A \cup \{d\})$ *and let a classification model* $\mathcal{M}$ *be given. We say that "specificity" (or "true negative rate" –* $TNR(\mathcal{M}, \mathbb{A})$*) is an evaluation metric expressing the fraction of non-relevant objects correctly classified by the model*

$\mathcal{M}$. It is defined as follows:

$$specificity(\mathcal{M},\mathbb{A}) = TNR(\mathcal{M},\mathbb{A}) = \begin{cases} \frac{|TN|}{|TN|+|FP|} & \text{if } |TN|+|FP|>0 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

It may be seen as a recall in case we switch the distinguished decision value indicating the relevant objects.

Definition 10 (class-specific recall)**.** *Let a classification task be defined on $\mathbb{A} = (U, A \cup \{d\})$ and let a classification model $\mathcal{M}$ be given. We say that "class-specific recall" ($recall_{v_d}(\mathcal{M},\mathbb{A})$) is an evaluation metric expressing the fraction of objects having the given decision value v_d that are correctly classified by the model $\mathcal{M}$. It is defined as follows:*

$$recall_{v_d}(\mathcal{M},\mathbb{A}) = \begin{cases} \frac{|\{u\in U: d(u)=v_d \wedge d(u)=\hat{d}_{\mathcal{M}}(u)\}|}{|\{u\in U: d(u)=v_d\}|} & \text{if } |\{u\in U: d(u)=v_d\}|>0 \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

It may be seen as a metric generalizing Definition 7 and Definition 9 for the case of multi-class classification tasks.

Definition 11 (accuracy)**.** *Let a classification task be defined on $\mathbb{A} = (U, A \cup \{d\})$ and let a classification model $\mathcal{M}$ be given. We say that "accuracy" is an evaluation metric expressing the fraction of objects classified correctly by the model $\mathcal{M}$. For a binary classification task it can be expressed using the notions defined in Definition 5:*

$$ACC(\mathcal{M},\mathbb{A}) = \frac{|TP|+|TN|}{|TP|+|FN|+|TN|+|FP|} \tag{2.10}$$

Or in a more general form usable also in the case of multi-class classification tasks:

$$ACC(\mathcal{M},\mathbb{A}) = \frac{|\{u\in U: d(u)=\hat{d}_{\mathcal{M}}(u)\}|}{|U|} \tag{2.11}$$

In an imbalanced data sets the number of objects in each decision class is not equal. In such case the accuracy measure's results may be misleading. For example in case of highly imbalanced data sets the accuracy result achieved by a model always predicting the majority class may be very high; however the model may not be useful in practical applications where the minority classes are important.

Definition 12 (balanced accuracy)**.** *Let a classification task be defined on $\mathbb{A} = (U, A \cup \{d\})$ and let a classification model $\mathcal{M}$ be given. We say that "balanced accuracy" is an evaluation metric expressing the average of class-specific recall values obtained on each decision value. For a binary classification task it is can be expressed using the notions defined in Definition 7 and Definition 9:*

$$BAC(\mathcal{M},\mathbb{A}) = \frac{TPR(\mathcal{M},\mathbb{A})+TNR(\mathcal{M},\mathbb{A})}{2} \tag{2.12}$$

Or in a more general form usable also in the case of multi-class classification tasks:

$$BAC(\mathcal{M},\mathbb{A}) = \frac{\sum_{v_d\in V_d} recall_{v_d}(\mathcal{M},\mathbb{A})}{|V_d|} \tag{2.13}$$

In the case of a binary classification task (though it can also be generalized to the multi-class case) and probabilistic classification models, we receive the response from the model in the form of a probability distribution for assigning a given object to the appropriate decision class. However, if we need to transform the returned values to actual labels, a decision must be made regarding the threshold value which will allow us to state whether a given object belongs to one decision class or the other. Furthermore, the estimated performance of the classifier may differ with different threshold values. To address this issue, the receiver operating characteristic (ROC) curve [99, 78, 33], which is a visualization technique, is commonly used to present the performance measurements at various threshold settings. The AUC (area under the ROC curve) is a scalar value that aggregates the classifier performance estimations over all possible threshold values, thus expressing how well the given model is capable of separating the decision classes.

The primary objective of the classification task is to develop a model that can identify relationships in the data and can generalize the discovered patterns to unseen objects. To achieve this, the most basic approach is to split the available data into two separate sets – a training subset for inducing (or training) a classifier, and a testing subset to asses the model's performance on previously unseen objects. In practical applications, it may not always be possible to have a designated large test set that can properly estimate the performance of classifiers. To address this issue, the cross-validation [48, 121] technique is commonly used in machine learning to evaluate the performance of a model splitting the available data into multiple parts. One common variant of this technique is k-fold cross-validation, where the data is divided into k subset of roughly equal size. The model is trained using $k-1$ folds as a training set, and the remaining one as a test set. Such a procedure is repeated k times, for each fold being used once as a test set. Ultimately, the results from all runs are averaged to obtain the final performance of the model. Various variants of cross-validation are available. *Leave-one-out* is a special case of k-fold cross-validation, where k is equal to the number of objects in the data set and the process is repeated for each sample from the data set by training on all but one of the samples and tested on the one left out. Another example is *leave-one-group-out*, where the data set is split based on arbitrary information defining the grouping of objects. The process is then repeated for each group using training subsets consisting of all samples except those belonging to the given group, and the model is tested on that specific group. We will be using such an approach in Chapter 6.

Chapter 3

Foundations of Decision Reducts

The objective of this chapter is to provide an introduction to the fundamental concepts of rough set theory, which will serve as the basis for the central study of the dissertation. We will recall both standard and approximate decision reducts, along with their properties, enabling us to establish analogies to these concepts in future chapters.

3.1 Basic Rough Set Concepts

The rough set theory is an extension of the classical set theory, suitable for describing concepts in presence of incomplete information [91]. The rough sets handle imperfect knowledge about objects and concepts using notions of lower and upper approximations. A basic information unit for the rough sets is an indiscernibility class of an object u, which is a set of those $u' \in U$ that cannot be distinguished from u using information available in a decision system $\mathbb{A}$. Some indiscernibility classes of objects from the same decision class can be aggregated to form information granules [116]. For numeric data, this aggregation can be done using a greedy discretization heuristic that is based on a discernibility measure [84].

Definition 13 (indiscernibility relation)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. An attribute subset $B \subseteq A \cup \{d\}$ determines a binary relation $IND(B)$ on U:*

$$u \; IND(B) \; u' \quad \iff \quad \underset{a \in B}{\forall} a(u) = a(u') \tag{3.1}$$

In such a case, we say that u and u' are indiscernible by attributes of B.

Proposition 1. *Let $\mathbb{A} = (U, A \cup \{d\})$ and an attribute subset $B \subseteq A \cup \{d\}$ be given. The indiscernibility relation $IND(B)$ is an equivalence relation.*

Proof. The observation that $IND(B)$ satisfies all the properties of the equivalence relation (that the relation is reflexive, symmetric, and transitive) follows straightforwardly from Definition 13, that is, from the equality of the values on all attributes from B. □

The indiscernibility relation $IND(B)$ as an equivalence relation partitions the universe of objects U into a quotient set, i.e., a set of all equivalence classes determined by $IND(B)$. The quotient set is denoted by $U/IND(B)$ or U/B for simplicity. An element of U/B that contains a given object $u \in U$ is denoted by $[u]_{IND(B)}$ (or $[u]_B$ for short).

In the dissertation, we will refer to the elements of a quotient set determined be the decision attribute d as $X^{\langle 1 \rangle}, \ldots, X^{\langle |V_d| \rangle} \in U/\{d\}$, where $|V_d|$ denotes the cardinality of the decision attribute's codomain and where the mapping $\{1, \ldots, |V_d|\} \to V_d$ is arbitrarily chosen without loss of generality. We will use a value based notation $X^{\langle d = v_d \rangle}$ to denote a decision class containing objects with the given decision value $v_d \in V_d$. In some places we will also use a simplified notation $X^{\langle d(u) \rangle}$ in order to denote a decision class containing a given object $u \in U$, which is equivalent to $X^{\langle d=d(u) \rangle}$.

Definition 14 (discernibility relation)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. An attribute subset $B \subseteq A \cup \{d\}$ determines a binary relation $DIS(B)$ on U defined as the complement of the relation $IND(B)$, i.e.,*

$$u \; DIS(B) \; u' \iff \neg(u \; IND(B) \; u') \iff \underset{a \in B}{\exists} a(u) \neq a(u') \tag{3.2}$$

In such a case, we say that an attribute subset B discerns objects $u, u' \in U$ (or that $u, u' \in U$ are discerned by B).

Definition 15 (approximations)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ and an attribute subset $B \subseteq A$ be given. For a subset of objects $X \subseteq U$ we define lower and upper approximations of X induced by B as follows:*

$$\underline{X}_B = \bigcup \{[u]_B \in U/B : [u]_B \subseteq X\} \qquad \text{(lower approximation)} \tag{3.3}$$

$$\overline{X}_B = \bigcup \{[u]_B \in U/B : [u]_B \cap X \neq \emptyset\} \qquad \text{(upper approximation)} \tag{3.4}$$

Table 3.1: Decision table $\mathbb{A} = (U, A \cup \{d\})$ with 14 objects in U, four attributes in A, and decision attribute "Play" with two decision values "yes" and "no".

	Outlook	***Temperature***	***Humidity***	***Wind***	***Play***
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

Definition 16 (consistent decision table)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. We say that $\mathbb{A}$ is consistent, if and only if $IND(A) \subseteq IND(\{d\})$. We say that $\mathbb{A}$ is inconsistent, if and only if it is not consistent.*

The condition $IND(A) \subseteq IND(\{d\})$ stated in the above definition can be expressed equivalently as one of the following:

- The attribute subset A discerns all objects $u_i, u_j \in U$ such that $d(u_i) \neq d(u_j)$.
- All objects u_i, u_j that are indiscernible by attributes from A have the same value of the decision attribute $d(u_i) = d(u_j)$.

Let us also recall the widely used notion of decision rules. A typical decision rule consists of two parts – a predecessor and a successor that are connected by the sign "⇒". They are often referred to as "if-then" rules. Following [96] let us introduce the notion formally.

Definition 17 (formulae)**.** *Let decision table $\mathbb{A} = (U, A \cup \{d\})$ be given and let $V = \bigcup_{a \in A \cup \{d\}} V_a$. Atomic formulae over $B \subseteq A \cup \{d\}$ and V are expressions of the form $a = v$ and called descriptors (or selectors), where $a \in B$ and $v \in V_a$. The set $\mathscr{F}(B, V)$ of formulae over B and V is the least set containing all atomic formulae over B and V and closed with respect to the propositional operators $\wedge$ (conjunction), $\vee$ (disjunction), $\neg$ (negation). By $\|\varphi\|_{\mathbb{A}}$ we denote the meaning of $\varphi \in \mathscr{F}(B, V)$ in the decision table $\mathbb{A}$– we say it is the set of all objects in U with the property φ. The definition of $\|\cdot\|_{\mathbb{A}}$ is given as follows:*

$$\|a = v\|_{\mathbb{A}} = \{u \in U : a = v\} \tag{3.5}$$

$$\|\varphi \wedge \varphi'\|_{\mathbb{A}} = \|\varphi\|_{\mathbb{A}} \cap \|\varphi'\|_{\mathbb{A}} \tag{3.6}$$

$$\|\varphi \vee \varphi'\|_{\mathbb{A}} = \|\varphi\|_{\mathbb{A}} \cup \|\varphi'\|_{\mathbb{A}} \tag{3.7}$$

$$\|\neg\varphi\|_{\mathbb{A}} = U \setminus \|\varphi\|_{\mathbb{A}} \tag{3.8}$$

$$\tag{3.9}$$

The formulae from $\mathscr{F}(A, V)$, $\mathscr{F}(\{d\}, V)$ are called condition formulae of $\mathbb{A}$ and decision formulae of $\mathbb{A}$, respectively.

Definition 18 (decision rule)**.** *Let decision table $\mathbb{A} = (U, A \cup \{d\})$ be given and let $V = \bigcup_{a \in A \cup \{d\}} V_a$. A decision rule for $\mathbb{A}$ is any expression of the form $\varphi \Rightarrow \psi$, where $\varphi \in \mathscr{F}(A, V)$ and $\psi \in \mathscr{F}(\{d\}, V)$. Formulae φ and ψ are referred to as the predecessor and the successor of the decision rule, respectively. A decision rule $\varphi \Rightarrow \psi$ is true in $\mathbb{A}$ if and only if $\|\varphi\|_{\mathbb{A}} \subseteq \|\psi\|_{\mathbb{A}}$.*

For a given $\mathbb{A} = (U, A \cup \{d\})$ an exemplary decision rule looks as follows:

$$\underbrace{(a_1 = v_{a_1}) \wedge \ldots \wedge (a_m = v_{a_m})}_{\text{predecessor}} \Rightarrow \underbrace{(d = v_d)}_{\text{successor}} \tag{3.10}$$

where $a_1, \ldots, a_m \in A$ are some of conditional attributes and each of the equalities $a_1 = v_{a_1}$, …, $a_m = v_{a_m}$, $d = v_d$ is a basic descriptor (or selector), i.e., an atomic formula of the form $a = v_a$, for $a \in A \cup \{d\}$ and $v_a \in V_a$. Descriptor $a = v_a$ is supported by objects $u \in U$ for which the condition $a(u) = v_a$ holds. In the simplest form, the successor consists of a single descriptor. The rule's support takes a form of a set of objects matching all descriptors occurring in the rule's predecessor and successor, i.e., it is an intersection of the sets of objects supporting particular descriptors. Moreover, we can associate a given rule with its confidence value, expressed as a ratio of the cardinality of the support of the rule to the cardinality of the set of objects matching all descriptors from the rule's predecessor.

3.2 Decision Reducts

Definition 19 (decision reduct)**.** *[96] Let a consistent decision table $\mathbb{A} = (U, A \cup \{d\})$ be given. We say that $B \subseteq A$ is a decision reduct for $\mathbb{A}$, if and only if it is an irreducible subset of attributes such that*

$$IND(B) \subseteq IND(\{d\}). \tag{3.11}$$

Equivalently, we may say that $B \subseteq A$ is a decision reduct, if and only if it is an irreducible subset of attributes such that each pair $u_i, u_j \in U$ satisfying the inequality $d(u_i) \neq d(u_j)$ is discerned by B.

It is worth noting that for inconsistent decision tables, the decision reducts defined as above do not exist. However, with different defined criteria, we can still consider the process of reduction [138]. For instance, we can focus on reducing attributes while preserving discernibility:

Definition 20 (discernibility-based decision reduct)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. We say that $B \subseteq A$ is a discernibility-based decision reduct for $\mathbb{A}$, if and only if it is an irreducible subset of attributes such that it discerns the same pair of objects with different decision values as A:*

$$\forall_{u,u' \in U} (u\ DIS(A)\ u' \wedge d(u) \neq d(u')) \implies (u\ DIS(B)\ u') \tag{3.12}$$

Definition 21 (discernibility measure)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. The discernibility measure $disc_{\mathbb{A}} : 2^A \to \mathbb{N}$ is defined as follows for $B \subseteq A$:*

$$disc_{\mathbb{A}}(B) = |\{(u, u') \in U \times U : u\ DIS(B)\ u' \wedge d(u) \neq d(u')\}| \tag{3.13}$$

A general method for computing decision reducts was proposed using the Boolean formula constructs in [95]:

Proposition 2. *Let a consistent decision table $\mathbb{A} = (U, A \cup \{d\})$ be given. Consider the following Boolean formula with propositional variables $\overline{a}$ for $a \in A$:*

$$\tau = \bigwedge\nolimits_{u_i, u_j \in U : i<j \wedge d(u_i) \neq d(u_j)} \bigvee\nolimits_{a \in A : a(u_i) \neq a(u_j)} \overline{a} \tag{3.14}$$

An arbitrary subset $B \subseteq A$ is a decision reduct, if and only if the Boolean formula $\bigwedge_{a \in B} \overline{a}$ is a prime implicant for τ.

Proof. See [115] □

Definition 22 (minimal decision reduct)**.** *A decision reduct is called minimal if its cardinality is minimum among all decision reducts.*

One can easily notice that a decision reduct $B \subseteq A$ has to determine decision values within $\mathbb{A}$, i.e., it should be possible to cover U by decision rules with predecessors taking a form of conjunctions of descriptors $a_i = v_{a_i}$ for attributes $a_i \in B$ and successors $d = v_d$. From this follows the concept of functional dependency known from relational databases which can be equivalently introduced using if-then rules and discernibility [76].

As an example, for the well-known data set in Table 3.1, there are two decision reducts: {Outlook, Temperature, Wind} and {Outlook, Humidity, Wind} (or $\{O, T, W\}$ and $\{O, H, W\}$ for short). In Table 3.2, decision rules generated on the base of $\{O, T, W\}$ and $\{O, H, W\}$ are presented.

For inconsistent decision tables, a number of extensions of Definition 19 have been proposed, basing on such rough set related notions as positive regions [95], generalized decision functions [137] or rough membership functions [131]. Let us concentrate on the notion of a positive region as an example.

Definition 23 (positive region)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ and $B \subseteq A$ be given. By positive region $POS_{\mathbb{A}}(B)$ (or $POS(B)$ for short) we mean the subset of U, consisting of all objects that can be uniquely classified to the decision classes using attributes in B:*

$$POS(B) = \{u \in U : \forall_{u' \in [u]_B} d(u') = d(u)\} \tag{3.15}$$

The above can be written equivalently also by means of the equivalence classes induced by B as follows:

$$POS(B) = \bigcup \{E \in U/B : \forall_{u, u' \in E} d(u) = d(u')\} \tag{3.16}$$

It means that $POS(B)$ is a union of the equivalence classes induced by B within which the objects have consistent decision value.

It is also useful to recall the following function $\gamma : 2^A \to [0, 1]$ which is commonly used in the rough set theory to express a degree of dependence between a subset of attributes and the decision [95]:

$$\gamma(B) = \frac{|POS(B)|}{|U|} \tag{3.17}$$

Definition 24 (γ-decision reduct)**.** *Let a decision table $\mathbb{A} = (U, A \cup \{d\})$ be given. We say that $B \subseteq A$ is a γ-decision reduct for $\mathbb{A}$, if and only if it is an irreducible subset of attributes such that $\gamma(B) = \gamma(A)$, or $POS(B) = POS(A)$ equivalently.*

Table 3.2: Decision rules generated from decision reducts $\{O, T, W\}$ and $\{O, H, W\}$ along with the subsets of objects that support them.

decision reduct $\boldsymbol{\{O, T, W\}}$	
decision rules	**support**
$(O = \text{overcast}) \wedge (T = \text{cool}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 7 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{hot}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 3, 13 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{mild}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 12 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{cool}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 6 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{cool}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 5 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{mild}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 14 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{mild}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 4, 10 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{cool}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 9 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{hot}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 2 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{hot}) \wedge (W = \text{weak}) \Rightarrow (d = \text{no})$	$\llbracket 1 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{mild}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 11 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{mild}) \wedge (W = \text{weak}) \Rightarrow (d = \text{no})$	$\llbracket 8 \rrbracket$

decision reduct $\boldsymbol{\{O, H, W\}}$	
decision rules	**support**
$(O = \text{overcast}) \wedge (H = \text{high}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 12 \rrbracket$
$(O = \text{overcast}) \wedge (H = \text{high}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 3 \rrbracket$
$(O = \text{overcast}) \wedge (H = \text{normal}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 7 \rrbracket$
$(O = \text{overcast}) \wedge (H = \text{normal}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 13 \rrbracket$
$(O = \text{rain}) \wedge (H = \text{high}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 14 \rrbracket$
$(O = \text{rain}) \wedge (H = \text{high}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 4 \rrbracket$
$(O = \text{rain}) \wedge (H = \text{normal}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 6 \rrbracket$
$(O = \text{rain}) \wedge (H = \text{normal}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 5, 10 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{high}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 2 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{high}) \wedge (W = \text{weak}) \Rightarrow (d = \text{no})$	$\llbracket 1, 8 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{normal}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 11 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{normal}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 9 \rrbracket$

The following is a well-known fact enabling to express γ-decision reducts in terms of decision reducts over appropriately modified consistent decision tables. A decision table $\mathbb{A} = (U, A \cup \{d\})$ is inconsistent (see Definition 16 and its equivalents), if there exist indiscernible objects with different decision values and therefore these objects do not belong to $POS(A)$. From Definition 24 we have that a prerequisite for an attribute subset $B \subseteq A$ to be a γ-decision reduct is to preserve the positive region induced by the set of all attributes of A. We can modify the original decision attribute d by adding a special decision value "$\circledast$" as a common replacement of decisions for all of the conflicting indiscernible (with respect to the set of all attributes A) objects with different decisions. The modified decision table will be then consistent and moreover, a reduct for that table will be an explicit counterpart to a γ-decision reduct for the original data set. Similar constructions, also for various other attribute reductions criteria, were presented in [138].

Let us formalize the construction process, as it will be used further in the dissertation for arbitrary subsets $B \subseteq A$. Let a decision table $\mathbb{A} = (U, A \cup \{d\})$ be given. For a given subset $B \subseteq A$ we construct a modified consistent decision table $\mathbb{A}^\gamma_B = (U, B \cup \{d^\gamma_B\})$ defining a new special value $\circledast \notin V_d$ and putting a decision attribute $d^\gamma_B : U \to V_d \cup \{\circledast\}$ as follows:

$$d^\gamma_B(u) = \begin{cases} \circledast & \text{if } u \notin POS(B) \\ d(u) & \text{otherwise} \end{cases} \tag{3.18}$$

Examples of such modified tables are presented in Table 3.3. For the data set presented in Table 3.1, the case of $B = A$ would not be so interesting because this table is already consistent. However, if one would like to search for γ-decision reducts in tables with smaller amount of attributes (like in Table 3.3), then the above way of defining decision d^γ_B turns out to be very useful.

Table 3.3: Examples of decision tables $\mathbb{A}^\gamma_B = (U, B \cup \{d^\gamma_B\})$ for attribute subsets $B = \{O, T, H\}$ and $B = \{T, H, W\}$. In the middle, the original decisions are given.

	Outlook	**Temp.**	**Humidity**	$\mathbf{d^\gamma_B}$
1	sunny	hot	high	no
2	sunny	hot	high	no
3	overcast	hot	high	yes
4	rain	mild	high	$\circledast$
5	rain	cool	normal	$\circledast$
6	rain	cool	normal	$\circledast$
7	overcast	cool	normal	yes
8	sunny	mild	high	no
9	sunny	cool	normal	yes
10	rain	mild	normal	yes
11	sunny	mild	normal	yes
12	overcast	mild	high	yes
13	overcast	hot	normal	yes
14	rain	mild	high	$\circledast$

	Play
1	no
2	no
3	yes
4	yes
5	yes
6	no
7	yes
8	no
9	yes
10	yes
11	yes
12	yes
13	yes
14	no

	Temp.	**Humidity**	**Wind**	$\mathbf{d^\gamma_B}$
1	hot	high	weak	$\circledast$
2	hot	high	strong	no
3	hot	high	weak	$\circledast$
4	mild	high	weak	$\circledast$
5	cool	normal	weak	yes
6	cool	normal	strong	$\circledast$
7	cool	normal	strong	$\circledast$
8	mild	high	weak	$\circledast$
9	cool	normal	weak	yes
10	mild	normal	weak	yes
11	mild	normal	strong	yes
12	mild	high	strong	$\circledast$
13	hot	normal	weak	yes
14	mild	high	strong	$\circledast$

Positive region $POS(B)$ can be interpreted in terms of decision rules generated from the combination of values of attributes in B for objects contained in it. Rules generated on the basis of such objects are deterministic in terms of the confidence value equals to 1. Thus, the role of γ-decision reducts is to use possibly small subsets of attributes to cover data with deterministic rules as thoroughly as it would be possible when the whole set of attributes is considered.

Table 3.4: Decision rules generated from γ-decision reducts obtained for the original decision table limited to the attribute subsets $\{O, T, H\}$ and $\{T, H, W\}$ along with the sets of objects that support them. In both cases γ-decision reducts consist of all attributes of the limited tables, i.e., $\{O, T, H\}$ and $\{T, H, W\}$, respectively.

γ-decision reduct $\{O, T, H\}$	
decision rules	**support**
$(O = \text{overcast}) \wedge (T = \text{cool}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 7 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{hot}) \wedge (H = \text{high}) \Rightarrow (d = \text{yes})$	$\llbracket 3 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{hot}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 13 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{mild}) \wedge (H = \text{high}) \Rightarrow (d = \text{yes})$	$\llbracket 12 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{mild}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 10 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{cool}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 9 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{hot}) \wedge (H = \text{high}) \Rightarrow (d = \text{no})$	$\llbracket 1, 2 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{mild}) \wedge (H = \text{high}) \Rightarrow (d = \text{no})$	$\llbracket 8 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{mild}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 11 \rrbracket$

γ-decision reduct $\{T, H, W\}$	
decision rules	**support**
$(T = \text{cool}) \wedge (H = \text{normal}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 5, 9 \rrbracket$
$(T = \text{hot}) \wedge (H = \text{high}) \wedge (W = \text{strong}) \Rightarrow (d = \text{no})$	$\llbracket 2 \rrbracket$
$(T = \text{hot}) \wedge (H = \text{normal}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 13 \rrbracket$
$(T = \text{mild}) \wedge (H = \text{normal}) \wedge (W = \text{strong}) \Rightarrow (d = \text{yes})$	$\llbracket 11 \rrbracket$
$(T = \text{mild}) \wedge (H = \text{normal}) \wedge (W = \text{weak}) \Rightarrow (d = \text{yes})$	$\llbracket 10 \rrbracket$

The approach and construction described above, i.e., a transformation of an inconsistent decision table to its consistent counterpart, using an appropriate replacement of the decision attribute, is a well-known mechanism in rough set literature [132, 85]. Moreover, as the interrelations between different decision reduct (or superreduct) variants have been well-studied over a number of years [65, 138], this allows us to formally express the intuitions presented above.

Proposition 3. *Let an inconsistent decision table $\mathbb{A} = (U, A \cup \{d\})$ be given. An arbitrary subset $B \subseteq A$ is a γ-decision reduct in $\mathbb{A}$, if and only if B is a decision reduct in $\mathbb{A}^{\gamma}_{A} = (U, A \cup \{d^{\gamma}_{A}\})$, where d^{γ}_{A} is defined as in Equation* (3.18)*.*

Proof. See [138]. □

Finally, one more formulation is worth noting in the context of γ-decision reducts. An appropriate result expressing the existence of decision reducts in terms of a Boolean formula and its prime implicants is already shown in Proposition 2. Below we present analogous result with respect to γ-decision reducts.

Proposition 4. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Consider the following Boolean formula with propositional variables $\overline{a}$ for $a \in A$:*

$$\tau^{\gamma} = \bigwedge\nolimits_{u_i \in POS(A)} \bigwedge\nolimits_{u_j \in U : d(u_i) \neq d(u_j)} \bigvee\nolimits_{a \in A : a(u_i) \neq a(u_j)} \overline{a} \tag{3.19}$$

An arbitrary subset $B \subseteq A$ is a γ-decision reduct, if and only if the Boolean formula $\bigwedge_{a \in B} \overline{a}$ is a prime implicant for τ^{γ}.

Proof. See [115] □

3.3 Approximate Decision Reducts

The concept of a decision reduct is fundamental to the rough set theory and its applications. However, practical applications often involve large and noisy data sets. To address such challenges, several generalizations of the decision reducts have been proposed, such as the dynamic reducts [4] and the approximate decision reducts [131].

An approximate decision reduct can be defined as an irreducible subset of attributes that, under specified criterion, retains the decision-related information above the specified threshold. Criteria for calculating approximate decision reducts are usually based on some functions evaluating degrees of decision information that are induced by subsets of attributes. The choice of a function may depend on the nature of a data set and methods of inducing classifiers based on the attributes in reducts. From this point of view, the rough set literature may be regarded as a source of evaluation functions that link attribute subsets and rule based classifiers corresponding to those subsets in further steps of data exploration.

Let us focus on the so-called F-decision ε-reducts, where $F : 2^{A} \to [0, 1]$ is a function that can represent the above-discussed decision information and $\varepsilon \in [0, 1)$ decides how much of quality of determining d we agree to lose when operating with smaller subsets $B \subseteq A$ (and, as a consequence, shorter rules). The following notion was used for several functions in the literature:

Definition 25 (relative approximate decision reduct)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Let 2^{A} denote the power set of the set A. Let $\varepsilon \in [0, 1)$ and a nondecreasing monotone (with respect to the set inclusion) function $F : 2^{A} \to [0, 1]$ be given. We say that a subset $B \subseteq A$ is a relative F-decision ε-superreduct, if and only if the following holds:*

$$F(B) \geq (1 - \varepsilon) F(A) \tag{3.20}$$

A subset $B \subseteq A$ is a relative F-decision ε-reduct, if and only if it satisfies the above inequality and none of its proper subsets does it.

We will also refer to a modified version of Definition 25. The basic idea is that, in the definition below, subsets of attributes are evaluated straightforwardly rather than in relation to the whole set of available attributes.

Definition 26 (approximate decision reduct)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Let $\varepsilon \in [0, 1)$ and a nondecreasing monotone (with respect to the set inclusion) function $F : 2^A \to [0, 1]$ be given. We say that a subset $B \subseteq A$ is a F-decision ε-superreduct, if and only if the following holds:*

$$F(B) \geq 1 - \varepsilon \tag{3.21}$$

A subset $B \subseteq A$ is a F-decision ε-reduct, if and only if it satisfies the above inequality and none of its proper subsets does it.

The first comparison of approximate decision reducts and relative approximate decision reducts took place in [86], for a function $F : 2^A \to [0, 1]$ reflecting a degree of discernibility of objects belonging to different decision classes. Over the years, the concept of a relative approximate decision reduct gained popularity because of its ability to handle inconsistent decision tables. Consequently, various types of relative approximate decision reducts are often referred to, simply, as approximate decision reducts. However, in this dissertation we keep original terminology reflected by Definitions 25 and 26.

Let us consider some examples of measures that can play the role of F. Certainly, one can think about function γ defined by Equation (3.17). Another example was introduced in [134] as a function modeling the accuracy of a rule based classifier pointing at decisions which are most frequent within particular indiscernibility classes induced by an evaluated subset of attributes:

Definition 27 (majority function)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. The majority function $M : 2^A \to [0, 1]$ is defined as follows for $B \subseteq A$:*

$$M(B) = \frac{1}{|U|} \sum_{E \in U/B} \max_{k=1,\ldots,|V_d|} |E \cap X^{\langle k \rangle}| \tag{3.22}$$

Yet another example of F refers to a Bayesian extension of the classical rough set model, where rules induced by a given subset of attributes are pointing at the decision classes which become maximally frequent comparing to their overall occurrence in the data [125]:

Definition 28 (relative gain function)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. The relative gain function $R : 2^A \to [0, 1]$ is defined as follows for $B \subseteq A$:*

$$R(B) = \frac{1}{|V_d|} \sum_{E \in U/B} \max_{k=1,\ldots,|V_d|} \frac{|E \cap X^{\langle k \rangle}|}{|X^{\langle k \rangle}|} \tag{3.23}$$

For a consistent $\mathbb{A} = (U, A \cup \{d\})$, there is $\gamma(A) = M(A) = R(A) = 1$. Thus, for consistent tables the notions of relative F-decision ε-reduct and F-decision ε-reduct are equivalent to each other whenever F takes a form of γ, M or R. Also, for consistent tables, relative $\gamma/M/R$-decision 0-reducts are all equivalent to standard decision reducts. On the other hand, for inconsistent tables, those notions can be regarded as alternative ways to extend decision reducts, leading toward potentially different subsets of attributes for the same data.

For F-decision ε-reducts the value of ε can be understood as a threshold for the allowed decrease of classifier accuracy and can address the balance between simplicity and confidence of rules. For a higher ε, F-decision ε-reducts usually contain fewer attributes and the generated decision rules become shorter. Thus, by the cost of slight inconsistencies we gain higher simplicity and higher chance that attribute values observed for previously unseen cases will match predecessors of some existing rules. For a lower ε, generated F-decision ε-reducts contain more attributes and rules generated based on those attributes are potentially more complex but also more accurate over $\mathbb{A}$ treated as the training data.

The way of generating decision rules on the basis of subsets $B \subseteq A$ derived from the data as M-decision ε-reducts and R-decision ε-reducts depends on the particular function M or R; that is, successors of particular decision rules are related to the most frequent or relatively most frequent decision classes, as implicitly assumed when using the mentioned functions. However, in case of decision bireducts we will have no such limitation and the choice of objects can be more flexible. The differences in the approaches – first undertaken in [117] – will be discussed more deeply in Section 4.1. As for now, let us focus on the complexity characteristics related to the search of F-decision ε-reducts for the above-discussed functions. The following results will be later useful to understand complexity challenges behind some classes of decision bireducts. The propositions below follow the standard way of proving NP-hardness, which involves using polynomial reduction to transform a selected problem that is known to be NP-hard into a problem of finding a given type of minimal approximate reduct. In particular, the problem stated in Proposition 9 has been already addressed in [134]. Nevertheless, we partially recall the main ideas of its proof as the other problems are shown by means of analogous constructions.

The following considerations are analogous to those presented in [134, 136] but we present them here for completeness of the dissertation.

Definition 29 (graph)**.** *A graph is an ordered pair $\mathbb{G} = (\mathbb{V}, \mathbb{E})$, where $\mathbb{V}$ is the set of vertices (also nodes or points) and $\mathbb{E} \subseteq \mathbb{V} \times \mathbb{V}$ is the set of edges (also arrows or links). We say that a graph is non-directed if the relation $\mathbb{E}$ is symmetric.*

Definition 30 (dominating set)**.** *Let a non-directed graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ be given. We say that subset $\mathbb{W} \subseteq \mathbb{V}$ is a dominating set for the graph $\mathbb{G}$ if and only if*

$$Cov_{\mathbb{G}}(\mathbb{W}) = \mathbb{V}$$

where

$$Cov_{\mathbb{G}}(\mathbb{W}) = \mathbb{W} \cup \{v \in \mathbb{V} : \exists_{w \in \mathbb{W}}\ (v, w) \in \mathbb{E}\}$$

is the set of vertices that either belong to $\mathbb{W}$ or are adjacent (in terms of the graph's edges) to at least one member of $\mathbb{W}$.

Definition 31 (minimal dominating set problem)**.** *The Minimal Dominating Set Problem is an optimization problem of finding a minimal subset of vertices, which is a dominating set for a given undirected graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$.*

Proposition 5. *The Minimal Dominating Set Problem is NP-hard.*

Proof. This is one of the most basic NP-hard problems, cf. [37]. □

Proposition 6. *Let $\varepsilon \in [0, 1)$ be given. The problem of finding a minimal relative γ-decision ε-reduct for an input decision table is NP-hard.*

Proof. Let $\varepsilon \in [0, 1)$ be given. We will show a polynomial reduction of the NP-hard problem of finding a minimal dominating set in a non-directed graph to the problem that we consider. Let us show a polynomial transformation of $\mathbb{G}$ to a decision table $\mathbb{A}^{\varepsilon}_{\mathbb{G}} = (U^{\varepsilon}_{\mathbb{G}}, A^{\varepsilon}_{\mathbb{G}} \cup \{d^{\varepsilon}_{\mathbb{G}}\})$ that will be an input for the problem of finding a minimal relative γ-decision ε-reduct. The constructed decision table will have $|\mathbb{V}|$ binary conditional attributes $A^{\varepsilon}_{\mathbb{G}} = \{a_1, a_2, \ldots, a_{|\mathbb{V}|}\}$ (one for each vertex in $\mathbb{G}$). Formally,

$$a_j(u_i) = \begin{cases} 1 & \text{if } i = j \vee (v_i, v_j) \in \mathbb{E} \\ 0 & \text{otherwise} \end{cases} \tag{3.24}$$

for $i, j = 1, \ldots, |\mathbb{V}|$. We also add another $t(\varepsilon) = \lfloor \frac{|\mathbb{V}|\varepsilon}{1-\varepsilon} + 1 \rfloor$ objects with all conditional attributes set to 0, i.e., $a_j(u_i) = 0$, for $j = 1, \ldots, |\mathbb{V}|$ and $i = |\mathbb{V}| + 1, \ldots, |\mathbb{V}| + t(\varepsilon)$. For objects that correspond to graph's vertices we set the value of $d^{\varepsilon}_{\mathbb{G}}$ according to the following formula:

$$d^{\varepsilon}_{\mathbb{G}}(u_i) = \begin{cases} 0 & \text{if } i \leq |\mathbb{V}| \\ 1 & \text{otherwise} \end{cases} \tag{3.25}$$

$\mathbb{A}^{\varepsilon}_{\mathbb{G}}$ is consistent, i.e., $POS(A^{\varepsilon}_{\mathbb{G}}) = U^{\varepsilon}_{\mathbb{G}}$. An example of construction of a decision table $\mathbb{A}^{\varepsilon}_{\mathbb{G}}$ corresponding to a graph $\mathbb{G}$ can be seen in Figure 3.1.

The value $t(\varepsilon) = \lfloor \frac{|\mathbb{V}|\varepsilon}{1-\varepsilon} + 1 \rfloor$ is selected so to ensure that the relative γ-decision ε-superreduct property, i.e., $\gamma(B) \geq (1-\varepsilon)\gamma(A^{\varepsilon}_{\mathbb{G}})$ can be fulfilled, only if there is $|POS(B)| > |\mathbb{V}|$. In other words, we want to ensure that at least one of the objects $\{u_{|\mathbb{V}|+1}, \ldots, u_{|\mathbb{V}|+t(\varepsilon)}\}$ belongs to $POS(B)$ for $B \subseteq A^{\varepsilon}_{\mathbb{G}}$. Consequently, if one of them belongs to $POS(B)$ then the others must also belong (as all are equal to each other). Indeed, we have the following derivation:

$$\begin{array}{rcl} \gamma(B) & \geq & (1-\varepsilon)\gamma(A^{\varepsilon}_{\mathbb{G}}) \\ \text{we know that } \gamma(A^{\varepsilon}_{\mathbb{G}}) & = & 1 \\ \gamma(B) & \geq & (1-\varepsilon) \\ |POS(B)|/(|\mathbb{V}| + t(\varepsilon)) & \geq & (1-\varepsilon) \\ |POS(B)| & \geq & (1-\varepsilon)(|\mathbb{V}| + t(\varepsilon)) \end{array} \tag{3.26}$$

By transforming $t(\varepsilon) = \lfloor \frac{|\mathbb{V}|\varepsilon}{1-\varepsilon} + 1 \rfloor$ we get the following inequalities:

$$\begin{array}{rcl} t(\varepsilon) & > & \frac{|\mathbb{V}|\varepsilon}{1-\varepsilon} \\ t(\varepsilon)(1-\varepsilon) & > & |\mathbb{V}|\varepsilon \\ (t(\varepsilon) + |\mathbb{V}|)(1-\varepsilon) & > & |\mathbb{V}| \end{array} \tag{3.27}$$

Thus, we get that $\gamma(B) \geq (1-\varepsilon)\gamma(A^{\varepsilon}_{\mathbb{G}})$ holds, if and only if:

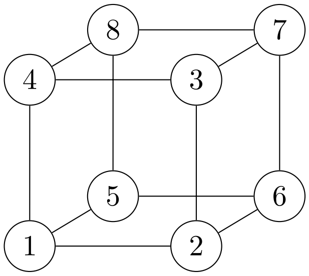
$$|POS(B)| \geq (1-\varepsilon)(|\mathbb{V}| + t(\varepsilon)) > |\mathbb{V}| \tag{3.28}$$

If $|POS(B)|$ is greater than $|\mathbb{V}|$, then it must include at least one object from $\{u_{|\mathbb{V}|+1}, \ldots, u_{|\mathbb{V}|+t(\varepsilon)}\}$. Since all such objects belong to decision class $X^{\langle d^{\varepsilon}_{\mathbb{G}}=1\rangle}$, then, for each $u_i \in X^{\langle d^{\varepsilon}_{\mathbb{G}}=0\rangle}$, there must exist at least one attribute $a_j \in B$ that discerns u_i from a considered object in $X^{\langle d^{\varepsilon}_{\mathbb{G}}=1\rangle}$, i.e., there must be $a_j(u_i) = 1$ for some $a_j \in B$. Hence, because of the way in which attributes in $A^{\varepsilon}_{\mathbb{G}}$ are defined, the relative γ-decision ε-superreduct B induces a dominating set for the graph $\mathbb{G}$, defined as $\mathbb{W}_B = \{v_j \in \mathbb{V} : a_j \in B\}$.

It remains to show that a subset $B \subseteq A$ which is the solution of the problem of finding a minimap γ-decision ε-reduct for decision table $\mathbb{A}^{\varepsilon}_{\mathbb{G}}$ corresponds to the smallest dominating set $\mathbb{W}_B$ for the graph $\mathbb{G}$. By contrary, suppose that there exists a dominating set $\mathbb{W}' \subseteq \mathbb{V}$ for graph $\mathbb{G}$ such that $|\mathbb{W}'| < |\mathbb{W}_B|$. Let $B' = \{a_j \in A^{\varepsilon}_{\mathbb{G}} : v_j \in \mathbb{W}'\}$. Because $\mathbb{W}'$ is a dominating set, we have that for each $u_i \in \{u_1, \ldots, u_{|\mathbb{V}|}\}$ either $v_i \in \mathbb{W}'$ or $(v_i, v_j) \in \mathbb{E}$ for some $v_j \in \mathbb{W}'$. In terms of the constructed decision table it means that for each $u_i \in \{u_1, \ldots, u_{|\mathbb{V}|}\}$ there exists at least one attribute $a_j \in B'$ such that $a_j(u_i) = 1$. So, the attribute subset B' discerns all objects $\{u_1, \ldots, u_{|\mathbb{V}|}\}$ from all objects $\{u_{|\mathbb{V}|+1}, \ldots, u_{|\mathbb{V}|+t(\varepsilon)}\}$. It means that B' is also a relative γ-decision ε-superreduct but with fewer elements. This means contradiction because then there would be also a γ-decision ε-reduct with fewer attributes than B. □

Proposition 7. *Let $\varepsilon \in [0, 1)$ be given. The problem of finding a minimal γ-decision ε-reduct for a decision table $\mathbb{A} = (U, A \cup \{d\})$ is NP-hard.*

Proof. The construction of $\mathbb{A}^{\varepsilon}_{\mathbb{G}}$ described in Proposition 6 gives a consistent decision table. As a consequence, the inequality solved to obtain the proper value for t has the same form here. Therefore, the proof for minimal γ-decision ε-reducts is the same as in Proposition 6. □



	$\boldsymbol{A^{\varepsilon}_{\mathbb{G}}}$								
$\boldsymbol{U^{\varepsilon}_{\mathbb{G}}}$	$\boldsymbol{a_1}$	$\boldsymbol{a_2}$	$\boldsymbol{a_3}$	$\boldsymbol{a_4}$	$\boldsymbol{a_5}$	$\boldsymbol{a_6}$	$\boldsymbol{a_7}$	$\boldsymbol{a_8}$	$\boldsymbol{d^{\varepsilon}_{\mathbb{G}}}$
u_1	1	1	0	1	1	0	0	0	0
u_2	1	1	1	0	0	1	0	0	0
u_3	0	1	1	1	0	0	1	0	0
u_4	1	0	1	1	0	0	0	1	0
u_5	1	0	0	0	1	1	0	1	0
u_6	0	1	0	0	1	1	1	0	0
u_7	0	0	1	0	0	1	1	1	0
u_8	0	0	0	1	1	0	1	1	0
u_{8+1}	0	0	0	0	0	0	0	0	1
u_{8+2}	0	0	0	0	0	0	0	0	1
...	...	...	...	...	...	...	...	...	...
$u_{8+t(\varepsilon)}$	0	0	0	0	0	0	0	0	1

Figure 3.1: Example of construction of decision table $\mathbb{A}^{\varepsilon}_{\mathbb{G}}$ for graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$.

	$\boldsymbol{A^{\varepsilon *}_{\mathbb{G}}}$								
$\boldsymbol{U^{\varepsilon *}_{G}}$	$\boldsymbol{a_1}$	$\boldsymbol{a_2}$	$\boldsymbol{a_3}$	$\boldsymbol{a_4}$	$\boldsymbol{a_5}$	$\boldsymbol{a_6}$	$\boldsymbol{a_7}$	$\boldsymbol{a_8}$	$\boldsymbol{d^{\varepsilon *}_{G}}$
u_1	1	1	1	1	1	1	1	1	1
u_2	2	2	1	2	2	1	1	1	2
u_3	3	3	1	3	3	1	1	1	3
u_4	4	4	4	4	4	4	4	4	1
u_5	5	5	5	4	4	5	4	4	2
u_6	6	6	6	4	4	6	4	4	3
u_7	7	7	7	7	7	7	7	7	1
u_8	7	8	8	8	7	7	8	7	2
u_9	7	9	9	9	7	7	9	7	3
u_{10}	10	10	10	10	10	10	10	10	1
u_{11}	11	10	11	11	10	10	10	11	2
u_{12}	12	10	12	12	10	10	10	12	3
u_{13}	13	13	13	13	13	13	13	13	1
u_{14}	14	13	13	13	14	14	13	14	2
u_{15}	15	13	13	13	15	15	13	15	3
u_{16}	16	16	16	16	16	16	16	16	1
u_{17}	16	17	16	16	17	17	17	16	2
u_{18}	16	18	16	16	18	18	18	16	3
u_{19}	19	19	19	19	19	19	19	19	1
u_{20}	19	19	20	19	19	20	20	20	2
u_{21}	19	19	21	19	19	21	21	21	3
u_{22}	22	22	22	22	22	22	22	22	1
u_{23}	22	22	22	23	23	22	23	23	2
u_{24}	22	22	22	24	24	22	24	24	3

Figure 3.2: Example of transformation of graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ displayed in Figure 3.1 to decision table $\mathbb{A}^{\varepsilon *}_{\mathbb{G}}$ for $\varepsilon = 0.6$.

Definition 32 (α-dominating set). *Let $\alpha \in (0,1]$ and a non-directed graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ be given. We say that subset $\mathbb{W} \subseteq \mathbb{V}$ is an α-dominating set for the graph $\mathbb{G}$ if and only if*

$$\frac{|Cov_{\mathbb{G}}(\mathbb{W})|}{|\mathbb{V}|} \geq \alpha$$

where $Cov_{\mathbb{G}}(\mathbb{W})$ is a set defined like in Definition 30.

Definition 33 (minimal α-dominating set problem). *For any $\alpha \in (0,1]$, a Minimal α-Dominating Set Problem is an optimization problem of finding a minimal subset of vertices, which is an α-dominating set for a given undirected graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$.*

Proposition 8. *For any $\alpha \in (0,1]$, the Minimal α-Dominating Set Problem is NP-hard.*

Proof. See Appendices in [134]. □

Proposition 9. *Let $\varepsilon \in [0,1)$ be given. The problem of finding a minimal relative M-decision ε-reduct for a decision table $\mathbb{A} = (U, A \cup \{d\})$ is NP-hard.*

Proof. The stated problem has been already addressed in [134]. For completeness of the dissertation we recall the ideas that lay behind this result. A polynomial reduction of the problem of finding a minimal α-dominating set in a graph $\mathbb{G}$ to the problem of finding a minimal relative M-decision ε-reduct was shown.

Using the notation from Definition 32, we have the condition for an α-dominating set $\mathbb{W}$ defined as $|Cov_{\mathbb{G}}(\mathbb{W})|/|\mathbb{V}| \geq \alpha$. The problem of finding a minimal α-dominating set was proved to be NP-hard in [134], for an arbitrary $\alpha > 0$.

For a given $\varepsilon \in [0,1)$, the formula for $\alpha(\varepsilon) \in (0,1]$ can be constructed in such a way that for each $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ being an input to the problem of finding a minimal $\alpha(\varepsilon)$-dominating set we can polynomially (with respect to the cardinality of $\mathbb{V}$) construct a decision table with its minimal relative M-decision ε-reduct equivalent to the minimal $\alpha(\varepsilon)$-dominating set in $\mathbb{G}$. Let us show a polynomial graph transformation to $\mathbb{A}^{\varepsilon *}_{\mathbb{G}} = (U^{\varepsilon *}_{\mathbb{G}}, A^{\varepsilon *}_{\mathbb{G}} \cup \{d^{\varepsilon *}_{\mathbb{G}}\})$ that will be an input for the problem of finding a minimal relative M-decision ε-reduct.

The constructed table has conditional attributes $A^{\varepsilon *}_{\mathbb{G}} = \{a_1, a_2, \ldots, a_{|\mathbb{V}|}\}$ (one for each vertex in $\mathbb{G}$) and one decision attribute $d^{\varepsilon *}_{\mathbb{G}}$. Let us define $m(\varepsilon) = \lfloor (1-\varepsilon)^{-1} + 1 \rfloor$. For each vertex from $v_i \in \mathbb{V}$, $m(\varepsilon)$ objects are added to $U^{\varepsilon *}_{\mathbb{G}}$, i.e., $U^{\varepsilon *}_{\mathbb{G}} = \{u_1, \ldots, u_{m(\varepsilon)|\mathbb{V}|}\}$. Each $a_j \in A^{\varepsilon *}_{\mathbb{G}}$ takes integers from a certain subset of $\{1, \ldots, m(\varepsilon)|\mathbb{V}|\}$. The decision attribute $d^{\varepsilon *}_{\mathbb{G}}$ takes integers from the set $\{1, \ldots, m(\varepsilon)\}$. Let us define $\text{chunk}(i) = \lceil \frac{i}{m(\varepsilon)} \rceil$. For any $u_i \in \{u_1, \ldots, u_{m(\varepsilon)|\mathbb{V}|}\}$ let us put

$$d^{\varepsilon *}_{\mathbb{G}}(u_i) = i - (\text{chunk}(i) - 1)m(\varepsilon) \tag{3.29}$$

and for each $a_j \in \{a_1, \ldots, a_{|\mathbb{V}|}\}$,

$$a_j(u_i) = \begin{cases} i & \text{if } \text{chunk}(i) = j \vee (v_{\text{chunk}(i)}, v_j) \in \mathbb{E} \\ (\text{chunk}(i) - 1)m(\varepsilon) + 1 & \text{otherwise} \end{cases} \tag{3.30}$$

It is important to notice that $\mathbb{A}^{\varepsilon *}_{\mathbb{G}}$ is a consistent decision table. An example of construction of reduction of a graph $\mathbb{G}$ to a decision table $\mathbb{A}^{\varepsilon *}_{\mathbb{G}}$ can be seen in Figure 3.2. Just like in the case of Proposition 6, let us put $\mathbb{W}_B = \{v_j \in \mathbb{V} : a_j \in B\}$. In [134], it was shown that in the above decision table $\mathbb{A}^{\varepsilon *}_{\mathbb{G}} = (U^{\varepsilon *}_{\mathbb{G}}, A^{\varepsilon *}_{\mathbb{G}} \cup \{d^{\varepsilon *}_{\mathbb{G}}\})$, for any $B \subseteq A^{\varepsilon *}_{\mathbb{G}}$, the value of majority function $M(B)$ satisfies the following equation:

$$M(B) = \frac{|Cov_{\mathbb{G}}(\mathbb{W}_B)|m(\varepsilon) + |\mathbb{V} \setminus Cov_{\mathbb{G}}(\mathbb{W}_B)|}{|U^{\varepsilon *}_{\mathbb{G}}|} \tag{3.31}$$

Let us now put:

$$\alpha(\varepsilon) = 1 - \frac{\varepsilon}{1 - m(\varepsilon)^{-1}} \tag{3.32}$$

For $\varepsilon \in [0, 1)$ it is quite easy to show that $\alpha(\varepsilon) \in (0, 1]$, so the problem of finding a minimal $\alpha(\varepsilon)$-dominating set is NP-hard. In [134], it was shown that the inequality $|Cov_{\mathbb{G}}(\mathbb{W}_B)|/|\mathbb{V}| \geq \alpha(\varepsilon)$ holds in $\mathbb{G}$, if and only if we have $M(B) \geq (1-\varepsilon)M(A_{\mathbb{G}}^{\varepsilon*})$ in $\mathbb{A}_{\mathbb{G}}^{\varepsilon*}$. Just like in the case of Proposition 6, we can now show that the smallest M-decision ε-reduct in $\mathbb{A}_{\mathbb{G}}^{\varepsilon*}$ must yield the smallest $\alpha(\varepsilon)$-dominating set in $\mathbb{G}$. □

Proposition 10. *Let $\varepsilon \in [0, 1)$ be given. The problem of finding a minimal M-decision ε-reduct for a decision table $\mathbb{A} = (U, A \cup \{d\})$ is NP-hard.*

Proof. Let us recall the proof of Proposition 9, where the transformation from the representation of the problem of finding a minimal $\alpha(\varepsilon)$-dominating set for a given graph $\mathbb{G}$ to the representation of the problem of finding a minimal relative M-decision ε-reduct was based on constructing consistent decision table $A_{\mathbb{G}}^{\varepsilon*}$. Thus, as for consistent decision tables the conditions $M(B) \geq (1-\varepsilon)M(A)$ and $M(B) \geq 1-\varepsilon$ are equivalent, the proof of NP-hardness for M-decision ε-reducts can be the same as for relative M-decision ε-reducts. □

Proposition 11. *Let $\varepsilon \in [0, 1)$ be given. The problem of finding a minimal relative R-decision ε-reduct for a decision table $\mathbb{A} = (U, A \cup \{d\})$ is NP-hard.*

Proof. To prove the proposition's claim let us focus on the Proposition 9. The constructions and calculations presented there, for the purposes of this proposition will remain unchanged. It is enough to notice that in case of the constructed decision table $\mathbb{A}_{\mathbb{G}}^{\varepsilon*}$ that corresponds to a graph $\mathbb{G}$, the use of measure R from Definition 28 instead of the measure M does not change the proof. Namely, in decision table $\mathbb{A}_{\mathbb{G}}^{\varepsilon*}$, we have $R(B) = M(B)$ for any $B \subseteq A_{\mathbb{G}}^{\varepsilon*}$. □

Proposition 12. *Let $\varepsilon \in [0, 1)$ be given. The problem of finding a minimal R-decision ε-reduct for a decision table $\mathbb{A} = (U, A \cup \{d\})$ is NP-hard.*

Proof. It can be based on the same observation as in the proof of Proposition 10, but now for function R instead of M. □

Chapter 4

Foundations of Decision Bireducts

The aim of this chapter is to introduce and provide a comprehensive understanding of the notion of decision bireducts, which can be seen as an extension of classical decision reducts. The notion emphasizes on both a subset of attributes that describe decisions and a subset of objects for which that description is valid. Decision bireducts offer a simple, flexible, and explicit approach to knowledge representation. We explore other variants of this concept and highlight analogies and differences compared to standard and approximate decision reducts from the theory of rough sets. Furthermore, our focus also includes ensembles of decision bireducts, where we investigate their properties and define optimization criteria.

4.1 Decision Bireducts

The first formulation of decision bireducts occurred in [140], where their Boolean characterization was described and a simple randomized algorithm aimed at their heuristic extraction from data was proposed by the analogy to classical decision reducts [3]. As mentioned in Chapter 1, the motivation to introduce decision bireducts was to facilitate an explicit analysis whether a classifier ensemble designed using different selected subsets of attributes do not repeat classification mistakes on the same objects in the training data set. Experiments reported in [140, 119], showed that diversification of subsets of attributes in this respect may be important in practice. This makes decision bireducts a clear and simple rough-set-based counterpart of some ensemble methods used in machine learning. In particular it was discussed in what sense ensembles of decision bireducts are better than ensembles of approximate reducts, which – although quite useful in practice [124] – do not allow for explicit analysis whether particular reducts repeat mistakes on the same objects.

Definition 34 (decision bireduct)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ and $B \subseteq A$, $\mathcal{X} \subseteq U$ be given. We say that B determines d within $\mathcal{X}$, further denoted as $B \Rrightarrow_{\mathcal{X}} d$, if and only if B discerns all pairs $u_i, u_j \in \mathcal{X}$ such that $d(u_i) \neq d(u_j)$. Further, we say that the pair $(\mathcal{X}, B)$ is a decision bireduct, if and only if the following holds:*

1. *There is $B \Rrightarrow_{\mathcal{X}} d$,*
2. *There is no proper subset $B' \subsetneq B$ such that $B' \Rrightarrow_{\mathcal{X}} d$,*
3. *There is no proper superset $\mathcal{X}' \supsetneq \mathcal{X}$ such that $B \Rrightarrow_{\mathcal{X}'} d$.*

We say that the objects in $\mathcal{X}$ are covered by the bireduct $(\mathcal{X}, B)$ and the objects in $U \setminus \mathcal{X}$ are uncovered by the bireduct $(\mathcal{X}, B)$.

A decision bireduct $(\mathcal{X}, B)$ can be regarded as the basis for an inexact functional dependency linking the subset of attributes B with the decision d in a degree $\mathcal{X}$, denoted as $B \Rrightarrow_{\mathcal{X}} d$ in Definition 34.

Every decision bireduct $(\mathcal{X}, B)$ may be understood as a pair consisting of an irreducible subset of attributes that can be evaluated by means of a non-extendable subset of objects for which it provides good classification. It was shown in [119] that $\mathcal{X}$ is actually the set-theoretic sum of objects supporting deterministic rules using the values of attributes in B to describe the values of d. Furthermore, the objects in $U \setminus \mathcal{X}$ can be treated as outliers of $B \Rrightarrow_{\mathcal{X}} d$.

Proposition 13. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. The following two monotony properties hold:*

1. *Let $B \subseteq B' \subseteq A$ and $\mathcal{X} \subseteq U$ be given. If $B \Rrightarrow_{\mathcal{X}} d$, then $B' \Rrightarrow_{\mathcal{X}} d$.*
2. *Let $B \subseteq A$ and $\mathcal{X}' \subseteq \mathcal{X} \subseteq U$ be given. If $B \Rrightarrow_{\mathcal{X}} d$, then $B \Rrightarrow_{\mathcal{X}'} d$.*

Proof. **(1.)** Since there is $B \Rrightarrow_{\mathcal{X}} d$, we know that B discerns all pairs $u_i, u_j \in \mathcal{X}$ for which $d(u_i) \neq d(u_j)$. It means that for each such pair there exists $a \in B$ that discerns the objects, i.e., $a(u_i) \neq a(u_j)$. As $B \subseteq B'$, the same fact holds for B' and therefore $B' \Rrightarrow_{\mathcal{X}} d$. **(2.)** We know that B discerns all pairs $u_i, u_j \in \mathcal{X}$ such that $d(u_i) \neq d(u_j)$. As $\mathcal{X}' \subseteq \mathcal{X}$, the same fact holds for $\mathcal{X}'$ which means that $B \Rrightarrow_{\mathcal{X}'} d$. □

The following result illustrates that decision reducts recalled in Definition 19 can be embedded into the space of decision bireducts.

Proposition 14. *Let $\mathbb{A} = (U, A \cup \{d\})$ and $B \subseteq A$ be given. B is a decision reduct, if and only if (U, B) is a decision bireduct.*

Proof. **($\Rightarrow$)** If B is a decision reduct, then from Definition 19 we have that B is an irreducible subset of attributes that discerns all pairs $u_i, u_j \in U$ such that $d(u_i) \neq d(u_j)$. Thus, (U, B) is a decision bireduct. **($\Leftarrow$)** If (U, B) is a decision bireduct, then from Definition 34 we know that $B \Rrightarrow_U d$ and there is no proper $B' \subsetneq B$ such that $B' \Rrightarrow_U d$. Thus, B is a decision reduct. □

On the other hand, decision bireducts can be represented using terminology of decision reducts as follows:

Proposition 15. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. For $B \subseteq A$ and $\mathcal{X} \subseteq U$, denote by $\mathbb{A}^B_{\mathcal{X}} = (\mathcal{X}, B \cup \{d\})$ the decision table obtained from $\mathbb{A}$ by removing objects outside $\mathcal{X}$ and attributes outside B. Then $(\mathcal{X}, B)$ is a decision bireduct for $\mathbb{A}$, if and only if both of the following conditions hold:*

1. *$\mathbb{A}^B_{\mathcal{X}}$ is consistent and there is no $\mathcal{X}' \supsetneq \mathcal{X}$ such that $\mathbb{A}^B_{\mathcal{X}'}$ is consistent,*
2. *B is a decision reduct for $\mathbb{A}^B_{\mathcal{X}}$.*

Proof. **($\Rightarrow$)** If $(\mathcal{X}, B)$ is a decision bireduct, then from Definition 34 we have that $B \Rrightarrow_{\mathcal{X}} d$ and there is no proper subset $B' \subsetneq B$ such that $B' \Rrightarrow_{\mathcal{X}} d$. Thus, B is a decision reduct for $\mathbb{A}^B_{\mathcal{X}}$. This implies also that $\mathbb{A}^B_{\mathcal{X}}$ is consistent. Finally, the fact that there is no proper superset $\mathcal{X}' \supsetneq \mathcal{X}$ such that $B \Rrightarrow_{\mathcal{X}'} d$, implies that there is no $\mathcal{X}' \supsetneq \mathcal{X}$ such that $\mathbb{A}^B_{\mathcal{X}'}$ is consistent. **($\Leftarrow$)** We need to check the three properties that define a decision bireduct in Definition 34. If B is a decision reduct for $\mathbb{A}^B_{\mathcal{X}}$, then from Definition 19 we know that B is irreducible subset of attributes that discern all pairs $u_i, u_j \in \mathcal{X}$ such that $d(u_i) \neq d(u_j)$. Thus, $B \Rrightarrow_{\mathcal{X}} d$ and there is no proper subset $B' \subsetneq B$ such that $B' \Rrightarrow_{\mathcal{X}} d$. Finally, using the statement that there is no $\mathcal{X}' \supsetneq \mathcal{X}$ such that $\mathbb{A}^B_{\mathcal{X}'}$ is consistent we have the third property of the decision bireduct definition, i.e., there is no proper superset $\mathcal{X}' \supsetneq \mathcal{X}$ such that $B \Rrightarrow_{\mathcal{X}'} d$. □

The above results help to adapt algorithms developed for searching standard decision reducts for the purpose of decision bireducts. Still, operating with decision bireducts provides wider possibilities of discovering and representing relationships in data. Each decision bireduct $(\mathcal{X}, B)$ can be regarded as the basis for an inexact functional dependency linking the subset of attributes B with the decision d in a degree $\mathcal{X}$. This means that B yields decision rules determining decision classes within $\mathcal{X}$, but possibly having counterexamples in $U \setminus \mathcal{X}$. This is a different rule-related interpretation than in the case of positive regions in Section 3.2, where only deterministic rules are considered. We can say that the rules corresponding to a decision bireduct $(\mathcal{X}, B)$ are deterministic only when we restrict the universe of objects to the subset $\mathcal{X} \subseteq U$.

Proposition 16. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given and $(\mathcal{X}, B)$ be a decision bireduct for $\mathbb{A}$. The following statements are true:*

1. *For each $E \in U/B$, all objects in $\mathcal{X} \cap E$ have the same decision value, further denoted as $v_{d(\mathcal{X}\cap E)} \in V_d$,*

2. *For each $E \in U/B$, all objects in E which have value $v_{d(\mathcal{X}\cap E)} \in V_d$ on d are contained in $\mathcal{X}$,*

3. *$\mathcal{X}$ equals to the union of supports of the following set of decision rules:*

$$Rules(\mathcal{X}, B) = \left\{ \bigwedge_{a\in B} (a = a(u)) \Rightarrow (d = d(u)) : u \in \mathcal{X} \right\} \tag{4.1}$$

Proof. (**1.**) Suppose that $\mathcal{X} \cap E$ contains objects from more than one decision class, i.e., there are objects $u_i, u_j \in \mathcal{X} \cap E$ such that $d(u_i) \neq d(u_j)$. It would violate Definition 34 – if $d(u_i) \neq d(u_j)$, then there must exist $a \in B$ such that $a(u_i) \neq a(u_j)$, so u_i and u_j cannot belong to the same E. (**2.**) Suppose that $\mathcal{X} \cap E$ contains objects with the same decision but there is also $u \in E \setminus \mathcal{X}$ with the same decision as objects in $\mathcal{X} \cap E$. It would violate Definition 34 – there would exist a superset $\mathcal{X} \cup \{u\} \supsetneq \mathcal{X}$ such that $B \Rrightarrow_{\mathcal{X}\cup\{u\}} d$. (**3.**) Because each $u \in \mathcal{X}$ belongs to the support of its corresponding rule $\bigwedge_{a\in B}(a = a(u)) \Rightarrow (d = d(u))$, we know that $\mathcal{X}$ is a subset of the considered union. Now, suppose that there exists $u_i \in U \setminus \mathcal{X}$ which supports a rule in $Rules(\mathcal{X}, B)$. It would mean that there exists $u_j \in \mathcal{X}$ such that $d(u_i) = d(u_j)$ and $a(u_i) = a(u_j)$ for every $a \in B$. However, in such a case $B \Rrightarrow_{\mathcal{X}} d$ could be actually extended toward $B \Rrightarrow_{\mathcal{X}\cup\{u_i\}} d$, so $(\mathcal{X}, B)$ would not be a decision bireduct. □

Proposition 16 was formulated to better illustrate how decision rules can be generated from decision bireducts. In other words, it shows in what sense decision bireducts $(\mathcal{X}, B)$ can be equivalently represented by collections of decision rules with predecessors based on attributes in B and supporting sets of objects summing up to $\mathcal{X}$. We refer to Table 4.1 for some examples of decision bireducts and their corresponding decision rules.

Table 4.1: Decision rules generated from decision bireducts $(\llbracket 1..3, 6..9, 11..14 \rrbracket, \{O, H\})$ and $(\llbracket 1..4, 6..10, 12, 13 \rrbracket, \{O, T\})$ along with the sets of objects that support them for the decision table from Table 3.1.

decision bireduct $(\llbracket 1..3, 6..9, 11..14 \rrbracket, \{O, H\})$	
decision rules	**support**
$(O = \text{overcast}) \wedge (H = \text{high}) \Rightarrow (d = \text{yes})$	$\llbracket 3, 12 \rrbracket$
$(O = \text{overcast}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 7, 13 \rrbracket$
$(O = \text{rain}) \wedge (H = \text{high}) \Rightarrow (d = \text{no})$	$\llbracket 14 \rrbracket$
$(O = \text{rain}) \wedge (H = \text{normal}) \Rightarrow (d = \text{no})$	$\llbracket 6 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{high}) \Rightarrow (d = \text{no})$	$\llbracket 1, 2, 8 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 9, 11 \rrbracket$

decision bireduct $(\llbracket 1..4, 6..10, 12, 13 \rrbracket, \{O, T\})$	
decision rules	**support**
$(O = \text{overcast}) \wedge (T = \text{cool}) \Rightarrow (d = \text{yes})$	$\llbracket 7 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{hot}) \Rightarrow (d = \text{yes})$	$\llbracket 3, 13 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{mild}) \Rightarrow (d = \text{yes})$	$\llbracket 12 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{cool}) \Rightarrow (d = \text{no})$	$\llbracket 6 \rrbracket$
$(O = \text{rain}) \wedge (T = \text{mild}) \Rightarrow (d = \text{yes})$	$\llbracket 4, 10 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{cool}) \Rightarrow (d = \text{yes})$	$\llbracket 9 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{hot}) \Rightarrow (d = \text{no})$	$\llbracket 1, 2 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{mild}) \Rightarrow (d = \text{no})$	$\llbracket 8 \rrbracket$

4.2 γ-Decision Bireducts

In [140], the following modification of Definition 34 was also considered:

Definition 35 (γ-decision bireduct)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ and subsets $\mathcal{X} \subseteq U$, $B \subseteq A$ be given. We say that B γ-determines d within $\mathcal{X}$, further denoted as $B \Rrightarrow^{\gamma}_{\mathcal{X}} d$, if and only if B discerns all pairs $u_i \in \mathcal{X}$, $u_j \in U$ such that $d(u_i) \neq d(u_j)$. Further, we say that the pair $(\mathcal{X}, B)$ is a γ-decision bireduct, if and only if the following holds:*

1. *There is $B \Rrightarrow^{\gamma}_{\mathcal{X}} d$,*

2. *There is no proper subset $B' \subsetneq B$ such that $B' \Rrightarrow^{\gamma}_{\mathcal{X}} d$,*

3. *There is no proper superset $\mathcal{X}' \supsetneq \mathcal{X}$ such that $B \Rrightarrow^{\gamma}_{\mathcal{X}'} d$.*

We say that the objects in $\mathcal{X}$ are γ-covered by the γ-decision bireduct $(\mathcal{X}, B)$ and the objects in $U \setminus \mathcal{X}$ are γ-uncovered by the γ-decision bireduct $(\mathcal{X}, B)$.

In contrary to decision bireducts it imposes the requirement that objects belonging to the γ-decision bireduct should be discerned not only in a context of $\mathcal{X}$ but with respect to the entire U. Basic properties of both considered versions of a decision bireduct remain similar to each other:

Proposition 17. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. The following two monotony properties hold:*

1. *Let $\mathcal{X} \subseteq U$ and $B \subseteq B' \subseteq A$ be given. If $B \Rrightarrow^{\gamma}_{\mathcal{X}} d$, then $B' \Rrightarrow^{\gamma}_{\mathcal{X}} d$.*

2. *Let $\mathcal{X}' \subseteq \mathcal{X} \subseteq U$ and $B \subseteq A$ be given. If $B \Rrightarrow^{\gamma}_{\mathcal{X}} d$, then $B \Rrightarrow^{\gamma}_{\mathcal{X}'} d$.*

Proof. The proof is analogous to the proof of Proposition 13. □

Proposition 18. *Let $\mathbb{A} = (U, A \cup \{d\})$ and $B \subseteq A$ be given. B is a decision reduct, if and only if (U, B) is a γ-decision bireduct.*

Proof. **(⇒)** If B is a decision reduct, then from Definition 19 we have that B is an irreducible subset of attributes that discerns all pairs $u_i, u_j \in U$ such that $d(u_i) \neq d(u_j)$. As a result, (U, B) is also a γ-decision bireduct. **(⇐)** If (U, B) is a γ-decision bireduct, then from Definition 35 we know that $B \Rrightarrow^{\gamma}_{U} d$ and there is no $B' \subsetneq B$ such that $B' \Rrightarrow^{\gamma}_{U} d$. It means that B is a decision reduct. □

As before, a γ-decision bireduct $(\mathcal{X}, B)$ can be regarded as an inexact functional dependency linking the subset of attributes B with the decision d in a degree $\mathcal{X}$ where γ in $\Rrightarrow^{\gamma}_{\mathcal{X}}$ indicates a type of dependence. The difference between the two mentioned kinds of decision bireducts lies in the scope of considered objects. The latter type focuses on the discernibility on the whole set of objects, while the former considers a local scope and ensures discernibility only among the objects covered by a given decision bireduct. This may lead toward different ways of applying both versions of decision bireducts in practice. For example, in [118] we considered incremental computation of decision bireducts over data stream buffers. In such a case, U cannot be accessed as a whole, so the only way to proceed is to adapt Definition 34. On the other hand, an advantage of γ-decision bireducts is their analogy to positive regions. Indeed, in Definition 35, the object can be added to $\mathcal{X}$, only if it belongs to $POS(B)$. This is actually why we use "γ" while talking about γ-decision bireducts.

Proposition 19. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Let subsets $B \subseteq A$ and $\mathcal{X} \subseteq U$ be given. Then $(\mathcal{X}, B)$ is a γ-decision bireduct, if and only if the following two properties hold:*

1. *$\mathcal{X} = POS(B)$,*
2. *There is no proper subset $B' \subsetneq B$ such that $POS(B') = POS(B)$.*

Proof. **(⇐)** We need to check three requirements for a γ-decision bireduct. Let us first show that $B \Rrightarrow^{\gamma}_{POS(B)} d$ holds. Suppose that $B \not\Rrightarrow^{\gamma}_{POS(B)} d$. Following Definition 35 there would exist a pair of objects $u_i \in POS(B), u_j \in U$ which is not discerned by B and such that $d(u_i) \neq d(u_j)$. However, this would mean that $u_j \in [u_i]_B$, so – according to Equation (3.15) – u_i could not belong to $POS(B)$. Let us now proceed with the third requirement in Definition 35. If $B \Rrightarrow^{\gamma}_{\mathcal{X}} d$ held for some $\mathcal{X} \supsetneq POS(B)$, then $POS(B)$ could be extended – a contradiction. Finally, consider the second requirement for $(\mathcal{X}, B)$ to be a γ-decision bireduct. By following the same reasoning as above, we know that for any $B' \subsetneq B$ we can have $B' \not\Rrightarrow^{\gamma}_{\mathcal{X}'} d$ only for subsets $\mathcal{X}' \subseteq POS(B')$. Given $POS(B') \subsetneq POS(B)$, we also know that there is no $B' \subsetneq B$ such that $B' \not\Rrightarrow^{\gamma}_{POS(B)} d$ holds. **(⇒)** Suppose that $\mathcal{X} \neq POS(B)$. This means that either $POS(B) \setminus \mathcal{X} \neq \emptyset$ or $\mathcal{X} \setminus POS(B) \neq \emptyset$. In the first case, we would have $u \in POS(B) \setminus \mathcal{X}$. From Definition 23 we know that for $u \in POS(B)$ all objects in $[u]_B$ have the same value for d. However, this means that the whole $[u]_B$ should be included in $\mathcal{X}$ because, otherwise, $\mathcal{X}$ would not be non-extendable. In the second case, we would have $u \in \mathcal{X} \setminus POS(B)$. It would mean that $[u]_B$ contains objects from more than one decision class because otherwise it would be contained in $POS(B)$. This also leads to contradiction because we would then have at least one pair of objects $u_i = u \in \mathcal{X}$ and $u_j \in U$ that is not discerned by B, such that $d(u_i) \neq d(u_j)$, thus violating the definition of a γ-decision bireduct. Hence, $\mathcal{X} = POS(B)$. Now, suppose that there exists $B' \subsetneq B$ such that $POS(B) = POS(B')$. As we have just shown above, we know that $B' \Rrightarrow^{\gamma}_{POS(B')} d$. Therefore, because $\mathcal{X} = POS(B) = POS(B')$, we would have that $B' \Rrightarrow^{\gamma}_{\mathcal{X}} d$. It would contradict with the requirement of irreducibility of B in Definition 35. □

Proposition 19 leads to several simple and useful observations. First of all, we can transform the problem of searching for γ-decision bireducts in a decision table $\mathbb{A} = (U, A \cup \{d\})$ to the problem of searching decision reducts in a modified decision table $\mathbb{A}^{\gamma}_{A} = (U, A \cup \{d^{\gamma}_{A}\})$ described in Section 3.2 as the applied modification leaves the objects from the positive region in the original table unchanged (see an example in Table 3.3).

Also, as a consequence of the considered proposition, if $(\mathcal{X}, B)$ is a γ-decision bireduct for $\mathbb{A}$, then there is no other subset $U \supseteq \mathcal{X}' \neq \mathcal{X}$, such that $(\mathcal{X}', B)$ is also a γ-decision bireduct. In other words, for a given B there can be at most one γ-decision bireduct with that attribute subset. The assumption of the existence of such a subset $U \supseteq \mathcal{X}' \neq \mathcal{X}$ such that $(\mathcal{X}', B)$ is a γ-decision bireduct will lead to a contradiction. Without loss of generality let us assume that $\mathcal{X}' \setminus \mathcal{X} \neq \emptyset$ and let $u \in \mathcal{X}' \setminus \mathcal{X}$. There are two possibilities either $[u]_B$ contains objects from exactly one decision class or more than one decision class. If the former possibility holds, then $(\mathcal{X}, B)$ is not a γ-decision bireduct as its subset of objects may be extended by $[u]_B$. If the latter possibility holds, then we can show at least one pair of objects $u_i = u \in \mathcal{X}'$, $u_j \in [u]_B$ not discerned by B, such that $d(u_i) \neq d(u_j)$, thus also violating the definition of γ-decision bireduct.

Table 4.2 presents decision bireducts and γ-decision bireducts for the data in Table 3.1. One can notice that, unlike in the case of γ-decision bireducts, the same $B \subseteq A$ can potentially occur as a component of many decision bireducts, with different subsets of objects. Moreover, when comparing the same subsets $B \subseteq A$, the corresponding subsets $\mathcal{X} \subseteq U$ are usually larger for decision bireducts than for γ-decision bireducts. This is because the conditions for belonging to $\mathcal{X}$ are more restrictive in Definition 35 than in Definition 34.

The last observation based on Proposition 19 is that γ-decision bireducts can be interpreted by means of decision rules just like in the case of γ-decision reducts in Section 3.2. More precisely, it is possible to formulate a statement analogous to Proposition 16, now for γ-decision bireducts. Certainly, for a γ-decision bireduct $(\mathcal{X}, B)$, where $\mathcal{X} = POS(B)$, the rules with supports summing up to $\mathcal{X}$ will be all deterministic with respect to the whole U. We can describe such rules as follows:

$$Rules_{\gamma}(POS(B), B) = \left\{ \bigwedge_{a \in B} (a = a(u)) \Rightarrow (d = d(u)) : u \in POS(B) \right\} \tag{4.2}$$

In Table 4.3 one can find rules generated for sample γ-decision bireducts. The way of their formulation and usage during classification of new cases is the same as for γ-decision reducts considered in Section 3.2. In general, the support and confidence coefficients of decision rules generated from decision bireducts and γ-decision bireducts can be utilized during the new case classification just like in other rule based decision systems. In particular, when dealing with ensembles of decision bireducts or γ-decision bireducts [140], we can use analogous voting mechanisms as those considered in other rough set related approaches based on reducts and rules [143].

Table 4.2: A complete list of decision bireducts and γ-decision bireducts for Table 3.1. It is presented in a tabular form where each row corresponds to a subset of attributes $B \subseteq A$, while the cells in the appropriate columns contain all possible subsets of objects $\mathcal{X} \subseteq U$ such that $(\mathcal{X}, B)$ forms a decision bireduct or a γ-decision bireduct. The cells containing NA mean that for a given attribute subset, there is no subset of objects such that they meet the criteria of Definition 34 or Definition 35, respectively. Specifically, for $\{O, T, H, W\}$ the results are not available because, due to the characteristics of the decision table, there is redundancy in the whole set of attributes and hence the attributes irreducibility condition is not satisfied for Definitions 34 and 35, respectively.

$B \subseteq A$	$\mathcal{X}$ **for decision bireducts** $(\mathcal{X}, B)$	$\mathcal{X}$ **for** γ**-decision bireduct** $(\mathcal{X}, B)$
$\emptyset$	$\llbracket 1,2,6,8,14 \rrbracket$; $\llbracket 3..5,7,9..13 \rrbracket$	$\emptyset$
$\{O\}$	$\llbracket 1..5,7,8,10,12,13 \rrbracket$; $\llbracket 1..3,6..8,12..14 \rrbracket$; $\llbracket 3,6,7,9,11..14 \rrbracket$	$\llbracket 3,7,12,13 \rrbracket$
$\{T\}$	$\llbracket 1,2,4,5,7,9..12 \rrbracket$; $\llbracket 1,2,4,6,10..12 \rrbracket$; $\llbracket 1,2,5,7..9,14 \rrbracket$; $\llbracket 3,4,6,10..13 \rrbracket$; $\llbracket 3,5,7..9,13,14 \rrbracket$; $\llbracket 3,6,8,13,14 \rrbracket$	NA
$\{H\}$	$\llbracket 1,2,5,7..11,13,14 \rrbracket$; $\llbracket 3,4,6,12 \rrbracket$	NA
$\{W\}$	$\llbracket 1,7,8,11,12 \rrbracket$; $\llbracket 2..6,9,10,13,14 \rrbracket$	NA
$\{O, T\}$	$\llbracket 1..5,7..10,12,13 \rrbracket$; $\llbracket 1..5,7,9..13 \rrbracket$; $\llbracket 1..4,6..10,12,13 \rrbracket$; $\llbracket 1..4,6,7,9..13 \rrbracket$; $\llbracket 1..3,5,7..9,12..14 \rrbracket$; $\llbracket 1..3,5,7,9,11..14 \rrbracket$; $\llbracket 1..3,6..9,12..14 \rrbracket$; $\llbracket 1..3,6,7,9,11..14 \rrbracket$	$\llbracket 1..3,7,9,12,13 \rrbracket$
$\{O, H\}$	$\llbracket 1..5,7..13 \rrbracket$; $\llbracket 1..4,6..9,11..13 \rrbracket$; $\llbracket 1..3,5,7..14 \rrbracket$; $\llbracket 1..3,6..9,11..14 \rrbracket$;	$\llbracket 1..3,7..9,11..13 \rrbracket$
$\{O, W\}$	$\llbracket 1..8,10,12..14 \rrbracket$; $\llbracket 1,3..8,10..14 \rrbracket$; $\llbracket 2..7,9,10,12..14 \rrbracket$; $\llbracket 3..7,9..14 \rrbracket$	$\llbracket 3..7,10,12..14 \rrbracket$
$\{T, H\}$	$\llbracket 1,2,4,5,7,9..13 \rrbracket$; $\llbracket 1,2,4,6,10..13 \rrbracket$; $\llbracket 1,2,6,8,10,11,13,14 \rrbracket$; $\llbracket 3,5,7..11,13,14 \rrbracket$; $\llbracket 3,6,8,10,11,13,14 \rrbracket$	$\llbracket 10,11,13 \rrbracket$
$\{T, W\}$	$\llbracket 1,2,4..6,9..12 \rrbracket$; $\llbracket 1,2,4..6,9,10,14 \rrbracket$; $\llbracket 1,2,4,5,7,9,10,14 \rrbracket$; $\llbracket 1,2,5,6,8,9,11,12 \rrbracket$; $\llbracket 1,2,5,6,8,9,14 \rrbracket$; $\llbracket 1,2,5,7..9,11,12 \rrbracket$; $\llbracket 2..6,9..13 \rrbracket$; $\llbracket 2..5,7,9..13 \rrbracket$; $\llbracket 2..5,7,9,10,13,14 \rrbracket$; $\llbracket 2,3,5,6,8,9,11..13 \rrbracket$; $\llbracket 2,3,5,6,8,9,13,14 \rrbracket$; $\llbracket 2,3,5,7..9,11..13 \rrbracket$; $\llbracket 2,3,5,7..9,13,14 \rrbracket$	$\llbracket 2,5,9 \rrbracket$
$\{H, W\}$	$\llbracket 1,2,5,6,8..10,13,14 \rrbracket$; $\llbracket 1,5,6,8..10,12,13 \rrbracket$; $\llbracket 1,5,7..13 \rrbracket$; $\llbracket 2..5,7,9..11,13,14 \rrbracket$; $\llbracket 3..6,9,10,12,13 \rrbracket$	$\llbracket 5,9,10,13 \rrbracket$
$\{O, T, H\}$	$\llbracket 1..4,6..13 \rrbracket$; $\llbracket 1..3,6..14 \rrbracket$	$\llbracket 1..3,7..13 \rrbracket$
$\{O, T, W\}$	$\llbracket 1..14 \rrbracket$	$\llbracket 1..14 \rrbracket$
$\{O, H, W\}$	$\llbracket 1..14 \rrbracket$	$\llbracket 1..14 \rrbracket$
$\{T, H, W\}$	$\llbracket 1,2,4..6,9..13 \rrbracket$; $\llbracket 1,2,4..6,9..11,13,14 \rrbracket$; $\llbracket 1,2,4,5,7,9..11,13,14 \rrbracket$; $\llbracket 1,2,5,6,8..13 \rrbracket$; $\llbracket 1,2,5,6,8..11,13,14 \rrbracket$; $\llbracket 1,2,5,7..13 \rrbracket$; $\llbracket 2..6,9..11,13,14 \rrbracket$; $\llbracket 2,3,5,6,8..13 \rrbracket$; $\llbracket 2,3,5,6,8..11,13,14 \rrbracket$; $\llbracket 2,3,5,7..13 \rrbracket$; $\llbracket 2,3,5,7..11,13,14 \rrbracket$	$\llbracket 2,5,9..11,13 \rrbracket$
$\{O, T, H, W\}$	NA	NA

Table 4.3: Decision rules generated from γ-decision bireducts $(\llbracket 1..3, 7..9, 11..13 \rrbracket, \{O, H\})$ and $(\llbracket 1..3, 7, 9, 12, 13 \rrbracket, \{O, T\})$ along with the sets of objects that support them.

$\boldsymbol{\gamma}$**-decision bireduct** $(\llbracket \mathbf{1..3, 7..9, 11..13} \rrbracket, \{\boldsymbol{O}, \boldsymbol{H}\})$	
decision rules	**support**
$(O = \text{overcast}) \wedge (H = \text{high}) \Rightarrow (d = \text{yes})$	$\llbracket 3, 12 \rrbracket$
$(O = \text{overcast}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 7, 13 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{high}) \Rightarrow (d = \text{no})$	$\llbracket 1, 2, 8 \rrbracket$
$(O = \text{sunny}) \wedge (H = \text{normal}) \Rightarrow (d = \text{yes})$	$\llbracket 9, 11 \rrbracket$

$\boldsymbol{\gamma}$**-decision bireduct** $(\llbracket \mathbf{1..3, 7, 9, 12, 13} \rrbracket, \{\boldsymbol{O}, \boldsymbol{T}\})$	
decision rules	**support**
$(O = \text{overcast}) \wedge (T = \text{cool}) \Rightarrow (d = \text{yes})$	$\llbracket 7 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{hot}) \Rightarrow (d = \text{yes})$	$\llbracket 3, 13 \rrbracket$
$(O = \text{overcast}) \wedge (T = \text{mild}) \Rightarrow (d = \text{yes})$	$\llbracket 12 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{cool}) \Rightarrow (d = \text{yes})$	$\llbracket 9 \rrbracket$
$(O = \text{sunny}) \wedge (T = \text{hot}) \Rightarrow (d = \text{no})$	$\llbracket 1, 2 \rrbracket$

4.3 Heuristic Search for Decision Bireducts

There are a number of possible approaches to searching for decision bireducts. Let us recall and extend some previously obtained results linking decision bireducts with prime implicants of Boolean formulae representing discernibility in decision tables [140]. As we know, information regarding all differences between objects from different decision classes in a table $\mathbb{A} = (U, A \cup \{d\})$ can be represented (see Proposition 2) by a formula $\tau = \bigwedge_{u_i, u_j \in U : i<j \wedge d(u_i) \neq d(u_j)} \bigvee_{a \in A : a(u_i) \neq a(u_j)} \overline{a}$ which encodes the discernibility matrix of $\mathbb{A}$. The set of all decision reducts of $\mathbb{A}$ can be constructed by considering all prime implicants of τ [95]. By an analogy to this result, we can work with decision bireducts by adapting some well-known Boolean techniques [16]. A similar approach was also taken by researchers working on other methods related to rough sets, e.g., in the context of association rules [86]. One can also refer to the methods presented in this subsection as algorithmic constructions which reformulate the input data in order to harness previously developed attribute reduction heuristics directly to the task of searching for decision bireducts. The first of such considered methods refers to the following representation:

Proposition 20. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Consider the following Boolean formula with propositional variables $\overline{i}$, for $i = 1, \ldots, |U|$, and $\overline{a}$ for $a \in A$:*

$$\tau_{bi} = \bigwedge_{u_i, u_j \in U : i<j \wedge d(u_i) \neq d(u_j)} \left(\overline{i} \vee \overline{j} \vee \bigvee_{a \in A : a(u_i) \neq a(u_j)} \overline{a} \right) \tag{4.3}$$

An arbitrary pair $(\mathcal{X}, B)$, where $\mathcal{X} \subseteq U$, $B \subseteq A$, is a decision bireduct, if and only if the Boolean formula $\bigwedge_{u_i \in U \setminus \mathcal{X}} \overline{i} \wedge \bigwedge_{a \in B} \overline{a}$ is a prime implicant for τ_{bi}.

Proof. A product term of literals P is an implicant of a Boolean formula τ, if the fact that P is evaluated to *true* always implies that τ is evaluated to *true* as well. A prime implicant of a Boolean formula is an implicant that is minimal with regard to inclusion, i.e., removing any of its literals causes that it is no longer an implicant.

Let $B \subseteq A$ and $\mathcal{X} \subseteq U$ be given. Consider $P = \bigwedge_{u_i \in U \setminus \mathcal{X}} \overline{i} \wedge \bigwedge_{a \in B} \overline{a}$. As a first step, we will show that:

$$B \Rrightarrow_{\mathcal{X}} d \quad \Longleftrightarrow \quad P \text{ is an implicant for } \tau_{bi} \tag{4.4}$$

(⇒) Suppose that P is not an implicant for τ_{bi}. Hence, there exists a valuation of propositional variables for which P is $true$ but τ_{bi} is $false$. Thus, there is at least one clause in the form of $f_\lambda = \left(\overline{i_\lambda} \vee \overline{j_\lambda} \vee \bigvee_{a\in A:a(u_{i_\lambda})\neq a(u_{j_\lambda})} \overline{a}\right)$, where $u_{i_\lambda}, u_{j_\lambda} \in U$ and $d(u_{i_\lambda}) \neq d(u_{j_\lambda})$, which is $false$ for the considered valuation. As f_λ is a disjunction of propositional variables, all of its elements have to be assigned with $false$. Since P is $true$ and both $\overline{i_\lambda}$ and $\overline{j_\lambda}$ are $false$, it means that neither $\overline{i_\lambda}$ nor $\overline{j_\lambda}$ are a part of P and in a consequence, as P contains the variables that correspond to all objects in $U \setminus \mathcal{X}$, we know that $u_{i_\lambda}, u_{j_\lambda} \in \mathcal{X}$. We also know that P cannot contain the variables that correspond to attributes for which u_{i_λ} and u_{j_λ} have different values, i.e., for all $a \in A$ such that $a(u_{i_\lambda}) \neq a(u_{j_\lambda})$, we know that $a \notin B$. This means that there are at least two objects $u_{i_\lambda}, u_{j_\lambda} \in \mathcal{X}$ such that $d(u_{i_\lambda}) \neq d(u_{j_\lambda})$, which are not discerned by B. Therefore, $B \Rrightarrow_{\mathcal{X}} d$ does not hold.
(⇐) Suppose that $B \Rrightarrow_{\mathcal{X}} d$ does not hold. This means that there exists at least one pair of objects $u_{i_\lambda}, u_{j_\lambda} \in \mathcal{X}$ such that $d(u_{i_\lambda}) \neq d(u_{j_\lambda})$, which is not discerned by B. Consider the corresponding clause $f_\lambda = \left(\overline{i_\lambda} \vee \overline{j_\lambda} \vee \bigvee_{a\in A,a(u_{i_\lambda})\neq a(u_{j_\lambda})} \overline{a}\right)$. Let us as assign $false$ to the variables in f_λ and $true$ to all others. For such valuation, P is $true$ because it does not share any elements with f_λ. On the other hand, τ_{bi} is $false$. Thus, P is not an implicant for τ_{bi}.
Now, using Equation (4.4), we show the desired equivalence:

$$(\mathcal{X}, B) \text{ is a decision bireduct} \quad \iff \quad P \text{ is a prime implicant for } \tau_{bi} \tag{4.5}$$

(⇒) Suppose that P is not a prime implicant for τ_{bi}. We have two possibilities. The first one is that P is not an implicant – this is, however, immediately resolved by Equation (4.4). The second possibility is that P is an implicant but not a prime implicant. This means that we can remove at least one element of P while still preserving the property of being an implicant. There are two cases:

- A variable that can be removed from P corresponds to an object $u_{i_\lambda} \in U \setminus \mathcal{X}$. Then the Boolean formula $P' = \bigwedge_{u_i\in U\setminus(\mathcal{X}\cup\{u_{i_\lambda}\})} \bar{i} \wedge \bigwedge_{a\in B} \overline{a}$ is an implicant for τ_{bi}. From Equation (4.4) we know that $B \Rrightarrow_{\mathcal{X}\cup\{u_{i_\lambda}\}} d$ and therefore $(\mathcal{X}, B)$ is not a decision bireduct, since there exists a proper superset $\mathcal{X}\cup\{u_{i_\lambda}\} \supsetneq \mathcal{X}$ for which inexact functional dependency between attributes and decision holds.

- A variable that can be removed from P corresponds to an attribute $a_\lambda \in B$. Then the Boolean formula $P' = \bigwedge_{u_i\in U\setminus\mathcal{X}} \bar{i} \wedge \bigwedge_{a\in(B\setminus\{a_\lambda\})} \overline{a}$ is an implicant for τ_{bi}. From Equation (4.4) we know that $B \setminus \{a_\lambda\} \Rrightarrow_{\mathcal{X}} d$ and therefore $(\mathcal{X}, B)$ is not a decision bireduct, since there exists a proper subset $B \setminus \{a_\lambda\} \subsetneq B$ for which inexact functional dependency between attributes and decision holds.

(⇐) Suppose that $(\mathcal{X}, B)$ is not a decision bireduct for $\mathbb{A}$. We have three possibilities:

- There is $B \not\Rrightarrow_{\mathcal{X}} d$. Then from Equation (4.4) we know that $P = \bigwedge_{u_i\in U\setminus\mathcal{X}} \bar{i} \wedge \bigwedge_{a\in B} \overline{a}$ is not an implicant.

- There exists a proper subset $B' \subsetneq B$ such that $B' \Rrightarrow_{\mathcal{X}} d$. Then from Equation (4.4) we have that $P' = \bigwedge_{u_i\in U\setminus\mathcal{X}} \bar{i} \wedge \bigwedge_{a\in B'} \overline{a}$ is an implicant for τ_{bi}. Therefore, P is not a prime implicant.

- There exists a proper superset $\mathcal{X}' \supsetneq \mathcal{X}$ such that $B \Rrightarrow_{\mathcal{X}'} d$. Then from Equation (4.4) we know that $P' = \bigwedge_{u_i\in U\setminus\mathcal{X}'} \bar{i} \wedge \bigwedge_{a\in B} \overline{a}$ is an implicant for τ_{bi}. Therefore, P is not a prime implicant.

All the above steps together finish the proof. □

Proposition 20 illustrates that the number of decision bireducts is usually far higher than the number of standard decision reducts. For instance, as already mentioned, there are only two standard decision reducts for the decision table illustrated in Table 3.1. On the other hand, following the fact that a Boolean formula can be represented as a disjunction of all its prime implicants, we can represent the set of all decision bireducts for this particular example, using the following derivation. The initial CNF (conjunctive normal form) formula encodes the pairs of objects with different decision values (such as u_1 and u_3 that differ on attribute O, or u_1 and u_4 that differ on O and T). Whereas the target DNF (disjunctive normal form) formula encodes decision bireducts (such as decision bireduct with attributes $\{O,T,W\}$ and no uncovered objects – decoded to the actual bireduct $(U,\{O,T,W\})$, or decision bireduct with attributes $\{O,T\}$ and uncovered objects u_4,u_6,u_{10},u_{11} – thus decoding to the actual bireduct $(\{u_1,u_2,u_3,u_5,u_7,u_8,u_9,u_{12},u_{13},u_{14}\},\{O,T\})$). Complete CNF and DNF formulae representing decision bireducts for the data set presented in Table 3.1 may be seen in Table 4.4.

Proposition 20 shows a way to utilize techniques known from Boolean reasoning to search for decision bireducts as prime implicants [85]. It also illustrates that attributes and objects are to some extent equally important while constructing decision bireducts, analogously to some other approaches to deriving knowledge from data [35]. This intuition has led us to the observation that we can use various attribute reduction heuristics directly to the task of searching for decision bireducts, after an appropriate transformation of the input data:

Proposition 21. *Let $\mathbb{A}=(U,A\cup\{d\})$ be given. Consider a new decision table $\mathbb{A}^{\boxbslash}=(U,A\cup A^{\boxbslash}\cup\{d\})$, where the number of objects in U as well as their values for original attributes in $\mathbb{A}$ remain unchanged, and where the new attributes $A^{\boxbslash}=\{a_1^{\boxbslash},\dots,a_{|U|}^{\boxbslash}\}$ are defined as follows:*

$$a_j^{\boxbslash}(u_i)=\begin{cases}1 & \text{if } i=j\\ 0 & \text{otherwise}\end{cases}\tag{4.6}$$

Then a pair $(\mathcal{X},B)$, where $B\subseteq A$ and $\mathcal{X}\subseteq U$, is a decision bireduct for $\mathbb{A}$, if and only if $B\cup A_{\mathcal{X}}^{\boxbslash}$, for $A_{\mathcal{X}}^{\boxbslash}=\{a_i^{\boxbslash}\in A^{\boxbslash}:u_i\notin\mathcal{X}\}$, is a decision reduct for $\mathbb{A}^{\boxbslash}$.

Proof. ($\Rightarrow$) Let a decision bireduct $(\mathcal{X},B)$ for $\mathbb{A}$ be given. We need to show that for $B\cup A_{\mathcal{X}}^{\boxbslash}$, where $A_{\mathcal{X}}^{\boxbslash}$ is defined as above, the decision reduct conditions for $\mathbb{A}^{\boxbslash}$ are met. First, we know that B discerns all objects from $\mathcal{X}$ with different decision values and – because it is non-extendable – no other object from $U\setminus\mathcal{X}$ can be added. Therefore, to ensure discernibility among all the objects with different decision values in $\mathbb{A}^{\boxbslash}$, we need to add some of the attributes from $A^{\boxbslash}$. One can see that a given $a_i^{\boxbslash}\in A^{\boxbslash}$ discerns object u_i from all other objects. Thus, it is enough to take $B\cup A_{\mathcal{X}}^{\boxbslash}$ to ensure discernibility in $\mathbb{A}^{\boxbslash}$. Secondly, we have to show that $B\cup A_{\mathcal{X}}^{\boxbslash}$ is irreducible. Using the above reasoning, we cannot remove any attribute from $A_{\mathcal{X}}^{\boxbslash}$ because – otherwise – some of the objects with different decision values would become indiscernible in $\mathbb{A}^{\boxbslash}$. We also cannot remove any attribute from B. If we could, i.e., if there is $B'\subsetneq B$ such that $B'\cup A_{\mathcal{X}}^{\boxbslash}$ discerns all objects with different decision values in $\mathbb{A}^{\boxbslash}$, then we would have $B'\Rrightarrow_{\mathcal{X}} d$ and $(\mathcal{X},B)$ would not be a decision bireduct.
($\Leftarrow$) Let a decision reduct $B\cup A_{\mathcal{X}}^{\boxbslash}$ for $\mathbb{A}^{\boxbslash}$ be given. We need to show that $(\mathcal{X},B)$ is a decision bireduct in $\mathbb{A}$. Clearly, there is $B\Rrightarrow_{\mathcal{X}} d$. Suppose that B may be reduced, i.e., there is $B'\subsetneq B$ such that $B'\Rrightarrow_{\mathcal{X}} d$. However, in such a case $B'\cup A_{\mathcal{X}}^{\boxbslash}$ would discern all objects with different decision values in $\mathbb{A}^{\boxbslash}$, so $B\cup A_{\mathcal{X}}^{\boxbslash}$ would not be a decision reduct – a contradiction. Secondly, suppose that we can extend $\mathcal{X}$ by some object $u_i\in U\setminus\mathcal{X}$, i.e., there is $B\Rrightarrow_{\mathcal{X}\cup\{u_i\}} d$. However, it would imply that $B\cup A_{\mathcal{X}}^{\boxbslash}$ is not a decision reduct because we would be able to reduce it by removing the attribute $a_i^{\boxbslash}$ – a contradiction. □

An illustrative example of the considered transformation can be seen in Table 4.5. Certainly, it should be treated just as a starting point for developing more efficient algorithms, because decision tables of the form $\mathbb{A}^{\boxbslash}=(U,A\cup A^{\boxbslash}\cup\{d\})$ cannot be explicitly materialized for large data. On the other hand, this representation may be a kind of inspiration for adapting attribute clustering techniques developed for high dimensional data [54, 55] to the problem of searching for ensembles of decision bireducts.

Formulating a representation analogous to Proposition 21 for γ-decision bireducts is still a matter of further research. However, we have the following Boolean representation analogous to Proposition 20.

Table 4.4: CNF and DNF formulae which encode, respectively, all decision bireduct constraints for decision table presented in Table 3.1 and a disjunction of all prime implicants corresponding to all decision bireducts for that data set.

CNF
$\tau_{bi} = (\overline{1} \vee \overline{3} \vee \overline{O}) \wedge (\overline{1} \vee \overline{4} \vee \overline{O} \vee \overline{T}) \wedge (\overline{1} \vee \overline{5} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{1} \vee \overline{7} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge$ $(\overline{1} \vee \overline{9} \vee \overline{T} \vee \overline{H}) \wedge (\overline{1} \vee \overline{10} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{1} \vee \overline{11} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{1} \vee \overline{12} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge$ $(\overline{1} \vee \overline{13} \vee \overline{O} \vee \overline{H}) \wedge (\overline{2} \vee \overline{3} \vee \overline{O} \vee \overline{W}) \wedge (\overline{2} \vee \overline{4} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{2} \vee \overline{5} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge$ $(\overline{2} \vee \overline{7} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{2} \vee \overline{9} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{2} \vee \overline{10} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{2} \vee \overline{11} \vee \overline{T} \vee \overline{H}) \wedge$ $(\overline{2} \vee \overline{12} \vee \overline{O} \vee \overline{T}) \wedge (\overline{2} \vee \overline{13} \vee \overline{O} \vee \overline{H} \vee \overline{W}) \wedge (\overline{3} \vee \overline{6} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{3} \vee \overline{8} \vee \overline{O} \vee \overline{T}) \wedge$ $(\overline{3} \vee \overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{4} \vee \overline{6} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{4} \vee \overline{8} \vee \overline{O}) \wedge (\overline{4} \vee \overline{14} \vee \overline{W}) \wedge (\overline{5} \vee \overline{6} \vee \overline{W}) \wedge$ $(\overline{5} \vee \overline{8} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{5} \vee \overline{14} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{6} \vee \overline{7} \vee \overline{O}) \wedge (\overline{6} \vee \overline{9} \vee \overline{O} \vee \overline{W}) \wedge$ $(\overline{6} \vee \overline{10} \vee \overline{T} \vee \overline{W}) \wedge (\overline{6} \vee \overline{11} \vee \overline{O} \vee \overline{T}) \wedge (\overline{6} \vee \overline{12} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{6} \vee \overline{13} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge$ $(\overline{7} \vee \overline{8} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{7} \vee \overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{8} \vee \overline{9} \vee \overline{T} \vee \overline{H}) \wedge (\overline{8} \vee \overline{10} \vee \overline{O} \vee \overline{H}) \wedge$ $(\overline{8} \vee \overline{11} \vee \overline{H} \vee \overline{W}) \wedge (\overline{8} \vee \overline{12} \vee \overline{O} \vee \overline{W}) \wedge (\overline{8} \vee \overline{13} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{9} \vee \overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge$ $(\overline{10} \vee \overline{14} \vee \overline{H} \vee \overline{W}) \wedge (\overline{11} \vee \overline{14} \vee \overline{O} \vee \overline{H}) \wedge (\overline{12} \vee \overline{14} \vee \overline{O}) \wedge (\overline{13} \vee \overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W})$

DNF
$\tau_{bi} = (\overline{3} \wedge \overline{4} \wedge \overline{5} \wedge \overline{7} \wedge \overline{9} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13}) \vee (\overline{1} \wedge \overline{2} \wedge \overline{6} \wedge \overline{8} \wedge \overline{14}) \vee (\overline{H} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{12}) \vee$ $(\overline{H} \wedge \overline{1} \wedge \overline{2} \wedge \overline{5} \wedge \overline{7} \wedge \overline{8} \wedge \overline{9} \wedge \overline{10} \wedge \overline{11} \wedge \overline{13} \wedge \overline{14}) \vee (\overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{7} \wedge \overline{11} \wedge \overline{12}) \vee$ $(\overline{H} \wedge \overline{W} \wedge \overline{2} \wedge \overline{3} \wedge \overline{4} \wedge \overline{7} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{H} \wedge \overline{W} \wedge \overline{2} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{14}) \vee (\overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{6} \wedge \overline{8} \wedge \overline{12}) \vee$ $(\overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{2} \wedge \overline{7} \wedge \overline{8} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{6} \wedge \overline{9} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{4} \wedge \overline{5} \wedge \overline{9} \wedge \overline{10} \wedge \overline{11}) \vee$ $(\overline{O} \wedge \overline{1} \wedge \overline{2} \wedge \overline{4} \wedge \overline{5} \wedge \overline{8} \wedge \overline{10}) \vee (\overline{O} \wedge \overline{H} \wedge \overline{6} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{H} \wedge \overline{5} \wedge \overline{10} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{H} \wedge \overline{4} \wedge \overline{6}) \vee$ $(\overline{O} \wedge \overline{H} \wedge \overline{4} \wedge \overline{5} \wedge \overline{10}) \vee (\overline{O} \wedge \overline{H} \wedge \overline{W}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{6} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{6} \wedge \overline{8} \wedge \overline{14}) \vee$ $(\overline{O} \wedge \overline{T} \wedge \overline{5} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{5} \wedge \overline{8} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{4} \wedge \overline{6} \wedge \overline{10} \wedge \overline{11}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{4} \wedge \overline{6} \wedge \overline{8} \wedge \overline{10}) \vee$ $(\overline{O} \wedge \overline{T} \wedge \overline{4} \wedge \overline{5} \wedge \overline{10} \wedge \overline{11}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{4} \wedge \overline{5} \wedge \overline{8} \wedge \overline{10}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{H} \wedge \overline{5} \wedge \overline{14}) \vee$ $(\overline{O} \wedge \overline{T} \wedge \overline{H} \wedge \overline{4} \wedge \overline{5}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{W}) \vee (\overline{O} \wedge \overline{W} \wedge \overline{9} \wedge \overline{11}) \vee (\overline{O} \wedge \overline{W} \wedge \overline{2} \wedge \overline{9}) \vee (\overline{O} \wedge \overline{W} \wedge \overline{1} \wedge \overline{8} \wedge \overline{11}) \vee$ $(\overline{O} \wedge \overline{W} \wedge \overline{1} \wedge \overline{2} \wedge \overline{8}) \vee (\overline{T} \wedge \overline{3} \wedge \overline{6} \wedge \overline{8} \wedge \overline{13} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{3} \wedge \overline{5} \wedge \overline{7} \wedge \overline{8} \wedge \overline{9} \wedge \overline{13} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13}) \vee (\overline{T} \wedge \overline{1} \wedge \overline{2} \wedge \overline{5} \wedge \overline{7} \wedge \overline{8} \wedge \overline{9} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{1} \wedge \overline{2} \wedge \overline{4} \wedge \overline{6} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{1} \wedge \overline{2} \wedge \overline{4} \wedge \overline{5} \wedge \overline{7} \wedge \overline{9} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12}) \vee$ $(\overline{T} \wedge \overline{H} \wedge \overline{3} \wedge \overline{6} \wedge \overline{8} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{3} \wedge \overline{5} \wedge \overline{7} \wedge \overline{8} \wedge \overline{9} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{3} \wedge \overline{4} \wedge \overline{5} \wedge \overline{7} \wedge \overline{9} \wedge \overline{12}) \vee$ $(\overline{T} \wedge \overline{H} \wedge \overline{1} \wedge \overline{2} \wedge \overline{4} \wedge \overline{6} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{1} \wedge \overline{2} \wedge \overline{4} \wedge \overline{5} \wedge \overline{7} \wedge \overline{9} \wedge \overline{12}) \vee$ $(\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{7} \wedge \overline{8} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{7} \wedge \overline{8} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{6} \wedge \overline{8} \wedge \overline{12}) \vee$ $(\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{7} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{7} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{7} \wedge \overline{8} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{7} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{7} \wedge \overline{12}) \vee$ $(\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{6} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{6} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{3} \wedge \overline{7} \wedge \overline{8} \wedge \overline{13} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{W} \wedge \overline{3} \wedge \overline{7} \wedge \overline{8} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{3} \wedge \overline{6} \wedge \overline{8} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13}) \vee$ $(\overline{T} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{7} \wedge \overline{10} \wedge \overline{13} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{7} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13}) \vee$ $(\overline{T} \wedge \overline{W} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{10} \wedge \overline{13} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{7} \wedge \overline{8} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{6} \wedge \overline{8} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{6} \wedge \overline{8} \wedge \overline{11} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{7} \wedge \overline{10} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{7} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{6} \wedge \overline{10} \wedge \overline{14}) \vee$ $(\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{4} \wedge \overline{6} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12}) \vee (\overline{W} \wedge \overline{2} \wedge \overline{3} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{9} \wedge \overline{10} \wedge \overline{13} \wedge \overline{14}) \vee$ $(\overline{W} \wedge \overline{1} \wedge \overline{7} \wedge \overline{8} \wedge \overline{11} \wedge \overline{12})$

Table 4.5: $\mathbb{A}^{\boxbslash} = (U, A \cup A^{\boxbslash} \cup \{d\})$ which corresponds to $\mathbb{A} = (U, A \cup \{d\})$ in Table 3.1.

	Outlook	***Temp.***	***Humid.***	***Wind***	$a_1^{\boxbslash}$	$a_2^{\boxbslash}$	$a_3^{\boxbslash}$	$a_4^{\boxbslash}$	$a_5^{\boxbslash}$	$a_6^{\boxbslash}$	$a_7^{\boxbslash}$	$a_8^{\boxbslash}$	$a_9^{\boxbslash}$	$a_{10}^{\boxbslash}$	$a_{11}^{\boxbslash}$	$a_{12}^{\boxbslash}$	$a_{13}^{\boxbslash}$	$a_{14}^{\boxbslash}$	***Play***
1	sunny	hot	high	weak	1	0	0	0	0	0	0	0	0	0	0	0	0	0	no
2	sunny	hot	high	strong	0	1	0	0	0	0	0	0	0	0	0	0	0	0	no
3	overcast	hot	high	weak	0	0	1	0	0	0	0	0	0	0	0	0	0	0	yes
4	rain	mild	high	weak	0	0	0	1	0	0	0	0	0	0	0	0	0	0	yes
5	rain	cool	normal	weak	0	0	0	0	1	0	0	0	0	0	0	0	0	0	yes
6	rain	cool	normal	strong	0	0	0	0	0	1	0	0	0	0	0	0	0	0	no
7	overcast	cool	normal	strong	0	0	0	0	0	0	1	0	0	0	0	0	0	0	yes
8	sunny	mild	high	weak	0	0	0	0	0	0	0	1	0	0	0	0	0	0	no
9	sunny	cool	normal	weak	0	0	0	0	0	0	0	0	1	0	0	0	0	0	yes
10	rain	mild	normal	weak	0	0	0	0	0	0	0	0	0	1	0	0	0	0	yes
11	sunny	mild	normal	strong	0	0	0	0	0	0	0	0	0	0	1	0	0	0	yes
12	overcast	mild	high	strong	0	0	0	0	0	0	0	0	0	0	0	1	0	0	yes
13	overcast	hot	normal	weak	0	0	0	0	0	0	0	0	0	0	0	0	1	0	yes
14	rain	mild	high	strong	0	0	0	0	0	0	0	0	0	0	0	0	0	1	no

This also shows that the γ-decision bireducts are more restrictive than the standard decision bireducts, i.e., if there is no attribute selected that discerns given two objects, then none of those objects can be contained in a γ-decision bireduct.

Proposition 22. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Consider the following Boolean formula with propositional variables $\bar{i}$ for $i = 1, \ldots, |U|$, and $\bar{a}$ for $a \in A$:*

$$\tau_{bi}^{\gamma} = \bigwedge_{u_i \in U} \bigwedge_{u_j \in U : d(u_i) \neq d(u_j)} \left(\bar{i} \vee \bigvee_{a \in A : a(u_i) \neq a(u_j)} \bar{a} \right) \tag{4.7}$$

An arbitrary pair $(\mathcal{X}, B)$, where $\mathcal{X} \subseteq U$, $B \subseteq A$, is a γ-decision bireduct, if and only if the Boolean formula $\bigwedge_{u_i \in U \setminus \mathcal{X}} \bar{i} \wedge \bigwedge_{a \in B} \bar{a}$ is a prime implicant for τ_{bi}^{γ}.

Before we comment on the proof, let us mention about the following transformation of the above expression that can give more insight on γ-decision bireducts:

$$\begin{aligned} \tau_{bi}^{\gamma} &= \bigwedge_{u_i \in U} \bigwedge_{u_j \in U : d(u_i) \neq d(u_j)} \left(\bar{i} \vee \bigvee_{a \in A : a(u_i) \neq a(u_j)} \bar{a} \right) \\ &= \bigwedge_{u_i \in U} \Big(\underbrace{\bar{i}}_{u_i \notin \mathcal{X}} \vee \underbrace{\bigwedge_{u_j \in U : d(u_i) \neq d(u_j)} \bigvee_{a \in A : a(u_i) \neq a(u_j)} \bar{a}}_{u_i \in POS(A)} \Big) \end{aligned} \tag{4.8}$$

This means that, as we are interested in prime implicants, either an object belongs to the positive region $POS(A)$ and then it belongs to a γ-decision bireduct or, in other case, it is definitively excluded from a γ-decision bireduct.

Proof. The proof is fully analogous to that presented for Proposition 20. At the first step, analogously to the proof of Equation (4.4), the following can be shown for $P^{\gamma} = \bigwedge_{u_i \in U \setminus \mathcal{X}} \bar{i} \wedge \bigwedge_{a \in B} \bar{a}$:

$$B \Rrightarrow_{\mathcal{X}}^{\gamma} d \quad \Longleftrightarrow \quad P^{\gamma} \text{ is an implicant for } \tau_{bi}^{\gamma} \tag{4.9}$$

At the second step, analogously to the proof of Equation (4.5), one can show that:

$$(\mathcal{X}, B) \text{ is a } \gamma\text{-decision bireduct} \quad \Longleftrightarrow \quad P^{\gamma} \text{ is a prime implicant for } \tau_{bi}^{\gamma} \tag{4.10}$$

□

Complete CNF and DNF formulae representing γ-decision bireducts for the data set presented in Table 3.1 may be seen in Table 4.6. Compared to the results presented in Table 4.4, the CNF formula for γ-decision bireducts is longer and the DNF formula is shorter than their counterparts for decision bireducts. The comparison of the formulae's lengths can strengthen the intuition that because γ-decision bireducts are more restricted by the established constraints there are fewer of them than decision bireducts.

Table 4.6: CNF and DNF formulae which encode, respectively, all γ-decision bireduct constraints for decision table presented in Table 3.1 and a disjunction of all prime implicants corresponding to all γ-decision bireducts for that data set.

CNF
$\tau_{bi}^{\gamma} = (\overline{1} \vee \overline{O}) \wedge (\overline{1} \vee \overline{O} \vee \overline{T}) \wedge (\overline{1} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{1} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{1} \vee \overline{T} \vee \overline{H}) \wedge (\overline{1} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{1} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{1} \vee \overline{O} \vee \overline{H}) \wedge (\overline{2} \vee \overline{O} \vee \overline{W}) \wedge (\overline{2} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{2} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{2} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{2} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{2} \vee \overline{T} \vee \overline{H}) \wedge (\overline{2} \vee \overline{O} \vee \overline{T}) \wedge (\overline{2} \vee \overline{O} \vee \overline{H} \vee \overline{W}) \wedge (\overline{3} \vee \overline{O}) \wedge (\overline{3} \vee \overline{O} \vee \overline{W}) \wedge (\overline{3} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{3} \vee \overline{O} \vee \overline{T}) \wedge (\overline{3} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{4} \vee \overline{O} \vee \overline{T}) \wedge (\overline{4} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{4} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{4} \vee \overline{O}) \wedge (\overline{4} \vee \overline{W}) \wedge (\overline{5} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{5} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{5} \vee \overline{W}) \wedge (\overline{5} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{6} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{6} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{6} \vee \overline{W}) \wedge (\overline{6} \vee \overline{O}) \wedge (\overline{6} \vee \overline{O} \vee \overline{W}) \wedge (\overline{6} \vee \overline{T} \vee \overline{W}) \wedge (\overline{6} \vee \overline{O} \vee \overline{T}) \wedge (\overline{6} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{6} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{7} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{7} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{7} \vee \overline{O}) \wedge (\overline{8} \vee \overline{O} \vee \overline{T}) \wedge (\overline{8} \vee \overline{O}) \wedge (\overline{8} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{8} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{8} \vee \overline{T} \vee \overline{H}) \wedge (\overline{8} \vee \overline{O} \vee \overline{H}) \wedge (\overline{8} \vee \overline{H} \vee \overline{W}) \wedge (\overline{8} \vee \overline{O} \vee \overline{W}) \wedge (\overline{9} \vee \overline{T} \vee \overline{H}) \wedge (\overline{9} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{9} \vee \overline{O} \vee \overline{W}) \wedge (\overline{9} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{10} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{10} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{10} \vee \overline{T} \vee \overline{W}) \wedge (\overline{10} \vee \overline{O} \vee \overline{H}) \wedge (\overline{10} \vee \overline{H} \vee \overline{W}) \wedge (\overline{11} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{11} \vee \overline{T} \vee \overline{H}) \wedge (\overline{11} \vee \overline{O} \vee \overline{T}) \wedge (\overline{11} \vee \overline{H} \vee \overline{W}) \wedge (\overline{11} \vee \overline{O} \vee \overline{H}) \wedge (\overline{12} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{12} \vee \overline{O} \vee \overline{T}) \wedge (\overline{12} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{12} \vee \overline{O} \vee \overline{W}) \wedge (\overline{12} \vee \overline{O}) \wedge (\overline{13} \vee \overline{O} \vee \overline{H}) \wedge (\overline{13} \vee \overline{O} \vee \overline{H} \vee \overline{W}) \wedge (\overline{13} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{13} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{13} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{W}) \wedge (\overline{14} \vee \overline{W}) \wedge (\overline{14} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{H}) \wedge (\overline{14} \vee \overline{O} \vee \overline{T} \vee \overline{H} \vee \overline{W}) \wedge (\overline{14} \vee \overline{H} \vee \overline{W}) \wedge (\overline{14} \vee \overline{O} \vee \overline{H}) \wedge (\overline{14} \vee \overline{O})$

DNF
$\tau_{bi}^{\gamma} = (\overline{1} \wedge \overline{2} \wedge \overline{3} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{7} \wedge \overline{8} \wedge \overline{9} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13} \wedge \overline{14}) \vee (\overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{2} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{7} \wedge \overline{8} \wedge \overline{11} \wedge \overline{12} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{1} \wedge \overline{2} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{8} \wedge \overline{9} \wedge \overline{10} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{H} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{10} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{H} \wedge \overline{W}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{8} \wedge \overline{10} \wedge \overline{11} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{H} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{14}) \vee (\overline{O} \wedge \overline{T} \wedge \overline{W}) \vee (\overline{O} \wedge \overline{W} \wedge \overline{1} \wedge \overline{2} \wedge \overline{8} \wedge \overline{9} \wedge \overline{11}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{1} \wedge \overline{2} \wedge \overline{3} \wedge \overline{4} \wedge \overline{5} \wedge \overline{6} \wedge \overline{7} \wedge \overline{8} \wedge \overline{9} \wedge \overline{12} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{H} \wedge \overline{W} \wedge \overline{1} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{7} \wedge \overline{8} \wedge \overline{12} \wedge \overline{14}) \vee (\overline{T} \wedge \overline{W} \wedge \overline{1} \wedge \overline{3} \wedge \overline{4} \wedge \overline{6} \wedge \overline{7} \wedge \overline{8} \wedge \overline{10} \wedge \overline{11} \wedge \overline{12} \wedge \overline{13} \wedge \overline{14})$

4.4 Decision Bireduct Optimization

Some questions arise as to what is the best or optimal decision bireduct, or when we can say that one decision bireduct is better than another. Following the idea from [140] it may be convenient to describe decision bireducts in terms of their attributes and uncovered objects. An implicit assumption is that decision bireducts shall minimize both those factors. Minimizing the size of the attribute subset is quite intuitive as to its analogy with minimizing the size of decision reducts. The smaller the size, the more general is the description of the decision table. The minimization of the size of the uncovered set of objects is also an intuitive tendency to minimize the number of objects that do not fit to (or disturb) the description of the decision table contained in a decision bireduct. The minimization task translates into maximization of the size of the objects contained in a decision bireduct. In case of imbalanced data sets (with a large disproportion between a number of objects with a certain decision value) the simplest form of measuring the size based on the cardinality of the subset of objects may be insufficient. In such a case one should pay more attention to a specified subset of objects, e.g., objects belonging to the minority decision classes.

There are a number of NP-hardness results related to extracting optimal decision reducts and approximate reducts from data [82]. In the case of decision bireducts, one may think about quite different optimization criteria with respect to a balance between the number of involved attributes and objects. For example, in [140] the following function was minimized while evaluating decision bireducts obtained from randomly generated permutations using Algorithm 1 (let us however emphasize that this is just one of many possible ways of interpreting optimal decision bireducts):

$$\mathit{DescLength}((\mathcal{X}, B)) = \frac{|B|}{|A|} + \frac{|U \setminus \mathcal{X}|}{|U|} \tag{4.11}$$

The description length of a decision bireduct can be interpreted as the length of the Boolean formula representing the corresponding prime implicant by means of the number of attributes in B and the number of objects outside $\mathcal{X}$, additionally normalized by $|A|$ and $|U|$, respectively. Yet another aspect of evaluation of decision bireducts may refer to their ensembles, where particular decision bireducts are supposed to help each other while covering the whole U by the supports of decision rules that they correspond to. We refer to the Section 4.5 for more details about this approach.

Regardless of the fact whether we use a single decision bireduct or their bigger ensemble, one can formulate also some independent constraints guaranteeing, e.g., that each considered decision bireduct cannot yield too many uncovered objects. Such form of a constraint for decision bireducts is somewhat analogous to those studied for frequent item sets and patterns [87]. In some sense, it is related to considering subsets of attributes that almost satisfy the constraints of decision reducts which makes it analogous to constraints formulated for approximate decision reducts in Section 3.3. Such subsets can be represented within the space of decision bireducts as well.

Definition 36 (decision ε-bireduct)**.** *Let $\varepsilon \in [0, 1)$ be given. We say that a pair $(\mathcal{X}, B)$, where $\mathcal{X} \subseteq U$ and $B \subseteq A$, is a decision ε-bireduct, if it is a decision bireduct and the following property holds:*

$$|\mathcal{X}| \geq (1 - \varepsilon)|U| \tag{4.12}$$

An analogous definition for γ-decision ε-bireduct can be also formulated as follows:

Definition 37 (γ-decision ε-bireduct)**.** *Let $\varepsilon \in [0, 1)$ be given. We say that a pair $(\mathcal{X}, B)$, where $B \subseteq A$ and $\mathcal{X} \subseteq U$, is a γ-decision ε-bireduct, if it is a γ-decision bireduct and the following holds:*

$$|\mathcal{X}| \geq (1 - \varepsilon)|U| \tag{4.13}$$

One can expect that a way of investigating the complexity of searching for decision ε-bireducts and γ-decision ε-bireducts should have something in common with the way of approaching approximate decision reducts. For γ-decision ε-bireducts it is obvious due to the following fact:

Proposition 23. *Let $\varepsilon \in [0, 1)$ be given. The following statements are true:*

1. *For every $\mathbb{A} = (U, A \cup \{d\})$, $\mathcal{X} \subseteq U$ and $B \subseteq A$, the pair $(\mathcal{X}, B)$ is a γ-decision ε-bireduct, if and only if $\mathcal{X} = POS(B)$ and B is a γ-decision ε-reduct,*

2. *The problem of finding a γ-decision ε-bireduct with the minimum number of attributes is NP-hard.*

Proof. Both above items are straightforward conclusions from the previous propositions. □

For decision ε-bireducts the problem is not so trivial, although the following connections seem to be quite intuitive as well. Such connections were first studied in [117] by means of correspondence between decision ε-bireducts and M-decision ε-reducts for function $M : 2^A \to [0, 1]$ introduced in Definition 27. The fact below is a kind of reformulation of the previously-known results but generally it follows the same path as presented in [117].

Proposition 24. *Let $\mathbb{A} = (U, A \cup \{d\})$ and $B \subseteq A$ be given. B is the smallest M-decision ε-reduct in $\mathbb{A}$, if and only if there exists subset $\mathcal{X} \subseteq U$ such that the pair $(\mathcal{X}, B)$ is a decision ε-bireduct and there are no other decision ε-bireducts with less attributes than the cardinality of B.*

Proof. **(⇒)** Suppose that $B \subseteq A$ is the smallest M-decision ε-reduct in $\mathbb{A}$. In Definition 27 the majority function $M : 2^A \to [0, 1]$ is defined as a sum of fractions of the number of objects with the most frequent decision to the number of all objects within the equivalence classes induced by a subset of attributes. From Definition 26 we know that $M(B) \geq 1 - \varepsilon$. Hence, if we put $\mathcal{X}$ equals to the union of the objects with the most frequent decision within each indiscernibility class induced by B, then $(\mathcal{X}, B)$ is a decision ε-bireduct. This is because B cannot be reduced in the context of $\mathcal{X}$– otherwise it could be also reduced in the context of the M-decision ε-reduct. Also, $\mathcal{X}$ cannot be extended – otherwise, if $U \setminus \mathcal{X}$ is not empty, then extending $\mathcal{X}$ by any $u \in U \setminus \mathcal{X}$ would violate the decision bireduct condition for $(\mathcal{X} \cup \{u\}, B)$, as u has a different decision than objects from $\mathcal{X}$ that belong to the same indiscernibility class induced by B. Let us continue the reasoning by a contradiction. Suppose that there exists a decision ε-bireduct$(\mathcal{X}', B')$ for which $|B'| < |B|$. Let us consider the indiscernibility classes induced by B' and the given $\mathcal{X}'$. Having no knowledge of the arrangement of the object belonging to $\mathcal{X}'$ within the indiscernibility classes, we know that $|\mathcal{X}'| \geq (1 - \varepsilon)|U|$. If we replace the objects having not the most common decision with those having the most frequent value within each indiscernibility class, then the condition will also hold. This gives us that B' is a M-decision ε-reduct that contains a smaller number of attributes than B – a contradiction. **(⇐)** Suppose that $(\mathcal{X}, B)$ for $\mathcal{X} \subseteq U$ and $B \subseteq A$ is a decision ε-bireduct and there are no other decision ε-bireducts with less attributes than the cardinality of B. Using the same line of reasoning as above we know that B is a M-decision ε-reduct in $\mathbb{A}$. However, suppose that it is not the smallest one and let B' be a M-decision ε-reduct for which $|B'| < |B|$. Using the argumentation from the beginning of the proof, we can show that there exists $\mathcal{X}'$ such that $(\mathcal{X}', B')$ is a decision ε-bireduct. This leads to a contradiction as $(\mathcal{X}', B')$ is a decision ε-bireduct with less attributes than in B. □

In [117], some relationships between decision ε-bireducts and R-decision ε-reducts, for function $R : 2^A \to [0, 1]$ introduced in Definition 28, were considered as well. In particular, it turned out that the subsets of attributes $B \subseteq A$ in the outcomes $(\mathcal{X}, B)$ of the decision bireduct search algorithms discussed in Section 5.1 are likely to correspond to R-decision ε-reducts, for $\varepsilon \approxeq 1 - R(B)/R(A)$, if – along the iterations of the algorithm – objects belonging to less frequent decision classes tend to be added to subsets $\mathcal{X} \subseteq U$ relatively earlier than in the case of more frequent decisions. In this subsection, let us however focus on studying relationships between decision ε-bireducts and M-decision ε-reducts. Let us start by formulating the following simple corollary of Proposition 24.

Given the computational complexity results for approximate decision reducts from Section 3.3, we are now ready to formulate an analogous result for decision ε-bireducts:

Definition 38 (minimal decision ε-bireduct problem)**.** *Let $\varepsilon \in [0, 1)$ be given. By the Minimal Decision ε-Bireduct Problem (MDεBP) we mean a task of finding for a given decision table $\mathbb{A} = (U, A \cup \{d\})$ a decision ε-bireduct $(\mathcal{X}, B)$ with the lowest cardinality of B.*

Proposition 25. *For any $\varepsilon \in [0, 1)$, the Minimal Decision ε-Bireduct Problem is NP-hard.*

Proof. In Proposition 10, we have stated that, for each $\varepsilon \in [0, 1)$ treated as a constant, the problem of finding a minimal M-decision ε-reduct for an input decision table is NP-hard. Thus, we can propose a straightforward reduction, where decision bireducts which are solutions for the considered optimization problem yield the smallest M-decision ε-reducts for the same data sets. Namely, suppose that a pair

$(\mathcal{X}, B)$ is a decision ε-bireduct with the lowest cardinality of B for a given table $\mathbb{A} = (U, A \cup \{d\})$. Then, according to Proposition 24, the same B needs to be the smallest M-decision ε-reduct for the same $\mathbb{A}$. □

To prove Proposition 25, we linked M-decision ε-reducts $B \subseteq A$ with potentially most strongly supported (by means of $|\mathcal{X}|$) decision bireducts $(\mathcal{X}, B)$. This linkage was needed to set up the polynomial reduction of the considered optimization problems. On the other hand, in practice, it is not always the best idea to maximize $|\mathcal{X}|$. It may turn out that some collections of decision ε-bireducts $(\mathcal{X}, B)$ with relatively lower $|\mathcal{X}|$ – but still satisfying the constraint specified in Inequality 4.12 for a predefined $\varepsilon \in [0, 1)$ – will better "cooperate" with each other. In [140], as it can be seen also in the next subsection, we were trying to express such cooperation by searching for ensembles of diversified decision bireducts $(\mathcal{X}, B)$, with possibly different subsets $\mathcal{X} \subseteq U$ and $B \subseteq A$. Such a diversity can be considered for both decision bireducts and γ-decision bireducts.

Let us consider an example based on the data set in Table 3.1 and focus on decision ε-bireducts. There are 9 objects with the decision value 'yes' and 5 objects with decision value 'no'. Without performing any computations we can conclude that for $\varepsilon \geq \frac{5}{14}$ there will be only one M-decision ε-reduct, i.e., an empty subset of attributes. For such an M-decision ε-reduct, in the decision rules generation procedure, only one default rule pointing at the decision 'yes' would be generated. From the ensembles' point of view, a smaller value $\varepsilon = \frac{4}{14}$ for which the set of all M-decision ε-reducts is no longer a singleton, would be far more interesting. A complete list of M-decision ε-reducts and decision ε-bireducts for the chosen $\varepsilon = \frac{4}{14}$ is presented in Table 4.7. We can observe that the number of generated decision ε-bireducts is much greater than for M-decision ε-reducts.

Table 4.7: A complete list of M-decision ε-reducts and decision ε-bireducts in Table 3.1 for $\varepsilon = \frac{4}{14}$. It is presented in a tabular form where each row corresponds to a subset of attributes $B \subseteq A$. The values in the second column indicate if the corresponding M-decision ε-reduct exists. Non-existence comes from two reasons, either the value of $M(B)$ is insufficient for $\varepsilon = \frac{4}{14}$ (e.g., $B = \emptyset$ or $B = \{T\}$) or the subset B can be reduced (e.g., $B = \{O, T\}$ where $B = \{O\}$ is already a M-decision ε-reduct). The third column contain subsets of objects $\mathcal{X}$ such that $(\mathcal{X}, B)$ form decision ε-bireducts. Comparing with the list of all decision bireducts presented in Table 4.2 one can notice that here we have only those with coverage $|\mathcal{X}| \geq (1 - \varepsilon)|U| = 10$, cf. Definition 36.

$\boldsymbol{B \subseteq A}$	**is $\boldsymbol{M}$-decision $\boldsymbol{\varepsilon}$-reduct**	**value of $\mathcal{X}$ for decision $\boldsymbol{\varepsilon}$-bireducts $(\boldsymbol{\mathcal{X}, B})$**
$\emptyset$	no	NA
$\{O\}$	yes	$\llbracket 1..5, 7, 8, 10, 12, 13 \rrbracket$
$\{T\}$	no	NA
$\{H\}$	yes	$\llbracket 1, 2, 5, 7..11, 13, 14 \rrbracket$
$\{W\}$	no	NA
$\{O, T\}$	no	$\llbracket 1..5, 7..10, 12, 13 \rrbracket$, $\llbracket 1..5, 7, 9..13 \rrbracket$, $\llbracket 1..4, 6..10, 12, 13 \rrbracket$, $\llbracket 1..4, 6, 7, 9..13 \rrbracket$, $\llbracket 1..3, 5, 7..9, 12..14 \rrbracket$, $\llbracket 1..3, 5, 7, 9, 11..14 \rrbracket$, $\llbracket 1..3, 6..9, 12..14 \rrbracket$, $\llbracket 1..3, 6, 7, 9, 11..14 \rrbracket$
$\{O, H\}$	no	$\llbracket 1..5, 7..13 \rrbracket$, $\llbracket 1..4, 6..9, 11..13 \rrbracket$, $\llbracket 1..3, 5, 7..14 \rrbracket$, $\llbracket 1..3, 6..9, 11..14 \rrbracket$
$\{O, W\}$	no	$\llbracket 1..8, 10, 12..14 \rrbracket$, $\llbracket 1, 3..8, 10..14 \rrbracket$, $\llbracket 2..7, 9, 10, 12..14 \rrbracket$, $\llbracket 3..7, 9..14 \rrbracket$
$\{T, H\}$	no	$\llbracket 1, 2, 4, 5, 7, 9..13 \rrbracket$
$\{T, W\}$	yes	$\llbracket 2..6, 9..13 \rrbracket$, $\llbracket 2..5, 7, 9..13 \rrbracket$
$\{H, W\}$	no	$\llbracket 2..5, 7, 9..11, 13, 14 \rrbracket$
$\{O, T, H\}$	no	$\llbracket 1..4, 6..13 \rrbracket$, $\llbracket 1..3, 6..14 \rrbracket$
$\{O, T, W\}$	no	$\llbracket 1..14 \rrbracket$
$\{O, H, W\}$	no	$\llbracket 1..14 \rrbracket$
$\{T, H, W\}$	no	$\llbracket 1, 2, 4..6, 9..13 \rrbracket$, $\llbracket 1, 2, 4..6, 9..11, 13, 14 \rrbracket$, $\llbracket 1, 2, 4, 5, 7, 9..11, 13, 14 \rrbracket$, $\llbracket 1, 2, 5, 6, 8..13 \rrbracket$, $\llbracket 1, 2, 5, 6, 8..11, 13, 14 \rrbracket$, $\llbracket 1, 2, 5, 7..13 \rrbracket$, $\llbracket 2..6, 9..11, 13, 14 \rrbracket$, $\llbracket 2, 3, 5, 6, 8..13 \rrbracket$, $\llbracket 2, 3, 5, 6, 8..11, 13, 14 \rrbracket$, $\llbracket 2, 3, 5, 7..13 \rrbracket$, $\llbracket 2, 3, 5, 7..11, 13, 14 \rrbracket$
$\{O, T, H, W\}$	no	NA

4.5 Decision Bireduct Ensembles

The example outlined in Table 4.7 can give us an intuition that decision ε-bireducts may be more flexible in construction of classifier ensembles than approximate decision reducts. In particular, such ensembles would involve a smaller number of attributes while giving simpler and more general models. In our example, for $\varepsilon = \frac{4}{14}$, the situation is even more intuitive. If we consider a task of searching for an ensemble consisting of three components such that each object from the data set is covered by at least two of them (i.e., any object from the training data is properly classified by at least two of the classifiers), then it would be impossible to obtain such an ensemble for M-decision ε-reducts. However, for decision ε-bireducts it can be accomplished in 637 (computed using a script) different ways. Several examples are shown in Table 4.8.

In the above example, the considered ensemble of three decision ε-bireducts could enable valid classification of the known objects and – because of simplicity of the corresponding decision rules – could provide good classification accuracy also for the previously unseen cases. The validity of classification on the training data is guaranteed by the fact that, in a simple voting, two out of three rule based classifiers would be always correct while only one of them would make a mistake, if a given object is out of the corresponding $\mathcal{X} \subseteq U$.

Surely, there is also a question how to automatically extract such ensembles of decision ε-bireducts from the data. Actually, a direct control over the approximation threshold $\varepsilon \in [0, 1)$ in the algorithms presented in Section 4.3 would require a deeper thought even for a single decision ε-bireducts. However, even though the *ratio* parameter introduced in [140] does not allow to generate a decision ε-bireducts with a specific value of ε, it can be helpful in guiding the computation process toward bireducts with desired characteristics. The meaning of the *ratio* parameter, as well as its correlation with the support of generated decision bireducts is further investigated in Section 5.4.

Figure 4.1 displays some decision bireducts derived for decision table $\mathbb{A} = (U, A \cup \{d\})$ with objects $U = \{u_1, \ldots, u_6\}$ and conditional attributes $A = \{a_1, a_2, a_3\}$. For instance, consider the pair $B = \{a_1, a_2\}$ and $\mathcal{X} = \{u_2, \ldots, u_6\}$. It corresponds to rules "if $a_1 = 0$ and $a_2 = 0$ then $d = 0$" (supported by u_2), "if $a_1 = 0$ and $a_2 = 1$ then $d = 0$" (supported by u_3), "if $a_1 = 1$ and $a_2 = 0$ then $d = 1$" (supported by u_4 and u_5) and "if $a_1 = 1$ and $a_2 = 1$ then $d = 0$" (supported by u_6). Neither a_1 nor a_2 would be sufficient by itself to cover $\mathcal{X}$ with shorter rules. Moreover, u_1 cannot be added to $\mathcal{X}$ because it is inconsistent with the first rule.

The first algorithms aimed at deriving decision bireducts from the data were proposed in [140]. They were based on random generation of mixed orderings of attributes and objects – they will be presented also later in the dissertation in Section 5.1. Such orderings were utilized to encode sequences of attempts to remove attributes from B (starting with $B = A$) and add objects to $\mathcal{X}$ (starting with $\mathcal{X} = \emptyset$) in order to obtain pairs $(\mathcal{X}, B)$ such that $B \Rrightarrow_{\mathcal{X}} d$, with possibly minimal B and maximal $\mathcal{X}$. By using an appropriate process of generation of families of diverse orderings, one could derive collections of bireducts with quite different subsets of attributes and objects involved.

The notion of a decision bireduct allows us to operate with subsets of conditional attributes treated as classification descriptions, and with the associated subsets of objects for which those descriptions are valid. This gives us an elegant way to investigate complementarity of bireducts interpreted as classifiers in the ensemble. For instance, the following formulation expresses the idea of majority voting between ensemble components which – if properly tuned on the training data – gives us a chance of efficient performance over new cases too.

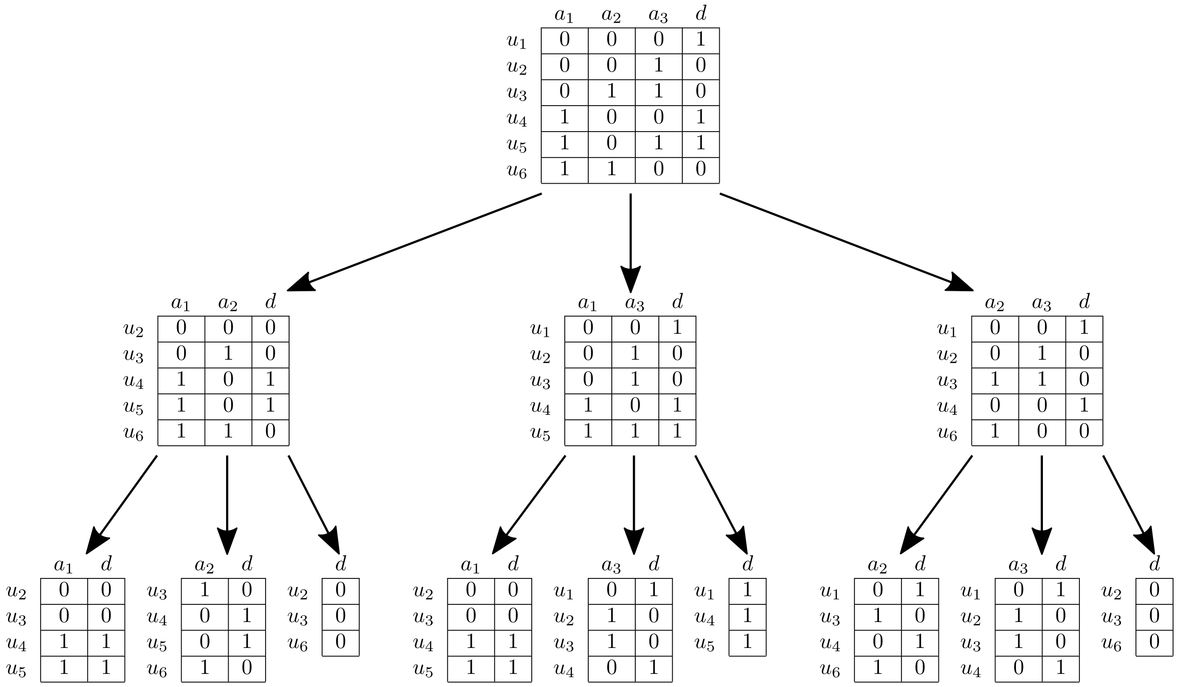
Definition 39 (correct decision bireduct ensemble)**.** *Let $\mathbb{A} = (U, A \cup \{d\})$ and the ensemble of decision bireducts $\mathcal{B} = \{(\mathcal{X}_1, B_1), \ldots, (\mathcal{X}_m, B_m)\}$ be given. We say that $\mathcal{B}$ is correct if and only if*

$$\underset{u \in U}{\forall} |\{i \in \{1, \ldots, m\} : u \in \mathcal{X}_i\}| > \frac{m}{2} \tag{4.14}$$

The above Inequality 4.14 means that more than 50% of decision rules triggered for $u \in U$ point at the valid decision $d(u)$. Figure 4.1 illustrates a kind of hierarchy of correct bireduct ensembles for

Table 4.8: Several examples of decision ε-bireduct 3-element ensembles. The listed ensembles cover all objects from the data set by at least two of its elements. Noticeable is the fact that such a covering is impossible to obtain using M-decision ε-reducts with the same value of ε. All ensemble elements used in the example are decision ε-bireducts obtained for $\varepsilon = \frac{4}{14}$, cf. Table 4.7.

Decision ε-bireduct ensemble	**Covering**														
$([\![1,2,5,7..11,13,14]\!],\{H\})$ $([\![1..4,6..9,11..13]\!],\{O,H\})$ $([\![1..8,10,12..14]\!],\{O,W\})$	object:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	count:	3	3	2	2	2	2	3	3	2	2	2	2	3	2
$([\![1,2,5,7..11,13,14]\!],\{H\})$ $([\![1..14]\!],\{O,T,W\})$ $([\![2..6,9..13]\!],\{T,W\})$	object:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	count:	2	3	2	2	3	2	2	2	3	3	3	2	3	2
$([\![1..5,7,8,10,12,13]\!],\{O\})$ $([\![1..3,6..9,11..14]\!],\{O,H\})$ $([\![3..7,9..14]\!],\{O,W\})$	object:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	count:	2	2	3	2	2	2	3	2	2	2	2	3	3	2
$([\![2..5,7,9..11,13,14]\!],\{H,W\})$ $([\![1..4,6..9,11..13]\!],\{O,H\})$ $([\![1..8,10,12..14]\!],\{O,W\})$	object:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	count:	2	3	3	3	2	2	3	2	2	2	2	2	3	2
$([\![1..5,7,8,10,12,13]\!],\{O\})$ $([\![1..14]\!],\{O,T,W\})$ $([\![2,3,5,6,8..11,13,14]\!],\{T,H,W\})$	object:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	count:	2	3	3	2	3	2	2	3	2	3	2	2	3	2
$([\![1..3,5,7..9,12..14]\!],\{O,T\})$ $([\![1,3..8,10..14]\!],\{O,W\})$ $([\![2..6,9..13]\!],\{T,W\})$	object:	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	count:	2	2	3	2	3	2	2	2	2	2	2	3	3	2



	a_1	a_2	a_3	d
u_1	0	0	0	1
u_2	0	0	1	0
u_3	0	1	1	0
u_4	1	0	0	1
u_5	1	0	1	1
u_6	1	1	0	0

	a_1	a_2	d
u_2	0	0	0
u_3	0	1	0
u_4	1	0	1
u_5	1	0	1
u_6	1	1	0

	a_1	a_3	d
u_1	0	0	1
u_2	0	1	0
u_3	0	1	0
u_4	1	0	1
u_5	1	1	1

	a_2	a_3	d
u_1	0	0	1
u_2	0	1	0
u_3	1	1	0
u_4	0	0	1
u_6	1	0	0

	a_1	d
u_2	0	0
u_3	0	0
u_4	1	1
u_5	1	1

	a_2	d
u_3	1	0
u_4	0	1
u_5	0	1
u_6	1	0

	d
u_2	0
u_3	0
u_6	0

	a_1	d
u_2	0	0
u_3	0	0
u_4	1	1
u_5	1	1

	a_3	d
u_1	0	1
u_2	1	0
u_3	1	0
u_4	0	1

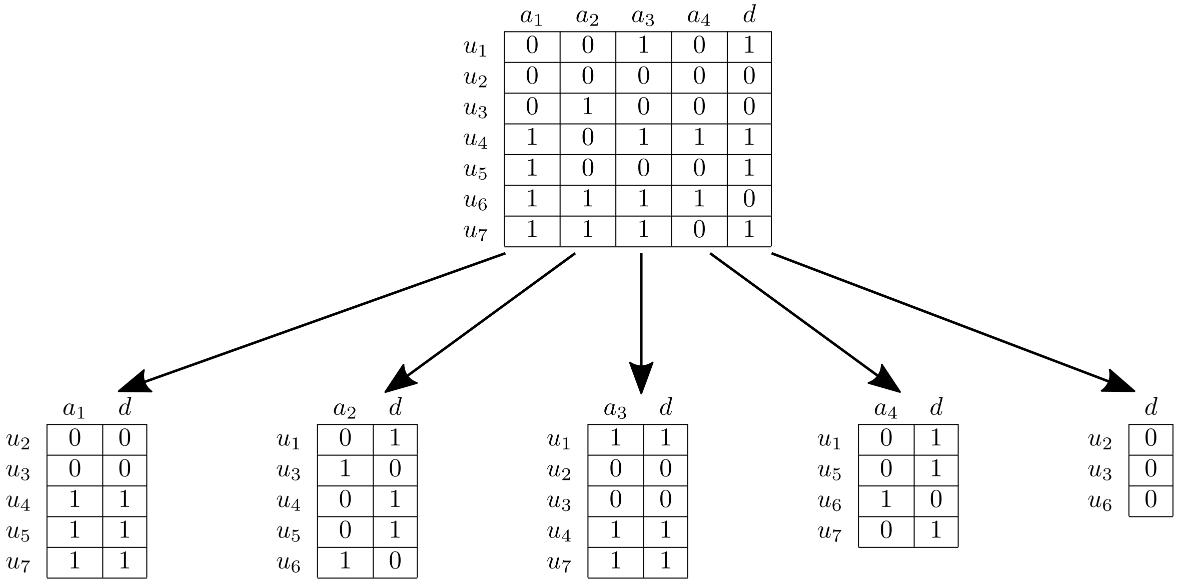
	d
u_1	1
u_4	1
u_5	1

	a_2	d
u_1	0	1
u_3	1	0
u_4	0	1
u_6	1	0

	a_3	d
u_1	0	1
u_2	1	0
u_3	1	0
u_4	0	1

	d
u_2	0
u_3	0
u_6	0

Figure 4.1: Examples of decision bireducts for table $\mathbb{A} = (\{u_1, \ldots, u_6\}, \{a_1, a_2, a_3\} \cup \{d\})$. Bireducts in the middle layer form a correct ensemble (each object is validly classified by at least two bireducts out of three). For each “middle” bireduct treated as a new decision table, its corresponding correct ensemble is provided in the lowest layer.



	a_1	a_2	a_3	a_4	d
u_1	0	0	1	0	1
u_2	0	0	0	0	0
u_3	0	1	0	0	0
u_4	1	0	1	1	1
u_5	1	0	0	0	1
u_6	1	1	1	1	0
u_7	1	1	1	0	1

	a_1	d
u_2	0	0
u_3	0	0
u_4	1	1
u_5	1	1
u_7	1	1

	a_2	d
u_1	0	1
u_3	1	0
u_4	0	1
u_5	0	1
u_6	1	0

	a_3	d
u_1	1	1
u_2	0	0
u_3	0	0
u_4	1	1
u_7	1	1

	a_4	d
u_1	0	1
u_5	0	1
u_6	1	0
u_7	0	1

	d
u_2	0
u_3	0
u_6	0

Figure 4.2: Examples of decision bireducts for table $\mathbb{A} = (\{u_1, \ldots, u_6\}, \{a_1, a_2, a_3\} \cup \{d\})$. Bireducts in the lower layer form a correct ensemble (each object is validly classified by at least three bireducts out of five).

$m = 3$. Alternatively, one can work with a "flat" collection of decision bireducts that are supposed to vote correctly on each of objects, even if some single bireducts are wrong for some single cases, cf., Figure 4.2 for $m = 5$.

Both, Figure 4.1 and Figure 4.2, implicitly suggest a top-down way of constructing correct ensembles, whereby each of m bireducts is derived in the same time, with an option of further decompositions into even smaller pieces. Such algorithms have been already considered in [133] for another type of (bi)reducts, i.e., so-called generalized decision reducts. On the other hand, one can proceed with the aforementioned ordering-based methods [140], whereby – somewhat reflecting the mechanisms of bagging and boosting – each consecutive ordering may take into account which objects were covered least frequently by decision bireducts derived up to now.

4.6 Decision Bireduct Ensemble Optimization

The rough set literature provides a great number of theoretical works on computational complexity of optimization problems focused on deriving the simplest possible decision models from the data [95]. Let us refer to a recent comparative study reflecting both decision bireducts and the so-called approximate reducts in this respect [119]. By "the simplest" one can mean (bi)reducts involving the minimal amounts of attributes, generating minimal amounts of decision rules, having the minimal information entropy, etc. However, all those formulations refer to single (bi)reducts which correspond to single classifiers.

In other words, as it was emphasized in [96], simplicity is a crucial aspect of decision models, in relation to paradigms such as Ockham's Razor or the Minimum Description Length Principle. However, there is no clear guidance how to understand simplicity of ensembles. Thus, if we want to define optimization problems for ensembles, we need to know how to aggregate "complexities" of particular ensemble components (e.g., the number of attributes in a single decision bireduct, the number of leaves in a single decision tree, etc.).

Intuitively, in case of ensembles of decision bireducts, the corresponding optimization problem should be stated by means of finding the smallest subsets of attributes $B_1, \ldots, B_m$ that satisfy – together with their counterparts for subsets of objects $\mathcal{X}_1, \ldots, \mathcal{X}_m$ – the constraints of Definition 39. The question remains what we should mean by "the smallest" in case of a collection of subsets. In [137], for the analogous task related to the already-mentioned generalized decision reducts, it was proposed to look at it from the perspective of the maximum cardinality out of all involved subsets. For the purpose of bireducts it can be phrased as follows:

Definition 40 (simpler bireduct ensemble)**.** *Let decision table $\mathbb{A} = (U, A \cup \{d\})$ and two correct ensembles of decision bireducts $\mathcal{B} = \{(\mathcal{X}_1, B_1), \ldots, (\mathcal{X}_m, B_m)\}$ and $\mathcal{C} = \{(\mathcal{Y}_1, C_1), \ldots, (\mathcal{Y}_n, C_n)\}$, $m, n \geq 0$, be given. We say that $\mathcal{B}$ is simpler than $\mathcal{C}$, denoted as $\mathcal{B} \prec \mathcal{C}$, if and only if they are obtained by the following procedure:*

1. *Sort sequences of cardinalities of attribute subsets in a descending order.*
2. *Add one more item with the value -1 to the end of each of sequences.*
3. *Find the first position for which the sorted sequences differ from each other.*
4. *If the value in the above-found position is lower for $\mathcal{B}$ than for $\mathcal{C}$, then $\mathcal{B} \prec \mathcal{C}$.*

The above procedure from Definition 40 – illustrated additionally by Figure 4.3 – induces a linear order over ensembles of bireducts for a given $\mathbb{A}$. We therefore propose to search through a space of all correct ensembles $\mathcal{B} = \{(\mathcal{X}_1, B_1), \ldots, (\mathcal{X}_m, B_m)\}$ for $m \geq 0$, paying special attention to cardinalities of their largest components along a kind of cardinality-based lexicographic order. This is because the largest subsets of attributes correspond to the largest collections of the longest rules, i.e., they affect complexity of the model more significantly than other subsets.

Let us formalize the optimization goal that is presented above:

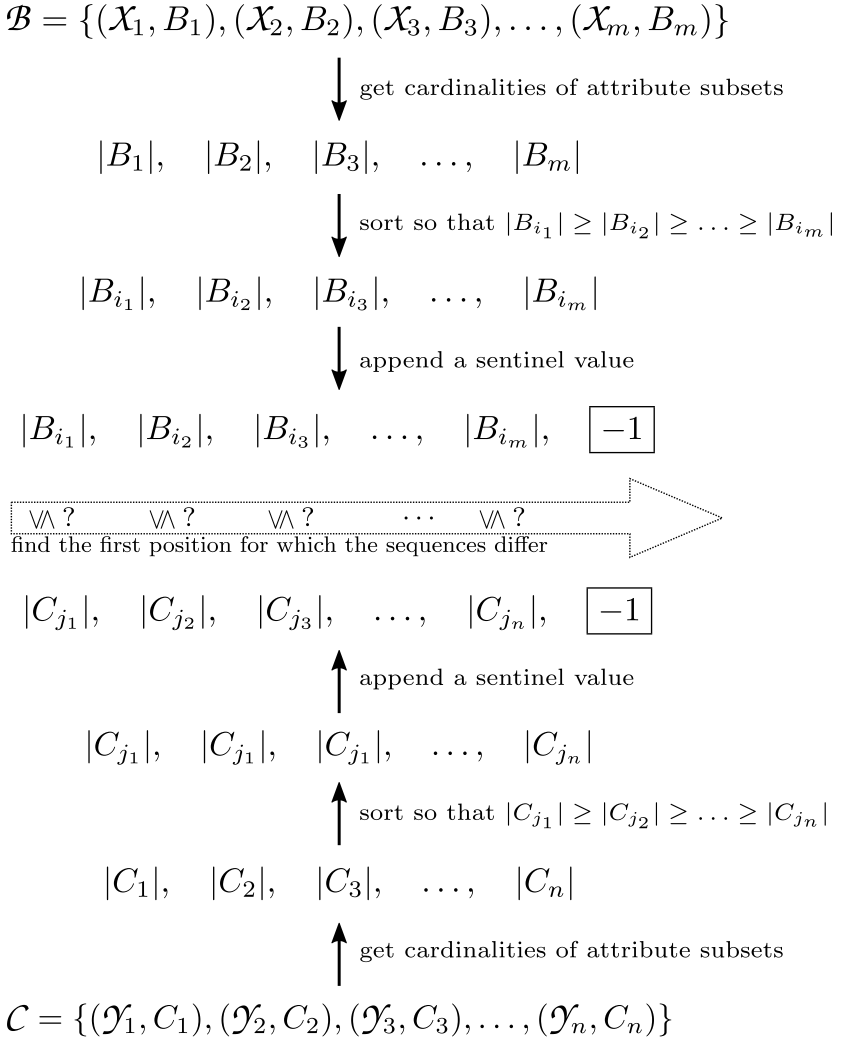
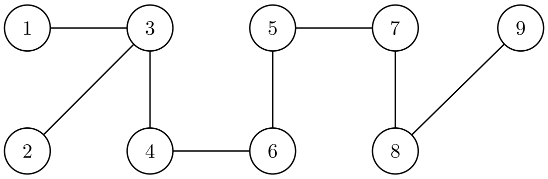

$\mathcal{B} = \{(\mathcal{X}_1, B_1), (\mathcal{X}_2, B_2), (\mathcal{X}_3, B_3), \dots, (\mathcal{X}_m, B_m)\}$
get cardinalities of attribute subsets
$|B_1|, \quad |B_2|, \quad |B_3|, \quad \dots, \quad |B_m|$
sort so that $|B_{i_1}| \geq |B_{i_2}| \geq \dots \geq |B_{i_m}|$
$|B_{i_1}|, \quad |B_{i_2}|, \quad |B_{i_3}|, \quad \dots, \quad |B_{i_m}|$
append a sentinel value
$|B_{i_1}|, \quad |B_{i_2}|, \quad |B_{i_3}|, \quad \dots, \quad |B_{i_m}|, \quad \boxed{-1}$
$\gtrless$? $\gtrless$? $\gtrless$? $\cdots$ $\gtrless$?
find the first position for which the sequences differ
$|C_{j_1}|, \quad |C_{j_2}|, \quad |C_{j_3}|, \quad \dots, \quad |C_{j_n}|, \quad \boxed{-1}$
append a sentinel value
$|C_{j_1}|, \quad |C_{j_1}|, \quad |C_{j_1}|, \quad \dots, \quad |C_{j_n}|$
sort so that $|C_{j_1}| \geq |C_{j_2}| \geq \dots \geq |C_{j_n}|$
$|C_1|, \quad |C_2|, \quad |C_3|, \quad \dots, \quad |C_n|$
get cardinalities of attribute subsets
$\mathcal{C} = \{(\mathcal{Y}_1, C_1), (\mathcal{Y}_2, C_2), (\mathcal{Y}_3, C_3), \dots, (\mathcal{Y}_n, C_n)\}$

Figure 4.3: Illustration of the procedure in Definition 40.



		$A_{\mathbb{G}}$								
		a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	$d_{\mathbb{G}}$
$U_{\mathbb{G}}$	u_1	1	0	1	0	0	0	0	0	0	0
	u_2	0	1	1	0	0	0	0	0	0	0
	u_3	1	1	1	1	0	0	0	0	0	0
	u_4	0	0	1	1	0	1	0	0	0	0
	u_5	0	0	0	0	1	1	1	0	0	0
	u_6	0	0	0	1	1	1	0	0	0	0
	u_7	0	0	0	0	1	0	1	1	0	0
	u_8	0	0	0	0	0	0	1	1	1	0
	u_9	0	0	0	0	0	0	0	1	1	0
	u_*	0	0	0	0	0	0	0	0	0	1

$\mathbb{A}_{\mathbb{G}} = (U_{\mathbb{G}} \cup \{u_*\}, A_{\mathbb{G}} \cup \{d_{\mathbb{G}}\})$

bireduct ensemble

$(\{u_1, u_2, u_3, u_4, u_*\}, \{a_3\})$, $(\{u_5, u_6, u_7, u_*\}, \{a_5\})$, $(\{u_7, u_8, u_9, u_*\}, \{a_8\})$, $(U_{\mathbb{G}}, \emptyset)$, $(U_{\mathbb{G}}, \emptyset)$

decision rules

covered objects	a_3		$d_{\mathbb{G}}$
$\{u_*\}$	0	→	1
$\{u_1, u_2, u_3, u_4\}$	1	→	0

covered objects	a_5		$d_{\mathbb{G}}$
$\{u_*\}$	0	→	1
$\{u_5, u_6, u_7\}$	1	→	0

covered objects	a_8		$d_{\mathbb{G}}$
$\{u_*\}$	0	→	1
$\{u_7, u_8, u_9\}$	1	→	0

covered objects			$d_{\mathbb{G}}$
$U_{\mathbb{G}}$	$\emptyset$	→	0

covered objects			$d_{\mathbb{G}}$
$U_{\mathbb{G}}$	$\emptyset$	→	0

voting

	$(\{u_1, u_2, u_3, u_4, u_*\}, \{a_3\})$	$(\{u_5, u_6, u_7, u_*\}, \{a_5\})$	$(\{u_7, u_8, u_9, u_*\}, \{a_8\})$	$(U_{\mathbb{G}}, \emptyset)$	$(U_{\mathbb{G}}, \emptyset)$	voting
u_1	$1 \to 0$	$0 \to 1$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_2	$1 \to 0$	$0 \to 1$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_3	$1 \to 0$	$0 \to 1$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_4	$1 \to 0$	$0 \to 1$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_5	$0 \to 1$	$1 \to 0$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_6	$0 \to 1$	$1 \to 0$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_7	$0 \to 1$	$1 \to 0$	$1 \to 0$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{4 \times 0}, 1 \times 1)$
u_8	$0 \to 1$	$0 \to 1$	$1 \to 0$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_9	$0 \to 1$	$0 \to 1$	$1 \to 0$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 0$ $(\underline{3 \times 0}, 2 \times 1)$
u_*	$0 \to 1$	$0 \to 1$	$0 \to 1$	$\emptyset \to 0$	$\emptyset \to 0$	$d_{\mathbb{G}} = 1$ $(2 \times 0, \underline{3 \times 1})$

Figure 4.4: Illustration for the proof of Proposition 26.

Definition 41 (simplest correct decision bireduct ensemble problem)**.** *By the Simplest Correct Decision Bireduct Ensemble Problem (SCDBEP) we mean the task of finding – for each input decision table $\mathbb{A} = (U, A \cup \{d\})$– the correct ensemble of decision bireducts $\mathcal{B}$ such that there is no other correct ensemble for $\mathbb{A}$ that would be simpler than $\mathcal{B}$ according to Definition 40.*

Proposition 26. *The Simplest Correct Decision Bireduct Ensemble Problem (SCDBEP) is NP-hard.*

Before we present the proof, let us refer to Figure 4.4. The proof is based on polynomial reduction of the problem of finding the smallest dominating sets in undirected graphs to SCDBEP. It requires encoding of each input graph $\mathbb{G}$ to its corresponding decision table $\mathbb{A}_{\mathbb{G}}$. This encoding is analogous to those that were utilized for other (bi)reduct-related optimization problems [95, 119].

Figure 4.4 can also serve as one more illustration of creation of correct ensembles of decision bireducts. It displays how to interpret those bireducts as rule-based classifiers. In particular, as it could be already noticed earlier in Figure 4.1, some bireducts can correspond to empty sets of attributes. We can interpret them as "dummy" classifiers which point always at the same decision class. They may help to tune the majority voting mechanism in the ensemble.

Proof. As already stated, we intend to show NP-hardness of SCDBEP by polynomial reduction of the minimal dominating set problem. Let us consider an undirected graph $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ and create decision table $\mathbb{A}_{\mathbb{G}} = (U_{\mathbb{G}} \cup \{u_*\}, A_{\mathbb{G}} \cup \{d_{\mathbb{G}}\})$, where $a_v \in A_{\mathbb{G}}$ corresponding to $v \in \mathbb{V}$ takes 1 on $u_{v'} \in U_{\mathbb{G}}$ corresponding to $v' \in \mathbb{V}$, i.e., $a_v(u_{v'}) = 1$, if and only if $v = v'$ or $(v, v') \in \mathbb{E}$, and where $a_v(u_*) = 0$. Let us also put $d_{\mathbb{G}}(u_{v'}) = 0$ and $d_{\mathbb{G}}(u_*) = 1$ (see Figure 4.4).

Clearly, any $B \subseteq \mathbb{V}$ is a dominating set in $\mathbb{G}$ if and only if it corresponds to a decision bireduct $(U_{\mathbb{G}}, B_{\mathbb{G}})$. It is obvious that a single-element bireduct ensemble $\{(U_{\mathbb{G}}, B_{\mathbb{G}})\}$ is correct according to Definition 39. However, we can (in terms of Definition 40) always construct a simpler (or equally simple if $B_{\mathbb{G}}$ is already a singleton) correct ensemble.

Assuming that $B_{\mathbb{G}} = \{a_{v_1}, a_{v_2}, \ldots, a_{v_n}\}$, let us define new subsets of attributes as $B_{\mathbb{G},1} = \{a_{v_1}\}$, ..., $B_{\mathbb{G},n} = \{a_{v_n}\}$, $B_{\mathbb{G},n+1} = \emptyset$, ..., $B_{\mathbb{G},2n-1} = \emptyset$ and new subsets of objects as $\mathcal{X}_{\mathbb{G},1} = \{u_*\} \cup \{u \in U_{\mathbb{G}} : a_{v_1}(u) = 1\}$, ..., $\mathcal{X}_{\mathbb{G},n} = \{u_*\} \cup \{u \in U_{\mathbb{G}} : a_{v_n}(u) = 1\}$, $\mathcal{X}_{\mathbb{G},n+1} = U_{\mathbb{G}}$, ..., $\mathcal{X}_{\mathbb{G},2n-1} = U_{\mathbb{G}}$. Then, the proposed simpler ensemble would be equal to $\{(\mathcal{X}_{\mathbb{G},1}, B_{\mathbb{G},1}), \ldots, (\mathcal{X}_{\mathbb{G},2n-1}, B_{\mathbb{G},2n-1})\}$ and it would be still correct according to Definition 39 (see Figure 4.4 again).

The consequence of the above is that the simplest correct ensemble of decision bireducts corresponds to the smallest dominating set in the graph $\mathbb{G}$. □

Chapter 5

Algorithms and First Experiments

In this chapter we introduce a number of possible algorithmic approaches to searching for decision bireducts. In general, we follow the techniques introduced earlier for decision reducts; like following what was done for permutation-based algorithms in [140], we extend the idea by considering the orderings on mixed codes of attributes and objects, instead of only orderings on attributes. We also attempt to translate some algorithms aiming at finding approximate decision reducts onto the case of decision bireducts, basing on the connections between both the notions.

5.1 Ordering Algorithms

Let us begin by recalling Algorithm 1 which was introduced in [140] as an extension of classical algorithms developed within the theory of rough sets for the search of standard decision reducts [3]. In the case of decision reducts, the above-mentioned algorithms worked with randomly generated permutations over the set of conditional attributes and needed to assure irreducibility of the produced subsets $B \subseteq A$. The idea was to start with a full set A and remove its elements along an ordering corresponding to a given permutation, unless removal of an attribute causes a violation of the constraints of keeping enough information about the decision. In case of the decision bireducts search we need to do something analogous but now assuring both, irreducibility of the subsets of attributes and non-extendability of the subsets of objects.

Algorithm 1 Decision Bireduct Ordering Algorithm

```
input: decision table $\mathbb{A} = (U, A \cup \{d\})$, $\sigma$ – permutation of a set $\{1, \dots, |U| + |A|\}$
$\mathcal{X}_0 \leftarrow \emptyset, B_0 \leftarrow A$
for $i = 1, \dots, |U| + |A|$ do
    $B_i \leftarrow B_{i-1}$
    $\mathcal{X}_i \leftarrow \mathcal{X}_{i-1}$
    if $\sigma(i) \leq |U|$ then
        if $B_i \Rrightarrow_{\mathcal{X}_i \cup \{u_{\sigma(i)}\}} d$ then
            $\mathcal{X}_i \leftarrow \mathcal{X}_i \cup \{u_{\sigma(i)}\}$
    else
        if $B_i \setminus \{a_{\sigma(i)-|U|}\} \Rrightarrow_{\mathcal{X}_i} d$ then
            $B_i \leftarrow B_i \setminus \{a_{\sigma(i)-|U|}\}$
return $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$
```

Proposition 27. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. For each permutation $\sigma : \{1, \dots, |U| + |A|\} \to \{1, \dots, |U| + |A|\}$ the final outcome $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$ of Algorithm 1 is a decision bireduct. Moreover,* *for each decision bireduct $(\mathcal{X}, B)$ there exists an input σ for which the algorithm's output equals to $(\mathcal{X}, B)$.*

Proof. Let a permutation $\sigma : \{1, \ldots, |U| + |A|\} \rightarrow \{1, \ldots, |U| + |A|\}$ be given. The procedure listed in Algorithm 1 initializes $\mathcal{X}_0 \leftarrow \emptyset$ and $B_0 \leftarrow A$ and therefore, at the beginning $B_0 \Rrightarrow_{\mathcal{X}_0} d$ holds. Later, in each step of the main loop, the action of removing an attribute or adding an object is performed only under the condition of preserving the dependence for the new layout of $\mathcal{X}_i$ and B_i. Therefore, at the end $B_{|U|+|A|} \Rrightarrow_{\mathcal{X}_{|U|+|A|}} d$ also holds. Now, to prove that the output pair $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$ is a decision bireduct, we need to show that neither $\mathcal{X}_{|U|+|A|}$ can be extended nor $B_{|U|+|A|}$ can be reduced, in a sense of conditions 2 and 3 in Definition 34. Let us prove this part by a contradiction. We have two possibilities. **(1.)** Suppose that for the output pair $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$ there exists proper superset $\mathcal{X}' \supsetneq \mathcal{X}_{|U|+|A|}$ such that $B_{|U|+|A|} \Rrightarrow_{\mathcal{X}'} d$. Let us take $u_j \in \mathcal{X}' \setminus \mathcal{X}_{|U|+|A|}$. There exists $i \in \{1, \ldots, |U|+|A|\}$ such that $\sigma(i) = j$. Since the algorithm starts with $\mathcal{X}_0 = \emptyset$ and it can only extend this set during its main loop, we know that at the i-th iteration there is $\mathcal{X}_i \subseteq \mathcal{X}_{|U|+|A|} \subsetneq \mathcal{X}'$ and from Proposition 13 for the set $\mathcal{X}_i \cup \{u_j\} \subseteq \mathcal{X}'$ the dependence $B_i \Rrightarrow_{\mathcal{X}_i \cup \{u_j\}} d$ is satisfied. Therefore, at this step object u_j is added to $\mathcal{X}_i$ and also as a consequence there must be $u_j \in \mathcal{X}_{|U|+|A|}$ – a contradiction. **(2.)** Suppose that for the output pair $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$ there exists $B' \subsetneq B_{|U|+|A|}$ such that $B' \Rrightarrow_{\mathcal{X}_{|U|+|A|}} d$ holds. Let us take $a_j \in B_{|U|+|A|} \setminus B'$. There exists $i \in \{1, \ldots, |U| + |A|\}$ for which $\sigma(i) = j + |U|$. Since the algorithm starts with $B_0 = A$ and it can only reduce it, we know that at the i-th iteration of the algorithm's loop there is $B_i \supseteq B_{|U|+|A|} \supsetneq B'$ and from Proposition 13 the set $B_i \setminus \{a_j\} \supseteq B'$ satisfies $B_i \setminus \{a_j\} \Rrightarrow_{\mathcal{X}_i} d$. Therefore, at this step a_j is removed from B_i and as a consequence it would not be present in the output attribute subset $B_{|U|+|A|}$. Hence, $a_j \notin B_{|U|+|A|} \setminus B'$ – a contradiction. This shows that the algorithm's output pair is a decision bireduct.

To finish the proof, we need to show that for each decision bireduct $(\mathcal{X}, B)$ there exists an input permutation σ for which the algorithm outputs $(\mathcal{X}, B)$. Let us consider a permutation with the following characteristics:

a. $a_{\sigma(i)-|U|} \in A \setminus B$ for $i \in \{1, \ldots, |A \setminus B|\}$,

b. $u_{\sigma(i)} \in \mathcal{X}$ for $i \in \{|A \setminus B| + 1, \ldots, |A \setminus B| + |\mathcal{X}|\}$,

c. $a_{\sigma(i)-|U|} \in B$ for $i \in \{|A \setminus B| + |\mathcal{X}| + 1, \ldots, |\mathcal{X}| + |A|\}$,

d. $u_{\sigma(i)} \in U \setminus \mathcal{X}$ for $i \in \{|\mathcal{X}| + |A| + 1, \ldots, |U| + |A|\}$.

Segment (a.) ensures that after the first $|A \setminus B|$ steps of the algorithm's loop $B_{|A \setminus B|} = B$ which is quite obvious because we start with $\mathcal{X}_0 = \emptyset$. Subsequently, because $B_{|A\setminus B|} = B$, segment (b.) ensures that after another $|\mathcal{X}|$ steps we have $\mathcal{X}_{|A\setminus B|+|\mathcal{X}|} = \mathcal{X}$, because the algorithm can at each step extend $\mathcal{X}_i$ with the elements of $\mathcal{X}$. Segments (c. and d.) are listed only for completeness of the permutation construction. Indeed, within the last $|B| + |U \setminus \mathcal{X}|$ steps the state $B_{|A\setminus B|+|\mathcal{X}|} = B$ and $\mathcal{X}_{|A\setminus B|+|\mathcal{X}|} = \mathcal{X}$ will not be changed as this pair already forms the decision bireduct, so the subsets of attributes and objects cannot be further reduced and extended, respectively. Therefore, $\mathcal{X}_{|U|+|A|} = \mathcal{X}$ and $B_{|U|+|A|} = B$. □

The method follows an idea of mixing the processes of reducing attributes and adding objects during the construction of decision bireducts. For a consistent data set and a set of permutations $\sigma : \{1, \ldots, |U| + |A|\} \rightarrow \{1, \ldots, |U| + |A|\}$ where all objects are added to $\mathcal{X}$ before removing attributes from A, we obtain standard decision reducts. For inconsistent tables, the ordering of objects at the beginning of permutation will decide which decision classes are going to be represented by $\mathcal{X}$ within the particular indiscernibility classes generated by A.

In [140, 119] it was noted that it is difficult to control permutations $\sigma : \{1, \ldots, |U| + |A|\} \rightarrow \{1, \ldots, |U| + |A|\}$ in order to obtain decision bireducts $(\mathcal{X}, B)$ with any pre-specified level of $|\mathcal{X}|$, $|B|$, or any kind of proportion between $|\mathcal{X}|$ and $|B|$. On the other hand, the process of searching for decision bireducts with desired properties (such as the average number of attributes and objects or

Table 5.1: Examples of results of Algorithm 1: decision bireducts from a data set in Table 3.1 and permutations that the algorithm followed while generating them. The values in the "permutation" column correspond to the input arguments of Algorithm 1 and denote the order in which the algorithm will handle either object or attributes from data table. A value at position i correspond to either an object $u_{\sigma(i)}$ (if $\sigma(i) \leq |U|$) or an attribute $a_{\sigma(i)-|U|}$ (if $\sigma(i) > |U|$). For convenience, in case of $\sigma(i) > |U|$ we add information in parentheses to which attribute the value corresponds. E.g., "15(O) 8 18(W) 1 4 7 2 14 10 12 9 16(T) 6 3 13 5 11 17(H)" means that the algorithm will process the elements in the following order: Outlook, u_8, Wind, $u_1, u_4, u_7, u_2, u_{14}, u_{10}, u_{12}, \ldots,$ etc.

permutation	**decision bireduct**
15(O) 8 18(W) 1 4 7 2 14 10 12 9 16(T) 6 3 13 5 11 17(H)	$(\llbracket 1,2,5,7..11,13,14 \rrbracket, \{H\})$
17(H) 13 16(T) 8 18(W) 6 11 3 14 10 15(O) 5 7 9 2 1 4 12	$(\llbracket 1..3,6..8,12..14 \rrbracket, \{O\})$
3 8 16(T) 1 18(W) 11 9 15(O) 14 12 6 4 7 17(H) 10 13 2 5	$(\llbracket 1..3,6..9,11..14 \rrbracket, \{O,H\})$
2 13 5 14 11 7 12 4 3 1 9 6 8 10 17(H) 15(O) 18(W) 16(T)	$(\llbracket 1..14 \rrbracket, \{O,T,W\})$
9 4 12 14 1 8 7 3 10 13 6 11 2 5 18(W) 16(T) 17(H) 15(O)	$(\llbracket 1..14 \rrbracket, \{O,H,W\})$
11 15(O) 2 17(H) 1 10 5 7 9 8 3 13 16(T) 6 14 12 4 18(W)	$(\llbracket 1,2,4,5,7,9..12 \rrbracket, \{T\})$
16(T) 2 5 17(H) 10 11 18(W) 14 1 12 7 9 13 6 4 8 3 15(O)	$(\llbracket 1..5,7,8,10,12,13 \rrbracket, \{O\})$
18(W) 6 17(H) 15(O) 5 8 4 7 3 2 10 9 12 11 13 14 1 16(T)	$(\llbracket 3,6,8,13,14 \rrbracket, \{T\})$
15(O) 2 3 13 1 17(H) 4 16(T) 18(W) 6 12 14 5 8 9 10 11 7	$(\llbracket 2..6,9,10,13,14 \rrbracket, \{W\})$
15(O) 17(H) 14 1 10 7 4 3 12 13 5 18(W) 9 16(T) 11 8 2 6	$(\llbracket 1,2,4,5,7,9,10,14 \rrbracket, \{T,W\})$
6 5 10 9 17(H) 15(O) 12 16(T) 8 18(W) 4 2 13 3 7 1 14 11	$(\llbracket 2..6,9..13 \rrbracket, \{T,W\})$
11 14 9 13 3 7 8 2 5 1 12 18(W) 6 4 10 17(H) 15(O) 16(T)	$(\llbracket 1..3,5,7..14 \rrbracket, \{O,H\})$
13 8 6 17(H) 7 18(W) 9 16(T) 5 3 4 12 15(O) 2 10 14 11 1	$(\llbracket 1..4,6..10,12,13 \rrbracket, \{O,T\})$
9 17(H) 2 4 6 13 14 7 16(T) 11 10 15(O) 18(W) 3 5 1 8 12	$(\llbracket 2..7,9,10,12..14 \rrbracket, \{O,W\})$
18(W) 5 3 15(O) 12 4 16(T) 17(H) 7 2 13 11 10 6 8 1 14 9	$(\llbracket 3..5,7,9..13 \rrbracket, \emptyset)$

more sophisticated criteria) can at least partially modeled by a way the permutations are generated. For example, we can consider a parameter that controls a likelihood of selecting an attribute in first place rather than an object during the random generation of σ. Then, when σ contains relatively more attributes at its beginning, a decision bireduct having a lower number of attributes but at the same time a higher number of uncovered objects is likely to be obtained. In [140, 119], where such a likelihood was called *ratio*, it was demonstrated how different values of this parameter influence the number of selected attributes and the size of $\mathcal{X}$ in bireducts generated for a few benchmark data sets. We will also discuss the subject more deeply in Section 5.4.

Algorithm 2 γ-Decision Bireduct Ordering Algorithm

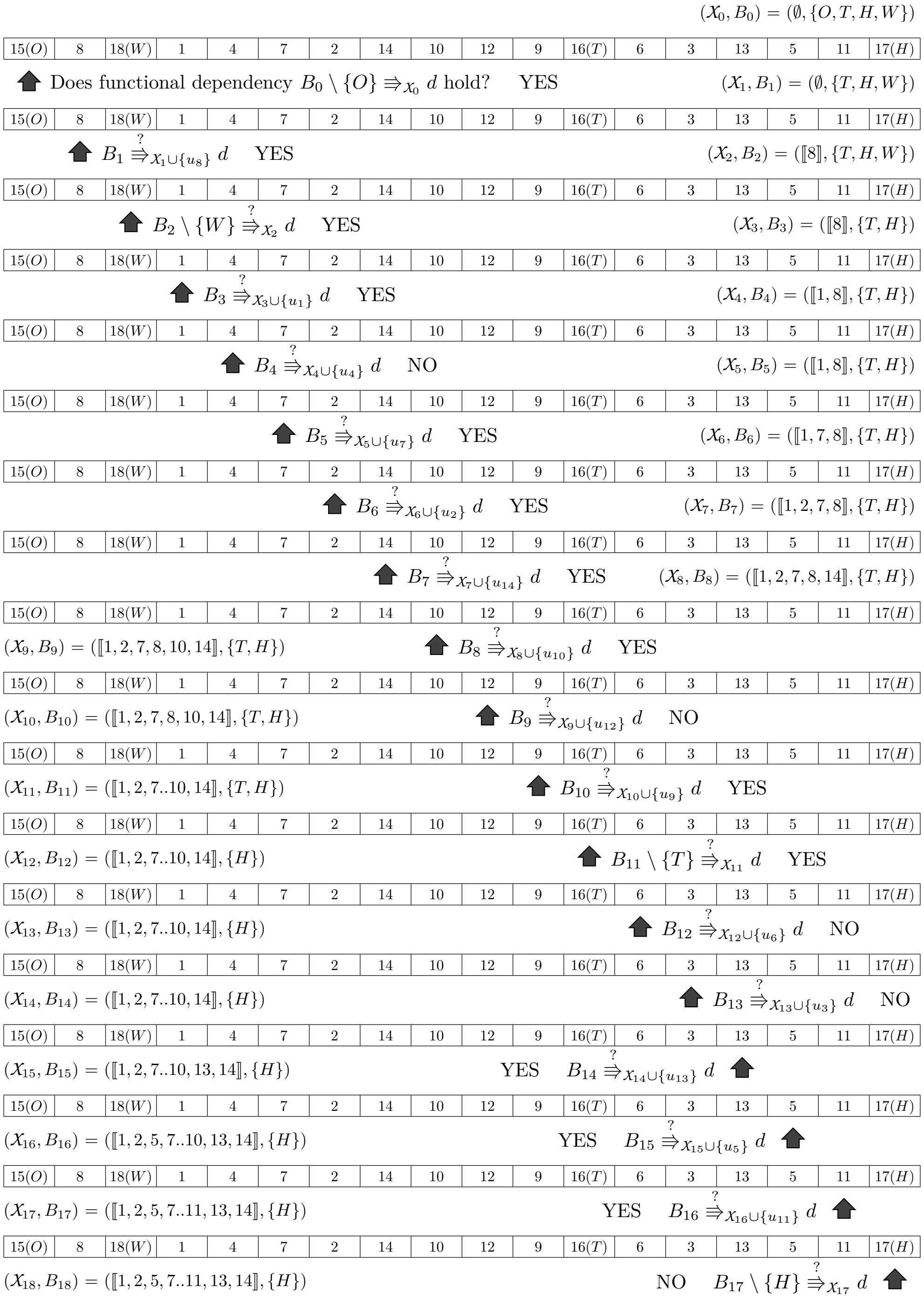
```
input: decision table $\mathbb{A} = (U, A \cup \{d\})$, $\sigma$ – permutation of a set $\{1, \ldots, |U| + |A|\}$
$\mathcal{X}_0 \leftarrow \emptyset, B_0 \leftarrow A$
for $i = 1, \ldots, |U| + |A|$ do
    $B_i \leftarrow B_{i-1}$
    $\mathcal{X}_i \leftarrow \mathcal{X}_{i-1}$
    if $\sigma(i) \leq |U|$ then
        if $B_i \Rrightarrow^{\gamma}_{\mathcal{X}_i \cup \{u_{\sigma(i)}\}} d$ then
            $\mathcal{X}_i \leftarrow \mathcal{X}_i \cup \{u_{\sigma(i)}\}$
    else
        if $B_i \setminus \{a_{\sigma(i)-|U|}\} \Rrightarrow^{\gamma}_{\mathcal{X}_i} d$ then
            $B_i \leftarrow B_i \setminus \{a_{\sigma(i)-|U|}\}$
return $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$
```

In the case of γ-decision bireduct the algorithm is almost the same as Algorithm 1. It is enough to replace the inexact functional dependency $\Rrightarrow_{\mathcal{X}}$ by its counterpart from the definition of the γ-decision bireduct, i.e., $\Rrightarrow^{\gamma}_{\mathcal{X}}$. For completeness of the dissertation we present the details of the ordering algorithm for γ-decision bireducts in Algorithm 2.

Proposition 28. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. For each given permutation $\sigma : \{1, \ldots, |U| + |A|\} \to \{1, \ldots, |U| + |A|\}$ the final outcome $(\mathcal{X}_{|U|+|A|}, B_{|U|+|A|})$ of Algorithm 2 is a γ-decision bireduct. Moreover, for each γ-decision bireduct $(\mathcal{X}, B)$ there exists an input permutation σ for which the algorithm's output equals to $(\mathcal{X}, B)$.*

Proof. The proof is analogous to the proof of Proposition 27. In its first stage, we refer to Definition 35 and Proposition 17. In the second stage, we use exactly the same (a., b., c. and d.)-characteristic as in the previous proof. □

Tables 5.1 and 5.2 present exemplary results of the ordering decision bireducts algorithm (Algorithm 1) and γ-decision bireducts algorithm (Algorithm 2) invoked for the same sample permutations, whereas Figures 5.1 and 5.2 present in details the computation for one selected permutation.



$(\mathcal{X}_0, B_0) = (\emptyset, \{O, T, H, W\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ Does functional dependency $B_0 \setminus \{O\} \Rrightarrow_{\mathcal{X}_0} d$ hold? YES $(\mathcal{X}_1, B_1) = (\emptyset, \{T, H, W\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_1 \overset{?}{\Rrightarrow}_{\mathcal{X}_1 \cup \{u_8\}} d$ YES $(\mathcal{X}_2, B_2) = ([\![8]\!], \{T, H, W\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_2 \setminus \{W\} \overset{?}{\Rrightarrow}_{\mathcal{X}_2} d$ YES $(\mathcal{X}_3, B_3) = ([\![8]\!], \{T, H\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_3 \overset{?}{\Rrightarrow}_{\mathcal{X}_3 \cup \{u_1\}} d$ YES $(\mathcal{X}_4, B_4) = ([\![1, 8]\!], \{T, H\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_4 \overset{?}{\Rrightarrow}_{\mathcal{X}_4 \cup \{u_4\}} d$ NO $(\mathcal{X}_5, B_5) = ([\![1, 8]\!], \{T, H\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_5 \overset{?}{\Rrightarrow}_{\mathcal{X}_5 \cup \{u_7\}} d$ YES $(\mathcal{X}_6, B_6) = ([\![1, 7, 8]\!], \{T, H\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_6 \overset{?}{\Rrightarrow}_{\mathcal{X}_6 \cup \{u_2\}} d$ YES $(\mathcal{X}_7, B_7) = ([\![1, 2, 7, 8]\!], \{T, H\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

⬆ $B_7 \overset{?}{\Rrightarrow}_{\mathcal{X}_7 \cup \{u_{14}\}} d$ YES $(\mathcal{X}_8, B_8) = ([\![1, 2, 7, 8, 14]\!], \{T, H\})$

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_9, B_9) = ([\![1, 2, 7, 8, 10, 14]\!], \{T, H\})$ ⬆ $B_8 \overset{?}{\Rrightarrow}_{\mathcal{X}_8 \cup \{u_{10}\}} d$ YES

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{10}, B_{10}) = ([\![1, 2, 7, 8, 10, 14]\!], \{T, H\})$ ⬆ $B_9 \overset{?}{\Rrightarrow}_{\mathcal{X}_9 \cup \{u_{12}\}} d$ NO

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{11}, B_{11}) = ([\![1, 2, 7..10, 14]\!], \{T, H\})$ ⬆ $B_{10} \overset{?}{\Rrightarrow}_{\mathcal{X}_{10} \cup \{u_9\}} d$ YES

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{12}, B_{12}) = ([\![1, 2, 7..10, 14]\!], \{H\})$ ⬆ $B_{11} \setminus \{T\} \overset{?}{\Rrightarrow}_{\mathcal{X}_{11}} d$ YES

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{13}, B_{13}) = ([\![1, 2, 7..10, 14]\!], \{H\})$ ⬆ $B_{12} \overset{?}{\Rrightarrow}_{\mathcal{X}_{12} \cup \{u_6\}} d$ NO

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{14}, B_{14}) = ([\![1, 2, 7..10, 14]\!], \{H\})$ ⬆ $B_{13} \overset{?}{\Rrightarrow}_{\mathcal{X}_{13} \cup \{u_3\}} d$ NO

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{15}, B_{15}) = ([\![1, 2, 7..10, 13, 14]\!], \{H\})$ YES $B_{14} \overset{?}{\Rrightarrow}_{\mathcal{X}_{14} \cup \{u_{13}\}} d$ ⬆

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{16}, B_{16}) = ([\![1, 2, 5, 7..10, 13, 14]\!], \{H\})$ YES $B_{15} \overset{?}{\Rrightarrow}_{\mathcal{X}_{15} \cup \{u_5\}} d$ ⬆

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{17}, B_{17}) = ([\![1, 2, 5, 7..11, 13, 14]\!], \{H\})$ YES $B_{16} \overset{?}{\Rrightarrow}_{\mathcal{X}_{16} \cup \{u_{11}\}} d$ ⬆

15(O)	8	18(W)	1	4	7	2	14	10	12	9	16(T)	6	3	13	5	11	17(H)

$(\mathcal{X}_{18}, B_{18}) = ([\![1, 2, 5, 7..11, 13, 14]\!], \{H\})$ NO $B_{17} \setminus \{H\} \overset{?}{\Rrightarrow}_{\mathcal{X}_{17}} d$ ⬆

Figure 5.1: Example of the ordering decision bireduct algorithm (Algorithm 1) computation for a data set in Table 3.1. A permutation (visible in all rows) corresponds to the input argument of Algorithm 1 and should be understood in the same way as in Table 5.1. The rows represent the consecutive iterations of the algorithm's loop – the arrow points to an element being processed in a given iteration.

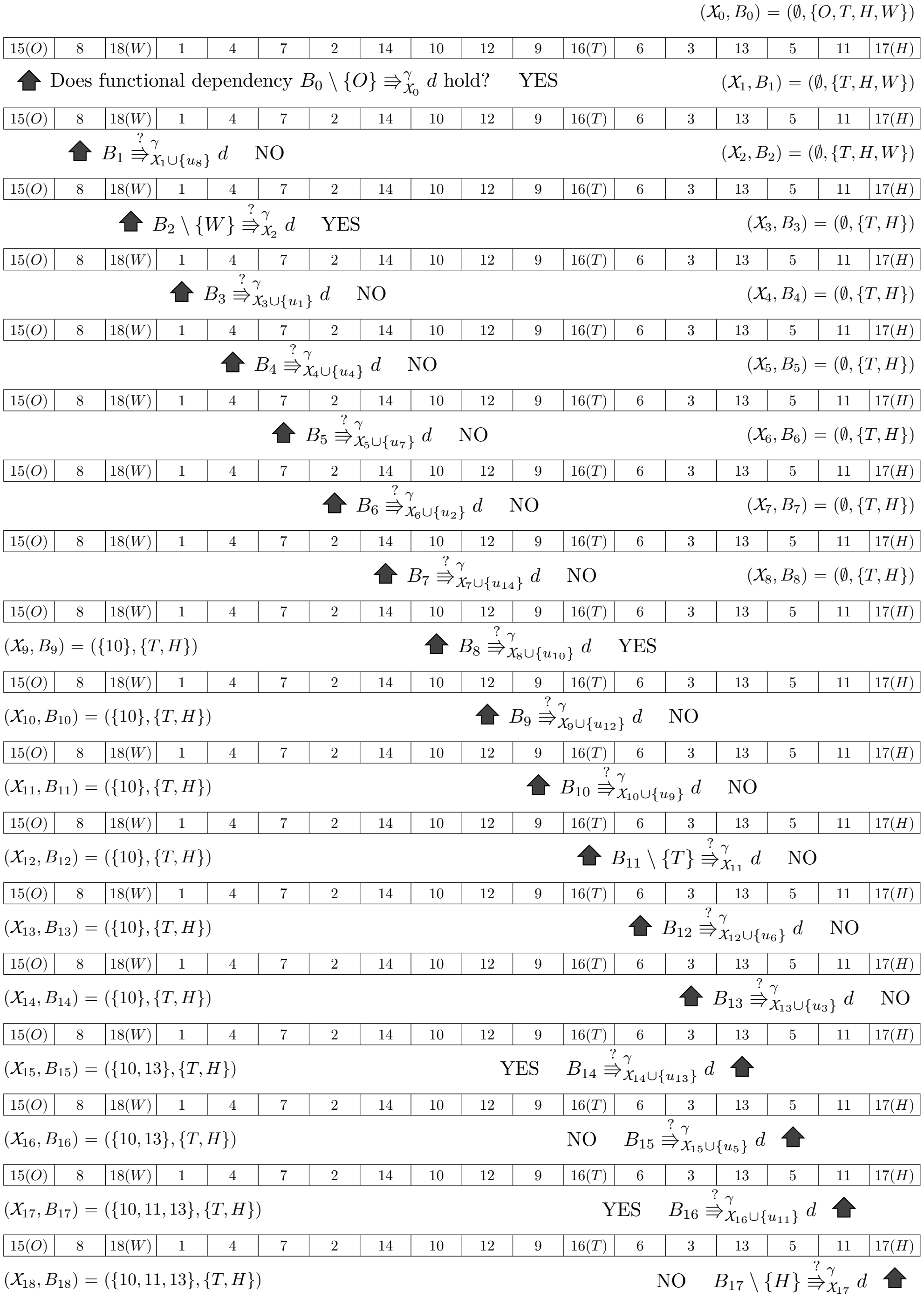

$(X_0, B_0) = (\emptyset, \{O, T, H, W\})$
15(O) 8 18(W) 1 4 7 2 14 10 12 9 16(T) 6 3 13 5 11 17(H)
Does functional dependency $B_0 \setminus \{O\} \Rrightarrow^{\gamma}_{X_0} d$ hold? YES
$(X_1, B_1) = (\emptyset, \{T, H, W\})$
$B_1 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_1 \cup \{u_8\}} d$ NO
$(X_2, B_2) = (\emptyset, \{T, H, W\})$
$B_2 \setminus \{W\} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_2} d$ YES
$(X_3, B_3) = (\emptyset, \{T, H\})$
$B_3 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_3 \cup \{u_1\}} d$ NO
$(X_4, B_4) = (\emptyset, \{T, H\})$
$B_4 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_4 \cup \{u_4\}} d$ NO
$(X_5, B_5) = (\emptyset, \{T, H\})$
$B_5 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_5 \cup \{u_7\}} d$ NO
$(X_6, B_6) = (\emptyset, \{T, H\})$
$B_6 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_6 \cup \{u_2\}} d$ NO
$(X_7, B_7) = (\emptyset, \{T, H\})$
$B_7 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_7 \cup \{u_{14}\}} d$ NO
$(X_8, B_8) = (\emptyset, \{T, H\})$
$(X_9, B_9) = (\{10\}, \{T, H\})$
$B_8 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_8 \cup \{u_{10}\}} d$ YES
$(X_{10}, B_{10}) = (\{10\}, \{T, H\})$
$B_9 \overset{?}{\Rrightarrow}{}^{\gamma}_{X_9 \cup \{u_{12}\}} d$ NO
$(X_{11}, B_{11}) = (\{10\}, \{T, H\})$
$B_{10} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{10} \cup \{u_9\}} d$ NO
$(X_{12}, B_{12}) = (\{10\}, \{T, H\})$
$B_{11} \setminus \{T\} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{11}} d$ NO
$(X_{13}, B_{13}) = (\{10\}, \{T, H\})$
$B_{12} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{12} \cup \{u_6\}} d$ NO
$(X_{14}, B_{14}) = (\{10\}, \{T, H\})$
$B_{13} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{13} \cup \{u_3\}} d$ NO
$(X_{15}, B_{15}) = (\{10, 13\}, \{T, H\})$
YES $B_{14} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{14} \cup \{u_{13}\}} d$
$(X_{16}, B_{16}) = (\{10, 13\}, \{T, H\})$
NO $B_{15} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{15} \cup \{u_5\}} d$
$(X_{17}, B_{17}) = (\{10, 11, 13\}, \{T, H\})$
YES $B_{16} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{16} \cup \{u_{11}\}} d$
$(X_{18}, B_{18}) = (\{10, 11, 13\}, \{T, H\})$
NO $B_{17} \setminus \{H\} \overset{?}{\Rrightarrow}{}^{\gamma}_{X_{17}} d$

Figure 5.2: Example of the ordering γ-decision bireduct algorithm (Algorithm 2) computation for a data set in Table 3.1. A permutation (visible in all rows) corresponds to the input argument of Algorithm 2 and should be understood in the same way as in Table 5.1. The rows represent consecutive iterations of the algorithm's loop – the arrow points to an element being processed in a given iteration.

Table 5.2: Examples of results of Algorithm 2: γ-decision bireducts from a data set in Table 3.1 and permutations that the algorithm followed while generating them. The values in the "permutation" column correspond to the input arguments of Algorithm 2 and should be understood in the same way as in Table 5.1.

permutation	**γ-decision bireduct**
15(O) 8 18(W) 1 4 7 2 14 10 12 9 16(T) 6 3 13 5 11 17(H)	$(\llbracket 10,11,13 \rrbracket, \{T,H\})$
17(H) 13 16(T) 8 18(W) 6 11 3 14 10 15(O) 5 7 9 2 1 4 12	$(\llbracket 3,7,12,13 \rrbracket, \{O\})$
3 8 16(T) 1 18(W) 11 9 15(O) 14 12 6 4 7 17(H) 10 13 2 5	$(\llbracket 1..3,7..9,11..13 \rrbracket, \{O,H\})$
2 13 5 14 11 7 12 4 3 1 9 6 8 10 17(H) 15(O) 18(W) 16(T)	$(\llbracket 1..14 \rrbracket, \{O,T,W\})$
9 4 12 14 1 8 7 3 10 13 6 11 2 5 18(W) 16(T) 17(H) 15(O)	$(\llbracket 1..14 \rrbracket, \{O,H,W\})$
11 15(O) 2 17(H) 1 10 5 7 9 8 3 13 16(T) 6 14 12 4 18(W)	$(\llbracket 2,5,9..11,13 \rrbracket, \{T,H,W\})$
16(T) 2 5 17(H) 10 11 18(W) 14 1 12 7 9 13 6 4 8 3 15(O)	$(\llbracket 1..14 \rrbracket, \{O,H,W\})$
18(W) 6 17(H) 15(O) 5 8 4 7 3 2 10 9 12 11 13 14 1 16(T)	$(\emptyset, \emptyset)$
15(O) 2 3 13 1 17(H) 4 16(T) 18(W) 6 12 14 5 8 9 10 11 7	$(\llbracket 2,5,9..11,13 \rrbracket, \{T,H,W\})$
15(O) 17(H) 14 1 10 7 4 3 12 13 5 18(W) 9 16(T) 11 8 2 6	$(\llbracket 2,5,9 \rrbracket, \{T,W\})$
6 5 10 9 17(H) 15(O) 12 16(T) 8 18(W) 4 2 13 3 7 1 14 11	$(\llbracket 1..14 \rrbracket, \{O,T,W\})$
11 14 9 13 3 7 8 2 5 1 12 18(W) 6 4 10 17(H) 15(O) 16(T)	$(\llbracket 1..14 \rrbracket, \{O,T,W\})$
13 8 6 17(H) 7 18(W) 9 16(T) 5 3 4 12 15(O) 2 10 14 11 1	$(\llbracket 1..14 \rrbracket, \{O,T,W\})$
9 17(H) 2 4 6 13 14 7 16(T) 11 10 15(O) 18(W) 3 5 1 8 12	$(\llbracket 1..14 \rrbracket, \{O,T,W\})$
18(W) 5 3 15(O) 12 4 16(T) 17(H) 7 2 13 11 10 6 8 1 14 9	$(\llbracket 3,7,12,13 \rrbracket, \{O\})$

5.2 Sampling Algorithms

We recall one more method introduced in [118] which can be generally referred to sampling approaches to attribute selection [64]. Its aim is to speed up the computation of the bireducts for high dimensional data by combining a Monte Carlo searching method with classical reduct computation algorithms. It can be also compared to well-known idea of bagging, i.e., inducing classifiers basing on randomly selected subsets of the training data set [28], as well as other sampling techniques developed for the search of approximate decision reducts [51], although in our case random selection of objects is performed under different constraints and at a different stage of decision model induction.

Let us recall that $X^{\langle d(u^\diamond)\rangle}$ denotes a decision class which contains a given object $u^\diamond \in U$. Thus, Algorithm 3, in particular its Line 9, has a lot in common with Proposition 16 and the way of summing up the supports of *Rules* – the set of decision rules defined by Equation (4.1).

Proposition 29. *Let the algorithm's input arguments $\mathbb{A} = (U, A \cup \{d\})$ and $A^\diamond \subseteq A$, be given. Then the outcome $(\mathcal{X}, B)$ of Algorithm 3 is a decision bireduct for $\mathbb{A}$. Moreover, each decision bireduct for $\mathbb{A}$ can be obtained as a result of Algorithm 3.*

Proof. For $A^\diamond \subseteq A$, the decision table $\mathbb{A}_\diamond = (U^\diamond, A^\diamond \cup \{d\})$ is consistent. Due to the fact that we construct the final $(\mathcal{X}, B)$ by taking those of the objects from U which have the same values on attributes $B \cup \{d\}$ as objects in $U^\diamond$, we know that $B \Rrightarrow_{\mathcal{X}} d$ and that the attribute subset B is irreducible. Furthermore, we have chosen one representative object from each indiscernibility class induced by $A^\diamond$. Thus, for the decision reduct $B \subseteq A^\diamond$ obtained for $\mathbb{A}_\diamond$, we know that each indiscernibility class induced by B has its representative object (at least one) in $U^\diamond$ and all objects from the same class have the same value for d. Hence, we cannot extend $\mathcal{X}$ obtained in Algorithm 3 with any other object $u \in U \setminus \mathcal{X}$ without violating the decision bireduct conditions. This is because u has a different decision value than objects from $\mathcal{X}$ that belong to the same indiscernibility class induced by B.

Algorithm 3 Decision Bireduct Sampling Algorithm

```
 1: input: decision table $\mathbb{A} = (U, A \cup \{d\})$
 2: $U^\diamond \leftarrow \emptyset$
 3: $\mathcal{X} \leftarrow \emptyset$
 4: $A^\diamond \leftarrow$ an attribute subset of $A$
 5: ▷ $A^\diamond$ may be either passed as an input argument or chosen within the algorithm according to
      a selected strategy, such as random selection
 6: for all $E \in U/A^\diamond$ do
 7:     $u^\diamond \leftarrow$ a single object chosen from $E$
 8:     $U^\diamond \leftarrow U^\diamond \cup \{u^\diamond\}$
 9:     $\mathcal{X} \leftarrow \mathcal{X} \cup (E \cap X^{\langle d(u^\diamond) \rangle})$
10: $B \leftarrow$ arbitrary decision reduct $B \subseteq A^\diamond$ for table $\mathbb{A}_\diamond = (U^\diamond, A^\diamond \cup \{d\})$
11: return $(\mathcal{X}, B)$
```

Now, consider a decision bireduct $(\mathcal{X}, B)$. It can be built in at least one way as a result of execution of Algorithm 3. For instance, let us put $A^\diamond = B$ and let $U^\diamond$ be formed by taking the representative objects solely from $\mathcal{X}$. Since $(\mathcal{X}, B)$ is a decision bireduct, $\mathcal{X}$ needs to have at least one object in each indiscernibility class induced by B. As a consequence, $U^\diamond$ has exactly one object from each indiscernibility class induced by B. The third phase of the algorithm is to obtain a standard decision reduct B for $\mathbb{A}_\diamond$. Since B is irreducible for $\mathcal{X}$, it is also irreducible for the chosen representative objects $U^\diamond$. Hence, B is the only decision reduct for $\mathbb{A}_\diamond$. □

We illustrate the above procedure by Tables 5.3, 5.4 and 5.5. Let us note that the reduced decision tables $\mathbb{A}_\diamond$ obtained in the third step of Algorithm 3 are compact representations of if-then rules generated by attributes in B, with their supports summing up to the overall support $\mathcal{X} \subseteq U$. However, successors of those rules are not necessarily chosen in a way aiming at maximizing $|\mathcal{X}|$. Quite oppositely, when combined with appropriate mechanisms of sampling, this process can lead to ensembles of decision bireducts based on possibly diversified subsets of attributes and objects, with the underlying if-then rules paying attention to the cases not covered by rules corresponding to other decision bireducts rather than the cases that are easiest to describe.

The method outlined in Algorithm 3 could be also modeled within the framework presented in Algorithm 1, by considering specific permutations $\sigma : \{1, \ldots, |U| + |A|\} \rightarrow \{1, \ldots, |U| + |A|\}$ with some amount of attributes at their beginning, an ordering of all objects in their middle, and the remainder of attributes at their very end. Indeed, in such a case, all attributes at the very beginning of σ would be removed and then, within each indiscernibility class induced by the remaining attributes, objects corresponding to only one of possible decision values would be added. It would be the decision value of the first element of a given indiscernibility class occurring in σ. Finally, the algorithm would try to remove each of the remaining attributes due to their ordering in σ, subject to the constraints imposed by the previously added objects.

This analogy may help us in defining a parameter similar to *ratio* discussed for Algorithms 1 and 2 which would enable searching for bireducts with expected characteristics related to the size of attribute and object sets. Such a parameter could correspond to an expected value of $\frac{|A^\diamond|}{|A|}$. For instance, if in Line 4 of Algorithm 3 we would have drawn a smaller set of attributes, then there would be less indiscernibility classes, and thus a decision reduct of $\mathbb{A}_\diamond$ could also be much smaller. As a result we would obtain a bireduct with a much smaller $\mathcal{X}$ described by a lower number of attributes. In the opposite situation, when the selected $A^\diamond$ would be relatively large, then the corresponding indiscernibility classes would likely consist of single objects and the set B of a constructed bireduct would likely correspond to a classical decision reduct.

One can easily formulate the representation analogous to Proposition 29, now for γ-decision bireducts considered in Algorithm 4. Certainly, the case of γ-decision bireducts is easier to handle than for decision bireducts.

Table 5.3: Indiscernibility classes induced by attributes $A^\diamond = \{T, H\}$ for decision table Table 3.1 processed in Algorithm 3.

	Temperature	**Humidity**	**Play**
1	hot	high	no
2	hot	high	no
3	hot	high	yes
13	hot	normal	yes
4	mild	high	yes
8	mild	high	no
12	mild	high	yes
14	mild	high	no
10	mild	normal	yes
11	mild	normal	yes
5	cool	normal	yes
6	cool	normal	no
7	cool	normal	yes
9	cool	normal	yes

Table 5.4: The decision table $\mathbb{A}_\diamond$ for $U^\diamond = \llbracket 1, 6, 8, 10, 13 \rrbracket$ in Algorithm 3. Decision reduct $\{T, H\}$ yields decision bireduct $(\llbracket 1, 2, 6, 8, 10, 11, 13, 14 \rrbracket, \{T, H\})$.

	Temperature	**Humidity**	**Play**
1	hot	high	no
6	cool	normal	no
8	mild	high	no
10	mild	normal	yes
13	hot	normal	yes

Table 5.5: The decision table $\mathbb{A}_\diamond$ for $U^\diamond = \llbracket 3, 6, 11, 12, 13 \rrbracket$ in Algorithm 3. Decision reduct $\{T\}$ yields decision bireduct $(\llbracket 3, 4, 6, 10, 11, 12, 13 \rrbracket, \{T\})$.

	Temperature	**Humidity**	**Play**
3	hot	high	yes
6	cool	normal	no
11	mild	normal	yes
12	mild	high	yes
13	hot	normal	yes

Algorithm 4 γ-Decision Bireduct Sampling Algorithm

```
input: decision table $\mathbb{A} = (U, A \cup \{d\})$
$U^\diamond \leftarrow \emptyset$
$A^\diamond \leftarrow$ an attribute subset of $A$
▷ $A^\diamond$ may be either passed as an input argument or chosen within the algorithm according to
  a selected strategy, such as random selection
for all $E \in U/A^\diamond$ do
    $u^\diamond \leftarrow$ a single object chosen from $E$
    $U^\diamond \leftarrow U^\diamond \cup \{u^\diamond\}$
$B \leftarrow$ arbitrary decision reduct $B \subseteq A^\diamond$ for table $\mathbb{A}^\gamma_\diamond = (U^\diamond, A^\diamond \cup \{d^\gamma_{A^\diamond}\})$
return $(POS_{\mathbb{A}}(B), B)$
```

Proposition 30. *Let the algorithm's input arguments $\mathbb{A} = (U, A \cup \{d\})$ and $A^\diamond \subseteq A$ be given. Then the outcome $(\mathcal{X}, B)$ of Algorithm 4 is a γ-decision bireduct for $\mathbb{A}$. Moreover, each γ-decision bireduct for $\mathbb{A}$ can be obtained as a result of Algorithm 4.*

Proof. The proof is a simpler version of the proof of Proposition 29. It is based on observation that in Algorithm 4, for a given $A^\diamond \subseteq A$, we actually construct a compact version of a γ-related table discussed in Section 3.2, now choosing a single representative object out of each indiscernibility class $E \in U/A^\diamond$ and using the decision value according to Equation (3.18) for $B = A^\diamond$. □

As previously, we illustrate the above procedure by presenting the intermediate steps in Table 5.6 and Table 5.7. Let us note that in the case of γ-decision bireducts, the selection of objects representing each indiscernibility class has no impact on the result obtained. Regardless of how we choose the representatives, the reduced decision table $\mathbb{A}_\diamond^\gamma$ will be the same (except to the numbers of the selected objects).

Table 5.6: Indiscernibility classes induced by attributes $A^\diamond = \{T, H\}$ for decision table Table 3.1 processed in Algorithm 4.

	Temperature	**Humidity**	**Play**
1	hot	high	no
2	hot	high	no
3	hot	high	yes
13	hot	normal	yes
4	mild	high	yes
8	mild	high	no
12	mild	high	yes
14	mild	high	no
10	mild	normal	yes
11	mild	normal	yes
5	cool	normal	yes
6	cool	normal	no
7	cool	normal	yes
9	cool	normal	yes

Table 5.7: $\mathbb{A}_\diamond^\gamma$ for $U^\diamond = [\![1, 6, 8, 10, 13]\!]$ in Algorithm 4. Decision reduct $\{T, H\}$ yields γ-decision bireduct $([\![10, 11, 13]\!], \{T, H\})$.

	Temperature	**Humidity**	**Play**
1	hot	high	⊛
6	cool	normal	⊛
8	mild	high	⊛
10	mild	normal	yes
13	hot	normal	yes

A comparison on computations of Algorithm 3 and Algorithm 4 for several exemplary input parameters is presented in Table 5.8. One of the main advantages of this approach in a comparison to the previously discussed algorithms is their computational efficiency. For a given selection of the attribute subset $A^\diamond$, it can be bounded in the worst case by $O(|A^\diamond| \cdot |U| + R)$, where R is the complexity of the algorithm used for the computation of a reduct. For instance, if the greedy heuristic is used [55], the overall computational complexity in the worst-case scenario would be $O(|A^\diamond| \cdot |U| + |A^\diamond|^2 \cdot |U| \cdot \log(|U|)) = O(|A^\diamond|^2 \cdot |U| \cdot \log(|U|))$. Of course, it is quite easy to select the attribute set $A^\diamond$ such that $|U^\diamond| << |U|$ and in practice, the computational cost of Algorithm 3 and Algorithm 4 would be considerably lower than the worst-case estimation – for small sets $A^\diamond$ it will be bounded by $O(|A^\diamond| \cdot |U|)$.

Table 5.8: The results of Algorithms 3 and 4 which run for several exemplary input parameters. In this example we assume that the algorithms use the ordering methods for obtaining the decision reducts and γ-decision reducts, i.e., the attributes are given in a specific order in which they are tried to be removed. In the presented example the input values consist of a subset of columns given as a permutation for the ordering algorithms and a permutation of objects to determine in which order the representative object will be chosen from the indiscernibility classes.

output from Algorithm 3 - decision bireduct	**output from Algorithm 4 - γ-decision bireduct**
input: object order = $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]$, attribute subset with order = $[W]$	
result: $(\llbracket 1, 2, 6, 8, 14 \rrbracket, \emptyset)$	result: $(\emptyset, \emptyset)$
input: object order = $[1, 3, 11, 6, 8, 9, 10, 13, 2, 4, 5, 7, 12, 14]$, attribute subset with order = $[W]$	
result: $(\llbracket 1, 7, 8, 11, 12 \rrbracket, \{W\})$	result: $(\emptyset, \emptyset)$
input: object order = $[12, 14, 3, 6, 4, 11, 9, 13, 1, 8, 2, 5, 7, 8, 10]$, attribute subset with order = $[W]$	
result: $(\llbracket 3..5, 7, 9..13 \rrbracket, \emptyset)$	result: $(\emptyset, \emptyset)$
input: objects order = $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]$, attribute subset with order = $[H, W]$	
result: $(\llbracket 1, 2, 5, 6, 8..10, 13, 14 \rrbracket, \{H, W\})$	result: $(\llbracket 5, 9, 10, 13 \rrbracket, \{H, W\})$
input: objects order = $[1, 3, 11, 6, 8, 9, 10, 13, 2, 4, 5, 7, 12, 14]$, attribute subset with order = $[H, W]$	
result: $(\llbracket 1, 2, 5, 7..11, 13, 14 \rrbracket, \{H\})$	result: $(\llbracket 5, 9, 10, 13 \rrbracket, \{H, W\})$
input: objects order = $[12, 14, 3, 6, 4, 11, 9, 13, 1, 8, 2, 5, 7, 8, 10]$, attribute subset with order = $[H, W]$	
result: $(\llbracket 3..6, 9, 10, 12, 13 \rrbracket, \{H, W\})$	result: $(\llbracket 5, 9, 10, 13 \rrbracket, \{H, W\})$
input: objects order = $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]$, attribute subset with order = $[T, H]$	
result: $(\llbracket 1, 2, 4, 5, 7, 9..13 \rrbracket, \{T, H\})$	result: $(\llbracket 10, 11, 13 \rrbracket, \{T, H\})$
input: objects order = $[1, 3, 11, 6, 8, 9, 10, 13, 2, 4, 5, 7, 12, 14]$, attribute subset with order = $[T, H]$	
result: $(\llbracket 1, 2, 6, 8, 10, 11, 13, 14 \rrbracket, \{T, H\})$	result: $(\llbracket 10, 11, 13 \rrbracket, \{T, H\})$
input: objects order = $[12, 14, 3, 6, 4, 11, 9, 13, 1, 8, 2, 5, 7, 8, 10]$, attribute subset with order = $[T, H]$	
result: $(\llbracket 3, 4, 6, 10..13 \rrbracket, \{T\})$	result: $(\llbracket 10, 11, 13 \rrbracket, \{T, H\})$
input: objects order = $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]$, attribute subset with order = $[T, H, W]$	
result: $(\llbracket 1, 2, 4..6, 9..13 \rrbracket, \{T, H, W\})$	result: $(\llbracket 2, 5, 9..11, 13 \rrbracket, \{T, H, W\})$
input: objects order = $[1, 3, 11, 6, 8, 9, 10, 13, 2, 4, 5, 7, 12, 14]$, attribute subset with order = $[T, H, W]$	
result: $(\llbracket 1, 2, 5, 6, 8..13 \rrbracket, \{T, H, W\})$	result: $(\llbracket 2, 5, 9..11, 13 \rrbracket, \{T, H, W\})$
input: objects order = $[12, 14, 3, 6, 4, 11, 9, 13, 1, 8, 2, 5, 7, 8, 10]$, attribute subset with order = $[T, H, W]$	
result: $(\llbracket 2..6, 9..13 \rrbracket, \{T, W\})$	result: $(\llbracket 2, 5, 9..11, 13 \rrbracket, \{T, H, W\})$
input: objects order = $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]$, attribute subset with order = $[O, T, H, W]$	
result: $(\llbracket 1..14 \rrbracket, \{O, H, W\})$	result: $(\llbracket 1..14 \rrbracket, \{O, H, W\})$
input: objects order = $[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14]$, attribute subset with order = $[W, H, T, O]$	
result: $(\llbracket 1..14 \rrbracket, \{O, T, W\})$	result: $(\llbracket 1..14 \rrbracket, \{O, T, W\})$

5.3 Dynamic Adjustment Heuristic

The sampling algorithm described in Section 5.2 can be summarized as consisting of the following three general steps:

1. select a subset of attributes by following some kind of heuristic,
2. for each of partition blocks induced by the obtained subset of attributes over the set of training objects, select a single representative (which practically means composing a decision rule by assigning a single decision class to a given combination of attribute values),
3. for the data subset of representatives produced this way, run an additional rough-set-based procedure of eliminating redundant attributes.

Flexibility of the approach let us adopt any heuristic searching for a proper subset of attributes at the first phase of the above procedure, e.g., a plain greedy heuristic maximizing gain with respect to the selected approximate function (see Definition 27 and eq. (3.23)) at each step or even more advanced like the *Dynamically Adjusted Approximate Reducts* heuristic (DAAR) [56]. The latter is the one we decided to implement and experiment with in [59]. The *dynamic adjustment* refers to the stopping criterion. Namely, we interrupt the process of attribute selection when we notice that the best attribute chosen from a randomly selected set of candidate attributes is not meaningful with high probability. We estimate such meaningfulness using a standard permutation test (repeated for a specified number of times), i.e., we create artificial attributes by shuffling the values of the considered candidate and we verify whether those shuffled attributes are less informative than the original one. Such an approach to randomizing attribute values in commonly employed in practical applications [14, 66].

It is worth mentioning that the DAAR-based strategy can be highly customized, e.g., we can parametrize:

- the number of candidates from which the best one will be selected in each iteration,
- the number of permutation test performed to estimate the meaningfulness of an attribute,
- the meaningfulness level below which we terminate the procedure

One more novelty that we introduce at the algorithmic level is a parameter deciding about the maximum number of attributes that can be selected at the first phase of the above process. This parameter is triggered if the DAAR-based stopping criterion does not break that phase before.

All phases of decision bireduct generation are randomized with respect to the choices of attributes and objects (representatives), whereby the corresponding randomizations are kept purposefully diversified when it comes to the construction of multiple bireducts to form a classifier ensemble.

Ultimately, the DAAR-based algorithm is presented in Algorithm 5. In the pseudocode of the algorithm, we can actually see the three steps mentioned at the beginning of this section marked appropriately with rectangles. Indeed, the layout of the algorithm is identical to that of Algorithm 3 with respect to the steps (2) and (3), while in the step (1) we put the code implementing the DAAR heuristic. Therefore, we can consider the algorithm to be just a specialized variant of Algorithm 3. Moreover, we can look at the sampling algorithm as a general framework on which a whole family of algorithms can be built – such the approach will be discussed later in Section 5.7.

Algorithm 5 Decision Bireduct DAAR-based Algorithm

```
input: decision table $\mathbb{A} = (U, A \cup \{d\})$
input: $N$ – maximum number of attributes
input: $DAAR\_params$ – parameters specific to the DAAR criterion
▷ iteratively select a subset of attributes using the DAAR stopping criterion
while $|A^\diamond| < N$ do
    $a \leftarrow$ the best attribute candidate chosen according to the selected informative gain heuristic
    ▷ the DAAR criterion – we interrupt the process of attribute selection when we notice that the
      best attribute candidate is not meaningful with high probability
    if $a$ does not meet the DAAR criterion (computed for the given $DAAR\_params$) then
        break
    $A^\diamond \leftarrow A^\diamond \cup \{a\}$
▷ select representatives
for all $E \in U/A^\diamond$ do
    $u^\diamond \leftarrow$ a single object chosen from $E$
    $U^\diamond \leftarrow U^\diamond \cup \{u^\diamond\}$
    $\mathcal{X} \leftarrow \mathcal{X} \cup (E \cap X^{\langle d(u^\diamond) \rangle})$
▷ run a procedure of eliminating redundant attributes
$B \leftarrow$ arbitrary decision reduct $B \subseteq A^\diamond$ for table $\mathbb{A}_\diamond = (U^\diamond, A^\diamond \cup \{d\})$
▷ prepare the result
return $(\mathcal{X}, B)$
```

5.4 Searching for Decision Bireduct Ensembles

The concept of decision bireducts has great potential for creating classifier ensembles [140, 119]. Such an approach can be efficient especially when the individual classifiers used in the ensemble are different from one another. For example, constructing ensembles of classifiers based on decision reducts that include diversified subsets of attributes may increase stability of classification and improve the ability to capture data interdependencies. However, the typical criteria for constructing approximate decision reducts do not allow to control which parts of data are problematic for particular subsets of attributes. For example, if we consider creating an ensemble consisted of approximate decision reducts that are supposed to correctly classify at least 90% of the training objects, we cannot anticipate that all of the resulting classifiers will have problems with the same 10% of objects.

In decision bireducts we have direct information about the set of uncovered objects. Therefore, we may construct an ensemble of classifiers which use different parts of data while uniformly covering the training cases. Moreover, an ensemble of decision bireducts or decision ε-bireducts (for a $\varepsilon \in [0, 1)$) can yield a group of shorter decision rules which do not work perfectly but still sufficiently help each other. We conducted a series of experiments to verify this claim, specifically investigating the usefulness of decision bireduct ensembles for classification purposes. We used three popular benchmark data sets from the UCI repository: *zoo*, *lymphography* and *SPECT* [71]. In our experiments we used decision bireducts and ordering-based algorithms, cf. Algorithm 1, however, γ-decision bireducts or other algorithms for searching decision bireducts could have been considered here too.

The generation of a decision bireduct ensemble with desired properties, such as the average number of attributes or uncovered objects, can be influenced by the method used to generate permutations. A special case is when all objects are at the beginning of $\sigma : \{1, \dots, |U| + |A|\} \to \{1, \dots, |U| + |A|\}$ – then, after $|U|$ steps of Algorithm 1, $\mathcal{X}$ becomes equal to U (or almost U, if $\mathbb{A}$ is inconsistent) and the process becomes similar to the search for standard decision reducts. A modification of the process leads to decision bireducts $(\mathcal{X}, B)$ with $\mathcal{X}$ being a bit smaller than U and B being significantly smaller than A – it is enough to increase probability that some attributes will occur closer to beginning of a permutation sequence. Let us introduce a *ratio* parameter that corresponds to the weight of attributes

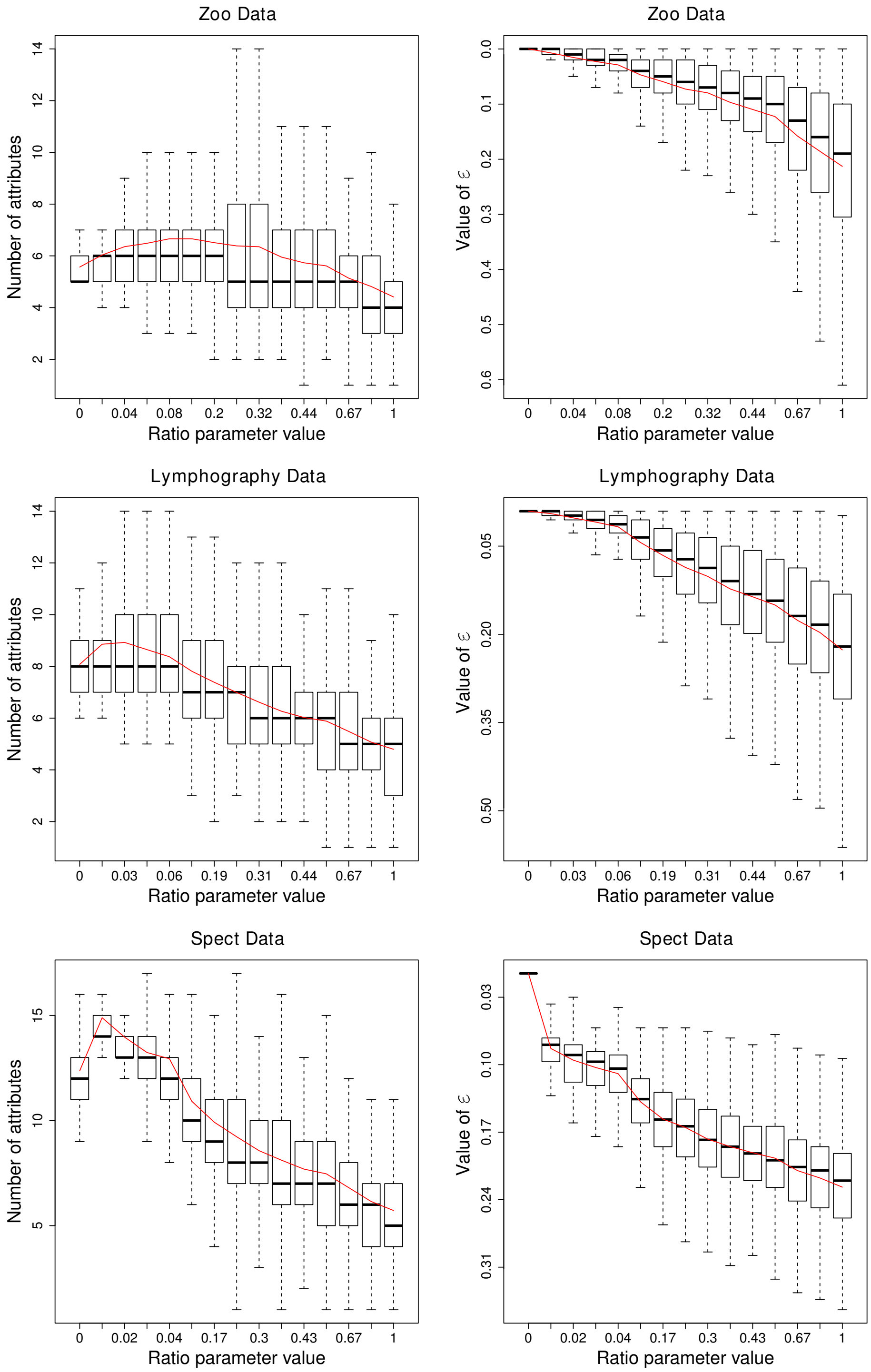


Figure 5.3: Quantiles and average values for the number of attributes (left) and the percentage of uncovered objects ε (right) in decision bireducts obtained for three benchmark data sets, for different values of the *ratio* parameter. Results for $ratio = 0$ correspond to standard decision reducts.

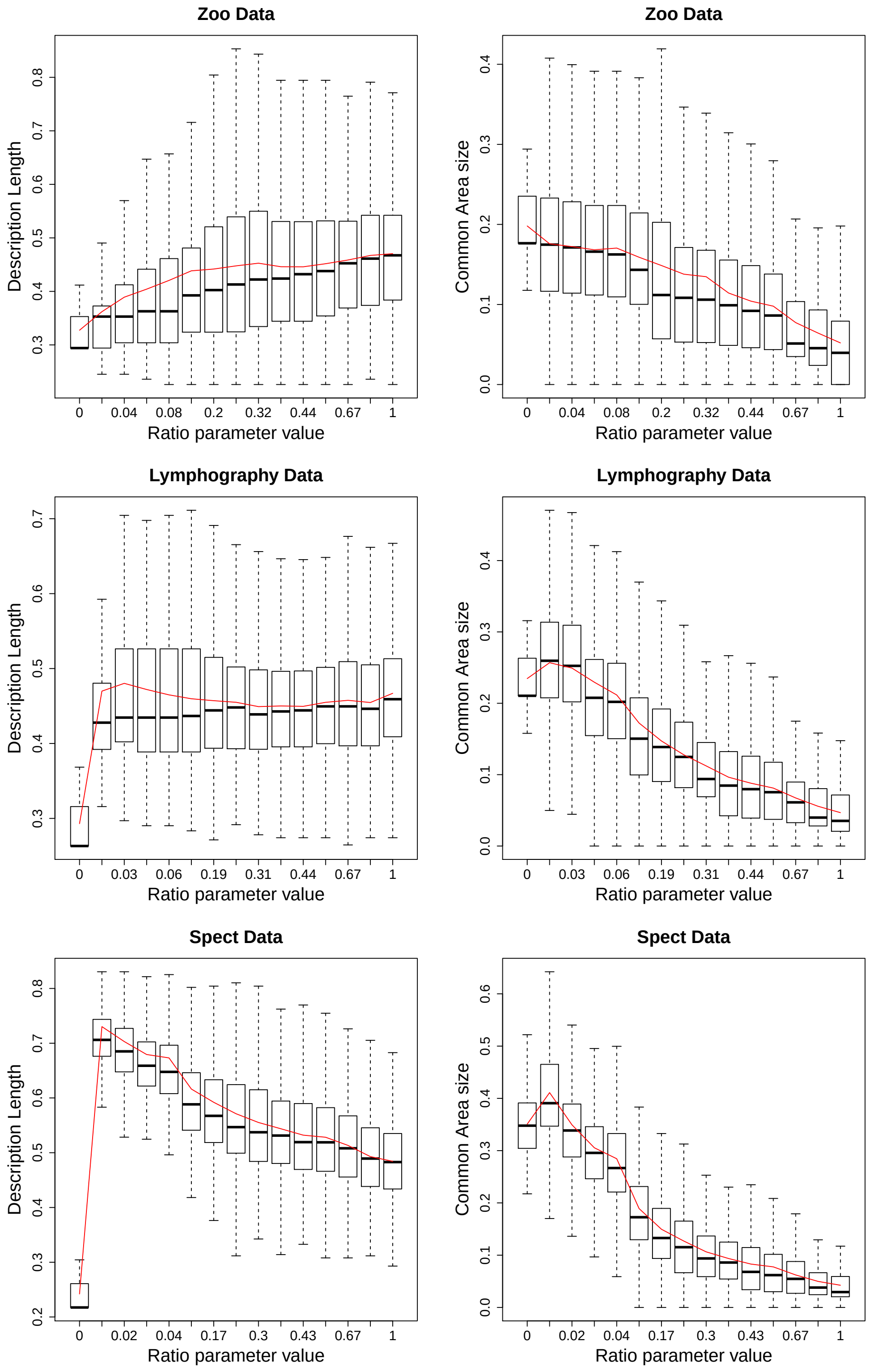


Figure 5.4: Quantiles and average values for description length (left) and intersection size (right) for decision bireducts of three benchmark data sets, for different values of the *ratio* parameter. Results for $ratio = 0$ correspond to standard decision reducts.

during the permutation construction – the higher the weight, the more attributes appear early in the sequence. If *ratio* value is equal to $|U|/|A|$, it results in a uniform distribution of attributes and objects within permutations. In experiments, we investigate the ratios spanning from 0 to $2 \cdot \frac{|U|}{|A|}$. To facilitate the comparison of results across various data sets, we further normalize the ratios to the interval of $[0, 1]$.

In our first experiment we compare decision bireducts with standard decision reducts in terms of their size. For each data set we computed 1000 decision reducts and 14000 decision bireducts for 14 different values of the *ratio* parameter – 1000 decision bireducts for each value. Figure 5.3 summarizes the number of attributes and the percentage of uncovered objects for different *ratio* values.

The average number of objects covered by a decision bireduct drops when the *ratio* is increased. In Figure 5.3 we present this tendency by means of the growth of the minimum values of ε (the solid line in the plots on the right) for which the obtained decision bireducts satisfy the properties of decision ε-bireducts. However, these observations do not hold for the average number of attributes (the solid line in the plots on the left) which slightly increases for small *ratio* values and then drops below the average for the standard reducts.

To further investigate the relation between the number of attributes and objects in decision bireducts we examined two indicators. The first one is the average decision bireduct description length defined using Equation (4.11). The second one is related to the overlap of the pairs of decision bireducts in an ensemble, defined as follows for a given pair $(\mathcal{X}_i, B_i)$ and $(\mathcal{X}_j, B_j)$:

$$OverlapSize((\mathcal{X}_i, B_i), (\mathcal{X}_j, B_j)) = \frac{|B_i \cap B_j|}{|A|} \cdot \frac{|\mathcal{X}_i \cap \mathcal{X}_j|}{|U|} \tag{5.1}$$

The average overlap size is related to the diversity of the decision bireducts in an ensemble. Intuitively, if for a given set of decision bireducts the average overlap size is small, then this set is more likely to cover a broader part of U. Figure 5.4 presents statistics for these two indicators, computed for different *ratio* values.

We investigated also the influence of the *ratio* parameter and classifier aggregation methods on the classification results. We carried out a total of 10 repetitions, where each repetition conssted of a 5-fold cross-validation test for each data set and *ratio* value. Therefore, to conduct the whole experiment and properly validate the results we computed 50000 decision reducts and 700000 decision bireducts. Decision rule sets corresponding to indiscernibility classes within decision bireducts were used as classifiers. To predict the decisions for objects in the test set, we employed two aggregation methods – majority voting and balanced support weighted voting. Such weighting strategies are quite popular in the machine learning literature, although we should emphasize that in the case of decision bireducts $(\mathcal{X}, B)$ they are computed with respect to $\mathcal{X}$ instead of U.

Decision classes in the data sets are imbalanced, therefore to evaluate our classifiers, we used two quality measures – mean accuracy and balanced accuracy. The mean accuracy is simply a percentage of correctly classified objects. The balanced accuracy is the mean percentage of correctly classified objects within each decision class. This measure is insensitive to imbalanced class distribution. The balanced accuracy gives more weight to instances from minority classes, whereas the mean accuracy treats all objects alike and usually favors the majority class.

The two different aggregation methods aim at maximizing different quality measures. The majority voting scheme classifies a test object to the decision class indicated by the highest number of triggered rules derived from decision bireducts in the ensemble. This voting scheme can be biased toward larger decision classes and usually favors the mean accuracy measure. More voting strategies for rule-based classifiers generated from decision reducts can be found, e.g., in [143, 125]. They can be adapted for decision bireducts too.

The balanced support weighted voting scheme weights each vote by a support of the corresponding rule. The class of an object is then decided by taking into account distribution of decisions in the training data. This method is preferable for maximizing the balanced accuracy. Figures 5.5 and 5.6 present the mean accuracy and balanced accuracy results obtained for the majority and the balanced support weighted voting schemes, respectively.

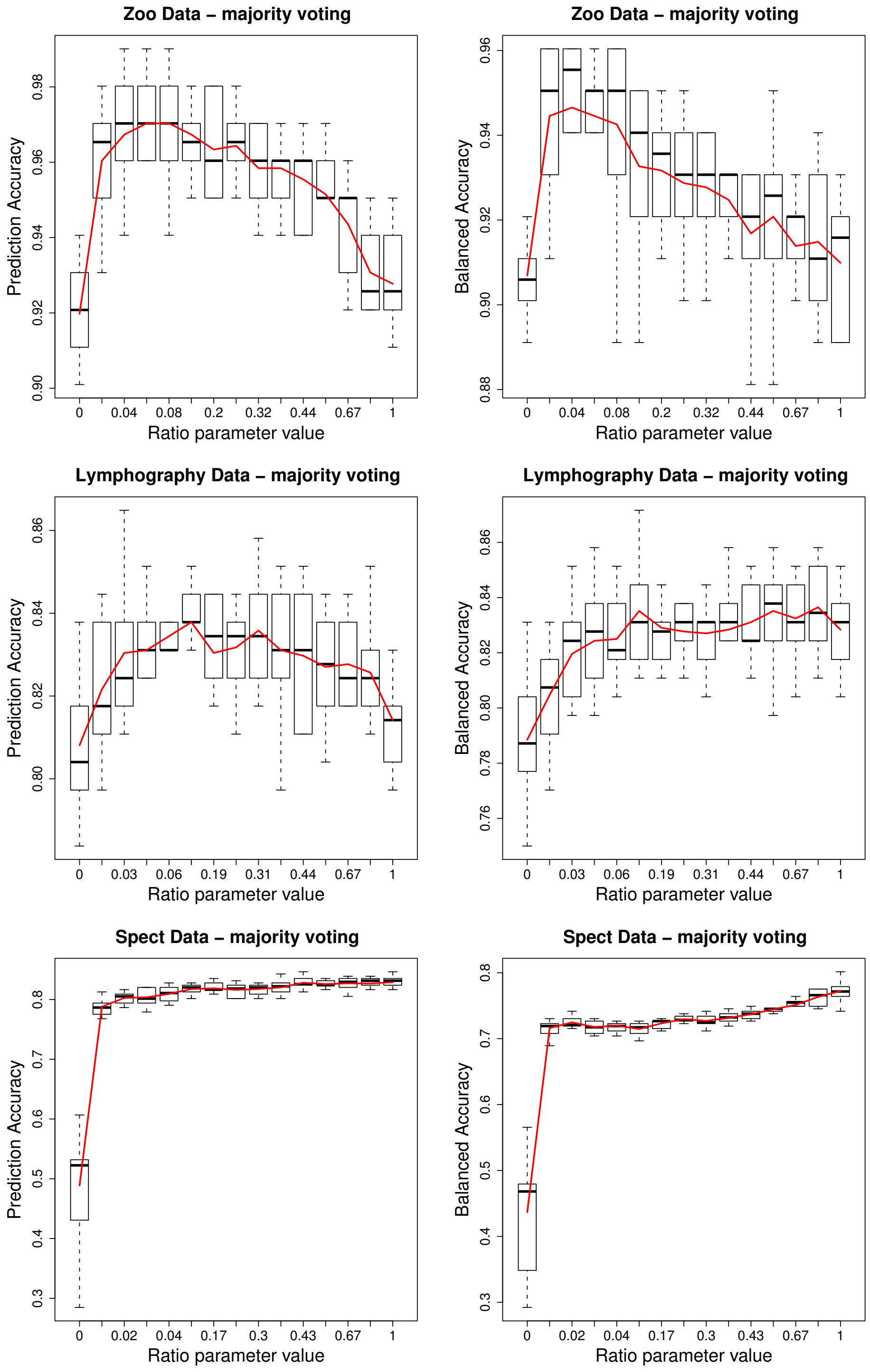


Figure 5.5: Classification (left) and balanced (right) accuracies of decision bireduct ensembles acquired by the majority voting, for different values of the *ratio* parameter. Results for $ratio = 0$ correspond to standard decision reducts.

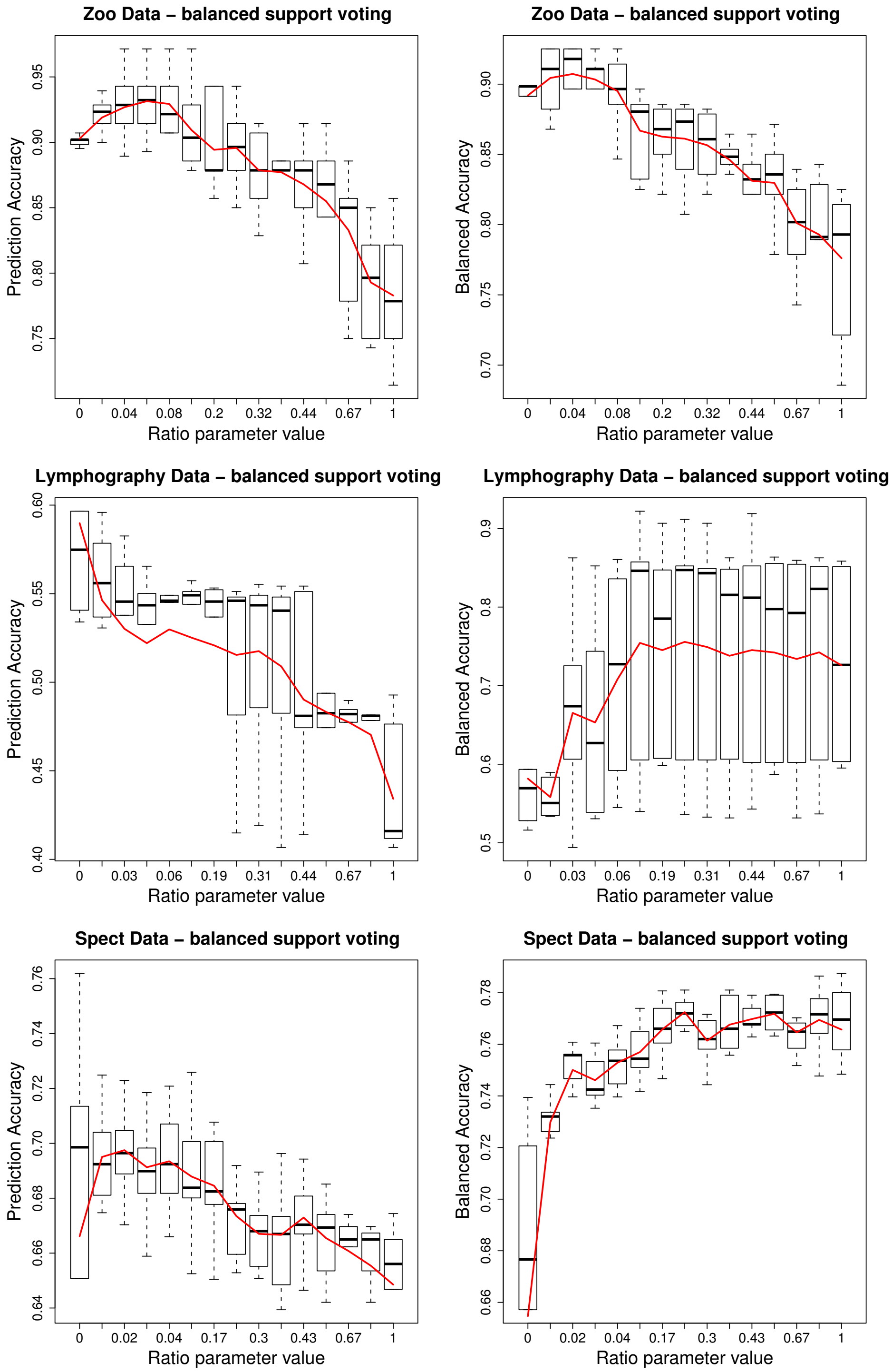


Figure 5.6: Classification (left) and balanced (right) accuracies of decision bireduct ensembles acquired by the balanced support weighted voting, for different values of the *ratio* parameter. Results for $ratio = 0$ correspond to standard decision reducts.

We also compared decision bireduct ensembles to other popular approaches in the machine learning literature [34, 130]. In particular, Table 5.9 includes comparison of accuracy and balanced accuracy results obtained for random forest (1000 trees, default settings from *randomForest* package of R System [106]), bagged logistic model ensemble and decision bireduct ensembles (using default settings described in [140]).

Table 5.9: A comparison of average results obtained for ensembles of decision bireducts and two other ensemble-based classification techniques, i.e., random forest and bagged logistic regression models. Mean and standard deviation of the results are given.

data set	RF	BaggedLog	Decision Bireducts
Accuracies (ACC)			
zoo	$0.78 \pm .01$	$0.96 \pm .02$	$0.97 \pm .01$
lymphography	$0.76 \pm .02$	$0.81 \pm .02$	$0.83 \pm .02$
SPECT	$0.75 \pm .01$	$0.82 \pm .01$	$0.82 \pm .01$
Balanced Accuracies (BAC)			
zoo	$0.77 \pm .01$	$0.90 \pm .05$	$0.93 \pm .03$
lymphography	$0.88 \pm .01$	$0.72 \pm .08$	$0.83 \pm .01$
SPECT	$0.75 \pm .01$	$0.70 \pm .01$	$0.74 \pm .01$

To summarize, regardless of the aggregation method and the quality measure used, for low values of the *ratio* parameter, the decision bireduct classifiers consistently outperformed the ensembles of standard reducts (the *ratio* value set to 0). Moreover, when compared to other commonly used classification algorithms, decision bireduct ensembles not only achieved comparable results but also, in some cases, outperformed them. Surely, there are many other studies showing that approximate attribute reduction and rough set decision models also lead toward efficient ensembles of rule-based classifiers [51, 9]. Nevertheless, we believe that having a direct control over the set of covered objects may allow us to develop better ensemble learning strategies in future.

5.5 Decision Bireducts in Data Streams

The main motivation for introducing decision bireducts in [140] was to establish a simple framework for constructing rough-set-based classifier ensembles, as well as to extend capabilities of decision reducts to model data dependencies. Going further, in [89] it was noticed that algorithms for extracting meaningful bireducts from data could be utilized to integrate the tasks of attribute and instance selection. Such a potential is also illustrated by Algorithm 3 and Proposition 29, where the objects in $U^{\diamond}$ selected as representatives of equivalence classes induced by the subset of attributes actually define a classifier based on the resulting $B \subseteq A$. Therefore, one can partition $\mathcal{X}$ with respect to its elements' values on B and treat combinations of values labeling partition classes as antecedents of rules pointing at specific decision values, uniquely defined within $\mathcal{X}$ due to the bireduct properties.

Some areas of applications were also pointed out for other types of bireducts. In [58], so called information bireducts were employed to model context-based object similarities in multi-dimensional data sets. Information bireducts may be also able to approximate data complexity analogously to some well-known mathematical tools [10]. Indeed, by investigating cardinalities of minimal subsets of attributes discerning maximal subsets of objects we can attempt to express a potential of a data source to define different concepts of interest.

In this section, we study one more potential application of bireducts. Let us consider a stream of objects that is too large to be stored or represents data collected online [1]. For our purposes, let us focus on a stream interpreted as a decision system $\mathbb{A} = (U, A \cup \{d\})$, where there is no possibility to look at the entire U at any moment of processing time. Instead, given a natural order over U, we can access some buffered data intervals, i.e., the subsets of objects that occur consecutively in a stream. The question is how to design and efficiently conduct a process of attribute reduction in such a dynamic situation.

One of the ways would be to fix the amount of objects in each data interval and compare decision reducts obtained for such narrowed down decision systems, in a kind of sliding window fashion. However, an arbitrary choice of the interval length may significantly influence the results. Thus, it may be more reasonable to adaptively adjust data intervals with respect to the currently observed attribute dependencies. Moreover, if our goal is to search for stable subsets of attributes that remain decision reducts for possibly wide areas of data, then we should tend to maximizing data intervals in parallel to minimizing the amounts of attributes necessary to determine decision classes within them.

Definition 42. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Let U be naturally ordered with its elements indexed by integers. Consider a pair $(\mathcal{X}, B)$, where $B \subseteq A$ and $\mathcal{X} = \{u_i \in U | i \in \{first..last\}\}$ is a subset of objects consisting of a continuos range of objects (with respect to their natural order) starting from the object u_{first} ending with the object u_{last} – or, $\mathcal{X} = [\![first..last]\!]$ for short. We say that $(\mathcal{X}, B)$ is a temporal decision bireduct if and only if the following conditions hold:*

- *An inexact functional dependency $B \Rrightarrow_{\mathcal{X}} d$ holds;*
- *There is no $C \subsetneq B$ such that $C \Rrightarrow_{\mathcal{X}} d$;*
- *$B \Rrightarrow_{\mathcal{Y}} d$ is not true for neither $\mathcal{Y} = [\![(first-1)..last]\!]$ nor $\mathcal{Y} = [\![first..(last+1)]\!]$.*

The above modification of Definition 34 can serve as a background for producing bireducts $(\mathcal{X}, B)$ with no *gaps* in $\mathcal{X}$ with respect to a given data flow, e.g., when we do not know the whole dataset upfront or it does not fit in memory but can be handled as a form of a stream of objects appearing one by one. Below we sketch an example of heuristic extraction of such bireducts from data. From a technical point of view, it resembles Proposition 29 with respect to a random choice of a subset of attributes to be analyzed. From a more strategic perspective, let us note that our goal is now to save the identified temporal bireducts analogously to micro-clusters [128] or data blocks [141] constructed within other applications for the purposes of further steps of online or offline analysis. This way of data stream processing may open new opportunities for the task of scalable attribute subset selection. For instance, basing on frequent occurrence of a given subset of attributes in the previously-found temporal bireducts, one can reason about its ability to induce a robust decision model.

Consider $(\mathcal{X}, B)$, where $\mathcal{X}$ is a buffer of objects that occurred most recently in a data stream. If the next $u \in U$ is contradictory with $\mathcal{X}$ subject to B, we can remove the oldest contradictory objects from $\mathcal{X}$ and/or add some attributes to B to be able to add u. If the next u can be added to $\mathcal{X}$ subject to B, we can just extend $\mathcal{X}$ or we can decrease B in order to avoid too rapid growth of $\mathcal{X}$, e.g., we have a limited amount of space for a buffer storing the currently processed objects. The whole approach allows for flexible customization of the data stream processing depending on what are the expected properties of the temporal bireducts obtained as a result.

Proposition 31. *Let $\mathbb{A} = (U, A \cup \{d\})$ be given. Let U be naturally ordered with its elements indexed by integers. Select an arbitrary subset of attributes $A' \subseteq A$ and put $\mathcal{X} = \emptyset$ and $B = \emptyset$. Consider the following steps for each consecutive i-th object in U:*

1. *If $B \Rrightarrow_{\mathcal{X} \cup \{u_i\}} d$, then add u_i to $\mathcal{X}$;*
2. *Else, save $(\mathcal{X}, B)$, add u_i to $\mathcal{X}$, and proceed as follows:*
   (a) *Put $B = A'$ and remove the oldest objects from $\mathcal{X}$ until there is $B \Rrightarrow_{\mathcal{X}} d$;*
   (b) *Heuristically reduce redundant attributes under the constraint $B \Rrightarrow_{\mathcal{X}} d$.*

Then, all pairs $(\mathcal{X}, B)$ saved during the above procedure are temporal bireducts for $\mathbb{A}$. Moreover, each temporal bireduct can be obtained as one of saved pairs $(\mathcal{X}, B)$ for some $A' \subseteq A$, no matter what method is used in the last step.

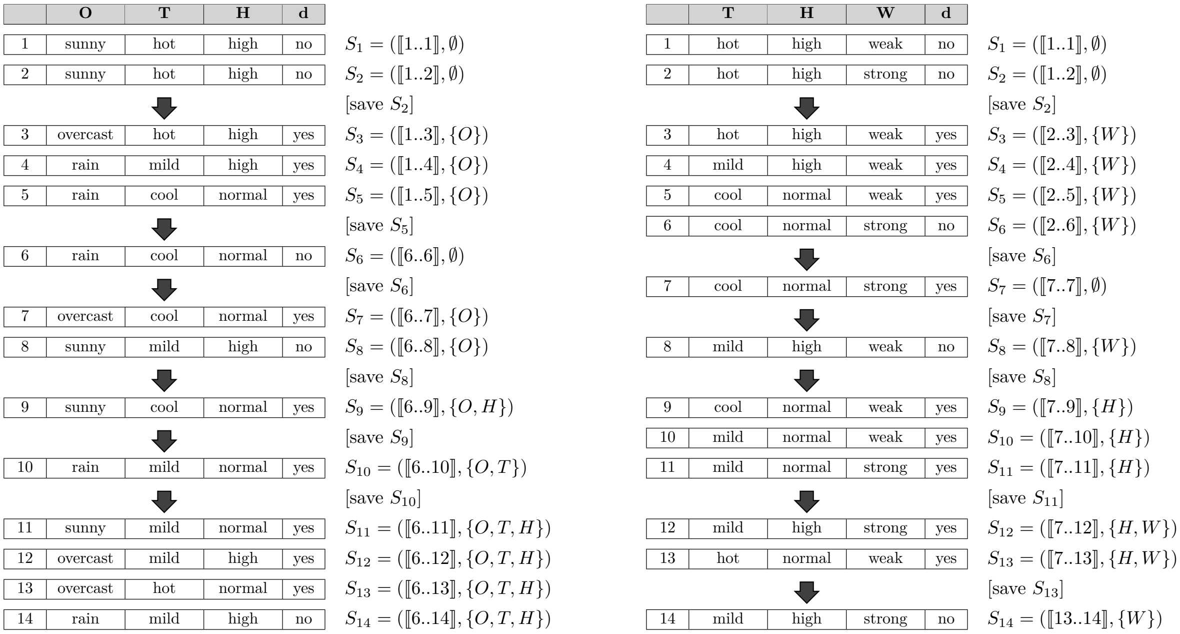
Proof. Consider a pair $(\mathcal{X}, B)$, where $\mathcal{X} = \llbracket first..last \rrbracket$, which was saved in step (2). For such a case, we know that $B \Rrightarrow_{\llbracket first..last \rrbracket} d$ and $B \not\Rrightarrow_{\llbracket first..(last+1) \rrbracket} d$. Also, there is $B \not\Rrightarrow_{\llbracket (first-1)..last \rrbracket} d$ because the oldest object in $\mathcal{X}$ is removed only when the newly joined object cannot be handled together with some elements of $\mathcal{X}$ even when using the whole A'. Therefore, $\mathcal{X}$ cannot be extended backwards beyond object u_{first}. Also, because of reduction of redundant attributes, B is irreducible for $\mathcal{X}$. Hence, all saved pairs $(\mathcal{X}, B)$ are temporal bireducts.

Now, consider a temporal bireduct $(\llbracket first..last \rrbracket, B)$ and put $A' = B$. Consider the first buffer including object u_{first}, i.e., $\llbracket older..first \rrbracket$, where $older \leq first$. Each next entry until object u_{last} will be added with no need of removing u_{first} (otherwise there would be no $B \Rrightarrow_{\llbracket first..last \rrbracket} d$). Moreover, when adding u_{last}, all objects older than u_{first} (if any of them are still present) will be erased from the buffer (otherwise there would be $B \Rrightarrow_{\llbracket (first-1)..last \rrbracket} d$). Finally, when adding object u_{last+1} to $\llbracket first..last \rrbracket$, we will need to remove u_{first} (otherwise there would be $B \Rrightarrow_{\llbracket first..(last+1) \rrbracket} d$), which results in saving $(\llbracket first..last \rrbracket, B)$. □

In Proposition 31, subsets $\mathcal{X} \subseteq U$ are treated as the buffers of objects that appeared most recently in a data stream, within which a currently considered $B \subseteq A$ is sufficient to determine decision classes.

One needs to remember that the above algorithm describes just one of many possible techniques of extracting bireducts from data streams. Both, strategies of retrieving the inexact functional dependency and heuristic approaches used in Proposition 31, may vary depending on the data source or desirable results, e.g., temporal bireducts of particular size or preferred ratio between cardinalities of $\mathcal{X}$ and B.

As an illustration, consider the data set presented in Table 3.1 and assume that we receive objects from $U = \{u_1, \ldots, u_{14}\}$ one after the other. Let the i-th state of the process be denoted by $S_i = (\mathcal{X}_i, B_i)$, where i is the number of objects already received from U and B_i is a decision reduct for the current buffer content $\mathcal{X}_i$.



	O	T	H	d	
1	sunny	hot	high	no	$S_1 = (\llbracket 1..1 \rrbracket, \emptyset)$
2	sunny	hot	high	no	$S_2 = (\llbracket 1..2 \rrbracket, \emptyset)$
		⬇			[save S_2]
3	overcast	hot	high	yes	$S_3 = (\llbracket 1..3 \rrbracket, \{O\})$
4	rain	mild	high	yes	$S_4 = (\llbracket 1..4 \rrbracket, \{O\})$
5	rain	cool	normal	yes	$S_5 = (\llbracket 1..5 \rrbracket, \{O\})$
		⬇			[save S_5]
6	rain	cool	normal	no	$S_6 = (\llbracket 6..6 \rrbracket, \emptyset)$
		⬇			[save S_6]
7	overcast	cool	normal	yes	$S_7 = (\llbracket 6..7 \rrbracket, \{O\})$
8	sunny	mild	high	no	$S_8 = (\llbracket 6..8 \rrbracket, \{O\})$
		⬇			[save S_8]
9	sunny	cool	normal	yes	$S_9 = (\llbracket 6..9 \rrbracket, \{O, H\})$
		⬇			[save S_9]
10	rain	mild	normal	yes	$S_{10} = (\llbracket 6..10 \rrbracket, \{O, T\})$
		⬇			[save S_{10}]
11	sunny	mild	normal	yes	$S_{11} = (\llbracket 6..11 \rrbracket, \{O, T, H\})$
12	overcast	mild	high	yes	$S_{12} = (\llbracket 6..12 \rrbracket, \{O, T, H\})$
13	overcast	hot	normal	yes	$S_{13} = (\llbracket 6..13 \rrbracket, \{O, T, H\})$
14	rain	mild	high	no	$S_{14} = (\llbracket 6..14 \rrbracket, \{O, T, H\})$

	T	H	W	d	
1	hot	high	weak	no	$S_1 = (\llbracket 1..1 \rrbracket, \emptyset)$
2	hot	high	strong	no	$S_2 = (\llbracket 1..2 \rrbracket, \emptyset)$
		⬇			[save S_2]
3	hot	high	weak	yes	$S_3 = (\llbracket 2..3 \rrbracket, \{W\})$
4	mild	high	weak	yes	$S_4 = (\llbracket 2..4 \rrbracket, \{W\})$
5	cool	normal	weak	yes	$S_5 = (\llbracket 2..5 \rrbracket, \{W\})$
6	cool	normal	strong	no	$S_6 = (\llbracket 2..6 \rrbracket, \{W\})$
		⬇			[save S_6]
7	cool	normal	strong	yes	$S_7 = (\llbracket 7..7 \rrbracket, \emptyset)$
		⬇			[save S_7]
8	mild	high	weak	no	$S_8 = (\llbracket 7..8 \rrbracket, \{W\})$
		⬇			[save S_8]
9	cool	normal	weak	yes	$S_9 = (\llbracket 7..9 \rrbracket, \{H\})$
10	mild	normal	weak	yes	$S_{10} = (\llbracket 7..10 \rrbracket, \{H\})$
11	mild	normal	strong	yes	$S_{11} = (\llbracket 7..11 \rrbracket, \{H\})$
		⬇			[save S_{11}]
12	mild	high	strong	yes	$S_{12} = (\llbracket 7..12 \rrbracket, \{H, W\})$
13	hot	normal	weak	yes	$S_{13} = (\llbracket 7..13 \rrbracket, \{H, W\})$
		⬇			[save S_{13}]
14	mild	high	strong	no	$S_{14} = (\llbracket 13..14 \rrbracket, \{W\})$

Figure 5.7: Extraction of temporal bireducts from a data set in Table 3.1. The left/right-hand side sequences correspond to subsets $A' = \{O, T, H\}$ and $A' = \{T, H, W\}$, respectively.

Figure 5.7 presents two examples of arbitrarily chosen subsets of attributes. Let us concentrate on $A' = \{T, H, W\}$ and refer one more time to the order-based characteristics of decision reducts outlined, e.g., in [3]. Namely, in the step (2b) of Proposition 31, we are going to reduce attributes along $\sigma = [T, H, W]$. In general, when following the same $\sigma : \{1, \ldots, n'\} \to \{1, \ldots, n'\}$, $n' = |A'|$, from the very beginning of a data stream, we can count on smoother evolution of subsets $B_i \subseteq A'$ for consecutive buffers. We can expect this because the fixed order of attributes used in the algorithm for the whole run will naturally prefer some attributes over others. On the other hand, by working with a larger family of diversified subsets $A' \subseteq A$, we have a chance to witness the most representative changes of the observed temporal bireducts in time.

Let us now take a closer look at $A' = \{T, H, W\}$. The first two objects share the same decision. Thus, there is $S_2 = (\llbracket 1..2 \rrbracket, \emptyset)$. Further, since $\emptyset \not\Rightarrow_{\llbracket 1..3 \rrbracket} d$, we save the temporal bireduct $(\llbracket 1..2 \rrbracket, \emptyset)$ and proceed with step (2) of the procedure from Proposition 31. As $\{T, H, W\}$ is insufficient to discern objects u_1 and u_3, we limit ourselves to $\llbracket 2..3 \rrbracket$. Starting from $B = A'$ and given $\sigma = [T, H, W]$, we reduce T and H, which results in the pair $S_3 = (\llbracket 2..3 \rrbracket, \{W\})$. The next three objects do not break the dependency between $\{W\}$ and d. However, object u_7 forces all earlier entries to be removed. A different situation can be observed when adding the next two objects. In both cases, A' determines decision values, so we can keep buffers $\llbracket 7..8 \rrbracket$ and then $\llbracket 7..9 \rrbracket$. However, subsets of attributes generated using the same σ will differ from each other. $\{W\}$ is not able to determine d within $\llbracket 7..9 \rrbracket$ although it was sufficient for $\llbracket 7..8 \rrbracket$. As a consequence, we need to restart from $B = A'$. We are allowed to remove T. Then, H turns out to be irreducible because of a need of keeping discernibility between objects u_8 and u_9. Given the fact that H was not removed, W is not important any more, resulting in $S_9 = (\llbracket 7..9 \rrbracket, \{H\})$. Another two objects cause no problems and $S_{11} = (\llbracket 7..11 \rrbracket, \{H\})$. After u_{12} arrives we can see that $\{H\}$ is insufficient to determine d within $\llbracket 7..12 \rrbracket$, thus we need to start again with step (2) of the procedure. As $\{T, H, W\}$ is sufficient to determine d within $\llbracket 7..12 \rrbracket$ we do not need to remove any objects at this point. Starting with $B = A'$, this time we can only reduce attribute T, getting $S_{12} = (\llbracket 7..12 \rrbracket, \{H, W\})$. The next object u_{13} does not cause conflicts and can be added without issues. Finally, the last object from our example u_{14} conflicts with the object u_{12} (even using the whole $A' = \{T, H, W\}$), thus forcing removal of the oldest objects $\llbracket 7..12 \rrbracket$. Once again we start with $B = A'$ and we are able to remove T and H as the attribute W alone is sufficient to discern objects u_{13} and u_{14} having different decision value. Our procedure finally ends with $S_{14} = (\llbracket 13..14 \rrbracket, \{W\})$.

5.6 Software Library

A Python software library *scikit-rough* (`https://github.com/sebov/scikit-rough`) hosted on GitHub platform is released as a result of our research. It is available to users under the permissive MIT license. The library provides a range of functions and algorithms associated with rough set theory, encompassing structures and methods for handling, analyzing, processing data, and enabling the application of techniques related to machine learning.
To install the newest version available in PyPI repository [1] (as of this writing, this is `0.1.0`), please use the following command:

```
pip install scikit-rough
```

For the exact version referred above, please use:

```
pip install scikit-rough==0.1.0
```

The library is currently in a phase of development where interface stability is not its primary focus. Instead, it is oriented toward introducing new features, experimentation, preparation of prototypes of various solutions, including the framework for working with rough-set-based algorithms in a structural-like pattern. If the actual instructions on the library usage were written here, they could quickly become outdated, therefore, we suggest that readers to refer to the automatically generated online documentation available at `https://scikit-rough.readthedocs.io` – for viewing the API reference

[1] `https://pypi.org/` - The Python Package Index is a repository of software for the Python programming language

and examples. However, to demonstrate some functionalities of the library through examples, we have included a few Jupyter Notebooks in Appendix B. Moreover, in Section 5.7 we present also a short description of one of its functionalities, which allows employing a structural pattern to the construction of algorithms.

We invite anyone who is interested to install the library, conduct experiments, review the source code, and report any issue they may encounter. All contributions and feedback are welcome.

5.7 Multi-Stage Processing Framework

Based on our research and numerous discussions, we come to this conclusion that the algorithms we use in various experiments related to decision reducts, approximate decision reducts, and decision bireducts often share a similar pattern. Namely, in practical applications, for performance reasons, the common steps were to obtain a superset of significant attributes (the “grow” phase) and at some point to reduce it keeping the given criterion (the “shrink” phase). Sometimes, these were extended with other intermediate steps, e.g., drawing objects in the process of generating decision bireducts. Similar approaches, in different variants, we can observe in multiple situations:

- In the general form of the decision bireduct sampling algorithm, namely Algorithm 3 we can distinguish the phases of obtaining an attribute subset and the reduction phase at the end.

- In Algorithm 4 we have similar situation but in the context of other reduction method.

- In Algorithms 1 and 2 we perform the “grow” phase for the subsets of objects and “shrink” phase for the subsets of attributes simultaneously.

- We can notice similar operation scheme in the greedy heuristic for computing decision reducts as well as in relation to the DAAR algorithm [56],

- In Algorithm 5 (DAAR-based) for computing decision bireducts which is presented in Section 5.3 the pattern is visible as well.

All of the above allow us to propose the following more general solution presented in Figure 5.8 and implemented within the *scikit-rough* software library (see Section 5.6) as a structural framework for constructing algorithms, e.g., those falling under the “grow-shrink” category. A Jupyter Notebook example presenting the ideas on how to use the framework can be seen on page 178.

An algorithm expressed using the framework consists of the following elements, which are usually optional and compositional (i.e., they can be assembled from the available or ad-hoc implemented hook functions) placeholders:

- the *initialization* placeholder, in which one or more functions can be invoked to prepare the internal state of the run, e.g., we can preprocess the data, prepare internal indexes, compute an approximation threshold from data, etc.,

- zero or more *stages*, that may correspond to some larger subtasks (e.g., “grow” or “shrink” phases mentioned above),

- the *finalize* placeholder, that may correspond to some operations preceding the preparation of the result,

- the *prepare-result* placeholder, used to prepare the result of the given run, e.g., for decision-reduct-based algorithm, it should be just a subset of attributes A, but for decision-bireduct-based algorithms, this should be a pair $(\mathcal{X}, B)$.

Going forward, each stage is implemented using two nested loops. The outer loop is just an infinite loop that can only be interrupted by the occurrence of a defined stop condition. Within each iteration we process some generally understood elements (e.g., representing attributes of a data set) obtaining a certain set of elements relevant in a given iteration (e.g., the best attribute candidate, like in Algorithm 5). Then, the inner loop processes those elements, updating the internal structures of the whole run, e.g., extending the set of resulting attributes. Within each stage we can define the following, usually optional and compositional, placeholders:

- the *initialization* placeholder, used to initialize a given stage run, e.g., generating an appropriate permutation for Algorithm 1,
- the *stop-check* placeholder, defining an appropriate stop criterion, e.g., in "grow" phase we reached a given maximum number of attributes, or the DAAR criterion does not hold for the best candidate attribute, cf. Section 5.3,
- the *pre-candidates* placeholder, to define functions "producing" elements that will be processed by the latter steps, e.g., to determine which attributes are still available for selection,
- the *candidates* placeholder, to define functions "pre-filtering" the elements, e.g., limiting the number of candidate attributes to the given parameter, cf. the DAAR algorithm [56],
- the *selected* placeholder, to define functions "selecting" the best candidate elements, e.g., selecting one or more attributes with the highest information gain,
- the *filtered* placeholder, to define functions "post-filtering" the best candidate elements, e.g., applying the DAAR criterion as described in Section 5.3,
- the *inner-** placeholders, to define functions that process the elements which have survived all earlier steps, e.g., adding the best attribute to the current set of selected attributes, or in the context of the attribute reduction, the inner loop will be responsible for the removal of attributes from the current set of resulting attributes and update of the internal structures,
- the *finalize* placeholder, to define functions updating the internal state at the end of the given stage run.

Overall, the multi-stage processing approach should be understood in terms of a Proof of Concept. It was prepared to demonstrate its feasibility and to verify if it has practical potential. On the one hand, it may seem complicated and require some overhead when designing solutions. On the other hand, it also has several advantages, i.e., reusability of individual hook functions as well as entire composed stages. Moreover, the imposed structure allows for runtime inspection, visualization, documentation and parameter verification based on its components – see a Jupyter notebook demonstrating a simple usage example on page 178.

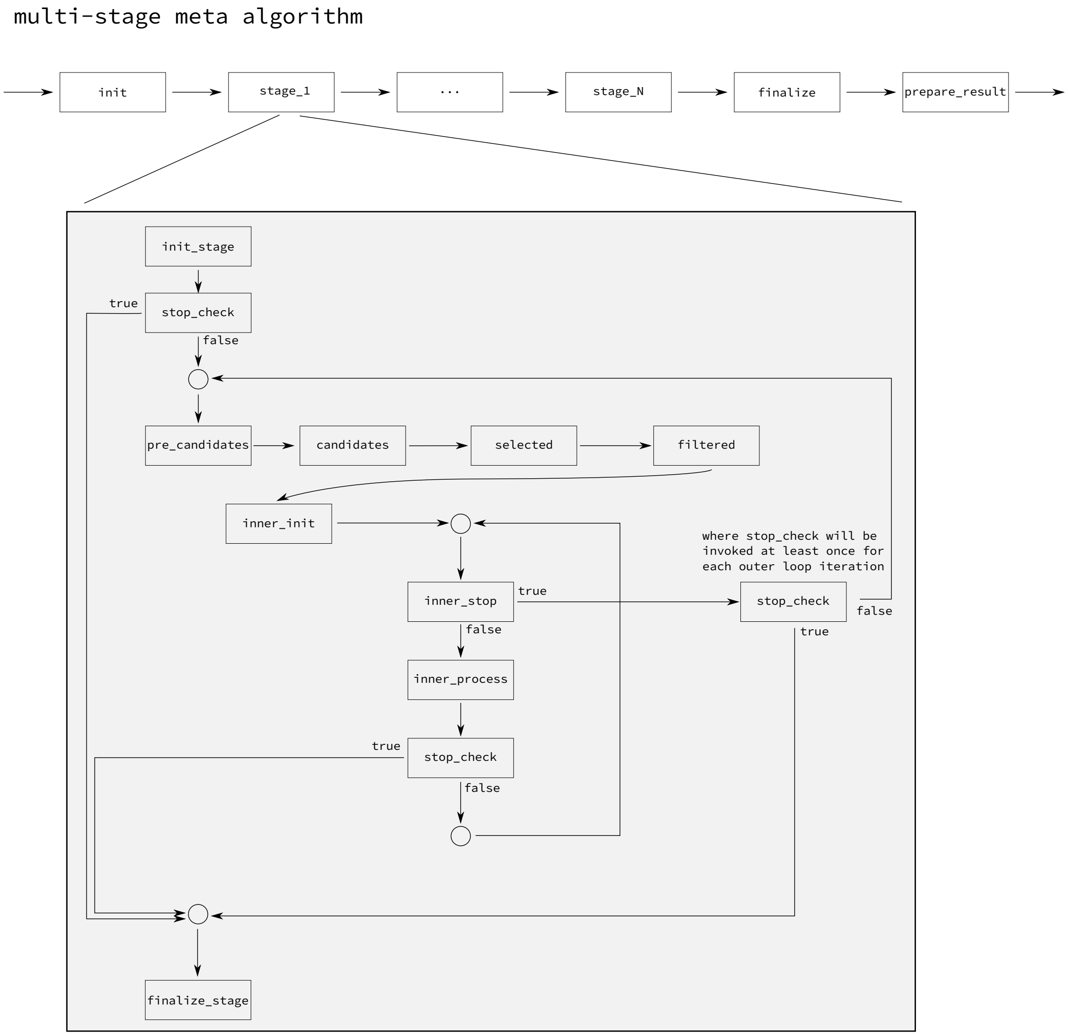


Figure 5.8: Multi-stage processing approach.

Chapter 6

Case Study Related to HR Industry

In this chapter our aim has been to apply the already developed ensembles of decision bireducts in the context of a real life problem related to recruitment processes from HR industry. Our case study refers to the data gathered in the system owned and used by the Toolbox for HR[1] (tb4hr) which is a recruitment process outsourcing company. The proposed system is responsible for continuous data acquisition and analytics. We designed it in cooperation with tb4hr's subject matter experts (SMEs), both with respect to its functionality and the shape of the data. In such companies, the sourcers, i.e., people who search for job applicants and convince them to participate in a recruitment process, need to contact hundreds of candidates in order to find only several who are likely to change their current employer.

Some of the results presented in this chapter have already been published previously [52]. Moreover, the further research from the continuation of this work, is submitted for publication [59].

6.1 Recruitment System

The system is deployed in a production environment with the task of scoring the likelihood that a given person would change employment. There are two modes of the system's work, namely *external* and *in-house*. The *in-house* mode enables us to explore the historical data collected by tb4hr's sourcers in order to improve the efficiency of new recruitment processes. It is the basis for the following process which takes place at tb4hr with respect to every new job offer:

1. deriving the most similar historical offers,
2. fetching the profiles of relevant candidates,
3. constructing a scoring model,
4. computing candidate profile scores.

The *external* mode is devoted to the external recruitment agencies which use the Chrome-browser-extension-based service delivered by tb4hr. As such agencies are usually not willing to share descriptions of their campaigns and especially job offers; the above process is simplified in this case.

The architecture of the tb4hr's system is depicted in Figure 6.1 and it was discussed for the first time in [52]. It is flexible and allows us to perform different types of data analyses. For instance, a data analyst can easily use it to explore historical data in a search for meaningful dependencies, try to select the most important attributes or visualize the data for the sourcers in the hope of providing them with some practical insights. For the *external* mode, some parts of the architecture are not used

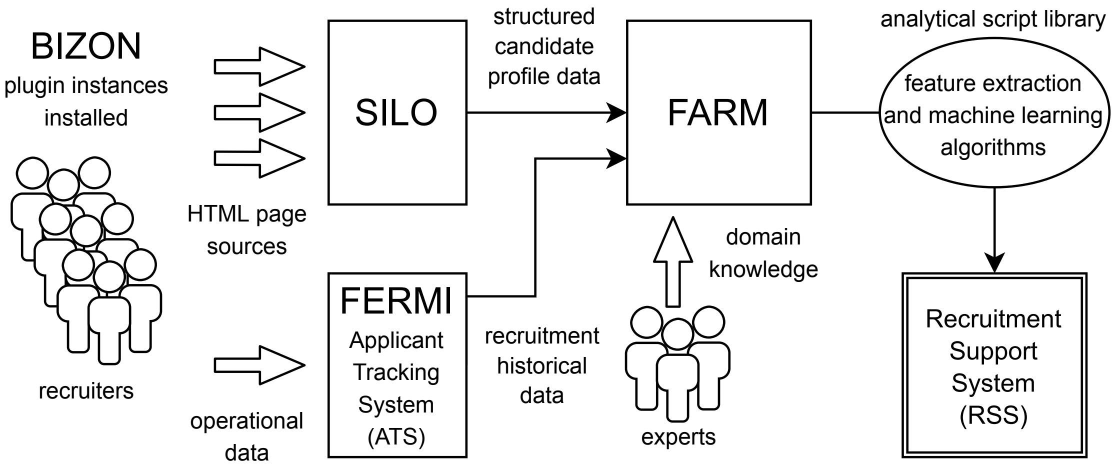

BIZON
plugin instances installed
recruiters
HTML page sources
operational data
SILO
FERMI
Applicant Tracking System (ATS)
structured candidate profile data
recruitment historical data
FARM
domain knowledge
experts
analytical script library
feature extraction and machine learning algorithms
Recruitment Support System (RSS)

Figure 6.1: The architecture and data flow in the described *in-house* RSS at tb4hr [52].

because of the above-mentioned reasons. Generally, the *in-house* functionalities assume scoring the candidate-job pairs while the *external* mode delivers the ability to score the candidates only.

As explained in more detail in [52], the *in-house* mode of tb4hr's system is built around the Applicant Tracking System (ATS) called FERMI, which was developed by an external company for the purpose of collecting information about potential job candidates from Internet CV databases and professional networking sites visited by the sourcers. Its main function is to provide the sourcers with convenient means for accessing and sharing candidate profiles. It also enables managing the recruitment process at each of its stages.

We extended FERMI with the three modules which facilitate the analysis of the data collected by the sourcers. The first module – called BIZON – is a Google Chrome plugin registered in Chrome Web Store[2]. It downloads the HTML profiles visited by the sourcers. Such acquired raw data is then persisted in the second module – SILO – which works in the client-server architecture of Google App Engine Standard[3]. SILO also stores meta-data about the visited profiles, as well as job offers which stand as a context for visiting particular candidate profiles by the sourcers. The final data extracted from the original profiles are loaded into the third module – FARM – which is a relational database. It also contains the dictionary data supplied by the HR experts.

The data flow in the tb4hr's system is acyclic and incremental. The relevant webpages that have been visited using a browser with the BIZON plugin installed are downloaded and stored in SILO. Then, the corresponding information is extracted and augmented using scripts and the auxiliary data, prepared either manually or obtained from external sources, e.g., GeoNames[4]. The final data is archived in FARM, where it can be used for further analysis, attribute evaluation, and constructing scoring models. The process is incremental since new web pages are processed independently from the web pages already stored in the system. To achieve higher processing performance, the computations can be also conducted for larger batches of the buffered web pages.

The above-discussed software components make it possible to support the employees of tb4hr *in-house*. The system ingests online materials browsed manually by the sourcers and advises them on particular candidates in the context of particular job offer campaigns. This makes the scoring task more complex than the *external* mode which focuses on assessing the general likeliness of changing employment by particular persons. In the *in-house* mode, the likeliness is just one of the ingredients

[1] `http://tb4hr.com/`

[2] `https://chrome.google.com/webstore`

[3] `https://cloud.google.com/appengine/docs/standard`

[4] GeoNames Gazetteer Data – `https://www.geonames.org` – licensed under CC BY 4.0

in the assessment of whether a given person might be interested in a particular new job opportunity. Such a goal requires modeling more sophisticated data and as already mentioned, operating with the feedback with respect to the pairs – a candidate and a job offer – not just a candidate.

6.2 Attribute Groups

A thorough domain-driven feature engineering process can help us in constructing interpretable scoring models. This makes it important for both, *in-house* and *external* scenarios deployed at tb4hr. As visible in Figure 6.1, after storing the augmented data representations in FARM, one can run additional processes defined in the analytical script library, aiming, e.g., at deriving new characteristics describing the pairs of profiles and job offers. We have defined 772 of such attributes in cooperation with the tb4hr sourcers. They can be categorized into three groups: attributes of candidates, attributes of job offers, and the relations between candidates and offers. This space of attributes goes far beyond the one investigated in [52].

The *candidate* group can be divided into five subgroups representing different aspects of candidate characteristics: *g1: employment history* (e.g., the number of jobs up to now, average time between switching jobs), *g2: skills* (e.g., the number of certifications, keywords from a list of skills), *g3: education* (e.g., academic degree, date of the latest education entry), *g4: place of residence* (e.g., country's GDP, population) and *g5: current status* (e.g., the current employment length). The *candidate* group is suitable also in the *external* mode. Actually, the accuracy parameters that we are reporting at the beginning of Section 6.3, for the task of scoring the likelihood that a person would change his/her job, are obtained using machine learning models based on these attributes. All of them can be calculated for a given profile once it is stored in FARM. Then we can pair them with any offer. After some time the profiles may become outdated but the sourcers can refresh them via BIZON.

The *g6: job offer* group contains characteristics of particular positions. The attributes are derived based only on information that can be revealed to potential candidates in the initial contact email. They need to be computed only once for a given offer. This group includes attributes such as the company name, position type, recruitment country, and city. We also store a short textual description of the offer, which is utilized as outlined in Section 6.3.

The *g7: person-offer relation* group contains attributes that express how a given offer fits to a given candidate. The items in g7 include an indicator of whether a candidate speaks the required language, a geographical distance between a candidate's city and the city from the offer, a Jaccard similarity value between a short description of the candidate and a textual description of the offer, and many others. Since information covered by those characteristics is relative to particular candidates and job offers, it needs to be computed for every single candidate-offer pair. From a technical standpoint, the derivation of such attributes is possible due to the augmentations of the data in FARM. The attributes in g7 (e.g., similarity between a job offer description and candidate's skill set) are particularly aimed to build models easy to interpret by SMEs.

Figure 6.2 depicts a sample decision tree based on the above attributes. Its analysis can reveal some interesting rules, e.g., "*EU-based developers, who receive a remote work offer, and who follow many companies in social media, are more likely to be interested in the offer*". This tree was induced using the standard scikit-learn [97] over the data set which is significantly bigger than in our previous studies. Besides the data acquired in Q4[5] of 2017 and analyzed in [52], we included also the profiles uploaded to BIZON in Q1 of 2018. In total, we collected the data on 20,000 profiles corresponding to 202 tb4hr's recruitment campaigns. (Our previous study was based on only 2,288 cases.) Each pair in the data (i.e., each profile and the corresponding job offer) was labeled by a tag expressing whether a candidate was interested in moving to a new job.

The distribution of classes in the considered data set is highly imbalanced, with only about 5% of positive cases. This reflects a typical success rate of the sourcers, whereby only very few out of contacted candidates are interested in the given offer. Accordingly, the above rule characterizing the

[5] Q1-Q4 denote four quarters of a year.

EU-based developers triggers an over six times higher chance than the average, although we should keep revisiting such rules and recalculating the underlying models because of, e.g., the COVID-19-related reasons discussed in Section 1.3. Moreover, the decision trees such as the one in Figure 6.2, should not be confused with the recently discussed popular trees which approximate complex models such as, XGBoost [111]. Although such approximations may be useful for SMEs to understand the dynamics of the models' performance, they still require an insightful method for ranking attributes on the basis of their importance.

To ensure transparency in our research, and with the pemission obtained from tb4hr, we are working toward making the considered new data set publicly available at *KnowledgePit.ai*[6] which is an online platform for the data science competitions [57]. Until achieving this objective, we have established a temporary link[7] through which the data set can currently be accessed. This data set can be useful to reproduce and extend the results presented in Sections 6.4 and 6.5. For the purpose of assuring the privacy of candidates' profiles, we could not disclose the attribute names. Precisely, we use the real attribute names in the dissertation, but we tokenized them in the disclosed data set. However, we indicated the respective membership of attributes from g1-g7 in the column names.

[6] `https://knowledgepit.ai/`

[7] `https://tinyurl.com/tbdata`

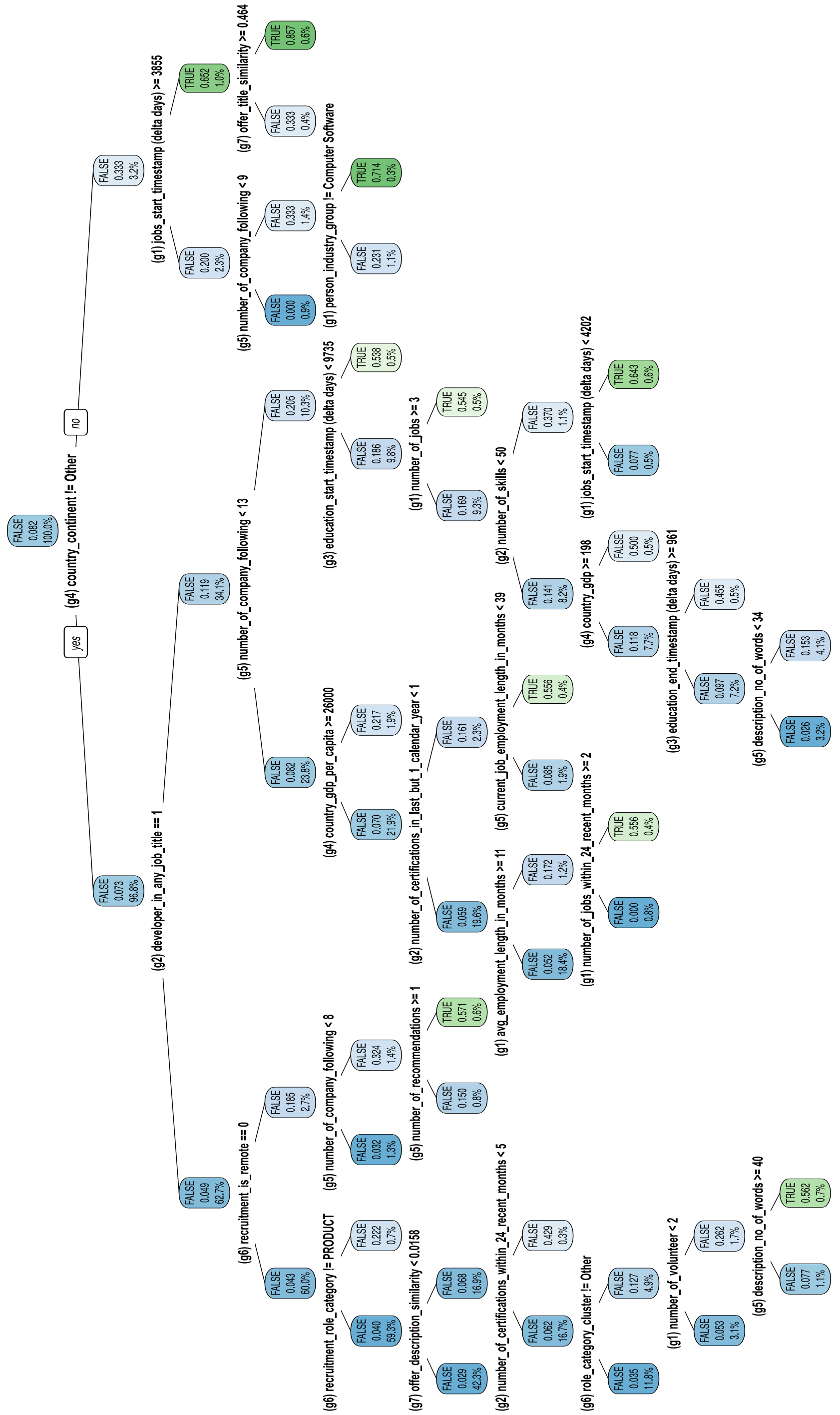

FALSE 0.082 100.0%
(g4) country_continent != Other
yes
no
FALSE 0.073 96.8%
(g2) developer_in_any_job_title == 1
FALSE 0.333 3.2%
(g1) jobs_start_timestamp (delta days) >= 3855
FALSE 0.049 62.7%
(g6) recruitment_is_remote == 0
FALSE 0.119 34.1%
(g5) number_of_company_following < 13
FALSE 0.200 2.3%
(g5) number_of_company_following < 9
TRUE 0.652 1.0%
FALSE 0.043 60.0%
(g6) recruitment_role_category != PRODUCT
FALSE 0.185 2.7%
(g5) number_of_company_following < 8
FALSE 0.082 23.8%
(g4) country_gdp_per_capita >= 26000
FALSE 0.205 10.3%
(g3) education_start_timestamp (delta days) < 9735
FALSE 0.000 0.9%
FALSE 0.333 1.4%
(g1) person_industry_group != Computer Software
FALSE 0.333 0.4%
TRUE 0.857 0.6%
(g7) offer_title_similarity >= 0.464
FALSE 0.040 59.3%
(g7) offer_description_similarity < 0.0158
FALSE 0.222 0.7%
FALSE 0.002 1.3%
FALSE 0.324 1.4%
(g5) number_of_recommendations >= 1
FALSE 0.070 21.9%
(g2) number_of_certifications_in_last_but_1_calendar_year < 1
FALSE 0.217 1.9%
FALSE 0.186 9.8%
(g1) number_of_jobs >= 3
TRUE 0.538 0.5%
FALSE 0.231 1.1%
TRUE 0.714 0.3%
FALSE 0.029 42.3%
FALSE 0.068 16.9%
(g2) number_of_certifications_within_24_recent_months < 5
FALSE 0.150 0.8%
TRUE 0.571 0.6%
FALSE 0.059 19.6%
(g1) avg_employment_length_in_months >= 11
FALSE 0.161 2.3%
(g5) current_job_employment_length_in_months < 39
FALSE 0.169 9.3%
(g2) number_of_skills < 50
TRUE 0.545 0.5%
FALSE 0.062 16.7%
(g6) role_category_cluster != Other
FALSE 0.429 0.3%
FALSE 0.052 18.4%
(g1) number_of_jobs_within_24_recent_months >= 2
FALSE 0.172 1.2%
FALSE 0.085 1.9%
TRUE 0.556 0.4%
FALSE 0.141 8.2%
(g4) country_gdp >= 198
FALSE 0.370 1.1%
(g1) jobs_start_timestamp (delta days) < 4202
FALSE 0.035 11.8%
FALSE 0.127 4.9%
(g1) number_of_volunteer < 2
FALSE 0.000 0.8%
TRUE 0.556 0.4%
FALSE 0.118 7.7%
(g3) education_end_timestamp (delta days) >= 961
FALSE 0.500 0.5%
FALSE 0.077 0.5%
TRUE 0.643 0.6%
FALSE 0.053 3.1%
FALSE 0.262 1.7%
(g5) description_no_of_words >= 40
FALSE 0.097 7.2%
(g5) description_no_of_words < 34
FALSE 0.455 0.5%
FALSE 0.077 1.1%
TRUE 0.562 0.7%
FALSE 0.026 3.2%
FALSE 0.153 4.1%

Figure 6.2: An exemplary decision tree induced from the data set gathered by tb4hr in Q4 of 2017 and Q1 of 2018. Features occurring in the tree belong to different groups. The first three tree layers are defined using: the time which has past since the last job's start (g1: the employment history feature group); an indicator whether the word 'developer' occurred in the candidate's title(s) (g2: skills); the continent (g4: place of residence); the amount of company profiles which are currently followed by a candidate in social media (g5: current status); an indicator whether a job is remote (g6: job offer); and a degree of similarity between job offer description and candidate's title(s) (g7: person-offer relation).

6.3 In-House Process

As already discussed, with respect to the complexity of data sets and the mechanisms that we needed to develop to acquire those data sets, the *external* mode of the tb4hr's system is simpler than the *in-house* one. In the *external* mode, it was relatively easy for tb4hr to train models scoring candidate profiles with respect to the likelihood that a given person is willing to change employment in the next 18 months. The accuracy was tested on the set of 4,300,000 external profiles. The AUC value of the best model was equal to 0.65, with F1-score 0.63, precision 0.57, and recall 0.71, whereby the ground truth (the fact of changing a job) was derived from the profile histories. The task of learning the analogous models for the *in-house* mode is harder. The process of collecting the data does not concern the profiles generally available – it concerns the profiles considered in the context of specific job campaigns conducted by tb4hr. Moreover, the ground truth is more tricky to build a model for – it is not a general fact of changing a job but the fact of being interested in a particular offer. In Section 6.4, we report our results obtained for this scenario.

When a new job offer campaign is started, it is natural for the HR experts to refer to the most similar previous campaigns and design a new recruitment strategy based on the experience related to them. In our design, we follow the same approach in order to meet the SMEs' intuitions. From the perspective of the scoring models, this idea could be expressed by the following sentence: "*over similar cases, the model has learned that if ... then ...*". The *if-then* part symbolizes that the considered machine learning methods should generate models providing the sourcers with some logical, interpretable descriptions over interpretable attributes. We should also consider how to understand *similar cases*, i.e., similar job offer campaigns.

The similarity is an important aspect of recommender systems. Therein, due to the high-dimensional nature of typical preference data, the most common choices to model similarity are Jaccard and cosine measures [11]. One can also attempt to construct a custom similarity measure using expert knowledge or employ methods capable of learning the similarity from the data [72]. Our solution needs to evaluate the similarity between job offers and score the candidates with that respect. The offers are described by short texts and a few simple attributes such as the city and country of the employer. One way to represent this type of data is to combine the available attributes and a bag-of-words representation of the text. Another way – the one we follow – is to compute the embeddings of the texts and the available attribute values into a relatively low-dimensional vector space. It can be done using, e.g., the neural probabilistic language models [5] or the techniques such as word2vec and its extensions [68].

In [52], we relied on the standard word2vec, i.e., we utilized the standard skip-gram method with the Noise-Contrastive Estimation loss [68] to train the similarity model on descriptions of historical job offers. The texts were divided into terms and common stop-words were removed. Different embedding sizes between 10 to 100 were explored. As the number of distinct terms was relatively low (after filtering out the most common phrases, less than 1,000 items were left in the dictionary), the final size was set to 50. The embeddings of job offers were created from the embeddings of individual terms by averaging vectors corresponding to keywords occurring in the descriptions.

In order to make sure that the obtained embeddings can indeed lead toward a reasonable process of choosing the most similar job offer campaigns, we needed to investigate to what extent our embeddings of offers can reflect their similarity perceived by the HR experts. This is indeed important from the perspectives of both, making our overall approach work with high accuracy and avoiding the impression that it is counter-intuitive for SMEs. Therefore, we asked the HR experts to manually divide job offers into categories. They assigned each offer to one of 10 of such categories (e.g., *BACKEND*, *BIG_DATA*). Then, we performed the *leave-one-out* classification using the 1-NN algorithm in the embedding space to predict the category of a given offer based on the category of the closest neighbor. The average accuracy of predictions is 77%, which seems to be a reasonably good result considering the relatively high number of categories. Additionally, we visualized the embeddings in a two-dimensional space using t-SNE [45]. Figure 6.3 shows that our simple embeddings can provide SMEs with an intuitive tool for identifying relevant job offer campaigns.

Besides the main data comprising the rows corresponding to the job-candidate pairs, we disclose

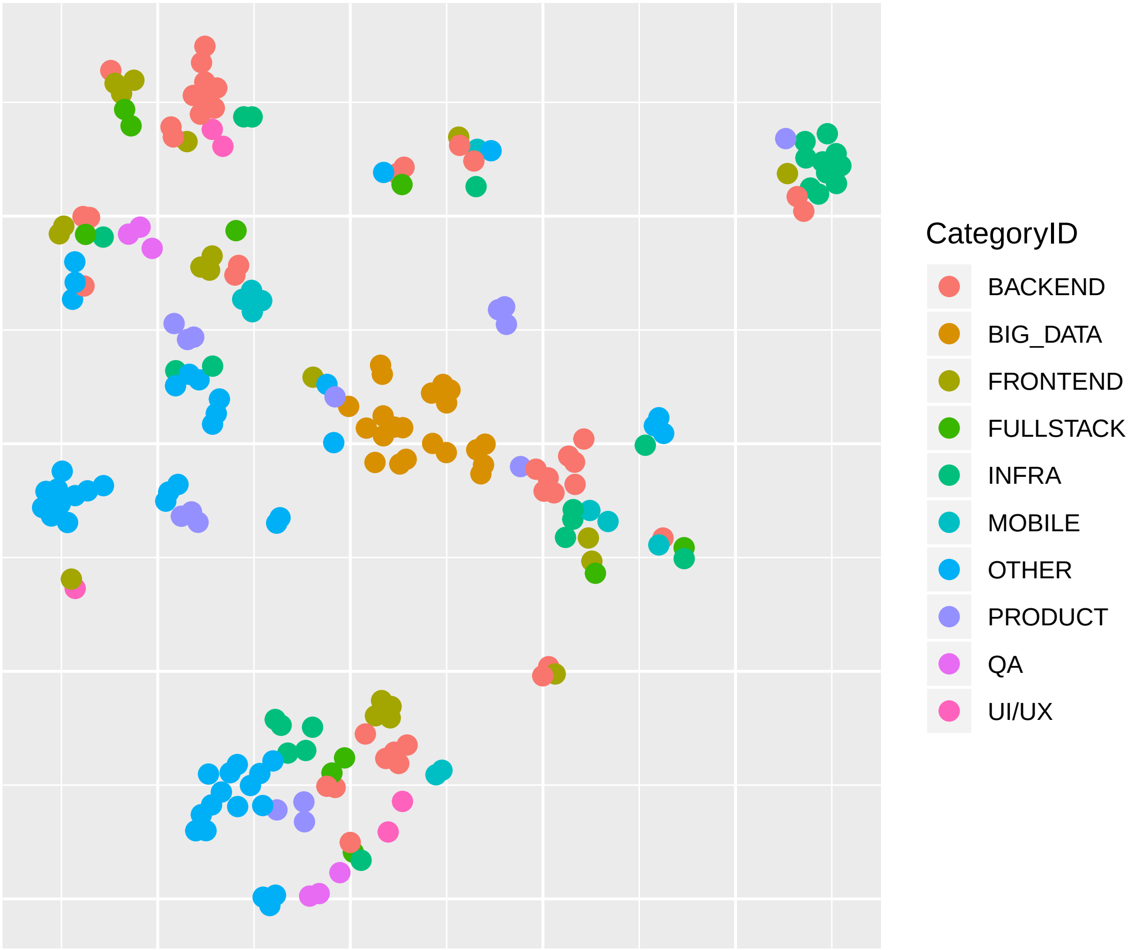


Figure 6.3: Visualization of job offers from our data using the t-SNE technique. The colors on the plot correspond to different categories of the offered positions, which were manually assigned by experts. The list of job offers significantly extends the one considered in [52].

also the embeddings of job offers (see Section 6.2). For privacy reasons, we cannot provide the original job descriptions. Still, their embeddings are useful to reproduce the outcomes of our research.

6.4 Experiments with Scoring Models

The *in-house* mode was deployed at tb4hr in Q4 of 2017. Following the outcomes of [52], we based our solution initially on XGBoost [19]. However, the feedback from tb4hr made us conclude that the analysts and recruitment process designers are interested not only in the efficiency of the scoring models but also in identifying the most important attributes that can be crucial to optimize the real-world interaction with job candidates. We decided to extend our research and focus on the comparison of XGBoost and decision bireducts and we show that although the former method can guarantee slightly better accuracy of the scoring model, the latter method is superior when it comes to producing the trustworthy attribute importance rankings.

The above-discussed models are going to be used for scoring. Precisely, we will operate with two types of evaluation measures for two types of decision problems. The first type refers to the measures such as precision, recall, and F1-score, which are relevant to the binary classification problem. The second type refers to the AUC (the area under the ROC curve) measure that examines the correctness of sorting job candidates by means of probabilities that they are likely to accept a given job offer. Many modern machine learning methods – like XGBoost – serve also as probabilistic classifiers. Therefore, by analogy to XGBoost, we should discuss how to equip the ensembles of decision bireducts with probability-based scorings.

As discussed in Chapter 4, each decision bireduct $(\mathcal{X}, B)$ induces a collection of rules associated with the combinations of values on attributes in B and pointing at specific decision classes. Such a rule-based classifier can perform prediction on any previously unseen object by either returning a single decision triggered by a matching rule or answering with the "I-do-not-know" response in case none of the rules match the object. For the ensembles of decision bireducts, we implemented a simple aggregation scheme with two variants. For each object whose prediction is calculated, the actual votes (without considering the "I-do-not-knows") of bireducts in the ensemble are counted and normalized. In the main variant each vote is equally important, while in the alternative approach, the votes are weighted by cardinalities of $\mathcal{X}$ for particular bireducts $(\mathcal{X}, B)$. Such aggregation can be applied to deliver the probability-based scores or it can serve as the means for voting in the standard case of decision making.

The first step in the construction of our scoring models is the identification of historical recruitment campaigns that are most similar to the offer at hand. We do it using the techniques described in Section 6.3. In the second step, we collect all candidate records processed in those campaigns and we fit models to recruitment results achieved in those campaigns.

In our experimental assessment of such a process, we divided the data set into chunks associated with different offers. For each chunk, we constructed three prediction models using all the remaining chunks as the training data and we tested them on the selected chunk. We call this method *leave-one-offer-out* keeping analogy to the *leave-one-group-out* approach described in Section 2.3. Let us note that if we would have used *leave-one-out* or, e.g., 10-fold cross-validation, we would not be able to reflect a typical deployment scenario of the discussed system and would be likely to produce over-optimistic estimations. Actually, AUC obtained for 10-fold cross-validation was nearly 20% higher than for our *leave-one-offer-out*.

The three compared models are: 1) decision trees built with the aforementioned scikit-learn's decision tree classifier [97], 2) boosting models constructed using the XGBoost library [19], and 3) ensembles of decision bireducts [119]. As for decision trees, they are included in our study to set up some baseline expectations. It is also worth noting that in the case of our *leave-one-offer-out* experiment, they may look different from the one visible in Figure 6.2, whereby the whole data set was used for training. Decision trees were constructed with the minimum Gini impurity decrease parameter in the set $\{\langle\!\langle 0.0\rangle\!\rangle$[8]$, 0.001, 0.01, 0.1\}$ and the maximum depth parameter in the set $\{2, 3, 4, \langle\!\langle 5\rangle\!\rangle\}$. Gini impurity was used also as the cut evaluation criterion. The number of boosting iterations in XGBoost and the size of decision bireduct ensembles were both set to 1,000. Moreover, the hyperparameters for XGBoost were obtained using a grid search over binary classification of decision trees with the maximum depths taken from the set $\{2, \langle\!\langle 3\rangle\!\rangle, 4, 5, 10\}$ and the learning rates taken from $\{\langle\!\langle 0.001\rangle\!\rangle, 0.01, 0.1\}$. On the other hand, each of decision bireducts was constructed using DAAR-based algorithm (see Algorithm 5 in Section 5.3) with the number of candidate attributes fixed to 100, the number of probes and the allowed randomness (both used in the stopping criterion [56]) set to 100 and 0.05 (respectively), and the maximum number of attributes either equal to 3 or to $\langle\!\langle$unlimited$\rangle\!\rangle$. Furthermore, in order to perform experiments with ensembles of decision bireducts, we employed an unsupervised quantile-based approach to discretize every numeric attribute into 3 intervals.

Table 6.1 shows the results obtained by the above mentioned test methods. Due to the imbalanced distribution of labels in the data, we evaluated the results using four different quality measures, namely precision, recall, F1-score and AUC. In order to check the impact of the retrieval of the relevant training data, we repeated this experiment with enabled filtering based on the similarity of the offers. At each step of the *leave-one-offer-out* evaluation, the training data set was filtered before the construction of a scoring model. Only the history of investigated candidate profiles corresponding to K most similar offers to the tested one was used. The experiment was repeated for values of K between 10 and 202 (the case when no filtering is done) with a step of 10. Figure 6.4 illustrates the results obtained by the tested algorithms for different values of K.

Table 6.1: Comparison of the scoring models using *leave-one-offer-out*. The values in the column *experts* were computed based on the actual responses to emails sent by the sourcers. (We assume that recall of the experts is 1.0.) The columns *top 40* reflect the models constructed on the training data filtered with respect to the similarity of offers (computed in the offer embedding space). The columns *same* indicate results when the training data was limited to recruitments with the same CategoryID (cf. Figure 6.3) of job offers as the tested one.

dec. tree	all	same	top 40
Precision	0.131	0.124	0.141
Recall	0.551	0.352	0.362
F1-score	0.211	0.183	0.203
AUC	0.659	0.558	0.630

bireducts	all	same	top 40
Precision	0.144	0.128	0.147
Recall	0.468	0.445	0.419
F1-score	0.220	0.199	0.217
AUC	0.671	0.630	0.668

XGBoost	all	same	top 40
Precision	0.162	0.138	0.154
Recall	0.391	0.341	0.445
F1-score	0.229	0.197	0.229
AUC	0.679	0.622	0.682

experts	
Precision	0.069
Recall	1.000
F1-score	0.131
AUC	–

The results show that the best performance was achieved by XGBoost, with respect to both F1-score and AUC. Filtering of the training data slightly increased the results achieved by XGBoost but at the same time, the performance of bireducts slightly decreased. A significant change was visible only for the simple decision tree, for which the model trained on the full data had AUC higher by 0.029 than the one that used only the 40 most similar campaigns. The highest result of XGBoost was obtained for $K = 40$, whereas the best bireduct ensemble was trained for $K = 190$. In that case, we achieved for bireducts $AUC = 0.677$, which is higher than 0.668 in Table 6.1.

For the sake of comparison, we also evaluated the algorithms in the scenario where the training data is limited to recruitment processes that correspond to the same offer category as the tested one. Then, the results were considerably lower than in the case when the filtering of the data was

[8] The optimal parameters found during grid search are marked using $\langle\!\langle . \rangle\!\rangle$.

conducted using the embeddings of textual descriptions of job offers. Nevertheless, in this specific scenario bireducts turn out to provide more accurate scoring models than XGBoost. This is a useful hint for those potential deployments of our approach, wherein the ability to rely on the historical job offer campaigns would be limited. This might be also the case if tb4hr undertook a new campaign which would be totally incomparable with the previous job offers.

For the purpose of providing an additional baseline to the above experimental results, we investigated also the success rate of human sourcers. If we assume that they sent the offer to all potentially relevant candidates, then their recall would equal 1.0. (Actually, it does not need to be true in practice.) We can estimate the precision of the sourcers by taking the percentage of positive examples in our data set and thus, we can compute their F1-score. The result of the sourcers is shown in the bottom-right fragment of Table 6.1.

The fact that the F1-score achieved by XGBoost is nearly 75% greater than that of humans clearly shows the benefits which our system can bring to tb4hr. Interestingly, a simple decision tree model – such as the one discussed in Section 6.2 and illustrated in Figure 6.2 – was able to achieve only a few percent worse score than the boosting and bireduct ensemble approaches.

We also analyzed our scoring models against a subset of the data, corresponding only to the most typical clients. Since tb4hr specializes in recruiting staff to technological startups, we measured the performance only on their most representative recruitments. On this subset ($\approx 80\%$ of the data), our best model achieved a slightly greater AUC than on the whole data (≈ 0.688).

In summary, when comparing to [52], the best models yielded better prediction results in terms of both, AUC and F1-score. As for AUC, our best model improved by over 4%. It demonstrates how much the extra data collected by the sourcers after the initial deployment of the data acquisition system has impact on the performance of a model. However, we need to remember that besides utilizing more training data, the methodology reported herein is based on a bigger set of attributes designed together with the HR experts (772 compared to 374 in [52]).

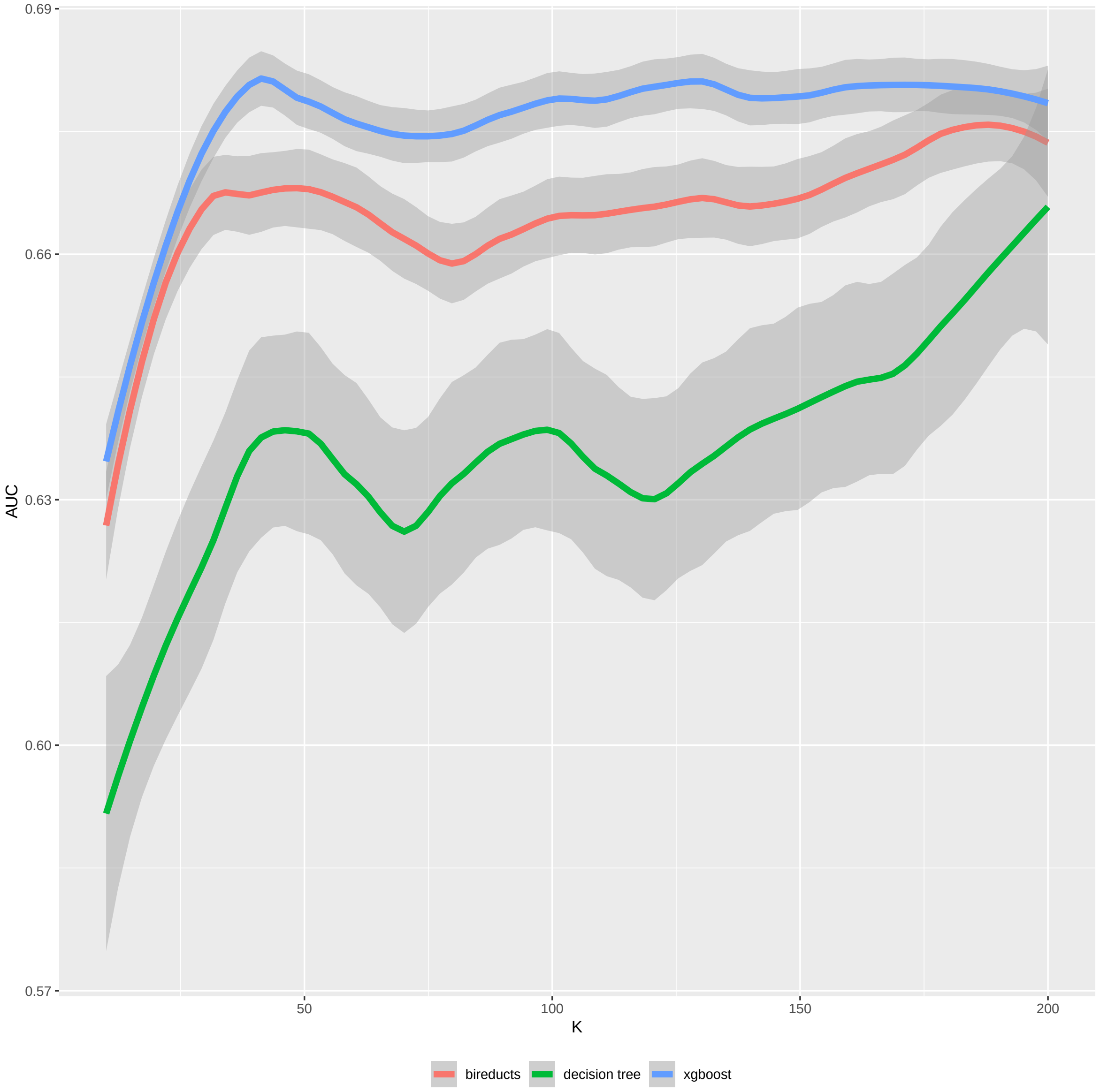


Figure 6.4: Results of the considered scoring algorithms evaluated using the *leave-one-offer-out* approach, when the model training is performed on the data corresponding to K most similar recruitment processes (K between 10 and 200).

6.5 Attribute Groups Contributions

To make our approach applicable in a range of different practical scenarios, we need to validate the impact of particular groups of attributes outlined in Section 6.2. In practice, not all of those groups could be available at the same time, or they may come with different costs of acquisition. Our trust in the correctness of attribute values in different groups could also vary. As a corner case, some of the attribute groups could be fully redundant (and therefore unnecessary to generate and maintain) because the remaining ones still deliver the scoring models of sufficient quality. The analysis of the impact of attributes from each group can be also important in order to provide HR experts with insights into factors influencing the chances of finding good candidates for a job. Therefore, it is useful for verifying the fairness of our models and detecting any undesired (e.g., social) biases [83].

Table 6.2: AUC computed for $K = 40$, with excluded attributes from the individual groups described in Section 6.2. The column N indicates the number of attributes in the corresponding group. Such a large number of attributes in the group ‘skills’ is due to the fact that this group includes one-hot encoded skill names listed in candidate’s CV.

attribute group id / name	N	dec. tree	bireducts	XGBoost
g1: employment history	36	0.641	0.667	0.684
g2: skills	570	0.615	0.657	0.674
g3: education	17	0.630	0.662	0.682
g4: place of residence	48	0.618	0.636	0.658
g5: current status	29	0.636	0.651	0.664
g6: job offer	50	0.614	0.654	0.666
g7: person-offer relation	22	0.630	0.675	0.680

Accordingly, we repeated the evaluation of the previously tested models, as described in Section 6.4, but this time with excluded attributes belonging to each indicated group. Table 6.2 shows that for nearly all groups, the exclusion of respective attributes does not have a significant impact on the AUC value. For XGBoost and bireducts, the most severe degradation in AUC is visible when attributes related to the candidate’s place of residence (g4) are excluded (XGBoost: $0.682 \longrightarrow 0.658$; bireducts: $0.668 \longrightarrow 0.636$). For bireducts, this means nearly 5% reduction compared to the AUC of models trained using all attributes. As for the decision trees, the most severe effect is observed in case of the exclusion of attributes describing the job offer ($0.630 \longrightarrow 0.614$), which may in particular mean that this is not a perfect choice for the *external* mode of tb4hr’s operations. Similar tendencies could be observed for other settings, in particular for $K \neq 40$.

Interestingly, the exclusion of some groups had no effect or even slightly improved the performance of the investigated models. Those results suggest a considerable redundancy in the information carried on by the engineered attribute set. This might be actually an advantage in the case of potential incompleteness of the data about job offers and candidates. Still, in Figure 6.2 one can see that the individual elements of all attribute groups are used in a decision tree constructed for all available data. This might suggest that attributes from all groups can work well together. In particular, all attribute groups have at least one representative among the attributes used in the top four layers of the induced decision tree, although one can also notice a kind of correspondence between the level of occurrence of those elements in the tree structure and the importance of the corresponding groups, which can be deduced from Table 6.2.

If all attributes are available, it might be reasonable to drop those belonging to *g1: employment history*, *g3: education*, and *g7: person-offer relation* before the model construction as their removal does not harm the performance of any of the investigated models. Figure 6.5 seems to provide an argument supporting this hypothesis. The accumulated Shapley values [74] in each of those three groups are nearly three times lower than the corresponding values for other attribute groups. Nevertheless, to confirm this observation, a more detailed investigation of the importance of individual attributes is needed. More research is also needed to understand redundancies and interactions between the considered groups using various measures of attribute importance and interchangeability.

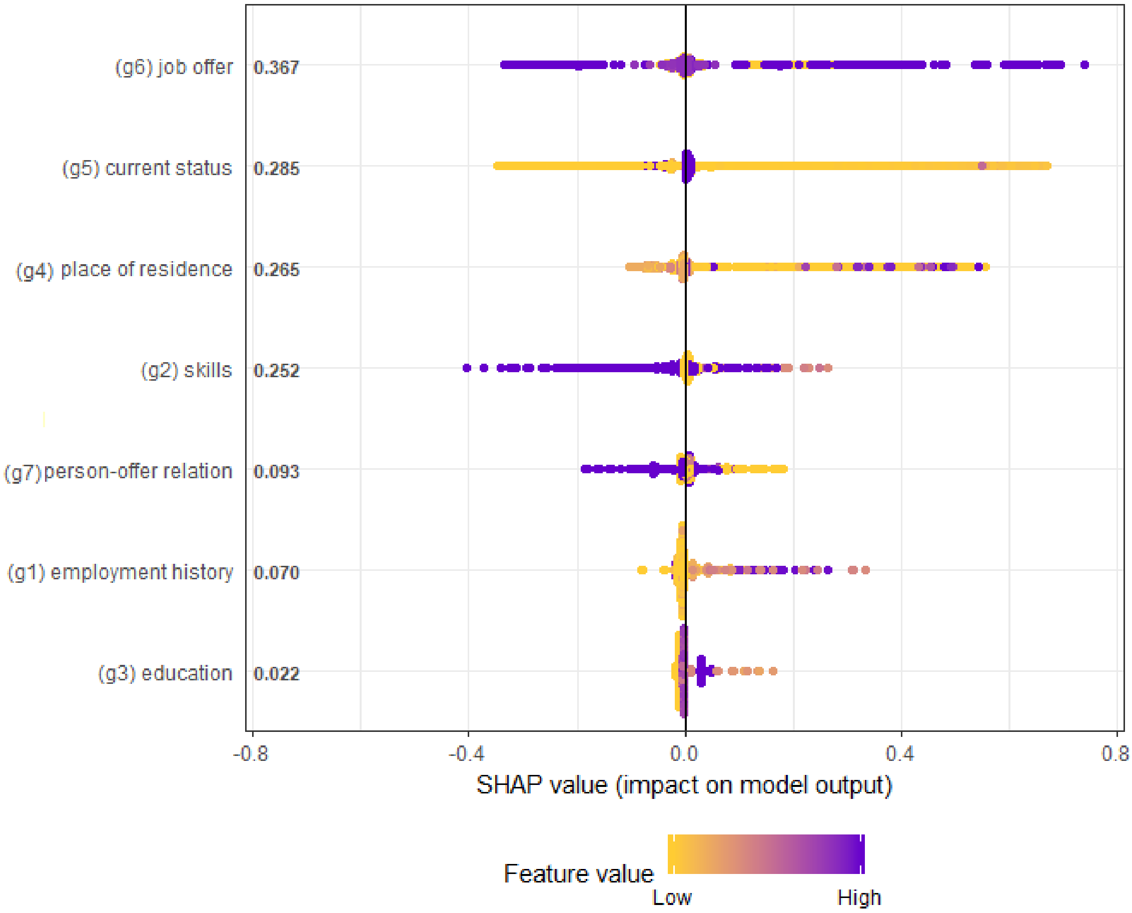


Figure 6.5: The aggregated SHAP values of XGBoost for the elements of each feature group.

Chapter 7

Feature Importance and Rankings

This chapter deals with different techniques of evaluating importance of attributes. Machine learning applications often require constructing interpretable decision models [32]. Interpretability means using attributes that are understandable by the subject matter experts (SMEs) and connecting those attributes in an understandable way within a model. One of the often considered aspects of interpretability is the ability to report the importance of particular attributes in a model to SMEs. Such information may help in better understanding what aspects of the data are crucial for the accuracy of a model. SMEs can, e.g., express their concerns if they see some attributes being too important or not sufficiently important from the perspective of their domain knowledge. The attribute importance analysis is also useful for acquiring new knowledge about complex data dependencies and can provide useful insights about the modeled problem. Such insights are beneficial for the experts who can analyze the model and verify whether the chosen attributes do not contradict basic intuitions [112]. In this context – since there are many methods of computing the importance of attributes – it is hard to decide which one is more helpful, i.e., provides an importance ranking for attributes that enables better identification of relevant attributes. In this regard we design attribute importance methods in the context of approximate decision reducts, decision bireducts and present some experimental results comparing usefulness of such methods.

7.1 Feature Selection Based on Approximate Decision Reducts

This section presents methods of attribute ranking that utilize the approximate decision reducts, to measure relevance of individual attributes [51]. All of these methods are multivariate as they consider attributes in the context of others and are able to detect dependencies between them. In our research, we needed to adapt existing methods to better suit the task of selecting relevant attributes from high-dimensional decision systems (≥ 1000). To generate approximate decision reducts we use an algorithm described in [49] with a modified stopping criteria. This method makes use of random sampling of the attribute set in order to discover reducts that capture diverse characteristics of data and reduce the computation cost. It constructs reducts from numeric data using the maximum-discernibility discretization heuristic [84].

Commonly used rough set feature ranking methods exploit the fact that the informative attributes are usually able to discern more objects from different decision classes, and thus are more likely to be present in an approximate decision reduct. We examine three modifications of frequency-based approaches to attribute ranking. We test them in combination with approximate decision reducts and compare their performances to our attribute ranking method.

Let us denote a finite set of approximate decision reducts of the decision table $\mathbb{A} = (U, A \cup \{d\})$ by $ARED(\mathbb{A})$. Let $|ARED(\mathbb{A})| = m$ and assume that each $AR_i \in ARED(\mathbb{A})$ was computed using a subset of attributes $B_i \subseteq A$, $i = 1, \ldots, m$. The first and the simplest of the compared methods ranks attributes by counting how many times they appear in a given set of approximate decision reducts. The method assigns the score to each attribute $a \in A$:

$$Score_1(a) = |\{AR \in ARED(\mathbb{A}) : a \in AR\}|. \tag{7.1}$$

The second ranking method takes into account the fact that the reducts can be computed for different subsets of attributes. It scales the attribute counts to express the probability that the attribute appears in a reduct if it is present in the subset used for computation:

$$Score_2(a) = \begin{cases} \frac{|\{AR \in ARED(\mathbb{A}) : a \in AR\}|}{|\{B \in \{B_1, B_2, \ldots, B_m\} : a \in B\}|} & \text{if } |\{B \in \{B_1, B_2, \ldots, B_m\} : a \in B\}| > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{7.2}$$

where $B_i \subseteq A$ is a set of attributes used for computation of AR_i.
The third method considers a predictive potential of the reducts. In this approach each approximate decision reduct AR is assigned with a score $Scr(AR)$ which expresses its quality. This value is treated as a weight during computation:

$$Score_3(a) = \begin{cases} \frac{\sum_{AR \in ARED(\mathbb{A}) : a \in AR} Scr(AR)}{\sum_{AR \in ARED(\mathbb{A})} Scr(AR)} & \text{if } \sum_{AR \in ARED(\mathbb{A})} Scr(AR) > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{7.3}$$

The value of $Scr(AR_i)$ could be computed using one of many heuristics. In the experiments we assess a quality of the approximate decision reduct by direct application of the discernibility measure to objects, which were removed from the training set during the approximate reduct assembling.

We propose also a more complex algorithm called Rough Attribute Ranker (RAR) that is an extension to the methods described above. It uses a discernibility-based scoring function Scr_{AR} to examine how particular attributes from an approximate decision reduct influence the reduct's quality. The impact of a single attribute is estimated as an average difference in the score assigned to the reduct after exchanging this attribute with a random permutation of the attribute's values, thus retaining its distribution. The value of Scr_{AR} can be expressed as follows:

$$ScrGain_{AR}(a) \begin{cases} Scr(AR) - \frac{\sum_{i=1}^{K} Scr(AR'_i)}{K} & \text{if } a \in AR \\ 0 & \text{otherwise,} \end{cases} \tag{7.4}$$

where AR is an approximate decision reduct, K is a number of random probes used for the estimation, and AR'_i, for $i = 1, \ldots, K$ are sets of attributes that were constructed from AR by replacing a with its permutation. The final score given to an attribute a is determined by its average impact on the quality of the reducts:

$$Score_{RAR}(a) = \begin{cases} \frac{\sum_{AR \in ARED(\mathbb{A}) : a \in AR} ScrGain_{AR}(a)}{|\{AR \in ARED(\mathbb{A}) : a \in AR\}|} & \text{if } |\{AR \in ARED(\mathbb{A}) : a \in AR\}| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.5}$$

This method may be seen as an analogy to the Breiman's relevance measure [14, 30] for the random forest, which assesses the importance by examining how randomization of particular attributes influences the error rate of trees.

The attribute ranking methods described above were empirically evaluated and compared to results of several commonly used statistical feature rankers, i.e., a correlation-based ranker, a Wilcoxon test-based ranker, the information gain and the relief algorithm. This comparison has been performed using two different evaluation methods. All the experiments were implemented and executed in R System [106].

The first test was conducted on synthetic data. In this experiment a dataset containing 10000 objects described by 1000 numeric attributes was generated from normal distribution. Three different decision attributes were constructed using the first 20 features so that each of the selected features have the same impact on the decision values. We will refer to the data sets built from a combination of the numeric attributes and the three decisions as *synthetic1*, *synthetic2*, and *synthetic3*, respectively. The decision attributes are defined in the following way:

$$decision_1(u) = 1 \Leftrightarrow \sum_{i=1}^{20} a_i(u) \geq 0, \tag{7.6}$$

$$decision_2(u) = 1 \Leftrightarrow a_1(u)a_{20}(u) + \sum_{i=1}^{19} a_i(u)a_{i+1}(u) \geq 0, \tag{7.7}$$

$$\begin{aligned} decision_3(u) = 1 \Leftrightarrow & \Big(a_1(u) \in [-\delta, \delta] \wedge a_{20}(u) \in (-\infty, -\delta) \cup (\delta, \infty)\Big) \vee \\ & \bigvee_{i=1,\ldots,19} \Big(a_i(u) \in [-\delta, \delta] \wedge a_{i+1}(u) \in (-\infty, -\delta) \cup (\delta, \infty)\Big). \end{aligned} \tag{7.8}$$

The value of the parameter δ was adjusted to roughly equalize the sizes of decision classes. The first decision is dependent linearly on the attribute values and as such might be seen as the easiest one, whereas the respective dependencies of the second and the third decision attributes are not linear. In order to better mimic a real-life situation an additional factor, namely noise was introduced to data. For each of the decision attributes 20% of randomly selected values were rearranged by a random permutation.

The compared ranking algorithms were used to select relevant features for each decision vector. The number of attributes that each of the algorithms should choose was estimated using the random probes test described in [41] and [23]. Quality of the selected sets of features was assessed using classic measures, e.g., precision, recall and F1-score (see Definitions 6, 7 and 8), from the information retrieval domain. Table 7.1 presents results achieved by the particular algorithms.

Table 7.1: Results of the attribute ranking methods on the synthetic data.

	*synthetic*1				*synthetic*2				*synthetic*3			
Ranker:	N	*Prec.*	*Recall*	F_{score}	N	*Prec.*	*Recall*	F_{score}	N	*Prec.*	*Recall*	F_{score}
CorrRank	32	0.63	1.0	0.77	14	0.07	0.05	0.06	20	0.05	0.05	0.05
Wilcoxon	28	0.71	1.0	0.83	11	0.09	0.05	0.06	19	0.05	0.05	0.05
InfoGain	27	0.74	1.0	0.85	8	0.13	0.05	0.07	16	0.56	0.45	0.5
Relief	39	0.51	1.0	0.68	19	0.16	0.15	0.15	22	0.09	0.1	0.09
AR_{Score1}	28	0.71	1.0	0.83	10	0.0	0.0	0.0	12	0.25	0.15	0.19
AR_{Score2}	28	0.71	1.0	0.83	12	0.0	0.0	0.0	10	0.3	0.15	0.2
AR_{Score3}	26	0.77	1.0	0.87	10	0.0	0.0	0.0	12	0.33	0.2	0.25
RAR	29	0.69	1.0	0.82	26	0.62	0.8	0.7	21	0.48	0.5	0.49

The second experiment was conducted on three microarray datasets related to different medical domains. The data was downloaded from a public genetic data repository ArrayExpress[1]. The *acuteLymphoblasticLeukemia* dataset (ArrayExpress experiment accession number E-GEOD-13425, 190 samples, 22277 genes) describes 5 genetic subtypes of acute lymphoblastic leukemia, the *hepatitisC* data (E-GEOD-14323, 124 samples, 22277 genes) regards a role of chronic hepatitis C virus in the pathogenesis of HCV-associated hepatocellular carcinoma and the *skinPsoriatic* dataset (E-GEOD-13355, 180 samples, 54675 genes) contains profiles of genetic changes related to the skin psoriasis.

In this test, the compared methods were used to selected gene sets in a repeated 5-fold cross-validation schema. A quality of each gene set was evaluated based on classification results achieved by two prediction models, namely k-NN and random forest, which are commonly used in the microarray experiments. Due to uneven sizes of the decision classes, the prediction accuracy was measured using

[1] www.ebi.ac.uk/arrayexpress

the balanced accuracy score (see Definition 12). As in the experiment on synthetic data, the number of genes selected in each fold of the cross-validation cycle was determined using the random probes method. Table 7.2 summarizes the mean results achieved by each ranking algorithm after 10 executions of 5-fold cross-validation tests.

Table 7.2: Results of the attribute ranking methods on the microarray data. The mean and standard deviation of the number of selected genes and the balanced accuracy are given.

	acuteLymphLeukemia			hepatitisC			skinPsoriatic		
Ranker:	N	$BAC : kNN$	$BAC : RF$	N	$BAC : kNN$	$BAC : RF$	N	$BAC : kNN$	$BAC : RF$
CorrRank	6880 ± 175	$0.91 \pm .01$	$0.75 \pm .08$	$> 13K \pm 630$	$0.86 \pm .02$	$0.79 \pm .04$	$> 30K \pm 999$	$0.76 \pm .02$	$0.81 \pm .02$
Wilcoxon	3538 ± 147	$0.91 \pm .01$	$0.75 \pm .07$	6348 ± 425	$0.87 \pm .01$	$0.77 \pm .05$	$> 22K \pm 737$	$0.76 \pm .02$	$0.82 \pm .01$
InfoGain	5733 ± 163	$0.91 \pm .01$	$0.75 \pm .08$	$> 12K \pm 644$	$0.86 \pm .02$	$0.78 \pm .04$	$> 24K \pm 798$	$0.76 \pm .02$	$0.82 \pm .02$
Relief	12054 ± 327	$0.92 \pm .01$	$0.82 \pm .07$	2551 ± 999	$0.89 \pm .01$	$0.79 \pm .02$	4904 ± 897	$0.76 \pm .03$	$0.81 \pm .02$
AR_{Score1}	1512 ± 233	$0.91 \pm .01$	$.82 \pm .02$	1471 ± 130	$0.90 \pm .01$	$0.80 \pm .04$	1492 ± 44	$0.78 \pm .03$	$0.83 \pm .02$
AR_{Score2}	1965 ± 165	$0.92 \pm .01$	$0.81 \pm .02$	1699 ± 195	$0.90 \pm .01$	$0.79 \pm .05$	1793 ± 176	$0.78 \pm .02$	$0.83 \pm .02$
AR_{Score3}	2021 ± 152	$0.91 \pm .02$	$0.81 \pm .02$	1696 ± 213	$0.90 \pm .01$	$0.80 \pm .04$	1993 ± 174	$0.77 \pm .02$	$0.84 \pm .02$
RAR	1953 ± 230	$0.92 \pm .01$	$0.81 \pm .02$	2068 ± 274	$0.91 \pm .02$	$0.80 \pm .05$	3186 ± 156	$0.77 \pm .03$	$0.84 \pm .02$

The reduct-based rankers outperformed the classic algorithms in both tests. For the synthetic data only RAR method was able to reasonably select attributes which are relevant for the second decision vector. Interestingly, it seems that the simplest frequency based attribute rankers may yield better results if the relation between relevant features and the decision is not very complex. It is also noticeable, especially for microarray data, that the frequency based approaches tend to select less attributes than the RAR. In terms of classification accuracy, the RAR on average performed slightly better than other algorithms but in cases of other reduct-based rankers the difference was not statistically significant[2].

7.2 Evaluation of Feature Importance Rankings

In Chapter 6 we performed classification experiments using XGBoost and decision-bireduct-based models on the real-world data related to the recruitment field. In this section we investigate the capabilities of the XGBoost-based and decision-bireduct-based models in providing rankings for importance of attributes. The XGBoost library has such rankings already implemented, while the analogous method for decision bireducts needs to be designed. This task is crucial for the goals of this section because XGBoost and decision bireducts will be the main "competitors" when it comes to how insightful their methods for ranking attributes are.

In XGBoost, there are several metrics that determine the importance of attributes involved in the ensemble. For the decision-tree-based weak learners, we outline the available options in Table 7.3. Let us note that there are slight differences between the Python and R XGBoost implementations, either in the set of values being returned or in the nomenclature that is followed. However, in all cases, the *total_gain* parameter is the key point. It reflects heuristically estimated contribution of particular attributes into the model's adjustment to the data. For this purpose the Gini impurity or information entropy measures (computed over the universe of objects, weighted in a boosting-based way) are used.

Factors *cover* and *total_cover* refer to the notion of coverage understood as the sum of weights of objects supporting a given tree node. XGBoost uses quite advanced mathematical techniques to derive those weights in relation to a given loss function and its properties. For the square loss function the coverage of a tree node simply corresponds to the number of objects collected by the node. However, this is not a general rule and for some other loss functions the value may not be even proportional to the node's size. Intuitively, the coverage of a node expresses recognition of a need for further splitting that node during a decision tree construction. A high coverage value means that a potential split on the corresponding subset of objects is more likely to be a valid split, which will generally improve the

[2] The significance was measured using the paired t-test at 0.95 confidence

classifier's quality. A low value indicates a potential risk of overfitting related to adding a split on the given node in the tree.

Table 7.3: Several metrics that determine the attribute importance based on an XGBoost model. A comparison of available options for the XGBoost implementations in Python and R.

metric	Python	R
weight	the number of times a given attribute is used to split the data across all trees in the model	not available
frequency	not available	a relative (fractional) number of times a given attribute is used to split the data across all trees in the model; it sums up to 1 and it equals to the normalized *weight*
gain	the average gain across all splits of a given attribute used in the model; it is equal to the ratio of $\frac{total\_gain}{weight}$	a relative (fractional) attribute's contribution based on the total gain of this attribute's splits; it sums up to 1 and it equals to the normalized *total_gain*
cover	the average coverage across all splits a given attribute is used in the model; it is equal to the ratio of $\frac{total\_cover}{weight}$	a relative (fractional) number of objects related to this attribute; it sums up to 1 and it equals to the normalized *total_cover*
total_gain	the total gain for all decision tree splits for which a given attribute is used in the model	not available
total_cover	the total coverage across all splits a given attribute is used in the model	not available

Table 7.4 shows the analogous metrics that we implemented for deriving the importance of attributes from the ensembles of decision bireducts. For a single decision bireduct $(\mathcal{X}, B)$, the *gain* for $a \in B$ equals the difference between the applied heuristic measure (Gini impurity, entropy, etc.) computed for $B \setminus \{a\}$ and B. We implemented two approaches – called "objects scope" and "cover weighting" – that can be used in various combinations to form different strategies for the computation of attribute's importance. For the "objects scope", we have two options: either we compute the *gain* value for the whole training data set (*global* variant) or we can take into account only the objects in $\mathcal{X}$ (*local* variant). For the "cover weighting", we have two options as well: either the computed *gain* is taken as it is or it is weighted by the cardinality of $\mathcal{X}$. That last option is referred to as *cover_weighted*. We can clearly see a connection between some of the metrics defined for decision bireduct ensembles and those defined for the approximate decision reduct ensembles in Section 7.1.

Having the attribute importance ranking methods available for the models under consideration, let us now discuss a model-agnostic procedure to compare them. For a given $\mathbb{A} = (U, A \cup \{d\})$, let us define its variant $\mathbb{A}^{\circlearrowright} = (U, A \cup A^{\circlearrowright} \cup \{d\})$, where $A^{\circlearrowright}$ is a set of shuffled attributes defined as follows:

$$a_i^{\circlearrowright} : U \to V_{a_i}, \quad a_i^{\circlearrowright}(u_j) = a_i(u_{\delta_i(j)}) \tag{7.9}$$

where δ_i corresponds to a random permutation of U. The shuffled attributes preserve the original distributions and they are created as random permutations of the original attributes' values. This approach to randomization subject to distribution preservation is quite popular in machine learning [14,

Table 7.4: Metrics determining attribute importance based on an ensemble of decision bireducts.

metric	description
count	the number of times a given attribute is used in the ensemble of decision bireducts
global_gain	the total gain induced by a given attribute across all decision bireducts in the ensemble when computing the *gain* value in context of *global* variant of "objects scope"
global_gain_cover_weighted	the total gain induced by a given attribute across all decision bireducts in the ensemble when computing the *gain* value in context of *global* variant "objects scope" and weighting the computed value for each decision bireduct by the ratio of the objects it covers
avg_global_gain	the ratio of *global_gain* to the number of occurrences of a given attribute in the ensemble, i.e., the value equals to the ratio of $\frac{global\_gain}{count}$
avg_global_gain_cover_weighted	$\frac{global\_gain\_cover\_weighted}{count}$
local_gain	the total gain induced by a given attribute across all decision bireducts in the ensemble when computing the *gain* value in the context of *local* variant of "objects scope"
local_gain_cover_weighted	the total gain induced by a given attribute across all decision bireducts in the ensemble when computing the *gain* value in the context of *local* variant of "objects scope" and weighting the computed value for each decision bireduct by the ratio of the objects it covers
avg_local_gain	$\frac{local\_gain}{count}$
avg_local_gain_cover_weighted	$\frac{local\_gain\_cover\_weighted}{count}$

67, 66]. Moreover, we use such an approach also in the internals of the aforementioned DAAR method that was also adopted for the decision bireduct derivation procedure, cf. Section 5.3.

Our initial idea on how to design a procedure to compare methods of ranking was to use a Wilcoxon rank-sum test. Let us assume that the algorithms for ranking importance of attributes score the attributes from $\mathbb{A}^{\circlearrowleft}$. We can use the statistical test to analyze the characteristics of the scores for the original and the shuffled attributes with the expectation that those characteristics will significantly differ. However, such a direct approach has a potential drawback – the comparison result is strongly biased and unreliable if only a small fraction of attributes is distinguished by a ranking algorithm and many attributes share the same rank.

In our experiments, we consider the two already-investigated methods, namely decision bireduct ensembles and XGBoost, used in Section 6.4. As a reference, we check also a simple feature scoring method based on the computation of absolute values of the Spearman correlation coefficients between the elements of A (our 772 features) and the target attribute d. For XGBoost and the ensembles of decision bireducts, we use the settings obtained by the grid search and described in detail in Section 6.4. In particular, we keep the balance between 1,000 decision trees and 1,000 decision bireducts. However, in order to make both methodologies even more comparable from the viewpoint of measuring the attribute importance, we fix the maximum tree depth (XGBoost) as equal to 2 and the maximum attribute number (decision bireducts) as equal to 3. This is because binary decision trees with the depth 2 usually use three attributes internally. Moreover, to compare the rankings based on the scores that can be interpreted similarly, we choose *total_gain* (Table 7.3) and *global_gain* (Table 7.4), as they both represent the closest related concepts for XGBoost and decision bireducts.

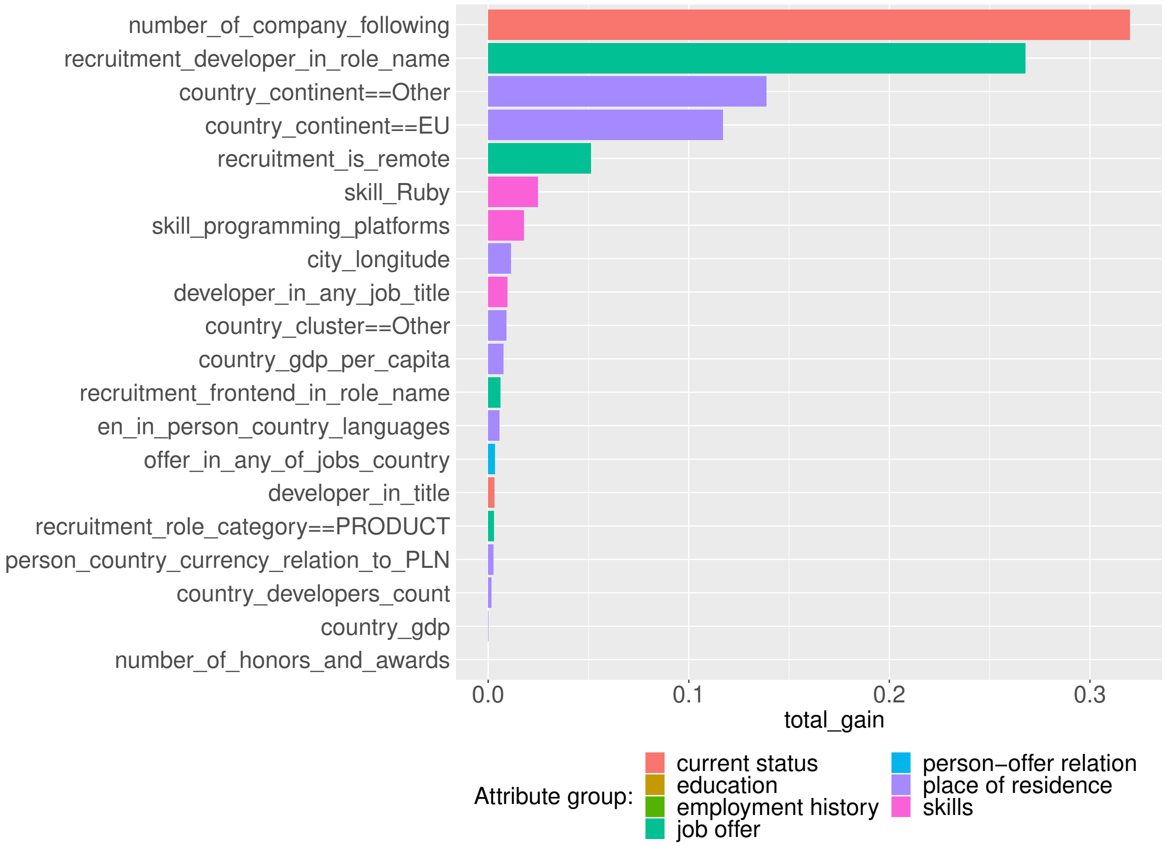


Figure 7.1: Visualization of the XGBoost-specific importance of individual attributes (top 20) obtained using the normalized *total_gain* score values.

Before we present the actual comparison of attribute importance ranking algorithms using the described data table $\mathbb{A}^{\circlearrowright}$; let us first show the context of how these algorithms assess the attributes from the original data table described in Section 6.2 (before adding the shuffled attributes).

The rankings of the top 20 attributes obtained for bireducts, XGBoost, and Spearman correlation on the base data table are given in Figures 7.1, 7.2 and 7.3 (using normalized score values *total_gain*, *global_gain*, *correlation* [actually, the absolute values]), respectively. Notably, all three rankings are quite similar. In particular, among the top five attributes in the ranking obtained for XGBoost, there are four that are also in the top five for bireduct ensemble and correlation.

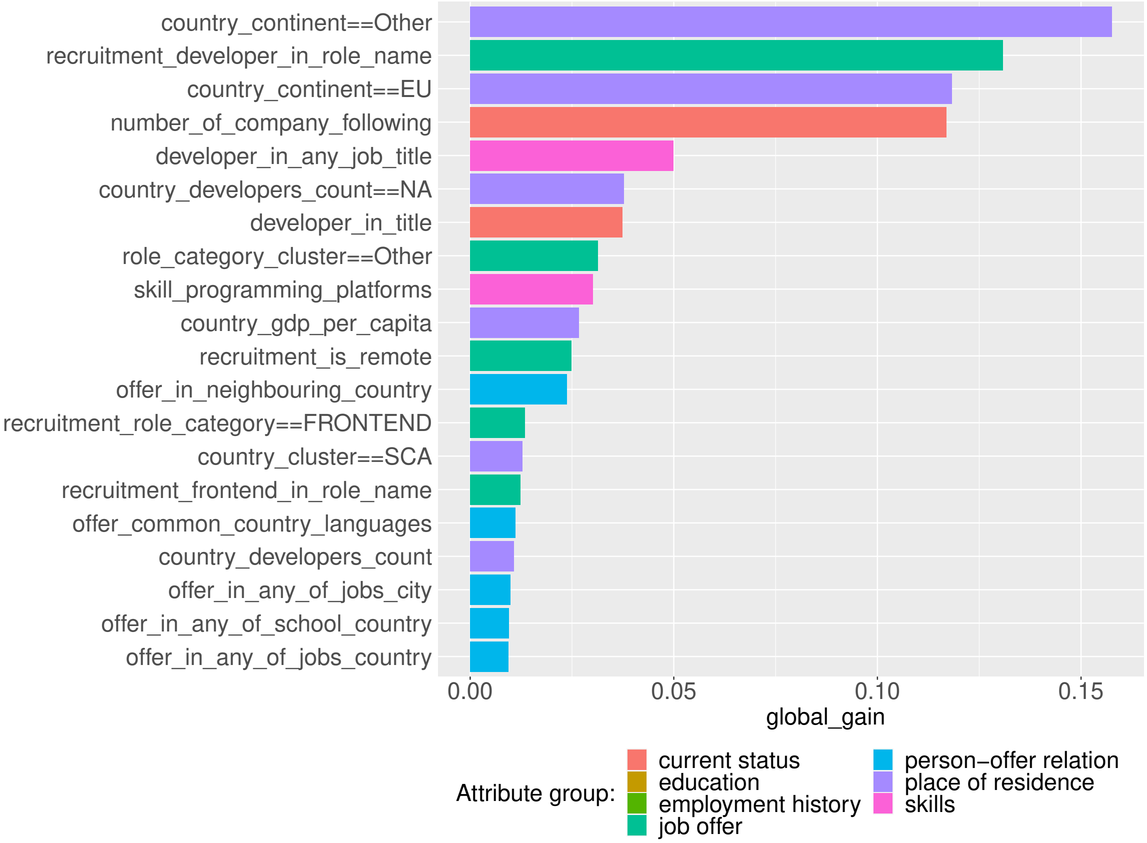


Figure 7.2: Visualization of the bireduct-specific importance of individual attributes (top 20) obtained using the normalized *global_gain* score values.

Let us now proceed with the discussion on comparing algorithms for ranking attributes' importance. For each considered ranking method, we conducted a statistical test to check whether the rankings of the original attributes from A and shuffled attributes from $A^{\circlearrowright}$ are significantly different. The Wilcoxon rank-sum test's null hypothesis is that two sets of values are drawn from the same distribution, whereby we assume that the ties in the attribute importance rankings are handled straightforwardly, by assigning the same rank to attributes with the same score. Table 7.5 presents the results for the chosen attribute importance ranking methods. Clearly, we cannot reject the null hypothesis for XGBoost (p-value $>$ 0.05). It confirmed our expectations – although the attribute importance ranks returned by XGBoost seem reasonable and consistent with the HR experts' intuition, we cannot judge them as sufficiently trustworthy insights. The observed situation is caused by the fact that XGBoost focuses on a small number of the most relevant attributes. As a result, during the model construction many original attributes were not taken into account at all, therefore, they were considered on a par with the shuffled ones. In turn, it makes the XGBoost's score distribution similar for attributes in A and $A^{\circlearrowright}$. Similarly to our approach in Section 6.4 for conducting experiments with ensembles of decision bireducts, we employed a quantile-based approach to discretize the numeric attributes into 3 intervals.

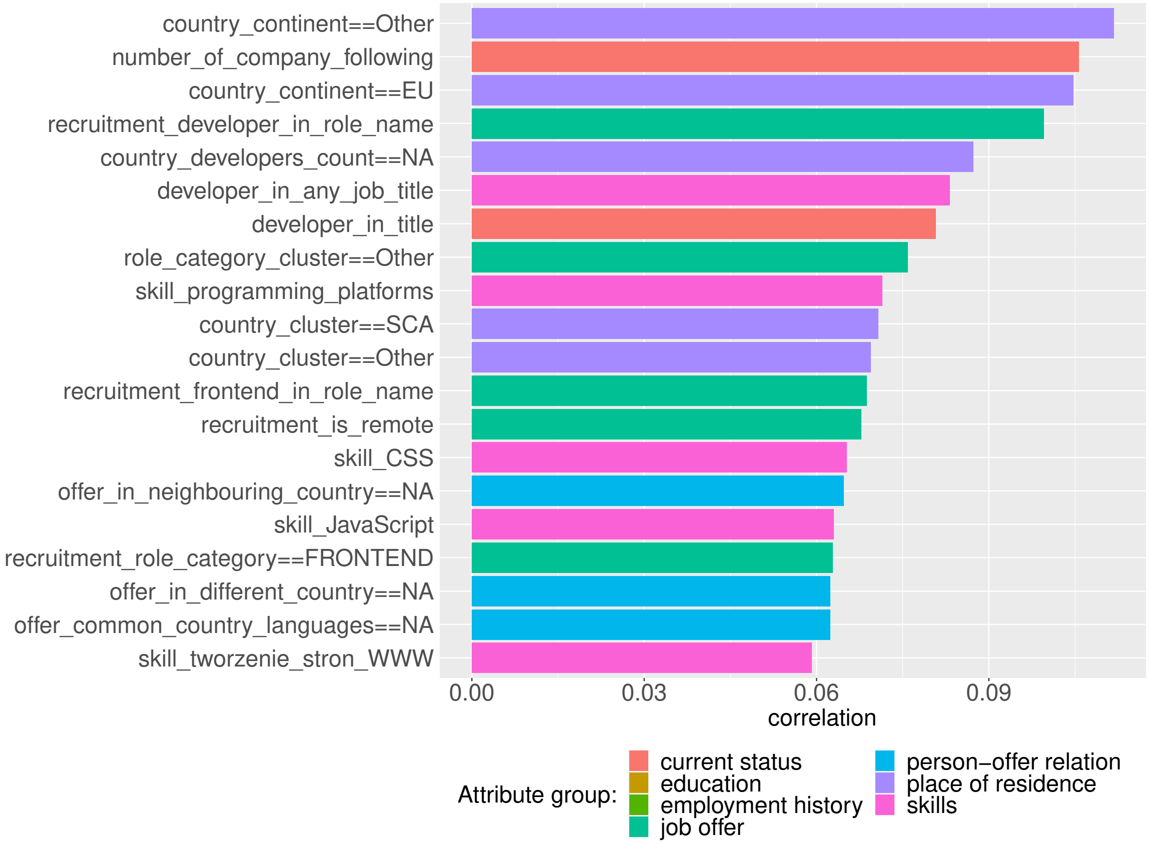


Figure 7.3: Visualization of the correlation-based importance of individual attributes (top 20) obtained using the normalized absolute values of *correlation*.

For this reason, we argue that a ranking comparison procedure should take into consideration and address the following two observations:

1. A good attribute importance ranking method should indeed distinguish between original and shuffled attributes.
2. Not all original attributes need to be considered as worth comparing, e.g., some of them might be irrelevant or noisy.

Thus, we propose to modify our initial approach by taking into account the ranks of only *top_k* top-ranked attributes among the original and shuffled attributes. In practice, the value of this parameter should reflect the expected (approximate) number of truly relevant features for the modeled prediction problem. The ranks of the chosen attributes from A and $A^{\circlearrowright}$ are then averaged giving the final output of two numbers – the average ranks of the *top_k* original and shuffled attributes. Such output can be interpreted in the following way:

1. The smaller the first number, the lower the number of the shuffled attributes that occur before the *top_k* original ones.

Table 7.5: Results for the compared feature ranking methods. We report the average rank of the original and shuffled attributes, and the p-value of the Wilcoxon rank-sum test.

method	average rank		p-value
	original attributes	shuffled attributes	
bireducts	709.27	835.73	2.5072e-08
XGBoost	767.00	778.00	6.2788e-01
correlation	642.58	902.42	2.3591e-30

2. The larger the second number, the lower the number of the original attributes that occur after *top_k* shuffled ones.

3. The greater the difference between the two numbers, the better the attribute importance ranking is in distinguishing between A and $A^{\circlearrowright}$.

The above observations can be used as a quality criterion for the comparison of different attribute ranking approaches. Since it does not depend on the type of the applied attribute scoring algorithm (general or model-dependent) and does not require any additional evaluation data, it allows for an objective assessment of the suitability of various attribute scoring methods to the given application. We can call this quality the *insightfulness* of an attribute ranking. This is because the unwanted results of the comparison between the elements of A and $A^{\circlearrowright}$ can indeed decrease the trust of SMEs not only in the attribute importance rankings themselves but also in the underlying decision models and machine learning methods applied to learn them.

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
10	5.50	548.70	5.50	778.00	5.50	149.50
20	10.50	694.10	353.40	778.00	10.50	181.45
30	15.50	742.57	494.93	778.00	15.50	201.17
50	25.50	781.34	608.16	778.00	25.52	230.94
100	50.55	810.42	693.08	778.00	50.80	296.93
all	709.27	835.73	767.00	778.00	642.58	902.42

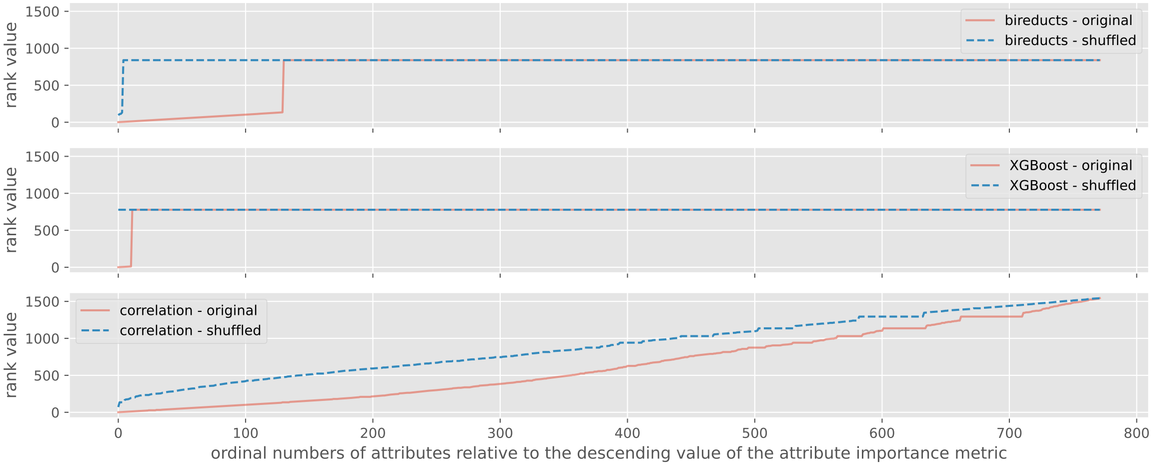


Figure 7.4: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the recruitment data table.

Figure 7.4 shows the results of our improved procedure for the chosen attribute importance ranking methods. Accordingly, we can reach following conclusions:

1. The bireducts and correlation methods are equally good at selecting large number of original attributes and do not focus on a relatively small subset of the best attributes – in some applications, it is preferable to operate on a set of diverse attributes, even though not all of them are of the highest quality.

2. The XGBoost algorithm tends to use only a small number of the most informative original attributes, and thus, is unable to distinguish between the less relevant original attributes and the shuffled ones.

3. The bireduct ensemble allows for much better distinguishing between the original and shuffled attributes than the Spearman correlation values and the attribute importance values computed using XGBoost.

4. The largest difference between the average ranks of *top_k* attributes, for nearly all investigated k values, was achieved by the ranking induced by the ensemble of bireducts. The only exception was for the smallest k – in that case, the highest difference was noted for XGBoost, however, the result for our bireduct ensemble was only slightly worse.

Overall, these results show that the ranking obtained using decision bireducts is more reliable than the one for XGBoost. The ensemble of bireducts distinguishes the importance scores of a much larger set of attributes than XGBoost, and at the same time, avoids mixing the original attributes with their shuffled counterparts as in the case of Spearman correlation-based ranking. This makes it more useful to SMEs who are looking to gain more insights from their data.

In addition to the aggregated perspective presented in Figure 7.4, we can also examine each result in detail separately. Such an approach can provide us with data for a deeper understanding of how a particular algorithm (with its hyperparameter settings) behaves with respect to the given data set in assessing the original and shuffled attributes. In Figures 7.5, 7.6 and 7.7 the appropriate (feature importance profiling) views are presented for bireducts, XGBoost and correlation-based algorithms assessing the attributes from $\mathbb{A}^{\circlearrowright} = (U, A \cup A^{\circlearrowright} \cup \{d\})$ prepared for our recruitment data table.

In Appendix A we also provide comprehensive results for all data used throughout the dissertation, i.e., UCI data sets (*zoo*, *lymphography*, and *SPECT*) used in Section 5.4 and synthetic (*synthetic1*, *synthetic2*, and *synthetic3*) as well as microarray (*acuteLymphoblasticLeukemia*, *hepatitisC*, *skinPsoriatic*) data sets from Section 7.1. For convenience, in Table 7.6 we present a summary of references navigating to appropriate results related to the particular data sets.

hyperparameters	
param	value
algorithm	bireducts
allowed_randomness	0.05
attrs_max_count	3
candidates_count	100
chaos_fun	gini_impurity
epsilon	0.00
n_bins	3
n_bireducts	1000
probes_count	100
model stats	
mean_attrs_size	3.00
mean_objs_size	17459.03
median_attrs_size	3.00
median_objs_size	18372.50

avg ranks		
top_k	original	shuffled
3	2.00	106.33
5	3.00	257.90
7	4.00	424.07
10	5.50	548.70
15	8.00	645.63
20	10.50	694.10
25	13.00	723.18
30	15.50	742.57
all	709.27	835.73

actual ranks			
column	count	global_gain	rank
g6_attr_714	172	0.27	1.00
g4_attr_641	149	0.23	2.00
g5_attr_691	176	0.23	3.00
g4_attr_640	141	0.19	4.00
g2_attr_047	110	0.11	5.00
g4_attr_668	113	0.10	6.00
g5_attr_676	107	0.10	7.00
g2_attr_309	95	0.08	8.00
g6_attr_730	90	0.07	9.00
g4_attr_633	76	0.06	10.00
g7_attr_756	76	0.06	11.00
g6_attr_701	70	0.06	12.00
g6_attr_711	78	0.06	13.00
g4_attr_658	78	0.05	14.00
g4_attr_657	69	0.04	15.00
g2_attr_458	53	0.04	16.00
g7_attr_753	53	0.04	17.00
g6_attr_718	47	0.03	18.00
g7_attr_755	40	0.03	19.00
g7_attr_757	42	0.03	20.00
...	...	...	...

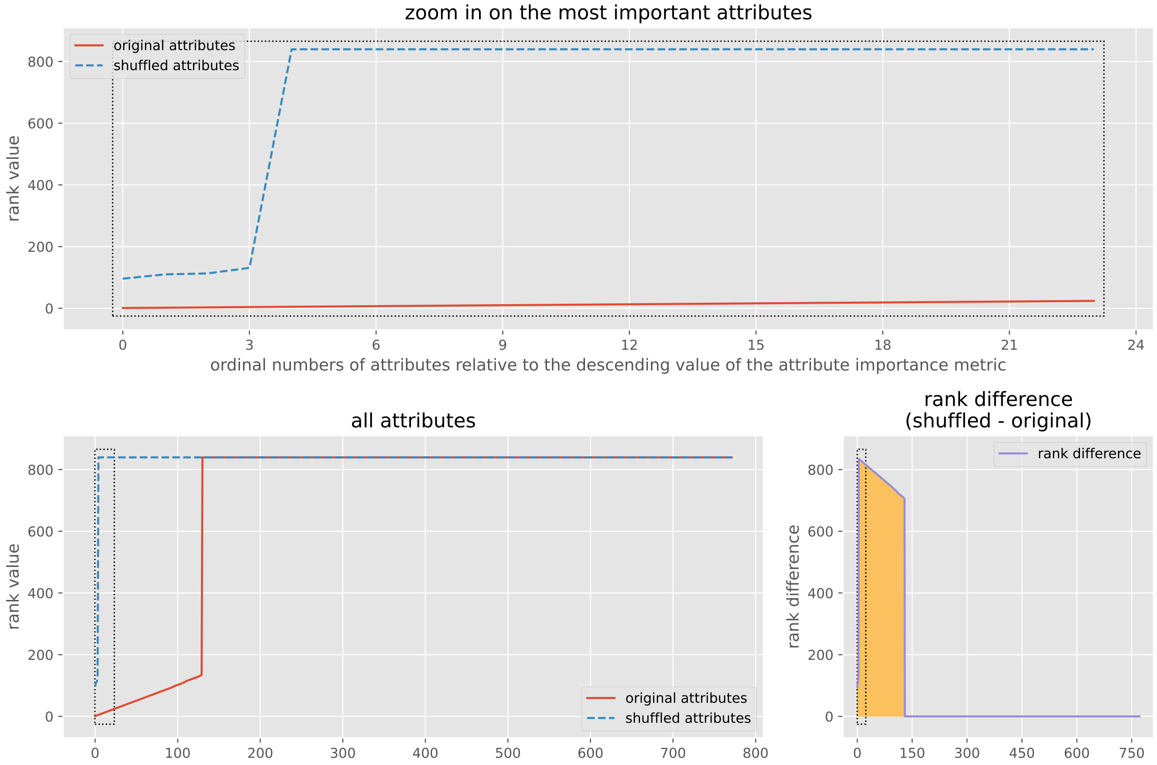


Figure 7.5: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the recruitment data table.

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	778.00	g5_attr_691	668	29931.64	1.00
learning_rate	0.00	5	3.00	778.00	g6_attr_714	967	26953.38	2.00
max_depth	2	7	4.00	778.00	g4_attr_641	705	22653.10	3.00
num_boost_round	1000	10	5.50	778.00	g4_attr_640	95	4342.96	4.00
objective	binary:logistic	15	211.87	778.00	g4_attr_638	156	3021.40	5.00
		20	353.40	778.00	g6_attr_701	136	2798.05	6.00
		25	438.32	778.00	g2_attr_182	110	1406.17	7.00
		30	494.93	778.00	g4_attr_635	53	768.85	8.00
		all	767.00	778.00	g1_attr_014	55	723.31	9.00
					g4_attr_625	50	662.17	10.00
					g4_attr_657	5	103.10	11.00
					shuffled_g2_attr_250	0	0.00	778.00
					shuffled_g2_attr_257	0	0.00	778.00
					shuffled_g2_attr_262	0	0.00	778.00
					shuffled_g2_attr_261	0	0.00	778.00
					shuffled_g2_attr_260	0	0.00	778.00
					shuffled_g2_attr_259	0	0.00	778.00
					shuffled_g2_attr_258	0	0.00	778.00
					shuffled_g2_attr_255	0	0.00	778.00
					shuffled_g2_attr_256	0	0.00	778.00
					...	...	...	...

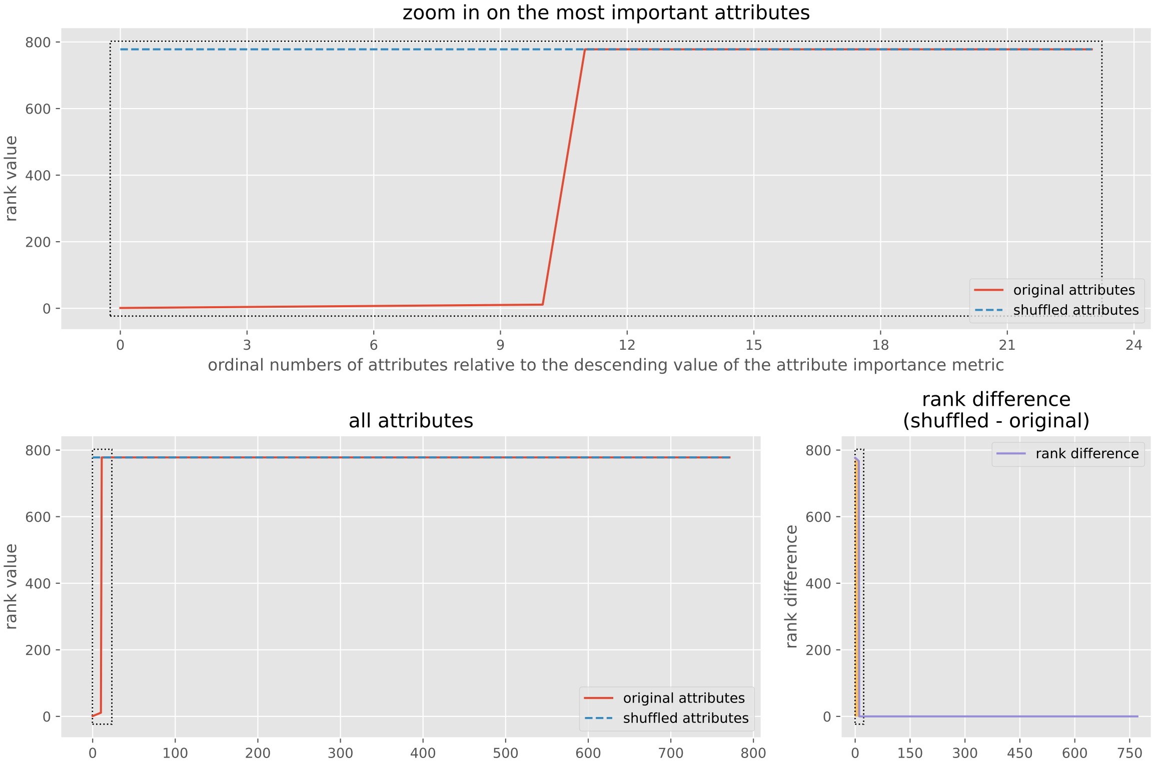
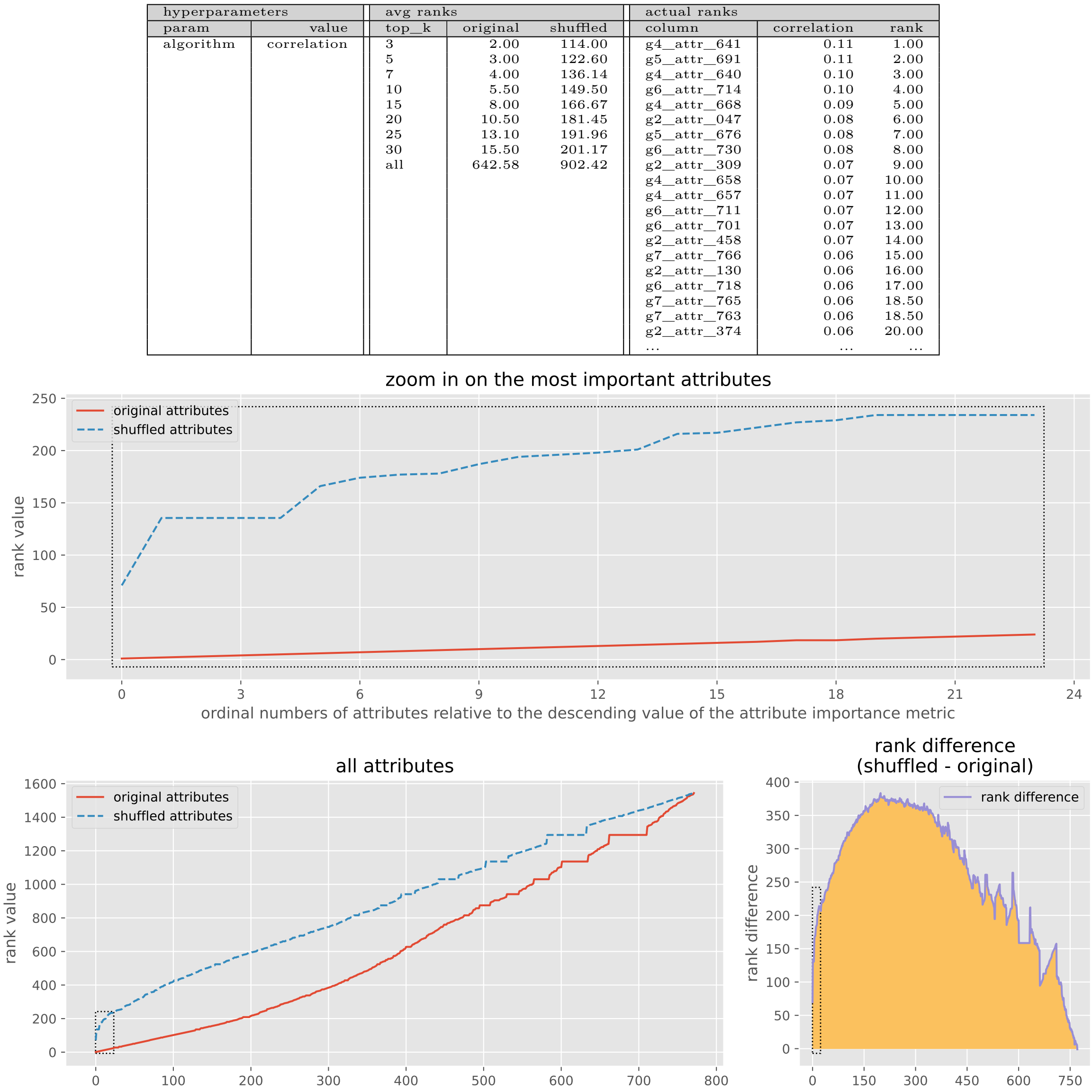


Figure 7.6: Detailed attribute importance profiling results obtained for XGBoost on the recruitment data table.



hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	114.00	g4_attr_641	0.11	1.00
		5	3.00	122.60	g5_attr_691	0.11	2.00
		7	4.00	136.14	g4_attr_640	0.10	3.00
		10	5.50	149.50	g6_attr_714	0.10	4.00
		15	8.00	166.67	g4_attr_668	0.09	5.00
		20	10.50	181.45	g2_attr_047	0.08	6.00
		25	13.10	191.96	g5_attr_676	0.08	7.00
		30	15.50	201.17	g6_attr_730	0.08	8.00
		all	642.58	902.42	g2_attr_309	0.07	9.00
					g4_attr_658	0.07	10.00
					g4_attr_657	0.07	11.00
					g6_attr_711	0.07	12.00
					g6_attr_701	0.07	13.00
					g2_attr_458	0.07	14.00
					g7_attr_766	0.06	15.00
					g2_attr_130	0.06	16.00
					g6_attr_718	0.06	17.00
					g7_attr_765	0.06	18.50
					g7_attr_763	0.06	18.50
					g2_attr_374	0.06	20.00
					...	...	...

zoom in on the most important attributes

original attributes
shuffled attributes
rank value
0
50
100
150
200
250
0
3
6
9
12
15
18
21
24
ordinal numbers of attributes relative to the descending value of the attribute importance metric

all attributes

rank difference
(shuffled - original)

Figure 7.7: Detailed attribute importance profiling results obtained for the correlation-based model on the recruitment data table.

Table 7.6: A complete list of available feature importance profiling results.

data set name	aggregate comparison	algorithm	detailed view
recruitment_data	Figure 7.4	bireducts	Figure 7.5
		XGBoost	Figure 7.6
		correlation	Figure 7.7
zoo	Figure A.1	bireducts	Figure A.2
		XGBoost	Figure A.3
		correlation	Figure A.4
lymphography	Figure A.5	bireducts	Figure A.6
		XGBoost	Figure A.7
		correlation	Figure A.8
SPECT	Figure A.9	bireducts	Figure A.10
		XGBoost	Figure A.11
		correlation	Figure A.12
synthetic1	Figure A.13	bireducts	Figure A.14
		XGBoost	Figure A.15
		correlation	Figure A.16
synthetic2	Figure A.17	bireducts	Figure A.18
		XGBoost	Figure A.19
		correlation	Figure A.20
synthetic3	Figure A.21	bireducts	Figure A.22
		XGBoost	Figure A.23
		correlation	Figure A.24
acuteLymphoblasticLeukemia	Figure A.25	bireducts	Figure A.26
		XGBoost	Figure A.27
		correlation	Figure A.28
hepatitisC	Figure A.29	bireducts	Figure A.30
		XGBoost	Figure A.31
		correlation	Figure A.32
skinPsoriatic	Figure A.33	bireducts	Figure A.34
		XGBoost	Figure A.35
		correlation	Figure A.36

Chapter 8

Concluding Remarks and Future Work

8.1 Summary

In the dissertation, we introduced decision bireducts, which can be seen as an extension of decision reducts in the theory of rough sets. The notion emphasizes on both a subset of attributes that describes decisions and a subset of objects for which that description is valid. We showed that the explicit attributes-objects duality utilized in decision bireducts provides a simple and flexible means of knowledge representation. In addition to the main definition, we also introduced other variants of the concept, such as the γ-decision bireduct related to the concept of positive region in rough set theory, as well as decision ε-bireducts and γ-decision ε-bireducts that express constraints regarding the proportion of covered objects.

From the theory of rough sets we revisited the notion of approximate decision reducts, which is defined as an irreducible subset of attributes that, under a specified criterion, preserves decision-related information of the entire attribute set above a specified threshold. Although the concept of decision bireducts is conceptually different from approximate decision reducts, our study revealed the presence of certain analogies. Furthermore, we explored the properties and relationships of both the concepts. Additionally, we demonstrated how decision bireducts can be utilized as rule-based classifiers that provide greater flexibility in assigning decision values to objects when compared to approximate decision reducts.

In the classical rough set approach, a propositional formula, called as "discernibility function", [115, 95] was formulated to represent the necessary conditions that need to be satisfied for all pairs of discernible objects. As a result, the collection of decision reducts corresponds to the set of all prime implicants of the discernibility function. We adapted this general approach to develop suitable Boolean formulae that capture the essential constraints related to decision bireducts and γ-decision bireducts.

We investigated the challenges associated with searching for decision bireducts in datasets and discussed effective methods for their computation. Through our research, we gained valuable insights into the practical applications of decision bireducts, highlighting their usefulness. For instance, we introduced a specific case of decision bireducts in the context of data streams and explored its utilization. Our focus was on scenarios where the complete dataset is not available during the computation process. Instead, we examined situations where events are processed incrementally, with each event arriving one at a time.

However, the primary focus of the dissertation was on ensembles of decision bireducts. We thoroughly examined their properties and characteristics. Moreover, we showed how decision bireducts may be used in constructing diverse and robust ensembles of classifiers. We introduced the notion of a correct ensemble which means that every object (training case) must be validly recognized using the corresponding rules by more than half of the classifiers from the ensemble. NP-hardness of the optimization problem was shown when it comes to finding the simplest correct ensemble of decision bireducts. In this context, the concept of "simplicity" was defined in a manner analogous to the approach presented in the study on generalized decision reducts [137]. Specifically, simplicity was determined based on the maximum cardinality among all subsets of attributes involved.

We comprehensively present a case study demonstrating the application of decision bireducts ensembles to a decision problem encountered while developing a solution for an HR company specializing in the recruitment of IT professionals. Moreover, we paid special attention to the interpretability of the models. We proposed several attribute importance metrics designed for decision bireduct ensembles. We also presented a general procedure for evaluating attribute importance methods. Furthermore, we used the procedure to compare the attribute importance scores provided by ensembles of decision bireducts against the importance scorings provided by XGBoost and correlation-based models on synthetic, benchmark, and real-life data sets.

Finally, as a result of our research, we have released a Python software library that provides a wide range of functions and algorithms associated with rough set theory. The library encompasses structures and methods for data processing and enables the application of techniques related to machine learning, in particular, the use of decision bireducts and their ensembles.

8.2 Future Work

In future, we will continue our studies on other types of bireducts. We also look forward to experimenting with various types of decision bireducts (e.g., decision bireducts for data streams – see Section 5.5) and various ways of constructing their ensembles, as well as testing the corresponding classifiers on a wider variety of benchmark data sets.

We would like to further investigate properties of bireducts in order to better utilize their advantages for providing more intuitive ways of visualization and interactive exploration of complex data, looking for inspiration, e.g., in the areas of formal concept analysis [35] or visual bi-clustering [44]. We also want to pay more attention to further extensions of our previous algorithmic approaches [140, 119] to deriving and applying decision bireducts for other examples of real-life data.

Regarding the theoretical computational complexity, it would be beneficial to utilize mathematical aparatus developed in [82] to strenghten the complexity results towards inapproximability theorems.

Another task that can help extend the range of applications is the problem of searching for ensembles of decision bireducts that would meet some more advanced criteria, e.g., how diversely an ensemble covers a data set with respect to both attributes and objects. The future work in this direction will concern theoretical basis of optimization criteria and computational complexity, as well as the development of practical algorithms applying different approaches ranging from greedy, to randomized or order-based solutions [137, 82].

Our great interest is also in induction of hierarchies of decision bireduct ensembles (see Figure 4.1). Efficient algorithms need to be developed to facilitate the construction process of such hierarchies. Moreover, the voting strategies used in such hierarchical classifiers may need to be researched for their influence on the achieved results. For that purpose, we want to investigate classifier fusion methods [110]. We can also draw inspiration from research related to conflict models and conflict resolutions [94, 25]. Considering ensembles of decision bireducts, where each component of the ensemble has its own local classification knowledge (represented by its subsets of attributes and objects), we can also benefit from exploring ideas proposed for dispersed systems [101, 102].

The procedure for evaluating the attribute importance methods requires further research. There are alternative approaches worth investigating. We can consider a decision problem of classifying the type of an attribute, whether it is an original or a shuffled (see Section 7.2) attribute. For such a defined task, we could utilize the methods from information retrieval domain, e.g., the receiver operating characteristic (ROC) curve or area under the ROC curve (AUC). We can also apply more advanced statistical-based approaches.

Bibliography

[1] Charu C. Aggarwal, editor. *Data Streams – Models and Algorithms*, volume 31 of *Advances in Database Systems*. Springer, 2007.

[2] Alejandro Barredo Arrieta, Natalia Diaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges Toward Responsible AI. *Information Fusion*, 58:82–115, 2020.

[3] Jan G. Bazan, Hung Son Nguyen, Sinh Hoa Nguyen, Piotr Synak, and Jakub Wróblewski. Rough Set Algorithms in Classification Problem. In Lech Polkowski, Shusaku Tsumoto, and Tsau Y. Lin, editors, *Rough Set Methods and Applications*, volume 56 of *Studies in Fuzziness and Soft Computing*, pages 49–88. Physica-Verlag, 2000.

[4] Jan G. Bazan, Andrzej Skowron, and Piotr Synak. Dynamic Reducts as a Tool for Extracting Laws from Decisions Tables. In Zbigniew W. Raś and Maria Zemankova, editors, *Methodologies for Intelligent Systems, 8th International Symposium, ISMIS '94, Charlotte, North Carolina, USA, October 16-19, 1994, Proceedings*, volume 869 of *Lecture Notes in Computer Science*, pages 346–355. Springer, 1994.

[5] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

[6] María Jose Benítez, Jesús Medina, and Dominik Ślęzak. *delta*-Information Reducts and Bireducts. In José Maria Alonso, Humberto Bustince, and Marek Z. Reformat, editors, *2015 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (IFSA-EUSFLAT-15), Gijón, Spain., June 30, 2015*, pages 1154 – 1160. Atlantis Press, 2015.

[7] María José Benítez-Caballero, Jesús Medina, and Eloísa Ramírez-Poussa. Towards a Classification of Rough Set Bireducts. In *IPMU (3)*, volume 1239 of *Communications in Computer and Information Science*, pages 759–770. Springer, 2020.

[8] María José Benítez-Caballero, Jesús Medina, Eloísa Ramírez-Poussa, and Dominik Ślęzak. Bireducts with tolerance relations. *Information Sciences*, 435:26–39, 2018.

[9] Jerzy Błaszczyński, Bartosz Prusak, and Roman Słowiński. Multi-objective Search for Comprehensible Rule Ensembles. In Victor Flores, Fernando Gomide, Andrzej Janusz, Claudio Meneses, Duoqian Miao, Georg Peters, Dominik Ślęzak, Guoyin Wang, Richard Weber, and Yiyu Yao, editors, *Proceedings of IJCRS 2016*, volume 9920 of *Lecture Notes in Computer Science*, pages 503–513, 2016.

[10] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis Dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[11] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender Systems Survey. *Knowledge-Based Systems*, 46(Supplement C):109–132, 2013.

[12] Sjoerd Boeschoten, Cagatay Catal, Bedir Tekinerdogan, Arjen Lommen, and Marco Blokland. The Automation of the Development of Classification Models and Improvement of Model Quality Using Feature Engineering Techniques. *Expert Systems with Applications*, 213(Part A):118912, 2023.

[13] Andrea Bommert, Xudong Sun, Bernd Bischl, Jörg Rahnenführer, and Michel Lang. Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis*, 143, 2020.

[14] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001.

[15] Leo Breiman, Jerome Friedman, Charles J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Taylor & Francis, 1984.

[16] Frank Markham Brown. *Boolean Reasoning: The Logic of Boolean Equations (2nd edition)*. Dover Books on Mathematics. Dover Publications, 2012.

[17] Jerzy Błaszczyński, Adiel Teixeira de Almeida Filho, Anna Matuszyk, Marcin Szeląg, and Roman Słowiński. Auto Loan Fraud Detection Using Dominance-based Rough Set Approach versus Machine Learning Methods. *Expert Systems with Applications*, 163:113740, 2021.

[18] Maura Cerioli, Maurizio Leotta, and Filippo Ricca. COVID-19 Hits the Job Market: An 88 Million Job Ads Analysis. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing (SAC 2021)*, pages 1721–1726, 2021.

[19] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, pages 785–794, 2016.

[20] Igor Chikalov, Vadim V. Lozin, Irina Lozina, Mikhail J. Moshkov, Hung Son Nguyen, Andrzej Skowron, and Beata Zielosko. *Three Approaches to Data Analysis – Test Theory, Rough Sets and Logical Analysis of Data*, volume 41 of *Intelligent Systems Reference Library*. Springer, 2013.

[21] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[22] David R. Cox. The regression analysis of binary sequences (with discussion). *Journal of the Royal Statistical Society*, 20:215–242, 1958.

[23] David R. Cox and David V. Hinkley. *Theoretical Statistics*. Chapman & Hall, London, England, 1974.

[24] Lynn D'eer and Chris Cornelis. Decision reducts and bireducts in a covering approximation space and their relationship to set definability. *International Journal of Approximate Reasoning*, 109:42–54, 2019.

[25] Rafał Deja and Andrzej Skowron. On some conflict models and conflict resolutions. *Romanian Journal of Information Science and Technology*, 3:69–82, 2002.

[26] Sebastien Delecraz, Loukman Eltarr, and Olivier Oullier. Transparency and Explainability of a Machine Learning Model in the Context of Human Resource Management. In *Proceedings of the Workshop on Ethical and Legal Issues in Human Language Technologies and Multilingual De-Identification of Sensitive Data In Language Resources within the 13th Language Resources and Evaluation Conference (LEGAL 2022)*, pages 38–43, 2022.

[27] Ren Diao, Neil Mac Parthaláin, Richard Jensen, and Qiang Shen. Heuristic search for fuzzy-rough bireducts and its use in classifier ensembles. In *FUZZ-IEEE*, pages 1504–1511. IEEE, 2014.

[28] Thomas G. Dietterich. An Experimental Comparison of Three Methods for Constructing Ensembles of Decision Trees: Bagging, Boosting, and Randomization. *Machine Learning*, 40(2):139–157, 2000.

[29] Chris H. Q. Ding and Hanchuan Peng. Minimum Redundancy Feature Selection from Microarray Gene Expression Data. In *2nd IEEE Computer Society Bioinformatics Conference, CSB 2003, Stanford, CA, USA, August 11-14, 2003*, pages 523–529. IEEE Computer Society, 2003.

[30] Michał Dramiński, Marcin Kierczak, Jacek Koronacki, and Henryk Jan Komorowski. Monte Carlo Feature Selection and Interdependency Discovery in Supervised Classification. In Jacek Koronacki, Zbigniew W. Raś, Sławomir T. Wierzchon, and Janusz Kacprzyk, editors, *Advances in Machine Learning II, Dedicated to the Memory of Professor Ryszard S. Michalski*, volume 263 of *Studies in Computational Intelligence*, pages 371–385. Springer, 2010.

[31] Włodzisław Duch, Tadeusz Wieczorek, Jacek Biesiada, and Marcin Blachnik. Comparison of Feature Ranking Methods Based on Information Entropy. In *Proceedings of the 2004 IEEE International Joint Conference on Neural Networks (IJCNN 2004) – Part II*, volume 2, pages 1415–1419, 2004.

[32] Kevin Fauvel, Élisa Fromont, Véronique Masson, Philippe Faverdin, and Alexandre Termier. XEM: An Explainable-by-Design Ensemble Method for Multivariate Time Series Classification. *Data Mining and Knowledge Discovery*, 36(3):917–957, 2022.

[33] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[34] Eibe Frank and Stefan Kramer. Ensembles of Nested Dichotomies for Multi-class Problems. In *Proceedings of ICML 2004*, volume 69 of *ACM International Conference Proceeding Series*, pages 84–95. ACM, 2004.

[35] Bernhard Ganter and Rudolf Wille. *Formal Concept Analysis - Mathematical Foundations.* Springer, 1999.

[36] Mateusz Garbulowski, Klev Diamanti, Karolina Smolińska, Nicholas Baltzer, Patricia Stoll, Susanne Bornelöv, Aleksander Øhrn, Lars Feuk, and Jan Komorowski. R.ROSETTA: An Interpretable Machine Learning Framework. *BMC Bioinformatics*, 22(1):110, 2021.

[37] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* A Series of Books in the Mathematical Sciences. W. H. Freeman and Company, 1979.

[38] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael A. Specter, and Lalana Kagal. Explaining Explanations: An Overview of Interpretability of Machine Learning. In *Proceedings of the 5th IEEE International Conference on Data Science and Advanced Analytics (DSAA 2018)*, pages 80–89, 2018.

[39] David Goretzko and Laura Sophia Finja Israel. Pitfalls of Machine Learning based Personnel Selection – Fairness, Transparency and Data Quality. *Journal of Personnel Psychology*, 2021.

[40] Isabelle Guyon and André Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[41] Isabelle Guyon, Masoud Nikravesh, Steve R. Gunn, and Lotfi A. Zadeh, editors. *Feature Extraction - Foundations and Applications*, volume 207 of *Studies in Fuzziness and Soft Computing*. Springer, 2006.

[42] Mark A. Hall. Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning. In Pat Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, pages 359–366. Morgan Kaufmann, 2000.

[43] Satoshi Hara and Kohei Hayashi. Making Tree Ensembles Interpretable: A Bayesian Model Selection Approach. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS 2018)*, pages 77–85, 2018.

[44] Timothy C. Havens and James C. Bezdek. A New Formulation of the coVAT Algorithm for Visual Assessment of Clustering Tendency in Rectangular Data. *International Journal of Intelligent Systems*, 27(6):590–612, 2012.

[45] Geoffrey Hinton and Sam Roweis. Stochastic Neighbor Embedding. *Advances in Neural Information Processing Systems*, 15:833–840, 2003.

[46] Xiaojuan Huang, Li Zhang, Bangjun Wang, Fanzhang Li, and Zhao Zhang. Feature clustering based support vector machine recursive feature elimination for gene selection. *Applied Intelligence*, 48(3):594–607, 2018.

[47] Pankhuri Jain, Anoop Kumar Tiwari, and Tanmoy Som. An intuitionistic fuzzy bireduct model and its application to cancer treatment. *Computers and Industrial Engineering*, 168:108124, 2022.

[48] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013.

[49] Andrzej Janusz. Utilization of Dynamic Reducts to Improve Performance of the Rule-Based Similarity Model for Highly-Dimensional Data. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops, Toronto, Canada, August 31 - September 3, 2010*, pages 432–435. IEEE Computer Society, 2010.

[50] Andrzej Janusz, Daniel Kałuża, Maciej Matraszek, Łukasz Grad, Maciej Świechowski, and Dominik Ślęzak. Learning Multimodal Entity Representations and Their Ensembles, with Applications in a Data-driven Advisory Framework for Video Game Players. *Information Sciences*, 617:193–210, 2022.

[51] Andrzej Janusz and Sebastian Stawicki. Applications of Approximate Reducts to the Feature Selection Problem. In Jingtao Yao, Sheela Ramanna, Guoyin Wang, and Zbigniew Suraj, editors, *Rough Sets and Knowledge Technology - 6th International Conference, RSKT 2011, Banff, Canada, October 9-12, 2011. Proceedings*, volume 6954 of *Lecture Notes in Computer Science*, pages 45–50. Springer, 2011.

[52] Andrzej Janusz, Sebastian Stawicki, Michał Drewniak, Krzysztof Ciebiera, Dominik Ślęzak, and Krzysztof Stencel. How to Match Jobs and Candidates - A Recruitment Support System Based on Feature Engineering and Advanced Analytics. In Jesús Medina, Manuel Ojeda-Aciego, José Luis Verdegay Galdeano, David A. Pelta, Inma P. Cabrera, Bernadette Bouchon-Meunier, and Ronald R. Yager, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations - 17th International Conference, IPMU 2018, Cádiz, Spain, June 11-15, 2018, Proceedings, Part II*, volume 854 of *Communications in Computer and Information Science*, pages 503–514. Springer, 2018.

[53] Andrzej Janusz, Sebastian Stawicki, Marcin S. Szczuka, and Dominik Ślęzak. Rough Set Tools for Practical Data Exploration. In Davide Ciucci, Guoyin Wang, Sushmita Mitra, and Wei-Zhi Wu, editors, *Rough Sets and Knowledge Technology - 10th International Conference, RSKT 2015, held as part of the International Joint Conference on Rough Sets, IJCRS 2015, Tianjin, China, November 20-23, 2015, Proceedings*, volume 9436 of *Lecture Notes in Computer Science*, pages 77–86. Springer, 2015.

[54] Andrzej Janusz and Dominik Ślęzak. Utilization of Attribute Clustering Methods for Scalable Computation of Reducts from High-Dimensional Data. In Maria Ganzha, Leszek A. Maciaszek, and Marcin Paprzycki, editors, *Federated Conference on Computer Science and Information Systems - FedCSIS 2012, Wroclaw, Poland, 9-12 September 2012, Proceedings*, pages 295–302. IEEE Computer Society, 2012.

[55] Andrzej Janusz and Dominik Ślęzak. Rough Set Methods for Attribute Clustering and Selection. *Applied Artificial Intelligence*, 28(3):220–242, 2014.

[56] Andrzej Janusz and Dominik Ślęzak. Computation of Approximate Reducts with Dynamically Adjusted Approximation Threshold. In Floriana Esposito, Olivier Pivert, Mohand-Saïd Hacid, Zbigniew W. Raś, and Stefano Ferilli, editors, *Foundations of Intelligent Systems - 22nd International Symposium, ISMIS 2015, Lyon, France, October 21-23, 2015, Proceedings*, volume 9384 of *Lecture Notes in Computer Science*, pages 19–28. Springer, 2015.

[57] Andrzej Janusz and Dominik Ślęzak. KnowledgePit Meets BrightBox: A Step Toward Insightful Investigation of the Results of Data Science Competitions. In *Proceedings of the 17th Conference on Computer Science and Intelligence Systems (FedCSIS 2022)*, pages 393–398, 2022.

[58] Andrzej Janusz, Dominik Ślęzak, and Hung Son Nguyen. Unsupervised Similarity Learning from Textual Data. *Fundamenta Informaticae*, 119(3-4):319–336, 2012.

[59] Andrzej Janusz, Dominik Ślęzak, Sebastian Stawicki, and Krzysztof Stencel. A Practical Study of Methods for Deriving Insightful Attribute Importance Rankings Using Decision Bireducts. *Information Sciences*, 2023.

[60] George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant Features and the Subset Selection Problem. In William W. Cohen and Haym Hirsh, editors, *Machine Learning, Proceedings of the Eleventh International Conference, Rutgers University, New Brunswick, NJ, USA, July 10-13, 1994*, pages 121–129. Morgan Kaufmann, 1994.

[61] Kenji Kira and Larry A. Rendell. A Practical Approach to Feature Selection. In Derek H. Sleeman and Peter Edwards, editors, *Proceedings of the Ninth International Workshop on Machine Learning (ML 1992), Aberdeen, Scotland, UK, July 1-3, 1992*, pages 249–256. Morgan Kaufmann, 1992.

[62] Ron Kohavi and George H. John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97(1-2):273–324, 1997.

[63] Andrei V. Konstantinov and Lev V. Utkin. Interpretable Machine Learning with an Ensemble of Gradient Boosting Machines. *Knowledge-Based Systems*, 222:106993, 2021.

[64] Marcin Kruczyk, Nicholas Baltzer, Jakub Mieczkowski, Michał Dramiński, Jacek Koronacki, and Jan Komorowski. Random Reducts: A Monte Carlo Rough Set-based Method for Feature Selection in Large Datasets. *Fundamenta Informaticae*, 127(1-4):273–288, 2013.

[65] Marzena Kryszkiewicz. Comparative study of alternative types of knowledge reduction in inconsistent systems. *International Journal of Intelligent Systems*, 16(1):105–120, 2001.

[66] Miron B. Kursa, Aleksander Jankowski, and Witold R. Rudnicki. Boruta - A System for Feature Selection. *Fundamenta Informaticae*, 101(4):271–285, 2010.

[67] Miron B. Kursa and Witold R. Rudnicki. Feature Selection with the Boruta Package. *Journal of Statistical Software*, 36(11):1–13, 2010.

[68] Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML 2014)*, pages 1188–1196, 2014.

[69] Jia Li, Dhruv Arya, Viet Ha-Thuc, and Shakti Sinha. How to Get Them a Dream Job?: Entity-Aware Features for Personalized Job Search Ranking. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, pages 501–510, 2016.

[70] Chen Liao, Shutao Li, and Zhiyuan Luo. Gene Selection Using Wilcoxon Rank Sum Test and Support Vector Machine for Cancer Classification. In Yuping Wang, Yiu-ming Cheung, and Hai-Lin Liu, editors, *Computational Intelligence and Security, International Conference, CIS 2006, Guangzhou, China, November 3-6, 2006, Revised Selected Papers*, volume 4456 of *Lecture Notes in Computer Science*, pages 57–66. Springer, 2006.

[71] Moshe Lichman. UCI Machine Learning Repository, 2013.

[72] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender System Application Developments: A Survey. *Decision Support Systems*, 74(Supplement C):12–32, 2015.

[73] Scott M. Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M. Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From Local Explanations to Global Understanding with Explainable AI for Trees. *Nature Machine Intelligence*, 2(1):56–67, 2020.

[74] Scott M. Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 4768–4777. Curran Associates Inc., 2017.

[75] Xi'ao Ma and Yiyu Yao. Min-max attribute-object bireducts: On unifying models of reducts in rough set theory. *Information Sciences*, 501:68–83, 2019.

[76] David Maier. *The Theory of Relational Databases*. Computer Science Press, 1983.

[77] Sebastián Maldonado and Julio López. Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for SVM classification. *Applied Soft Computing*, 67:94–105, 2018.

[78] Simon J. Mason and Nicholas E. Graham. Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation. *Quarterly Journal of the Royal Meteorological Society*, 128(584):2145–2166, 2002.

[79] Sameep Mehta, Rakesh Pimplikar, Amit Singh, Lav R. Varshney, and Karthik Visweswariah. Efficient Multifaceted Screening of Job Applicants. In *Proceedings of the 16th International Conference on Extending Database Technology (EDBT 2013)*, pages 661–671, 2013.

[80] Maciej Modrzejewski. Feature Selection Using Rough Sets Theory. In Pavel Brazdil, editor, *Machine Learning: ECML-93, European Conference on Machine Learning, Vienna, Austria, April 5-7, 1993, Proceedings*, volume 667 of *Lecture Notes in Computer Science*, pages 213–226. Springer, 1993.

[81] Jose Morales-Arilla and Carlos Daboin. Is Remote Work in High Demand? Evidence from Job Postings during COVID-19. In *Proceedings of the ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS 2021)*, pages 27–37, 2021.

[82] Mikhail J. Moshkov, Marcin Piliszczuk, and Beata Zielosko. *Partial Covers, Reducts and Decision Rules in Rough Sets – Theory and Applications*, volume 145 of *Studies in Computational Intelligence*. Springer, 2008.

[83] Dang Nguyen, Sunil Gupta, Santu Rana, Alistair Shilton, and Svetha Venkatesh. Fairness Improvement for Black-Box Classifiers with Gaussian Process. *Information Sciences*, 576:542–556, 2021.

[84] Hung Son Nguyen. On Efficient Handling of Continuous Attributes in Large Data Bases. *Fundamenta Informaticae*, 48(1):61–81, 2001.

[85] Hung Son Nguyen. Approximate Boolean Reasoning: Foundations and Applications in Data Mining. *LNCS Transactions on Rough Sets*, V:334–506, 2006.

[86] Hung Son Nguyen and Dominik Ślęzak. Approximate Reducts and Association Rules - Correspondence and Complexity Results. In Ning Zhong, Andrzej Skowron, and Setsuo Ohsuga, editors, *New Directions in Rough Sets, Data Mining, and Granular-Soft Computing, 7th International Workshop, RSFDGrC '99, Yamaguchi, Japan, November 9-11, 1999, Proceedings*, volume 1711 of *Lecture Notes in Computer Science*, pages 137–145. Springer, 1999.

[87] Sinh Hoa Nguyen and Hung Son Nguyen. Pattern Extraction from Data. *Fundamenta Informaticae*, 34(1-2):129–144, 1998.

[88] M. Tamer Özsu. A Systematic View of Data Science. *IEEE Data Engineering Bulletin*, 43(3):3–11, 2020.

[89] Neil Mac Parthaláin and Richard Jensen. Simultaneous Feature And Instance Selection Using Fuzzy-Rough Bireducts. In *FUZZ-IEEE 2013, IEEE International Conference on Fuzzy Systems, Hyderabad, India, 7-10 July, 2013, Proceedings*, pages 1–8. IEEE, 2013.

[90] Neil Mac Parthaláin, Richard Jensen, and Ren Diao. Fuzzy-Rough Set Bireducts for Data Reduction. *IEEE Transactions on Fuzzy Systems*, 28(8):1840–1850, 2020.

[91] Zdzisław Pawlak. *Rough Sets: Theoretical Aspects of Reasoning about Data*, volume 9 of *Theory and decision library : series D*. Kluwer, 1991.

[92] Zdzisław Pawlak. Rough Sets: Present State and the Future. *Foundations of Computing and Decision Sciences*, 18(3–4):157–166, 1993.

[93] Zdzisław Pawlak. Rough Set Elements. In Lech Polkowski and Andrzej Skowron, editors, *Rough Sets in Knowledge Discovery 1 – Methodology and Applications*, volume 18 of *Studies in Fuzziness and Soft Computing*, pages 10–30. Physica-Verlag, 1998.

[94] Zdzisław Pawlak. Conflicts and Negotations. In *RSKT*, volume 4062 of *Lecture Notes in Computer Science*, pages 12–27. Springer, 2006.

[95] Zdzisław Pawlak and Andrzej Skowron. Rough Sets and Boolean Reasoning. *Information Sciences*, 177(1):41–73, 2007.

[96] Zdzisław Pawlak and Andrzej Skowron. Rudiments of Rough Sets. *Information Sciences*, 177(1):3–27, 2007.

[97] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[98] Hanchuan Peng, Fuhui Long, and Chris H. Q. Ding. Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1226–1238, 2005.

[99] W. Peterson, T. Birdsall, and W. Fox. The theory of signal detectability. *Transactions of the IRE Professional Group on Information Theory*, 4(4):171–212, 1954.

[100] Robi Polikar, Joseph DePasquale, Hussein Syed Mohammed, Gavin Brown, and Ludmilla I. Kuncheva. Learn++.MF: A Random Subspace Approach for the Missing Feature Problem. *Pattern Recognition*, 43(11):3817–3832, 2010.

[101] Małgorzata Przybyła-Kasperek. Selected Methods of Combining Classifiers, when Predictions are Stored in Probability Vectors, in a Dispersed Decision-making System. *Fundamenta Informaticae*, 147(2-3):353–370, 2016.

[102] Małgorzata Przybyła-Kasperek. Three Conflict Methods in Multiple Classifiers that Use Dispersed Knowledge. *International Journal of Information Technology and Decision Making*, 18(2):555–599, 2019.

[103] Jose Quevedo, Antonio Bahamonde, and Oscar Luaces. A Simple and Efficient Method for Variable Ranking According to Their Usefulness for Learning. *Computational Statistics & Data Analysis*, 52:578–595, 2007.

[104] J. Ross Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[105] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[106] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[107] Iman Ramezani, Mojtaba Khorram Niaki, Milad Dehghani, and Mostafa Rezapour. Stability Analysis of Feature Ranking Techniques in the Presence of Noise: A Comparative Study. *International Journal of Business Intelligence and Data Mining*, 17:413, 2020.

[108] Payam Refaeilzadeh, Lei Tang, and Huan Liu. On Comparison of Feature Selection Algorithms. In *Proceedings of the AAAI 2007 Workshop on Evaluation Methods for Machine Learning II*, 2007.

[109] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-Precision Model-Agnostic Explanations. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*, pages 1527–1535, 2018.

[110] Dymitr Ruta and Bogdan Gabrys. An Overview of Classifier Fusion Methods. *Computing and Information Systems*, 7:1–10, 01 2000.

[111] Omer Sagi and Lior Rokach. Approximating XGBoost with an Interpretable Decision Tree. *Information Sciences*, 572:522–542, 2021.

[112] Borja Seijo-Pardo, Veronica Bolón-Canedo, Iago Porto-Díaz, and Amparo Alonso-Betanzos. Ensemble Feature Selection for Rankings of Features. In *Proceedings of the International Work-Conference on Artificial Neural Networks (IWANN 2015) – Part II*, pages 29–42, 2015.

[113] Wojciech W. Siedlecki and Jack Sklansky. On Automatic Feature Selection. In Chi Hau Chen, L. F. Pau, and Patrick S. P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision*, pages 63–87. World Scientific, 1993.

[114] Amit Singh, Catherine Rose, Karthik Visweswariah, Vijil Chenthamarakshan, and Nandakishore Kambhatla. PROSPECT: A System for Screening Candidates for Recruitment. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, pages 659–668, 2010.

[115] Andrzej Skowron and Cecylia Rauszer. *The Discernibility Matrices and Functions in Information Systems*, pages 331–362. Springer Netherlands, Dordrecht, 1992.

[116] Andrzej Skowron and Jarosław Stepaniuk. Information granules: Towards foundations of granular computing. *International Journal of Intelligent Systems*, 16(1):57–85, 2001.

[117] Sebastian Stawicki and Sebastian Widz. Decision Bireducts and Approximate Decision Reducts: Comparison of Two Approaches to Attribute Subset Ensemble Construction. In Maria Ganzha, Leszek A. Maciaszek, and Marcin Paprzycki, editors, *Federated Conference on Computer Science and Information Systems - FedCSIS 2012, Wroclaw, Poland, 9-12 September 2012, Proceedings*, pages 331–338, 2012.

[118] Sebastian Stawicki and Dominik Ślęzak. Recent Advances in Decision Bireducts: Complexity, Heuristics and Streams. In Pawan Lingras, Marcin Wolski, Chris Cornelis, Sushmita Mitra, and Piotr Wasilewski, editors, *Rough Sets and Knowledge Technology - 8th International Conference, RSKT 2013, Halifax, NS, Canada, October 11-14, 2013, Proceedings*, volume 8171 of *Lecture Notes in Computer Science*, pages 200–212. Springer, 2013.

[119] Sebastian Stawicki, Dominik Ślęzak, Andrzej Janusz, and Sebastian Widz. Decision Bireducts and Decision Reducts – A Comparison. *International Journal of Approximate Reasoning*, 84:75–109, 2017.

[120] Roger Alan Stein, Patricia A. Jaques, and João Francisco Valiati. An Analysis of Hierarchical Text Classification Using Word Embeddings. *Information Sciences*, 471:216–232, 2019.

[121] Trevor Hastie and Robert Tibshirani and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer series in statistics. Springer, 2009.

[122] Julio J. Valdés and Alan J. Barton. Relevant Attribute Discovery in High Dimensional Data: Application to Breast Cancer Gene Expressions. In Guoyin Wang, James F. Peters, Andrzej Skowron, and Yiyu Yao, editors, *Rough Sets and Knowledge Technology, First International Conference, RSKT 2006, Chongqing, China, July 24-26, 2006, Proceedings*, volume 4062 of *Lecture Notes in Computer Science*, pages 482–489. Springer, 2006.

[123] A. Wayne Whitney. A Direct Method of Nonparametric Measurement Selection. *IEEE Transactions on Computers*, 20(9):1100–1103, 1971.

[124] Sebastian Widz and Dominik Ślęzak. Approximation Degrees in Decision Reduct-based MRI Segmentation. In Daniel Howard and Phill-Kyu Rhee, editors, *Frontiers in the Convergence of Bioscience and Information Technologies 2007, FBIT 2007, Jeju Island, Korea, October 11-13, 2007*, pages 431–436. IEEE Computer Society, 2007.

[125] Sebastian Widz and Dominik Ślęzak. Attribute Subset Quality Functions Over a Universe of Weighted Objects. In Marzena Kryszkiewicz, Chris Cornelis, Davide Ciucci, Jesús Medina-Moreno, Hiroshi Motoda, and Zbigniew W. Raś, editors, *Rough Sets and Intelligent Systems Paradigms - Second International Conference, RSEISP 2014, Held as Part of JRS 2014, Granada and Madrid, Spain, July 9-13, 2014. Proceedings*, volume 8537 of *Lecture Notes in Computer Science*, pages 99–110. Springer, 2014.

[126] Maksymilian Wojtas and Ke Chen. Feature Importance Ranking for Deep Learning. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems (NIPS 2020)*, pages 5105–5114, 2020.

[127] Xing Yi, James Allan, and W. Bruce Croft. Matching Resumes and Jobs Based on Relevance Models. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 809–810, 2007.

[128] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In H. V. Jagadish and Inderpal Singh Mumick, editors, *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, pages 103–114. ACM Press, 1996.

[129] Chen Zhu, Hengshu Zhu, Hui Xiong, Chao Ma, Fang Xie, Pengliang Ding, and Pan Li. Person-Job Fit: Adapting the Right Talent for the Right Job with Joint Representation Learning. *ACM Transactions on Management Information Systems*, 9(3):12:1–12:17, 2018.

[130] Łukasz A. Kurgan, Krzysztof J. Cios, Ryszard Tadeusiewicz, Marek R. Ogiela, and Lucy S. Goodenday. Knowledge Discovery Approach to Automated Cardiac SPECT Diagnosis. *Artificial Intelligence in Medicine*, 23(2):149–169, 2001.

[131] Dominik Ślęzak. Approximate Reducts in Decision Tables. In *Proceedings of IPMU 1996*, volume 3, pages 1159–1164, 1996.

[132] Dominik Ślęzak. Searching for dynamic reducts in inconsistent decision tables. In *Proceedings of IPMU 1996*, pages 1362–1369, 1998.

[133] Dominik Ślęzak. Decomposition and Synthesis of Decision Tables with Respect to Generalized Decision Functions. In Sankar K. Pal and Andrzej Skowron, editors, *Rough Fuzzy Hybridization – A New Trend in Decision Making*, pages 110–135. Springer, 1999.

[134] Dominik Ślęzak. Normalized Decision Functions and Measures for Inconsistent Decision Tables Analysis. *Fundamenta Informaticae*, 44(3):291–319, 2000.

[135] Dominik Ślęzak. *Various Approaches to Reasoning with Frequency Based Decision Reducts: A Survey*, pages 235–285. Physica-Verlag HD, 2000.

[136] Dominik Ślęzak. Approximate Entropy Reducts. *Fundamenta Informaticae*, 53(3-4):365–390, 2002.

[137] Dominik Ślęzak. On Generalized Decision Functions: Reducts, Networks and Ensembles. In Yiyu Yao, Qinghua Hu, Hong Yu, and Jerzy W. Grzymała-Busse, editors, *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing - 15th International Conference, RSFDGrC 2015, Tianjin, China, November 20-23, 2015, Proceedings*, volume 9437 of *Lecture Notes in Computer Science*, pages 13–23. Springer, 2015.

[138] Dominik Ślęzak and Soma Dutta. Dynamic and Discernibility Characteristics of Different Attribute Reduction Criteria. In Hung Son Nguyen, Quang-Thuy Ha, Tianrui Li, and Małgorzata Przybyła-Kasperek, editors, *Rough Sets - International Joint Conference, IJCRS 2018, Quy Nhon, Vietnam, August 20-24, 2018, Proceedings*, volume 11103 of *Lecture Notes in Computer Science*, pages 628–643. Springer, 2018.

[139] Dominik Ślęzak, Marek Grzegorowski, Andrzej Janusz, Michał Kozielski, Sinh Hoa Nguyen, Marek Sikora, Sebastian Stawicki, and Łukasz Wróbel. A framework for learning and embedding multi-sensor forecasting models into a decision support system: A case study of methane concentration in coal mines. *Information Sciences*, 451-452:112–133, 2018.

[140] Dominik Ślęzak and Andrzej Janusz. Ensembles of Bireducts: Towards Robust Classification and Simple Representation. In Tai-Hoon Kim, Hojjat Adeli, Dominik Ślęzak, Frode Eika Sandnes, Xiaofeng Song, Kyo-Il Chung, and Kirk P. Arnett, editors, *Future Generation Information Technology - Third International Conference, FGIT 2011 in Conjunction with GDC 2011, Jeju Island, Korea, December 8-10, 2011. Proceedings*, volume 7105 of *Lecture Notes in Computer Science*, pages 64–77. Springer, 2011.

[141] Dominik Ślęzak, Marcin Kowalski, Victoria Eastwood, and Jakub Wróblewski. Methods and Systems for Database Organization. US Patent 8,266,147 B2, 2012.

[142] Dominik Ślęzak and Sebastian Stawicki. The Problem of Finding the Simplest Classifier Ensemble is NP-Hard - A Rough-Set-Inspired Formulation Based on Decision Bireducts. In Rafael Bello, Duoqian Miao, Rafael Falcon, Michinori Nakata, Alejandro Rosete, and Davide Ciucci, editors, *Rough Sets - International Joint Conference, IJCRS 2020, Havana, Cuba, June 29 - July 3, 2020, Proceedings*, volume 12179 of *Lecture Notes in Computer Science*, pages 204–212. Springer, 2020.

[143] Dominik Ślęzak and Sebastian Widz. Is It Important Which Rough-Set-Based Classifier Extraction and Voting Criteria Are Applied Together? In Marcin S. Szczuka, Marzena Kryszkiewicz, Sheela Ramanna, Richard Jensen, and Qinghua Hu, editors, *Proceedings of RSCTC 2010*, volume 6086 of *Lecture Notes in Computer Science*, pages 187–196. Springer, 2010.

[144] Dominik Ślęzak and Jakub Wróblewski. Roughfication of Numeric Decision Tables: The Case Study of Gene Expression Data. In Jingtao Yao, Pawan Lingras, Wei-Zhi Wu, Marcin S. Szczuka, Nick Cercone, and Dominik Ślęzak, editors, *Rough Sets and Knowledge Technology, Second International Conference, RSKT 2007, Toronto, Canada, May 14-16, 2007, Proceedings*, volume 4481 of *Lecture Notes in Computer Science*, pages 316–323. Springer, 2007.

Appendix A

Feature Importance Profiling

A.1 Feature Importance Profiling - *zoo* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
3	2.00	16.67	2.00	10.67	2.00	13.00
5	3.00	17.80	3.00	14.20	3.00	14.80
7	4.00	18.86	4.00	17.29	4.00	16.14
10	5.50	20.75	5.90	19.60	5.50	18.20
15	8.07	23.33	10.47	21.40	9.53	21.47
20	9.34	23.66	11.38	21.62	10.88	22.12
all	9.34	23.66	11.38	21.62	10.88	22.12

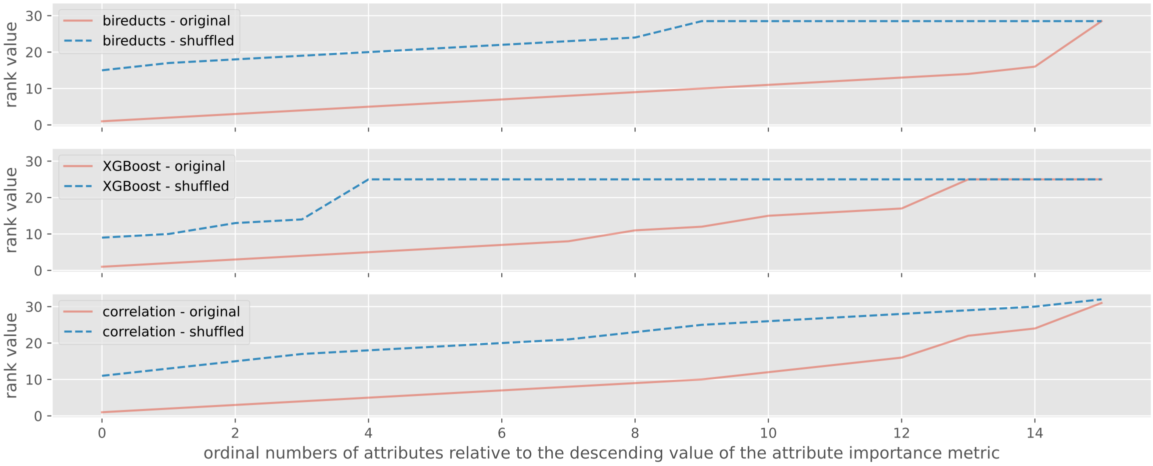


Figure A.1: The average attribute importance ranks of the original (A) and shuffled $(A^{\circlearrowright})$ attributes computed for the compared ranking methods on the *zoo* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	count	global_gain	rank
algorithm	bireducts	3	2.00	16.67	legs	332	68.30	1.00
allowed_randomness	0.05	5	3.00	17.80	milk	285	64.77	2.00
attrs_max_count	3	7	4.00	18.86	eggs	258	52.55	3.00
candidates_count	5	10	5.50	20.75	toothed	259	42.22	4.00
chaos_fun	gini_impurity	all	9.34	23.66	hair	241	41.20	5.00
epsilon	0.00				feathers	236	33.62	6.00
n_bins	3				backbone	206	29.87	7.00
n_bireducts	1000				breathes	198	26.65	8.00
probes_count	100				fins	170	17.24	9.00
model stats					tail	136	17.01	10.00
mean_attrs_size	2.78				airborne	137	13.40	11.00
mean_objs_size	71.45				aquatic	105	9.14	12.00
median_attrs_size	3.00				catsize	74	4.02	13.00
median_objs_size	75.00				venomous	41	1.55	14.00
					shuffled_aquatic	25	0.92	15.00
					predator	21	0.86	16.00
					shuffled_domestic	15	0.49	17.00
					shuffled_backbone	21	0.47	18.00
					shuffled_legs	7	0.36	19.00
					shuffled_catsize	7	0.17	20.00
					...	...	...	...

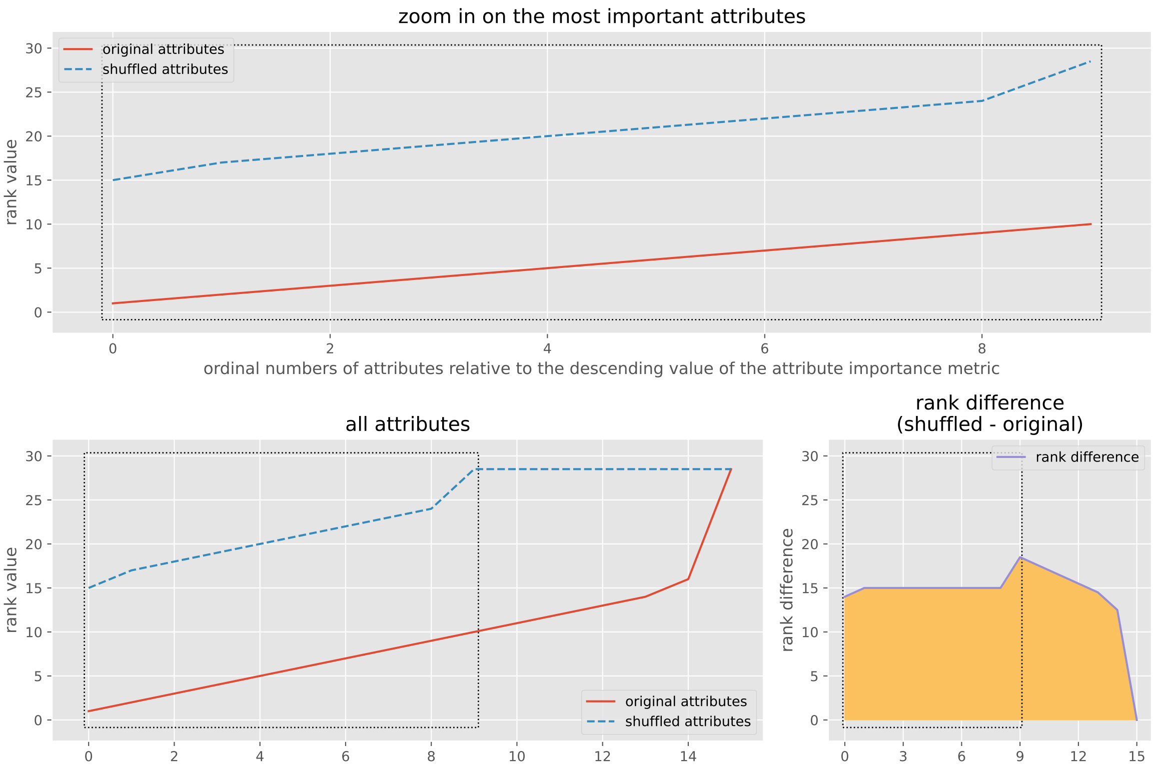


Figure A.2: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *zoo* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	10.67	milk	1000	32597.78	1.00
learning_rate	0.00	5	3.00	14.20	feathers	1000	24074.37	2.00
max_depth	2	7	4.00	17.29	fins	1131	14881.32	3.00
num_boost_round	1000	10	5.90	19.60	legs	1967	10906.16	4.00
objective	multi:softmax	all	11.38	21.62	backbone	1000	9445.51	5.00
					airborne	1000	3237.11	6.00
					eggs	552	2133.03	7.00
					predator	1000	1327.39	8.00
					shuffled_fins	735	894.04	9.00
					shuffled_tail	1000	828.87	10.00
					venomous	869	809.44	11.00
					aquatic	488	479.76	12.00
					shuffled_venomous	735	375.05	13.00
					shuffled_aquatic	172	169.57	14.00
					tail	199	160.73	15.00
					toothed	71	55.02	16.00
					catsize	78	0.02	17.00
					shuffled_breathes	0	0.00	25.00
					hair	0	0.00	25.00
					shuffled_legs	0	0.00	25.00
					...	...	...	...

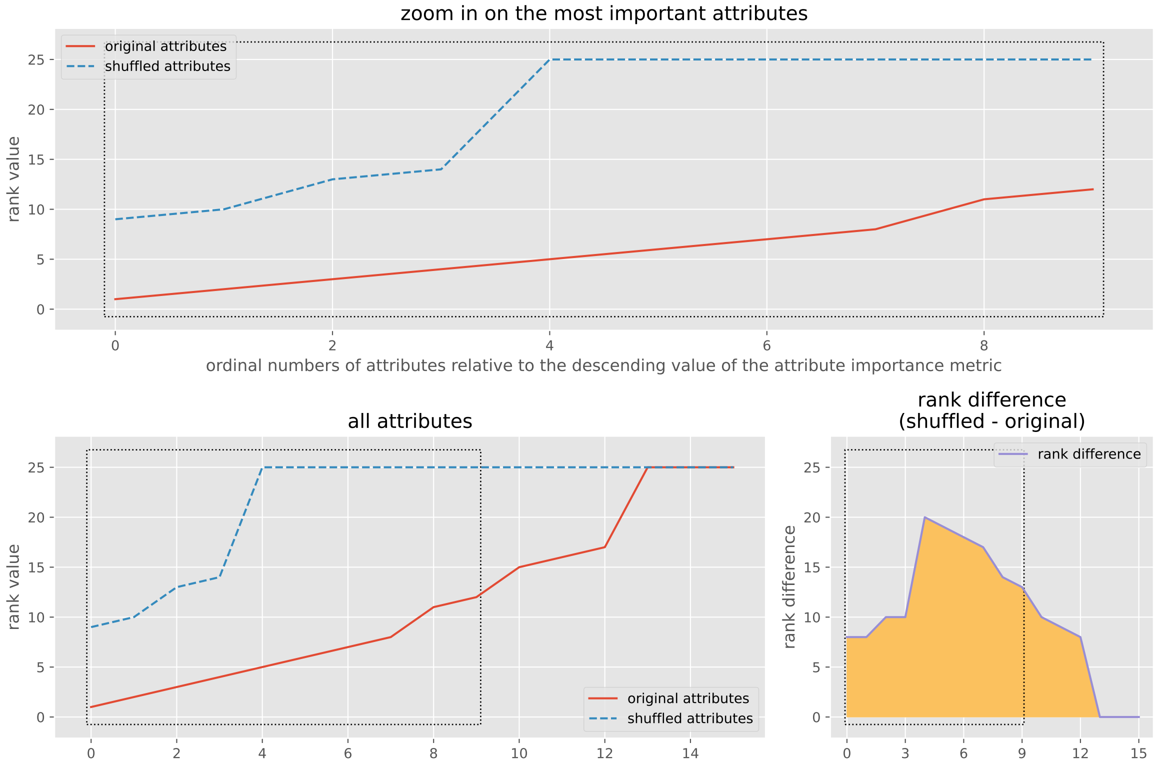


Figure A.3: Detailed attribute importance profiling results obtained for XGBoost on the *zoo* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	13.00	milk	0.89	1.00
		5	3.00	14.80	eggs	0.82	2.00
		7	4.00	16.14	hair	0.73	3.00
		10	5.50	18.20	backbone	0.69	4.00
		all	10.88	22.12	catsize	0.58	5.00
					breathes	0.54	6.00
					toothed	0.54	7.00
					tail	0.50	8.00
					aquatic	0.38	9.00
					venomous	0.32	10.00
					shuffled_domestic	0.20	11.00
					domestic	0.19	12.00
					shuffled_catsize	0.17	13.00
					fins	0.16	14.00
					shuffled_toothed	0.16	15.00
					airborne	0.14	16.00
					shuffled_milk	0.11	17.00
					shuffled_airborne	0.10	18.00
					shuffled_predator	0.10	19.00
					shuffled_venomous	0.07	20.00
					...	...	...

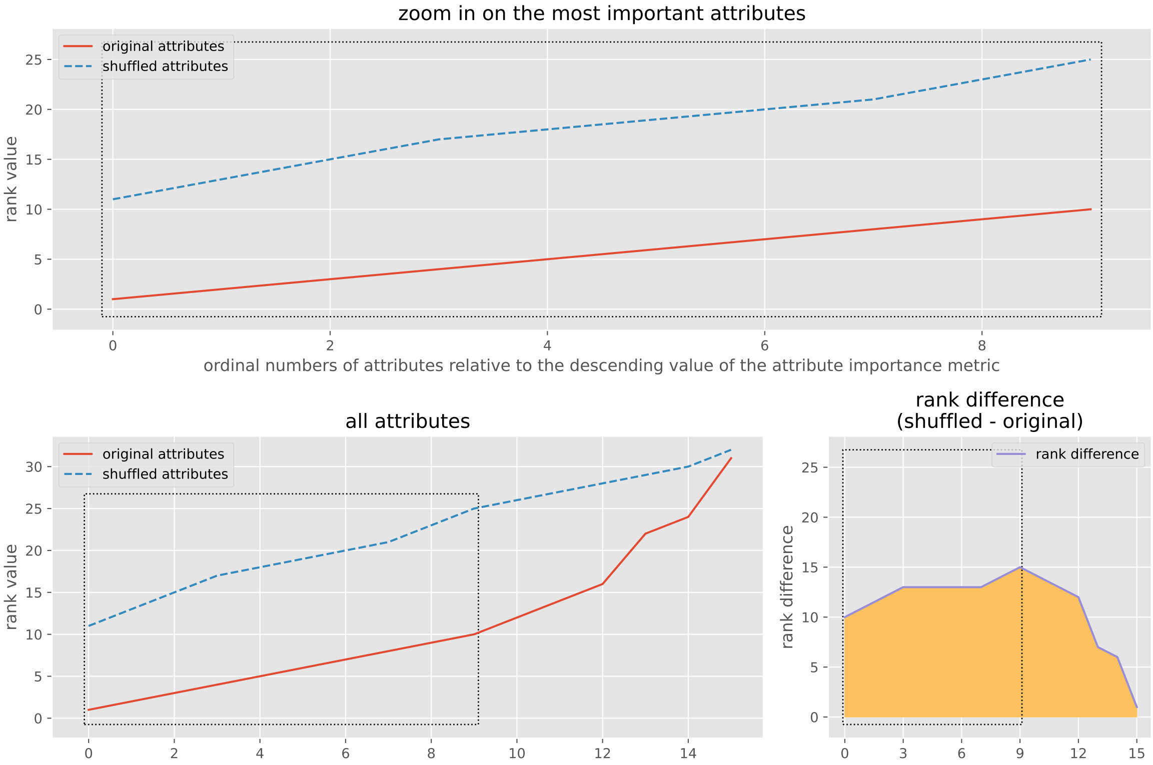
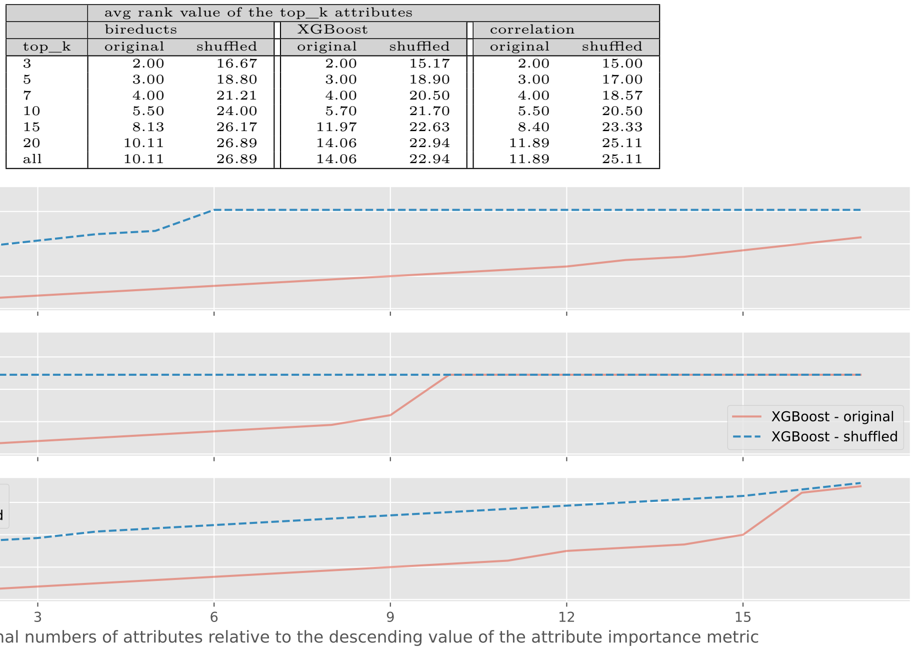


Figure A.4: Detailed attribute importance profiling results obtained for the correlation-based model on the *zoo* data set.

A.2 Feature Importance Profiling - *lymphography* data set



	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
3	2.00	16.67	2.00	15.17	2.00	15.00
5	3.00	18.80	3.00	18.90	3.00	17.00
7	4.00	21.21	4.00	20.50	4.00	18.57
10	5.50	24.00	5.70	21.70	5.50	20.50
15	8.13	26.17	11.97	22.63	8.40	23.33
20	10.11	26.89	14.06	22.94	11.89	25.11
all	10.11	26.89	14.06	22.94	11.89	25.11

Figure A.5: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowleft}$) attributes computed for the compared ranking methods on the *lymphography* data set.

Bireducts - detailed profile

hyperparameters	
param	value
algorithm	bireducts
allowed_randomness	0.05
attrs_max_count	3
candidates_count	5
chaos_fun	gini_impurity
epsilon	0.00
n_bins	3
n_bireducts	1000
probes_count	100
model stats	
mean_attrs_size	2.42
mean_objs_size	92.43
median_attrs_size	3.00
median_objs_size	96.00

avg ranks		
top_k	original	shuffled
3	2.00	16.67
5	3.00	18.80
7	4.00	21.21
10	5.50	24.00
15	8.13	26.17
all	10.11	26.89

actual ranks			
column	count	global_gain	rank
block_of_affere	323	35.43	1.00
no_of_nodes_in	293	26.78	2.00
special_forms	282	24.64	3.00
lymnodes_enlar	237	18.27	4.00
early_uptake_in	220	13.58	5.00
changes_in_stru	172	8.99	6.00
changes_in_lym	160	8.50	7.00
lymnodes_dimin	144	5.61	8.00
regeneration_of	122	4.92	9.00
changes_in_node	99	3.87	10.00
dislocation_of	80	2.84	11.00
exclusion_of_no	63	2.04	12.00
by_pass	43	1.87	13.00
shuffled_lymphatics	44	1.68	14.00
lymphatics	47	1.56	15.00
bl_of_lymph_c	31	1.02	16.00
shuffled_changes_in_stru	17	0.75	17.00
extravasates	18	0.65	18.00
shuffled_exclusion_of_no	8	0.23	19.00
defect_in_node	8	0.14	20.00
...	...	...	...

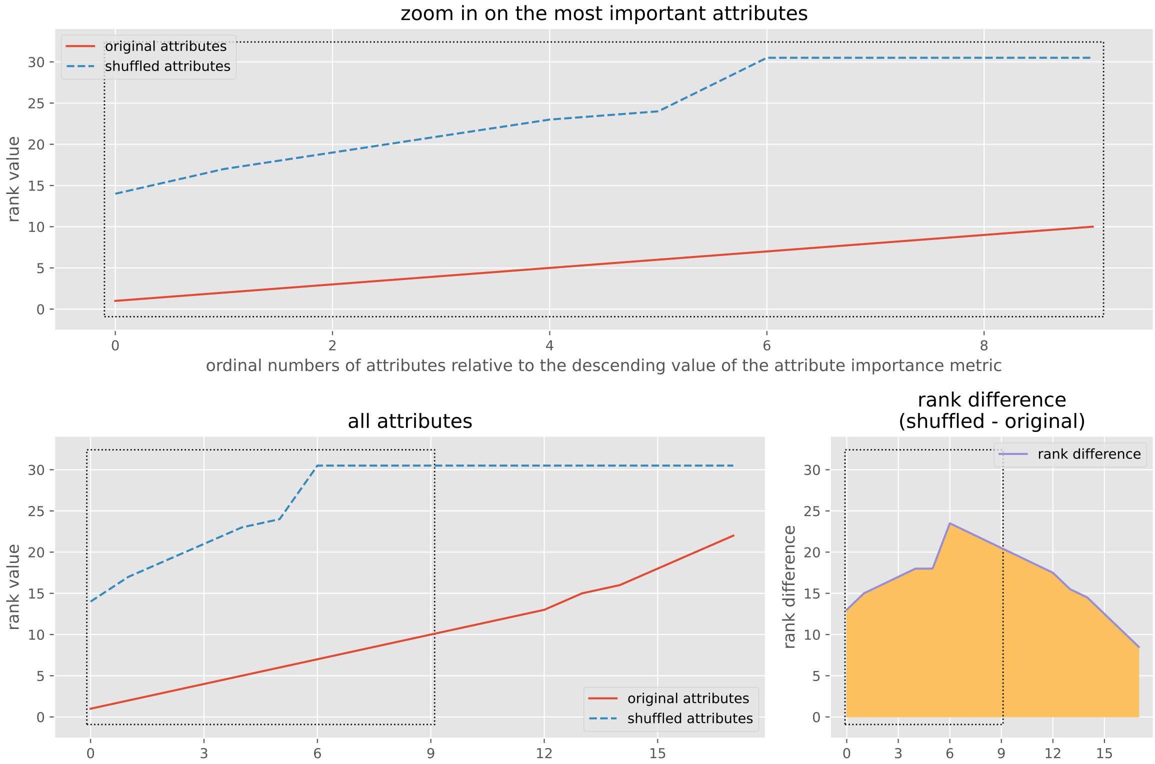


Figure A.6: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *lymphography* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	15.17	changes_in_node	1383	10324.98	1.00
learning_rate	0.00	5	3.00	18.90	block_of_affere	1437	8932.14	2.00
max_depth	2	7	4.00	20.50	no_of_nodes_in	1290	7846.27	3.00
num_boost_round	1000	10	5.70	21.70	special_forms	843	5964.19	4.00
objective	multi:softmax	15	11.97	22.63	lymnodes_dimin	1489	4601.46	5.00
		all	14.06	22.94	early_uptake_in	487	3522.22	6.00
					defect_in_node	1000	2257.43	7.00
					lymnodes_enlar	316	2195.74	8.00
					changes_in_lym	151	197.51	9.00
					shuffled_lymphatics	357	139.53	10.00
					shuffled_block_of_affere	93	87.68	11.00
					changes_in_stru	154	64.62	12.00
					shuffled_changes_in_node	0	0.00	24.50
					shuffled_regeneration_of	0	0.00	24.50
					shuffled_changes_in_stru	0	0.00	24.50
					shuffled_special_forms	0	0.00	24.50
					shuffled_defect_in_node	0	0.00	24.50
					shuffled_changes_in_lym	0	0.00	24.50
					shuffled_dislocation_of	0	0.00	24.50
					shuffled_exclusion_of_no	0	0.00	24.50
					...	...	...	...

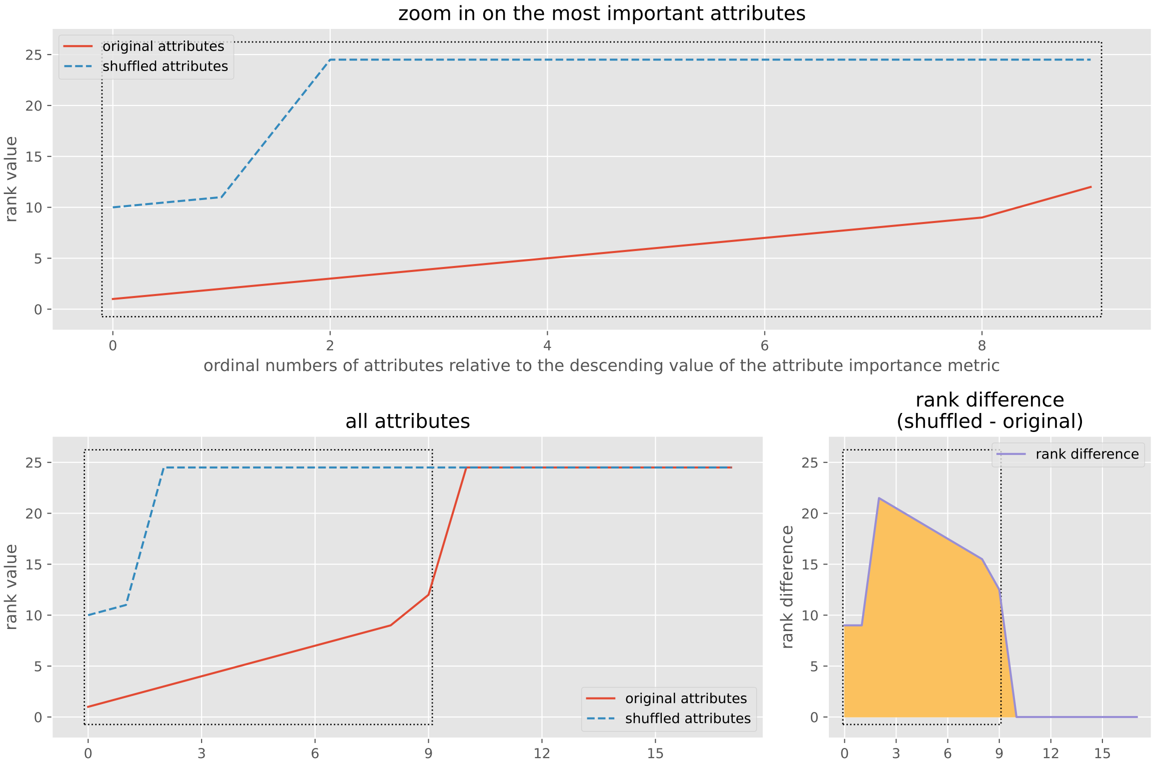


Figure A.7: Detailed attribute importance profiling results obtained for XGBoost on the *lymphography* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	15.00	no_of_nodes_in	0.54	1.00
		5	3.00	17.00	special_forms	0.41	2.00
		7	4.00	18.57	block_of_affere	0.40	3.00
		10	5.50	20.50	regeneration_of	0.33	4.00
		15	8.40	23.33	lymnodes_enlar	0.31	5.00
		all	11.89	25.11	early_uptake_in	0.30	6.00
					changes_in_stru	0.29	7.00
					lymnodes_dimin	0.27	8.00
					exclusion_of_no	0.24	9.00
					dislocation_of	0.23	10.00
					bl_of_lymph_s	0.17	11.00
					changes_in_lym	0.16	12.00
					shuffled_lymnodes_dimin	0.15	13.00
					shuffled_dislocation_of	0.15	14.00
					lymphatics	0.12	15.00
					extravasates	0.10	16.00
					defect_in_node	0.10	17.00
					shuffled_lymnodes_enlar	0.09	18.00
					shuffled_defect_in_node	0.09	19.00
					bl_of_lymph_c	0.09	20.00
					...	...	...

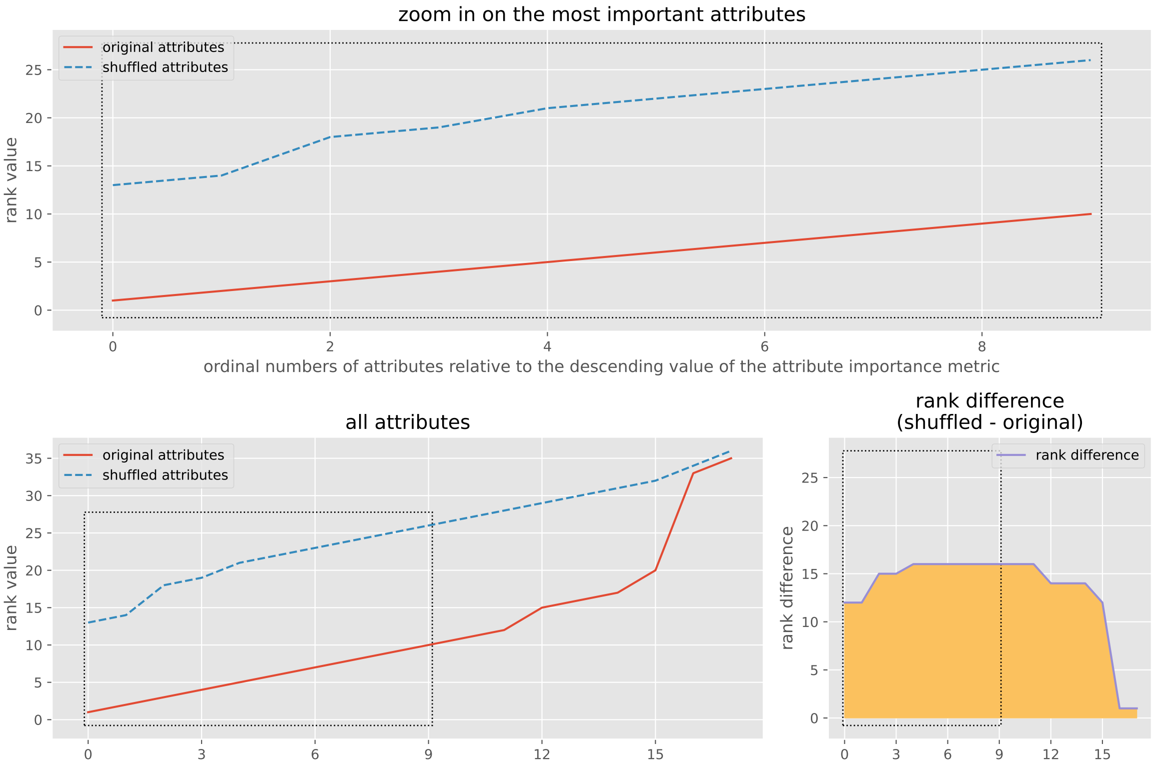


Figure A.8: Detailed attribute importance profiling results obtained for the correlation-based model on the *lymphography* data set.

A.3 Feature Importance Profiling - *SPECT* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
3	2.00	17.67	2.00	19.67	2.00	23.67
5	3.00	21.20	3.40	22.80	3.00	24.80
7	4.00	23.29	4.57	24.14	4.00	25.86
10	5.50	25.60	7.85	25.15	5.50	27.40
15	8.27	29.90	14.40	25.93	8.00	29.93
20	11.15	32.05	17.68	26.32	10.50	32.45
all	12.36	32.64	18.57	26.43	11.55	33.45

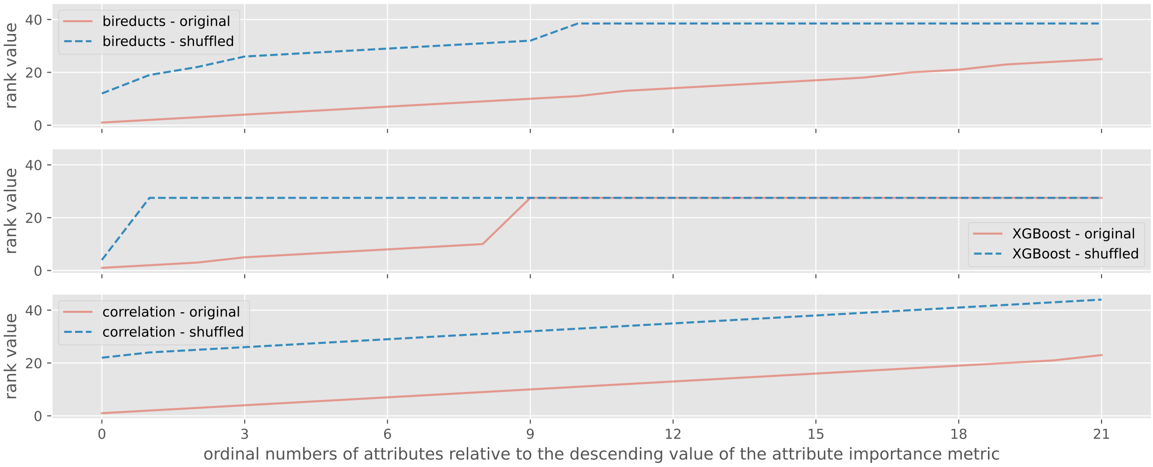


Figure A.9: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *SPECT* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	count	global_gain	rank
algorithm	bireducts	3	2.00	17.67	F13	291	11.33	1.00
allowed_randomness	0.05	5	3.00	21.20	F8	241	6.05	2.00
attrs_max_count	3	7	4.00	23.29	F22	205	4.69	3.00
candidates_count	5	10	5.50	25.60	F16	207	4.50	4.00
chaos_fun	gini_impurity	15	8.27	29.90	F21	191	4.08	5.00
epsilon	0.00	20	11.15	32.05	F10	146	2.94	6.00
n_bins	3	all	12.36	32.64	F1	141	2.75	7.00
n_bireducts	1000				F20	146	2.31	8.00
probes_count	100				F7	139	2.02	9.00
model stats					F17	140	1.96	10.00
mean_attrs_size	2.80				F11	112	1.89	11.00
mean_objs_size	194.19				shuffled_F22	101	1.62	12.00
median_attrs_size	3.00				F2	87	1.09	13.00
median_objs_size	200.00				F3	106	1.05	14.00
					F6	79	1.05	15.00
					F4	74	0.95	16.00
					F5	59	0.89	17.00
					F12	69	0.84	18.00
					shuffled_F21	51	0.64	19.00
					F14	38	0.45	20.00
					...	...	...	...

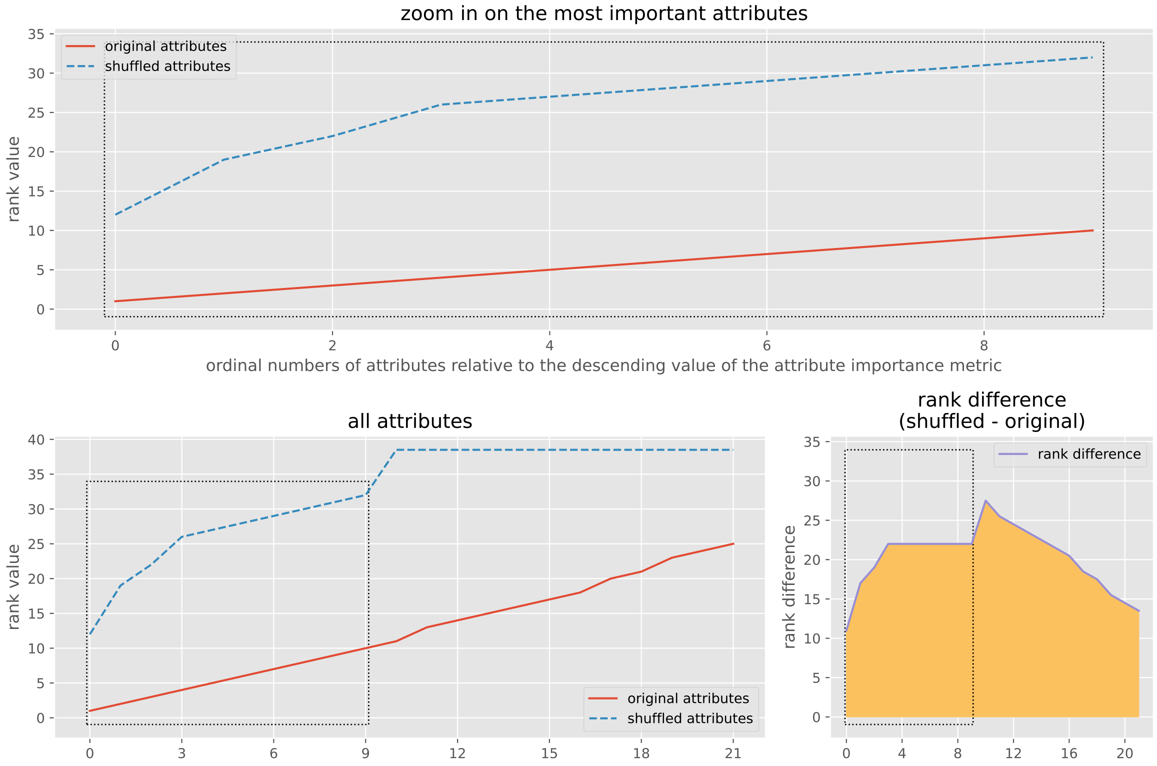


Figure A.10: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *SPECT* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	19.67	F13	1000	12080.84	1.00
learning_rate	0.00	5	3.40	22.80	F1	223	1545.04	2.00
max_depth	2	7	4.57	24.14	F16	255	1467.72	3.00
num_boost_round	1000	10	7.85	25.15	shuffled_F3	212	1181.34	4.00
objective	binary:logistic	15	14.40	25.93	F6	148	847.01	5.00
		20	17.68	26.32	F20	100	520.51	6.00
		all	18.57	26.43	F11	58	291.98	7.00
					F22	560	89.14	8.00
					F8	70	21.69	9.00
					F7	4	18.66	10.00
					shuffled_F13	0	0.00	27.50
					shuffled_F8	0	0.00	27.50
					shuffled_F9	0	0.00	27.50
					shuffled_F10	0	0.00	27.50
					shuffled_F11	0	0.00	27.50
					shuffled_F12	0	0.00	27.50
					shuffled_F15	0	0.00	27.50
					shuffled_F14	0	0.00	27.50
					shuffled_F6	0	0.00	27.50
					shuffled_F16	0	0.00	27.50
					...	...	...	...

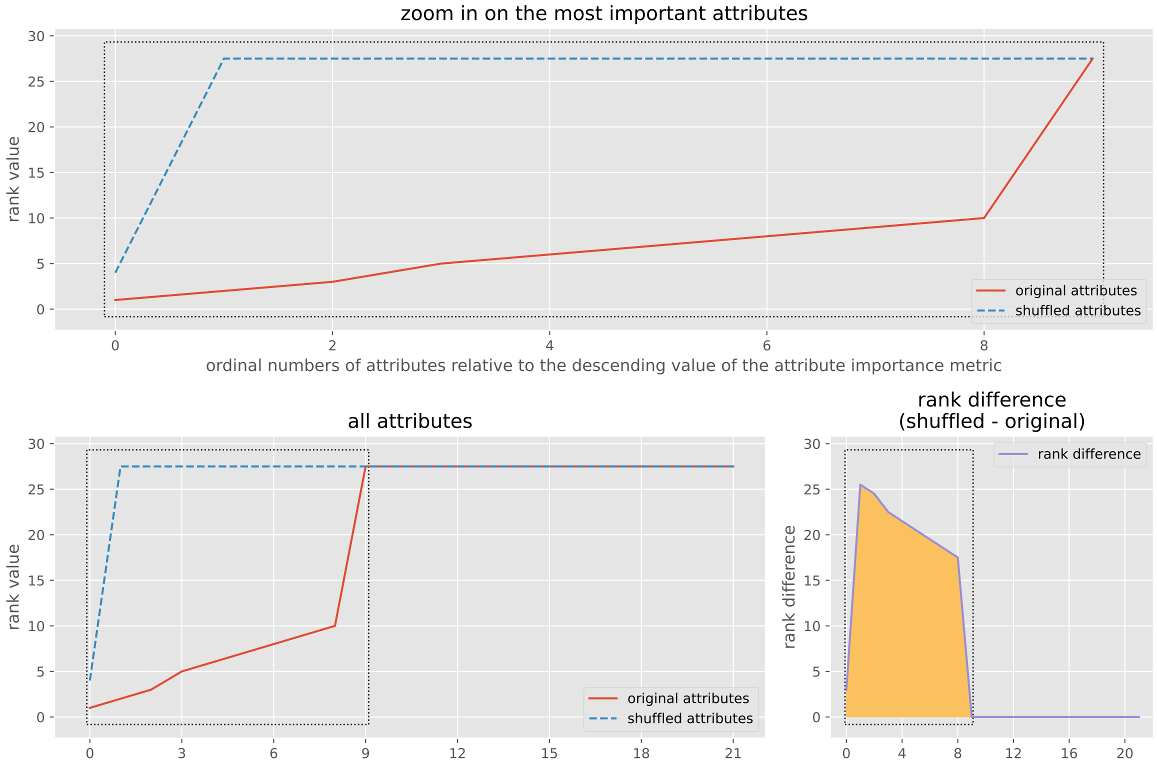


Figure A.11: Detailed attribute importance profiling results obtained for XGBoost on the *SPECT* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	23.67	F13	0.37	1.00
		5	3.00	24.80	F8	0.29	2.00
		7	4.00	25.86	F21	0.29	3.00
		10	5.50	27.40	F22	0.28	4.00
		15	8.00	29.93	F16	0.26	5.00
		20	10.50	32.45	F20	0.23	6.00
		all	11.55	33.45	F3	0.22	7.00
					F7	0.22	8.00
					F12	0.21	9.00
					F17	0.21	10.00
					F10	0.21	11.00
					F2	0.21	12.00
					F11	0.20	13.00
					F4	0.20	14.00
					F18	0.20	15.00
					F1	0.20	16.00
					F6	0.20	17.00
					F15	0.19	18.00
					F14	0.17	19.00
					F5	0.17	20.00
					...	...	...

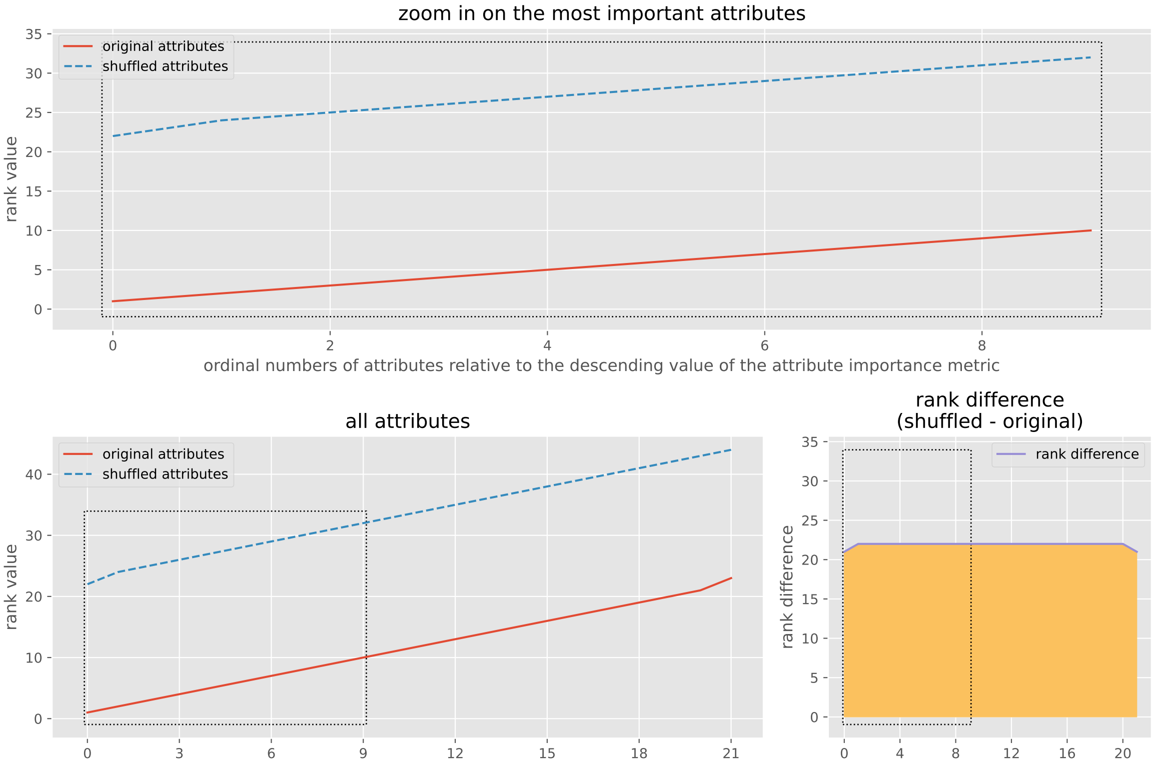


Figure A.12: Detailed attribute importance profiling results obtained for the correlation-based model on the *SPECT* data set.

A.4 Feature Importance Profiling - *synthetic1* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
10	5.50	28.90	5.50	1010.00	5.50	33.20
20	10.50	36.00	60.00	1010.00	10.50	43.45
30	18.53	44.20	376.67	1010.00	16.70	56.07
50	39.04	63.22	630.00	1010.00	32.42	77.26
100	88.58	112.80	820.00	1010.00	77.60	124.57
all	1013.02	987.98	991.00	1010.00	997.14	1003.86

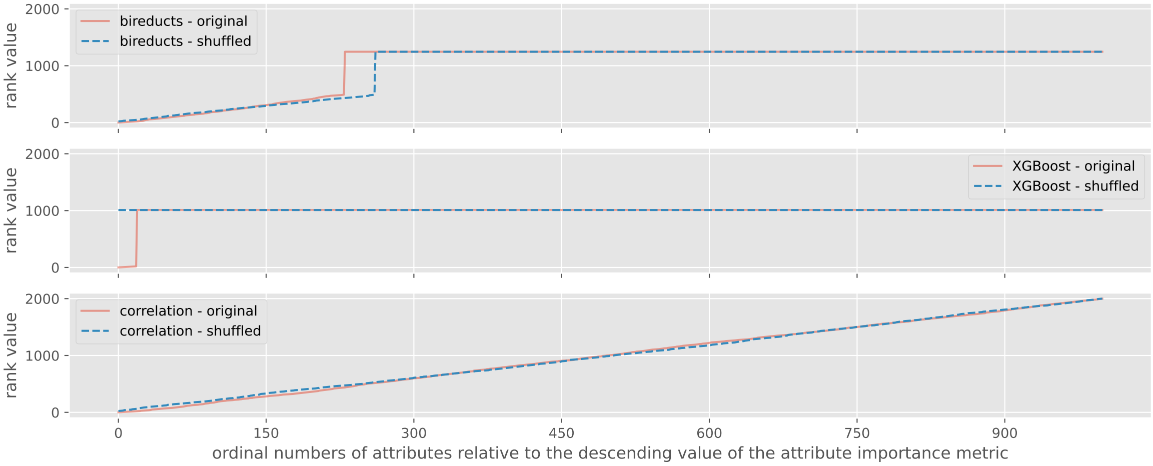


Figure A.13: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *synthetic1* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	count	global_gain	rank
algorithm	bireducts	3	2.00	22.33	a008	144	1.51	1.00
allowed_randomness	0.05	5	3.00	23.80	a012	124	1.38	2.00
attrs_max_count	3	7	4.00	26.29	a006	132	1.28	3.00
candidates_count	100	10	5.50	28.90	a016	127	1.27	4.00
chaos_fun	gini_impurity	15	8.00	32.60	a007	114	1.14	5.00
epsilon	0.00	20	10.50	36.00	a003	108	1.03	6.00
n_bins	3	25	13.84	39.72	a002	96	0.91	7.00
n_bireducts	1000	30	18.53	44.20	a010	95	0.89	8.00
probes_count	100	all	1013.02	987.98	a015	93	0.87	9.00
model stats					a014	91	0.84	10.00
mean_attrs_size	2.93				a001	85	0.77	11.00
mean_objs_size	5169.28				a013	85	0.74	12.00
median_attrs_size	3.00				a017	74	0.65	13.00
median_objs_size	5169.50				a005	79	0.65	14.00
					a011	78	0.64	15.00
					a000	71	0.59	16.00
					a018	68	0.57	17.00
					a004	66	0.53	18.00
					a019	58	0.44	19.00
					a009	58	0.41	20.00
					...	...	...	...

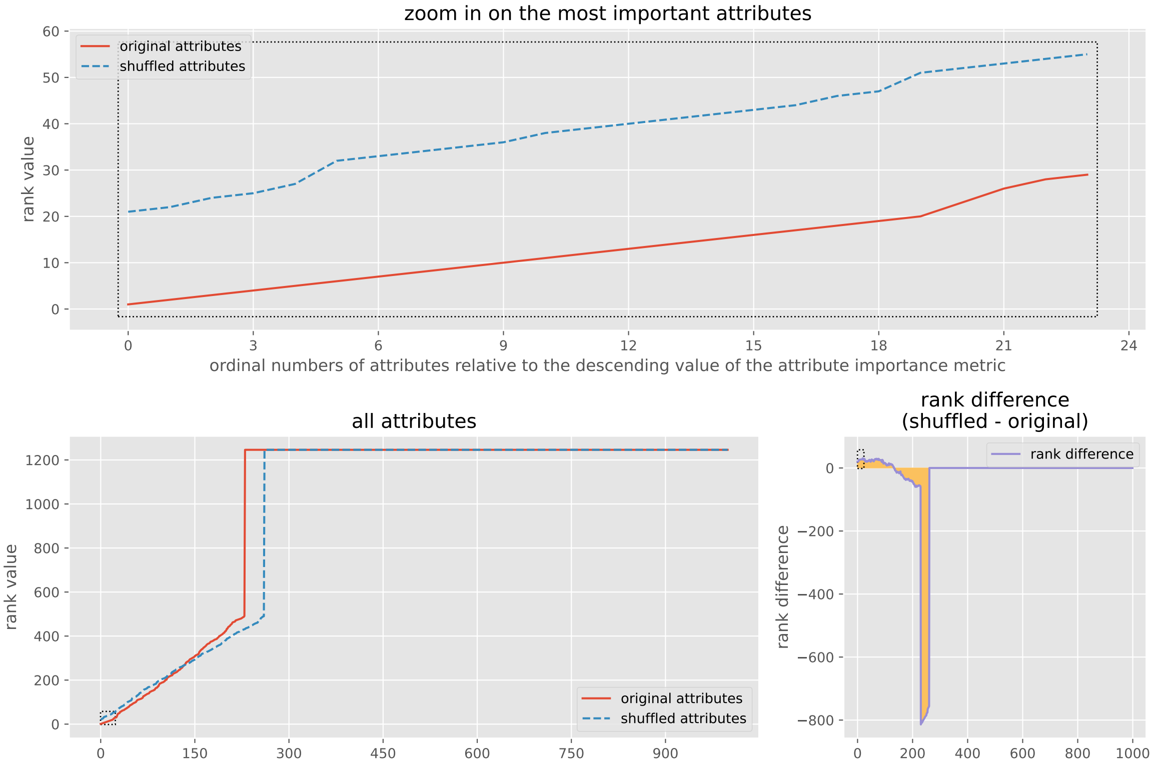


Figure A.14: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *synthetic1* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	1010.00	a012	290	34774.16	1.00
learning_rate	0.00	5	3.00	1010.00	a016	203	26231.05	2.00
max_depth	2	7	4.00	1010.00	a008	248	23562.57	3.00
num_boost_round	1000	10	5.50	1010.00	a003	234	22505.47	4.00
objective	binary:logistic	15	8.00	1010.00	a002	164	20598.32	5.00
		20	60.00	1010.00	a010	185	19412.69	6.00
		25	250.00	1010.00	a007	145	18339.68	7.00
		30	376.67	1010.00	a014	133	17750.26	8.00
		all	991.00	1010.00	a015	140	16021.95	9.00
					a006	170	15038.38	10.00
					a001	128	15037.23	11.00
					a013	107	14544.72	12.00
					a017	173	14202.34	13.00
					a004	145	12793.78	14.00
					a019	152	11914.64	15.00
					a018	143	11870.74	16.00
					a005	135	11818.54	17.00
					a000	64	5619.57	18.00
					a011	41	4274.31	19.00
					shuffled_a334	0	0.00	1010.00
					...	...	...	...

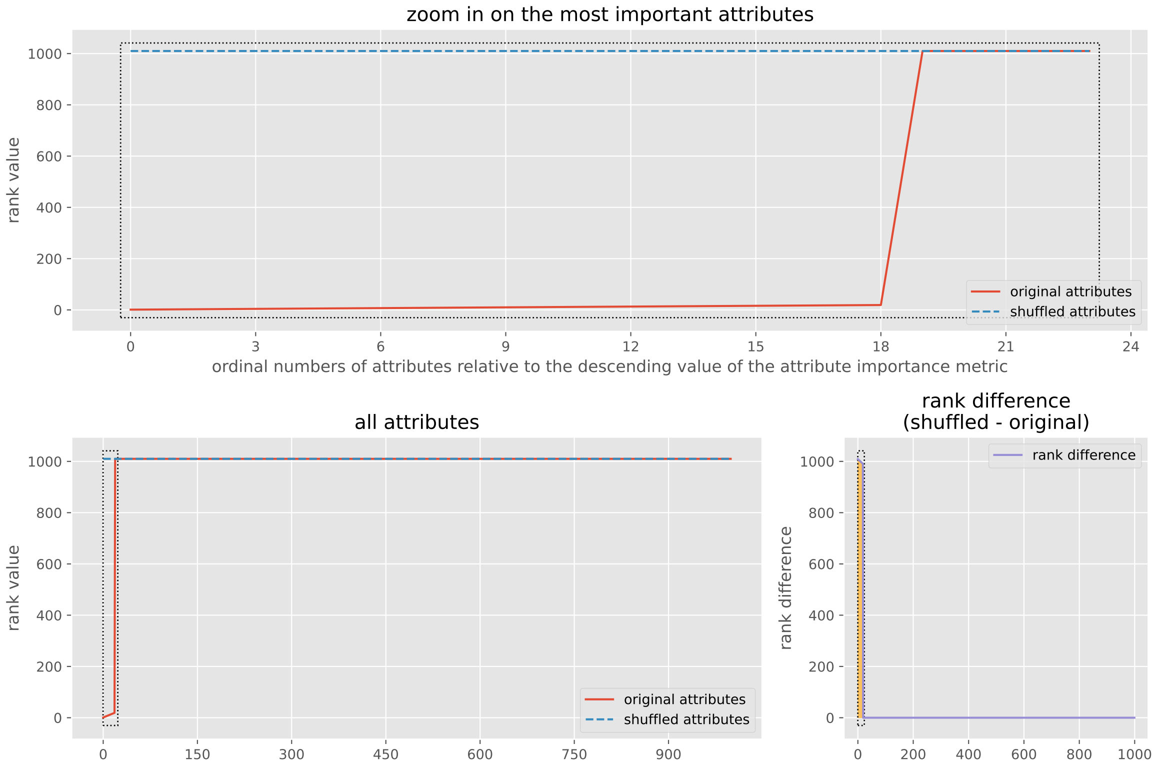


Figure A.15: Detailed attribute importance profiling results obtained for XGBoost on the *synthetic1* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top__k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	25.00	a012	0.15	1.00
		5	3.00	26.40	a008	0.15	2.00
		7	4.00	29.14	a016	0.15	3.00
		10	5.50	33.20	a002	0.15	4.00
		15	8.00	38.13	a003	0.15	5.00
		20	10.50	43.45	a010	0.14	6.00
		25	13.32	49.76	a007	0.14	7.00
		30	16.70	56.07	a014	0.14	8.00
		all	997.14	1003.86	a006	0.14	9.00
					a015	0.14	10.00
					a013	0.14	11.00
					a001	0.14	12.00
					a017	0.14	13.00
					a000	0.13	14.00
					a018	0.13	15.00
					a005	0.13	16.00
					a011	0.13	17.00
					a004	0.13	18.00
					a019	0.13	19.00
					a009	0.12	20.00
					...	...	...

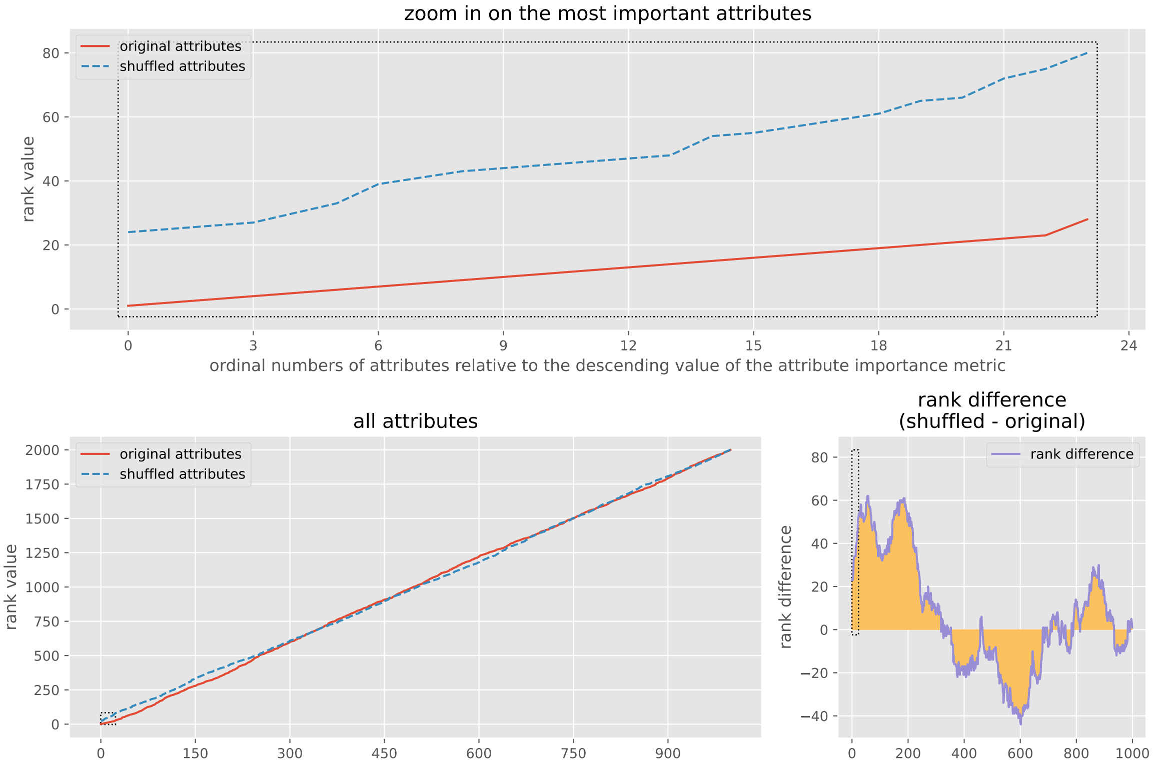


Figure A.16: Detailed attribute importance profiling results obtained for the correlation-based model on the *synthetic1* data set.

A.5 Feature Importance Profiling - *synthetic2* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
10	17.60	6.30	6.80	19.00	13.20	8.20
20	26.35	15.00	13.80	28.20	25.85	17.20
30	35.13	25.93	24.20	38.10	37.73	25.27
50	51.66	49.98	138.63	266.37	57.92	43.82
100	95.64	106.16	590.57	654.43	108.86	92.17
all	992.44	1008.56	997.31	1003.69	999.06	1001.94

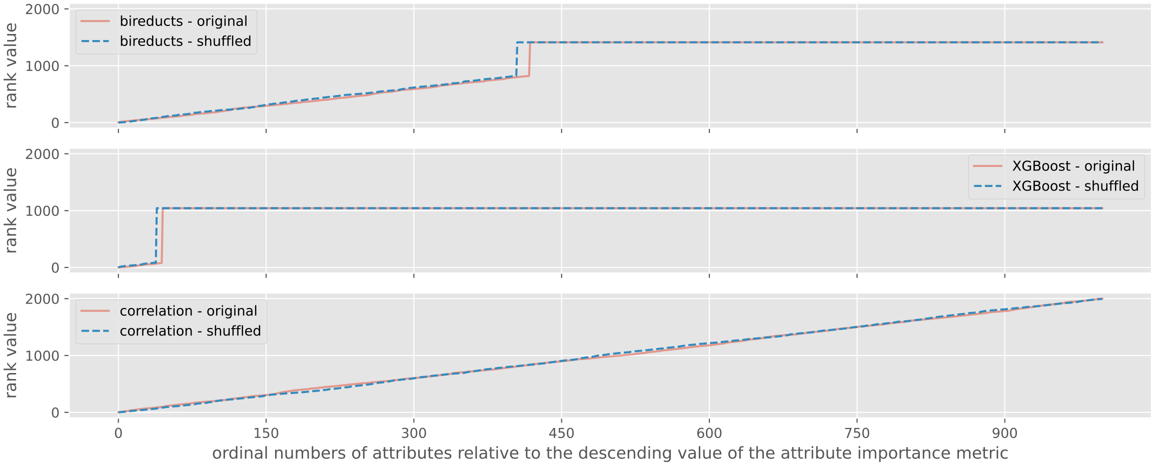


Figure A.17: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *synthetic2* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	count	global_gain	rank
algorithm	bireducts	3	9.33	2.00	shuffled_a906	95	0.23	1.00
allowed_randomness	0.05	5	11.80	3.20	shuffled_a786	86	0.21	2.00
attrs_max_count	3	7	14.57	4.43	shuffled_a382	68	0.16	3.00
candidates_count	100	10	17.60	6.30	shuffled_a096	65	0.15	4.00
chaos_fun	gini_impurity	15	22.00	10.07	a390	56	0.13	5.00
epsilon	0.00	20	26.35	15.00	shuffled_a274	56	0.12	6.00
n_bins	3	25	30.84	20.32	shuffled_a391	48	0.12	7.00
n_bireducts	1000	30	35.13	25.93	shuffled_a450	50	0.11	8.00
probes_count	100	all	992.44	1008.56	shuffled_a642	44	0.10	9.00
model stats					a351	40	0.09	10.00
mean_attrs_size	2.87				shuffled_a547	37	0.08	11.00
mean_objs_size	5035.03				shuffled_a721	34	0.08	12.00
median_attrs_size	3.00				a854	33	0.08	13.00
median_objs_size	5039.50				shuffled_a578	34	0.07	14.00
					a979	34	0.07	15.00
					a841	33	0.07	16.00
					shuffled_a573	32	0.07	17.00
					shuffled_a892	31	0.07	18.00
					shuffled_a249	30	0.07	19.00
					shuffled_a406	27	0.06	20.00
					...	...	...	...

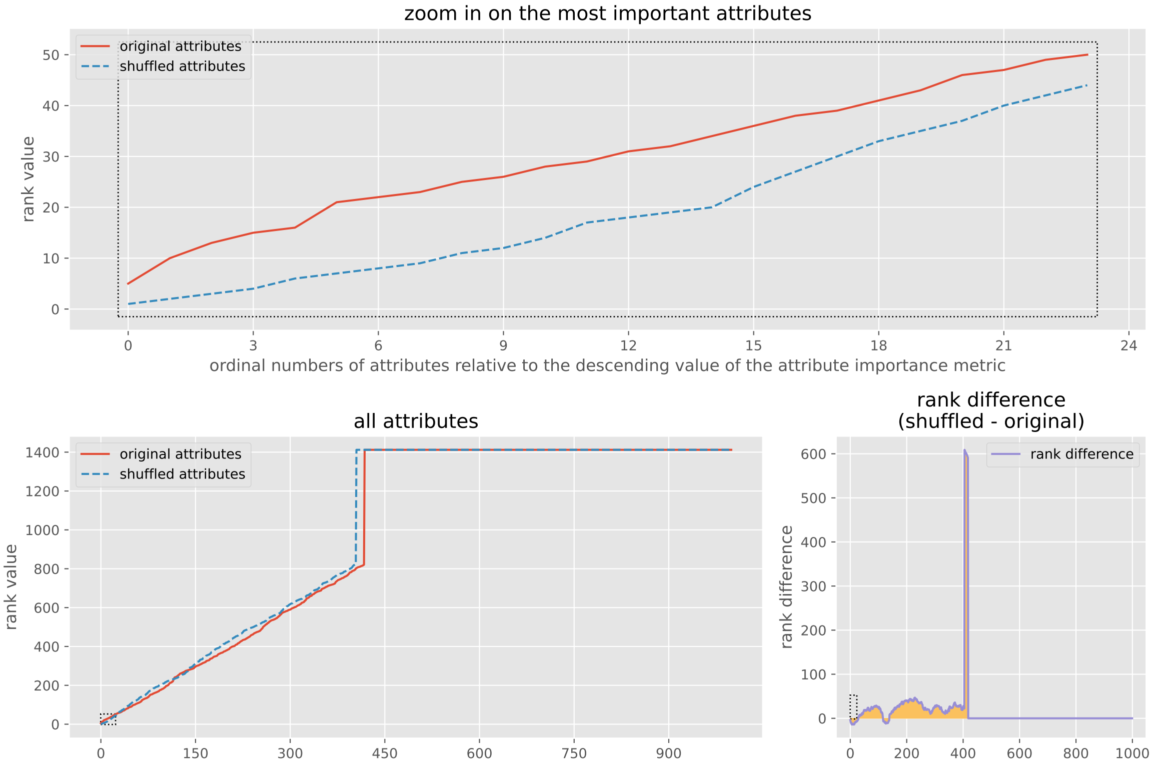


Figure A.18: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *synthetic2* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.67	8.00	a009	131	3759.01	1.00
learning_rate	0.00	5	3.80	12.20	shuffled_a090	165	2724.36	2.00
max_depth	2	7	5.00	15.14	a573	149	2328.73	3.00
num_boost_round	1000	10	6.80	19.00	a265	120	2026.15	4.00
objective	binary:logistic	15	10.00	23.87	a047	127	1998.87	5.00
		20	13.80	28.20	a854	101	1881.74	6.00
		25	18.84	32.56	a658	115	1746.61	7.00
		30	24.20	38.10	shuffled_a701	92	1683.46	8.00
		all	997.31	1003.69	a352	85	1570.33	9.00
					a128	80	1547.82	10.00
					a291	92	1441.25	11.00
					a011	32	1423.91	12.00
					a157	80	1367.60	13.00
					shuffled_a700	80	1214.07	14.00
					a958	67	1135.83	15.00
					a010	74	1073.52	16.00
					a692	70	1004.06	17.00
					shuffled_a494	48	979.18	18.00
					shuffled_a919	65	952.84	19.00
					shuffled_a530	64	924.55	20.00
					...	...	...	...

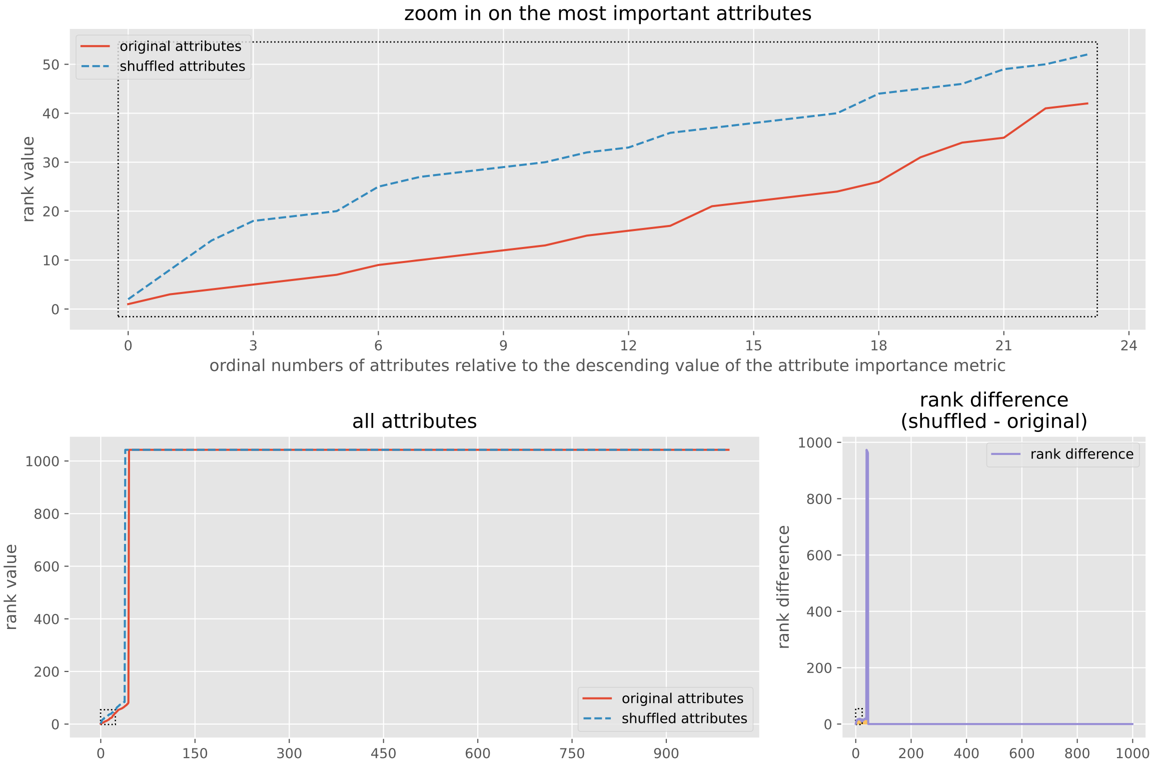
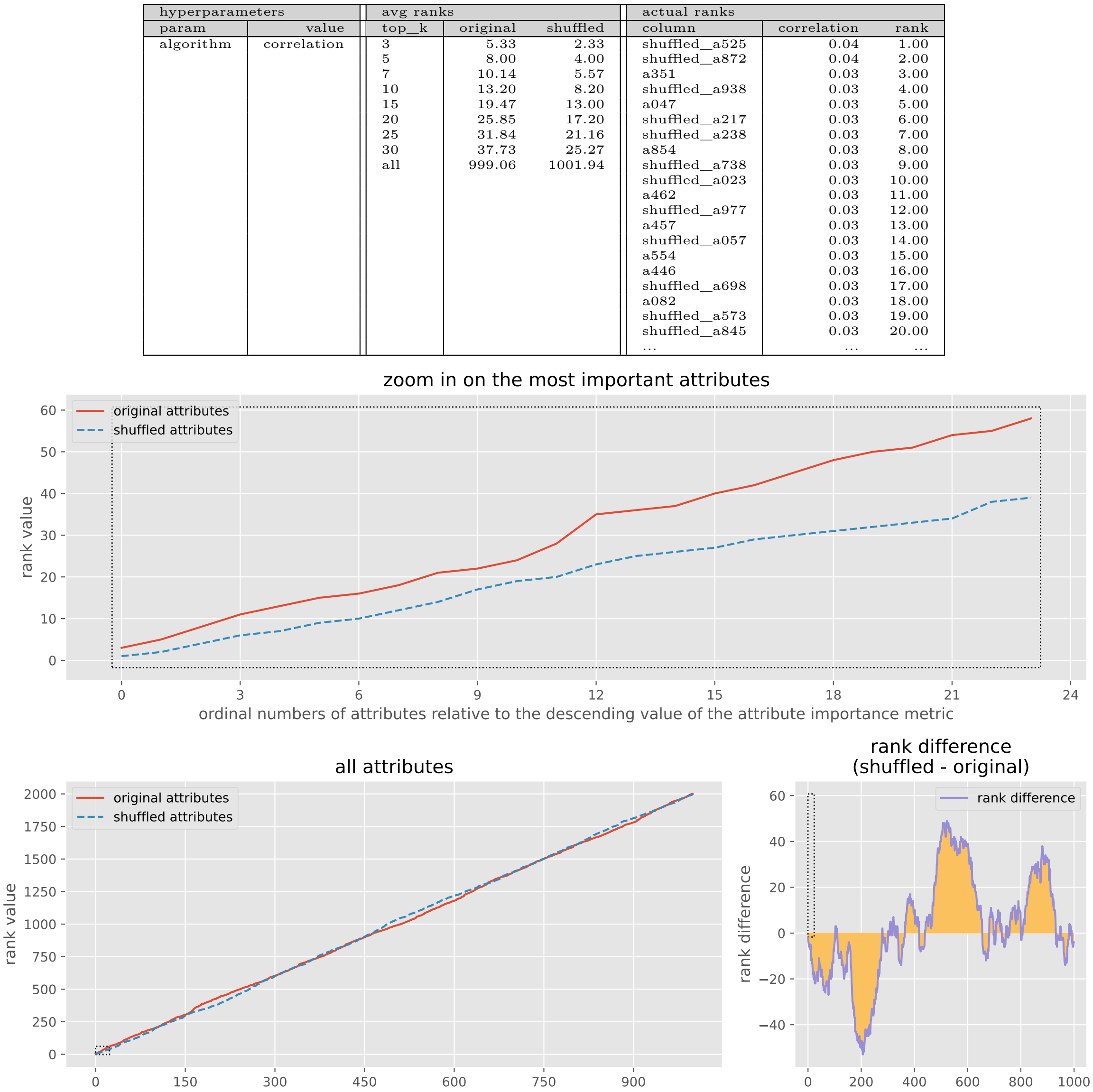


Figure A.19: Detailed attribute importance profiling results obtained for XGBoost on the *synthetic2* data set.

Correlation - detailed profile



hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	5.33	2.33	shuffled_a525	0.04	1.00
		5	8.00	4.00	shuffled_a872	0.04	2.00
		7	10.14	5.57	a351	0.03	3.00
		10	13.20	8.20	shuffled_a938	0.03	4.00
		15	19.47	13.00	a047	0.03	5.00
		20	25.85	17.20	shuffled_a217	0.03	6.00
		25	31.84	21.16	shuffled_a238	0.03	7.00
		30	37.73	25.27	a854	0.03	8.00
		all	999.06	1001.94	shuffled_a738	0.03	9.00
					shuffled_a023	0.03	10.00
					a462	0.03	11.00
					shuffled_a977	0.03	12.00
					a457	0.03	13.00
					shuffled_a057	0.03	14.00
					a554	0.03	15.00
					a446	0.03	16.00
					shuffled_a698	0.03	17.00
					a082	0.03	18.00
					shuffled_a573	0.03	19.00
					shuffled_a845	0.03	20.00
					...	...	...

Figure A.20: Detailed attribute importance profiling results obtained for the correlation-based model on the *synthetic2* data set.

A.6 Feature Importance Profiling - *synthetic3* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
10	5.50	30.20	6.60	19.20	6.60	17.20
20	10.75	41.25	13.75	30.45	15.75	25.30
30	17.27	50.40	22.37	203.03	29.60	32.70
50	34.94	67.62	154.18	535.82	53.12	48.30
100	83.98	120.11	594.59	785.41	112.09	92.30
all	979.75	1021.25	990.96	1010.04	1000.14	1000.86

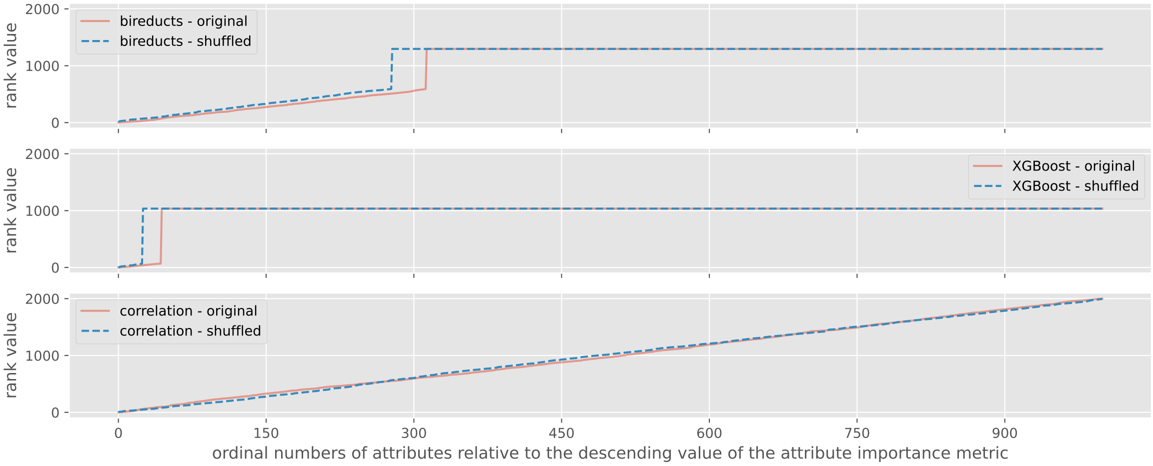


Figure A.21: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *synthetic3* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top__k	original	shuffled	column	count	global__gain	rank
algorithm	bireducts	3	2.00	21.33	a017	146	0.44	1.00
allowed__randomness	0.05	5	3.00	24.40	a009	128	0.39	2.00
attrs__max__count	3	7	4.00	27.29	a011	129	0.37	3.00
candidates__count	100	10	5.50	30.20	a001	110	0.29	4.00
chaos__fun	gini__impurity	15	8.00	36.13	a005	102	0.26	5.00
epsilon	0.00	20	10.75	41.25	a019	103	0.24	6.00
n__bins	3	25	13.72	45.96	a004	101	0.24	7.00
n__bireducts	1000	30	17.27	50.40	a003	91	0.21	8.00
probes__count	100	all	979.75	1021.25	a007	86	0.21	9.00
model stats					a010	84	0.20	10.00
mean__attrs__size	2.93				a015	74	0.17	11.00
mean__objs__size	5047.51				a000	75	0.17	12.00
median__attrs__size	3.00				a014	63	0.14	13.00
median__objs__size	5051.00				a018	57	0.12	14.00
					a013	53	0.11	15.00
					shuffled__a206	44	0.09	16.00
					a887	43	0.09	17.00
					a016	43	0.08	18.00
					a012	37	0.08	19.00
					a002	38	0.08	20.00
					...	...	...	...

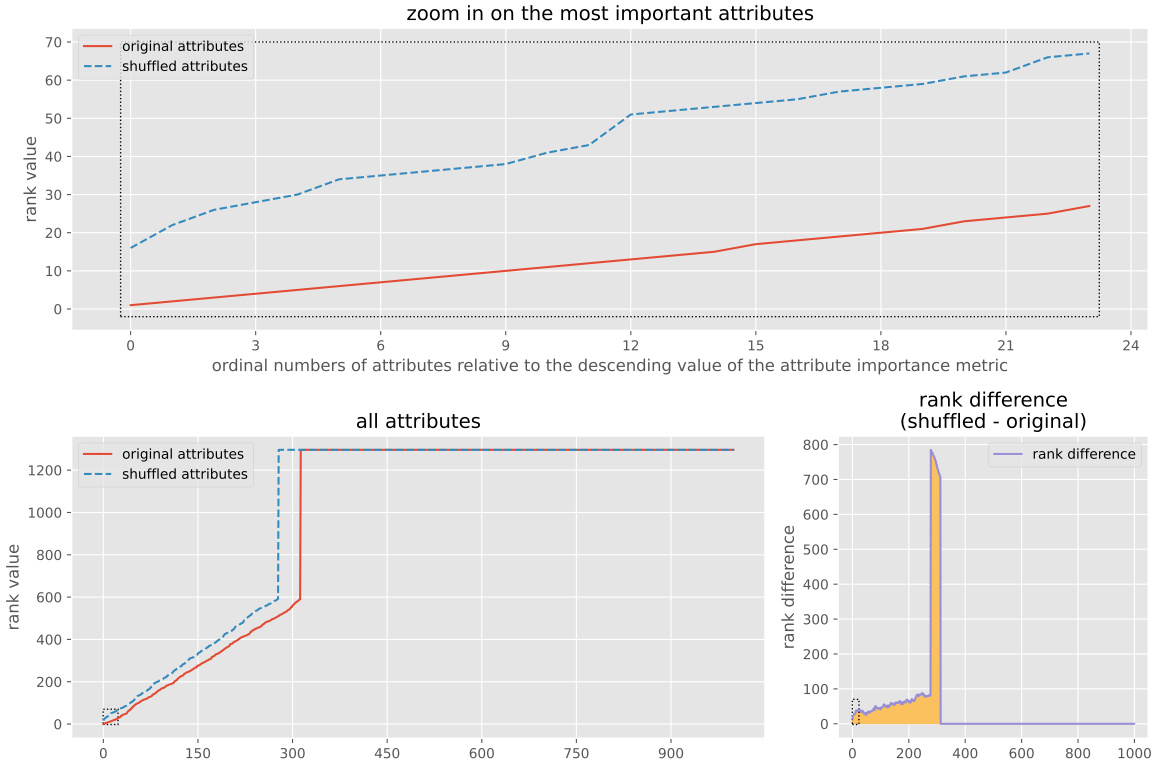


Figure A.22: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *synthetic3* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.33	9.33	a011	377	38736.11	1.00
learning_rate	0.00	5	3.60	13.40	a017	207	5942.99	2.00
max_depth	2	7	4.71	15.71	shuffled_a712	174	4327.80	3.00
num_boost_round	1000	10	6.60	19.20	a015	68	3031.16	4.00
objective	binary:logistic	15	9.53	24.67	a127	159	2988.02	5.00
		20	13.75	30.45	a009	149	2517.66	6.00
		25	18.04	36.64	a012	44	2463.10	7.00
		30	22.37	203.03	a010	30	1809.74	8.00
		all	990.96	1010.04	shuffled_a395	99	1757.76	9.00
					a185	97	1756.07	10.00
					a605	55	1537.56	11.00
					a862	84	1469.75	12.00
					a928	81	1357.76	13.00
					a424	67	1279.41	14.00
					a002	72	1245.24	15.00
					shuffled_a996	58	1152.80	16.00
					a819	67	1094.58	17.00
					a466	67	1091.70	18.00
					shuffled_a371	82	906.33	19.00
					shuffled_a250	50	903.83	20.00
					...	...	...	...

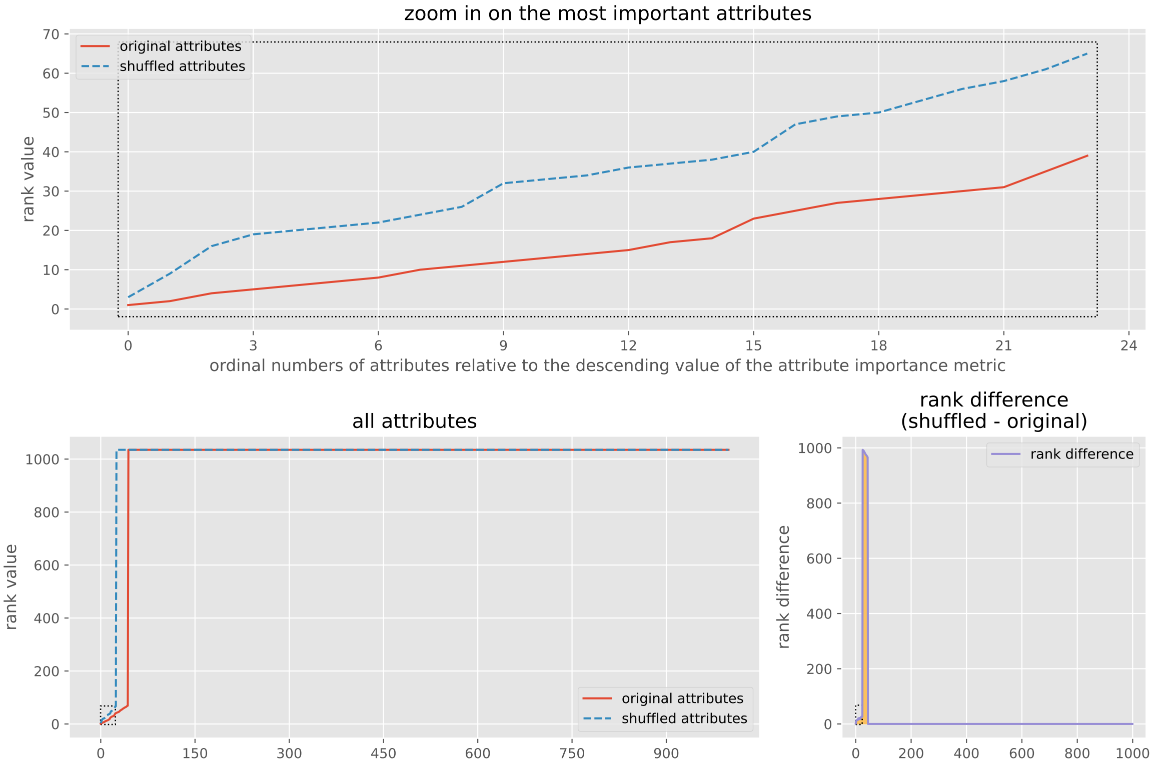


Figure A.23: Detailed attribute importance profiling results obtained for XGBoost on the *synthetic3* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	9.00	a862	0.03	1.00
		5	3.20	10.80	a725	0.03	2.00
		7	4.43	13.86	a928	0.03	3.00
		10	6.60	17.20	a300	0.03	4.00
		15	10.40	21.53	shuffled_a061	0.03	5.00
		20	15.75	25.30	a600	0.03	6.00
		25	22.40	29.04	a127	0.03	7.00
		30	29.60	32.70	a848	0.03	8.00
		all	1000.14	1000.86	a819	0.03	9.00
					shuffled_a087	0.03	10.00
					a983	0.03	11.00
					shuffled_a487	0.03	12.00
					shuffled_a326	0.03	13.00
					shuffled_a345	0.03	14.00
					a608	0.03	15.00
					a407	0.03	16.00
					a033	0.03	17.00
					a167	0.03	18.00
					a871	0.03	19.00
					a032	0.03	20.00
					...	...	...

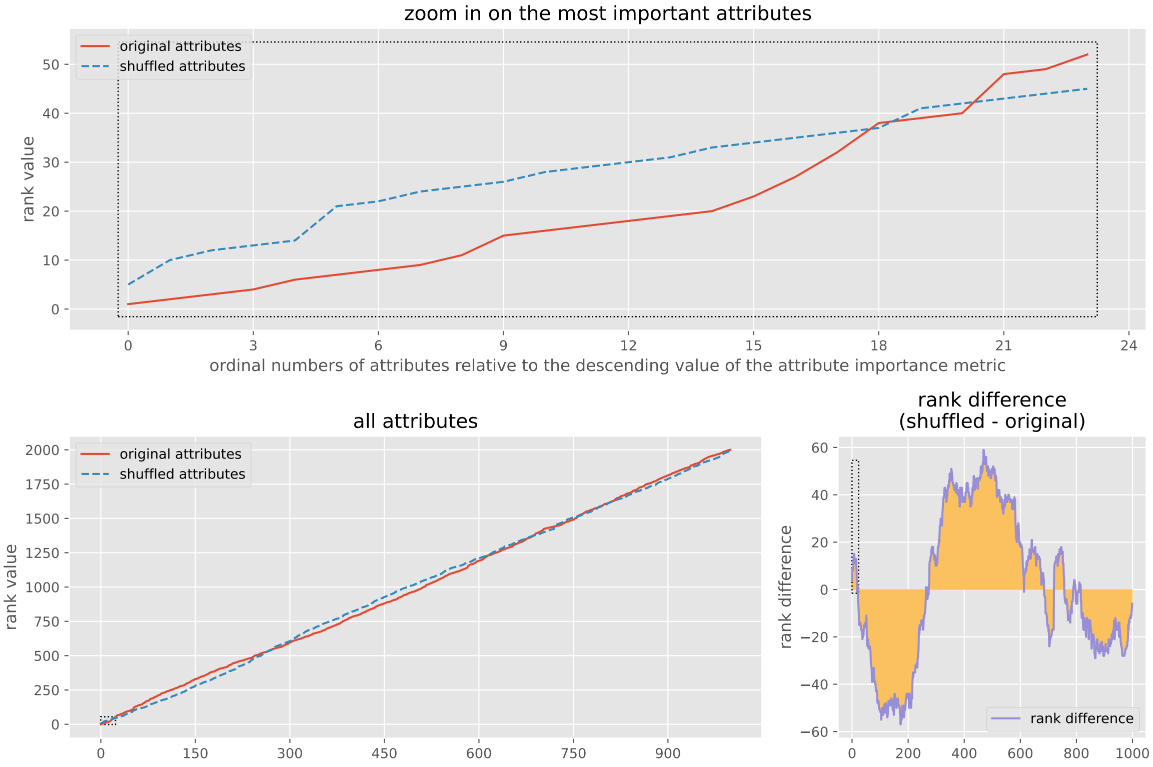


Figure A.24: Detailed attribute importance profiling results obtained for the correlation-based model on the *synthetic3* data set.

A.7 Feature Importance Profiling - *acuteLymphoblasticLeukemia* data set

	avg rank value of the top__k attributes					
	bireducts		XGBoost		correlation	
top__k	original	shuffled	original	shuffled	original	shuffled
10	5.50	7675.10	5.50	39.40	5.50	3000.30
20	10.50	15307.55	10.75	59.80	10.50	3302.80
30	15.50	17851.70	16.10	75.80	15.50	3498.77
50	25.50	19887.02	28.80	7208.24	25.50	3802.06
100	50.50	21413.51	62.71	14780.12	50.50	4295.56
all	21621.85	22933.15	22236.99	22318.01	17871.52	26683.48

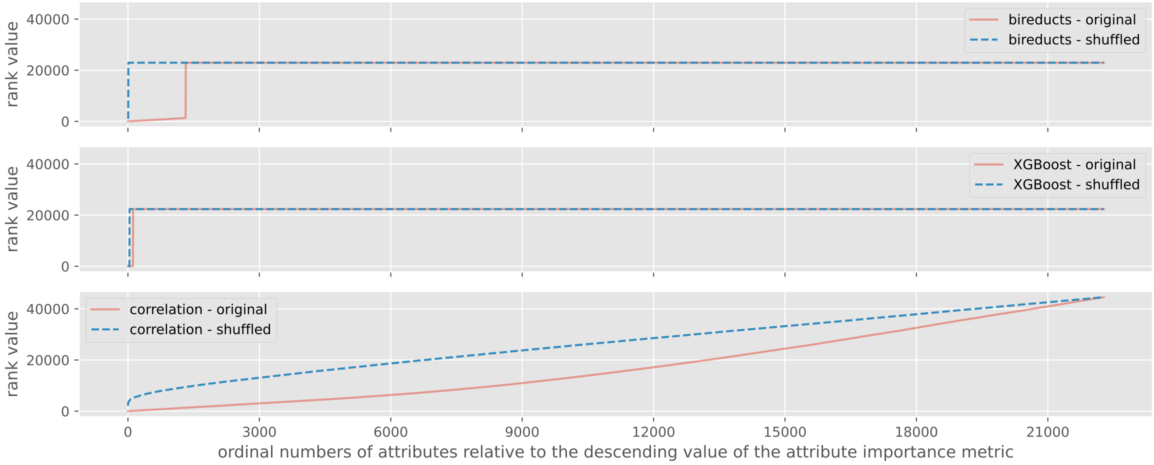


Figure A.25: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *acuteLymphoblasticLeukemia* data set.

Bireducts - detailed profile

hyperparameters	
param	value
algorithm	bireducts
allowed_randomness	0.05
attrs_max_count	3
candidates_count	100
chaos_fun	gini_impurity
epsilon	0.00
n_bins	3
n_bireducts	1000
probes_count	100
model stats	
mean_attrs_size	3.00
mean_objs_size	107.95
median_attrs_size	3.00
median_objs_size	108.00

avg ranks		
top_k	original	shuffled
3	2.00	1043.00
5	3.00	1082.40
7	4.00	1133.00
10	5.50	7675.10
15	8.00	12763.40
20	10.50	15307.55
25	13.00	16834.04
30	15.50	17851.70
all	21621.85	22933.15

actual ranks			
column	count	global_gain	rank
X221748_s_at	13	2.44	1.00
X218418_s_at	14	2.41	2.00
X218625_at	11	2.27	3.00
X221773_at	11	2.08	4.00
X204115_at	10	1.84	5.00
X209184_s_at	11	1.74	6.00
X212154_at	11	1.71	7.00
X221349_at	8	1.66	8.00
X219471_at	9	1.65	9.00
X212956_at	10	1.65	10.00
X221246_x_at	8	1.57	11.00
X219686_at	7	1.53	12.00
X202517_at	9	1.53	13.00
X38269_at	9	1.53	14.00
X213394_at	9	1.51	15.00
X204914_s_at	9	1.51	16.00
X219866_at	8	1.50	17.00
X202853_s_at	8	1.46	18.00
X215001_s_at	9	1.45	19.00
X209604_s_at	8	1.45	20.00
...	...	...	...

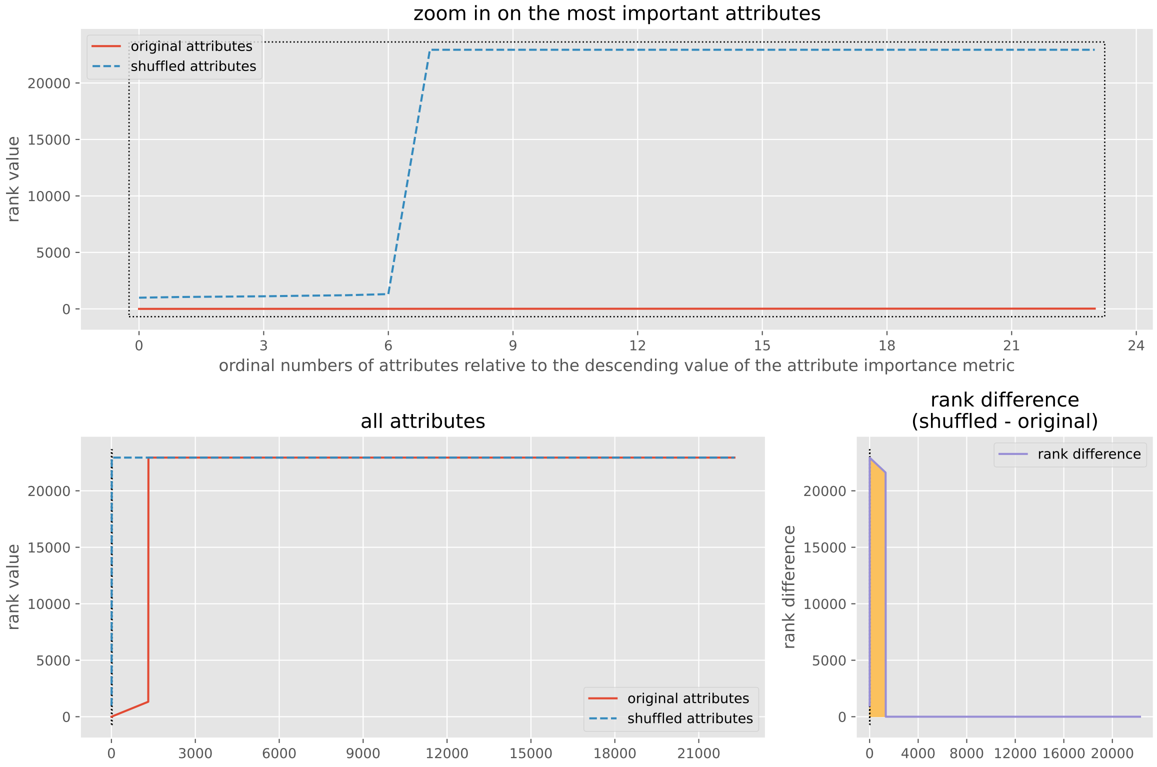


Figure A.26: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *acuteLymphoblasticLeukemia* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	26.00	X206231_at	1000	42521.44	1.00
learning_rate	0.00	5	3.00	30.20	X205456_at	1000	41606.96	2.00
max_depth	2	7	4.00	33.86	X201811_x_at	490	16045.94	3.00
num_boost_round	1000	10	5.50	39.40	X221932_s_at	1000	14230.27	4.00
objective	multi:softmax	15	8.00	49.93	X201136_at	391	11138.38	5.00
		20	10.75	59.80	X213394_at	396	9426.82	6.00
		25	13.40	68.64	X202711_at	440	5459.08	7.00
		30	16.10	75.80	X216620_s_at	390	3656.45	8.00
		all	22236.99	22318.01	X202853_s_at	196	3519.00	9.00
					X35974_at	821	3295.83	10.00
					X201612_at	187	2982.84	11.00
					X204793_at	262	2709.81	12.00
					X201121_s_at	267	2571.88	13.00
					X201443_s_at	119	2230.20	14.00
					X217652_at	490	2159.33	15.00
					shuffled_X220232_at	177	1751.82	16.00
					X203372_s_at	141	1526.27	17.00
					X203744_at	137	1476.40	18.00
					X218668_s_at	153	1298.15	19.00
					X215146_s_at	281	1025.35	20.00
					...	...	...	...

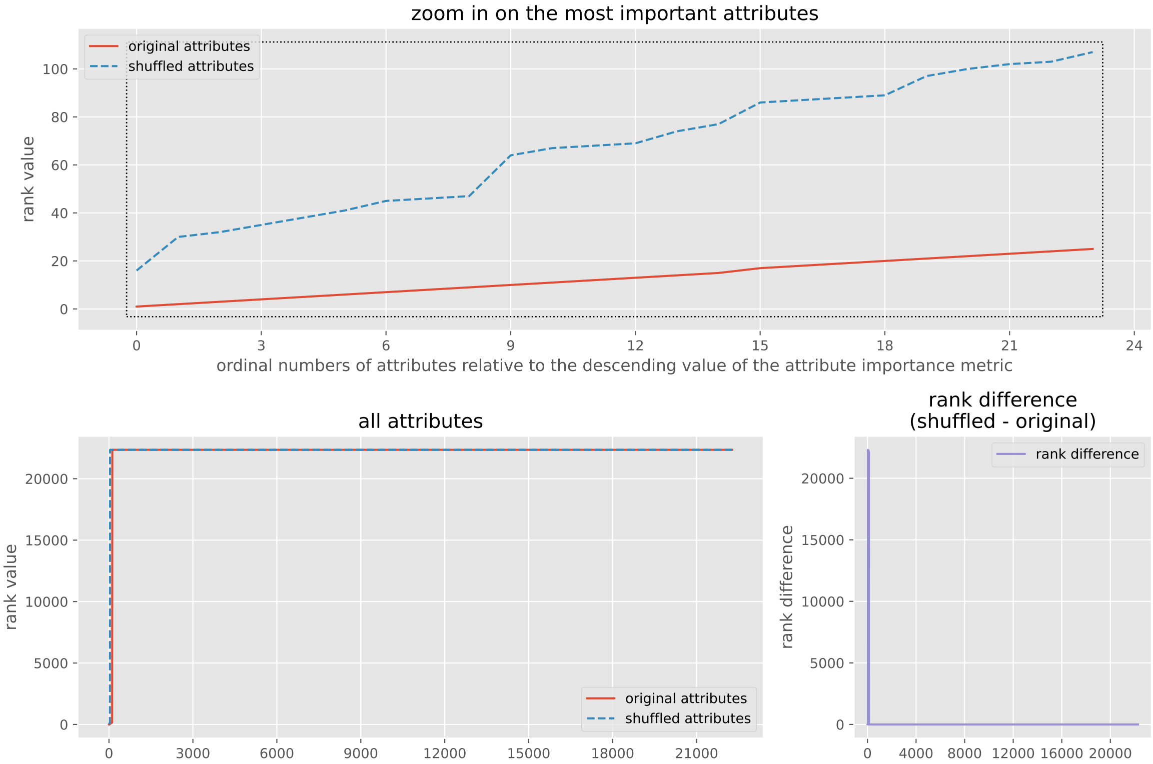


Figure A.27: Detailed attribute importance profiling results obtained for XGBoost on the *acuteLymphoblasticLeukemia* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	2579.00	X221840_at	0.79	1.00
		5	3.00	2751.00	X206001_at	0.70	2.00
		7	4.00	2857.71	X213891_s_at	0.69	3.00
		10	5.50	3000.30	X218418_s_at	0.68	4.00
		15	8.00	3177.40	X222146_s_at	0.68	5.00
		20	10.50	3302.80	X212386_at	0.67	6.00
		25	13.00	3409.48	X212387_at	0.67	7.00
		30	15.50	3498.77	X209604_s_at	0.67	8.00
		all	17871.52	26683.48	X209568_s_at	0.67	9.00
					X203753_at	0.66	10.00
					X212385_at	0.65	11.00
					X219157_at	0.65	12.00
					X217436_x_at	0.65	13.00
					X212382_at	0.64	14.00
					X201720_s_at	0.63	15.00
					X50221_at	0.63	16.00
					X221773_at	0.63	17.00
					X201015_s_at	0.63	18.00
					X204806_x_at	0.63	19.00
					X205067_at	0.63	20.00
					...	...	...

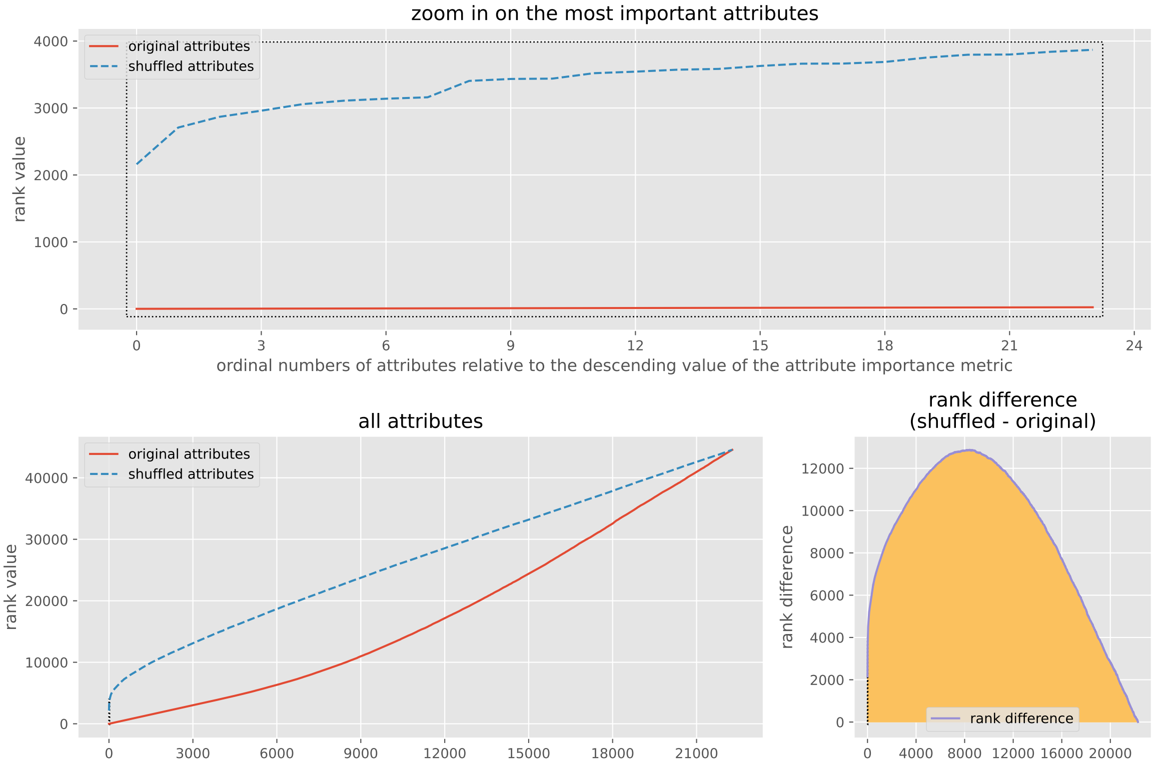


Figure A.28: Detailed attribute importance profiling results obtained for the correlation-based model on the *acuteLymphoblasticLeukemia* data set.

A.8 Feature Importance Profiling - *hepatitisC* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
10	5.50	1512.40	5.50	8950.00	5.50	5551.50
20	10.50	11249.88	11.00	15637.25	10.50	5931.50
30	15.50	15211.75	16.63	17866.33	15.50	6291.08
50	25.50	18381.25	27.88	19649.60	25.50	6861.59
100	50.50	20758.38	2721.57	20987.05	50.50	7693.22
all	21430.17	23124.83	22236.50	22318.50	15062.49	29492.51

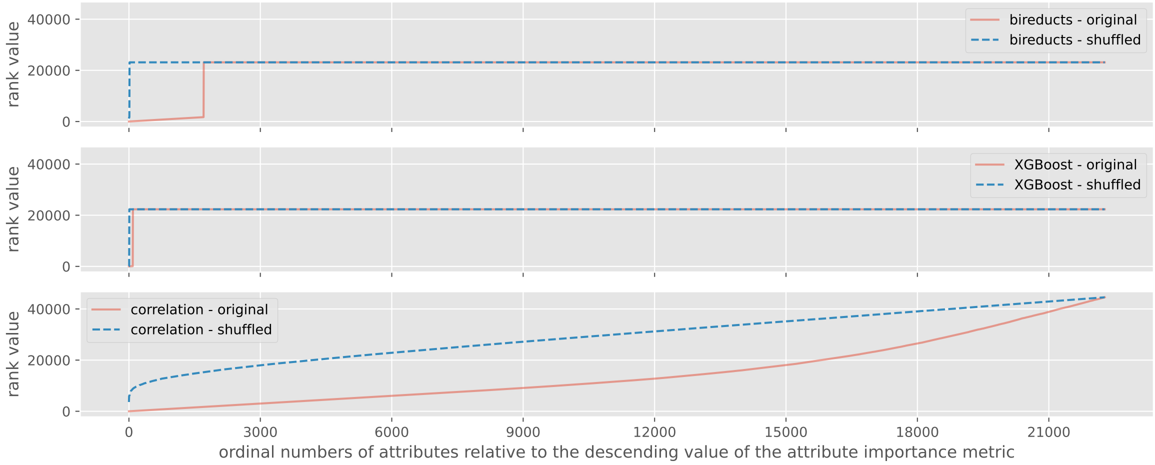


Figure A.29: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *hepatitisC* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	count	global_gain	rank
algorithm	bireducts	3	2.00	1333.33	X65521_at	9	1.82	1.00
allowed_randomness	0.05	5	3.00	1421.40	X61874_at	6	1.82	2.00
attrs_max_count	3	7	4.00	1467.43	X217917_s_at	9	1.80	3.00
candidates_count	100	10	5.50	1512.40	X214313_s_at	9	1.75	4.00
chaos_fun	gini_impurity	15	8.00	7288.00	X202286_s_at	9	1.75	5.00
epsilon	0.00	20	10.50	11249.88	X221817_at	8	1.69	6.00
n_bins	3	25	13.00	13627.00	X214842_s_at	8	1.60	7.00
n_bireducts	1000	30	15.50	15211.75	X213813_x_at	7	1.60	8.00
probes_count	100	all	21430.17	23124.83	X207057_at	7	1.58	9.00
model stats					X51774_s_at	7	1.57	10.00
mean_attrs_size	3.00				X58900_at	8	1.56	11.00
mean_objs_size	91.12				X204061_at	8	1.54	12.00
median_attrs_size	3.00				X210596_at	8	1.51	13.00
median_objs_size	92.00				X216609_at	6	1.51	14.00
					X208915_s_at	7	1.44	15.00
					X213642_at	7	1.43	16.00
					X214413_at	7	1.39	17.00
					X49327_at	6	1.39	18.00
					X214057_at	8	1.36	19.00
					X213484_at	7	1.35	20.00
					...	...	...	...

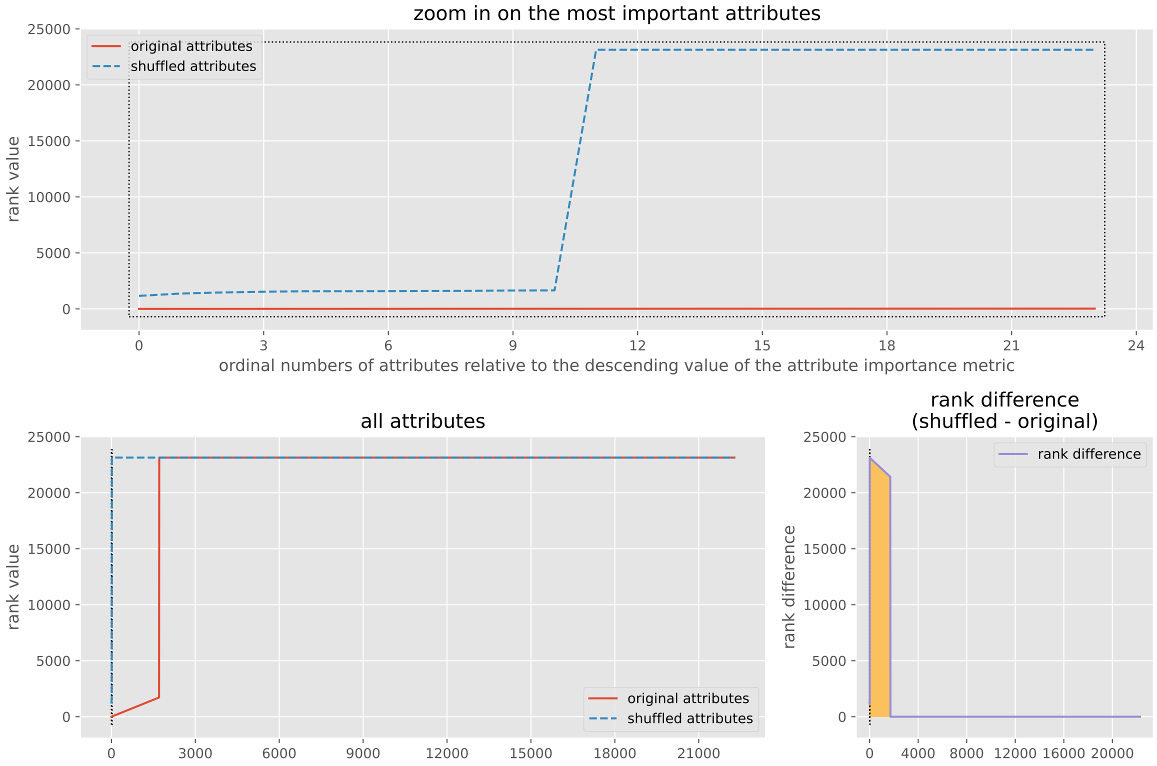


Figure A.30: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *hepatitisC* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	20.67	X208306_x_at	1000	21218.46	1.00
learning_rate	0.00	5	3.00	29.60	X51774_s_at	774	20926.04	2.00
max_depth	2	7	4.00	3218.07	X216563_at	712	18114.08	3.00
num_boost_round	1000	10	5.50	8950.00	X213490_s_at	735	8040.81	4.00
objective	multi:softmax	15	8.07	13408.17	X212649_at	288	8004.32	5.00
		20	11.00	15637.25	X204454_at	989	5431.93	6.00
		25	13.80	16974.70	X48030_i_at	226	3035.64	7.00
		30	16.63	17866.33	X211109_at	495	2525.01	8.00
		all	22236.50	22318.50	X1405_i_at	735	2405.13	9.00
					X217912_at	238	1438.01	10.00
					X205519_at	288	1406.15	11.00
					X204301_at	324	1364.16	12.00
					X211122_s_at	299	1194.54	13.00
					X33646_g_at	288	1101.88	14.00
					shuffled_X222255_at	238	745.86	15.00
					X208984_x_at	226	532.62	16.00
					X219937_at	121	487.52	17.00
					shuffled_X208249_s_at	119	480.17	18.00
					X211947_s_at	126	395.15	19.00
					X217737_x_at	112	348.77	20.00
					...	...	...	...

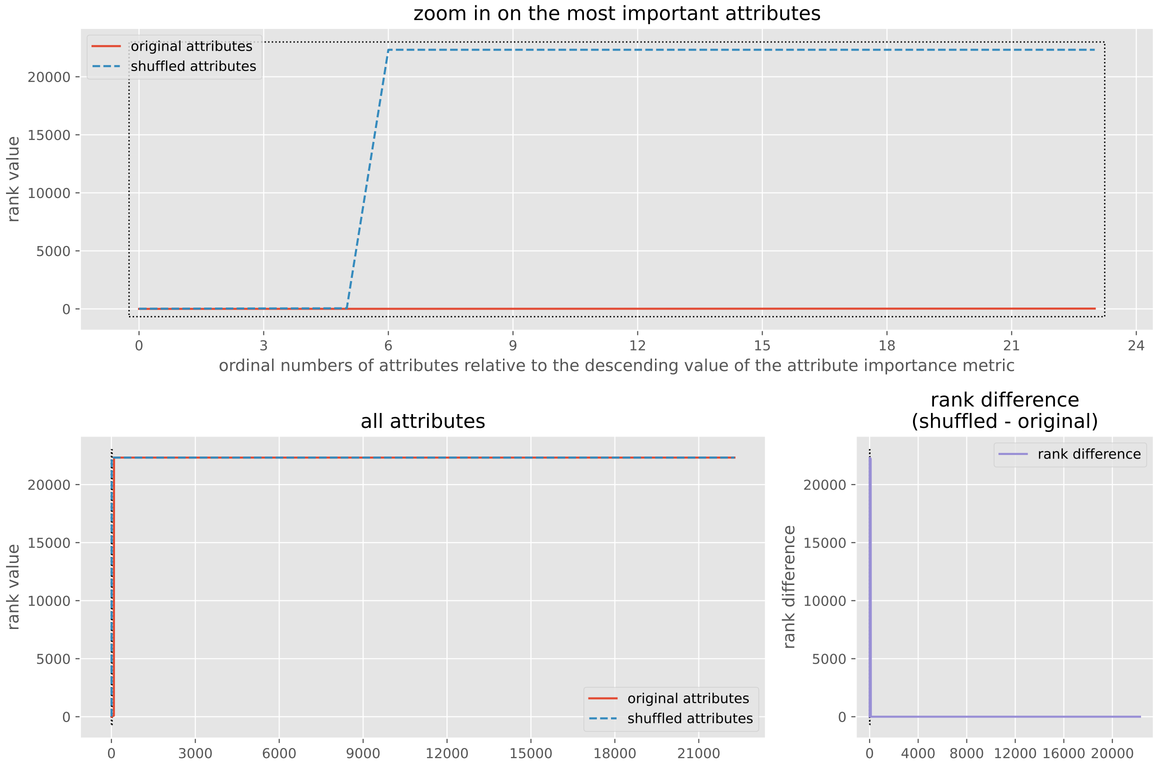


Figure A.31: Detailed attribute importance profiling results obtained for XGBoost on the *hepatitisC* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	4673.00	X216563_at	0.77	1.00
		5	3.00	5065.00	X214305_s_at	0.75	2.00
		7	4.00	5316.00	X201846_s_at	0.74	3.00
		10	5.50	5551.50	X214056_at	0.74	4.00
		15	8.00	5755.57	X201070_x_at	0.74	5.00
		20	10.50	5931.50	X216450_x_at	0.73	6.00
		25	13.00	6122.94	X219426_at	0.73	7.00
		30	15.50	6291.08	X214422_at	0.73	8.00
		all	15062.49	29492.51	X212649_at	0.73	9.00
					X211317_s_at	0.72	10.00
					X213813_x_at	0.72	11.00
					X210717_at	0.71	12.00
					X201844_s_at	0.71	13.00
					X202082_s_at	0.71	14.00
					X212451_at	0.71	15.00
					X215992_s_at	0.71	16.00
					X212212_s_at	0.70	17.00
					X210755_at	0.70	18.00
					X209226_s_at	0.70	19.00
					X213850_s_at	0.70	20.00
					...	...	...

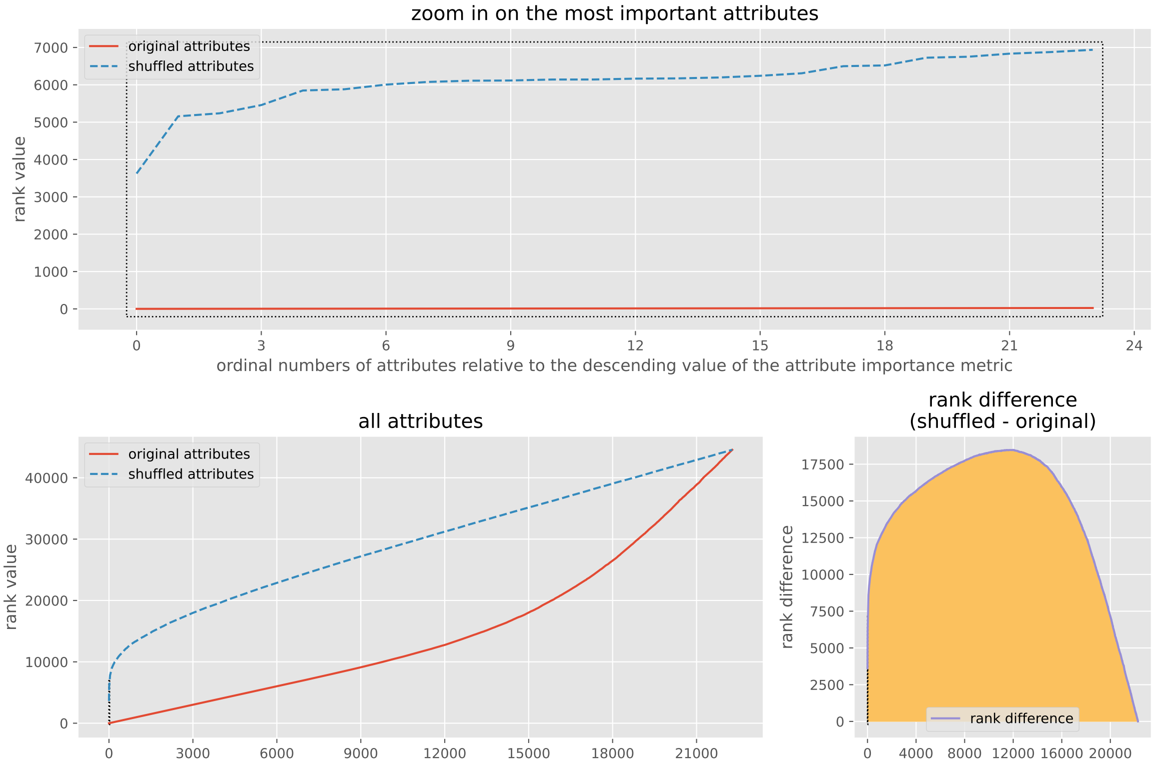


Figure A.32: Detailed attribute importance profiling results obtained for the correlation-based model on the *hepatitisC* data set.

A.9 Feature Importance Profiling - *skinPsoriatic* data set

	avg rank value of the top_k attributes					
	bireducts		XGBoost		correlation	
top_k	original	shuffled	original	shuffled	original	shuffled
10	5.50	740.10	5.50	34.00	5.50	16791.05
20	10.50	915.50	11.30	51.15	10.50	17775.90
30	15.50	1025.73	17.20	63.20	15.52	18405.58
50	25.50	1180.68	29.90	6634.26	25.50	19122.47
100	50.50	1403.09	16998.12	30683.13	50.51	20146.08
all	53702.89	55648.11	54662.99	54688.01	38626.78	70724.22

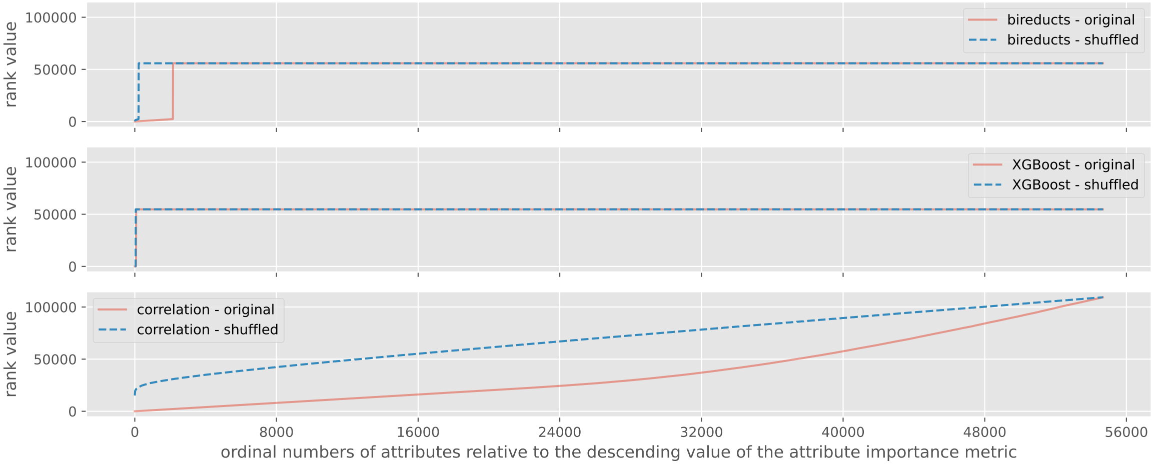


Figure A.33: The average attribute importance ranks of the original (A) and shuffled ($A^{\circlearrowright}$) attributes computed for the compared ranking methods on the *skinPsoriatic* data set.

Bireducts - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	count	global_gain	rank
algorithm	bireducts	3	2.00	412.33	Aff[...].U133_Plus_2.205349_at	5	1.38	1.00
allowed_randomness	0.05	5	3.00	569.60	Aff[...]133_Plus_2.207121_s_at	5	1.18	2.00
attrs_max_count	3	7	4.00	650.29	Aff[...]133_Plus_2.202805_s_at	4	1.16	3.00
candidates_count	100	10	5.50	740.10	Aff[...].U133_Plus_2.209921_at	4	1.13	4.00
chaos_fun	gini_impurity	15	8.00	844.20	Aff[...]133_Plus_2.218357_s_at	4	1.02	5.00
epsilon	0.00	20	10.50	915.50	Aff[...].U133_Plus_2.202338_at	5	0.99	6.00
n_bins	3	25	13.00	977.64	Aff[...].U133_Plus_2.231033_at	4	0.99	7.00
n_bireducts	1000	30	15.50	1025.73	Aff[...].U133_Plus_2.206149_at	4	0.97	8.00
probes_count	100	all	53702.89	55648.11	Aff[...]33_Plus_2.1557915_s_at	3	0.95	9.00
model stats					Aff[...]133_Plus_2.201584_s_at	4	0.95	10.00
mean_attrs_size	3.00				Aff[...].U133_Plus_2.201577_at	3	0.92	11.00
mean_objs_size	135.15				Aff[...].U133_Plus_2.202804_at	5	0.91	12.00
median_attrs_size	3.00				Aff[...].U133_Plus_2.223541_at	3	0.91	13.00
median_objs_size	136.00				Aff[...].U133_Plus_2.206337_at	4	0.91	14.00
					Aff[...].U133_Plus_2.217755_at	3	0.90	15.00
					Aff[...]133_Plus_2.223032_x_at	4	0.86	16.00
					Aff[...]133_Plus_2.208651_x_at	3	0.86	17.00
					Aff[...].U133_Plus_2.202934_at	4	0.85	18.00
					Aff[...]33_Plus_2.1554556_a_at	4	0.84	19.00
					Aff[...]133_Plus_2.202070_s_at	3	0.82	20.00
					...	...	...	...

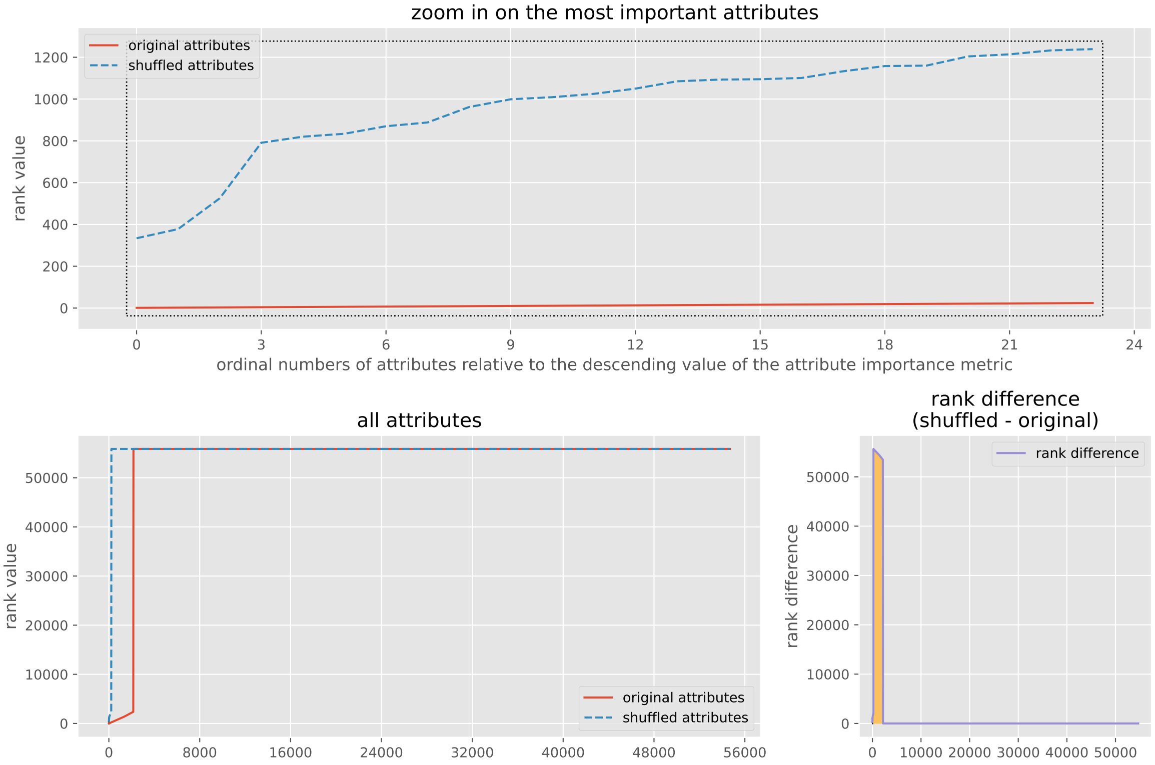


Figure A.34: Detailed attribute importance profiling results obtained for the bireduct-based ensembles on the *skinPsoriatic* data set.

XGBoost - detailed profile

hyperparameters		avg ranks			actual ranks			
param	value	top_k	original	shuffled	column	weight	total_gain	rank
algorithm	xgboost	3	2.00	16.67	Aff[...].U133_Plus_2.204698_at	1000	45768.19	1.00
learning_rate	0.00	5	3.00	23.00	Aff[...]133_Plus_2.233687_s_at	316	8542.27	2.00
max_depth	2	7	4.00	27.86	Aff[...]33_Plus_2.1555730_a_at	359	8069.96	3.00
num_boost_round	1000	10	5.50	34.00	Aff[...].U133_Plus_2.220528_at	315	6206.05	4.00
objective	multi:softmax	15	8.33	44.13	Aff[...]133_Plus_2.224890_s_at	245	4931.53	5.00
		20	11.30	51.15	Aff[...].U133_Plus_2.228360_at	245	3653.13	6.00
		25	14.24	57.68	Aff[...]133_Plus_2.224329_s_at	128	3188.15	7.00
		30	17.20	63.20	Aff[...].U133_Plus_2.222431_at	150	2996.40	8.00
		all	54662.99	54688.01	Aff[...]133_Plus_2.217992_s_at	140	1939.88	9.00
					Aff[...].U133_Plus_2.211958_at	161	1909.49	10.00
					shu[...]U133_Plus_2.1554483_at	244	1701.35	11.00
					Aff[...]133_Plus_2.214218_s_at	161	1434.62	12.00
					Aff[...].U133_Plus_2.218611_at	315	1335.25	13.00
					Aff[...].U133_Plus_2.243182_at	122	1326.11	14.00
					Aff[...].U133_Plus_2.230949_at	148	1202.67	15.00
					Aff[...].U133_Plus_2.226589_at	235	843.94	16.00
					Aff[...]133_Plus_2.228279_s_at	75	831.58	17.00
					Aff[...].U133_Plus_2.222491_at	93	828.17	18.00
					shu[...]U133_Plus_2.1560475_at	105	811.55	19.00
					shu[...].U133_Plus_2.216145_at	85	733.03	20.00
					...	...	...	...

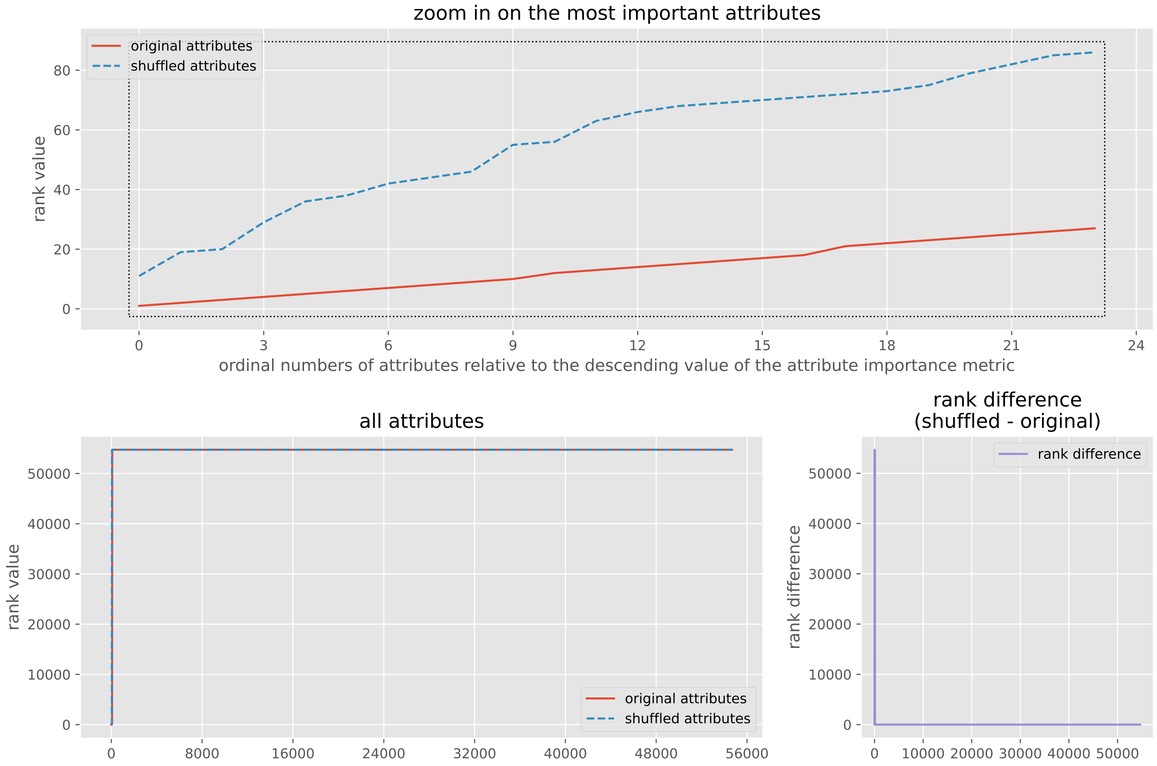


Figure A.35: Detailed attribute importance profiling results obtained for XGBoost on the *skinPsoriatic* data set.

Correlation - detailed profile

hyperparameters		avg ranks			actual ranks		
param	value	top_k	original	shuffled	column	correlation	rank
algorithm	correlation	3	2.00	15692.50	Aff[...]133_Plus_2.224890_s_at	0.81	1.00
		5	3.00	16025.20	Aff[...].U133_Plus_2.226565_at	0.80	2.00
		7	4.00	16268.50	Aff[...].U133_Plus_2.205568_at	0.78	3.00
		10	5.50	16791.05	Aff[...]133_Plus_2.202805_s_at	0.78	4.00
		15	8.00	17325.60	Aff[...]133_Plus_2.217917_s_at	0.77	5.00
		20	10.50	17775.90	Aff[...]133_Plus_2.224013_s_at	0.77	6.00
		25	13.00	18133.20	Aff[...].U133_Plus_2.217918_at	0.77	7.00
		30	15.52	18405.58	Aff[...]133_Plus_2.224555_x_at	0.77	8.00
		all	38626.78	70724.22	Aff[...].U133_Plus_2.228567_at	0.77	9.00
					Aff[...]133_Plus_2.209293_x_at	0.77	10.00
					Aff[...].U133_Plus_2.206149_at	0.77	11.50
					Aff[...]133_Plus_2.200822_x_at	0.77	11.50
					Aff[...].U133_Plus_2.221599_at	0.77	13.00
					Aff[...].U133_Plus_2.201874_at	0.77	14.00
					Aff[...]133_Plus_2.212442_s_at	0.76	15.00
					Aff[...].U133_Plus_2.200693_at	0.76	16.00
					Aff[...]133_Plus_2.202289_s_at	0.76	17.00
					Aff[...].U133_Plus_2.228360_at	0.76	18.00
					Aff[...].U133_Plus_2.202575_at	0.76	19.00
					Aff[...]133_Plus_2.203573_s_at	0.76	20.00
					...	...	...

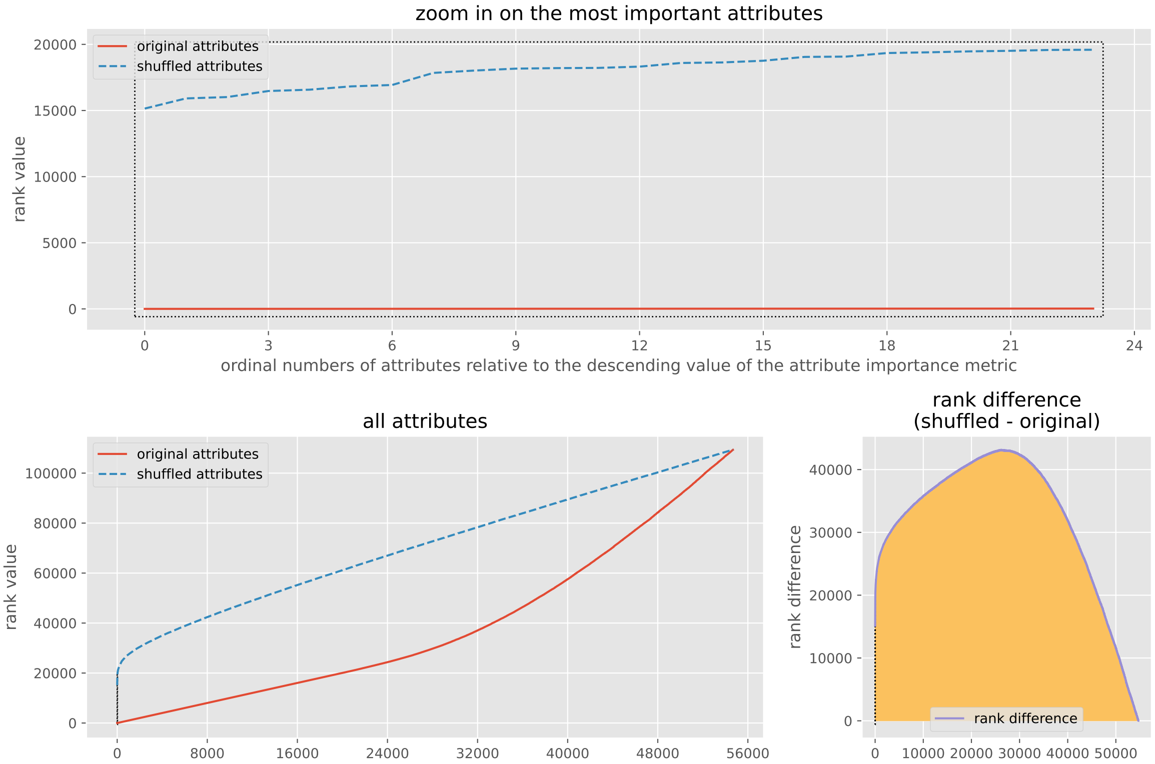
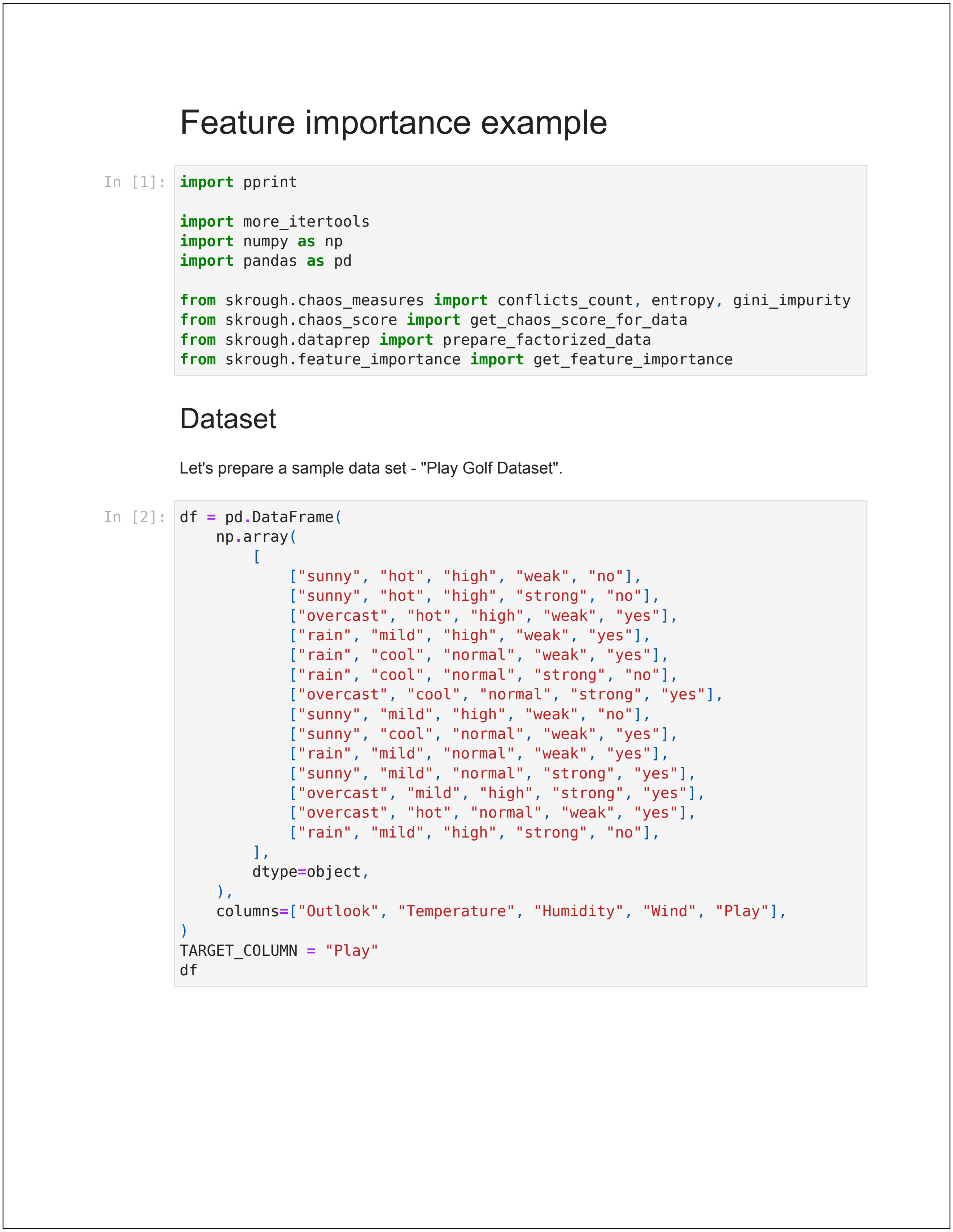


Figure A.36: Detailed attribute importance profiling results obtained for the correlation-based model on the *skinPsoriatic* data set.

Appendix B

Notebook Examples



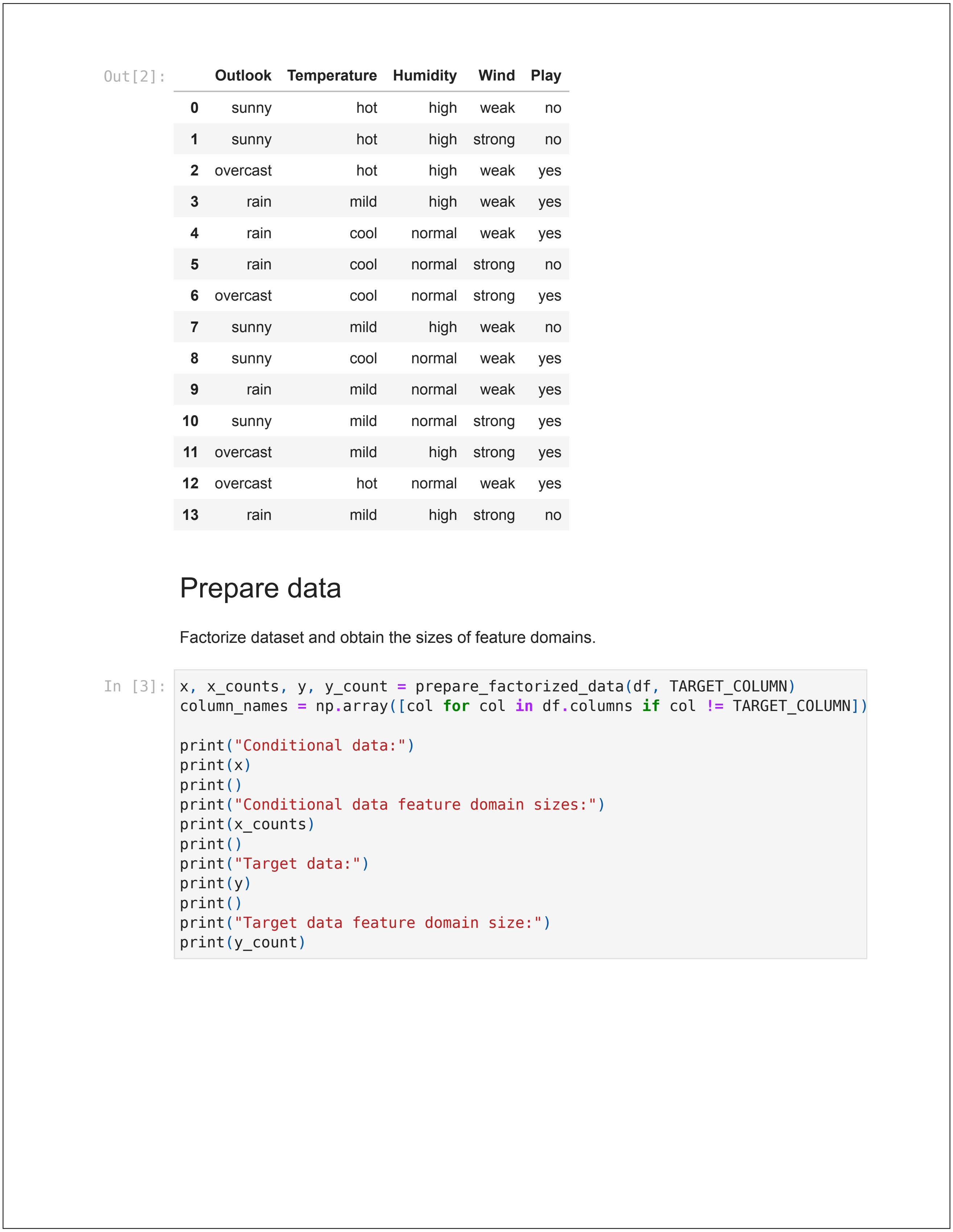
Feature importance example

In [1]:
```
import pprint

import more_itertools
import numpy as np
import pandas as pd

from skrough.chaos_measures import conflicts_count, entropy, gini_impurity
from skrough.chaos_score import get_chaos_score_for_data
from skrough.dataprep import prepare_factorized_data
from skrough.feature_importance import get_feature_importance
```

Dataset

Let's prepare a sample data set - "Play Golf Dataset".

In [2]:
```
df = pd.DataFrame(
    np.array(
        [
            ["sunny", "hot", "high", "weak", "no"],
            ["sunny", "hot", "high", "strong", "no"],
            ["overcast", "hot", "high", "weak", "yes"],
            ["rain", "mild", "high", "weak", "yes"],
            ["rain", "cool", "normal", "weak", "yes"],
            ["rain", "cool", "normal", "strong", "no"],
            ["overcast", "cool", "normal", "strong", "yes"],
            ["sunny", "mild", "high", "weak", "no"],
            ["sunny", "cool", "normal", "weak", "yes"],
            ["rain", "mild", "normal", "weak", "yes"],
            ["sunny", "mild", "normal", "strong", "yes"],
            ["overcast", "mild", "high", "strong", "yes"],
            ["overcast", "hot", "normal", "weak", "yes"],
            ["rain", "mild", "high", "strong", "no"],
        ],
        dtype=object,
    ),
    columns=["Outlook", "Temperature", "Humidity", "Wind", "Play"],
)
TARGET_COLUMN = "Play"
df
```



Out[2]:

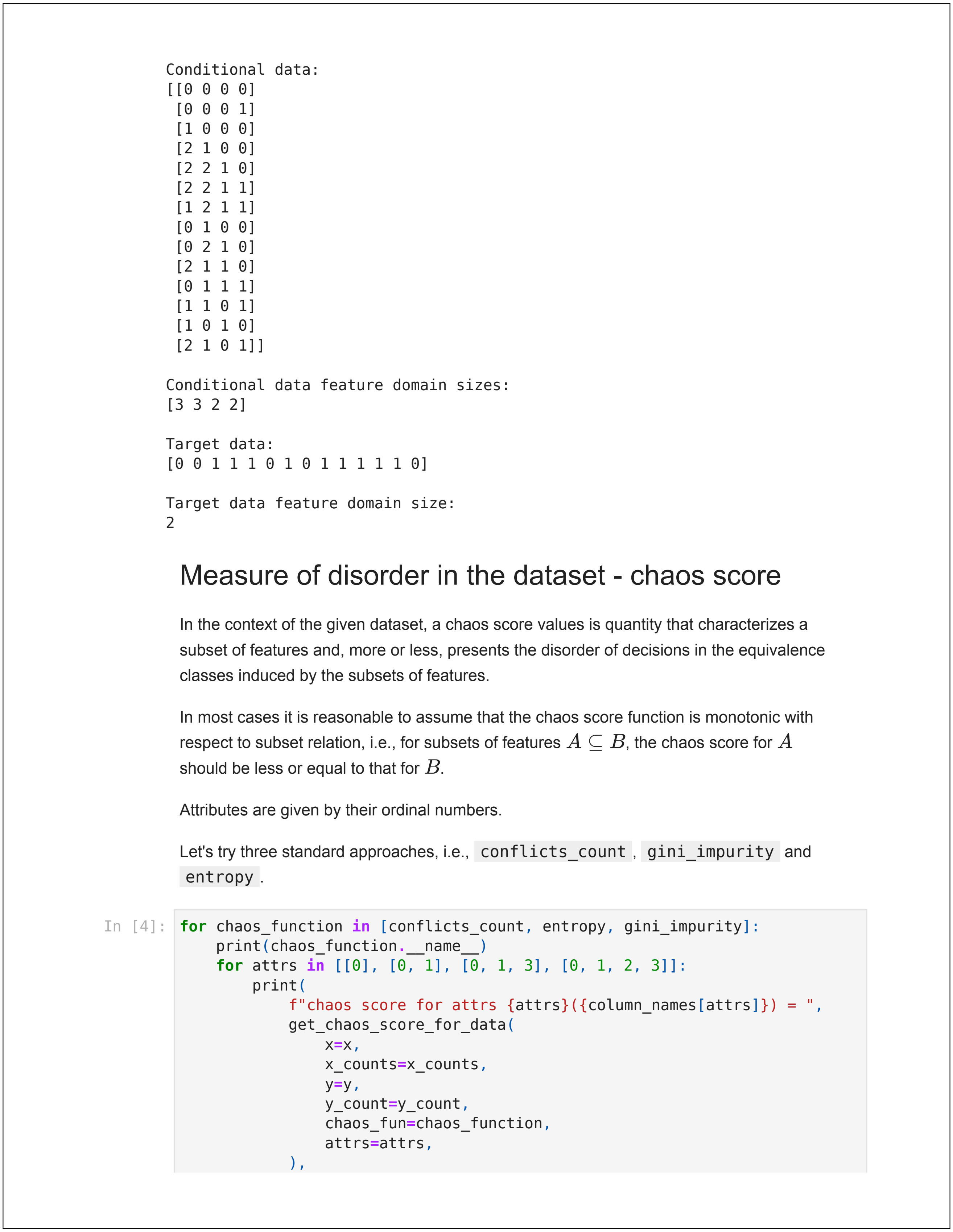
	Outlook	Temperature	Humidity	Wind	Play
0	sunny	hot	high	weak	no
1	sunny	hot	high	strong	no
2	overcast	hot	high	weak	yes
3	rain	mild	high	weak	yes
4	rain	cool	normal	weak	yes
5	rain	cool	normal	strong	no
6	overcast	cool	normal	strong	yes
7	sunny	mild	high	weak	no
8	sunny	cool	normal	weak	yes
9	rain	mild	normal	weak	yes
10	sunny	mild	normal	strong	yes
11	overcast	mild	high	strong	yes
12	overcast	hot	normal	weak	yes
13	rain	mild	high	strong	no

Prepare data

Factorize dataset and obtain the sizes of feature domains.

In [3]:

```python
x, x_counts, y, y_count = prepare_factorized_data(df, TARGET_COLUMN)
column_names = np.array([col for col in df.columns if col != TARGET_COLUMN])

print("Conditional data:")
print(x)
print()
print("Conditional data feature domain sizes:")
print(x_counts)
print()
print("Target data:")
print(y)
print()
print("Target data feature domain size:")
print(y_count)
```



```
Conditional data:
[[0 0 0 0]
 [0 0 0 1]
 [1 0 0 0]
 [2 1 0 0]
 [2 2 1 0]
 [2 2 1 1]
 [1 2 1 1]
 [0 1 0 0]
 [0 2 1 0]
 [2 1 1 0]
 [0 1 1 1]
 [1 1 0 1]
 [1 0 1 0]
 [2 1 0 1]]

Conditional data feature domain sizes:
[3 3 2 2]

Target data:
[0 0 1 1 1 0 1 0 1 1 1 1 1 0]

Target data feature domain size:
2
```

Measure of disorder in the dataset - chaos score

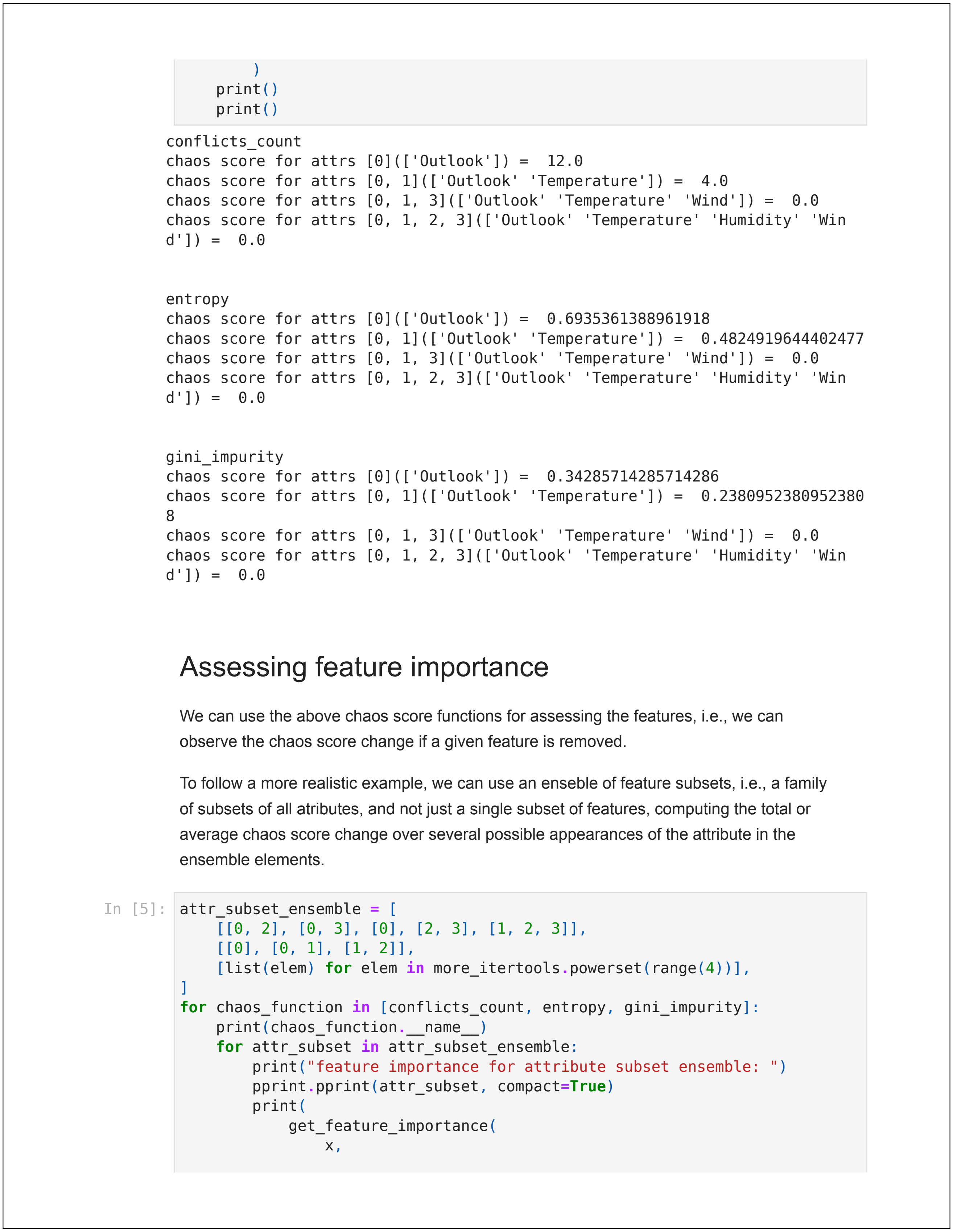
In the context of the given dataset, a chaos score values is quantity that characterizes a subset of features and, more or less, presents the disorder of decisions in the equivalence classes induced by the subsets of features.

In most cases it is reasonable to assume that the chaos score function is monotonic with respect to subset relation, i.e., for subsets of features $A \subseteq B$, the chaos score for A should be less or equal to that for B.

Attributes are given by their ordinal numbers.

Let's try three standard approaches, i.e., `conflicts_count`, `gini_impurity` and `entropy`.

In [4]:
```
for chaos_function in [conflicts_count, entropy, gini_impurity]:
    print(chaos_function.__name__)
    for attrs in [[0], [0, 1], [0, 1, 3], [0, 1, 2, 3]]:
        print(
            f"chaos score for attrs {attrs}({column_names[attrs]}) = ",
            get_chaos_score_for_data(
                x=x,
                x_counts=x_counts,
                y=y,
                y_count=y_count,
                chaos_fun=chaos_function,
                attrs=attrs,
            ),
```



```
        )
    print()
    print()
```

```
conflicts_count
chaos score for attrs [0](['Outlook']) =  12.0
chaos score for attrs [0, 1](['Outlook' 'Temperature']) =  4.0
chaos score for attrs [0, 1, 3](['Outlook' 'Temperature' 'Wind']) =  0.0
chaos score for attrs [0, 1, 2, 3](['Outlook' 'Temperature' 'Humidity' 'Win
d']) =  0.0


entropy
chaos score for attrs [0](['Outlook']) =  0.6935361388961918
chaos score for attrs [0, 1](['Outlook' 'Temperature']) =  0.4824919644402477
chaos score for attrs [0, 1, 3](['Outlook' 'Temperature' 'Wind']) =  0.0
chaos score for attrs [0, 1, 2, 3](['Outlook' 'Temperature' 'Humidity' 'Win
d']) =  0.0


gini_impurity
chaos score for attrs [0](['Outlook']) =  0.34285714285714286
chaos score for attrs [0, 1](['Outlook' 'Temperature']) =  0.2380952380952380
8
chaos score for attrs [0, 1, 3](['Outlook' 'Temperature' 'Wind']) =  0.0
chaos score for attrs [0, 1, 2, 3](['Outlook' 'Temperature' 'Humidity' 'Win
d']) =  0.0
```

Assessing feature importance

We can use the above chaos score functions for assessing the features, i.e., we can observe the chaos score change if a given feature is removed.

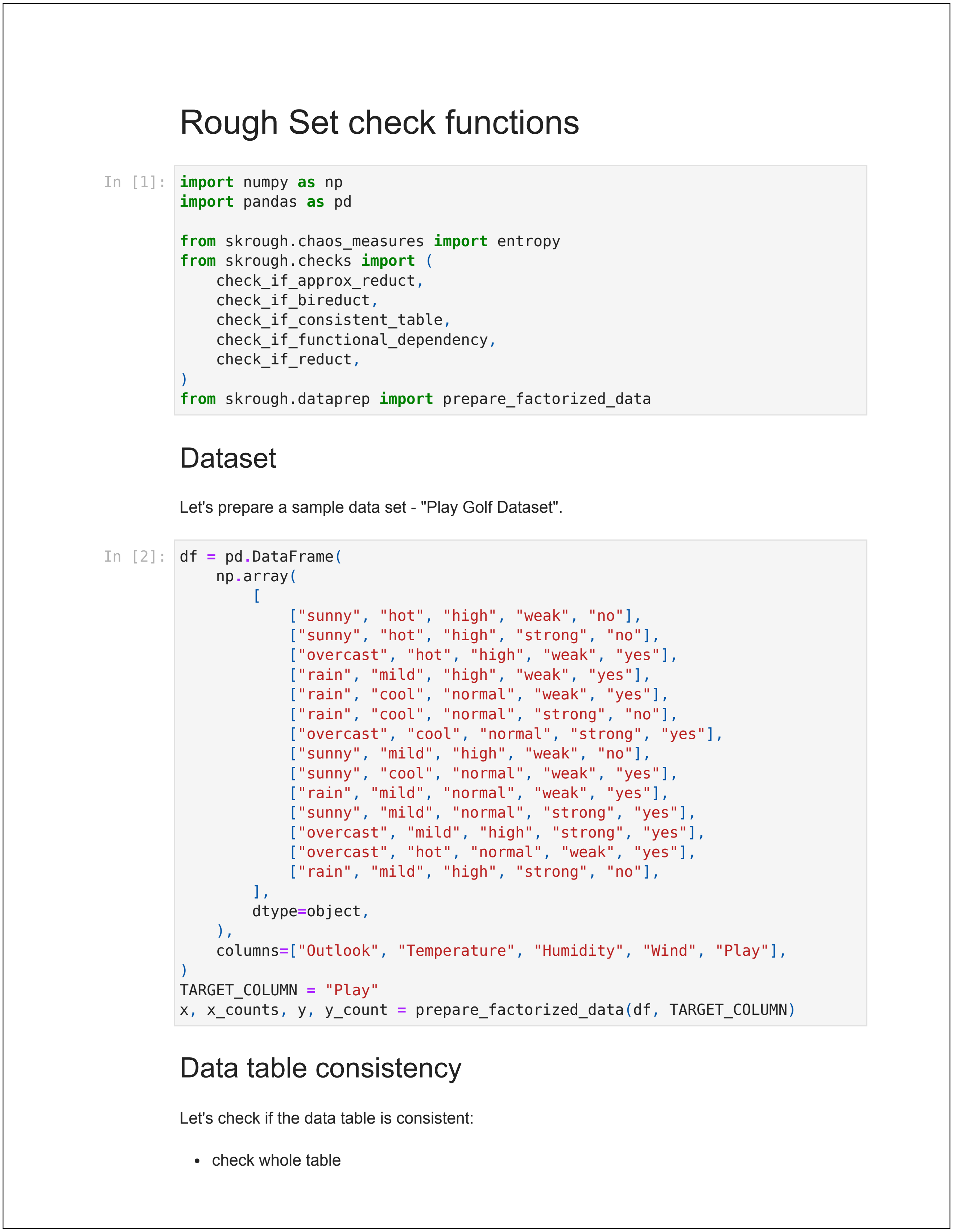
To follow a more realistic example, we can use an enseble of feature subsets, i.e., a family of subsets of all atributes, and not just a single subset of features, computing the total or average chaos score change over several possible appearances of the attribute in the ensemble elements.

In [5]:

```
attr_subset_ensemble = [
    [[0, 2], [0, 3], [0], [2, 3], [1, 2, 3]],
    [[0], [0, 1], [1, 2]],
    [list(elem) for elem in more_itertools.powerset(range(4))],
]
for chaos_function in [conflicts_count, entropy, gini_impurity]:
    print(chaos_function.__name__)
    for attr_subset in attr_subset_ensemble:
        print("feature importance for attribute subset ensemble: ")
        pprint.pprint(attr_subset, compact=True)
        print(
            get_feature_importance(
                x,
                x_counts,
                y,
                y_count,
                column_names,
                attr_subset,
                chaos_fun=chaos_function,
            )
        )
        print()
    print()
    print()
```

```
conflicts_count
feature importance for attribute subset ensemble:
[[0, 2], [0, 3], [0], [2, 3], [1, 2, 3]]
        column  count  global_gain  avg_global_gain
0      Outlook    3.0         66.0        22.000000
1  Temperature    1.0          4.0         4.000000
2     Humidity    3.0         25.0         8.333333
3         Wind    3.0         24.0         8.000000

feature importance for attribute subset ensemble:
[[0], [0, 1], [1, 2]]
        column  count  global_gain  avg_global_gain
0      Outlook    2.0         44.0             22.0
1  Temperature    2.0         17.0              8.5
2     Humidity    1.0          6.0              6.0
3         Wind    0.0          0.0              0.0

feature importance for attribute subset ensemble:
[[], [0], [1], [2], [3], [0, 1], [0, 2], [0, 3], [1, 2], [1, 3], [2, 3],
 [0, 1, 2], [0, 1, 3], [0, 2, 3], [1, 2, 3], [0, 1, 2, 3]]
        column  count  global_gain  avg_global_gain
0      Outlook    8.0        103.0           12.875
1  Temperature    8.0         69.0            8.625
2     Humidity    8.0         63.0            7.875
3         Wind    8.0         65.0            8.125



entropy
feature importance for attribute subset ensemble:
[[0, 2], [0, 3], [0], [2, 3], [1, 2, 3]]
        column  count  global_gain  avg_global_gain
0      Outlook    3.0     1.248090         0.416030
1  Temperature    1.0     0.107841         0.107841
2     Humidity    3.0     0.728552         0.242851
3         Wind    3.0     0.605939         0.201980

feature importance for attribute subset ensemble:
[[0], [0, 1], [1, 2]]
        column  count  global_gain  avg_global_gain
0      Outlook    2.0     0.675321         0.337661
1  Temperature    2.0     0.285209         0.142604
2     Humidity    1.0     0.196778         0.196778
3         Wind    0.0     0.000000         0.000000

feature importance for attribute subset ensemble:
[[], [0], [1], [2], [3], [0, 1], [0, 2], [0, 3], [1, 2], [1, 3], [2, 3],
 [0, 1, 2], [0, 1, 3], [0, 2, 3], [1, 2, 3], [0, 1, 2, 3]]
        column  count  global_gain  avg_global_gain
0      Outlook    8.0     4.089121         0.511140
1  Temperature    8.0     0.974797         0.121850
2     Humidity    8.0     1.613578         0.201697
3         Wind    8.0     1.939781         0.242473
```

```
gini_impurity
feature importance for attribute subset ensemble:
[[0, 2], [0, 3], [0], [2, 3], [1, 2, 3]]
        column  count  global_gain  avg_global_gain
0      Outlook    3.0     0.578912         0.192971
1  Temperature    1.0     0.047619         0.047619
2     Humidity    3.0     0.342857         0.114286
3         Wind    3.0     0.269728         0.089909

feature importance for attribute subset ensemble:
[[0], [0, 1], [1, 2]]
        column  count  global_gain  avg_global_gain
0      Outlook    2.0     0.318707         0.159354
1  Temperature    2.0     0.126871         0.063435
2     Humidity    1.0     0.095238         0.095238
3         Wind    0.0     0.000000         0.000000

feature importance for attribute subset ensemble:
[[], [0], [1], [2], [3], [0, 1], [0, 2], [0, 3], [1, 2], [1, 3], [2, 3],
 [0, 1, 2], [0, 1, 3], [0, 2, 3], [1, 2, 3], [0, 1, 2, 3]]
        column  count  global_gain  avg_global_gain
0      Outlook    8.0     1.959864         0.244983
1  Temperature    8.0     0.455102         0.056888
2     Humidity    8.0     0.791837         0.098980
3         Wind    8.0     0.931293         0.116412
```



Rough Set check functions

In [1]:

```
import numpy as np
import pandas as pd

from skrough.chaos_measures import entropy
from skrough.checks import (
    check_if_approx_reduct,
    check_if_bireduct,
    check_if_consistent_table,
    check_if_functional_dependency,
    check_if_reduct,
)
from skrough.dataprep import prepare_factorized_data
```

Dataset

Let's prepare a sample data set - "Play Golf Dataset".

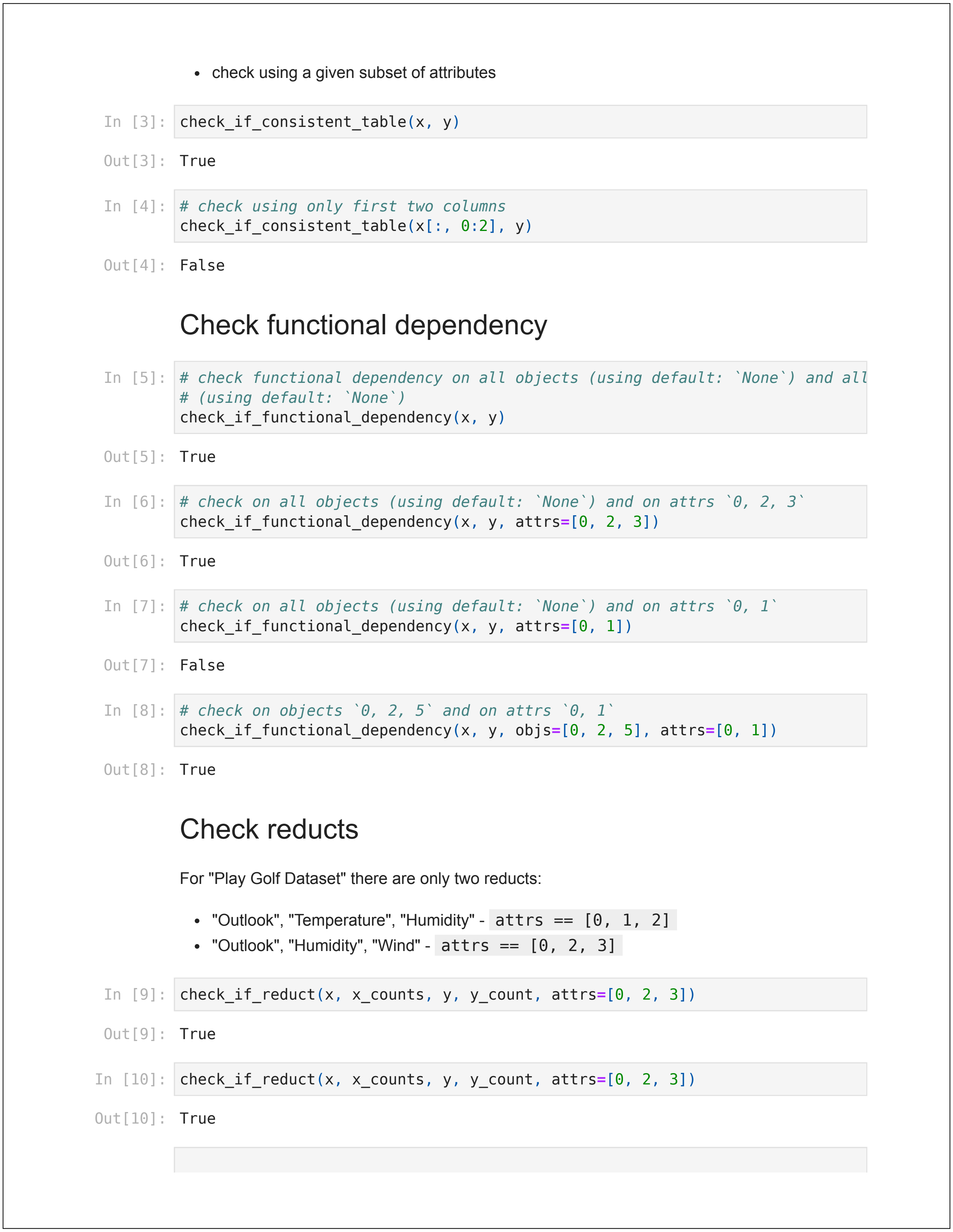
In [2]:

```
df = pd.DataFrame(
    np.array(
        [
            ["sunny", "hot", "high", "weak", "no"],
            ["sunny", "hot", "high", "strong", "no"],
            ["overcast", "hot", "high", "weak", "yes"],
            ["rain", "mild", "high", "weak", "yes"],
            ["rain", "cool", "normal", "weak", "yes"],
            ["rain", "cool", "normal", "strong", "no"],
            ["overcast", "cool", "normal", "strong", "yes"],
            ["sunny", "mild", "high", "weak", "no"],
            ["sunny", "cool", "normal", "weak", "yes"],
            ["rain", "mild", "normal", "weak", "yes"],
            ["sunny", "mild", "normal", "strong", "yes"],
            ["overcast", "mild", "high", "strong", "yes"],
            ["overcast", "hot", "normal", "weak", "yes"],
            ["rain", "mild", "high", "strong", "no"],
        ],
        dtype=object,
    ),
    columns=["Outlook", "Temperature", "Humidity", "Wind", "Play"],
)
TARGET_COLUMN = "Play"
x, x_counts, y, y_count = prepare_factorized_data(df, TARGET_COLUMN)
```

Data table consistency

Let's check if the data table is consistent:

- check whole table



- check using a given subset of attributes

```
In [3]: check_if_consistent_table(x, y)

Out[3]: True

In [4]: # check using only first two columns
        check_if_consistent_table(x[:, 0:2], y)

Out[4]: False
```

Check functional dependency

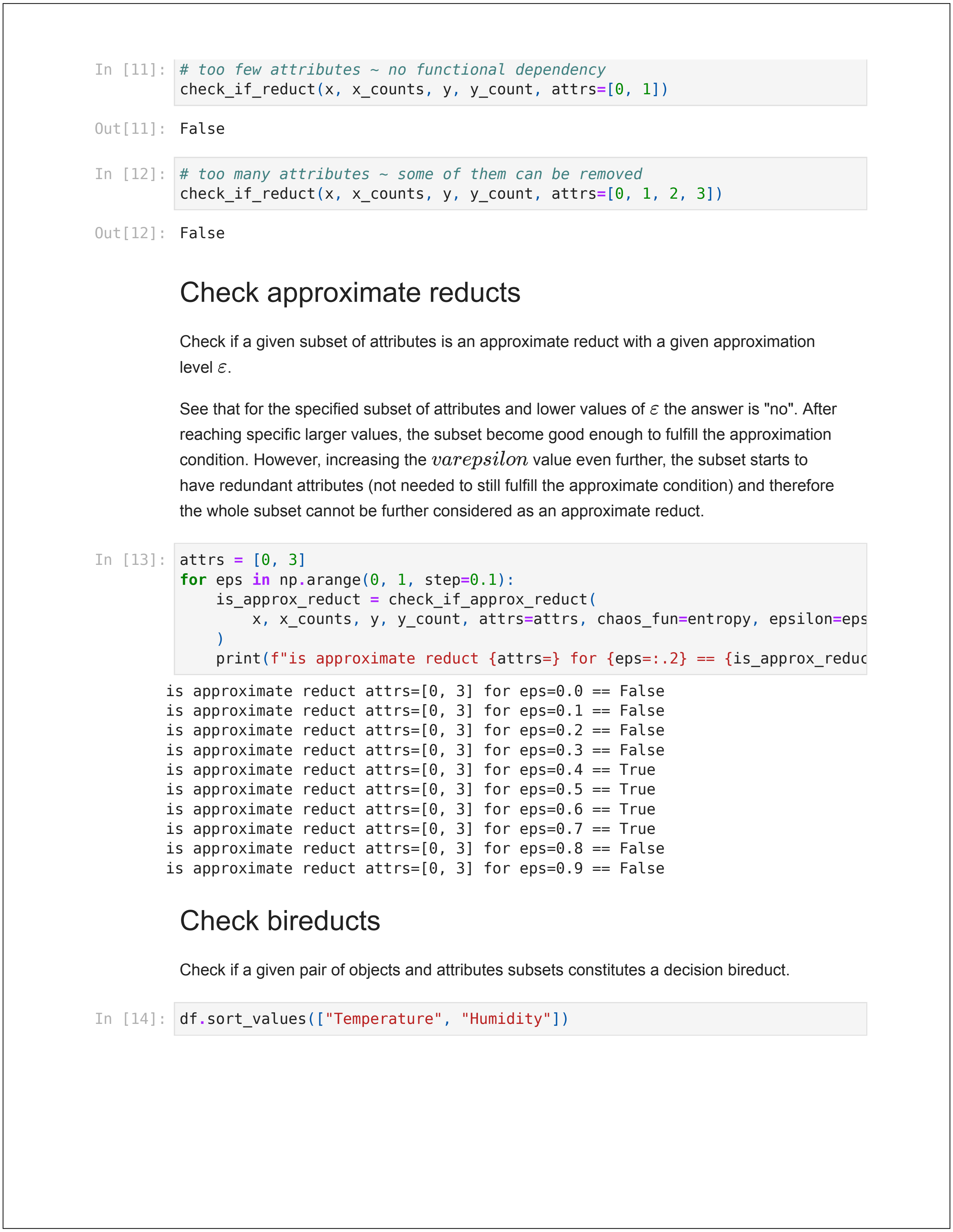
```
In [5]: # check functional dependency on all objects (using default: `None`) and all
        # (using default: `None`)
        check_if_functional_dependency(x, y)

Out[5]: True

In [6]: # check on all objects (using default: `None`) and on attrs `0, 2, 3`
        check_if_functional_dependency(x, y, attrs=[0, 2, 3])

Out[6]: True

In [7]: # check on all objects (using default: `None`) and on attrs `0, 1`
        check_if_functional_dependency(x, y, attrs=[0, 1])

Out[7]: False

In [8]: # check on objects `0, 2, 5` and on attrs `0, 1`
        check_if_functional_dependency(x, y, objs=[0, 2, 5], attrs=[0, 1])

Out[8]: True
```

Check reducts

For "Play Golf Dataset" there are only two reducts:

- "Outlook", "Temperature", "Humidity" - `attrs == [0, 1, 2]`
- "Outlook", "Humidity", "Wind" - `attrs == [0, 2, 3]`

```
In [9]: check_if_reduct(x, x_counts, y, y_count, attrs=[0, 2, 3])

Out[9]: True

In [10]: check_if_reduct(x, x_counts, y, y_count, attrs=[0, 2, 3])

Out[10]: True
```



```
In [11]: # too few attributes ~ no functional dependency
         check_if_reduct(x, x_counts, y, y_count, attrs=[0, 1])

Out[11]: False
```

```
In [12]: # too many attributes ~ some of them can be removed
         check_if_reduct(x, x_counts, y, y_count, attrs=[0, 1, 2, 3])

Out[12]: False
```

Check approximate reducts

Check if a given subset of attributes is an approximate reduct with a given approximation level ε.

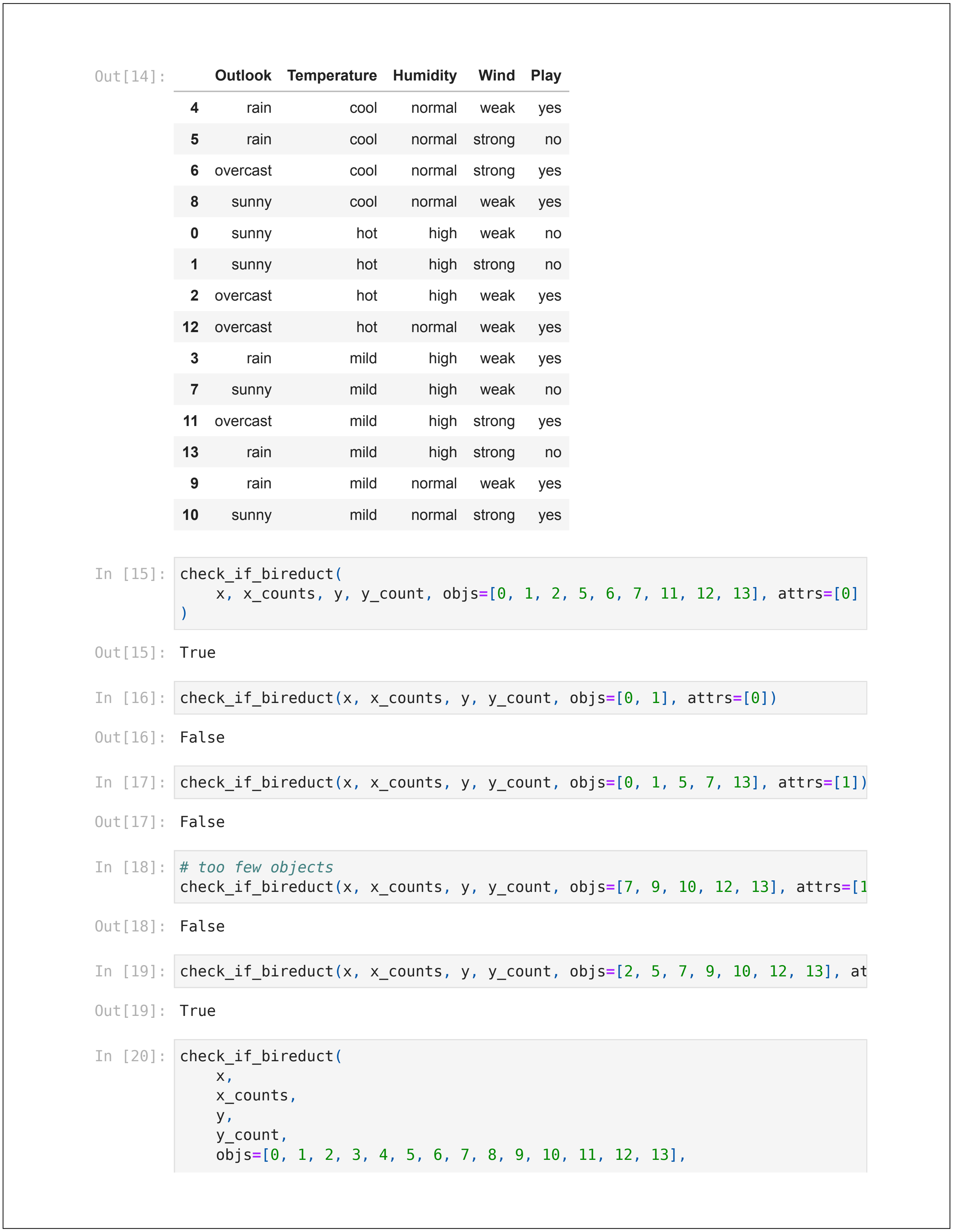
See that for the specified subset of attributes and lower values of ε the answer is "no". After reaching specific larger values, the subset become good enough to fulfill the approximation condition. However, increasing the $varepsilon$ value even further, the subset starts to have redundant attributes (not needed to still fulfill the approximate condition) and therefore the whole subset cannot be further considered as an approximate reduct.

```
In [13]: attrs = [0, 3]
         for eps in np.arange(0, 1, step=0.1):
             is_approx_reduct = check_if_approx_reduct(
                 x, x_counts, y, y_count, attrs=attrs, chaos_fun=entropy, epsilon=eps
             )
             print(f"is approximate reduct {attrs=} for {eps=:.2} == {is_approx_reduc

        is approximate reduct attrs=[0, 3] for eps=0.0 == False
        is approximate reduct attrs=[0, 3] for eps=0.1 == False
        is approximate reduct attrs=[0, 3] for eps=0.2 == False
        is approximate reduct attrs=[0, 3] for eps=0.3 == False
        is approximate reduct attrs=[0, 3] for eps=0.4 == True
        is approximate reduct attrs=[0, 3] for eps=0.5 == True
        is approximate reduct attrs=[0, 3] for eps=0.6 == True
        is approximate reduct attrs=[0, 3] for eps=0.7 == True
        is approximate reduct attrs=[0, 3] for eps=0.8 == False
        is approximate reduct attrs=[0, 3] for eps=0.9 == False
```

Check bireducts

Check if a given pair of objects and attributes subsets constitutes a decision bireduct.

```
In [14]: df.sort_values(["Temperature", "Humidity"])
```



Out[14]:

	Outlook	Temperature	Humidity	Wind	Play
4	rain	cool	normal	weak	yes
5	rain	cool	normal	strong	no
6	overcast	cool	normal	strong	yes
8	sunny	cool	normal	weak	yes
0	sunny	hot	high	weak	no
1	sunny	hot	high	strong	no
2	overcast	hot	high	weak	yes
12	overcast	hot	normal	weak	yes
3	rain	mild	high	weak	yes
7	sunny	mild	high	weak	no
11	overcast	mild	high	strong	yes
13	rain	mild	high	strong	no
9	rain	mild	normal	weak	yes
10	sunny	mild	normal	strong	yes

In [15]:

```
check_if_bireduct(
    x, x_counts, y, y_count, objs=[0, 1, 2, 5, 6, 7, 11, 12, 13], attrs=[0]
)
```

Out[15]: True

In [16]:

```
check_if_bireduct(x, x_counts, y, y_count, objs=[0, 1], attrs=[0])
```

Out[16]: False

In [17]:

```
check_if_bireduct(x, x_counts, y, y_count, objs=[0, 1, 5, 7, 13], attrs=[1])
```

Out[17]: False

In [18]:

```
# too few objects
check_if_bireduct(x, x_counts, y, y_count, objs=[7, 9, 10, 12, 13], attrs=[1
```

Out[18]: False

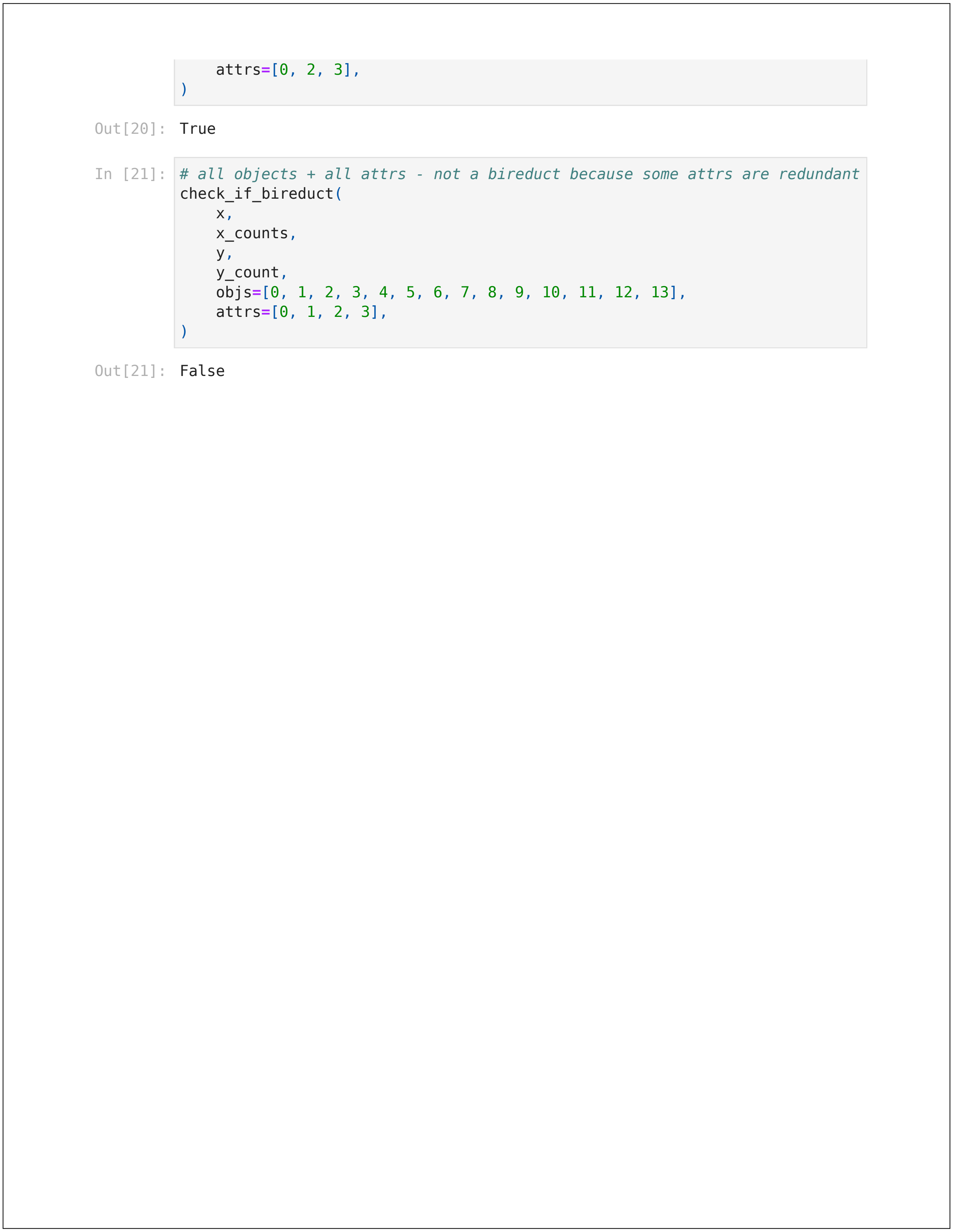
In [19]:

```
check_if_bireduct(x, x_counts, y, y_count, objs=[2, 5, 7, 9, 10, 12, 13], at
```

Out[19]: True

In [20]:

```
check_if_bireduct(
    x,
    x_counts,
    y,
    y_count,
    objs=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13],
```

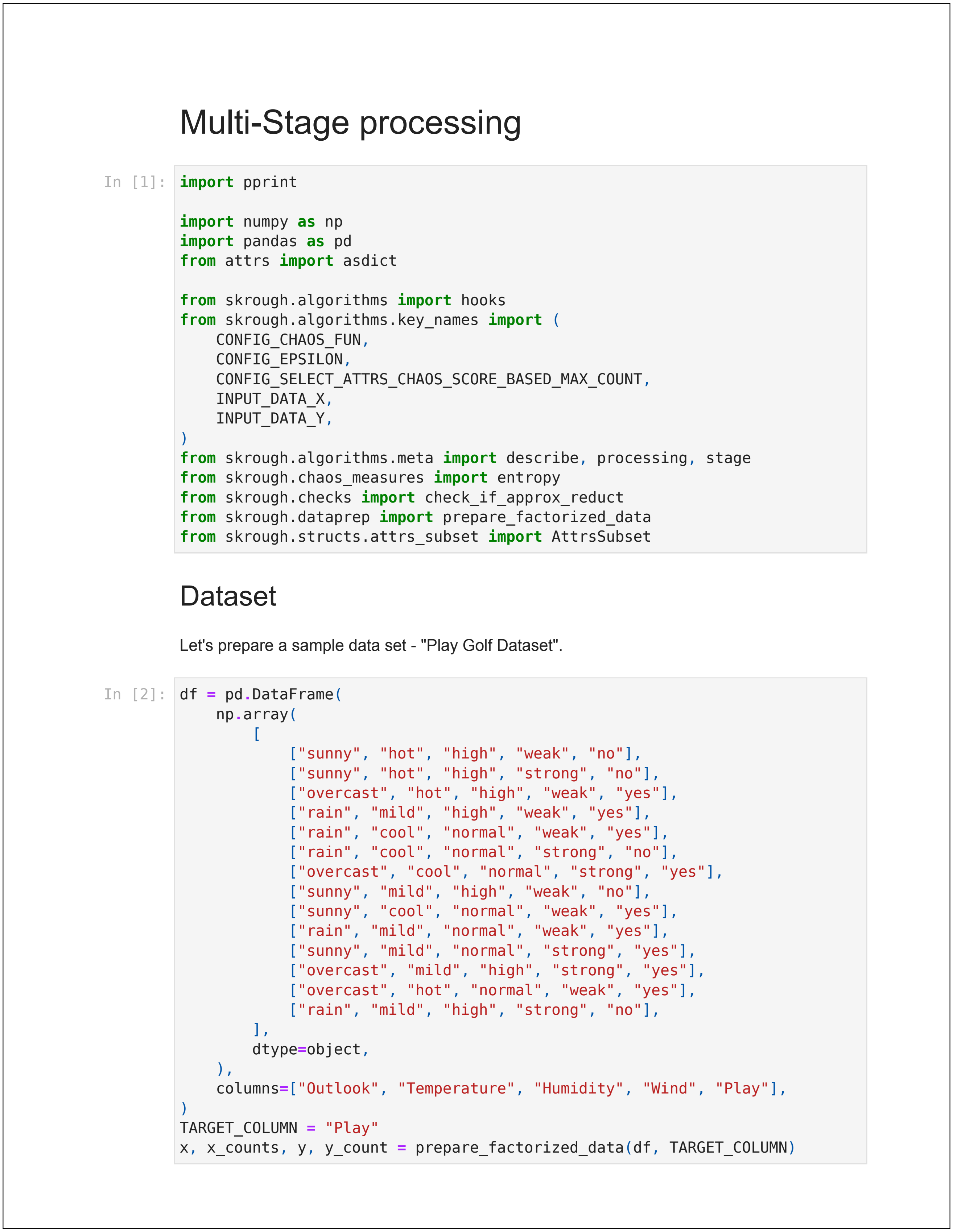


```
    attrs=[0, 2, 3],
)
```

```
Out[20]: True
```

```
In [21]: # all objects + all attrs - not a bireduct because some attrs are redundant
         check_if_bireduct(
             x,
             x_counts,
             y,
             y_count,
             objs=[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13],
             attrs=[0, 1, 2, 3],
         )
```

```
Out[21]: False
```



Multi-Stage processing

In [1]:

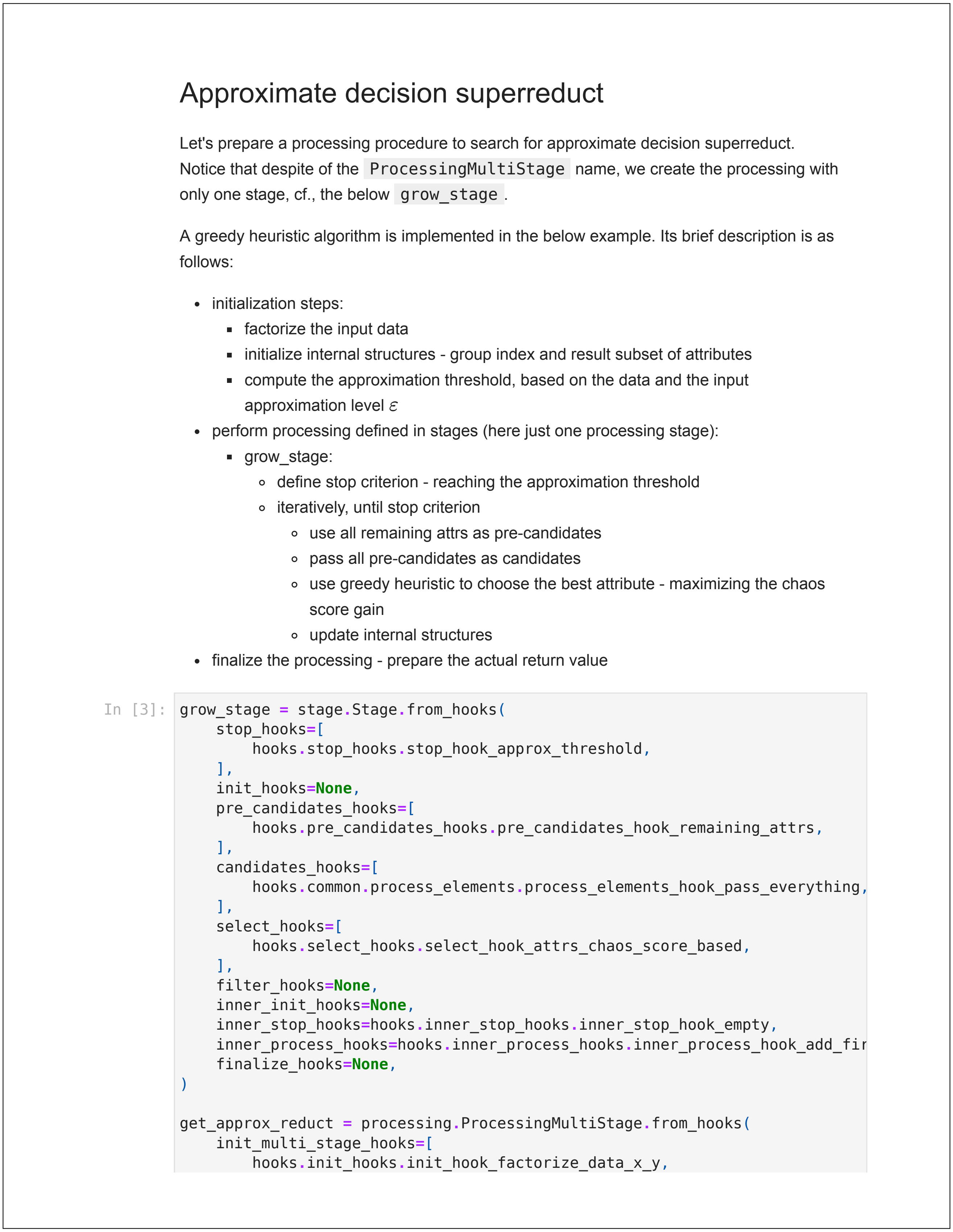
```
import pprint

import numpy as np
import pandas as pd
from attrs import asdict

from skrough.algorithms import hooks
from skrough.algorithms.key_names import (
    CONFIG_CHAOS_FUN,
    CONFIG_EPSILON,
    CONFIG_SELECT_ATTRS_CHAOS_SCORE_BASED_MAX_COUNT,
    INPUT_DATA_X,
    INPUT_DATA_Y,
)
from skrough.algorithms.meta import describe, processing, stage
from skrough.chaos_measures import entropy
from skrough.checks import check_if_approx_reduct
from skrough.dataprep import prepare_factorized_data
from skrough.structs.attrs_subset import AttrsSubset
```

Dataset

Let's prepare a sample data set - "Play Golf Dataset".

In [2]:

```
df = pd.DataFrame(
    np.array(
        [
            ["sunny", "hot", "high", "weak", "no"],
            ["sunny", "hot", "high", "strong", "no"],
            ["overcast", "hot", "high", "weak", "yes"],
            ["rain", "mild", "high", "weak", "yes"],
            ["rain", "cool", "normal", "weak", "yes"],
            ["rain", "cool", "normal", "strong", "no"],
            ["overcast", "cool", "normal", "strong", "yes"],
            ["sunny", "mild", "high", "weak", "no"],
            ["sunny", "cool", "normal", "weak", "yes"],
            ["rain", "mild", "normal", "weak", "yes"],
            ["sunny", "mild", "normal", "strong", "yes"],
            ["overcast", "mild", "high", "strong", "yes"],
            ["overcast", "hot", "normal", "weak", "yes"],
            ["rain", "mild", "high", "strong", "no"],
        ],
        dtype=object,
    ),
    columns=["Outlook", "Temperature", "Humidity", "Wind", "Play"],
)
TARGET_COLUMN = "Play"
x, x_counts, y, y_count = prepare_factorized_data(df, TARGET_COLUMN)
```



Approximate decision superreduct

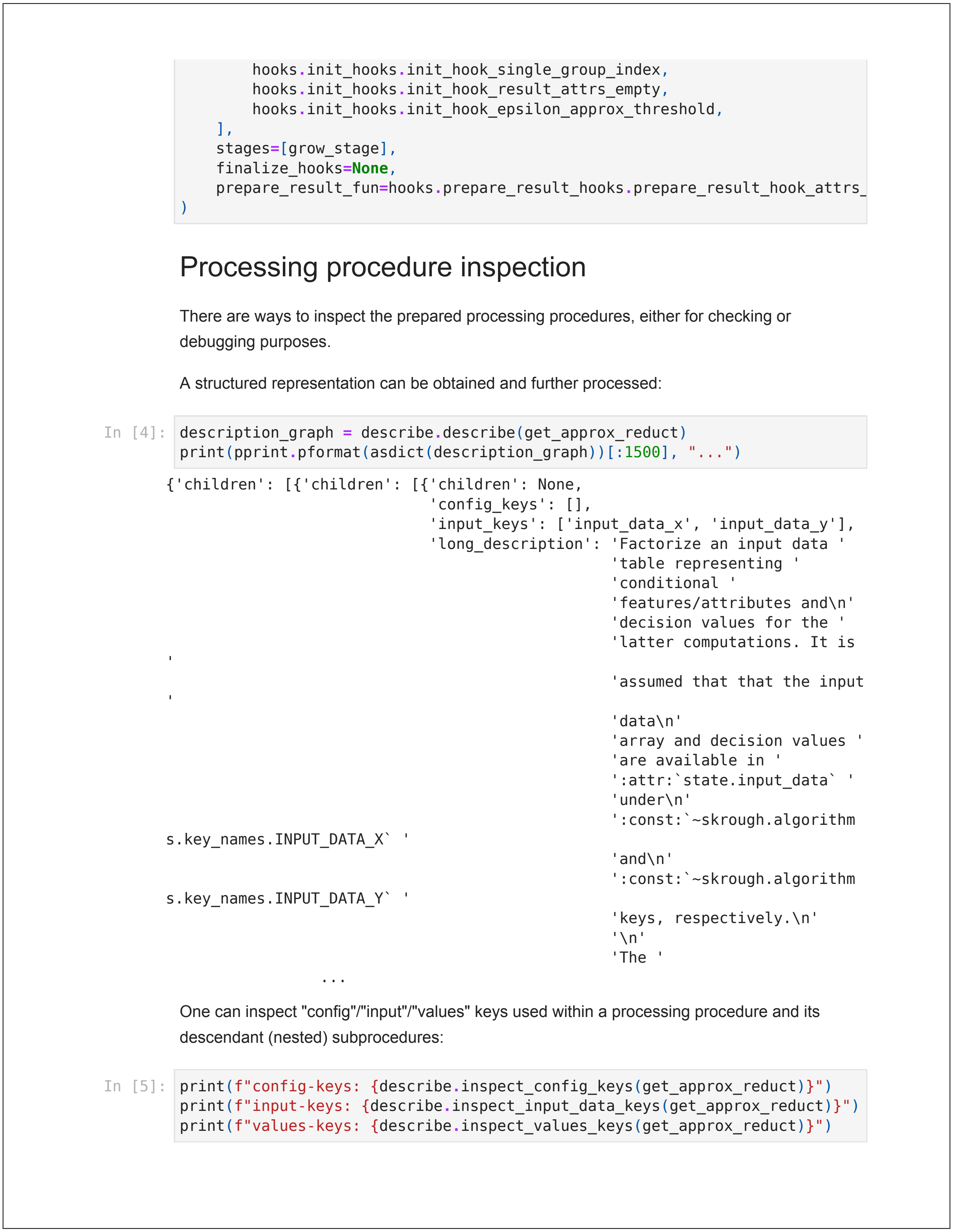
Let's prepare a processing procedure to search for approximate decision superreduct. Notice that despite of the `ProcessingMultiStage` name, we create the processing with only one stage, cf., the below `grow_stage`.

A greedy heuristic algorithm is implemented in the below example. Its brief description is as follows:

- initialization steps:
  - factorize the input data
  - initialize internal structures - group index and result subset of attributes
  - compute the approximation threshold, based on the data and the input approximation level ε
- perform processing defined in stages (here just one processing stage):
  - grow_stage:
    - define stop criterion - reaching the approximation threshold
    - iteratively, until stop criterion
      - use all remaining attrs as pre-candidates
      - pass all pre-candidates as candidates
      - use greedy heuristic to choose the best attribute - maximizing the chaos score gain
      - update internal structures
- finalize the processing - prepare the actual return value

In [3]:

```
grow_stage = stage.Stage.from_hooks(
    stop_hooks=[
        hooks.stop_hooks.stop_hook_approx_threshold,
    ],
    init_hooks=None,
    pre_candidates_hooks=[
        hooks.pre_candidates_hooks.pre_candidates_hook_remaining_attrs,
    ],
    candidates_hooks=[
        hooks.common.process_elements.process_elements_hook_pass_everything,
    ],
    select_hooks=[
        hooks.select_hooks.select_hook_attrs_chaos_score_based,
    ],
    filter_hooks=None,
    inner_init_hooks=None,
    inner_stop_hooks=hooks.inner_stop_hooks.inner_stop_hook_empty,
    inner_process_hooks=hooks.inner_process_hooks.inner_process_hook_add_fir
    finalize_hooks=None,
)

get_approx_reduct = processing.ProcessingMultiStage.from_hooks(
    init_multi_stage_hooks=[
        hooks.init_hooks.init_hook_factorize_data_x_y,
```



```
        hooks.init_hooks.init_hook_single_group_index,
        hooks.init_hooks.init_hook_result_attrs_empty,
        hooks.init_hooks.init_hook_epsilon_approx_threshold,
    ],
    stages=[grow_stage],
    finalize_hooks=None,
    prepare_result_fun=hooks.prepare_result_hooks.prepare_result_hook_attrs_
)
```

Processing procedure inspection

There are ways to inspect the prepared processing procedures, either for checking or debugging purposes.

A structured representation can be obtained and further processed:

In [4]:

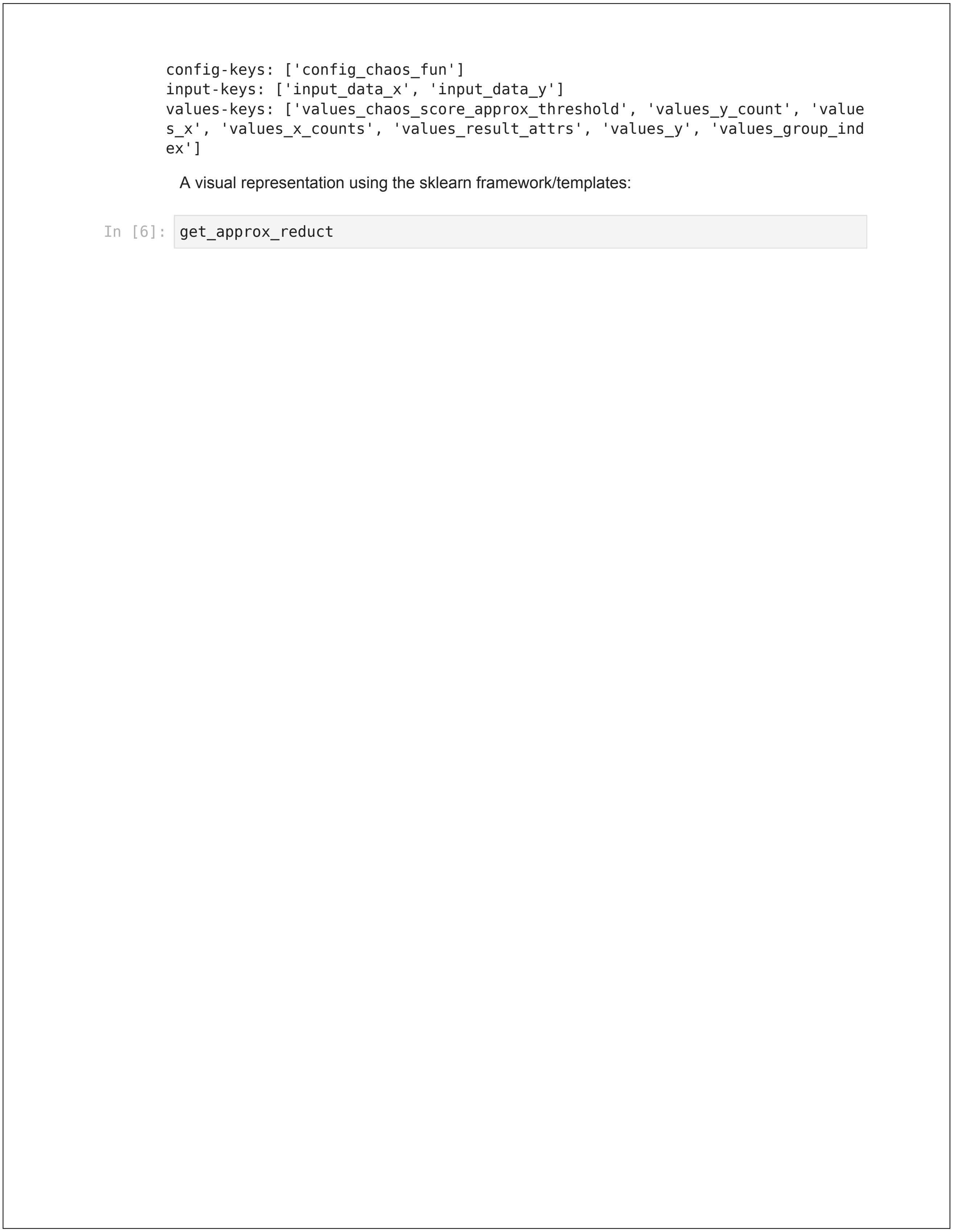
```
description_graph = describe.describe(get_approx_reduct)
print(pprint.pformat(asdict(description_graph))[:1500], "...")
```

```
{'children': [{'children': [{'children': None,
                             'config_keys': [],
                             'input_keys': ['input_data_x', 'input_data_y'],
                             'long_description': 'Factorize an input data '
                                                 'table representing '
                                                 'conditional '
                                                 'features/attributes and\n'
                                                 'decision values for the '
                                                 'latter computations. It is
'
                                                 'assumed that that the input
'
                                                 'data\n'
                                                 'array and decision values '
                                                 'are available in '
                                                 ':attr:`state.input_data` '
                                                 'under\n'
                                                 ':const:`~skrough.algorithm
s.key_names.INPUT_DATA_X` '
                                                 'and\n'
                                                 ':const:`~skrough.algorithm
s.key_names.INPUT_DATA_Y` '
                                                 'keys, respectively.\n'
                                                 '\n'
                                                 'The '
                 ...
```

One can inspect "config"/"input"/"values" keys used within a processing procedure and its descendant (nested) subprocedures:

In [5]:

```
print(f"config-keys: {describe.inspect_config_keys(get_approx_reduct)}")
print(f"input-keys: {describe.inspect_input_data_keys(get_approx_reduct)}")
print(f"values-keys: {describe.inspect_values_keys(get_approx_reduct)}")
```

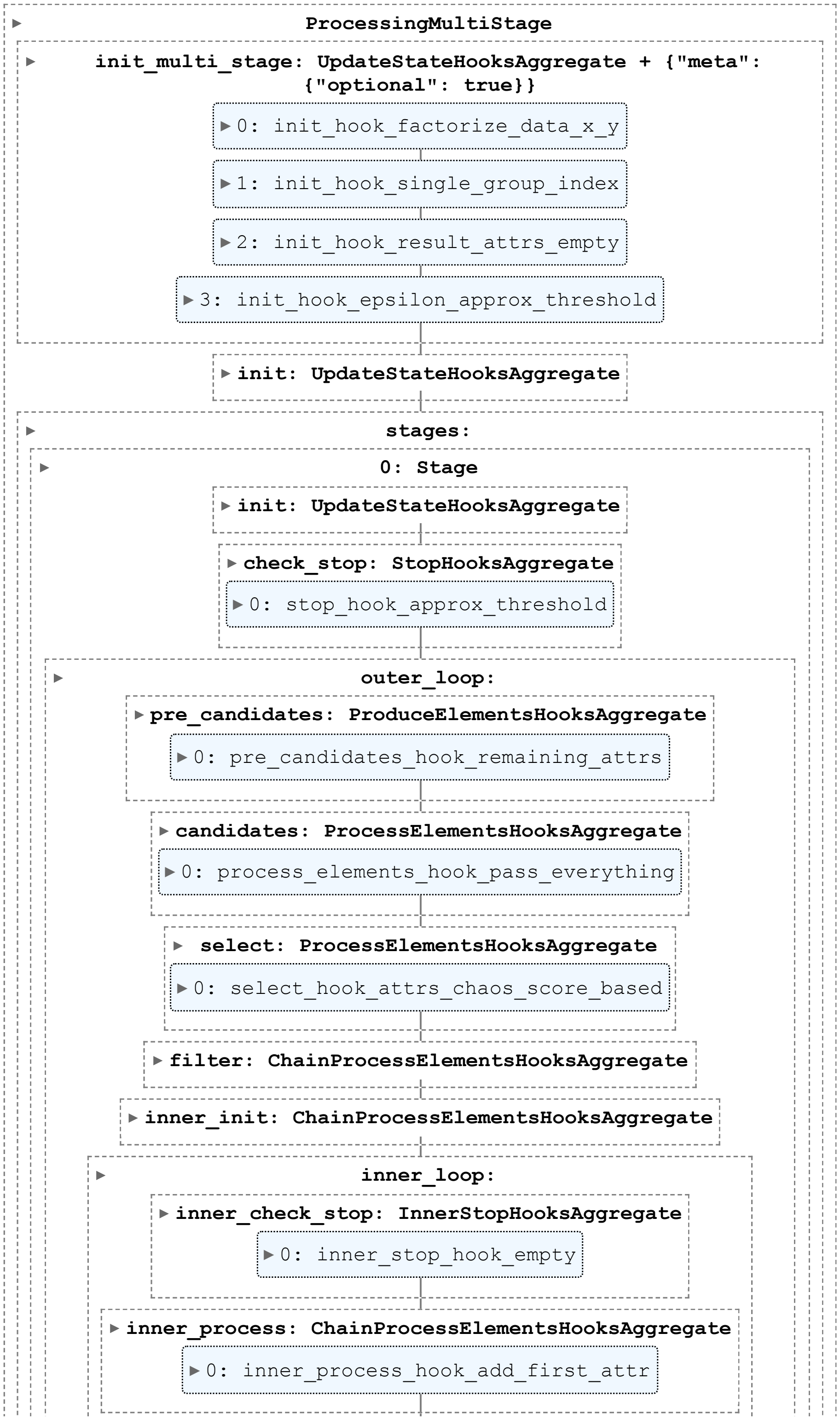


```
config-keys: ['config_chaos_fun']
input-keys: ['input_data_x', 'input_data_y']
values-keys: ['values_chaos_score_approx_threshold', 'values_y_count', 'value
s_x', 'values_x_counts', 'values_result_attrs', 'values_y', 'values_group_ind
ex']
```

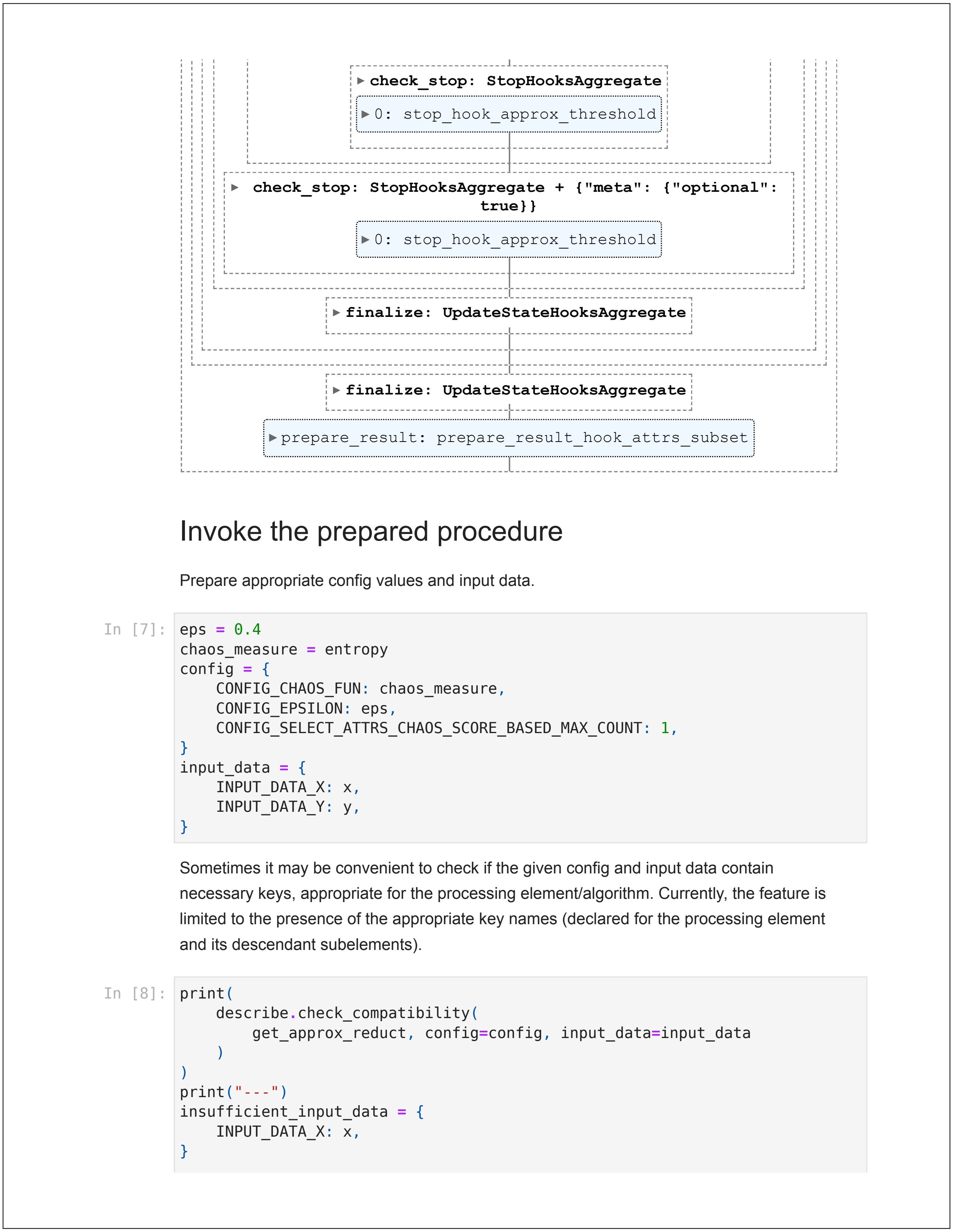
A visual representation using the sklearn framework/templates:

```
In [6]: get_approx_reduct
```

Out[6]:



```
▶ ProcessingMultiStage
  ▶ init_multi_stage: UpdateStateHooksAggregate + {"meta":
    {"optional": true}}
      ▶ 0: init_hook_factorize_data_x_y
      ▶ 1: init_hook_single_group_index
      ▶ 2: init_hook_result_attrs_empty
      ▶ 3: init_hook_epsilon_approx_threshold
  ▶ init: UpdateStateHooksAggregate
  ▶ stages:
    ▶ 0: Stage
      ▶ init: UpdateStateHooksAggregate
      ▶ check_stop: StopHooksAggregate
          ▶ 0: stop_hook_approx_threshold
      ▶ outer_loop:
        ▶ pre_candidates: ProduceElementsHooksAggregate
            ▶ 0: pre_candidates_hook_remaining_attrs
        ▶ candidates: ProcessElementsHooksAggregate
            ▶ 0: process_elements_hook_pass_everything
        ▶ select: ProcessElementsHooksAggregate
            ▶ 0: select_hook_attrs_chaos_score_based
        ▶ filter: ChainProcessElementsHooksAggregate
        ▶ inner_init: ChainProcessElementsHooksAggregate
        ▶ inner_loop:
          ▶ inner_check_stop: InnerStopHooksAggregate
              ▶ 0: inner_stop_hook_empty
          ▶ inner_process: ChainProcessElementsHooksAggregate
              ▶ 0: inner_process_hook_add_first_attr
```


▸ check_stop: StopHooksAggregate
▸ 0: stop_hook_approx_threshold
▸ check_stop: StopHooksAggregate + {"meta": {"optional": true}}
▸ 0: stop_hook_approx_threshold
▸ finalize: UpdateStateHooksAggregate
▸ finalize: UpdateStateHooksAggregate
▸ prepare_result: prepare_result_hook_attrs_subset

Invoke the prepared procedure

Prepare appropriate config values and input data.

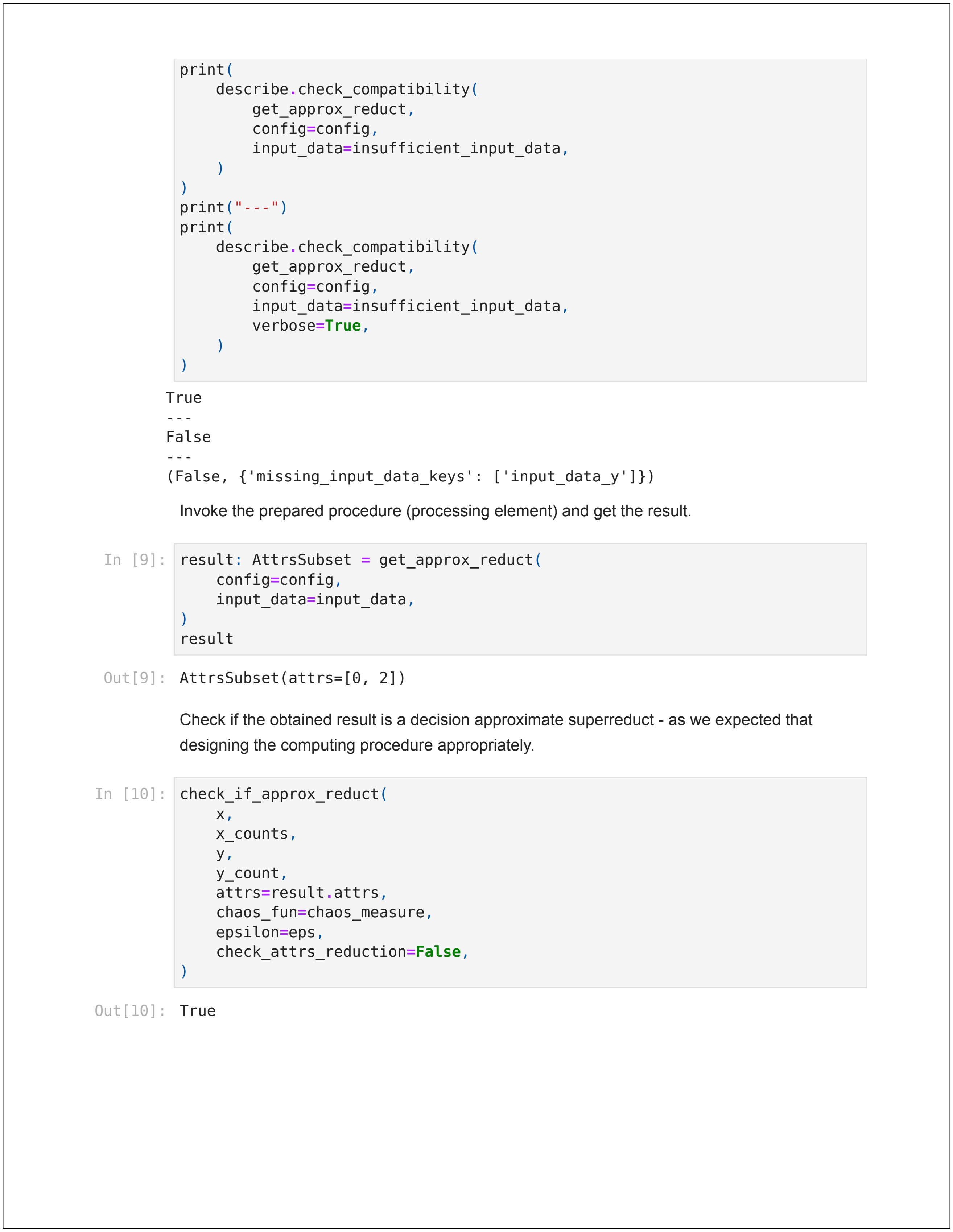
In [7]:

```
eps = 0.4
chaos_measure = entropy
config = {
    CONFIG_CHAOS_FUN: chaos_measure,
    CONFIG_EPSILON: eps,
    CONFIG_SELECT_ATTRS_CHAOS_SCORE_BASED_MAX_COUNT: 1,
}
input_data = {
    INPUT_DATA_X: x,
    INPUT_DATA_Y: y,
}
```

Sometimes it may be convenient to check if the given config and input data contain necessary keys, appropriate for the processing element/algorithm. Currently, the feature is limited to the presence of the appropriate key names (declared for the processing element and its descendant subelements).

In [8]:

```
print(
    describe.check_compatibility(
        get_approx_reduct, config=config, input_data=input_data
    )
)
print("---")
insufficient_input_data = {
    INPUT_DATA_X: x,
}
```



```
print(
    describe.check_compatibility(
        get_approx_reduct,
        config=config,
        input_data=insufficient_input_data,
    )
)
print("---")
print(
    describe.check_compatibility(
        get_approx_reduct,
        config=config,
        input_data=insufficient_input_data,
        verbose=True,
    )
)
```

```
True
---
False
---
(False, {'missing_input_data_keys': ['input_data_y']})
```

Invoke the prepared procedure (processing element) and get the result.

```
In [9]: result: AttrsSubset = get_approx_reduct(
            config=config,
            input_data=input_data,
        )
        result
```

```
Out[9]: AttrsSubset(attrs=[0, 2])
```

Check if the obtained result is a decision approximate superreduct - as we expected that designing the computing procedure appropriately.

```
In [10]: check_if_approx_reduct(
             x,
             x_counts,
             y,
             y_count,
             attrs=result.attrs,
             chaos_fun=chaos_measure,
             epsilon=eps,
             check_attrs_reduction=False,
         )
```

```
Out[10]: True
```