Classifier Monitoring using Statistical Tests

Rafał Latkowski^{1,2} and Cezary Głowiński¹

¹ SAS Institute,
ul. Gdańska 27/31, 01-633 Warszawa, Poland,
Cezary.Glowinski@spl.sas.com

² Warsaw University, Institute of Computer Science,
ul. Banacha 2, 02-097 Warszawa, Poland,
R.Latkowski@mimuw.edu.pl

Summary. This paper is addressed to methods for early detection of classifier fall-down phenomenon, what gives a possibility to react in advance and avoid making incorrect decisions. For many applications it is very essential that decisions made by machine learning algorithms were as accurate as it is possible. The proposed approach consists in applying a monitoring mechanism only to results of classification, what not cause an additional computational overhead. The empirical evaluation of monitoring method is presented based on data extracted from simulated robotic soccer as an example of autonomous agent domain and synthetic data that stands for standard industrial application.

1.1 Introduction

The achievements of machine learning make it possible to apply it to many areas. Predictive models and built with their help classifiers not only enable us to create autonomous agents, but are commonly used also in business and industry. It is very essential that decisions made by machine learning algorithms were as accurate as it is possible. In other case they cannot achieve the expected targets, wherever applied: to marketing, to industry or in autonomous systems. Generally speaking the correctness of the decision making strictly depends on the accuracy of applied classifier. Obviously, the accuracy of the classifier is measured during the training phase. While creating the predictive model we select for deployment the model that achieves the highest accuracy and stability measured over prepared test data sets. Such verification is not possible during the productive life cycle of classifier, when it is applied to the real data gathered in dynamic and nondeterministic environment. The question that arises from such a situation is how we can trust the results of classifier? The first phenomenon that makes it doubtful to trust the classifier is that every natural process is evolving in time, e.g., customers are learning other offer and products, machines are changing their physical parameters

and autonomous agents learn new strategies, what is frequently described as “concept drift” (see, e.g., [7]). It is known fact, that the classification results are continuously getting weaker and such a process is called ageing of the model. Usually the process of model ageing is slow and the reporting is employed to identify it in a posteriori process, when the actual decision is known. The actual value of the decision is known not exactly at the same point of time when the classification is made, but dependently on the application, from fraction of second up to several months after the classification. The second phenomenon is sudden change of process of the revolutionary character, e.g., introduction of completely new product on market, machine failure or reprogramming autonomous agent with new meta-strategy of learning. The sudden classifier ageing or classifier fall-down phenomenon can be a consequence of many circumstances, even errors or changes in data preprocessing. It is a very dangerous phenomenon because it result in making wrong decision for a period of time (a couple of months in worst case), what can result in severe losses.

To better express the necessity of the classifier monitoring let take some examples. The first example is related to autonomous agents. The open research community concentrated on the robotic soccer and RoboCup world championships has an aim to compete by the 2050 a human team of soccer players with a team of autonomous humanoid robots (see, e.g., [4]). Many research groups build software simulators or hardware robots for achieve this goal. Such an artificial soccer player should have special classifier that recognizes the strategy of opponent. This classifier can be misled by opponent that is completely reprogrammed or comes from newly created team. In such a situation classifier fall-down phenomenon can result in losing the game. The second example comes from business application. The telecommunication operators collect a lot of data on their customers. This data is used, e.g., to avoid the customer resignations by predicting them in advance. Such systems for customer retention are suffering from classifier fall-down phenomenon, e.g., when completely new categories of products are introduced. With false prediction the marketing campaigns are directed not to the desired target group. In this case reduced accuracy results in measurable losses even comparing to the case without classifier at all.

This paper is addressed to methods for early detection of classifier fall-down phenomenon, what gives a possibility to react in advance and avoid making incorrect decisions. The proposed method consists in applying a monitoring mechanism only to results of classification, what not cause an additional computational overhead.

The paper is organized as follows. In next Section the classifier monitoring method is described. Section 3 provides empirical evaluation with detailed description of the data sets and experiments. Section 4 contain final conclusions and remarks.

1.2 Method description

1.2.1 Motivation

The initial idea on how to monitor a classifier could be checking the distributions of variables that are used to make the decision (predictors). In such an approach all variables are independently tested before classification is performed. This approach can be applied only to the cases, when the distribution of one variable is significantly different in training and test set. If the distribution is changing on more than one variable than even insignificant changes on one variable can result in classifier fall-down. The proposed here approach is free from such deficiencies because it consist in testing the classifier answer. There is also another common situation that results in classifier fall-down. Training data used to build the model does not cover the full scope of universe, because, even when universe is finite, it is enormous huge. We believe that inductive learning find the proper generalization of presented facts. However, in real applications the classification of objects very far from presented ones in training phase results in pure accuracy. The one-variable test can easily do not capture such a situation. There were proposed some solutions to this problem (see, e.g., [6]), but they assume monitoring the object space by nearest neighbor methods or neural networks. These algorithms require additional computational effort comparable to the cost of creating classifier itself. Our approach require only a linearly proportional time to the number of objects in test and training set.

1.2.2 Classifier Monitoring

The proposed approach consists in applying a monitoring mechanism only to results of classification. The classifier monitoring compares the distribution of answers on data set used for training with distribution of answers on data set currently being classified. If the applied test shows the significant difference, than it is a signal to perform detailed checking of classifier and, e.g., build new model.

There are a number of statistical tests for comparing different properties of one, two or a number of distributions. In this research we utilize nonparametric statistical tests and we do not assume any particular distribution. Only several statistical tests satisfy such a conditions, in particular: Wilcoxon rank sum test (equivalent to the Mann-Whitney test) and Kolmogorov-Smirnov test (see, e.g., [2, 3, 5]). These tests detect the differences in location and shape of two distributions. The Wilcoxon and Kolmogorov-Smirnov tests have the advantage of making no assumption about the distribution of data, i.e., they are non-parametric and distribution free.

The result of classification process usually can be of two types. The simpler type is one-valued decision that assigns classified object to a particular decision class. The more expressive result of classification is the probability

Fig. 1.1. The procedure of classifier monitoring applies a statistical test to results of classification.

vector that assigns to each possible decision a predicted probability that classified object belongs to considered decision class. For our research we use the second type of answer, what gives more detailed information on how model works on provided data.

The classification or prediction process frequently proceed in bunches or in data streams, where not one object is classified, but whole set of objects. Such a situation occurs when we are performing stand-alone tests on previously prepared data or classification (prediction) is performed for, e.g., total base of customers. The result of classification is then a set of answers, i.e., probability assignments. In this paper we are limited to the binary decision — yes or no, what corresponds to classification that object belongs to a concept versus classification that object does not belong to a concept. The procedure of classifier monitoring is following (cf. Fig. 1.1):

1. Let $\mathbb{C}$ is a classifier, $T = \{t_1, \dots, t_n\}$ is data set used for training and $P = \{p_1, \dots, p_m\}$ is new data set, being currently classified.
2. Select one decision class d for which the probability assignments will be considered. From now on we will assume, that $\mathbb{C}_{|d} : U \rightarrow [0, 1]$ gives an probability assignment that an object x belongs to decision class d with probability $\mathbb{C}_{|d}(x) = s$.
3. Prepare set of probability assignment S_T , called *scoring*, for data set used for training T . The set $S_T = \{s_1^T, \dots, s_n^T\}$ consist of all answers of classifier $\mathbb{C}$, such that $\mathbb{C}_{|d}(t_i) = s_i^T$.
4. Prepare set of probability assignment (scoring) S_P for data set used for testing P . The set $S_P = \{s_1^P, \dots, s_m^P\}$ consist of all answers of classifier $\mathbb{C}$, such that $\mathbb{C}_{|d}(p_i) = s_i^P$. Scoring S_P can be computed without knowing the actual decision value, so also before gathering the data on decision.
5. Perform statistical test on S_T and S_P that compares whether changes in classification process are significant or not. If the test value exceed a specified threshold, than notify of potential classifier fall-down.

1.2.3 Classifier Fall-Down Identification

The proposed approach for classifier monitoring consists in comparing two scorings: for training data and for currently classified data. There are several issues on proper classifier fall-down identification using this approach.

The empirical evaluation presented further shows, that not all statistical tests are applicable to this problem, even in spite of satisfying requirements, e.g., that a test is model free. Besides presented here method of classifier monitoring we evaluated also another approach that compares not the scorings, but the distributions of tested objects to final leaves of decision tree. However, in such an approach we found no test or measure that correctly recognizes the classifier fall-down phenomenon. The Wilcoxon signed rank test, cosine measure, Kullback-Leibner divergence measure or six-sigma rule either do not capture the classifier fall-down or notify of nonexistent one. We suspect that the problem with those measures comes from the fact, that they do not consider the actual score value s that is assign to each decision tree leaf. If we consider the Kolmogorov-Smirnov test on two scorings, then this test depends not only on distribution of objects to decision tree leaves but also on the actual score value in each leaf. The *empirical distribution function* (EDF) of scoring, which is used to calculate the KS-test, can be fully determined form distribution of objects to leaves combining with leaf score value. Perhaps other measures that take into consideration also the score value of leaves can be successfully applied to this problem. In fact the transformation of the Kolmogorov-Smirnov test from EDF to distribution of objects to leaves combining with leaf score values results in reduction of computational complexity of testing and in great compression of the classifier control data that has to be stored.

The unresolved issue is how to estimate the optimal threshold value that delimitate predicted acceptable classifier accuracy from accuracy fall-down. Even if we precise the border between acceptable and unacceptable classifier accuracy it is unknown how to estimate this threshold. In our research we are familiar with considered data and classifier properties, so the threshold can be determined based on an expert experience. However, we do not have a general answer on how to estimate the threshold for proposed statistical tests.

The proposed classifier monitoring is able to detect the accuracy fall-down only if there are some differences in description of classified objects. We can imagine another situation, where all object descriptions are untouched, but the concept is changing itself. In spite of that such a case is unobserved in real applications, it is possible to, e.g., generate the same synthetic data but with other concept labeling, where differences are only in decision attribute (target variable). There is no method at all to identify that prior knowing the actual decision (concept), while it touches the problem of learning the proper concept itself. In particular the proposed method of classifier monitoring is not able to recognize such a situation.

Table 1.1. The results of experiments with synthetic data, where decision tree classifier was induced for first data set.

Data Set	Accuracy	Error rate	Standardized Wilcoxon Statistic	P-value Wilcoxon Test	Kolmogorov- Smirnov Statistic
1	83.83%	16.17%	0	1	0
2	70.83%	29.17%	0.409571	0.682121	0.017434
3	57.20%	42.80%	-0.3174	0.75094	0.031917
4	43.71%	56.29%	0.200541	0.841057	0.037072

1.3 Empirical Evaluation

1.3.1 Data Description

We used two groups of data sets for experimental evaluation of proposed method. The first group is synthesized in such a way that simulates an industrial data mining application. The second group is extracted from the RoboCup World Championship 2003 in soccer simulation league.

The datasets for simulating an industrial application are synthesized. They contain samples from two multinormal distributions in eight dimensional space $[0, 1]^8$. There are four data sets, where the standard deviations are constant, but locations are getting closer in consecutive data sets. Each data set contain about 10000 observations (objects). The data sets from RoboCup domain are extracted from log files of soccer simulator games that held at the finals of RoboCup World Championship 2003. The data contain the overall information about playfield, like position of players or number of executed already actions of each type. Each simulated player on the playfield was manually marked, whether it plays using an offensive strategy (attacker) or defensive strategy (defender or goalie). The data was desymmetrized and transformed to a special form, where each record describes one player at given time point of game. The finally transformed data contains 46 conditional attributes and one decision (target) attribute, namely strategy. There are eight data sets collected from four games with four participating teams, so each team is represented in two data sets. Each data set contain about 70000 observations (objects).

1.3.2 Experiments

We carried out experiments separately for RoboCup domain data sets and synthetic data sets. The experiments were performed using an algorithm for decision tree induction implemented in SAS Enterprise Miner (see, e.g., [1]). The automatically generated scoring code allows storing both, scoring and distribution of leaves.

Table 1.2. The accuracy results of experiments with data sets from RoboCup domain.

Training data set	Test data set							
	TsinghuAeolus Game1	Game4	UvA_Trilearn Game2	Game4	Everest Game2	Game3	Brainstormers03 Game1	Game3
Tsinghu- G1	100%	98.56%	99.03%	97.13%	91.44%	96.53%	94.48%	99.25%
Aeolus G4	97.07%	100%	89.69%	87.96%	99.39%	99.07%	99.34%	95.26%
UvA_- G2	98.26%	99.86%	99.99%	99.59%	99.47%	97.37%	98.81%	98.93%
Trilearn G4	97.14%	90.19%	98.13%	100%	76.84%	76.4%	78.28%	96.11%
Everest G2	97.61%	100%	89.91%	89.28%	100%	98.66%	98.63%	96.12%
G3	99.36%	98.98%	88.32%	88.1%	99.26%	99.99%	99.25%	93.27%
Brain- G1	36.36%	63.64%	63.64%	72.73%	72.73%	45.45%	100%	100%
stormers G3	36.36%	63.64%	63.64%	72.73%	72.73%	45.45%	100%	100%

Table 1.3. The Kolmogorov-Smirnov statistic results of experiments with data sets from RoboCup domain.

Training data set	Test data set							
	TsinghuAeolus Game1	Game4	UvA_Trilearn Game2	Game4	Everest Game2	Game3	Brainstormers03 Game1	Game3
Tsinghu- G1	0	0.0007	0.0042	0.0143	0.0393	0.0135	0.0037	0.0030
Aeolus G4	0.0147	0	0.0515	0.0602	0.0010	0.0017	0.0027	0.0237
UvA_- G2	0.0090	0.0007	0	0.0017	0.0027	0.0112	0.0060	0.0056
Trilearn G4	0.0143	0.0490	0.0076	0	0.1158	0.1180	0.1086	0.0194
Everest G2	0.0120	0.0001	0.0504	0.0536	0	0.0015	0.0013	0.0194
G3	0.0016	0.0039	0.0583	0.0595	0.0018	0	0.0030	0.0329
Brain- G1	0.0455	0.0909	0.1818	0.1364	0.1364	0.0909	0	0
stormers G3	0.0454	0.0909	0.1817	0.1363	0.1363	0.0909	0	0

The first group of experiments were carried out for synthetic data sets. The decision tree model was induced for the first data set, where centers of two normal distributions are distant. Then the classifier was applied to all four data sets. The classification results were gathered and tested as described in previous section. The results of this experiment are presented in Table 1.1. The first data set was used in both, training and testing. In case of first data set we can observe the highest classification accuracy and obviously no differences detected by statistical tests at all. The consecutive data sets, that contain samples from closer distributions, are worse classified by model induced for the first data set. The Wilcoxon statistic does not capture the essential classifier fall-down that occur for third and fourth data set. In the case of Kolmogorov-Smirnov statistic we can easily observe that first and second data set receive values less than 0.2, while third and fourth on more than 0.2. If we put a threshold at level 0.2, than Kolmogorov-Smirnov statistic perfectly detects the classifier fall-down.

The experiments for data sets from RoboCup domain were performed differently. The model for predicting strategy was built for each data set. Each classifier was applied to all data sets. There are eight data sets, so also eight models were induced. In total $8 \cdot 8 = 64$ experiments were carried out to cover all combinations. Such a proceeding simulates a strategy detection classifier that is faced to unknown team or known team but in other game.

The results of classification accuracy are presented in Table 1.2. As we expect the diagonal elements, which correspond to classifying the data set on which the model as built, present fully accurate or almost fully accurate classification. The similar observation holds for classifying the same team, for which model was built, but from the other game. The weakest classification accuracy in this category is 97.07% for model built on team TsinghuAeolus in game 4 (final) and tested on game 1 (third level group game). The classification accuracy of other teams varies from 36.36% up to 100%.

The results of Kolmogorov-Smirnov test are presented in Table 1.3. The results presented in this table are almost perfectly correlated to accuracy results. The diagonal elements are obviously equal to zero and classification of the same team gives KS-test value below 0.015. Figure 1.2 presents the same results in graphical form, where experiments are sorted with respect to classification accuracy. It is easy to observe that while the accuracy is decreasing the KS-test value is almost always increasing. If we set the threshold between 0.04 and 0.045 then all 22 worst classification results in range from 36% to 90% are recognized as doubtful. If we set the threshold between 0.061 and 0.09 then the classification accuracy fall-down from level 88% to 78% is correctly recognized except two the worst experiments. It means that 12 out of 14 cases are correctly recognized. The p-value of Wilcoxon rank sum test, presented on Figure 1.2, does not manifest similar properties. The p-value for experiments with 100% classification accuracy is 1.0. However, for other experiments the p-value is extremely variable and is almost zero also for tests with classification accuracy above 90%.

1.4 Conclusions

The empirical evaluation shows that the application of proper statistical test makes it possible to detect the classifier malfunctioning. The experimental results showed that the Kolmogorov-Smirnov test is recommended for detecting the classifier fall-down phenomenon. The proposed method can be applied to monitor any type of classifier under assumption that it generates scoring if form of probability estimation, e.g., probability of belonging to a decision class.

The proposed approach is suitable for detection of classification accuracy fall-down in case of binary classifiers. For other purposes it is necessary to extend the scoring definition in order to apply similar statistical tests or replace the testing technique. The other deficiency of proposed method is lack

Fig. 1.2. The classification accuracy and statistical test results on data from RoboCup domain. The results are sorted by classification accuracy.

of strict guidelines how to determine the proper threshold value and its confidence interval. In our further research we will try to overcome this problem by providing strict estimations on the possible classification accuracy fall-down with respect to the KS-test value.

Although presented experiments were carried out using decision tree induction algorithm, there is no obstacle to apply this method to other classifiers, e.g., based on decision rules or artificial neural networks. The proposed method of classifier monitoring is applicable to classifiers induced by any algorithm. The only requirement is the availability of scoring or similar probability-like values that are produced by classifier.

References

1. *Data Mining Using SAS Enterprise Miner: A Case Study Approach, Second Edition*. SAS Publishing (2003)
2. Conover W.J.: *Practical Nonparametric Statistics, Second Edition*. John Wiley & Sons (1980)
3. Hollander M., Wolfe D. A.: *Nonparametric statistical inference*. John Wiley & Sons (1973)
4. Kaminka G. A., Lima P. U., Rojas R.: *RoboCup 2002: Robot Soccer World Cup VI*. LNCS 2752. Springer (2003)
5. Koronacki J., Mielniczuk J.: *Statystyka dla studentów kierunków technicznych i przyrodniczych*. WNT (2001)
6. Liu Y., Menzies T., Cukic B.: Data Sniffing — Monitoring of Machine Learning for Online Adaptive Systems. In *14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'02)*. IEEE (2002)
7. Freund Y., Mansour Y.: Learning under persistent drift. In S. Ben-David, editor, *Proceedings of the EuroCOLT'97*. LNCS 1208, 94–108. Springer (1997)