

# SZTUCZNA INTELIGENCJA I SYSTEMY DORADCZE

## UCZENIE MASZYNOWE - SYSTEMY REGULOWE

# Reguły

## Warunek

Koniunkcja selektorów, każdy selektor reprezentuje test wartości pojedynczego atrybutu, warunek odpowiada obiektom spełniającym wszystkie selektory

## Decyzja

Każda reguła związana jest z jedną decyzją, przypisywaną obiektom spełniającym warunek reguły

## Przykład

$$Wind = Weak \wedge Temp > 20 \wedge Outlook \neq Rain \Rightarrow PlayTennis = Yes$$

## Reguly: selektory

Atrybuty symboliczne:

- ◇ Selektor równościowy  $X = v$
- ◇ Selektor wykluczający  $X \neq v$
- ◇ Selektor ogólny  $X \in \{v_1, \dots, v_k\}$

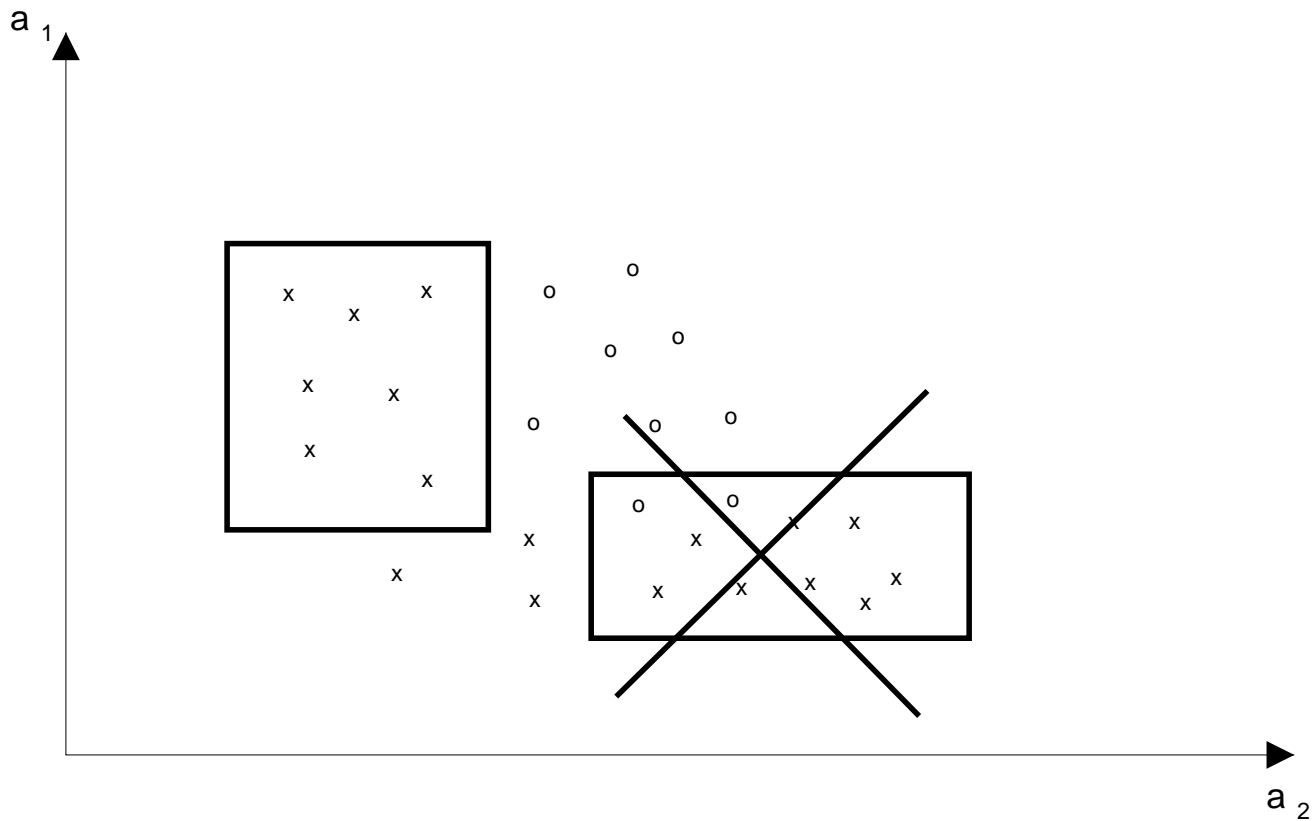
Atrybuty numeryczne:

- ◇ Selektor przedziałowy  $X \in (a, b)$

Przedział może być jednostronnie nieograniczony,  
może też być jedno- lub obustronnie domknięty

# Reguły spójne

Reguła  $\alpha \Rightarrow dec = d$  jest **spójna** ze zbiorem treningowym  $U_{trn}$  jeśli każdy przykład  $x \in U_{trn}$  spełniający warunek  $\alpha$  ma decyzję  $dec(x) = d$



## Reguly spojne: przyklad

<i>Day</i>	<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
<i>D1</i>	<i>Sunny</i>	<i>Hot</i>	<i>High</i>	<i>Weak</i>	<i>No</i>
<i>D2</i>	<i>Sunny</i>	<i>Hot</i>	<i>High</i>	<i>Strong</i>	<i>No</i>
<i>D3</i>	<i>Overcast</i>	<i>Hot</i>	<i>High</i>	<i>Weak</i>	<i>Yes</i>
<i>D4</i>	<i>Rain</i>	<i>Mild</i>	<i>High</i>	<i>Weak</i>	<i>Yes</i>
<i>D5</i>	<i>Rain</i>	<i>Cool</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D6</i>	<i>Rain</i>	<i>Cool</i>	<i>Normal</i>	<i>Strong</i>	<i>No</i>
<i>D7</i>	<i>Overcast</i>	<i>Cool</i>	<i>Normal</i>	<i>Strong</i>	<i>Yes</i>
<i>D8</i>	<i>Sunny</i>	<i>Mild</i>	<i>High</i>	<i>Weak</i>	<i>No</i>
<i>D9</i>	<i>Sunny</i>	<i>Cool</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D10</i>	<i>Rain</i>	<i>Mild</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D11</i>	<i>Sunny</i>	<i>Mild</i>	<i>Normal</i>	<i>Strong</i>	<i>Yes</i>
<i>D12</i>	<i>Overcast</i>	<i>Mild</i>	<i>High</i>	<i>Strong</i>	<i>Yes</i>
<i>D13</i>	<i>Overcast</i>	<i>Hot</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D14</i>	<i>Rain</i>	<i>Mild</i>	<i>High</i>	<i>Strong</i>	<i>No</i>

*Outlook = Overcast ⇒ PlayTennis = Yes??*

## Reguły spojne: przykład

<i>Day</i>	<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
<i>D1</i>	<i>Sunny</i>	<i>Hot</i>	<i>High</i>	<i>Weak</i>	<i>No</i>
<i>D2</i>	<i>Sunny</i>	<i>Hot</i>	<i>High</i>	<i>Strong</i>	<i>No</i>
<i>D3</i>	<i>Overcast</i>	<i>Hot</i>	<i>High</i>	<i>Weak</i>	<i>Yes</i>
<i>D4</i>	<i>Rain</i>	<i>Mild</i>	<i>High</i>	<i>Weak</i>	<i>Yes</i>
<i>D5</i>	<i>Rain</i>	<i>Cool</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D6</i>	<i>Rain</i>	<i>Cool</i>	<i>Normal</i>	<i>Strong</i>	<i>No</i>
<i>D7</i>	<i>Overcast</i>	<i>Cool</i>	<i>Normal</i>	<i>Strong</i>	<i>Yes</i>
<i>D8</i>	<i>Sunny</i>	<i>Mild</i>	<i>High</i>	<i>Weak</i>	<i>No</i>
<i>D9</i>	<i>Sunny</i>	<i>Cool</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D10</i>	<i>Rain</i>	<i>Mild</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D11</i>	<i>Sunny</i>	<i>Mild</i>	<i>Normal</i>	<i>Strong</i>	<i>Yes</i>
<i>D12</i>	<i>Overcast</i>	<i>Mild</i>	<i>High</i>	<i>Strong</i>	<i>Yes</i>
<i>D13</i>	<i>Overcast</i>	<i>Hot</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D14</i>	<i>Rain</i>	<i>Mild</i>	<i>High</i>	<i>Strong</i>	<i>No</i>

*Outlook = Overcast ⇒ PlayTennis = Yes??* spójna

*Humidity = Normal ⇒ PlayTennis = Yes??*

## Reguły spójne: przykład

<i>Day</i>	<i>Outlook</i>	<i>Temperature</i>	<i>Humidity</i>	<i>Wind</i>	<i>PlayTennis</i>
<i>D1</i>	<i>Sunny</i>	<i>Hot</i>	<i>High</i>	<i>Weak</i>	<i>No</i>
<i>D2</i>	<i>Sunny</i>	<i>Hot</i>	<i>High</i>	<i>Strong</i>	<i>No</i>
<i>D3</i>	<i>Overcast</i>	<i>Hot</i>	<i>High</i>	<i>Weak</i>	<i>Yes</i>
<i>D4</i>	<i>Rain</i>	<i>Mild</i>	<i>High</i>	<i>Weak</i>	<i>Yes</i>
<i>D5</i>	<i>Rain</i>	<i>Cool</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D6</i>	<i>Rain</i>	<i>Cool</i>	<i>Normal</i>	<i>Strong</i>	<i>No</i>
<i>D7</i>	<i>Overcast</i>	<i>Cool</i>	<i>Normal</i>	<i>Strong</i>	<i>Yes</i>
<i>D8</i>	<i>Sunny</i>	<i>Mild</i>	<i>High</i>	<i>Weak</i>	<i>No</i>
<i>D9</i>	<i>Sunny</i>	<i>Cool</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D10</i>	<i>Rain</i>	<i>Mild</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D11</i>	<i>Sunny</i>	<i>Mild</i>	<i>Normal</i>	<i>Strong</i>	<i>Yes</i>
<i>D12</i>	<i>Overcast</i>	<i>Mild</i>	<i>High</i>	<i>Strong</i>	<i>Yes</i>
<i>D13</i>	<i>Overcast</i>	<i>Hot</i>	<i>Normal</i>	<i>Weak</i>	<i>Yes</i>
<i>D14</i>	<i>Rain</i>	<i>Mild</i>	<i>High</i>	<i>Strong</i>	<i>No</i>

*Outlook = Overcast*  $\Rightarrow$  *PlayTennis = Yes??* spójna

*Humidity = Normal*  $\Rightarrow$  *PlayTennis = Yes??* niespójna, bo *D6* sprzeczne

# Systemy regulowe

- ◇ CN2 (Clark, Niblett 91)
- ◇ AQ (Michalski 86)
- ◇ Zbiory przybliżone (Skowron, Rauszer 92)
- ◇ C4.5rules (Quinlan 93)

# Systemy regulowe: uczenie i klasyfikacja

## Uczenie

Generowanie zbioru reguł na podstawie zbioru przykładów treningowych

## Klasyfikacja

Wyszukiwane są reguły pasujące do klasyfikowanego obiektu  $x$ , tzn. te, których warunek jest spełniany przez obiekt  $x$ , możliwe są dwie strategie podejmowania decyzji:

### 1. Najlepszy wygrywa:

regułom przypisana jest miara ważności *Importance*, decyzja podejmowana jest na podstawie pasującej do  $x$  reguły  $r$  o najwyższej wartości  $Importance(r)$

### 2. Głosowanie:

reguły mają przypisane wagi *Weight*, obiekt  $x$  klasyfikowany jest decyzją o najwyższej sumie wag reguł pasujących

$$\max \arg_{d_j} \sum_{\alpha \Rightarrow d_j: x \text{ spelnia } \alpha} Weight(\alpha \Rightarrow d_j)$$

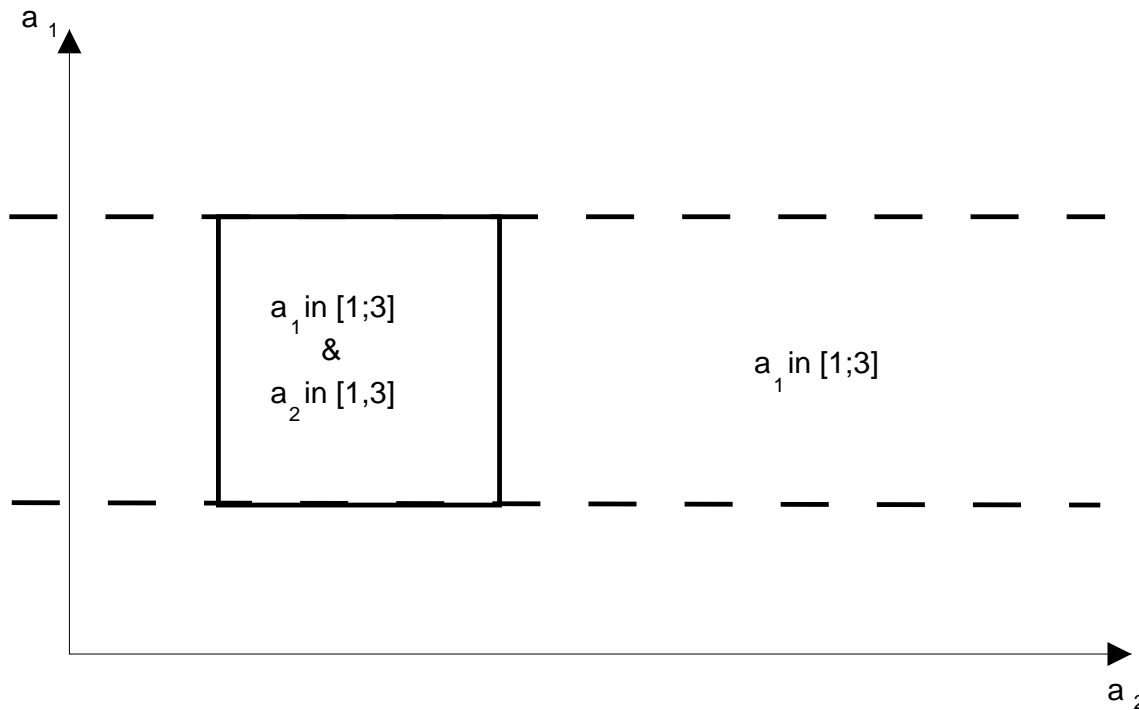
# Generowanie reguł

- ◇ Bezpośrednio ze zbioru przykładów
  - zupełne
  - sekwencyjne pokrywanie (CN2, AQ)
  
- ◇ Przy użyciu struktur pośrednich
  - z reduktu (teoria zbiorów przybliżonych)
  - z drzewa decyzyjnego (C4.5rules)

# Generowanie reguł zupełne

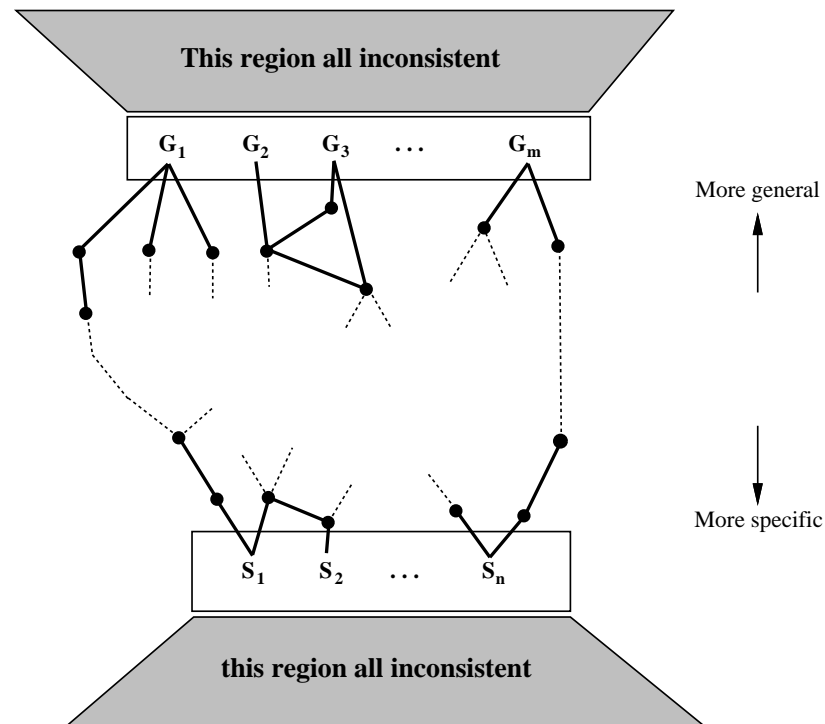
## Fakt

Dla dowolnej reguły  $s_1 \wedge \dots \wedge s_m \Rightarrow d$  wszystkie obiekty rozpoznawane przez nią rozpoznawane są także przez każdą regułę zbudowaną z podzbioru jej selektorów  $s_{i_1} \wedge \dots \wedge s_{i_k} \Rightarrow d$



# Generowanie reguł zupełne

**Wniosek:** Obszar przestrzeni obiektów pokrywany przez wszystkie maksymalnie ogólne reguły spójne ( $G_1, \dots, G_m$ ) jest taki sam jak obszar pokrywany przez wszystkie reguły spójne

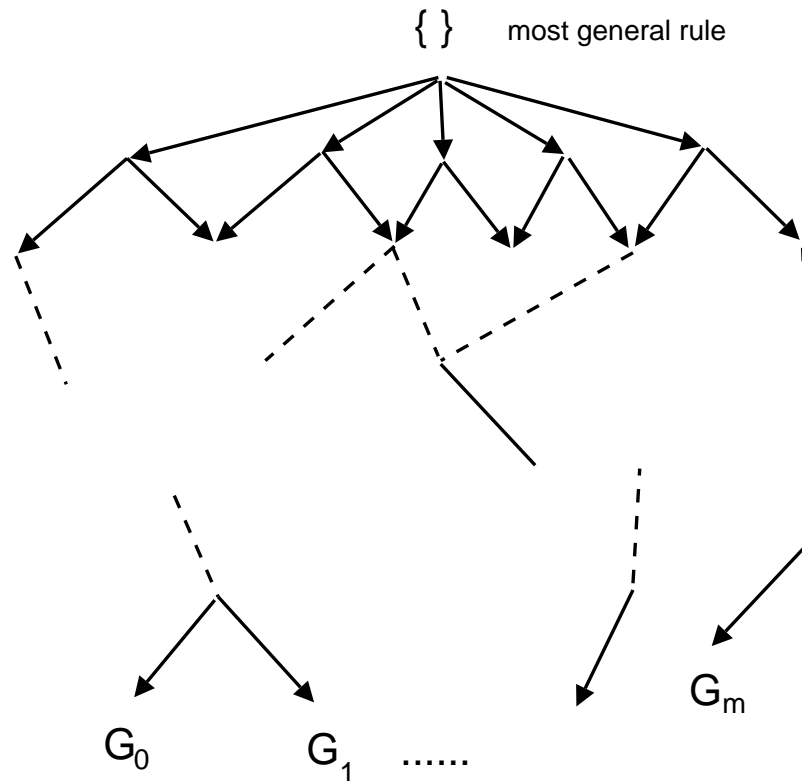


⇒ Wystarczy wyszukać wszystkie reguły spójne o minimalnym zbiorze selektorów, tzn. takim, że usunięcie dowolnego selektora daje regułę niespójną.

# Generowanie reguł zupełne

## Jak to robić??

Można przeszukiwać przestrzeń wszystkich reguł zaczynając od reguł najbardziej ogólnych. Dopóki reguły nie są spójne ze zbiorem treningowym, są rozszerzane o selektory wykluczające przykłady powodujące ich niespójność.



# Generowanie reguł zupełne: algorytm

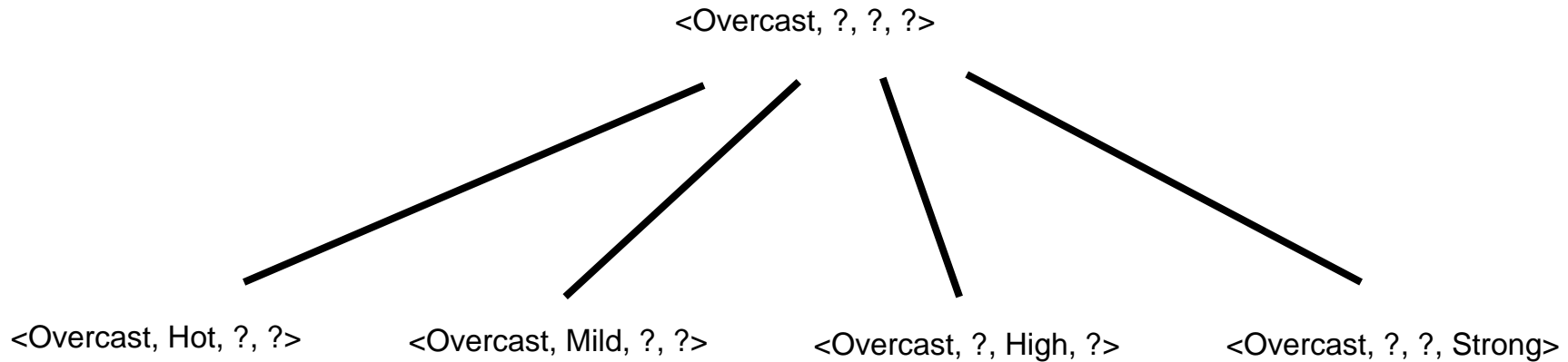
```
function EXHAUSTIVE-RULES(examples, decisions, selectors) returns a rule set  
rules  $\leftarrow$  {}  
for each decision  $d \in$  decisions do  
  candidates  $\leftarrow$  {}  $\Rightarrow$   $d$   
  repeat  
    newCandidates  $\leftarrow$  {}  
    for each candidate rule  $\alpha \Rightarrow d \in$  candidates do  
       $e_{neg} \leftarrow$  a random example matching  $\alpha$  but with a decision  $\neq d$   
      for each selector  $s \in$  selectors excluding  $e_{neg}$  do  
         $r_{new} \leftarrow \alpha \wedge s \Rightarrow d$   
        if  $r_{new}$  covers one or more objects with decision  $d$  in examples  
          and is not subsumed by another rule from  $rules \cup newCandidates$   
          if  $r_{new}$  is consistent with examples then  $rules \leftarrow rules \cup r_{new}$   
          else  $newCandidates \leftarrow newCandidates \cup r_{new}$   
         $candidates \leftarrow newCandidates$   
      until candidates is empty  
  return rules
```

## Generowanie reguł zupełne: przykład

Reguły z decyzją  $PlayTennis = Yes$

*candidates: Outlook = Overcast  $\Rightarrow$  PlayTennis = Yes*

Kontrprzykład:  $\langle Overcast, Cool, Normal, Weak, PlayTennis = No \rangle$



## Generowanie reguł: sekwencyjne pokrywanie

Generowanie reguł zupełne przegląda zazwyczaj wykładniczo dużą podprzestrzeń reguł, w praktyce niewykonalne

**Pomysł (heurystyczny):** Reguły można generować pojedynczo do momentu pokrycia przez nie wszystkich obiektów treningowych

```
function SEQUENTIAL-COVERING(examples) returns a rule set
```

```
rules ← { }
```

```
uncovered ← examples
```

```
repeat
```

```
  r ← LEARN-ONE-RULE(examples, uncovered)
```

```
  rules ← rules ∪ r
```

```
  remove all examples covered by r from uncovered
```

```
until uncovered is empty
```

```
return rules
```

Funkcja LEARN-ONE-RULE wyszukuje heurystycznie jak najlepszą regułę względem pewnej miary jakości reguł

## CN2

Clark, Niblett, 1991

◇ Używa atrybutów symbolicznych

Traktuje wszystkie atrybuty jako symboliczne, atrybuty numeryczne zamieniane są na symboliczne w ten sposób, że zakres wartości każdego atrybutu dzielony jest na równe przedziały, wartości z jednego przedziału zamieniane są na taką samą wartość symboliczną

◇ Używa metody sekwencyjnego pokrywania

Szukanie kolejnej reguły (procedura LEARN-ONE-RULE) podobnie jak generowanie reguł zupełne rozpoczyna od najbardziej ogólnych reguł (warunków) i uszczegóławia je dodając kolejne selektory, ale:

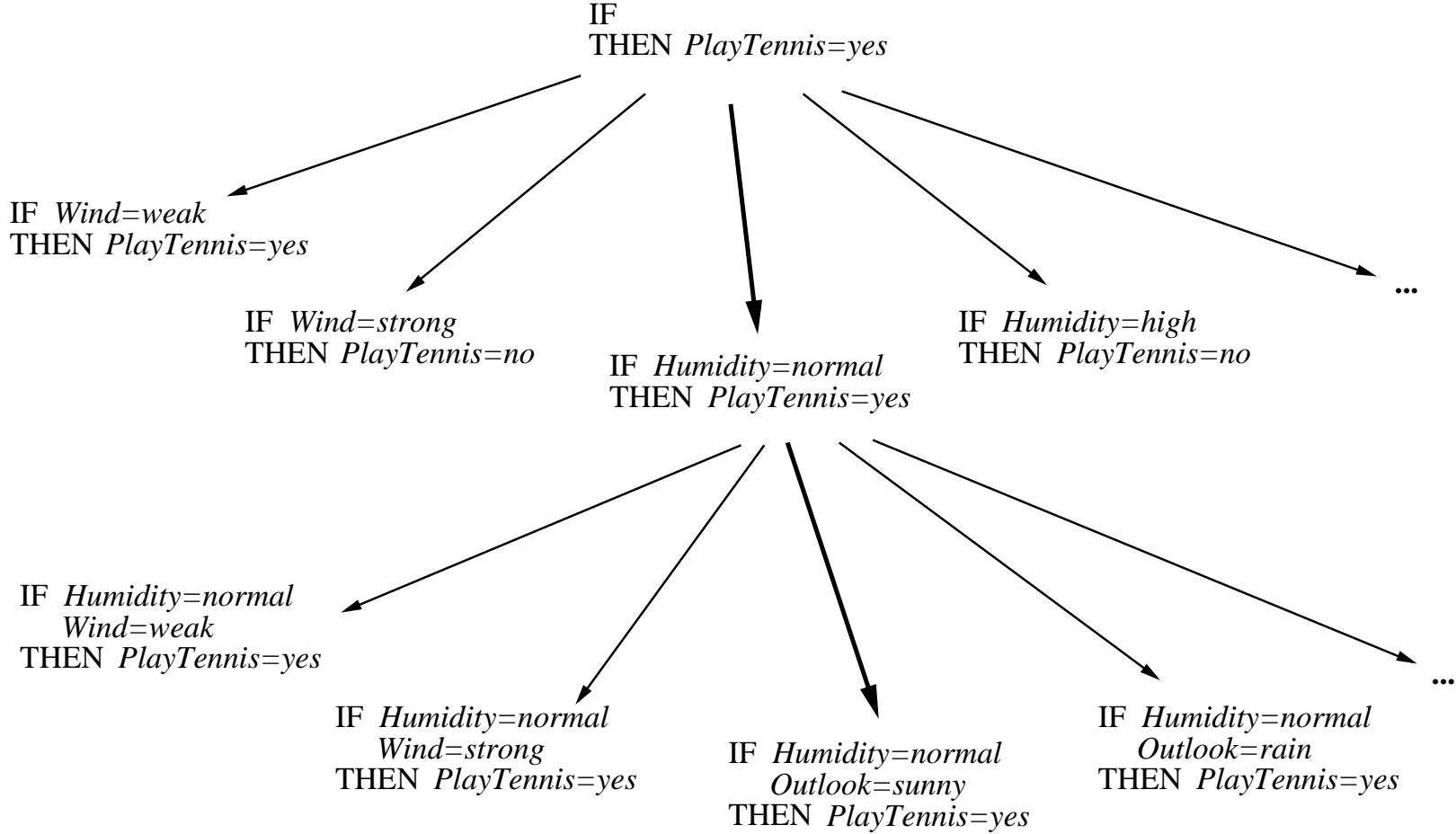
- zbiór reguł-kandydatów ograniczony jest do rozmiaru  $k$  określanego przez użytkownika, do rozszerzania brane są najlepsze kandydujące reguły,
- jako wynik zwracana jest najlepsza reguła spośród wygenerowanych kandydatów

## CN2: szukanie najlepszej reguły

```
function LEARN-ONE-RULE-CN2(uncov, k) returns a rule
  inputs: uncov, the examples not covered by the previous rules
           k, the width of searching
  best ← the most general empty condition
  candidates ← { best }
  repeat
    newCandidates ← { }
    for each candidate  $\alpha \in \text{candidates}$  do
      for each selector s of the form  $a=v$  or  $a \neq v$  consistent with  $\alpha$  do
        if  $\alpha \wedge s \notin \text{candidates} \cup \text{newCandidates}$  then
          newCandidates ← newCandidates  $\cup$  {  $\alpha \wedge s$  }
          if PERFORMANCE( $\alpha \wedge s, \text{uncov}$ ) > PERFORMANCE(best, uncov)
            then best ←  $\alpha \wedge s$ 
      retain only k best candidates in newCandidates according to PERFORMANCE
      candidates ← newCandidates
  until candidates is empty
  return best  $\Rightarrow$  d (the most frequent decision among objects matching best)
```

# CN2: szukanie najlepszej reguły, przykład

Rozmiar zbioru kandydatów = 1  $\Rightarrow$  przeszukiwanie zachłanne



## CN2: miara jakości reguły

Funkcja  $\text{PERFORMANCE}(\alpha, \text{uncov})$  szacuje jakość warunku  $\alpha$  na podstawie dotychczas niepokrytych przykładów  $\text{uncov}$

$n$  — liczba przykładów z  $\text{uncov}$  pasujących do  $\alpha$

$n_d$  — liczba przykładów z  $\text{uncov}$  pasujących do  $\alpha$  z najczęstszą decyzją  $d$

$m$ -estymata prawdopodobieństwa

$$\frac{n_d + mp_d}{n + m}$$

$\langle p_{d_1}, \dots, p_{d_D} \rangle$  — pierwotny rozkład prawdopodobieństwa w danych

$m$  — parametr estymacji

CN2 używa szczególnego przypadku, **estymaty Laplace'a**:

równomierny rozkład pierwotny  $\langle \frac{1}{D}, \dots, \frac{1}{D} \rangle$  i  $m = D$  ( $D$  — liczba decyzji)

$$\frac{n_d + D\frac{1}{D}}{n + D} = \frac{n_d + 1}{n + D}$$

## Inne miary jakości reguły

Funkcja  $\text{PERFORMANCE}(\alpha, uncov)$  szacuje jakość warunku  $\alpha$  na podstawie dotychczas niepokrytych przykładów  $uncov$

$n$  — liczba przykładów z  $uncov$  pasujących do  $\alpha$

$n_d$  — liczba przykładów z  $uncov$  pasujących do  $\alpha$  z najczęstszą decyzją  $d$

Częstość względna

$$\frac{n_d}{n}$$

Negacja entropii

$$\sum_{d_i} \frac{n_{d_i}}{n} \log_2 \frac{n_{d_i}}{n}$$

$n_{d_i}$  — liczba przykładów z  $uncov$  pasujących do  $\alpha$  z decyzją  $d_i$

## Miary jakości reguły: przykład

$\alpha_1$  pokrywa 1000 przykładów z decyzją  $d_1$  i 1 przykład z decyzją  $d_2$

$\alpha_2$  pokrywa 5 przykładów z decyzją  $d_1$  i 0 przykładów z decyzją  $d_2$

$\alpha_3$  pokrywa 1 przykład z decyzją  $d_1$  i 0 przykładów z decyzją  $d_2$

	$\alpha_1 \Rightarrow d_1$	$\alpha_2 \Rightarrow d_1$	$\alpha_3 \Rightarrow d_1$
Częstość względna	99.9%	100%	100%
Negacja entropii	<0	0	0

Częstość względna i negacja entropii faworyzują reguły  $\alpha_2 \Rightarrow d_1$  i  $\alpha_3 \Rightarrow d_1$

Wartości estymaty Laplace'a ( $D = 2$ ):

99.8% dla  $\alpha_1 \Rightarrow d_1$

85.7% dla  $\alpha_2 \Rightarrow d_1$

66.6% dla  $\alpha_3 \Rightarrow d_1$

## CN2: klasyfikacja pierwszy wygrywa

Lista decyzyjna to lista reguł utworzona przez algorytm sekwencyjnego pokrywania uporządkowana w kolejności takiej, w jakiej reguły były generowane, z dodatkową regułą domyślną na końcu

$$\rightarrow R_0 \rightarrow R_1 \rightarrow R_2 \rightarrow \dots \rightarrow R_m \rightarrow Default$$

W CN2:

$R_0$  — reguła wygenerowana ze wszystkich przykładów

$R_1$  — reguła wygenerowana z przykładów niepokrywanych przez  $R_0$

$R_2$  — reguła wygenerowana z przykładów niepokrywanych przez  $R_0, R_1$ , itd.

*Default* — reguła bezwarunkowa zwracająca najczęstszą decyzję w zbiorze treningowym

Obiektowi przypisywana jest decyzja  $d$  z pierwszej reguły  $\alpha \Rightarrow d$  na liście decyzyjnej, której warunek  $\alpha$  pasuje do obiektu.

## CN2: klasyfikacja przez głosowanie reguł

*rules* — zbiór warunków wygenerowany przez algorytm sekwencyjnego pokrywania (bez ustalonych decyzji)

Rozkład decyzyjny warunku  $\alpha \in rules$ :

$$\langle n_1(\alpha), \dots, n_{|D|}(\alpha) \rangle$$

$n_i$  — liczba przykładów z decyzją  $d_i$  spełniających  $\alpha$  w zbiorze *uncov*, tzn. tylko tych przykładów, które nie spełniają żadnego z wcześniej wygenerowanych warunków

Wybór decyzji dla obiektu  $x$  przez sumowanie rozkładów:

$$\max \arg_{d_i} \sum_{\alpha \in rules: x \text{ spełnia } \alpha} n_i(\alpha)$$

# AQ

AQ15, Michalski, 1986

◇ Używa atrybutów symbolicznych i numerycznych

Do atrybutów symbolicznych stosuje selektory równościowe i wykluczające, do atrybutów numerycznych stosuje selektory ograniczające ( $<$ ,  $\leq$ ,  $>$ ,  $\geq$ ).

◇ Używa metody sekwencyjnego pokrywania, ale oddzielnie dla każdej decyzji

Szukanie kolejnej reguły (procedura LEARN-ONE-RULE) podobnie jak w CN2 przebiega od najbardziej ogólnych do bardziej specyficznych reguł, ale:

- przeszukiwanie sterowane jest wybranym przykładem
- reguły-kandydatki poprawiane są tak długo, dopóki nie osiągną warunku spójności ze zbiorem treningowym, reguła najlepsza wybierana jest spośród końcowych reguł spójnych

## AQ: szukanie najlepszej reguły

**function** LEARN-ONE-RULE-AQ(*examples*, *uncov*, *d*, *k*) **returns** a rule

**inputs:** *examples*, all training examples

*uncov*, the examples not covered by the previous rules

*d*, decision of a return rule

*k*, the width of searching

$e_{pos} \leftarrow$  a random example from *uncov* with decision *d*

*candidates*  $\leftarrow$  {the most general empty condition}

**repeat**

$e_{neg} \leftarrow$  example with decision  $\neq d$  covered by one or more conditions in *candidates*  
with the maximum number of values = the corresponding values of  $e_{pos}$

*selectors*  $\leftarrow$  all selectors consistent with  $e_{pos}$  excluding  $e_{neg}$

*candidates*  $\leftarrow$  { $x \wedge s$ :  $x \in candidates$ ,  $y \in selectors$ }

*candidates*  $\leftarrow$  { $x \in candidates$ :  $\neg \exists y \in candidates$  more general than  $x$ }

retain only *k* best candidates in *candidates* according to PERFORMANCE

**until** *candidates* cover no examples with decision  $\neq d$

*best*  $\leftarrow$  the best condition in *candidates* according to PERFORMANCE

**return** *best*  $\Rightarrow d$

## AQ: miara jakości reguły

$$\text{PERFORMANCE}(\alpha \Rightarrow d, \text{examples}) = \text{pos}_{\text{included}} + \text{neg}_{\text{excluded}}$$

$\text{pos}_{\text{included}}$  – liczba przykładów w *examples* z decyzją  $d$  pasujących do warunku  $\alpha$

    tzw. **wsparcie** reguły

$\text{neg}_{\text{excluded}}$  – liczba przykładów w *examples* z decyzją  $\neq d$  wykluczanych przez warunek  $\alpha$

### Uwaga

Jeśli reguła jest spójna ze zbiorem treningowym, tzn. warunek reguły wyklucza wszystkie przykłady z decyzją  $\neq d$ , to miary jakości reguły jest równa wsparciu reguły

$$\text{PERFORMANCE}(\alpha \Rightarrow d, \text{examples}) = \text{pos}_{\text{included}}$$

## AQ: klasyfikacja

Klasyfikacja przez głosowanie reguł

Waga pojedynczej reguły:

$$Weight(\alpha \Rightarrow d) = \frac{|pos_{included}(\alpha \Rightarrow d)|}{|examples|}$$

Wybór decyzji dla obiektu  $x$ :

$$\max \arg_d \sum_{\alpha \Rightarrow d: x \text{ spelnia } \alpha} \frac{|pos_{included}(\alpha \Rightarrow d)|}{|examples|}$$

# Generowanie reguł: CN2 vs AQ

## Cechy wspólne

- ◇ metoda sekwencyjnego pokrywania
- ◇ szukanie pojedynczej reguły:
  - metoda pierwszy najlepszy ustalonej szerokości
  - od najbardziej ogólnych w kierunku bardziej specyficznych

## Różnice

	Przeszukiwanie sterowane	Wymaganie spójności	Miara jakości reguł
CN2	całym zbiorem	NIE	estymata Laplace'a
AQ	pojedynczym przykładem	TAK	wsparcie

# Teoria zbiorów przybliżonych

◇ Zbiory przybliżone (Pawlak, 1981)

◇ Redukty i reguły generowane z reduktów (Skowron, Rauszer, 1992)

$A = \{a_1, \dots, a_n\}$  — zbiór cech (atrybutów) opisujących przykłady

$U_{trn}$  — zbiór przykładów opisanych wektorami wartości cech  $\langle x_1, \dots, x_n \rangle$

## Definicja

Zbiór atrybutów  $R \subseteq A$  jest **reduktem** dla zbioru przykładów  $U_{trn}$ , jeśli

— dla każdej pary przykładów  $x, y \in U_{trn}$

o różnych decyzjach  $dec(x) \neq dec(y)$

istnieje  $a_i \in R$  rozróżniający tę parę przykładów:  $x_i \neq y_i$

—  $R$  jest minimalnym zbiorem mającym powyższą własność,  
tzn. dla dowolnego  $R' \subset R$  istnieje para przykładów w  $U_{trn}$

o różnych decyzjach i takich samych wartościach

na wszystkich atrybutach  $a_i \in R'$

# Redukty

## Definicja

Redukt  $R$  jest minimalny, jeśli zawiera najmniejszą możliwą liczbę atrybutów, tzn. dla każdego reduktu  $R'$ :  $|R| \leq |R'|$

**Fakt:** Problem znalezienia minimalnego reduktu jest NP-trudny

# Redukty

## Definicja

Redukt  $R$  jest minimalny, jeśli zawiera najmniejszą możliwą liczbę atrybutów, tzn. dla każdego reduktu  $R'$ :  $|R| \leq |R'|$

Fakt: Problem znalezienia minimalnego reduktu jest NP-trudny

Przykład:

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

Redukty??

# Redukty

## Definicja

Redukt  $R$  jest minimalny, jeśli zawiera najmniejszą możliwą liczbę atrybutów, tzn. dla każdego reduktu  $R'$ :  $|R| \leq |R'|$

Fakt: Problem znalezienia minimalnego reduktu jest NP-trudny

Przykład:

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

Redukty??

$\{a, d\}, \{b, c, d\}$

Redukty minimalne??

# Redukty

## Definicja

Redukt  $R$  jest minimalny, jeśli zawiera najmniejszą możliwą liczbę atrybutów, tzn. dla każdego reduktu  $R'$ :  $|R| \leq |R'|$

Fakt: Problem znalezienia minimalnego reduktu jest NP-trudny

Przykład:

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

Redukty??

$\{a, d\}, \{b, c, d\}$

Redukty minimalne??

$\{a, d\}$

## Generowanie reguł z reduktu

$$Rules(R) := \left\{ \bigwedge_{a_i \in R} a_i = x_i \Rightarrow dec = dec(x) : x \in U_{trn} \right\}$$

## Generowanie reguł z reduktu

$$Rules(R) := \left\{ \bigwedge_{a_i \in R} a_i = x_i \Rightarrow dec = dec(x) : x \in U_{trn} \right\}$$

Przykład:

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

$$R = \{b, c, d\}$$

Reguły??

## Generowanie reguł z reduktu

$$Rules(R) := \left\{ \bigwedge_{a_i \in R} a_i = x_i \Rightarrow dec = dec(x) : x \in U_{trn} \right\}$$

Przykład:

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

$$R = \{b, c, d\}$$

Reguły??

$$b = 2 \wedge c = 1 \wedge d = 0 \Rightarrow dec = 0$$

$$b = 2 \wedge c = 2 \wedge d = 1 \Rightarrow dec = 0$$

$$b = 0 \wedge c = 2 \wedge d = 1 \Rightarrow dec = 1$$

$$b = 2 \wedge c = 1 \wedge d = 1 \Rightarrow dec = 2$$

## Skracanie reguł z reduktu

Skracanie reguły polega na odrzuceniu niektórych selektorów z warunku reguły

### Metoda

Reguła  $\alpha \wedge s \Rightarrow dec = d$  może zostać zastąpiona przez  $\alpha \Rightarrow dec = d$ , jeśli  $\alpha \Rightarrow dec = d$  pozostaje spójna ze zbiorem treningowym

### Fakt

Może się zdarzyć, że różne reguły z tą samą decyzją zostaną skrócone do tej samej postaci

$\Rightarrow$  zbiór reguł po skróceniu może być mniejszy niż oryginalny

## Skracanie reguł z reduktu: przykład

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

$$b = 2 \wedge c = 1 \wedge d = 0 \Rightarrow dec = 0$$

$$b = 2 \wedge c = 2 \wedge d = 1 \Rightarrow dec = 0$$

$$b = 0 \wedge c = 2 \wedge d = 1 \Rightarrow dec = 1$$

$$b = 2 \wedge c = 1 \wedge d = 1 \Rightarrow dec = 2$$

## Skracanie reguł z reduktu: przykład

	a	b	c	d	dec
$x_1$	0	2	1	0	0
$x_2$	1	2	2	1	0
$x_3$	2	0	2	1	1
$x_4$	0	2	1	1	2

$$b = 2 \wedge c = 1 \wedge d = 0 \Rightarrow dec = 0$$

$$b = 2 \wedge c = 2 \wedge d = 1 \Rightarrow dec = 0$$

$$b = 0 \wedge c = 2 \wedge d = 1 \Rightarrow dec = 1$$

$$b = 2 \wedge c = 1 \wedge d = 1 \Rightarrow dec = 2$$

Po skróceniu:

$$b = 2 \wedge d = 0 \Rightarrow dec = 0 \text{ lub } c = 1 \wedge d = 0 \Rightarrow dec = 0$$

$$b = 2 \wedge c = 2 \Rightarrow dec = 0$$

$$b = 0 \Rightarrow dec = 1$$

$$c = 1 \wedge d = 1 \Rightarrow dec = 2$$

# Klasyfikacja oparta na wsparciu

$rules$  — zbiór reguł z jednoznaczną decyzją

$U_{trn}$  — zbiór przykładów treningowych

$x$  - obiekt do klasyfikacji

Klasyfikacja przez **maksymalizację wsparcia**:

$$rules(x) = \{\alpha \Rightarrow d \in rules : x \text{ spełnia } \alpha\}$$

$$\max \arg_d |\{y \in U_{trn} : \exists \alpha \Rightarrow d \in rules(x) (y \text{ spełnia } \alpha \wedge dec(y) = d)\}|$$

## Redukty lokalne

Zbiór atrybutów  $R \subseteq A$  jest **reduktem lokalnym** dla przykładu  $x \in U_{trn}$  w zbiorze przykładów  $U_{trn}$ , jeśli

- dla każdego przykładu  $y \in U_{trn}$  z inną decyzją  $dec(y) \neq dec(x)$  istnieje  $a_i \in R$  rozróżniający  $x$  od  $y$ :  $x_i \neq y_i$
- $R$  jest minimalnym zbiorem mającym powyższą własność, tzn. dla dowolnego  $R' \subset R$  istnieje przykład w  $U_{trn}$  z inną decyzją i wartościami taki samymi jak  $x$  na wszystkich atrybutach  $a_i \in R'$

### Fakt 1:

Liczba reduktów lokalnych dla jednego przykładu może być wykładnicza względem liczby atrybutów i liczby przykładów treningowych

### Fakt 2:

Problem znalezienia minimalnego reduktu lokalnego dla danego przykładu jest NP-trudny

## Generowanie reguł z reduktów lokalnych

Reguła generowana z reduktu lokalnego  $R$  dla przykładu  $x$ :

$$\bigwedge_{a_i \in R} a_i = x_i \Rightarrow dec = dec(x)$$

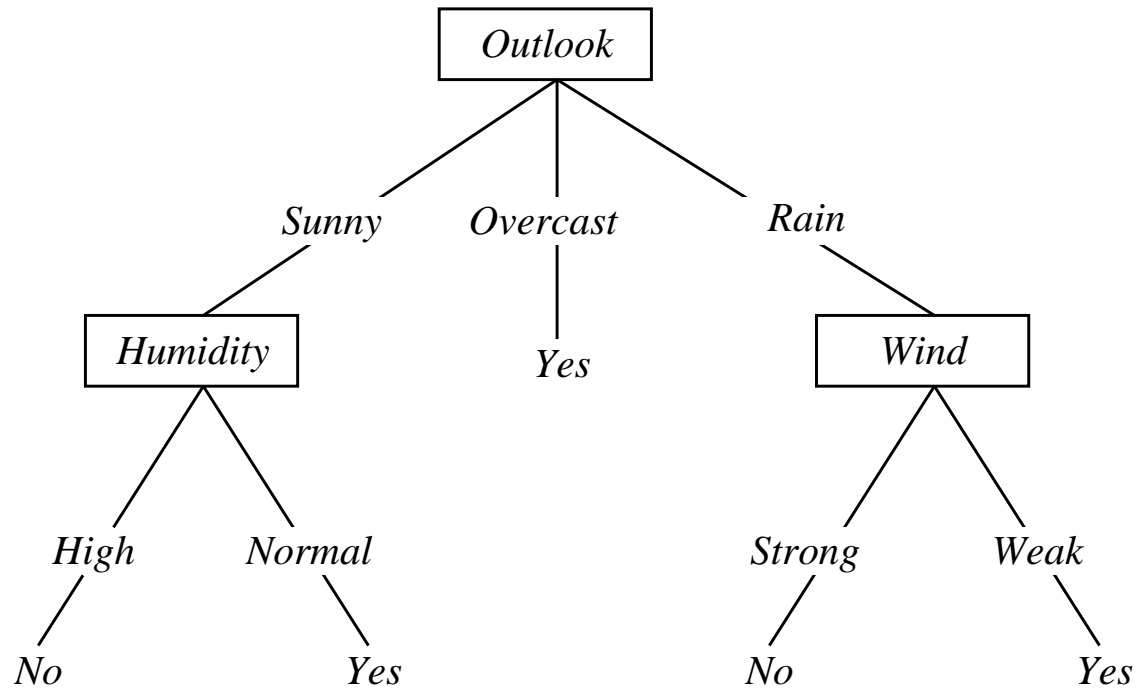
**Fakt 1:** Reguła generowana z reduktu lokalnego jest regułą spójną minimalną (tzn. usunięcie któregoś selektora powoduje utratę spójności)

**Fakt 2:** Zbiór reguł wygenerowanych ze wszystkich reduktów lokalnych = zbiór wszystkich minimalnych reguł spójnych = zbiór wszystkich reguł generowanych przez algorytm zupełny (z selektorami równościowymi)

**Przypomnienie:** Liczba wszystkich minimalnych reguł spójnych może być wykładnicza względem liczby atrybutów i przykładów treningowych

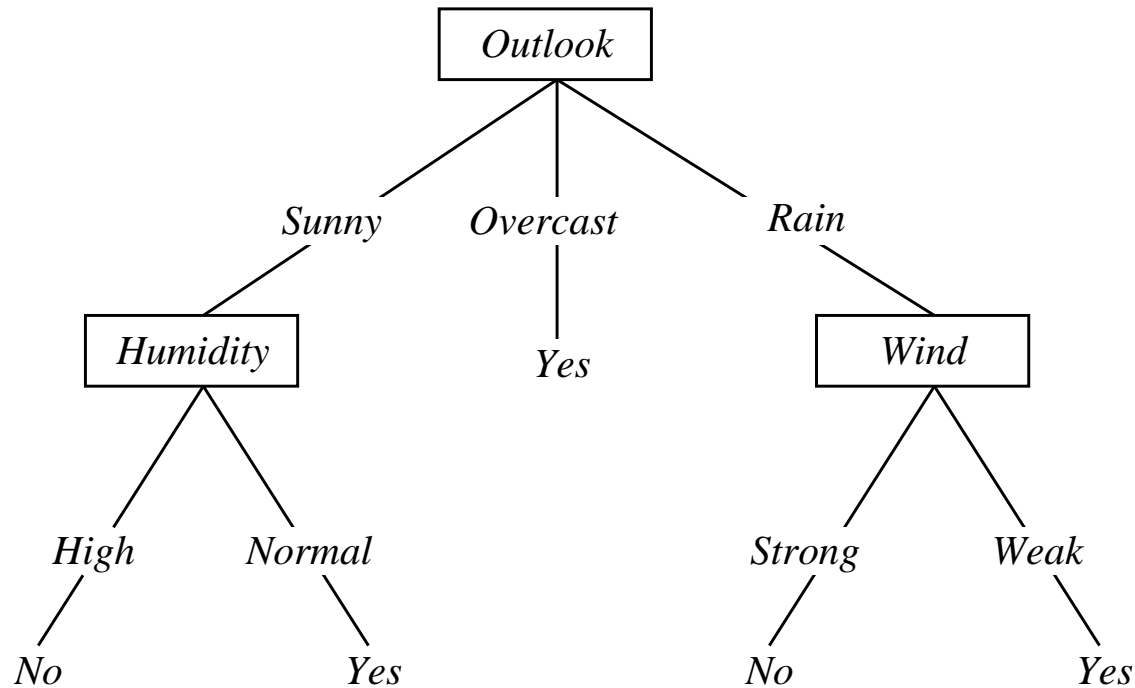
**Fakt 3** (Bazan, 1998): Niech  $rules_{all}$  – zbiór wszystkich minimalnych reguł spójnych. Istnieje algorytm symulujący klasyfikację z maksymalizacją wsparcia w zbiorze reguł  $rules_{all}$  (bez jawnego liczenia reguł) wykonujący klasyfikację pojedynczego obiektu w czasie  $O(|U_{trn}|^2|A|)$ .

## C4.5rules: generowanie reguł



**Pomysł:** mając dane drzewo decyzyjne można generować reguły na podstawie jego struktury

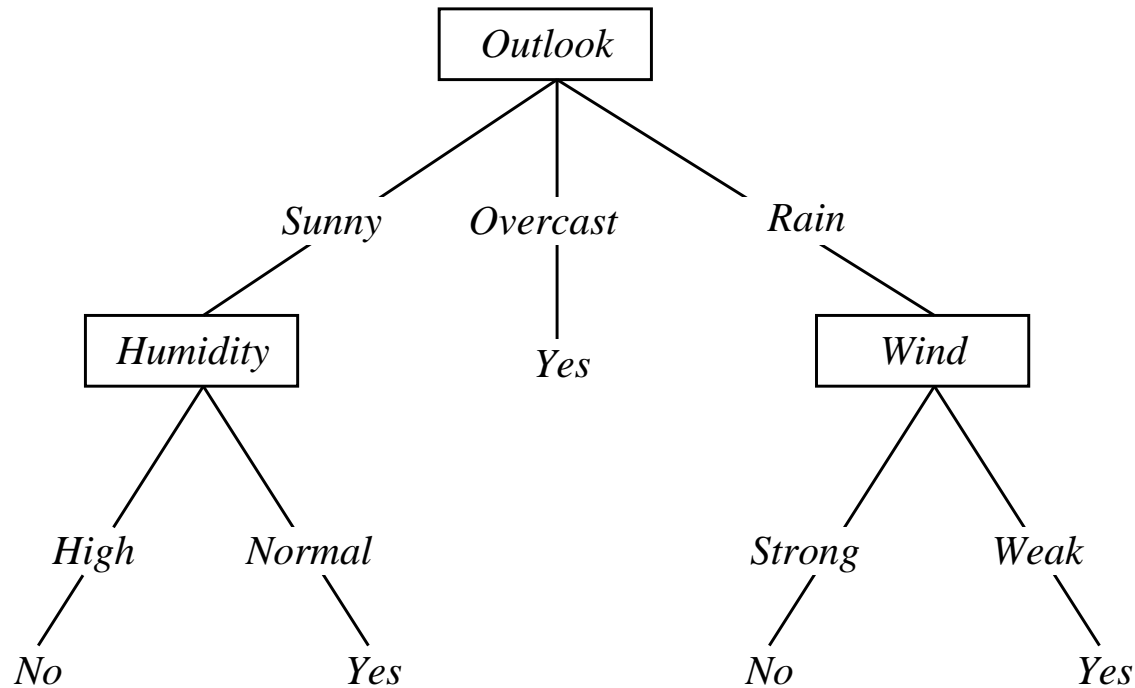
## C4.5rules: generowanie reguł



**Pomysł:** mając dane drzewo decyzyjne można generować reguły na podstawie jego struktury

⇒ Drzewo jest generowane algorytmem C4.5 opisanym na wykładzie o drzewach decyzyjnych

## C4.5rules: przykład



$Outlook = Sunny \wedge Humidity = High \Rightarrow PlayTennis = No$

$Outlook = Sunny \wedge Humidity = Normal \Rightarrow PlayTennis = Yes$

$Outlook = Overcast \Rightarrow PlayTennis = Yes$

$Outlook = Rain \wedge Wind = Strong \Rightarrow PlayTennis = No$

$Outlook = Rain \wedge Wind = Weak \Rightarrow PlayTennis = Yes$

## C4.5rules: skracanie reguł

$\alpha \wedge s \Rightarrow d$  — reguła przed skróceniem

$\alpha \Rightarrow d$  — reguła po skróceniu

C4.5rules wylicza statystyczne górne oszacowanie błędów obu reguł na podstawie przykładów ze zbioru treningowego pokrywanych przez te reguły, i zastępuje regułę  $\alpha \wedge s \Rightarrow d$  regułą skróconą  $\alpha \Rightarrow d$ , jeśli górne oszacowanie błędu dla reguły skróconej jest nie większe niż dla reguły oryginalnej

Reguła może być skrócona wielokrotnie, jeśli usuwanie kolejnych selektorów nie powoduje zwiększenia górnego oszacowania błędu reguły

## C4.5rules: klasyfikacja

### Fakt

Warunki reguł przed skróceniem wykluczały się wzajemnie,  
po skróceniu już nie muszą się wykluczać

### Wniosek

Klasyfikacja wymaga zastosowania wyboru najlepszej reguły  
lub głosowania reguł

⇒ C4.5rules stosuje zaawansowane metody  
do usunięcia niektórych reguł skróconych  
i uporządkowania pozostałych według ważności  
Obiekty klasyfikowane są według najlepszej pasującej reguły