

Hurtownie Danych

Eksploracja danych / Wydobywanie wiedzy

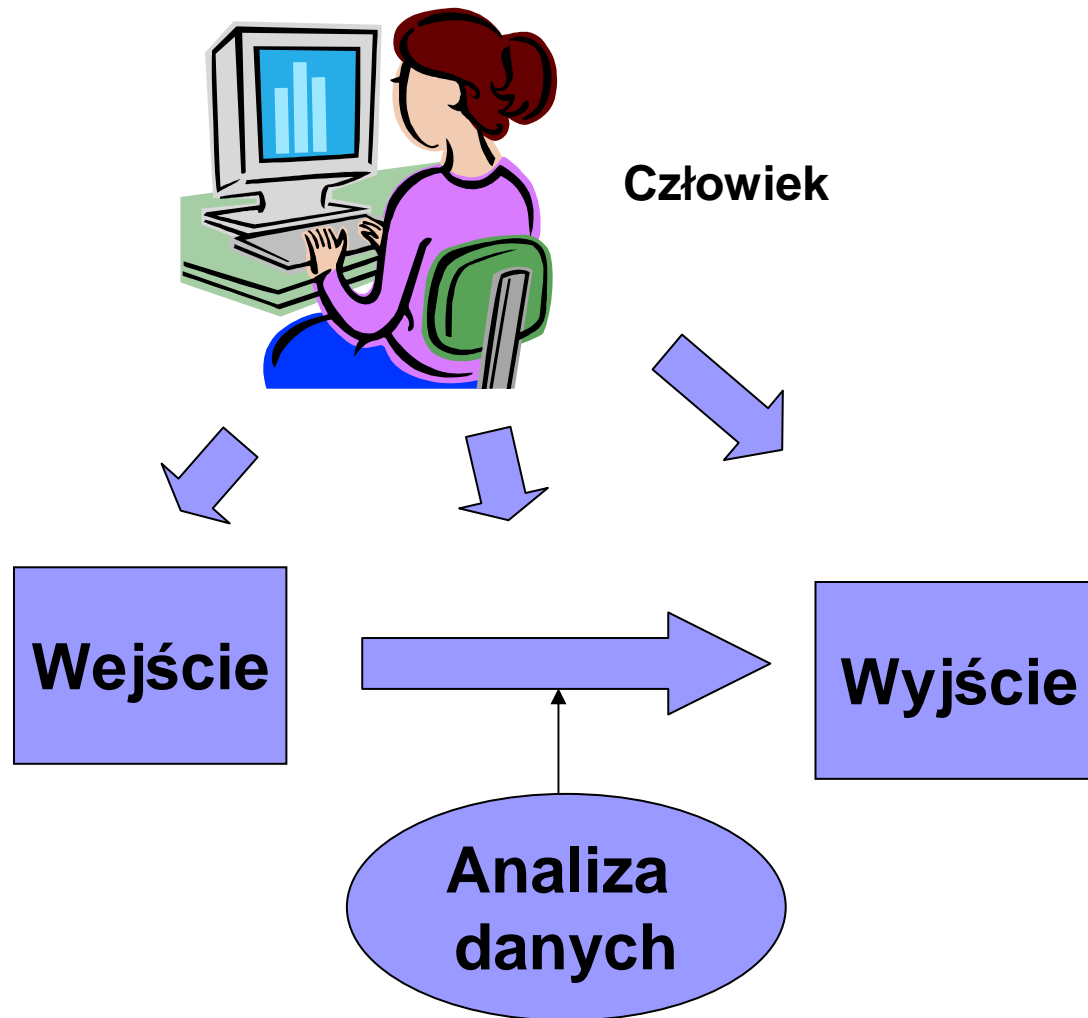
Wojciech Waloszek



Motywacje

- Wzrost powszechności baz danych,
- Hurtownie danych i OLAP,
- Dane jako repozytorium wiedzy mogącej służyć do podejmowania przyszłych decyzji.

Proces eksploracji danych





Metody eksploracji danych

- Oparte na szeroko rozumianej sztucznej inteligencji (*Machine learning*),
- Główne narzędzia:
 - Metody statystyczne,
 - Metody oparte na logice matematycznej,
 - Sztuczne sieci neuronowe



Dane wejściowe

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|-------------|---------------|-------------|----------------------|-------------|-------------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

Dane wejściowe

Wartość

Atrybut

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

Przykład

Zbiór trenujący

Rodzaje atrybutów

■ Nominalne

- Zbiór wartości dyskretnych

| |
|-------------|
| Sam. |
| tak |
| nie |

■ Porządkowe

- Zbiór wartości dyskretnych z relacją porządku

| |
|----------------------|
| Wykształcenie |
| wyższe |
| średnie |
| podstawowe |

■ Numeryczne

- Ciągłe z zakresu liczb rzeczywistych

| |
|-------------|
| Wiek |
| 28 |
| 35 |
| 26 |



Rodzaje klasyfikatorów

- Klasyfikatory decyzyjne
 - Decyzje podejmowane względem wartości jednego atrybutu,
- Klasyfikatory asocjacyjne
 - Odnajdywanie regularności w danych
- Grupowanie danych
 - Odnajdywanie grup podobnych przykładów

Klasyfikator decyzyjny

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. Wyróżniamy pewien atrybut _____
2. Na podstawie jego wartości wyróżniamy *klasy* _____
3. Budujemy zasady przynależności do wyróżnionych klas w zależności od wartości pozostałych atrybutów

Klasyfikator decyzyjny - reguły

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

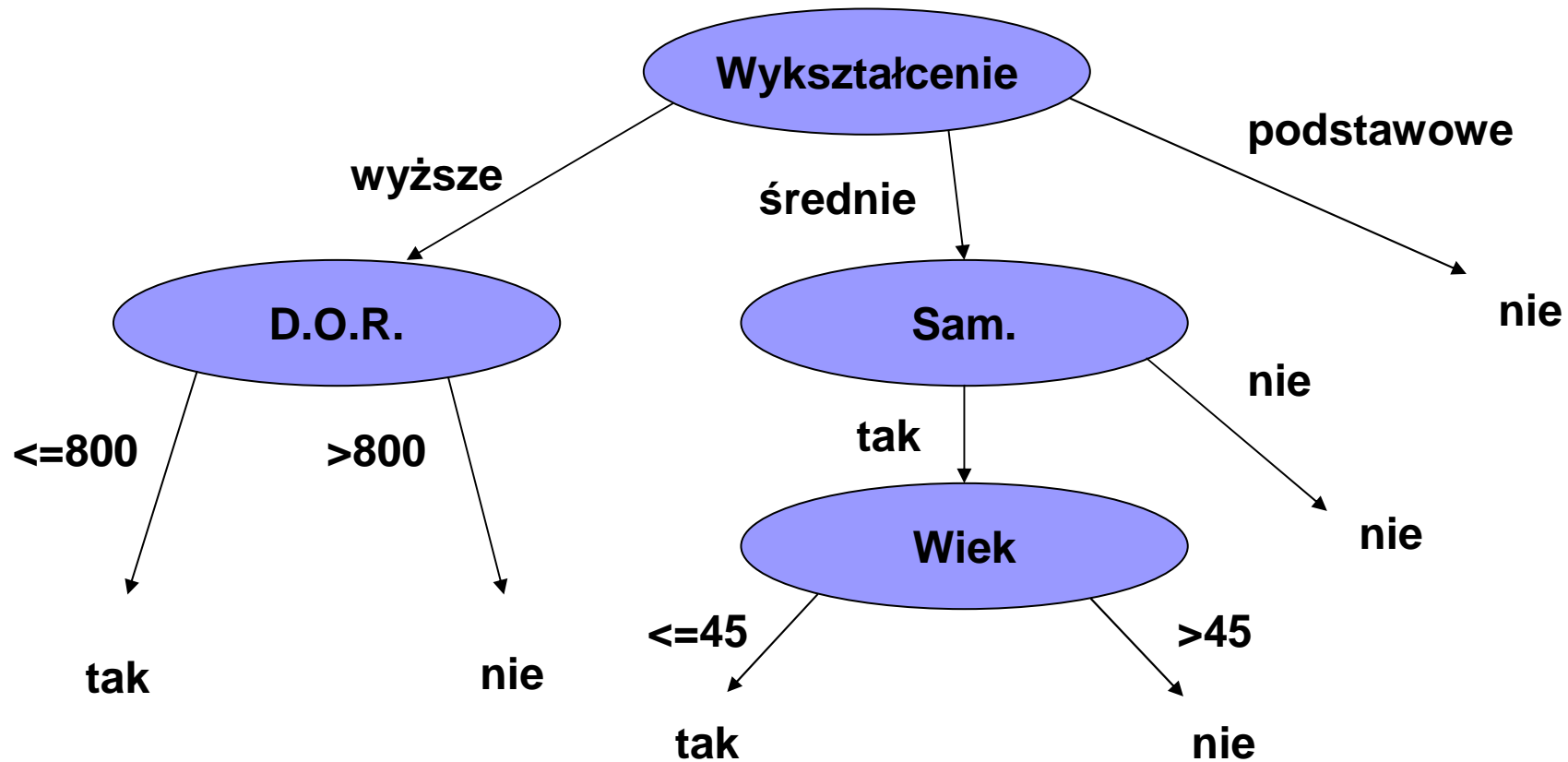
if Wykształcenie=podstawowe then Z.K.=nie

if Sam.=tak and Wiek<=45 then Z.K.=tak

if S.C.=S then Z.K.=nie

if D.O.R.<=500 then Z.K.=nie else Z.K.=tak

Klasyfikator decyzyjny – drzewa





Klasyfikator asocjacyjny

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. Nie wyróżniamy żadnego atrybutu
2. Szukamy zależności pomiędzy wartościami atrybutów w ramach przykładów

Reguły asocjacyjne

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

if Wykształcenie=średnie and Z.K.=tak then Sam.=tak

if Wykształcenie=podstawowe then S.C.=S



Grupowanie

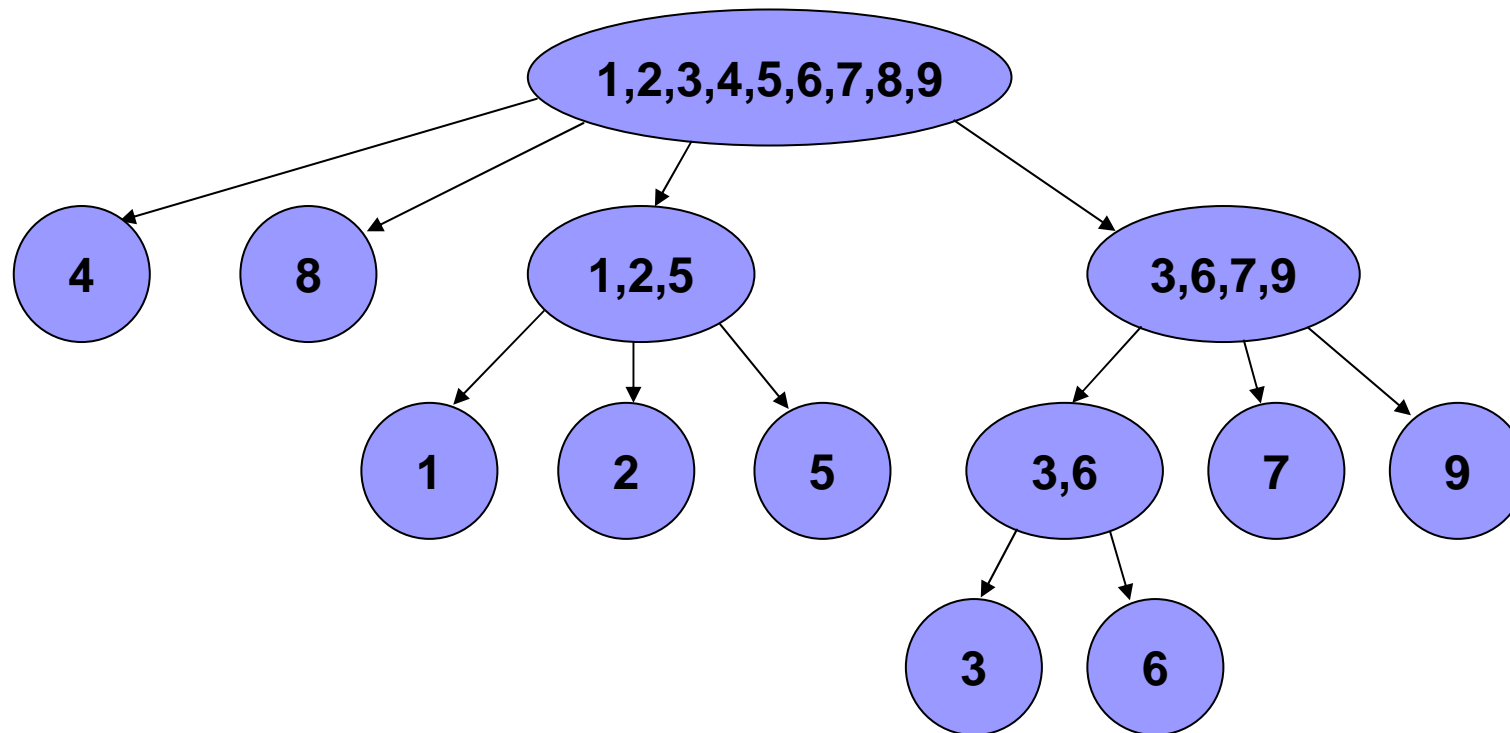
| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. Polega na odnajdywaniu grup „podobnych” do siebie przykładów
2. Grupy te nazywamy *klastrami*

Grupowanie - przykład

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

Grupowanie hierarchiczne





Metody analizy – komentarz

- Podstawowym zastosowaniem metod eksploracji danych jest *predykcja*
- Wyniki są nieodwracalnie obciążone błędami:
 - Skończone i niepełne zbiory trenujące,
 - Błędy w zbiorach trenujących,
 - Uproszczenia w stosowanych algorytmach,
 - Brak znajomości semantyki przetwarzanych danych

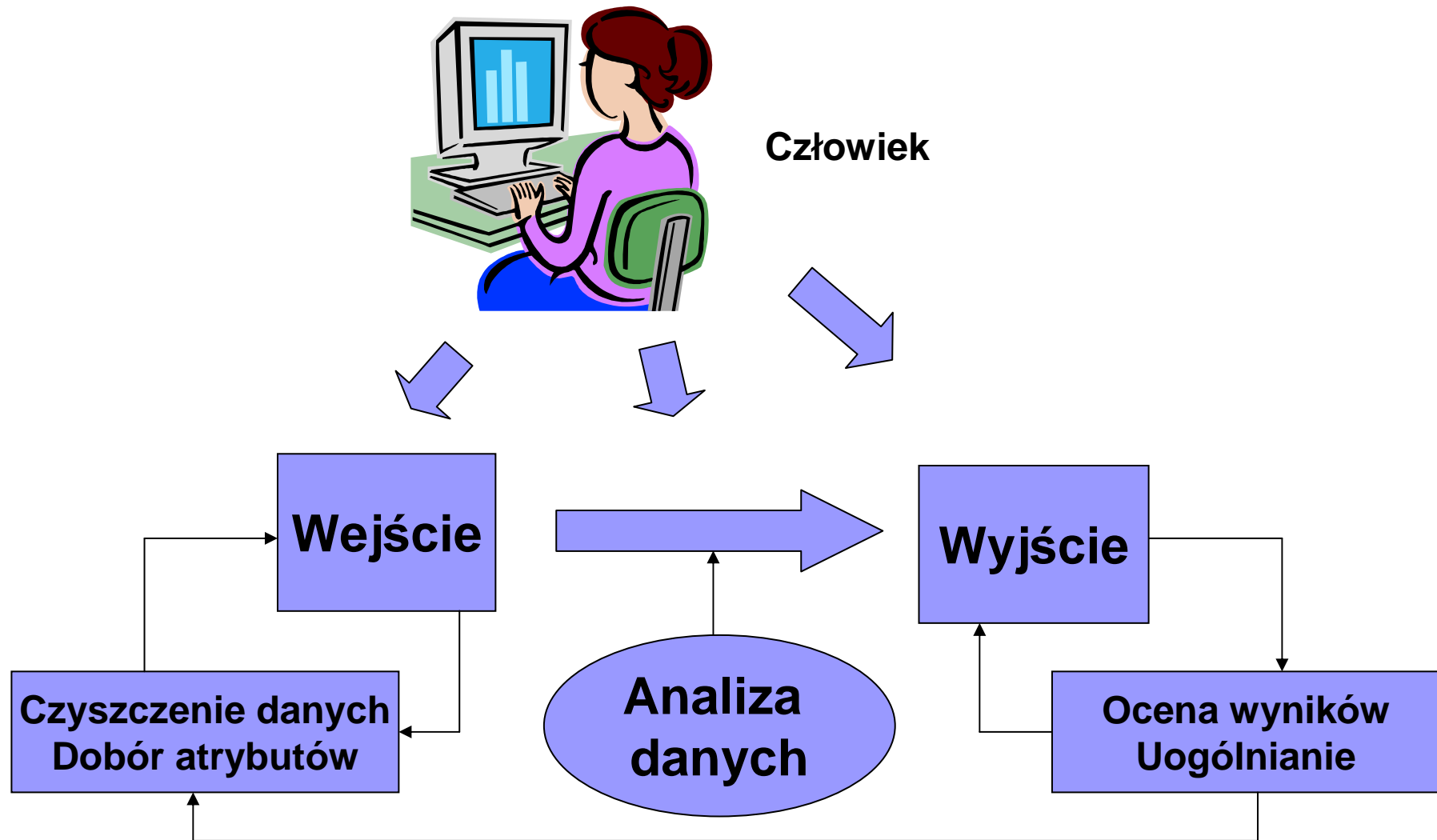


Błąd nadmiernego dopasowania

| Nazwisko | Wiek | Wykształcenie | Sam. | Z.K. |
|----------|------|---------------|------|------|
| Abacki | 32 | wyższe | tak | tak |
| Babacka | 35 | średnie | tak | tak |
| Cabacki | 26 | podstawowe | nie | nie |
| Dabacka | 45 | wyższe | nie | tak |
| Ebacki | 38 | średnie | tak | tak |
| Fabacki | 28 | wyższe | nie | nie |
| Gabacka | 65 | średnie | tak | nie |
| Habacka | 22 | średnie | nie | nie |
| Ibacki | 43 | podstawowe | tak | nie |

```
if Nazwisko=Abacki then Z.K.=tak  
if Nazwisko=Babacka then Z.K.=tak  
if Nazwisko=Cabacki then Z.K.=nie  
...
```

Uszczegółowiony proces





Inżynieria wejścia

- Czyszczenie danych:
 - Odnajdywanie wartości błędnych,
 - Uzupełnianie wartości brakujących
- Dobór atrybutów
- Zmiana typów atrybutów:
 - Dyskretyzacja wartości ciągłych,
 - Kwantyfikacja wartości dyskretnych



Inżynieria wyjścia

- Ocena wyników:

- Podział zbioru trenującego na część trenującą i testującą

- Uogólnianie wyników:

- Przycinanie drzew decyzyjnych,
- Przycinanie reguł,
- Łączenie klastrów



Kroki procesu eksploracji danych

- Dokładnie uświadomić sobie cele,
- Dokładnie poznać swoje dane,
- Zastosować odpowiednie metody przygotowania wejścia,
- Ocenąć wyjście,
- W razie potrzeby powtórzyć proces,
- Nie unikać zastosowania kilku metod łącznie do osiągnięcia celu



Etyczne aspekty wykorzystania eksploracji danych

- Efektywność metod eksploracji danych dla dużych zbiorów przykładów,
- Brak analizy semantyki przetwarzanych danych,
- Potencjalne błędy wynikające z obciążenia danych wejściowych i stosowanych metod,
- Podatność na tworzenie reguł noszących znamiona dyskryminacji.



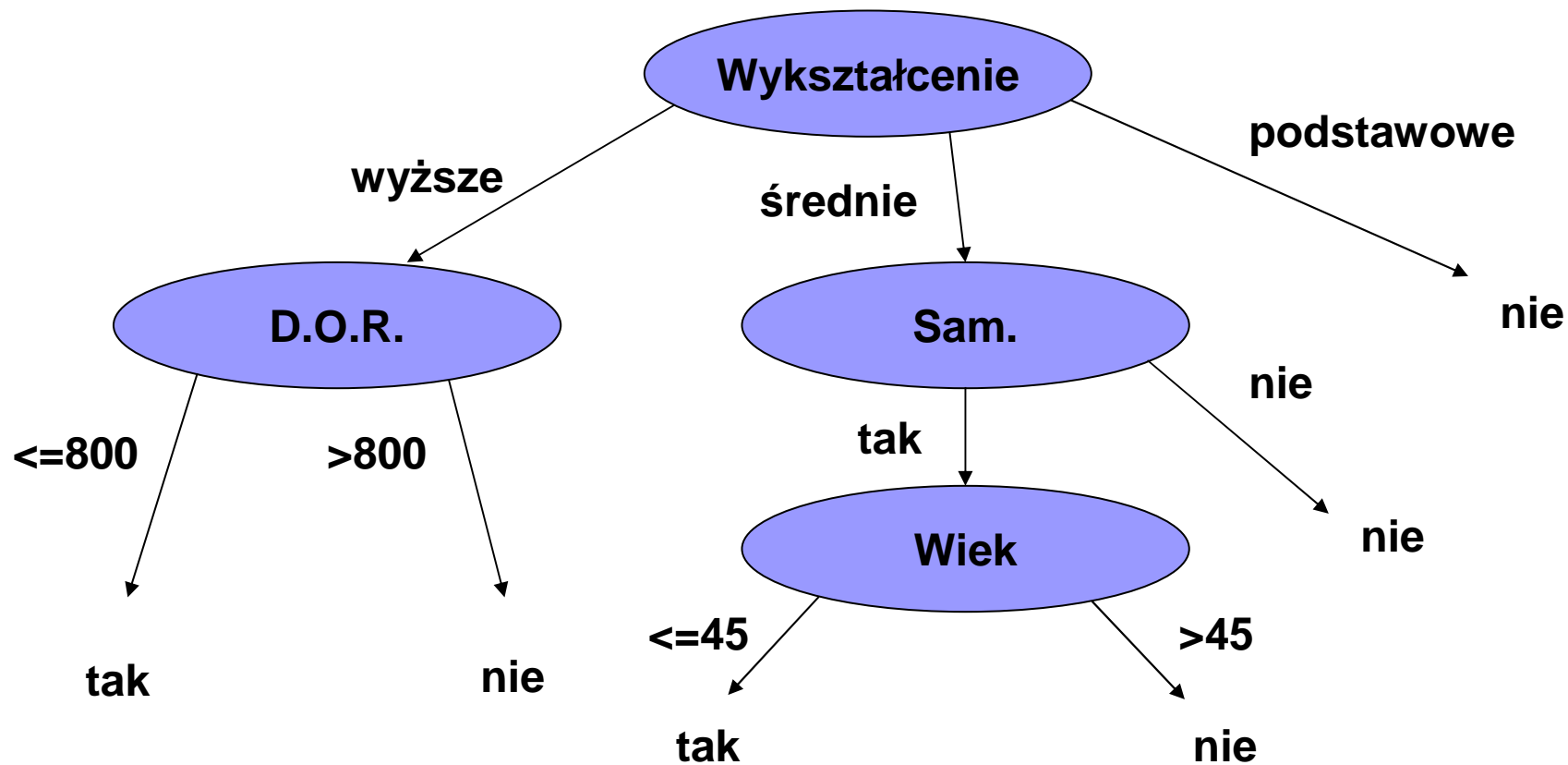
Budowa drzew decyzyjnych



Budowa drzew decyzyjnych

- Drzewa decyzyjne to najpopularniejsza forma klasyfikatorów decyzyjnych,
- Najczęściej budowane są metodą zstępującą, na zasadzie podejścia divide-and-conquer

Przykład drzewa decyzyjnego

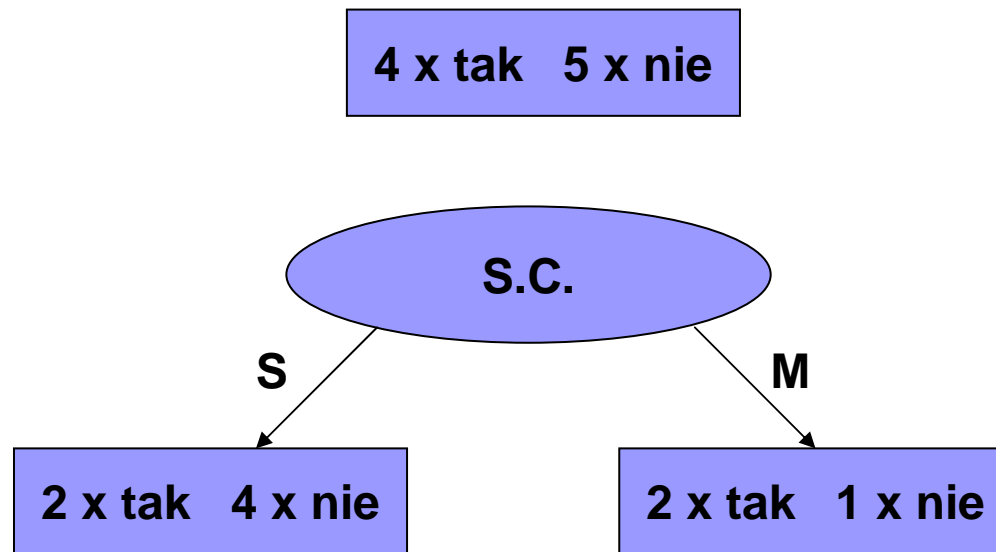


Budowa drzewa

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. Mamy wyróżniony atrybut decyzyjny, wyznaczający *klasy*
2. Na każdym poziomie drzewa wybieramy jeden z pozostałych atrybutów, najlepszy pod kątem dyskryminowania klas
3. Rozpoczynamy od pustego drzewa wyznaczając korzeń

Dobór atrybutu



Czy podział pod względem wartości atrybutu S.C. jest korzystny?

I w jakiej mierze?



Miara jakości podziału

- Miarą jakości podziału jest *przyrost zawartości informacji*
- Przyrost zawartości informacji jest określony jako różnica zawartości *informacji* w dzielonym zbiorze przykładów a *entropią* zastosowanego podziału (*testu*).



Miara jakości podziału – wzory

$$I(P) = \sum_{d \in C} -\frac{|P^d|}{|P|} \log_2 \frac{|P^d|}{|P|}$$

$$E_t(P) = \sum_{r \in R_t} \frac{|P_{tr}|}{|P|} I(P_{tr})$$

$I(P)$ – zawartość informacyjna zbioru przykładów P

C – zbiór klas wyznaczony przez atrybut decyzyjny

P^d – podzbiór tych przykładów ze zbioru P , które należą do klasy d

$E_t(P)$ – entropia testu t dla zbioru przykładów P

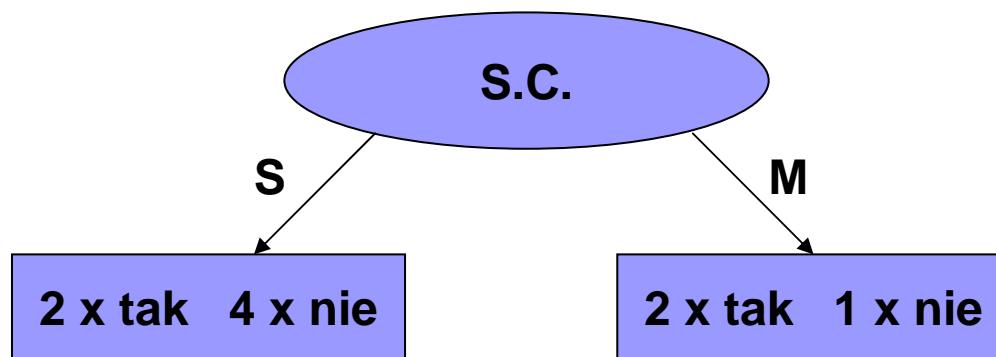
R_t – zbiór możliwych wyników testu t

P_{tr} – podzbiór tych przykładów ze zbioru P , które dają dla testu R wynik t

Dobór atrybutu - przykład

4 x tak 5 x nie

$$I(P) = -\frac{4}{9} \log_2 \frac{4}{9} - \frac{5}{9} \log_2 \frac{5}{9} \approx 0,99$$



$t = "S.C. = ?"$
 $R_t = \{S, M\}$

$$I(P_{S.C.=S}) = -\frac{2}{6} \log_2 \frac{2}{6} - \frac{4}{6} \log_2 \frac{4}{6} \approx 0,92$$

$$I(P_{S.C.=M}) = -\frac{2}{3} \log_2 \frac{2}{3} - \frac{1}{3} \log_2 \frac{1}{3} \approx 0,92$$

$$E_t(P) \approx \frac{6}{9} \cdot 0,92 + \frac{3}{9} \cdot 0,92 \approx 0,92$$

$$g_t(P) \approx 0,99 - 0,92 \approx 0,07$$

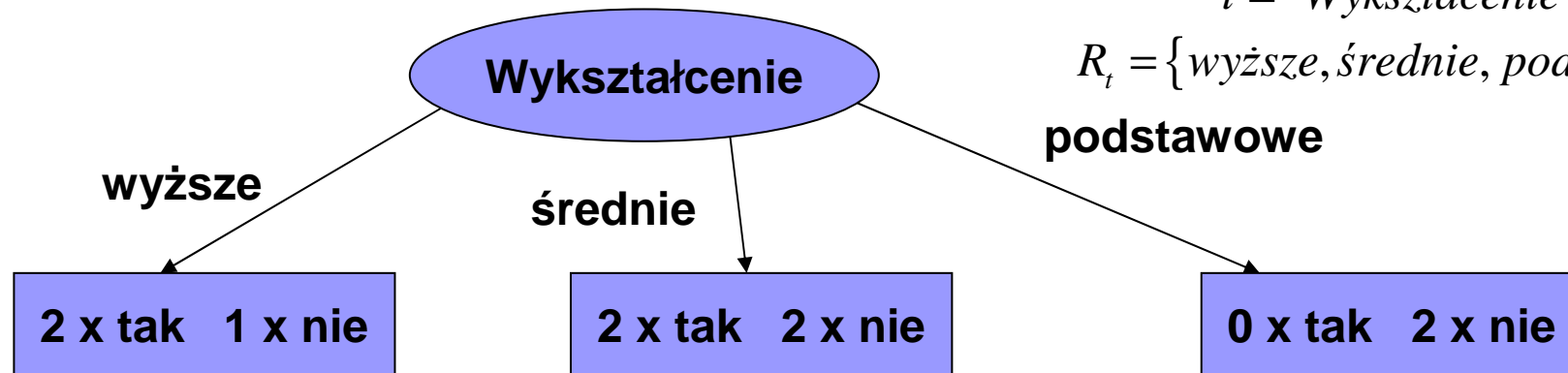
Dobór atrybutu – przykład (2)

4 x tak 5 x nie

$$I(P) = -\frac{4}{9} \log_2 \frac{4}{9} - \frac{5}{9} \log_2 \frac{5}{9} \approx 0,99$$

$t = \text{"Wykształcenie = ?"}$

$R_t = \{ \text{wyższe, średnie, podstawowe} \}$



$$I(P_{\text{Wykształcenie=wyższe}}) \approx 0,92$$

$$I(P_{\text{Wykształcenie=średnie}}) = 1$$

$$I(P_{\text{Wykształcenie=podstawowe}}) = 0$$

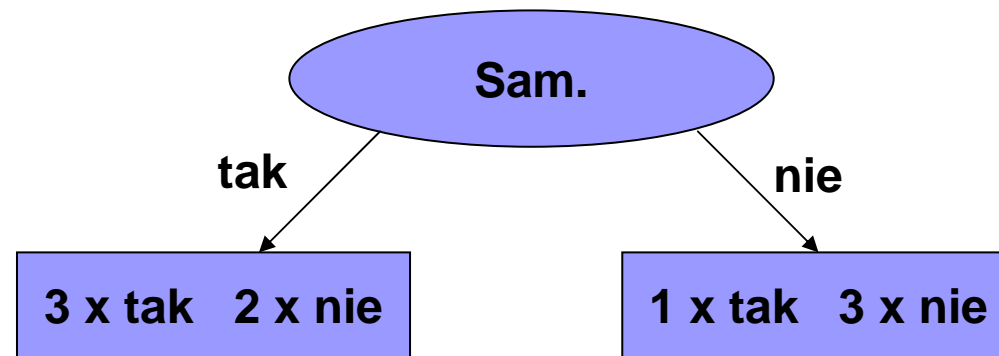
$$E_t(P) \approx \frac{3}{9} \cdot 0,92 + \frac{4}{9} \cdot 1 + \frac{2}{9} \cdot 0 \approx 0,75$$

$$g_t(P) \approx 0,99 - 0,75 \approx 0,24$$

Dobór atrybutu – przykład (3)

4 x tak 5 x nie

$$I(P) = -\frac{4}{9} \log_2 \frac{4}{9} - \frac{5}{9} \log_2 \frac{5}{9} \approx 0,99$$



$t = \text{"Sam."} = ?$

$R_t = \{tak, nie\}$

$$I(P_{Sam.=tak}) \approx 0,97$$

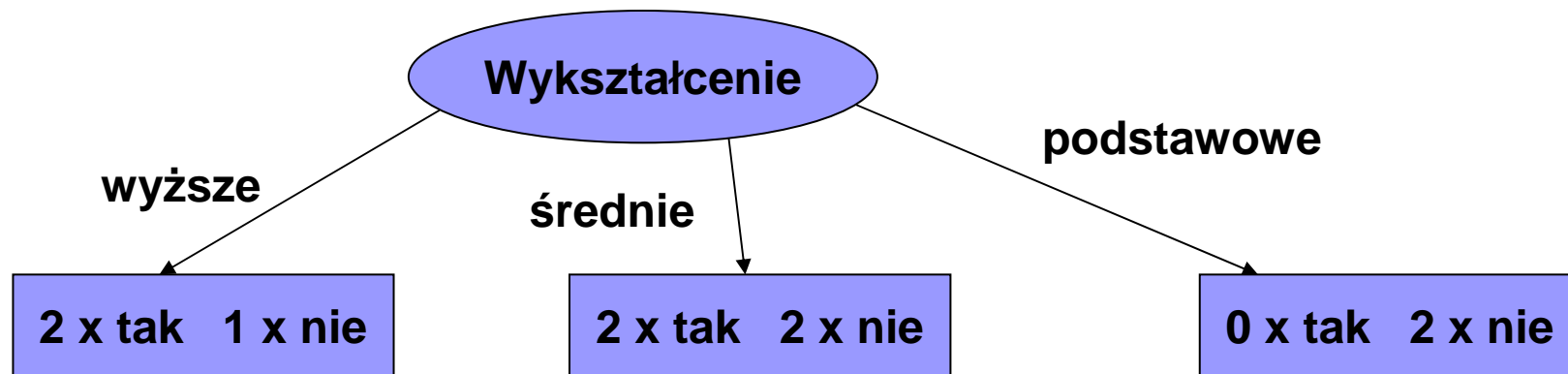
$$I(P_{Sam.=nie}) \approx 0,81$$

$$E_t(P) \approx \frac{5}{9} \cdot 0,97 + \frac{4}{9} \cdot 0,81 \approx 0,90$$

$$g_t(P) \approx 0,99 - 0,90 \approx 0,1$$

Dobór atrybutu – przykład (4)

- Najwyższy zysk informacji (0,24) osiągnął atrybut *Wykształcenie* i on zostaje zapisany w korzeniu drzewa decyzyjnego



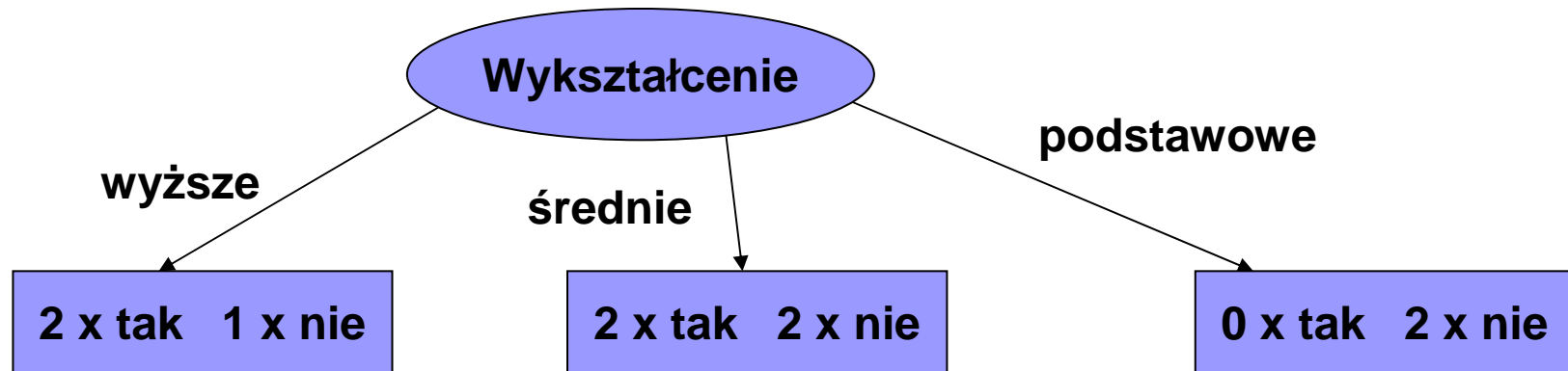
Divide-and-conquer

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

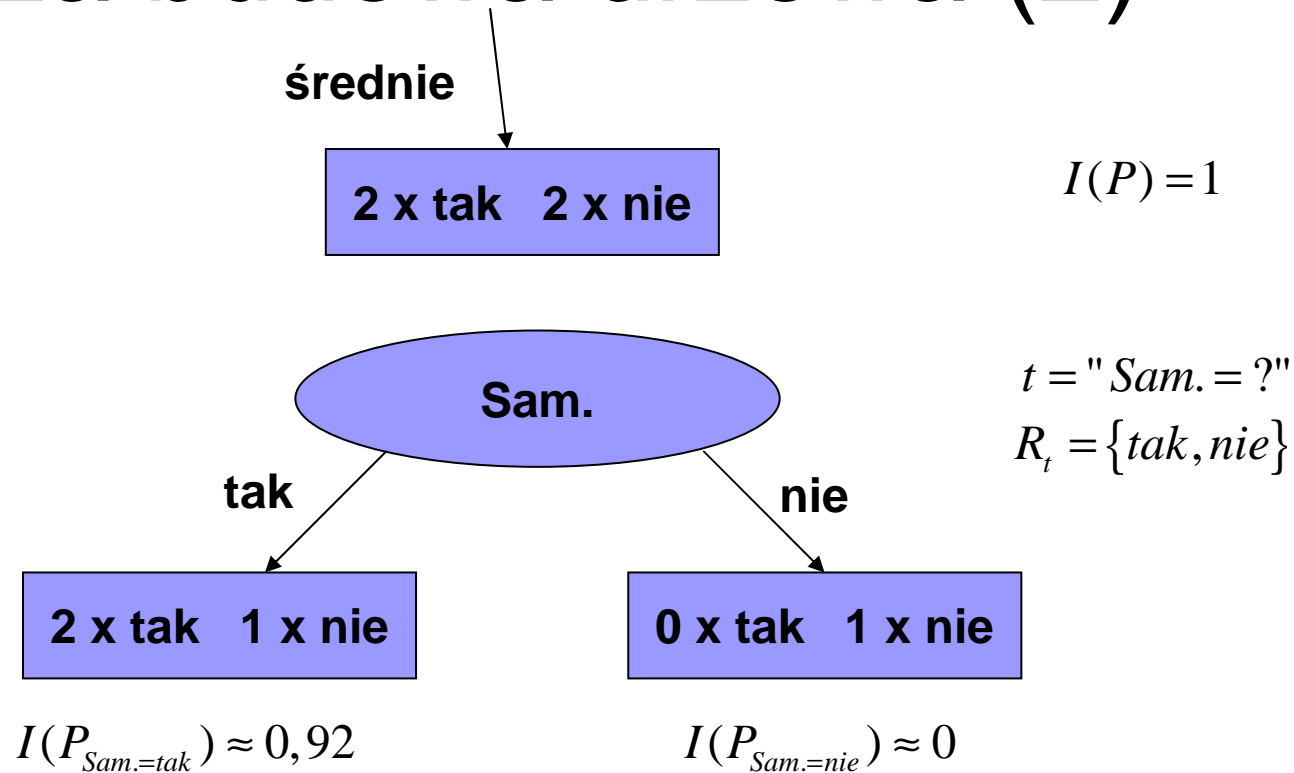
1. Zbiór przykładów został podzielony na trzy części
2. Dla każdej z części może zostać zastosowany ten sam algorytm dalszego działania

Dalsza budowa drzewa

- Wzdłuż prawej gałęzi drzewa nie trzeba już rozbudowywać



Dalsza budowa drzewa (2)



$$E_t(P) \approx \frac{3}{4} \cdot 0,92 + \frac{1}{4} \cdot 0 \approx 0,69$$

$$g_t(P) \approx 1 - 0,69 \approx 0,31$$



Atrybuty numeryczne

- Do tej pory zakładaliśmy użycie tylko atrybutów nominalnych,
- W trakcie budowy drzewa wykorzystywane mogą być też atrybuty numeryczne,
- Tutaj przedstawimy zasadę podziału binarnego minimalizującego entropię

Atrybuty numeryczne (2)

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. W trakcie budowy drzewa doszliśmy do wydzielenia 3 przykładów
2. W tym miejscu drzewa najlepiej zastosować podział względem wartości atrybutu numerycznego (wcześniej oczywiście takie podziały też były rozważane ale odrzucane)

Dalsza budowa drzewa (2)

tak

2 x tak 1 x nie

$$I(P) \approx 0,92$$

| | | | |
|-------|-----|-----|-----|
| Wiek: | 35 | 38 | 65 |
| Z.K.: | tak | tak | nie |

$t = \text{"Wiek} \leq x\text{"}$

$R_t = \{\leq x, > x\}$

$$I(P_{\text{Wiek} \leq x}) \approx 0 \quad I(P_{\text{Wiek} > x}) \approx 1$$

$$E_t(P) \approx \frac{1}{3} \cdot 0 + \frac{2}{3} \cdot 1 \approx 0,67$$

$$g_t(P) \approx 0,92 - 0,67 \approx 0,25$$

Dalsza budowa drzewa (3)

tak

2 x tak 1 x nie

$$I(P) \approx 0,92$$

| | | | |
|-------|-----|-----|-----|
| Wiek: | 35 | 38 | 65 |
| Z.K.: | tak | tak | nie |

$t = \text{"Wiek} \leq x\text{"}$

$R_t = \{\leq x, > x\}$

$$I(P_{\text{Wiek} \leq x}) \approx 0 \quad I(P_{\text{Wiek} > x}) \approx 0$$

$$E_t(P) \approx \frac{1}{3} \cdot 0 + \frac{2}{3} \cdot 0 = 0$$

$$g_t(P) \approx 0,92 - 0 \approx 0,92$$



Algorytm budowy drzew decyzyjnych

- Budowa drzewa polega na doborze najlepszego atrybutu nominalnego lub najlepszego podziału binarnego atrybutu numerycznego, powtarzanym iteracyjnie,
- Rozszerzenia:
 - Obsługa brakujących wartości atrybutów,
 - Przycinanie drzew – generalizacja.

Brakujące wartości atrybutów

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |
| M | 750 | 47 | ? | nie | nie |

Zakładamy, że mamy dodatkowy przykład o nieznannej wartości atrybutu *Wykształcenie*

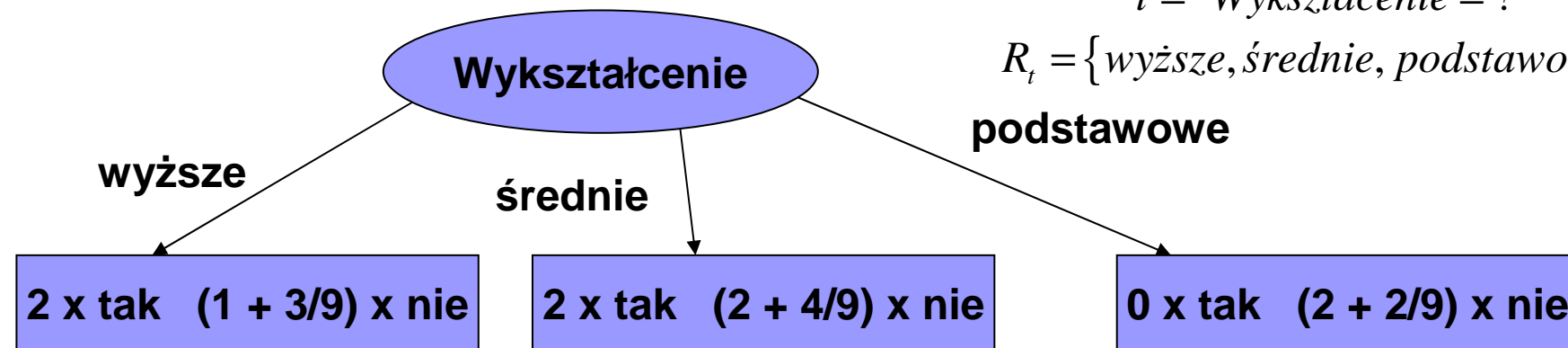
Brakujące wartości atrybutów (2)

4 x tak 6 x nie

$$I(P) \approx 0,92$$

$t = \text{"Wykształcenie = ?"}$

$R_t = \{ \text{wyższe, średnie, podstawowe} \}$



$$I(P_{\text{Wykształcenie=wyższe}}) \approx 0,97$$

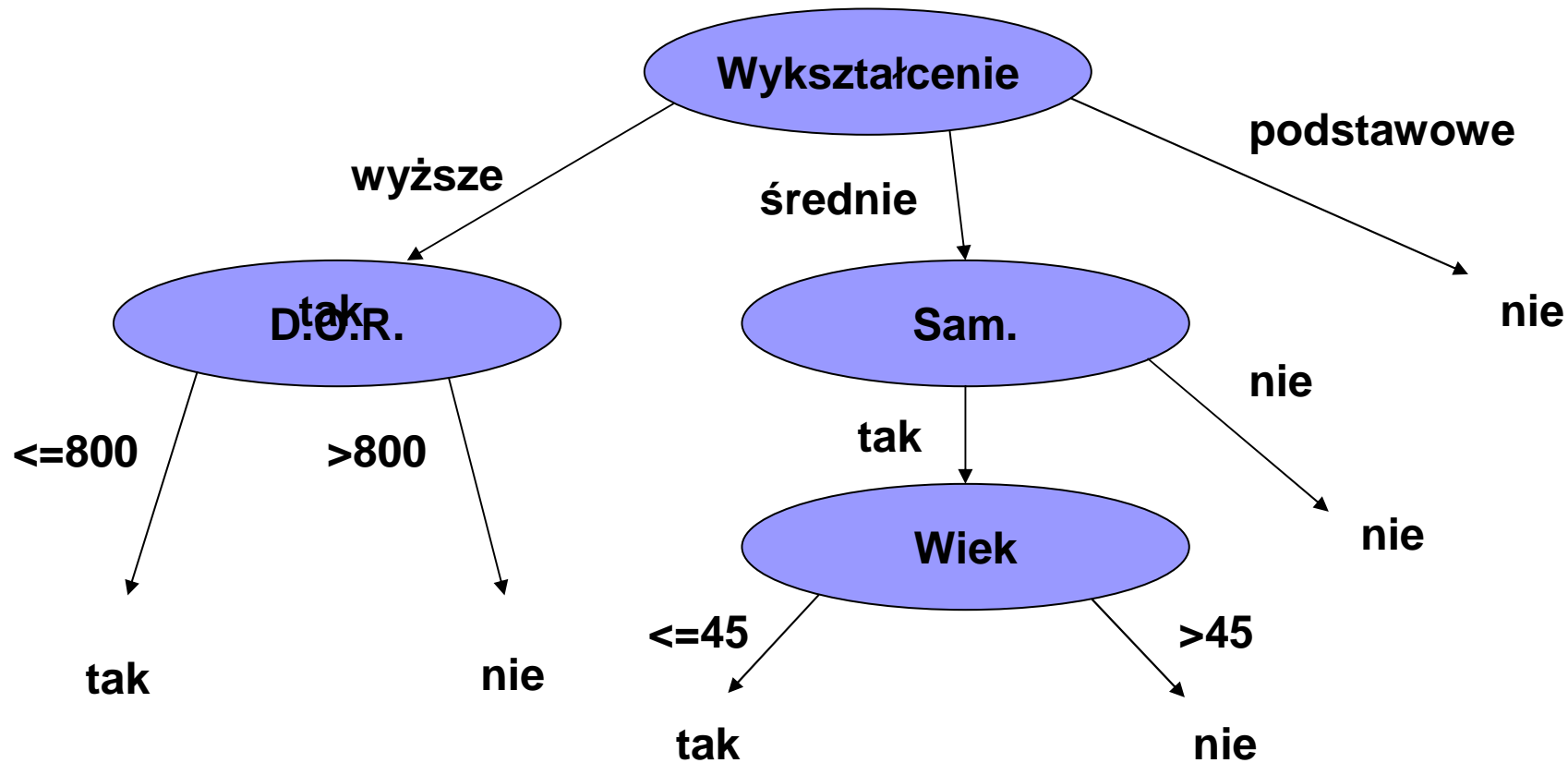
$$I(P_{\text{Wykształcenie=średnie}}) = 0,99$$

$$I(P_{\text{Wykształcenie=podstawowe}}) = 0$$

$$E_t(P) \approx \frac{3+3/9}{10} \cdot 0,97 + \frac{4+4/9}{10} \cdot 0,99 + \frac{2+2/9}{10} \cdot 0 \approx 0,76$$

$$g_t(P) \approx 0,92 - 0,76 \approx 0,16$$

Przycinanie drzewa decyzyjnego





Przycinanie

- Przycinanie ma na celu uogólnienie wyników i zapobieżenie błędowi nadmiernego dopasowania,
- Stosuje się różne kryteria przycinania,
- Innym sposobem uogólnienia drzewa jest ustalenie minimalnej liczby przykładów pokrywanych przez liść.



Podsumowanie

- Budowane iteracyjnie, metodą zachłanną, zstępującą, divide-and-conquer,
- Algorytmy budowy zawierają rozszerzenia pozwalające na obsługę atrybutów numerycznych, obsługę wartości brakujących oraz uogólnianie metodą przycinania.



Budowa reguł decyzyjnych



Budowa reguł decyzyjnych

- Reguły decyzyjne są bardzo popularną formą wyrażania zasad klasyfikacji,
- Budowane są różnymi metodami, jedną z nich jest metoda oparta na podejściu separate-and-conquer

Pokrycie i poprawność reguły

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

`if Sam.=nie then Z.K.=nie`

Pokrycie: $s = 4$

Poprawność: $a = 3 / 4$



Pokrycie i poprawność reguły (2)

- Pokrycie (ang. *coverage, support*) to liczba przykładów, dla których zadziała reguła (które spełniają część testową reguły).
- Poprawność (ang. *accuracy, confidence*) to liczba przykładów poprawnie klasyfikowanych przez regułę.

Budowa reguł

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. Mamy wyróżniony atrybut decyzyjny, wyznaczający *klasy*
2. Budujemy regułę pod kątem najlepszej poprawności i, w drugim rzędzie, największego pokrycia.
3. Rozpoczynamy od pustej reguły wyznaczając kolejne testy.

Budowa reguły decyzyjnej

Do reguły pustej:

```
if ? then Z.K.=?
```



wstawiamy „na próbę” wszystkie możliwe testy proste (z rezultatem):

```
S.C.=S
```

```
S.C.=M
```

```
Wykształcenie=podstawowe
```

```
Wykształcenie=średnie
```

```
Wykształcenie=wyższe
```

```
Sam.=tak
```

```
Sam.=nie
```

Budowa reguły decyzyjnej (2)

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

if S.C.=S then Z.K.=?

Pokrycie: $s = 6$

if S.C.=S then Z.K.=nie

Poprawność: $a = 4 / 6$

Budowa reguły decyzyjnej (3)

- Po zbadaniu wszystkich testów otrzymujemy:

| | <i>a</i> | <i>s</i> |
|--------------------------|----------|----------|
| S.C.=S | 4/6 | 6 |
| S.C.=M | 2/3 | 3 |
| Wykształcenie=podstawowe | 2/2 | 2 |
| Wykształcenie=średnie | 2/4 | 4 |
| Wykształcenie=wyższe | 2/3 | 3 |
| Sam.=tak | 3/5 | 5 |
| Sam.=nie | 3/4 | 4 |

Wybieramy test o największej poprawności



Budowa reguły decyzyjnej (4)

- Ponieważ osiągnięta poprawność wynosi 100%, pierwsza reguła jest gotowa:

```
if Wykształcenie=podstawowe then Z.K.=nie
```

- Następne reguły budowane są analogicznie po zastosowaniu zasady separate-and-conquer

Separate-and-conquer

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

1. W zbiorze przykładów wyróżniliśmy dwa pokrywane przez regułę
2. Usuwamy te przykłady z naszego zbioru trenującego, a dla pozostałej jego części stosujemy ponownie procedurę budowy najlepszej reguły



Budowa kolejnych reguł

- Ponownie rozpoczynamy od reguły pustej, rozważając pozostałe testy:

S.C.=S

S.C.=M

Wykształcenie=średnie

Wykształcenie=wyższe

Sam.=tak

Sam.=nie

Budowa kolejnych reguł (2)

- Po zbadaniu wszystkich testów otrzymujemy:

| | <i>a</i> | <i>s</i> |
|-----------------------|----------|----------|
| S.C.=S | 2/4 | 4 |
| S.C.=M | 2/3 | 3 |
| Wykształcenie=średnie | 2/4 | 4 |
| Wykształcenie=wyższe | 2/3 | 3 |
| Sam.=tak | 3/4 | 4 |
| Sam.=nie | 2/3 | 3 |

Wybieramy test o największej poprawności



Budowa kolejnych reguł (3)

- Ponieważ osiągnięta poprawność wynosi 75%, regułę można jeszcze poprawić:

```
if Sam.=tak then Z.K.=tak
```

- Regułę możemy rozbudowywać, wybierając kolejne testy:

```
if Sam.=tak and ? then Z.K.=?
```

Rozbudowa reguły

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

Przy rozbudowywaniu reguły ograniczmy się do rozpatrywania przykładów pokrywanych przez pierwszy test niepokrywanych przez poprzednie reguły.

if Sam.=tak and ? then Z.K.=?



Rozbudowa reguły (2)

- W miejsce „?” moglibyśmy wstawić jeden z testów wartości atrybutów nominalnych:

`S.C.=S`

`S.C.=M`

`Wykształcenie=średnie`

`Wykształcenie=wyższe`

- Pokrycie i poprawność tak wygenerowanej reguły sprawdzamy analogicznie jak poprzednio

Rozbudowa reguły (3)

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

if Sam.=tak and S.C.=S then Z.K.=?

Pokrycie: $s = 3$

Poprawność: $a = 2 / 3$



Rozbudowa reguły (4)

- Po zbadaniu testów otrzymujemy:

| | <i>a</i> | <i>s</i> |
|-----------------------|----------|----------|
| S.C.=S | 2/3 | 3 |
| S.C.=M | 1/1 | 1 |
| Wykształcenie=średnie | 2/3 | 3 |
| Wykształcenie=wyższe | 1/1 | 1 |

Moglibyśmy na tym poprzestać, wybierając test o największej poprawności, ale tutaj włączymy do rozważań atrybuty numeryczne



Obsługa atrybutów numerycznych

Testy dla atrybutów numerycznych mogą być wyznaczane na różne sposoby – również omówioną metodą podziału binarnego minimalizującego entropię:

| | | | | |
|--------------|------------|------------|------------|------------|
| Wiek: | 32 | 35 | 38 | 65 |
| Z.K.: | tak | tak | tak | nie |

Co daje nam dwa dodatkowe testy do rozpatrzenia:

Wiek \leq 45

Wiek $>$ 45

Rola pokrycia

- Tym razem mamy wynik:

| | <i>a</i> | <i>s</i> |
|-----------------------|----------|----------|
| S.C.=S | 2/3 | 3 |
| S.C.=M | 1/1 | 1 |
| Wykształcenie=średnie | 2/3 | 3 |
| Wykształcenie=wyższe | 1/1 | 1 |
| Wiek≤45 | 3/3 | 3 |
| Wiek>45 | 1/1 | 1 |
| D.O.R.≤900 | 2/2 | 2 |
| D.O.R.>900 | 1/2 | 2 |

Wybieramy test o największej poprawności i największym pokryciu



Kolejne kroki

- Ponieważ osiągnięta poprawność wynosi 100%, druga reguła jest gotowa:

```
if Sam.=tak and Wiek<=45 then Z.K.=tak
```

- Cały proces jest powtarzany (następna iteracja) po kolejnym zastosowaniu zasady separate-and-conquer

Kolejne kroki (2)

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

if Wykształcenie=podstawowe then Z.K.=nie

if Sam.=tak and Wiek<=45 then Z.K.=tak

if S.C.=S then Z.K.=nie

if D.O.R.<=500 then Z.K.=nie else Z.K.=tak



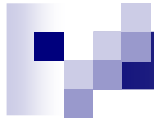
Budowa reguł – komentarz

- Zastosowanie zasady separate-and-conquer sprawia, że wygenerowane reguły muszą być rozpatrywane łącznie i w kolejności,
- Metoda w sposób naturalny radzi sobie z brakującymi wartościami atrybutów (są one „spychane” do następnej iteracji),
- Metodę można rozszerzyć o pewne mechanizmy uogólniające, przycinające reguły według pewnych zależności statystycznych



Budowa reguł – podsumowanie

- Budowane iteracyjnie, metodą zachłanną według zasady separate-and-conquer,
- W trakcie budowy reguł maksymalizowana jest poprawność i, w drugiej kolejności, pokrycie reguł,
- Metodę można rozszerzyć o obsługę atrybutów numerycznych i pewne mechanizmy uogólniające (przycinanie); naturalnie obsługiwane są wartości brakujące



Budowa reguł asocjacyjnych



Budowa reguł asocjacyjnych

- Przy budowie reguł asocjacyjnych nie mamy wyróżnionego atrybutu decyzyjnego,
- Próbujemy uchwycić różne zależności między atrybutami bez wyróżniania żadnego z nich,
- Naiwne podejście mogłoby polegać na wygenerowaniu klasyfikatora decyzyjnego dla każdego z atrybutów – jest ono jednak skrajnie nieefektywne



Algorytm Apriori

- Algorytm działa jedynie na atrybutach nominalnych,
- Dla generowanych reguł ustalamy progi pokrycia i poprawności (s_{min} i a_{min}),
- Algorytm ten buduje reguły tworząc testy złożone,
- Na początek brane pod uwagę jest jedynie pokrycie testu, które musi przekraczać ustalony próg minimalny



Apriori – przykład

- Próg pokrycia ustalamy na $s_{min}=2$,
- Najpierw bierzemy po uwagę testy proste:

S.C.=S

S.C.=M

Wykształcenie=podstawowe

Wykształcenie=średnie

Wykształcenie=wyższe

Sam.=tak

Sam.=nie

Z.K.=tak

Z.K.=nie

Obliczanie pokrycia testu

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |

S.C. = S

Pokrycie: $s = 6$

Apriori – testy proste

- Otrzymujemy następujące pokrycia:

| | s |
|--------------------------|---|
| S.C.=S | 6 |
| S.C.=M | 3 |
| Wykształcenie=podstawowe | 2 |
| Wykształcenie=średnie | 4 |
| Wykształcenie=wyższe | 3 |
| Sam.=tak | 5 |
| Sam.=nie | 4 |
| Z.K.=tak | 4 |
| Z.K.=nie | 5 |

Wszystkie testy proste przekraczają próg pokrycia i mogą być wykorzystane do budowy testów złożonych



Apriori – testy złożone

- Testy podwójne budujemy z testów pojedynczych z poprzedniego kroku (oba składowe testy pojedyncze muszą mieć minimalne pokrycie),
- Pary budujemy według zasady każdy-z-każdym (oprócz odrzuconych):

`S.C.=S and Wykształcenie=podstawowe`

`S.C.=S and Wykształcenie=średnie`

`S.C.=S and Wykształcenie=wyższe`

`S.C.=S and Sam.=tak`

`...`

Apriori – testy złożone (2)

| | |
|--|--------------|
| ■ Następnie badamy pokrycia testów: | s |
| S.C.=S and Wykształcenie=podstawowe | 2 |
| S.C.=S and Wykształcenie=średnie | 2 |
| S.C.=S and Wykształcenie=wyższe | 2 |
| ... | |
| S.C.=M and Wykształcenie=wyższe | 1 |
| ... | |

Spośród 30 testów złożonych kryterium pokrycia spełnia 19



Apriori – generacja reguł

- Otrzymaliśmy 19 testów złożonych spełniających warunek minimalnego pokrycia,
- Z każdego testu możemy spróbować wygenerować reguły; dla testu:

`s.c.=s` and `Wykształcenie=podstawowe`
mogą to być:

`if s.c.=s then Wykształcenie=podstawowe`

`if Wykształcenie=podstawowe then s.c.=s`

Apriori – generacja reguł (2)

- Reguły akceptujemy tylko jeżeli przekraczają one próg poprawności (ustalmy $a_{min}=1$):

~~if S.C.=S then Wykształcenie=podstawowe 2/6~~ *a*

if Wykształcenie=podstawowe then S.C.=S 2/2

Kryterium poprawności spełnia tylko jedna z wygenerowanych reguł



Apriori – generacja reguł (3)

- Z 19 testów podwójnych możemy wygenerować tylko 2 reguły spełniające progi poprawności:

```
if Wykształcenie=podstawowe then S.C.=S
```

```
if Wykształcenie=podstawowe then Z.K.=nie
```

- Po wygenerowaniu tych reguł możemy przystąpić do generacji testów potrójnych



Apriori – testy potrójne

- Testy potrójne budujemy z testów podwójnych z poprzedniego kroku (wszystkie składowe testy podwójne muszą mieć minimalne pokrycie):

`S.C.=S and Sam.=nie and Z.K.=tak`

na pewno nie spełnia kryterium pokrycia,
bo nie spełnia go test podwójny:

`Sam.=nie and Z.K.=tak`



Apriori – testy potrójne (2)

- Kryterium pokrycia spełniają następujące testy potrójne:

Wykształcenie=wyższe and Sam.=tak and Z.K.=tak

S.C.=S and Sam.=nie and Z.K.=nie

S.C.=S and Sam.=tak and Z.K.=nie

S.C.=S and Sam.=tak and Z.K.=tak

S.C.=S and Wykształcenie=wyższe and Sam.=tak

S.C.=S and Wykształcenie=podstawowe and Z.K.=tak

- Dla każdego testu generowane są reguły, dla których sprawdzane jest kryterium poprawności

Apriori – przykład

■ Dla testu:

Wykształcenie=wyższe and Sam.=tak and Z.K.=tak

generowane są następujące reguły:

if Wykształcenie=wyższe then Sam.=tak and Z.K.=tak

if Sam.=tak then Wykształcenie=wyższe and Z.K.=tak

if Z.K.=tak then Wykształcenie=wyższe and Sam.=tak

if Wykształcenie=wyższe and Sam.=tak then Z.K.=tak

if Wykształcenie=wyższe and Z.K.=tak then Sam.=tak

if Sam.=tak and Z.K.=tak then Wykształcenie=wyższe

Tylko jedna spełnia kryterium poprawności



Apriori – generacja reguł

- Z 6 testów potrójnych generujemy 8 reguł spełniających próg poprawności:

```
if Wykształcenie=wyższe then Sam.=tak and Z.K.=tak
if S.C.=S and Sam.=nie then Z.K.=nie
if Sam.=tak and Z.K.=nie then S.C.=S
if S.C.=S and Z.K.=tak then Sam.=tak
if S.C.=S and Wykształcenie=wyższe then Sam.=tak
if Wykształcenie=postawowe then S.C.=S and Z.K.=nie
if S.C.=S and Wykształcenie=postawowe then Z.K.=nie
if Wykształcenie=postawowe and Z.K.=nie then S.C.=S
```

- Po wygenerowaniu tych reguł możemy przystąpić do generacji testów poczwórnych



Apriori – testy poczwórne

- Testy poczwórne budujemy z testów potrójnych z poprzedniego kroku (wszystkie składowe testy podwójne muszą mieć minimalne pokrycie),
- W podanym przykładzie nie ma testów poczwórnych spełniających warunek minimalnego pokrycia,
- Oznacza to, że algorytm Apriori kończy pracę



Apriori – wynik

- Ostatecznie wygenerowano 10 reguł:

```
if Wykształcenie=wyższe then Sam.=tak and Z.K.=tak
if S.C.=S and Sam.=nie then Z.K.=nie
if Sam.=tak and Z.K.=nie then S.C.=S
if S.C.=S and Z.K.=tak then Sam.=tak
if S.C.=S and Wykształcenie=wyższe then Sam.=tak
if Wykształcenie=postawowe then S.C.=S and Z.K.=nie
if S.C.=S and Wykształcenie=postawowe then Z.K.=nie
if Wykształcenie=postawowe and Z.K.=nie then S.C.=S
if Wykształcenie=podstawowe then S.C.=S
if Wykształcenie=podstawowe then Z.K.=nie
```

Wszystkie spełniają kryteria poprawności i pokrycia



Algorytm Apriori

Wejście: s_{min} - próg pokrycia,
 a_{min} - próg poprawności,
 P - zbiór przykładów

Wyjście: R - zbiór reguł

1. $n := 1, R := \{\}$
2. Generuj zbiór testów pojedynczych T_1
3. Eliminuj testy o pokryciu mniejszym od s_{min}
4. Dopóki T_n jest niepusty wykonuj:
 5. $n := n + 1$
 6. Generuj zbiór testów n -krotnych T_n
 7. Eliminuj z T_n testy o pokryciu mniejszym od s_{min}
 8. Dla każdego testu t z T_n :
 9. Generuj zbiór R_t reguł dla testu t
 10. Eliminuj z R_t reguły o poprawności mniejszej od a_{min}
 11. $R := R \cup R_t$
 12. Koniec pętli wewnętrznej
13. Koniec pętli zewnętrznej



Apriori – drugi przykład

- Prześledźmy jeszcze działanie algorytmu dla $s_{min} = 4$ i $a_{min} = 0,8$

Apriori – drugi przykład (2)

- Dla testów prostych otrzymujemy następujące pokrycia:

| | s |
|-------------------------------------|--------------|
| S.C.=S | 6 |
| S.C.=M | 3 |
| Wykształcenie=podstawowe | 2 |
| Wykształcenie=średnie | 4 |
| Wykształcenie=wyższe | 3 |
| Sam.=tak | 5 |
| Sam.=nie | 4 |
| Z.K.=tak | 4 |
| Z.K.=nie | 5 |

Testy niespełniające kryterium pokrycia możemy wyeliminować.



Apriori – drugi przykład (3)

- Testy podwójne budujemy z testów pojedynczych z poprzedniego kroku (oba składowe testy pojedyncze muszą mieć minimalne pokrycie),
- Po odrzuceniu testów niespełniających kryterium pokrycia otrzymujemy dwa testy podwójne:

S.C.=S and Sam.=tak

S.C.=S and Z.K.=nie



Apriori – drugi przykład (4)

- Z testów podwójnych możemy wygenerować 2 reguły spełniające progi poprawności:

```
if Z.K.=nie then S.C.=S
```

```
if Sam.=tak then S.C.=S
```

- Po wygenerowaniu tych reguł możemy przystąpić do generacji testów potrójnych



Apriori – drugi przykład (5)

- Żaden z testów potrójnych nie spełnia kryterium minimalnego pokrycia, zatem ostateczny wynik pracy algorytmu to:

```
if Z.K.=nie then S.C.=S
```

```
if Sam.=tak then S.C.=S
```



Apriori – komentarz

- Apriori w większości przypadków pozwala stosunkowo szybko (mimo teoretycznie wykładniczej złożoności) wygenerować reguły asocjacyjne,
- Liczba generowanych reguł jest zazwyczaj bardzo duża, zatem zalecane jest rozpoczęcie pracy do wysokiego progu pokrycia i redukowanie go w trakcie analizy



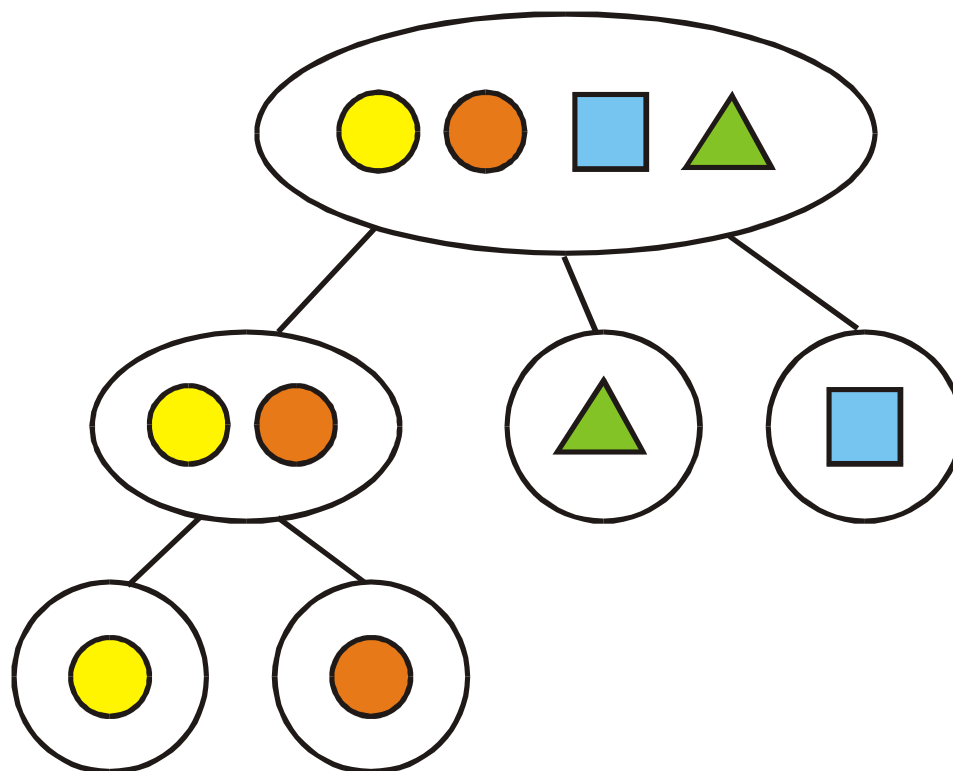
Grupowanie



COBWEB

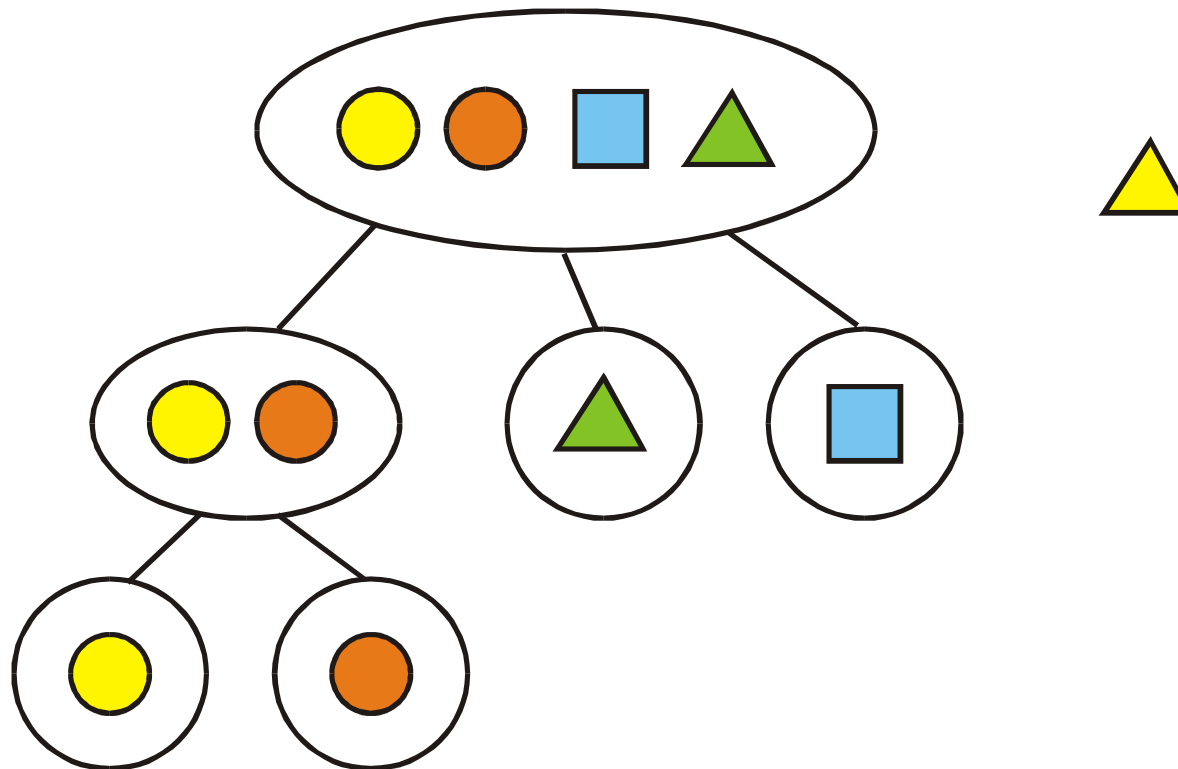
- Grupowanie polega na wygenerowaniu pewnych zasad podziału danych na zasadzie podobieństwa,
- Algorytm COBWEB jest przykładem algorytmu grupującego,
- Algorytm COBWEB generuje hierarchiczny podział danych w postaci drzewa.

Drzewo COBWEB



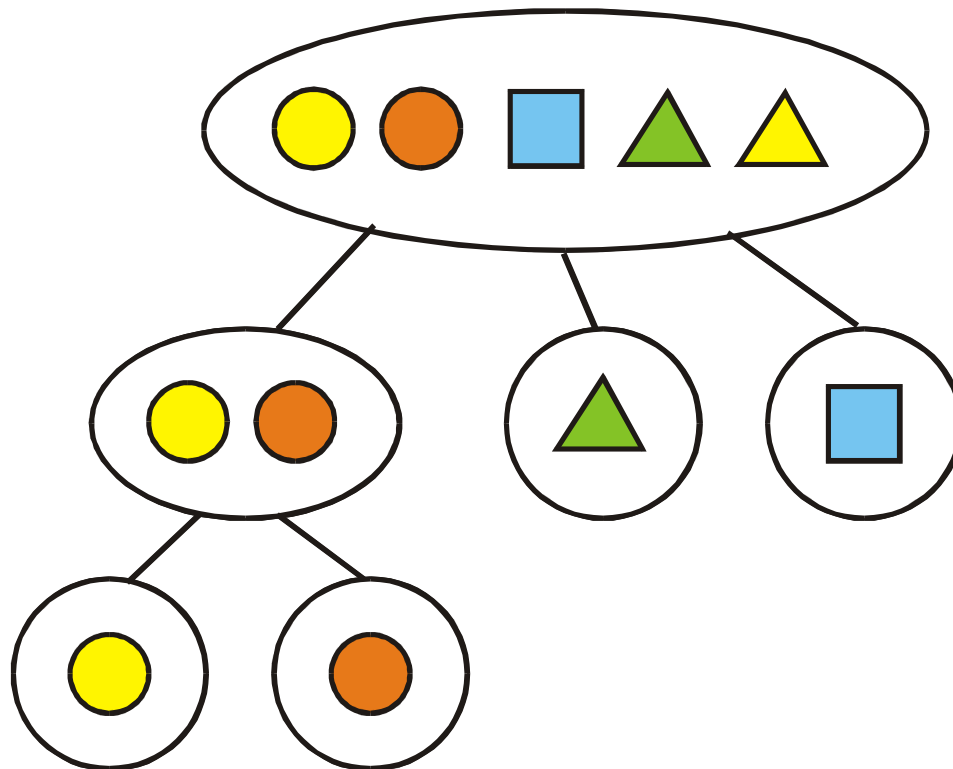
COBWEB – grupowanie

Założmy, że chcemy przeprowadzić proces grupowania dla kolejnego przykładu:



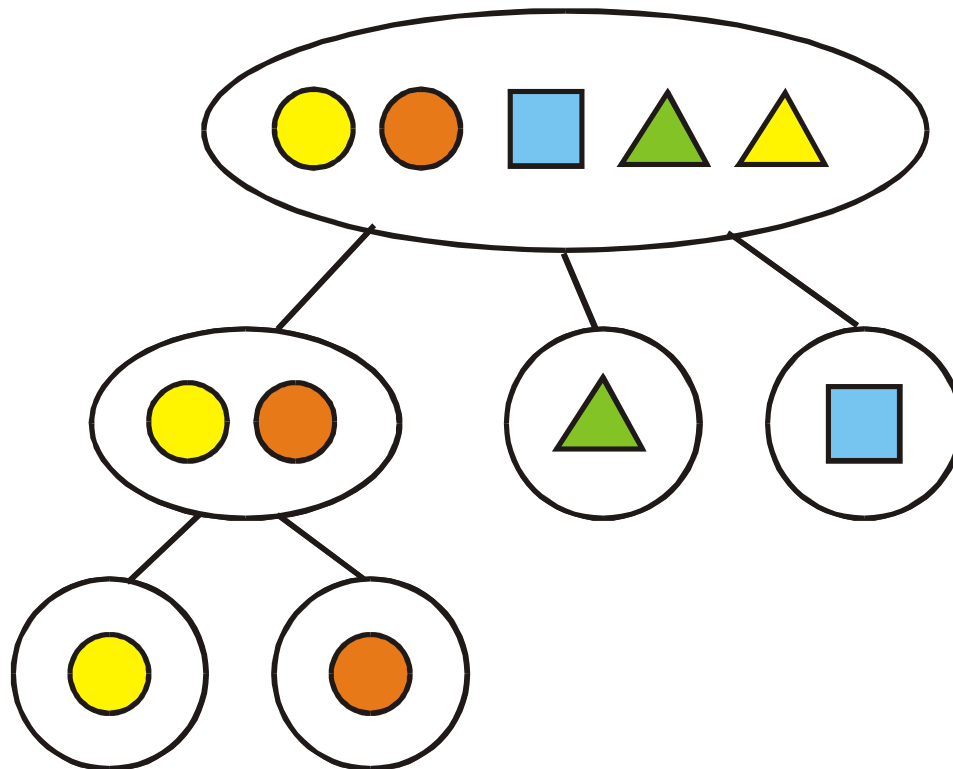
COBWEB – grupowanie


COBWEB dokłada przykład do głównej kategorii.



COBWEB – grupowanie

Następnie próbuje różnych możliwości dalszego grupowania.



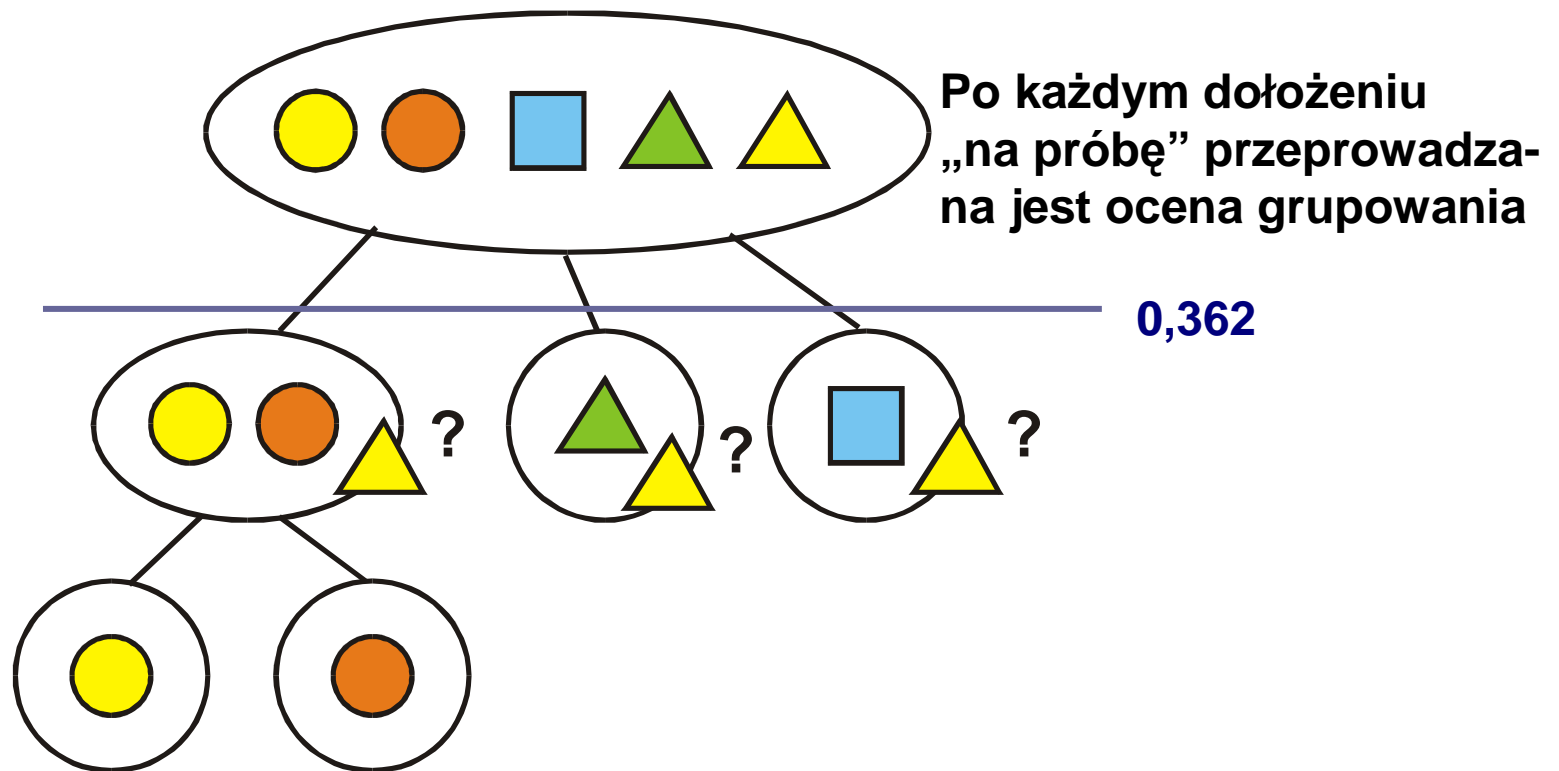


COBWEB – operacje

- Na każdym z poziomów drzewa algorytm może wykonać jedną z operacji:
 - zaliczyć przykład do istniejącej podkategorii,
 - utworzyć osobną podkategorię dla przykładu,
 - dokonać podziału podkategorii i zaliczyć przykład do jednej z jej kategorii potomnych,
 - połączyć dwie podkategorie i zaliczyć przykład do podkategorii połączonej,
- O tym, którą operację wykonać, decyduje wartość funkcji oceny jakości grupowania.

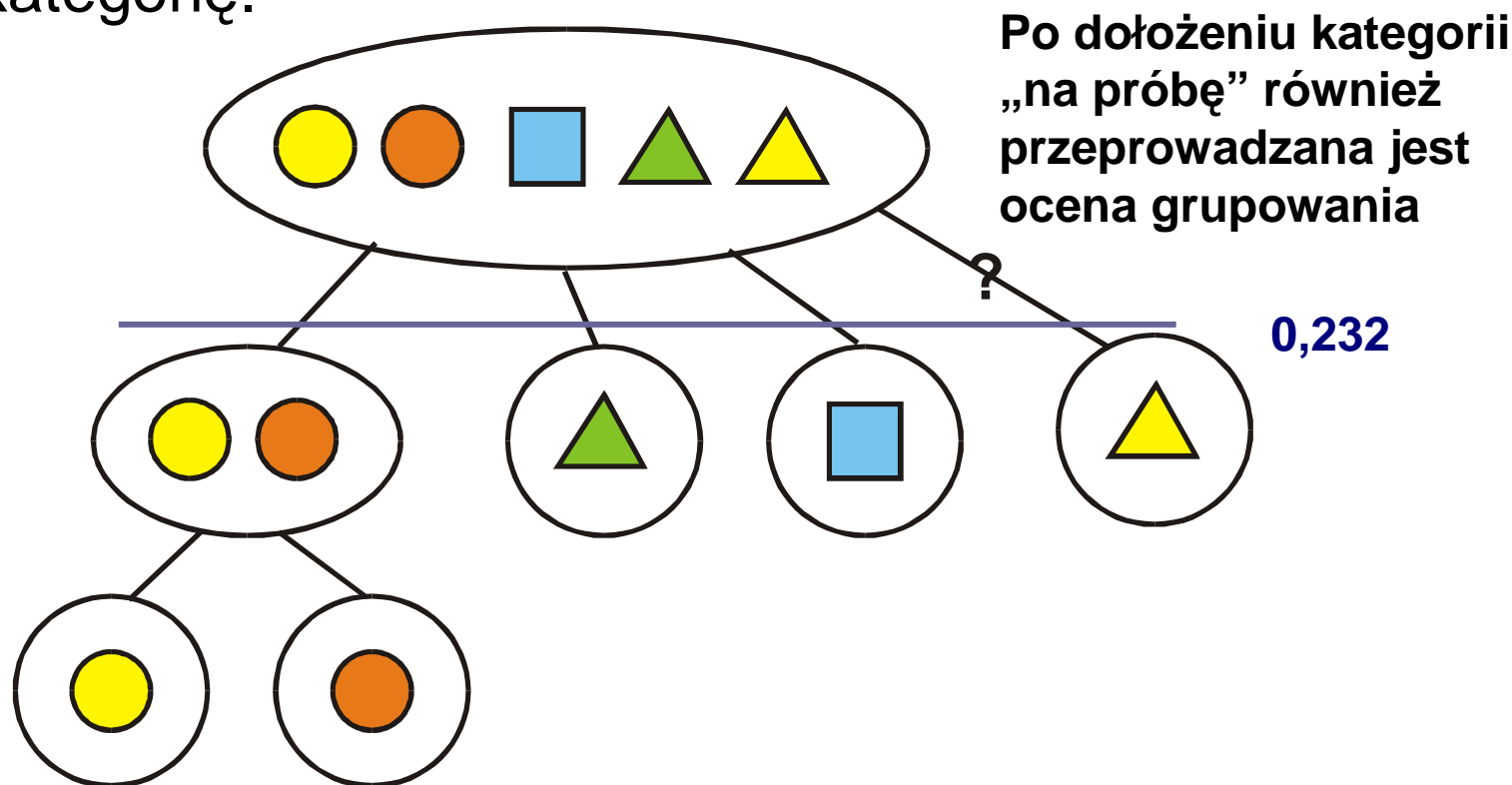
COBWEB – grupowanie

Algorytm dokłada „na próbę” przykład do każdej podkategorii.



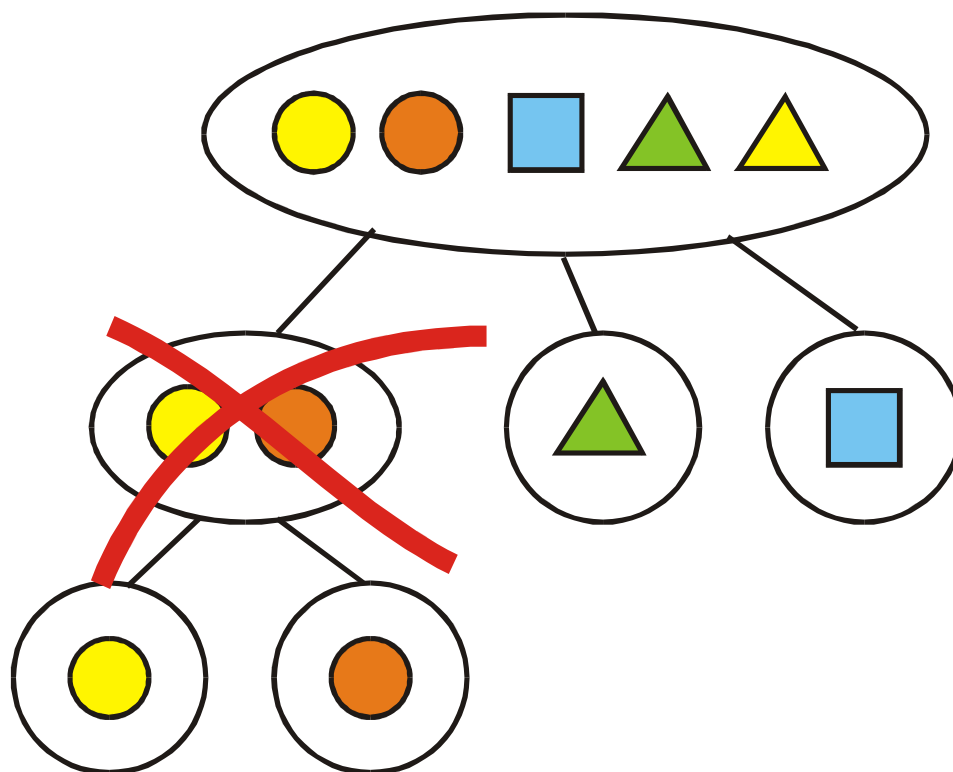
COBWEB – grupowanie

Algorytm próbuje też stworzyć dla przykładu osobną podkategorię.



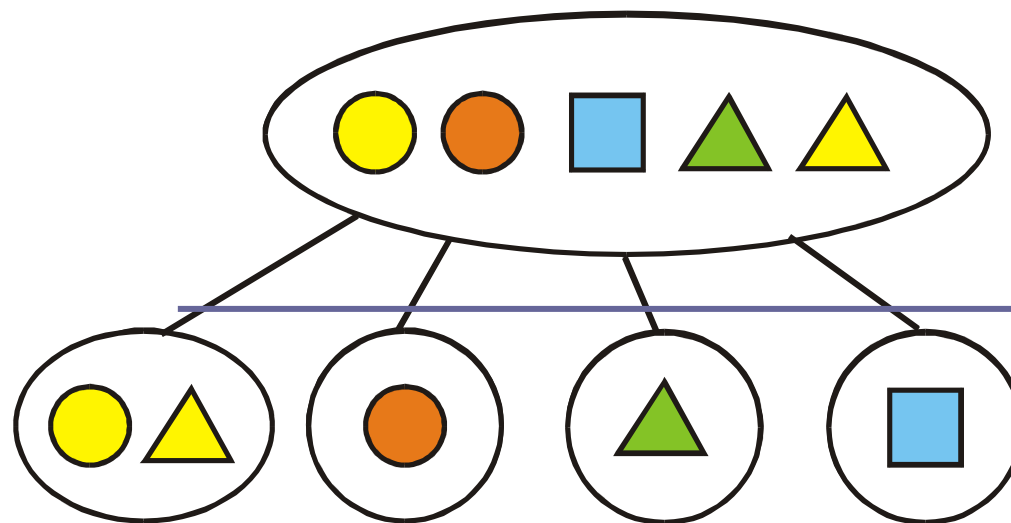
COBWEB – podział

COBWEB próbuje też usunąć jedną z podkategorii i dodać przykład do jednej z jej kategorii potomnych.



COBWEB – podział

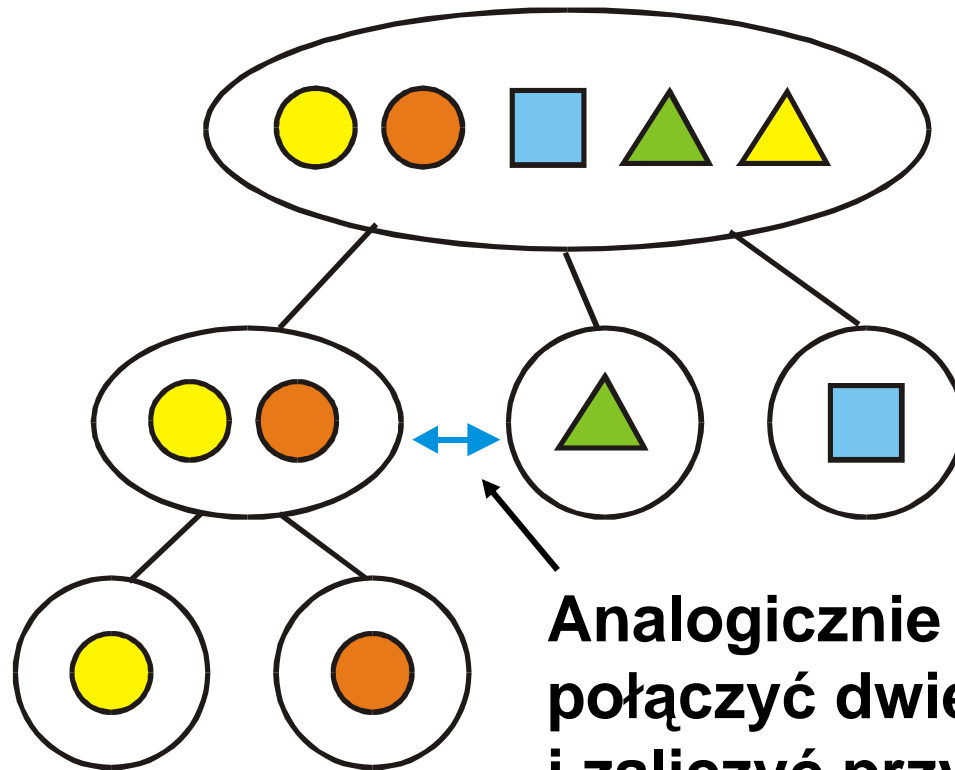
COBWEB próbuje też usunąć jedną z podkategorii i dodać przykład do jednej z jej kategorii potomnych.



Oczywiście po tych posunięciach (nadal wykonywanych „na próbę”) obliczamy ocenę grupowania

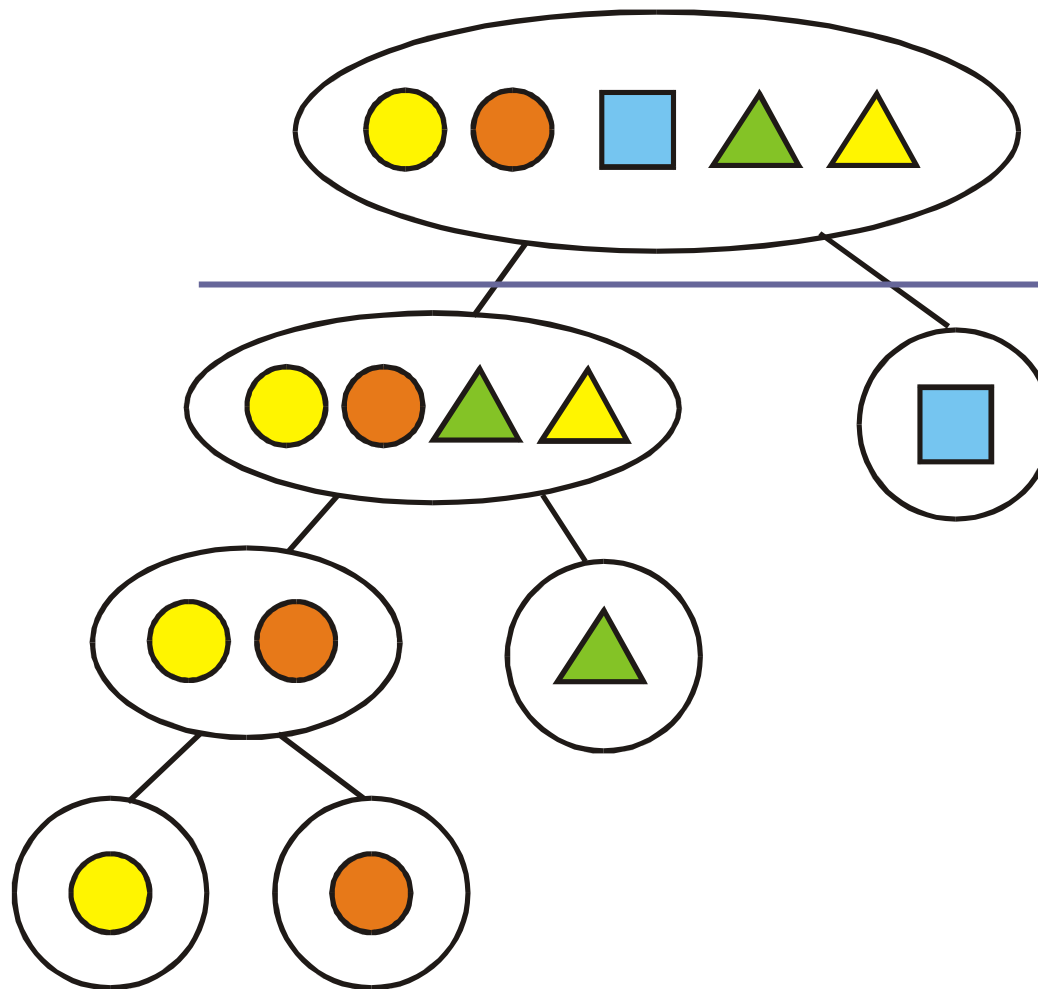
0,326

COBWEB – scalenie



**Analogicznie COBWEB może
połączyć dwie kategorie
i zaliczyć przykład do kategorii
połączonej**

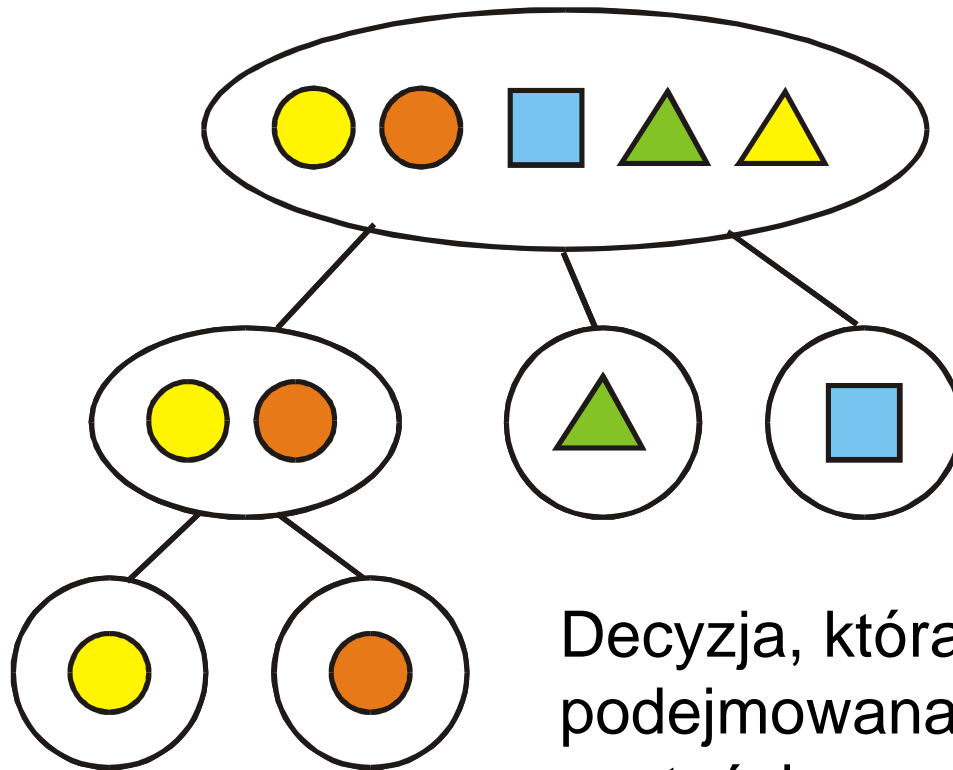
COBWEB – scalenie



0,428

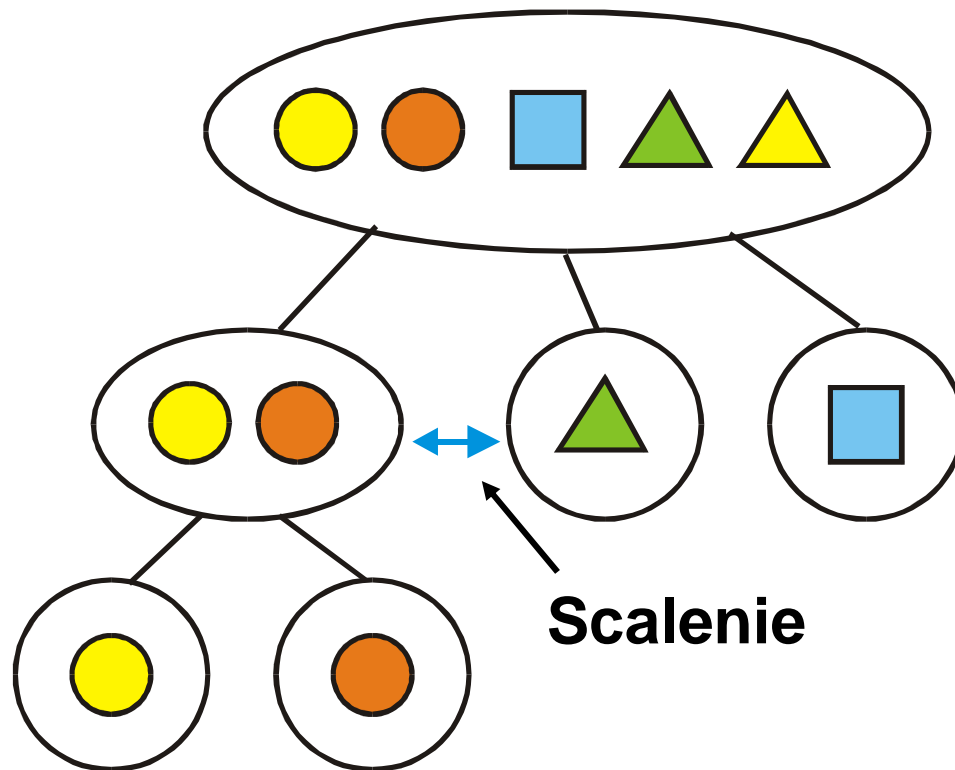
Po tym, ostatnim już,
posunięciu „na
próbę” obliczamy
ocenę grupowania

COBWEB – przykład

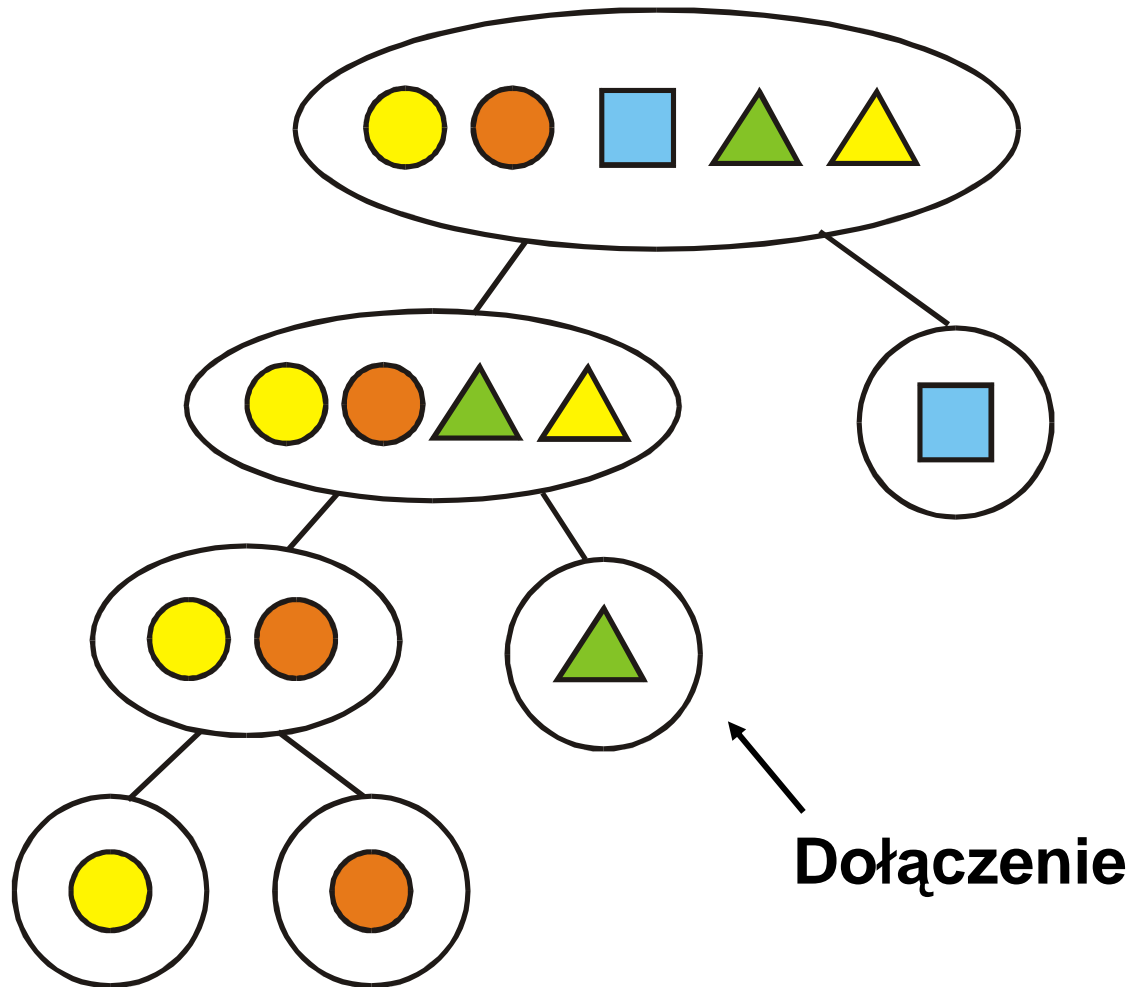


Decyzja, którą operację wykonać podejmowana jest na podstawie wartości oceny grupowania na każdym poziomie drzewa.

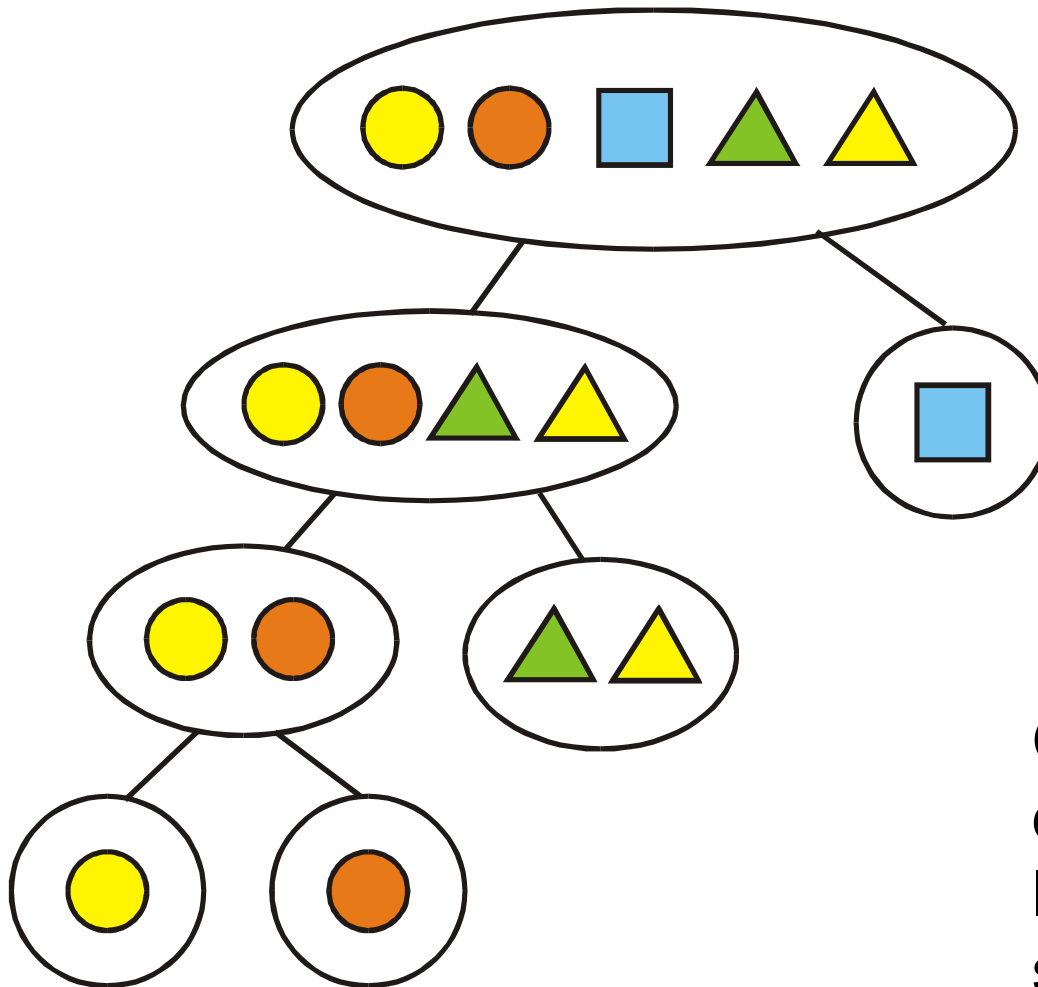
COBWEB – przykład



COBWEB – przykład

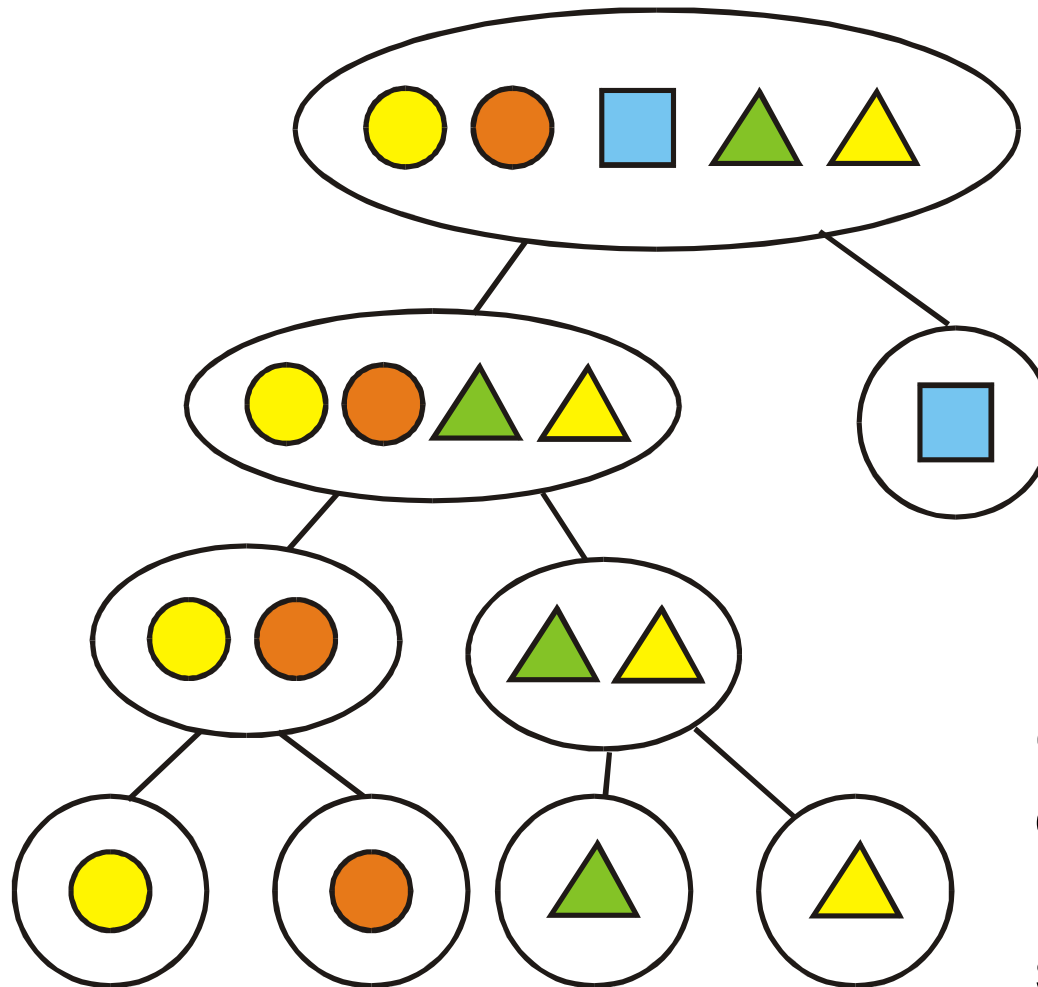


COBWEB – przykład



Gdy przykład dołączany jest do liścia, generowane są dwa nowe liście

COBWEB – przykład



Gdy przykład dołączany jest do liścia, generowane są dwa nowe liście

COBWEB – ocena grupowania

$$f(h) = \frac{1}{|C_h|} \sum_{d \in C_h} \Pr_{x \in \Omega}(h(x)=d).$$

związłość

$$\left[\sum_{i=1}^n \sum_{v \in A_i} \Pr_{x \in \Omega}(a_i(x)=v | h(x)=d) \right]^2 - \sum_{i=1}^n \sum_{v \in A_i} \Pr_{x \in \Omega}(a_i(x)=v)^2$$

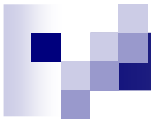
precyzja



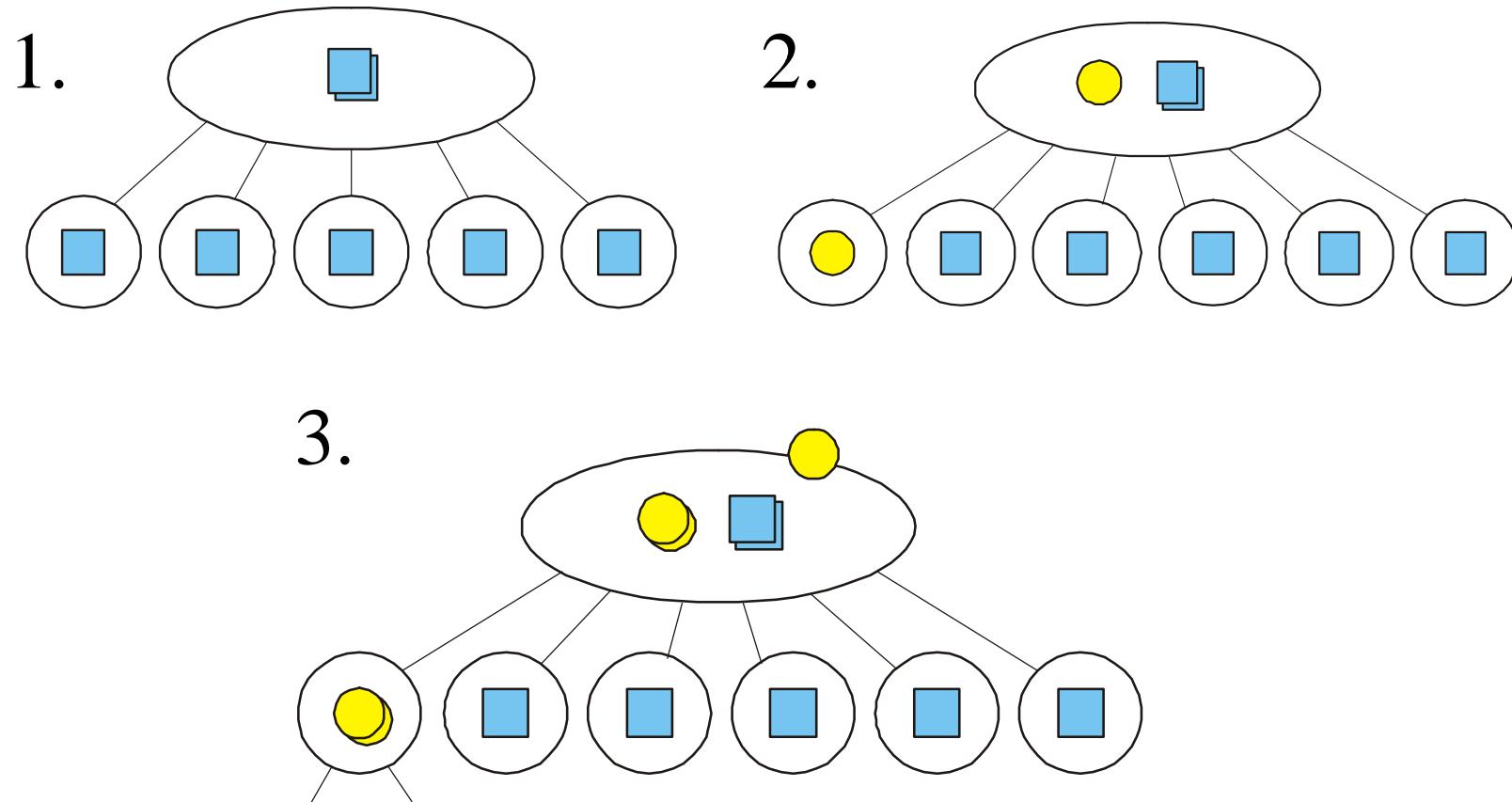


COBWEB – zalety

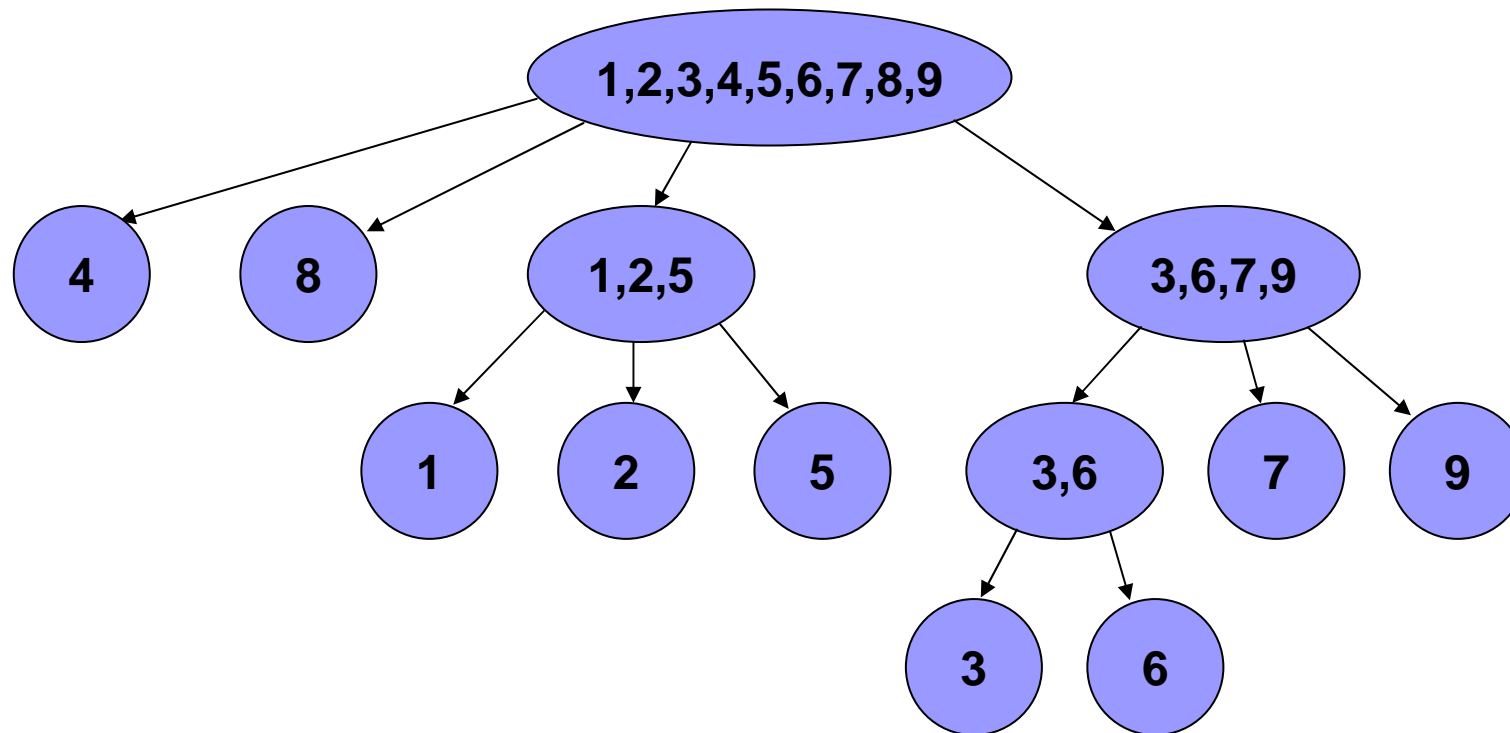
- COBWEB może być używany w trybie inkrementacyjnym,
- Produkowany jest hierarchiczny podział przykładów,
- Funkcja oceny jakości grupowania może być dostosowana do grupowania niestandardowych typów atrybutów lub nawet niestandardowych przykładów



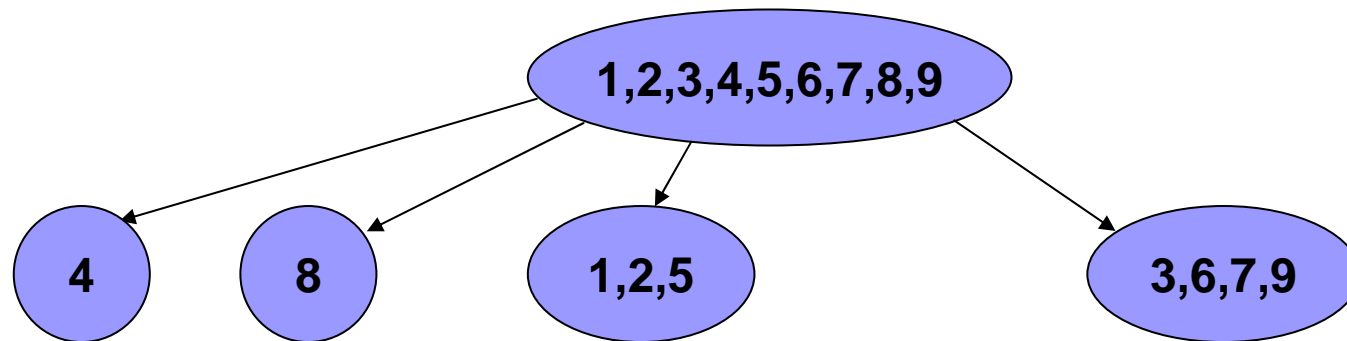
COBWEB – wada



COBWEB – przykład „Z.K.”



COBWEB – klasyfikacja



W praktyce jako wynik grupowania traktuje się drugi poziom drzewa COBWEB licząc od korzenia

Takie zredukowane drzewo może być traktowane również jako klasyfikator

Grupowanie - przykład

| S.C. | D.O.R. | Wiek | Wykształcenie | Sam. | Z.K. |
|------|--------|------|---------------|------|------|
| S | 800 | 32 | wyższe | tak | tak |
| S | 1200 | 35 | średnie | tak | tak |
| S | 700 | 26 | podstawowe | nie | nie |
| M | 600 | 45 | wyższe | nie | tak |
| M | 650 | 38 | średnie | tak | tak |
| S | 900 | 28 | wyższe | nie | nie |
| S | 1100 | 65 | średnie | tak | nie |
| M | 500 | 22 | średnie | nie | nie |
| S | 800 | 43 | podstawowe | tak | nie |



COBWEB – podsumowanie

- COBWEB dołącza każdy przykład do drzewa kategorii wykonując proste operacje,
- O tym, którą operację wykonać, decyduje funkcja oceny jakości grupowania według kryteriów precyzji i zwięzłości,
- Działa w trybie inkrementacyjnym, można dostosowywać sposób jego działania,
- Algorytm zachłanny, efekt jego działania mocno zależy od kolejności podawania przykładów na wejście



Inne popularne metody grupowania

- K-means:

- Iteracyjne grupowanie na zasadzie odnajdywania centrów podzbiorów przykładów,

- EM:

- Próba odnalezienia optymalnego globalnie podziału poprzez poszukiwanie parametrów rozkładów Gaussa.



Inżynieria wejścia i wyjścia



Ocena wyjścia

- Omówione metody budują klasyfikatory, które osiągają bardzo dobre wyniki na *zbiorze uczącym*,
- Często okazuje się, że klasyfikator osiągający dobry wynik na zbiorze uczącym ma bardzo słabe właściwości predykcyjne,
- Zjawisko to nazywamy *błędem nadmiernego dopasowania*.

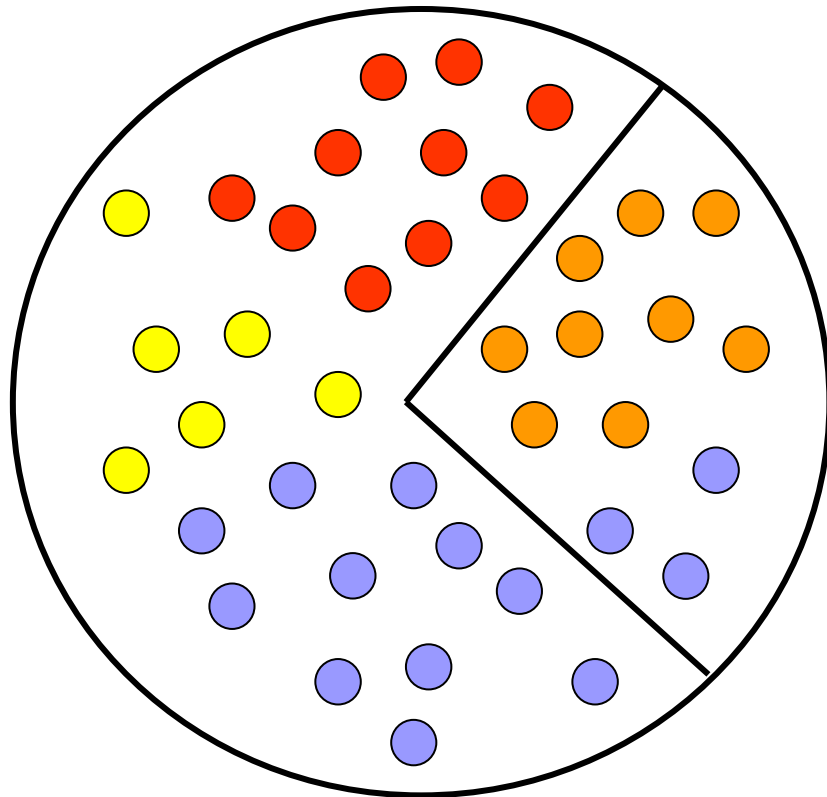


Błąd nadmiernego dopasowania

| Nazwisko | Wiek | Wykształcenie | Sam. | Z.K. |
|----------|------|---------------|------|------|
| Abacki | 32 | wyższe | tak | tak |
| Babacka | 35 | średnie | tak | tak |
| Cabacki | 26 | podstawowe | nie | nie |
| Dabacka | 45 | wyższe | nie | tak |
| Ebacki | 38 | średnie | tak | tak |
| Fabacki | 28 | wyższe | nie | nie |
| Gabacka | 65 | średnie | tak | nie |
| Habacka | 22 | średnie | nie | nie |
| Ibacki | 43 | podstawowe | tak | nie |

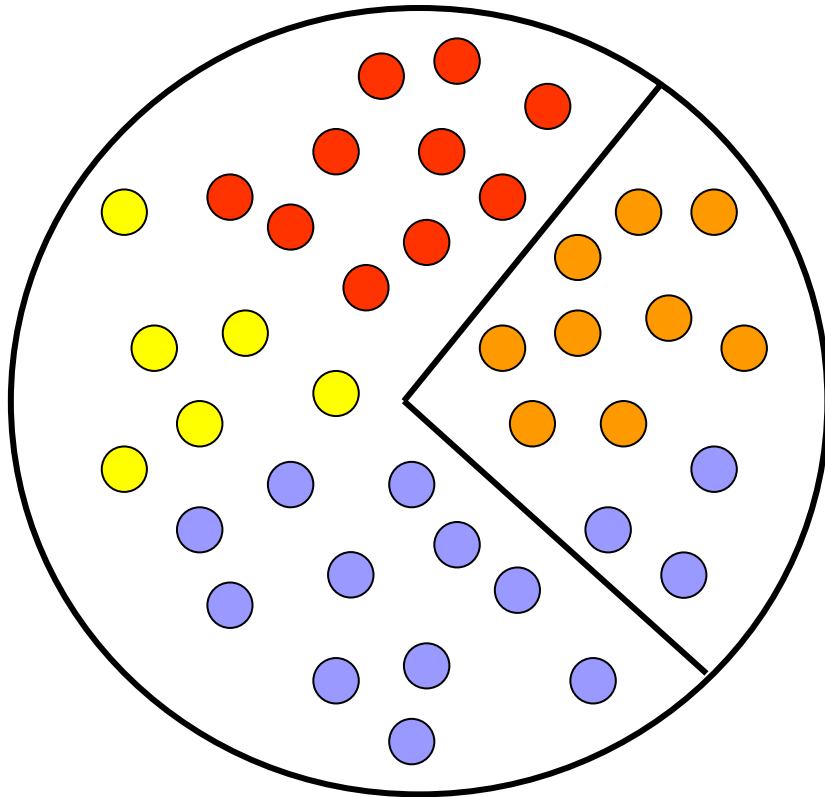
```
if Nazwisko=Abacki then Z.K.=tak  
if Nazwisko=Babacka then Z.K.=tak  
if Nazwisko=Cabacki then Z.K.=nie  
...
```

Zbiór uczący i weryfikujący



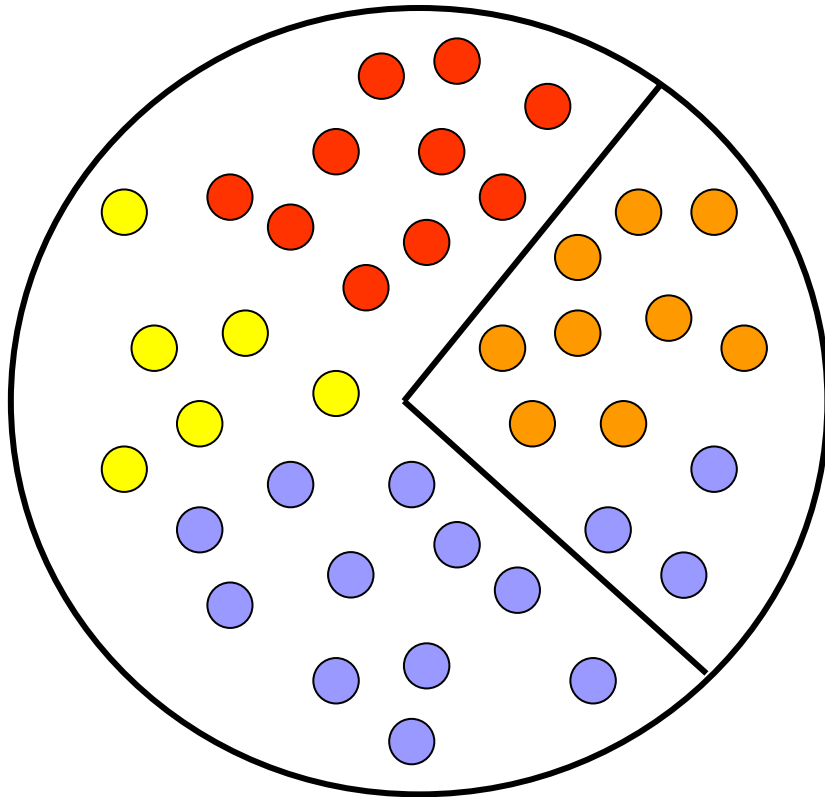
- Standardowe podejście do oceny klasyfikatora to podział zbioru przykładów (zbioru uczącego) na *trenujący* i *testujący*

Zbiór uczący i weryfikujący (2)



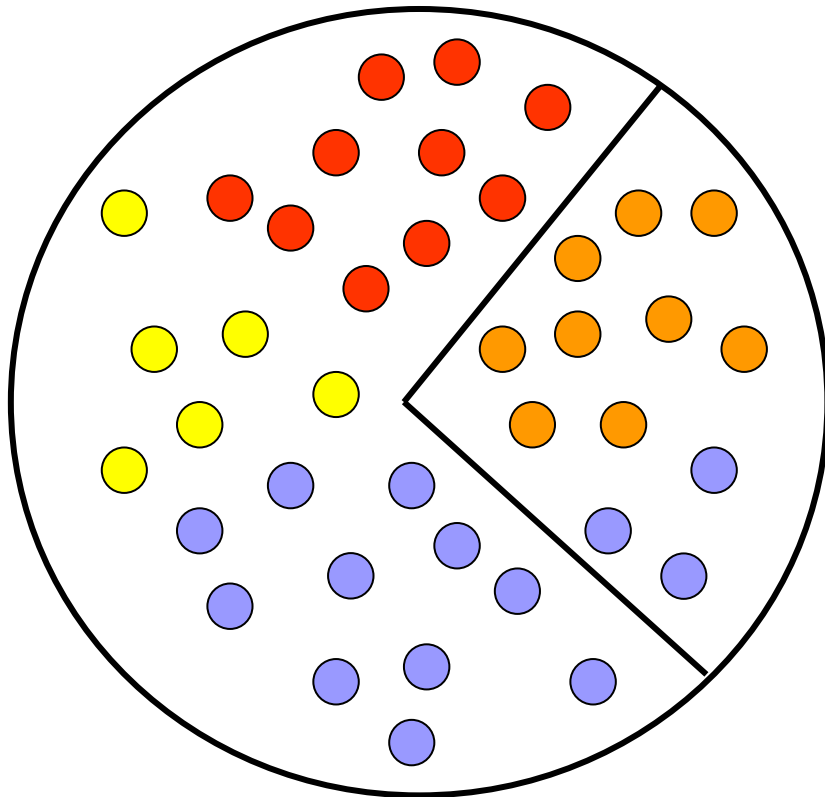
- Klasyfikator budujemy na podstawie zbioru trenującego, a następnie badamy jego skuteczność na zbiorze testującym

Zbiór uczący i weryfikujący (3)



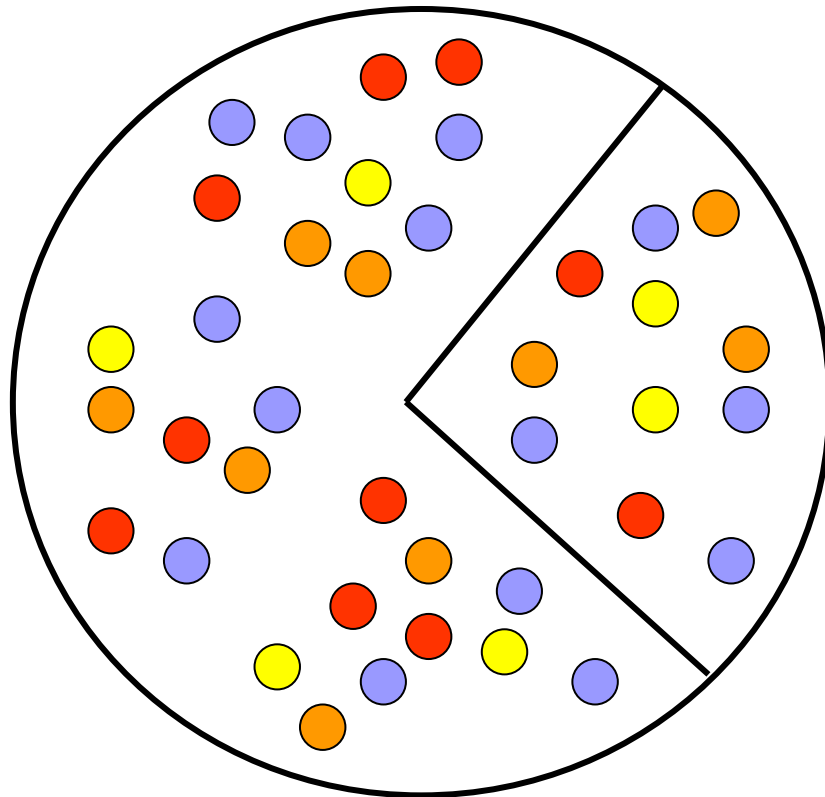
- Podejście to stosujemy tylko do oceny siły predykcyjnej,
- Właściwy klasyfikator budujemy używając całego zbioru przykładów.

Stratyfikacja



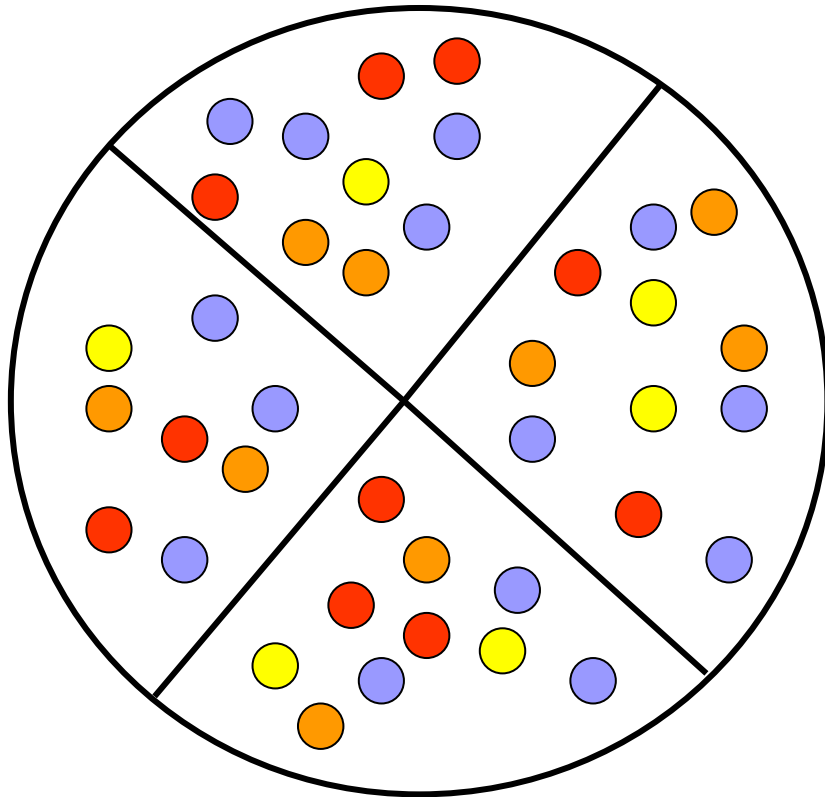
- Przy wyborze zbioru trenującego istnieje niebezpieczeństwo nierównomiernej reprezentacji niektórych klas,
- Temu efektowi zapobiega *stratyfikacja*.

Efekt stratyfikacji



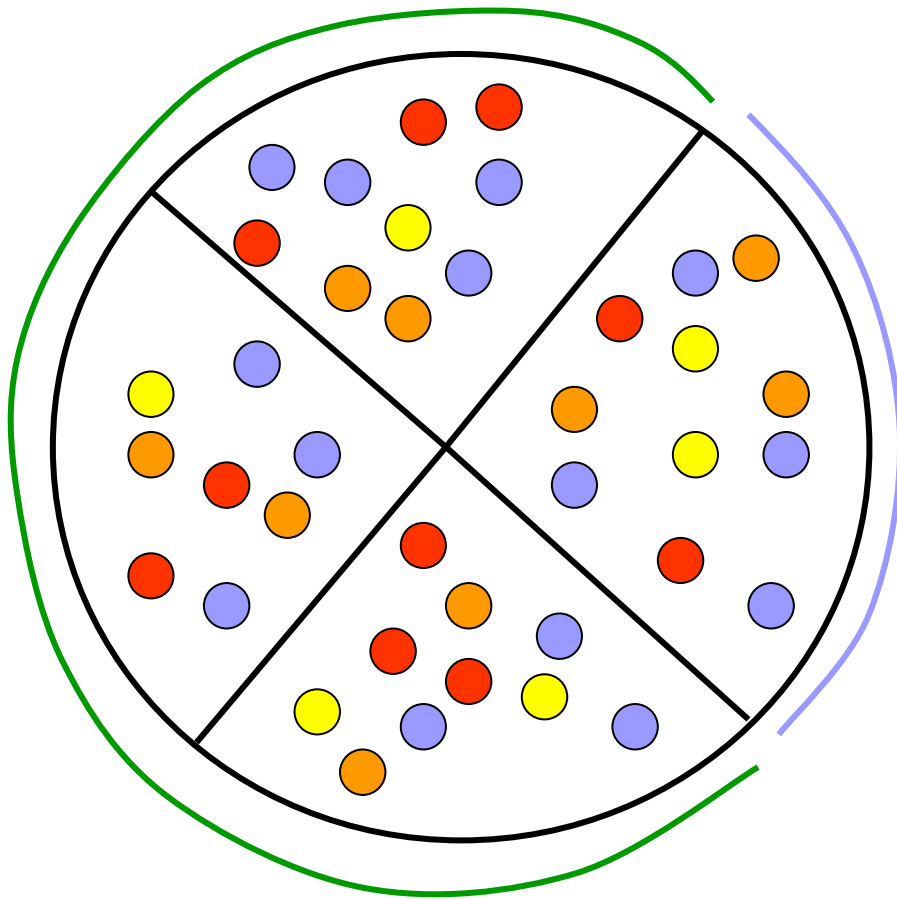
- Stratyfikacja polega na dokonywaniu takich podziałów zbioru przykładów, aby klasy były równomiernie reprezentowane w każdej jego części

Podział na n części



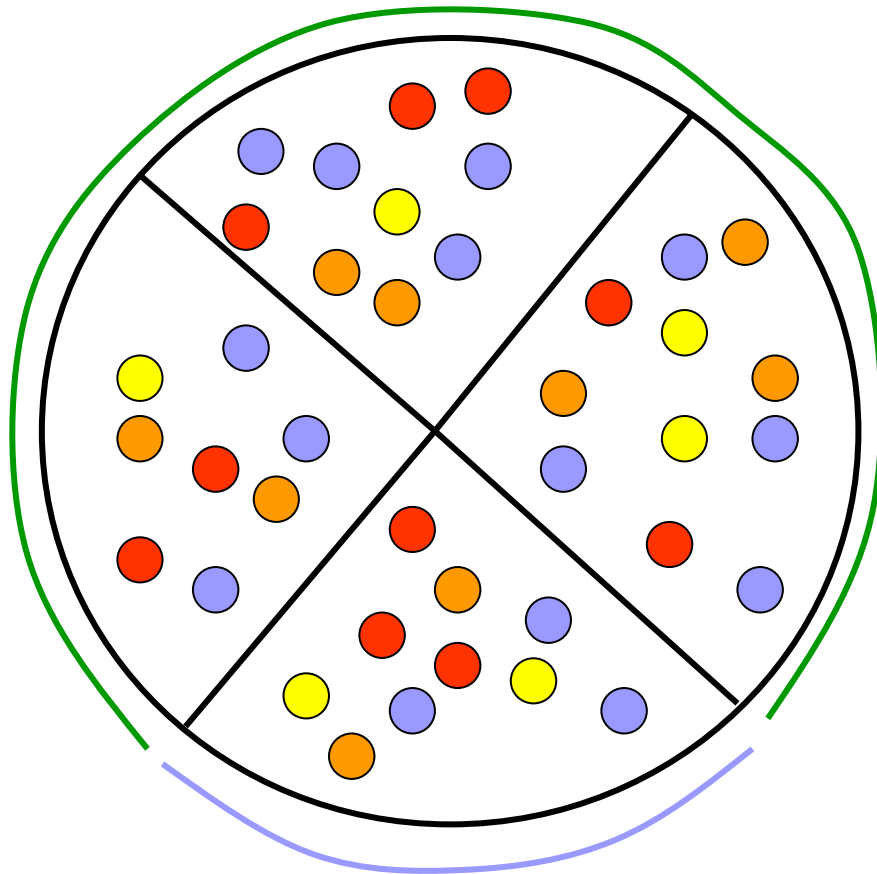
- Pomimo dokonania stratyfikacji, wciąż możemy mieć wątpliwości co do wyboru zbioru testującego,
- Odpowiedzią może być podział zbioru na n części

Podział na n części (2)



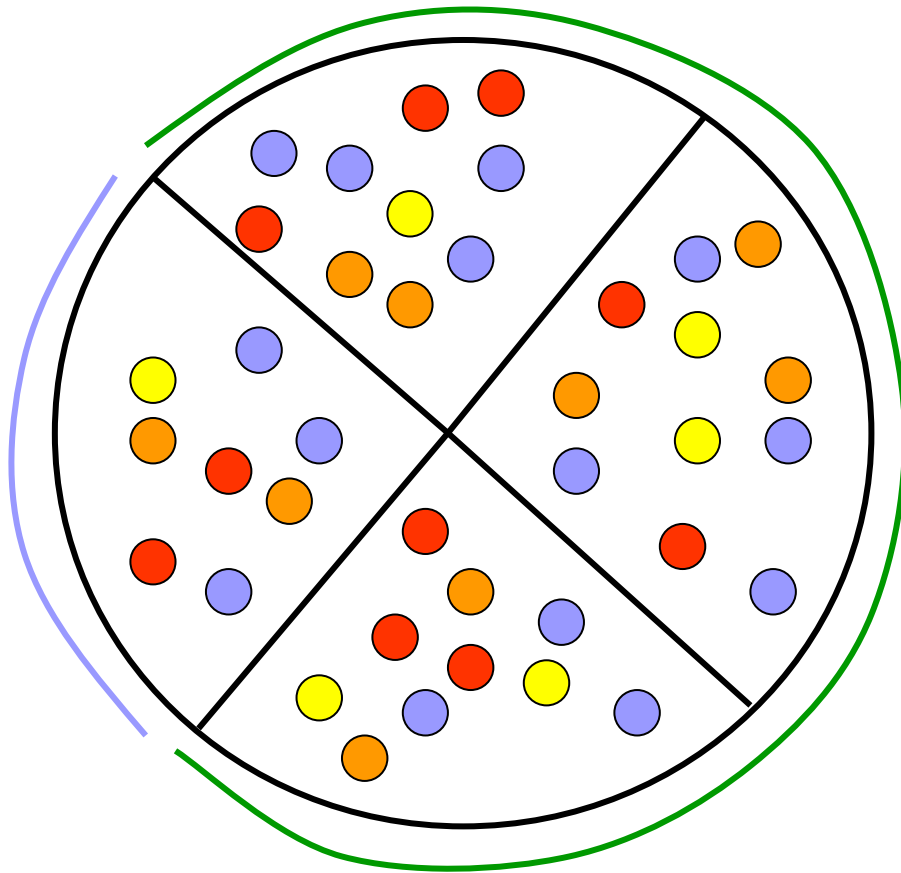
- Przy zastosowaniu tego podejścia proces uczenia powtarzamy n razy,
- Przy każdym powtórzeniu jedną część zbioru (tu: $\frac{1}{4}$) przyjmujemy za zbiór testujący, resztę jako trenujący (tu: $\frac{3}{4}$)

Podział na n części (2)



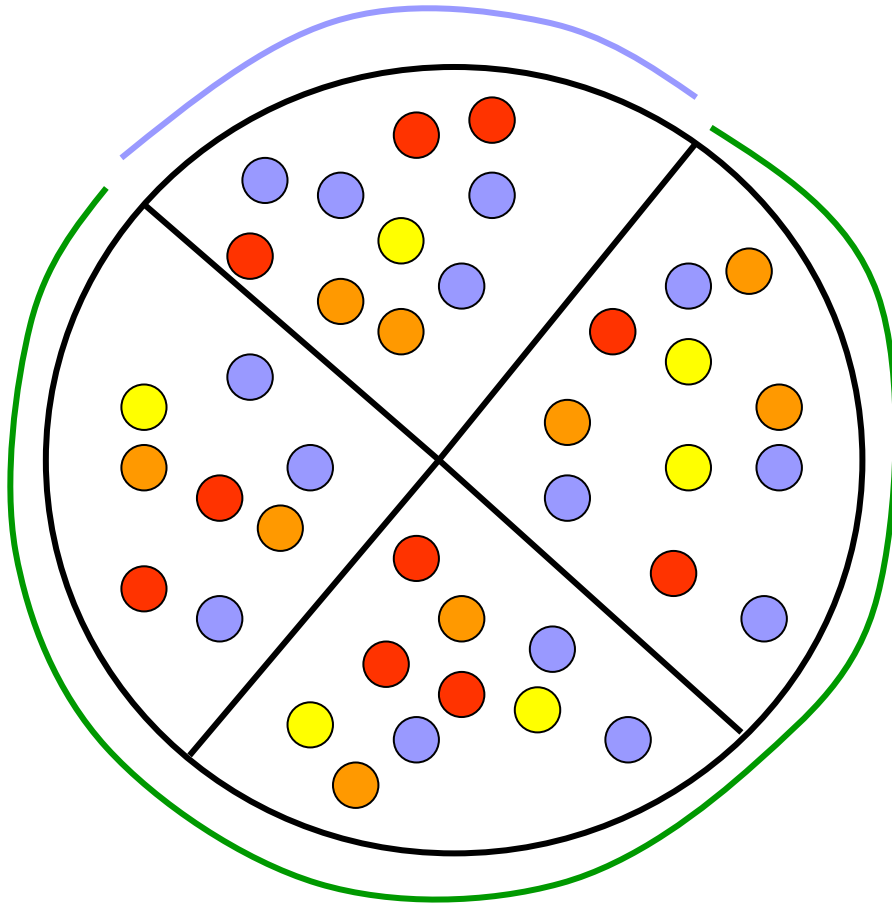
- Przy zastosowaniu tego podejścia proces uczenia powtarzamy n razy,
- Przy każdym powtórzeniu jedną część zbioru (tu: $\frac{1}{4}$) przyjmujemy za zbiór testujący, resztę jako trenujący (tu: $\frac{3}{4}$)

Podział na n części (2)



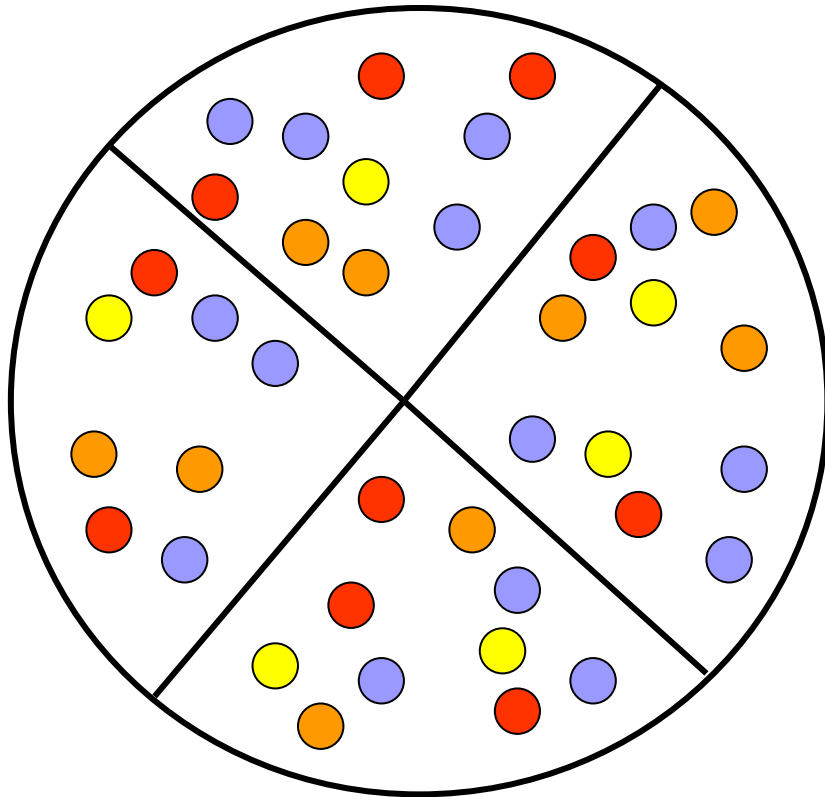
- Przy zastosowaniu tego podejścia proces uczenia powtarzamy n razy,
- Przy każdym powtórzeniu jedną część zbioru (tu: $\frac{1}{4}$) przyjmujemy za zbiór testujący, resztę jako trenujący (tu: $\frac{3}{4}$)

Podział na n części (2)



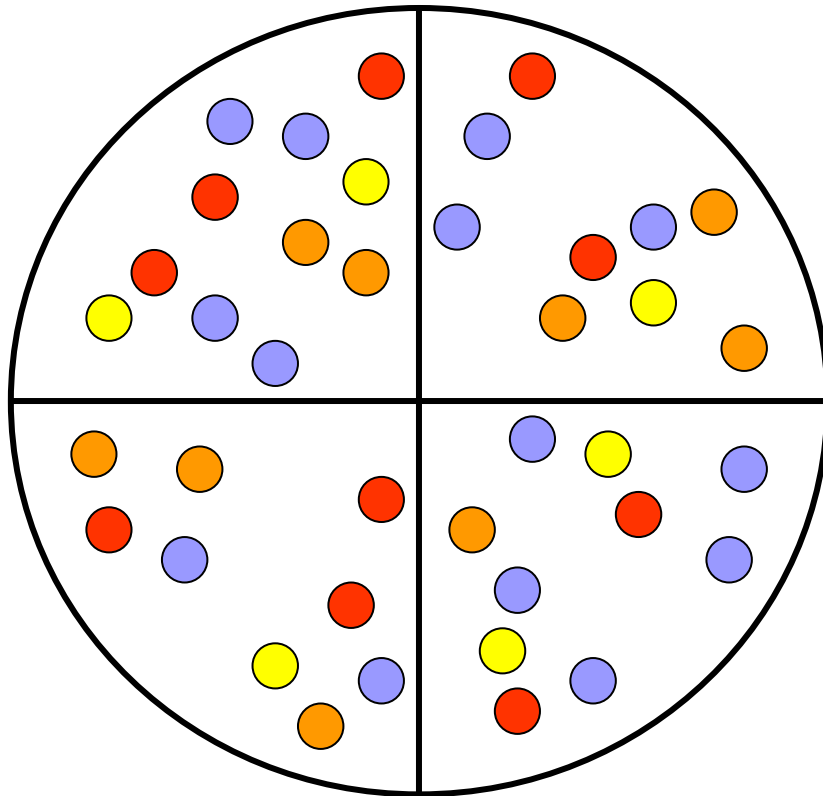
- Przy zastosowaniu tego podejścia proces uczenia powtarzamy n razy,
- Przy każdym powtórzeniu jedną część zbioru (tu: $\frac{1}{4}$) przyjmujemy za zbiór testujący, resztę jako trenujący (tu: $\frac{3}{4}$)

m-krotne dokonanie podziału



- Można jednak wskazać same granice podziału jako potencjalne źródło błędów,
- Rozwiązanie może być *m*-krotne powtórzenie podziału na *n* części,
- Proces uczenia powtarzamy wówczas $m \cdot n$ razy

m-krotne dokonanie podziału



- Można jednak wskazać same granice podziału jako potencjalne źródło błędów,
- Rozwiązanie może być *m*-krotne powtórzenie podziału na *n* części,
- Proces uczenia powtarzamy wówczas $m \cdot n$ razy



Ocena jakości klasyfikatora

- Opisana metoda jest jedną z najpopularniejszych metod oceny klasyfikatorów:

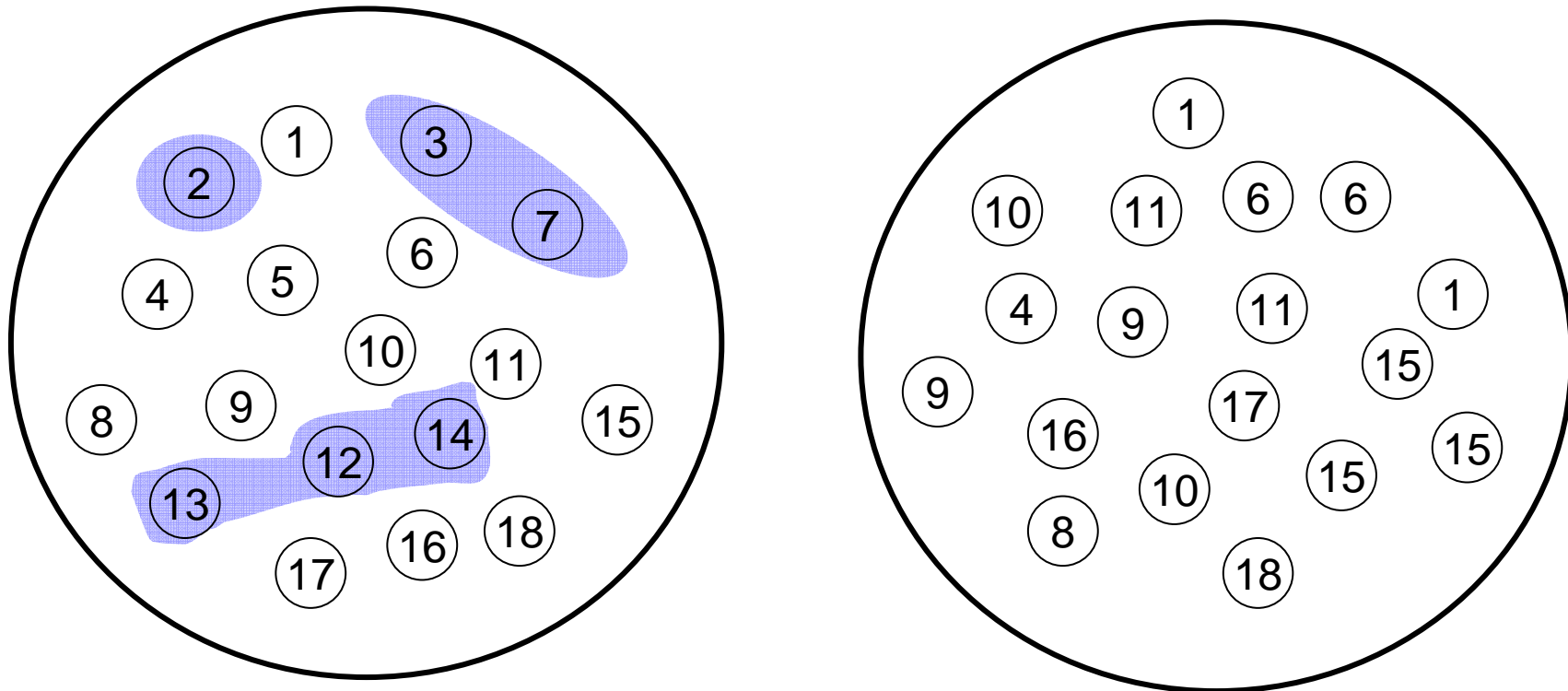
***m n*-fold cross-validation with stratification**

- Najczęściej przyjmujemy $m=n=10$ otrzymując:

ten ten-fold cross-validation with stratification

- Metoda ta wymaga jednak wielu powtórzeń procesu uczenia i dlatego czasem stosowane są metody mniej zasobochłonne.

0.632 bootstrap

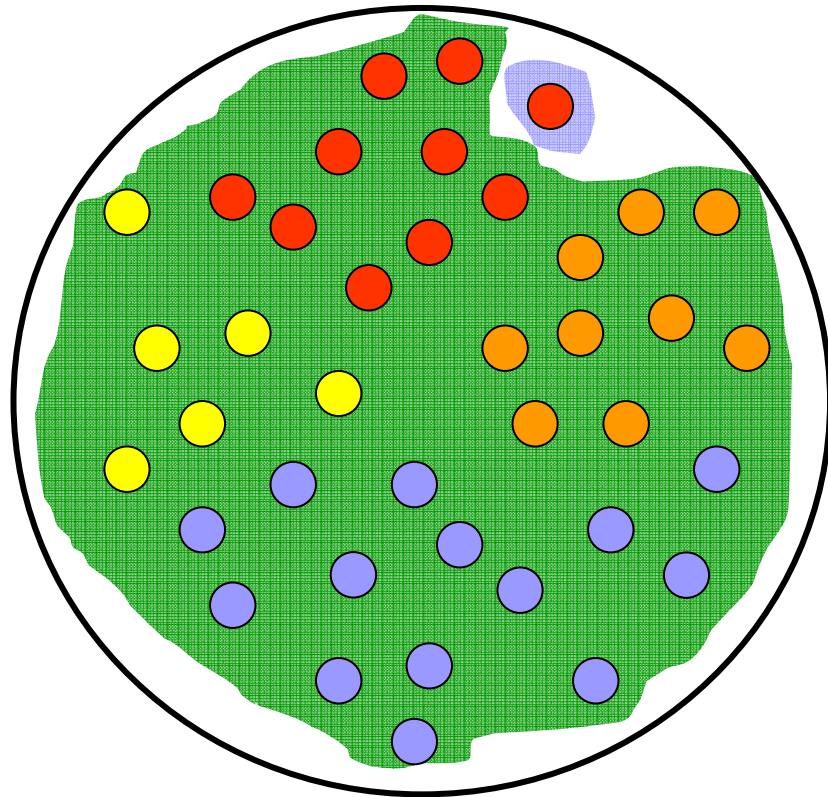


**k razy wybieramy losowo przykład, który umieszczamy w zbiorze trenującym
(k – moc zbioru przykładów, tu: 18)**

Przykłady niewybrane ani razu stają się naszym zbiorem testującym

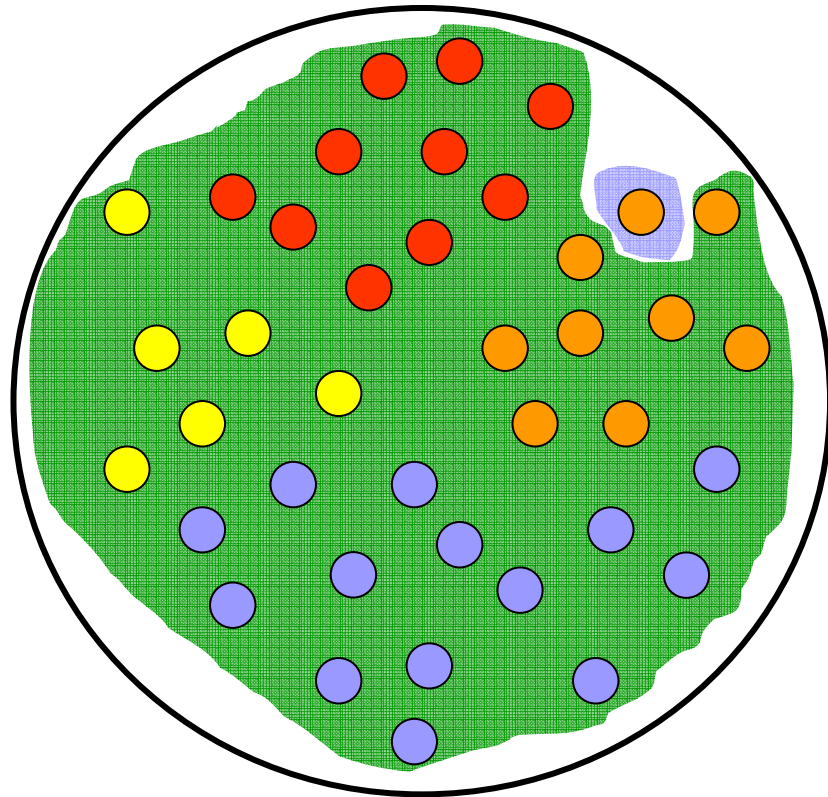
Liczba przykładów niewybranych stanowi statystycznie 0.368 k (wybranych 0.632)

Leave-one-out



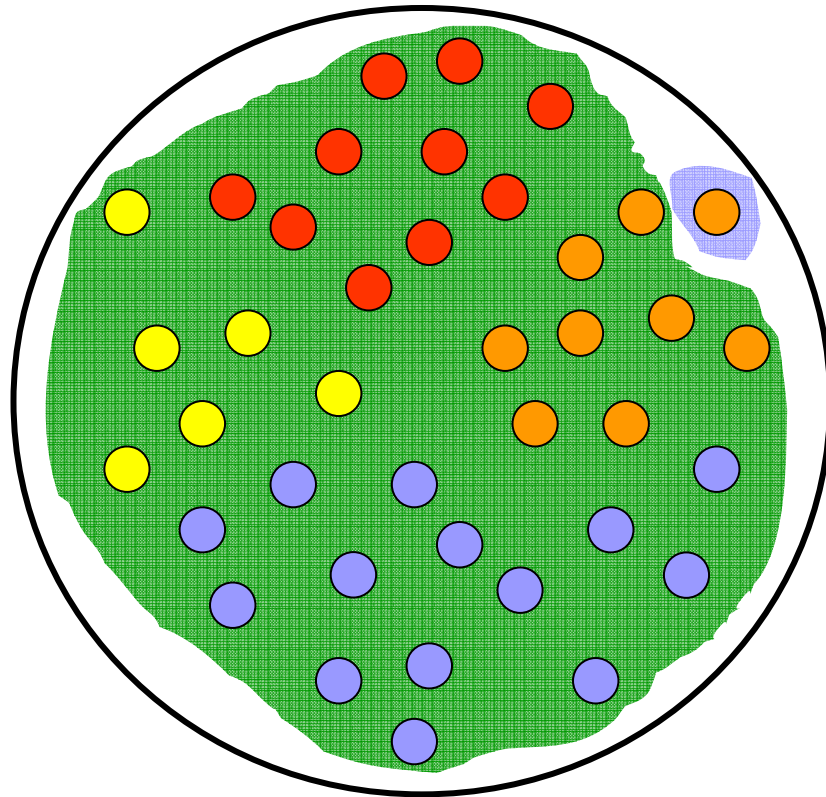
- Tutaj za zbiór testujący przyjmujemy jeden przykład,
- Proces uczenia powtarzamy k razy, ale nie musimy m razy „dokonywać podziału” ani wykonywać kosztownego procesu stratyfikacji

Leave-one-out



- Tutaj za zbiór testujący przyjmujemy jeden przykład,
- Proces uczenia powtarzamy k razy, ale nie musimy m razy „dokonywać podziału” ani wykonywać kosztownego procesu stratyfikacji

Leave-one-out



itd.

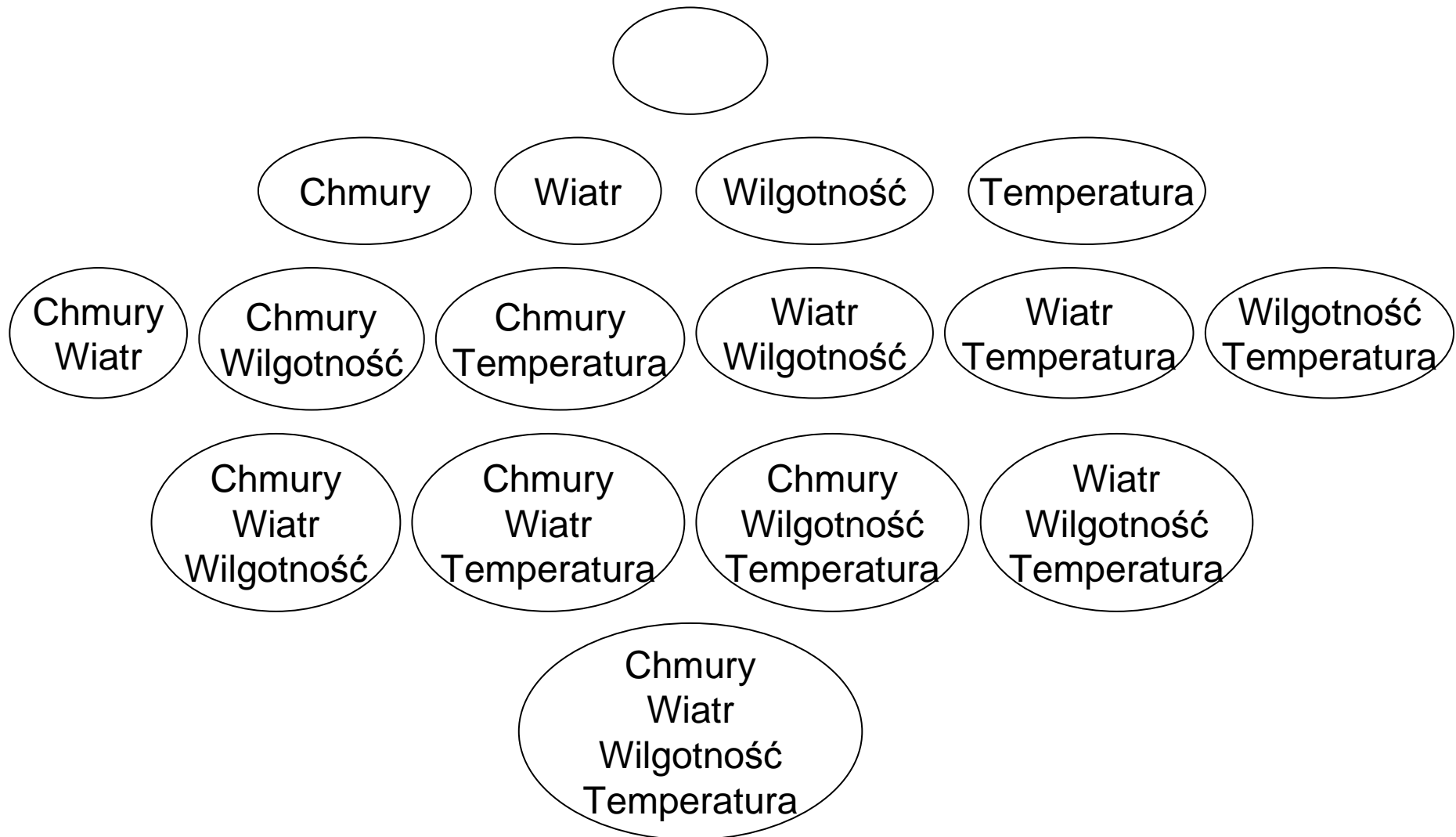
- Tutaj za zbiór testujący przyjmujemy jeden przykład,
- Proces uczenia powtarzamy k razy, ale nie musimy m razy „dokonywać podziału” ani wykonywać kosztownego procesu stratyfikacji



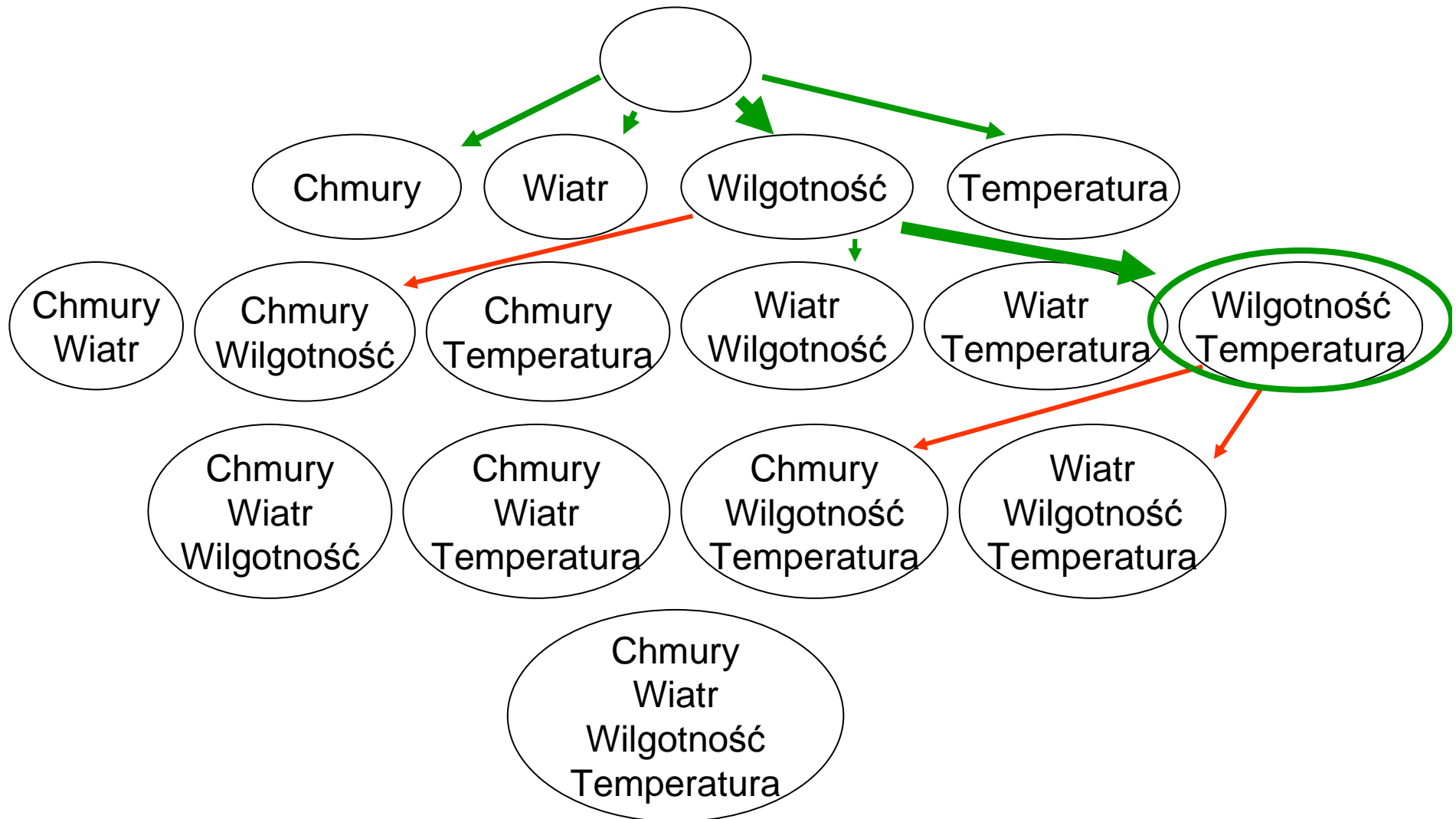
Techniki doboru atrybutów

- Ze względu na zazwyczaj zachłanny charakter metod tworzenia klasyfikatorów, zmniejszenie liczby atrybutów może często doprowadzić do uzyskania lepszych efektów,
- Stosuje się dobór atrybutów *naprzód* i *wstecz* metodą zachłanną.

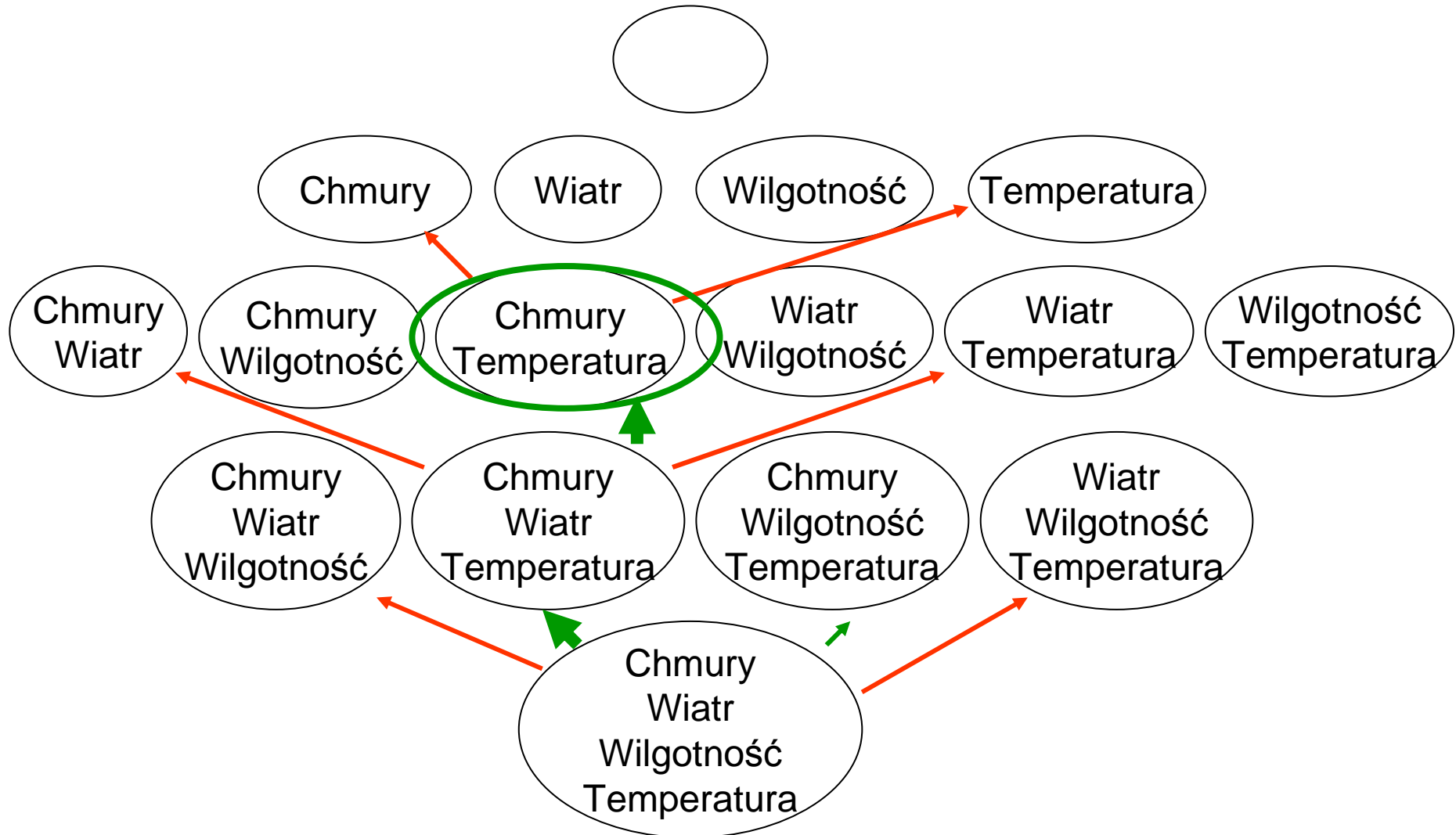
Przestrzeń atrybutów



Dobór atrybutów *naprzód*



Dobór atrybutów *wstecz*

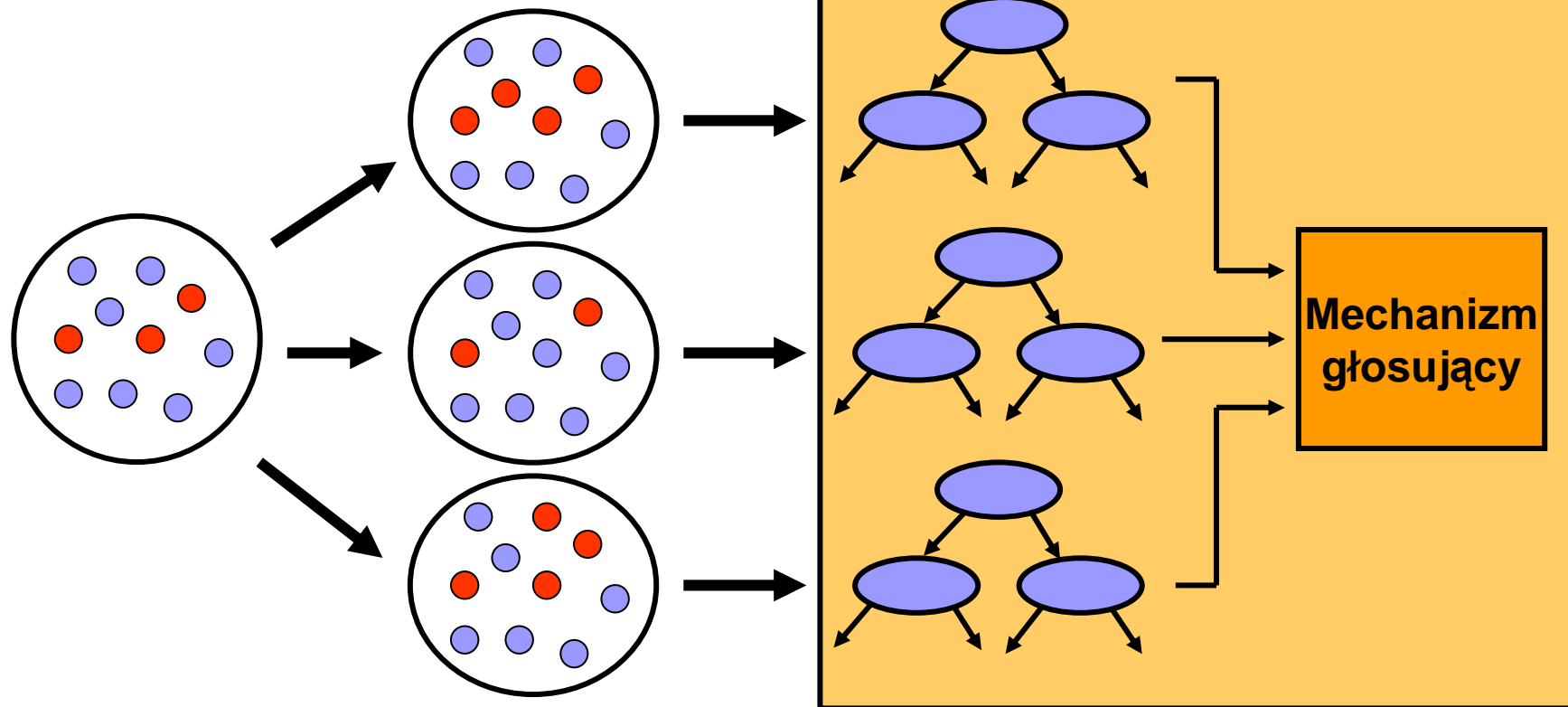




Łączenie kilku klasyfikatorów

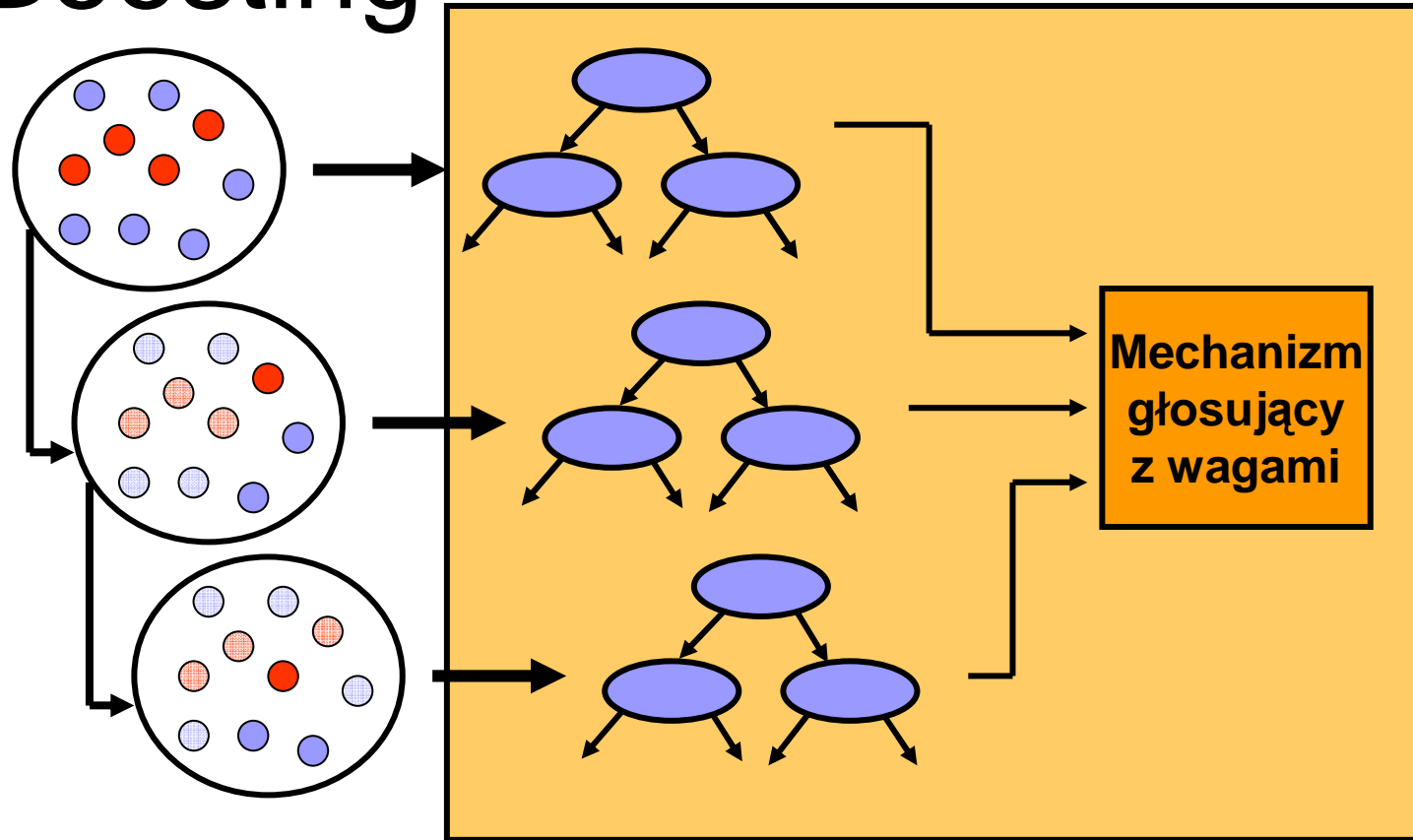
- W celu poprawy zdolności predykcyjnych można spróbować połączenia kilku klasyfikatorów w jeden,
- Istnieje kilka metod łączenia klasyfikatorów.

Bagging



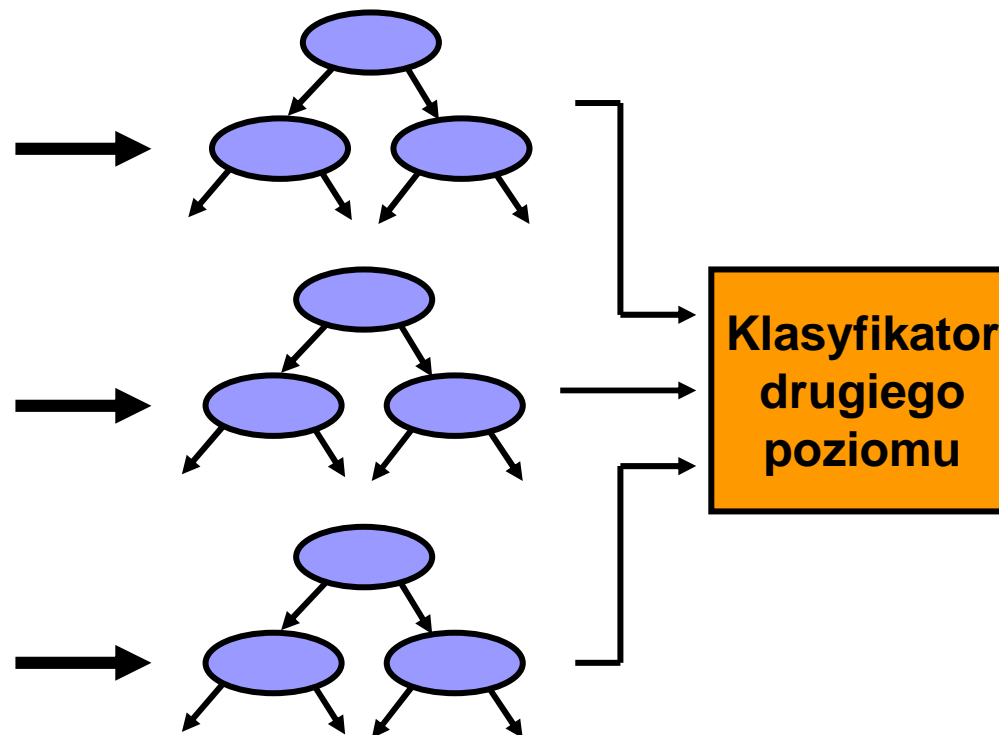
1. Z jednego zbioru przykładów tworzymy n zbiorów na zasadzie bootstrap
2. Na podstawie każdego zbioru tworzymy klasyfikator cząstkowy
3. Klasyfikator zagregowany tworzymy, dodając mechanizm głosujący

Boosting



1. Budujemy klasyfikator na podstawie zbioru uczącego
2. Przykładom źle sklasyfikowanym zwiększamy wagę; budujemy klasyfikator
3. Iteracyjnie powtarzamy krok 2 do osiągnięcia warunku stopu
4. Dodajemy mechanizm głosujący z wagami

Stacking



1. Tworzymy n klasyfikatorów

2. Dodajemy na ich wyjściu klasyfikator drugiego poziomu, który jest odpowiedzialny za decyzję, któremu klasyfikatorowi cząstkowemu zaufać

3. Istnieje wiele wariantów tej metody




Podsumowanie



Analiza danych

- Najlepsze skutki daje połączenie kilku metod i staranna analiza wyników,
- Przykładowy scenariusz może obejmować:
 - Grupowanie danych,
 - Wyznaczenie grup potencjalnie interesujących,
 - Bliższa eksploracja danych za pomocą reguł asocjacyjnych,
 - Określenie ważnych czynników i budowa klasyfikatorów decyzyjnych.



Przykładowe obszary zastosowań eksploracji danych

- **Biologia i genetyka:**

- Klasyfikacja struktur białkowych i fragmentów łańcuchów genetycznych,

- **Astronomia i astrofizyka:**

- Analiza danych dostarczanych z teleskopów i radioteleskopów,

- **Medycyna:**

- Klasyfikacja danych pacjentów dla wspomaganie procesu diagnozowania.



Kierunki rozwoju metod eksploracji danych

- Poprawa wydajności stosowanych metod w celu analizy coraz większych ilości danych,
- Modyfikacja metod w stronę analizy bardziej złożonych struktur danych (hipertekst, dane semistrukturalne),
- Rozwój technik wizualizacji wyników analizy.