Analiza danych z pakietem R

Uczenie maszynowe - wprowadzenie

Paweł Lula, Katedra Systemów Obliczeniowych,
Uniwersytet Ekonomiczny w Krakowie
pawel.lula@uek.krakow.pl

WPROWADZENIE DO UCZENIA MASZYNOWEGO

Uczenie maszynowe

- Uczenie maszynowe – zdolność systemu do polepszania jakości swojego działania w wyniku analizy zaprezentowanych danych uczących.

- Związki uczenia maszynowego z innymi dziedzinami wiedzy:
  - statystyka i analiza danych (data science),
  - matematyka,
  - optymalizacja,
  - informatyka (sztuczna inteligencja, bazy i hurtownie danych, pakiety obliczeniowe),
  - grafika komputerowa (wizualizacja danych).

Uczenie maszynowe

- Elementy
  - system poddawany uczeniu (model),
  - dane,
  - funkcja określająca jakość systemu,
  - sposób wprowadzania zmian w systemie (algorytm uczenia).

Model

- Służy do rozwiązania zadania określonego typu:
  - regresja,
  - klasyfikacja wzorcowa,
  - klasyfikacja bezwzorcowa (analiza skupień),
  - odkrywanie reguł asocjacyjnych,
  - redukcja wymiaru przestrzeni danych,
  - estymacja funkcji gęstości.

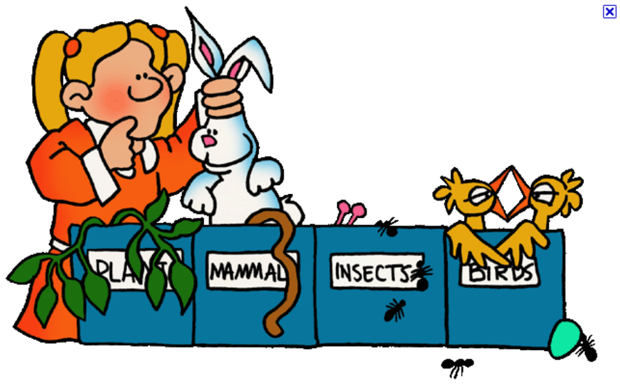
Regresja

Model regresyjny – model opisujący zależność pomiędzy zestawem zmiennych niezależnych (wejściowych) a zmienną zależną (wyjściową) mającą charakter ilościowy.

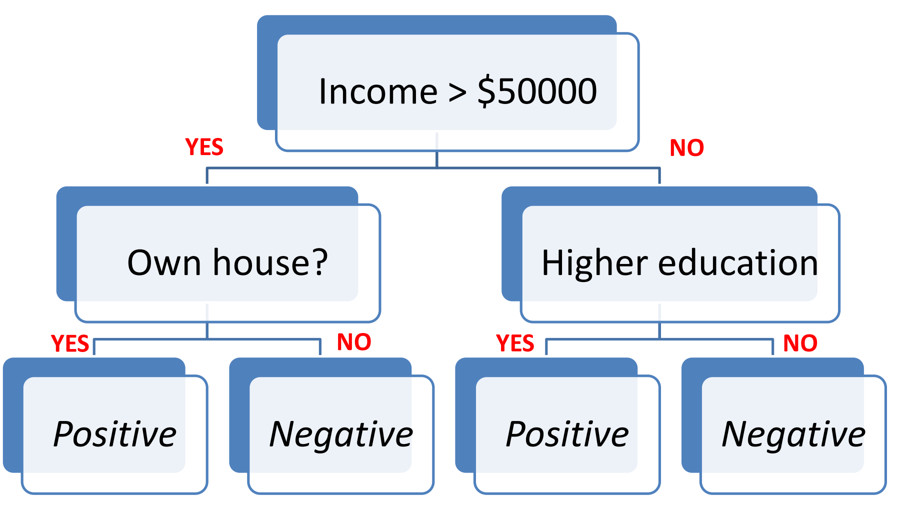
Zmienne niezależne → MODEL → *Zmienna zależna (ilościowa)*

Klasyfikacja wzorcowa

Klasyfikacja wzorcowa – problem polegający na przypisaniu obiektu do jednej z wcześniej znanych klas.

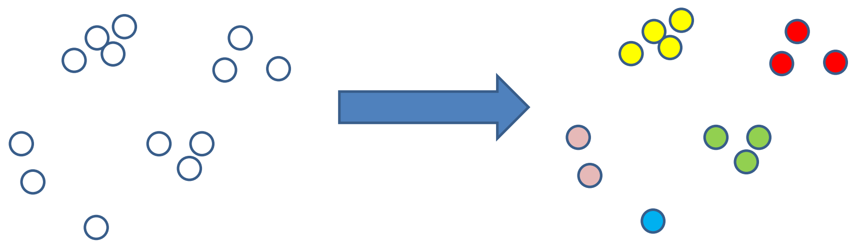


Analiza wniosków kredytowych jako przykład problemu z zakresu klasyfikacji wzorcowej.



Klasyfikacja bezwzorcowa / Analiza skupień

- Klasyfikacja bezwzorcowa (analiza skupień) – proces grupowania obiektów w klasy (podzbiory podobnych do siebie obiektów).



Segmentacja klientów jako przykład klasyfikacji bezwzorcowej

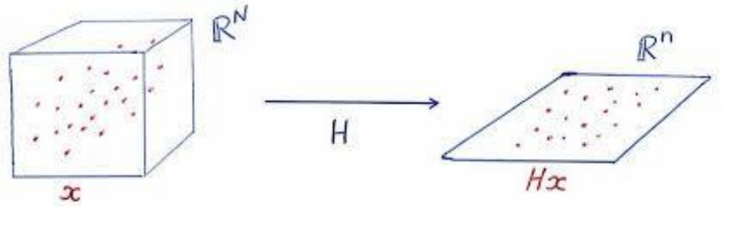
Segmentacja klientów – podział klientów na homogeniczne grupy ze względu na takie cechy jak: wiek, poziom dochodów, zainteresowania, preferencje zakupowe.

Analiza asocjacji

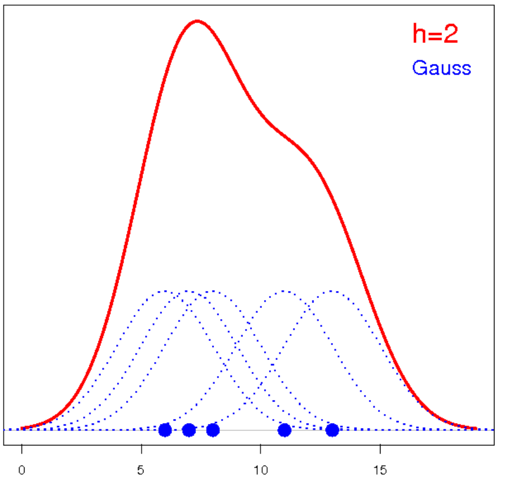
Analiza asocjacji – analiza pozwalająca zidentyfikować powiązane ze sobą zjawiska i obiekty.

Redukcja wymiaru przestrzeni danych

Redukcja wymiaru przestrzeni danych – transformacja obiektów z przestrzeni o dużej liczbie wymiarów do przestrzeni o mniejszej ich liczbie w sposób zapewniający w możliwie najlepszy sposób zachowanie struktury zbioru obiektów.



Estymacja funkcji gęstości

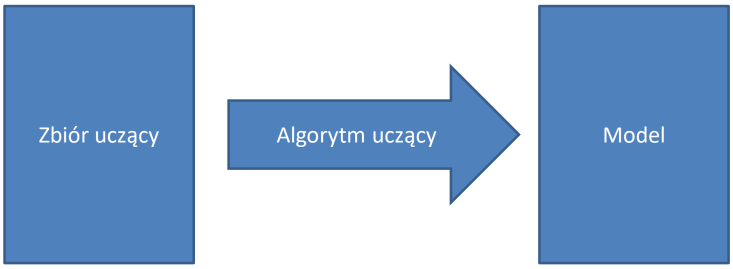


Dane

- Podział zbioru danych na:
  - Zbiór uczący
  - Zbiór walidacyjny,
  - zbiór testowy.

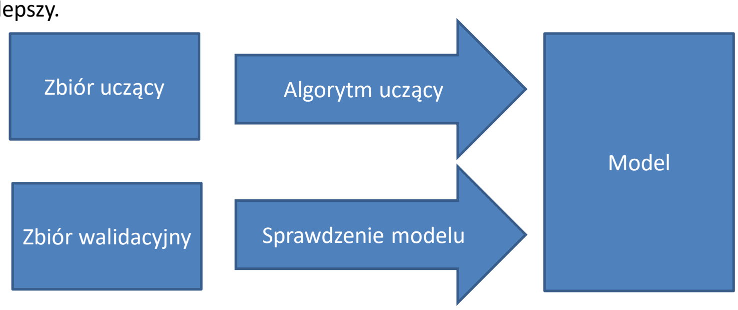
Zbiór uczący

Zbiór uczący – stanowi podstawę do modyfikacji modelu przez algorytm uczący



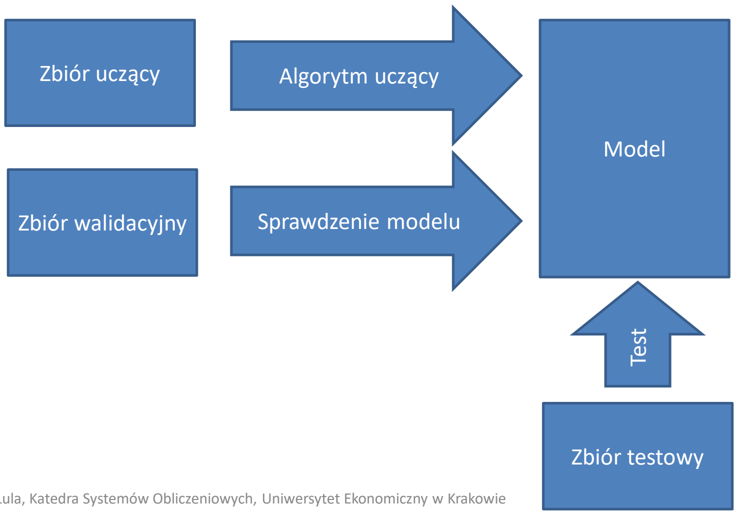
Zbiór walidacyjny

Zbiór walidacyjny – służy do oceny jakości modelu w trakcie jego uczenia. Jako wynik uczenia zwracany jest model dla którego błąd dla zbioru walidacyjnego jest najlepszy.



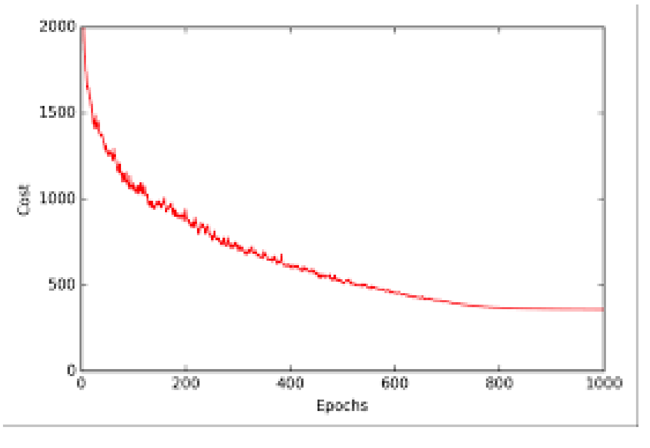
Zbiór testowy

Zbiór testowy – służy do ostatecznej (po zakończeniu uczenia) oceny uzyskanego modelu.



Kryterium uczenia (funkcja błędu)

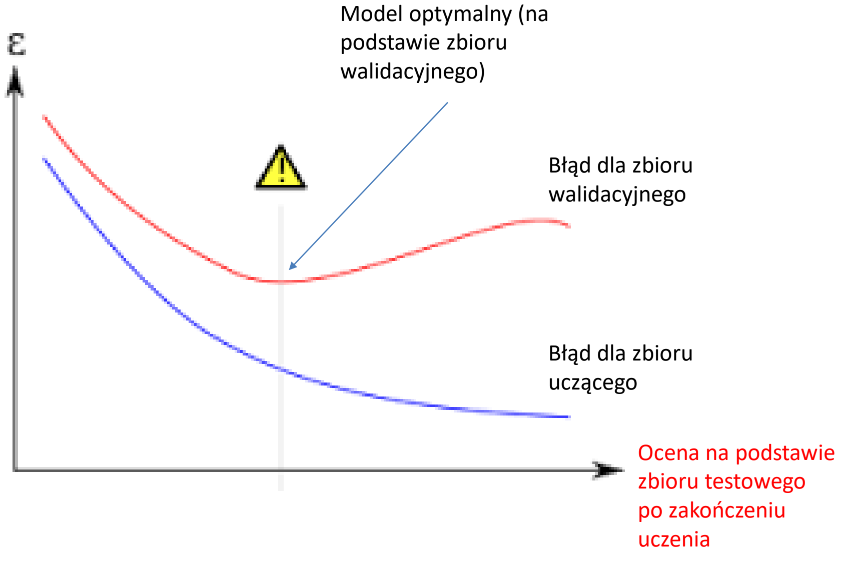
Kryterium uczenia – funkcja, której wartość powinna być minimalizowana przez algorytm uczący.



Zdolność generalizacji modelu i problem przeuczenia

- **Generalizacja** – zdolność modelu do prawidłowego działania dla danych różnych od danych uczących.
- **Przeuczenie** – zjawisko polegające na bardzo dobrym działania modelu dla danych uczących i jednoczesnym błędnym działaniem dla danych różnych od danych uczących.

Problem przeuczenia



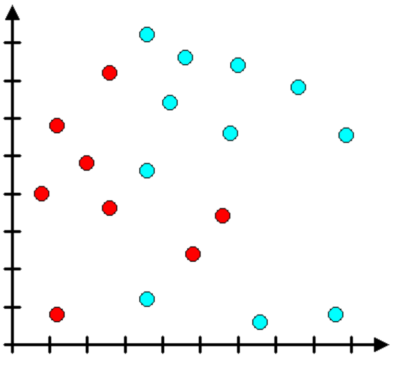
DRZEWA KLASYFIKACYJNE (DECYZYJNE) JAKO NARZĘDZIE ROZWIĄZYWANIA PROBLEMÓW Z ZAKRESU KLASYFIKACJI WZORCOWEJ

Drzewo klasyfikacyjne (decyzyjne)

- Drzewo klasyfikacyjne (decyzyjne)
  - reprezentuje proces podziału obiektów na klasy (reprezentowane przez etykiety),
  - decyzja o przypisaniu obiektu do klasy podejmowana jest w oparciu o wartości zmiennych objaśniających;

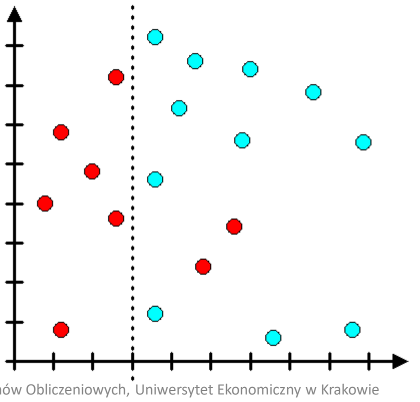
Drzewo klasyfikacyjne – cel budowy modelu

- Cel budowy drzewa: zidentyfikować reguły przypisujące obiekty do klas



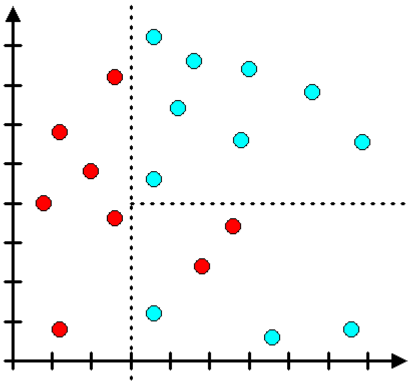
Drzewo klasyfikacyjne – zasada działania (1/3)

- Spośród zmiennych opisujących obiekty wybierz tę, która pozwala podzielić obiekty na dwie grupy w taki sposób, aby jednorodność powstałych grup była maksymalna



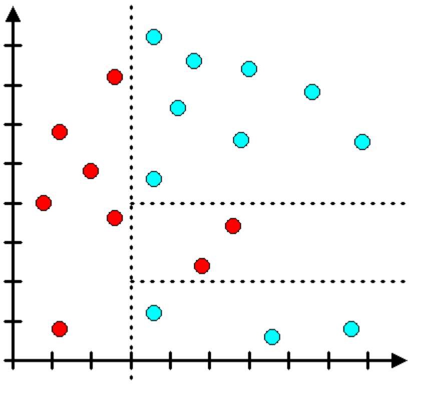
Drzewo klasyfikacyjne – zasada działania (2/3)

- Wybierz jedną z powstałych grup i dokonaj jej podziału w analogiczny sposób.

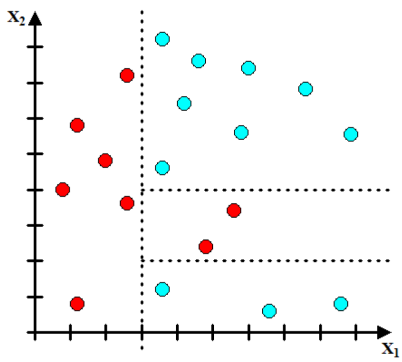


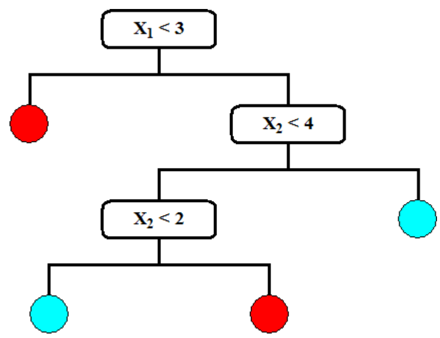
Drzewo klasyfikacyjne – zasada działania (3/3)

- Proces powtarzaj, aż do momentu uzyskania podziału na jednorodne grupy.

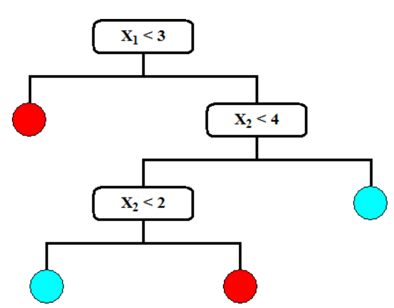


Zasady klasyfikacji w postaci drzewa





Zasady klasyfikacji w postaci reguł decyzyjnych



Jeżeli $X_1 < 3$ to:
  obiekt jest czerwony
w przeciwnym przypadku:
  jeżeli $X_2 < 4$ to:
    jeżeli $X_2 < 2$ to obiekt jest niebieski
    w przeciwnym przypadku obiekt jest czerwony
  w przeciwnym przypadku:
    obiekt jest niebieski

Tworzenie drzewa decyzyjnego

- Sprawdzenie, czy posiadany zbiór obiektów jest jednorodny. Jeśli tak, to algorytm kończy pracę. Jeśli nie, to realizowana jest dalsza część algorytmu.
- Rozważanie wszystkich możliwych podziałów zbioru obiektów na podzbiory (segmenty) i określenie, który z podziałów tworzy **najbardziej jednorodne segmenty**.
- Podział zbioru w wybrany sposób.
- Zastosowanie powyższego algorytmu do każdego z segmentów.

Ocena jednorodności segmentów

Zbiór losów:
- 1 los → samochód
- 3 losy → skuter
- 30 losów → aparat
- 999966 losów → puste

RAZEM: 1000000 losów

Ocena jednorodności segmentów

p(samochód) = 0,000001

p(skuter) = 0,000003

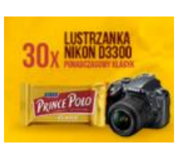
p(aparat) = 0,00003

p(nic) = 0,999966

Ilość informacji w wiadomości

p(samochód) = 0,000001

p(skuter) = 0,000003

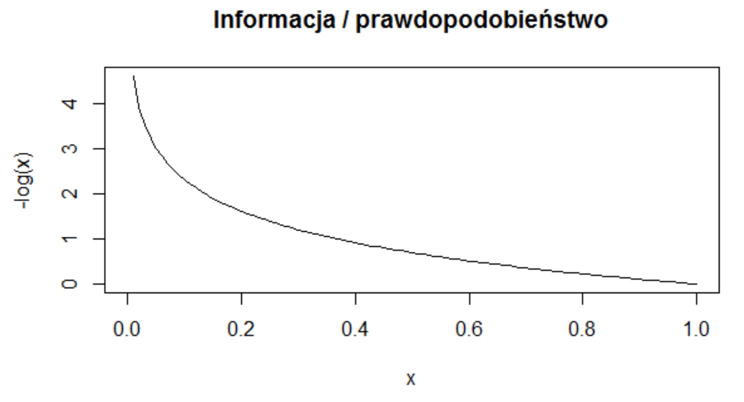


p(aparat) = 0,00003

p(nic) = 0,999966

```
> -log(0.000001)
[1] 13.81551
> -log(0.000003)
[1] 12.7169
> -log(0.00003)
[1] 10.41431
> -log(0.999966)
[1] 3.400058e-05
>
```

Ilość informacji w wiadomości




```
> curve(-log(x),from=0,to=1,main="Informacja / prawdopodobieństwo")
```


Średnia ilość informacji w wiadomościach (entropia)


```
> p1<-0.000001
> p2<-0.000003
> p3<-0.00003
> p4<-0.999966
> entropia<--p1*log(p1)-p2*log(p2)-p3*log(p3)-p4*log(p4)
> entropia
[1] 0.000398395
```


```
> p1<-0.01
> p2<-0.03
> p3<-0.3
> p4<-0.66
> entropia<--p1*log(p1)-p2*log(p2)-p3*log(p3)-p4*log(p4)
> entropia
[1] 0.7866805
>
```

```
> p1<-0.25
> p2<-0.25
> p3<-0.25
> p4<-0.25
> entropia<--p1*log(p1)-p2*log(p2)-p3*log(p3)-p4*log(p4)
> entropia
[1] 1.386294
>
```

równe prawdopodobieństwa
→ maksymalna entropia

Entropia jako miara jednorodności


```
> p1<-0.000001
> p2<-0.000003
> p3<-0.00003
> p4<-0.999966
> entropia<--p1*log(p1)-p2*log(p2)-p3*log(p3)-p4*log(p4)
> entropia
[1] 0.000398395
```

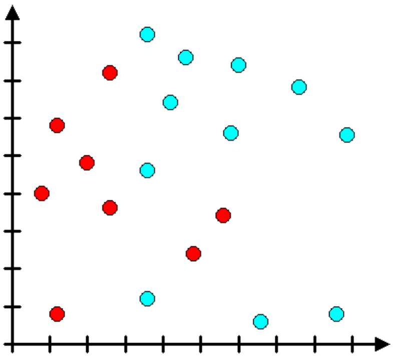

```
> p1<-0.01
> p2<-0.03
> p3<-0.3
> p4<-0.66
> entropia<--p1*log(p1)-p2*log(p2)-p3*log(p3)-p4*log(p4)
> entropia
[1] 0.7866805
>
```


```
> p1<-0.25
> p2<-0.25
> p3<-0.25
> p4<-0.25
> entropia<--p1*log(p1)-p2*log(p2)-p3*log(p3)-p4*log(p4)
> entropia
[1] 1.386294
>
```

równe prawdopodobieństwa
→ maksymalna entropia

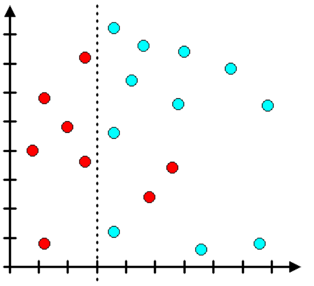
- **Entropia – miara niejednorodności (zróżnicowania)**
- **Im mniejsza entropia, tym większa jednorodność!**

Entropia



```
> p.czerw<-8/19
> p.ziel<-11/19
> entropia<--p.czerw*log(p.czerw)-p.ziel*log(p.ziel)
> entropia
[1] 0.6806295
```

Entropia

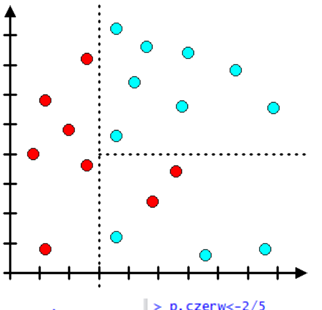


```
> p.czerw<-6/6
> entropia1<--p.czerw*log(p.czerw)
> entropia1
[1] 0
```

```
> p.czerw<-2/13
> p.ziel<-11/13
> entropia2<--p.czerw*log(p.czerw)-p.ziel*log(p.ziel)
> entropia2
[1] 0.429323
```

```
> entropia<-(6/19)*entropia1+(13/19)*entropia2
> entropia
[1] 0.2937473
>
```


Entropia



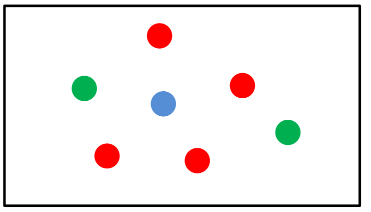
```
> p.ziel<-8/8
> entropia2<--p.ziel*log(p.ziel)
> entropia2
[1] 0
>
```

```
> p.czerw<-6/6
> entropia1<--p.czerw*log(p.czerw)
> entropia1
[1] 0
```

```
> p.czerw<-2/5
> p.ziel<-3/5
> entropia3<--p.czerw*log(p.czerw)-p.ziel*log(p.ziel)
> entropia3
[1] 0.6730117
```

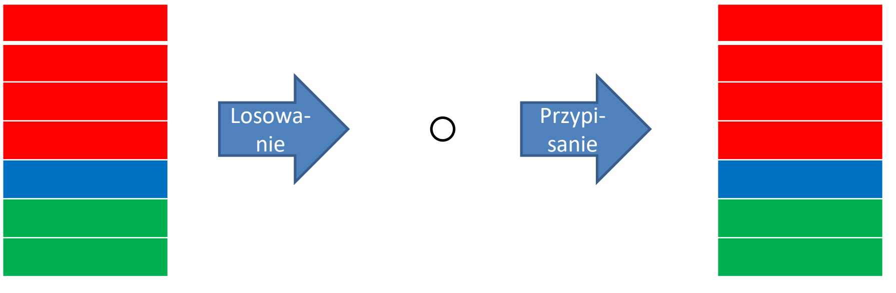
```
> entropia<-(6/19)*entropia1+(8/19)*entropia2+(5/19)*entropia3
> entropia
[1] 0.1771083
>
```

Indeks Giniego jako miara niejednorodności



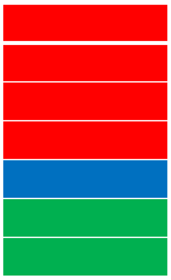
N = 7
N(z) = 2
N(c) = 4
N(n) = 1

Indeks Giniego jako miara niejednorodności



Indeks Giniego – prawdopodobieństwo przypisania wylosowanego obiektu do niewłaściwej klasy

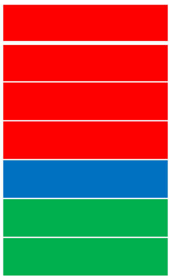
Indeks Giniego jako miara niejednorodności



Przypisanie

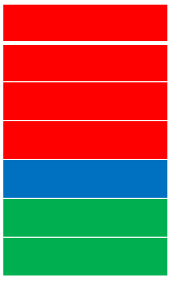
Losowanie

	p(c)	***p(n)***	***p(z)***
p(c)	*p(c)*p(c)*	*p(c)*p(n)*	*p(c)*p(z)*
p(n)	*p(n)*p(c)*	*p(n)*p(n)*	*p(n)*p(z)*
p(z)	*p(z)*p(c)*	*p(z)*p(n)*	*p(z)*p(z)*



Indeks Giniego – prawdopodobieństwo przypisania wylosowanego obiektu do niewłaściwej klasy

Indeks Giniego jako miara niejednorodności



Przypisanie

Losowanie

	p(c)	***p(n)***	***p(z)***
p(c)	*p(c)*p(c)*	*p(c)*p(n)*	*p(c)*p(z)*
p(n)	*p(n)*p(c)*	*p(n)*p(n)*	*p(n)*p(z)*
p(z)	*p(z)*p(c)*	*p(z)*p(n)*	*p(z)*p(z)*

prawidłowe przypisanie
błędne przypisanie

Indeks Giniego – prawdopodobieństwo przypisania wylosowanego obiektu do niewłaściwej klasy

$G = 1 – p(c)*p(c) - p(n)*p(n) - p(z)*p(z)$

Współczynnik Giniego

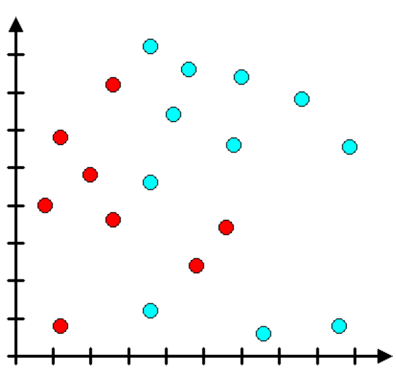

```
> p1<-0.000001
> p2<-0.000003
> p3<-0.00003
> p4<-0.999966
> gini<-1-p1*p1-p2*p2-p3*p3-p4*p4
> gini
[1] 6.799793e-05
```


```
> p1<-0.01
> p2<-0.03
> p3<-0.3
> p4<-0.66
> gini<-1-p1*p1-p2*p2-p3*p3-p4*p4
> gini
[1] 0.4734
```


```
> p1<-0.25
> p2<-0.25
> p3<-0.25
> p4<-0.25
> gini<-1-p1*p1-p2*p2-p3*p3-p4*p4
> gini
[1] 0.75
>
```

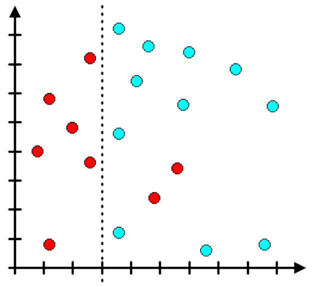
równe prawdopodobieństwa
→ maksymalny współczynnik Giniego

Współczynnik Giniego



```
> p.czerw<-8/19
> p.ziel<-11/19
> gini<-1-p.czerw^2-p.ziel^2
> gini
[1] 0.4875346
>
```

Współczynnik Giniego

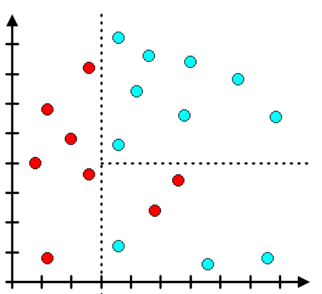


```
> p.czerw<-6/6
> gini1<-1-p.czerw^2
> gini1
[1] 0
```

```
> p.czerw<-2/13
> p.ziel<-11/13
> gini2<-1-p.czerw^2-p.ziel^2
> gini2
[1] 0.260355
```

```
> gini<-(6/19)*gini1+(13/19)*gini2
> gini
[1] 0.1781377
```


Współczynnik Giniego



```
> p.ziel<-8/8
> gini2<-1-p.ziel^2
> gini2
[1] 0
```

```
> p.czerw<-6/6
> gini1<-1-p.czerw^2
> gini1
[1] 0
```

```
> p.czerw<-2/5
> p.ziel<-3/5
> gini3<-1-p.czerw^2-p.ziel^2
> gini3
[1] 0.48
```

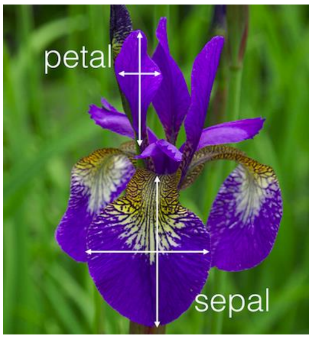
```
> gini<-(6/19)*gini1+(8/19)*gini2+(5/19)*gini3
> gini
[1] 0.1263158
```

Drzewa decyzyjne - pakiety

- Pakiety:
  - library(rpart)
  - library(rpart.plot)
- Dane:
  - data(iris)

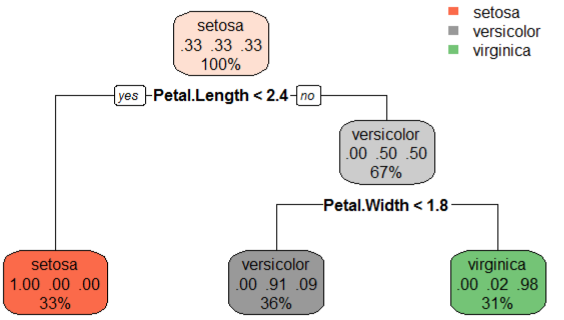
Zbiór iris

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
>
```



Tworzenie drzewa

```
> model1<-rpart(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,data=iris)
> rpart.plot(model1)
>
```

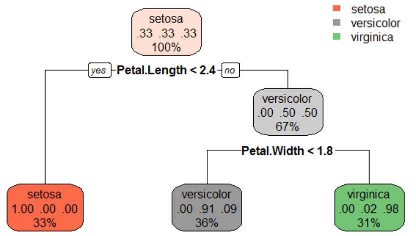


Reprezentacja drzewa

```
> print(model1)
n= 150

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 150 100 setosa (0.33333333 0.33333333 0.33333333)
  2) Petal.Length< 2.45 50   0 setosa (1.00000000 0.00000000 0.00000000) *
  3) Petal.Length>=2.45 100  50 versicolor (0.00000000 0.50000000 0.50000000)
    6) Petal.Width< 1.75 54   5 versicolor (0.00000000 0.90740741 0.09259259) *
    7) Petal.Width>=1.75 46   1 virginica (0.00000000 0.02173913 0.97826087) *
> |
```



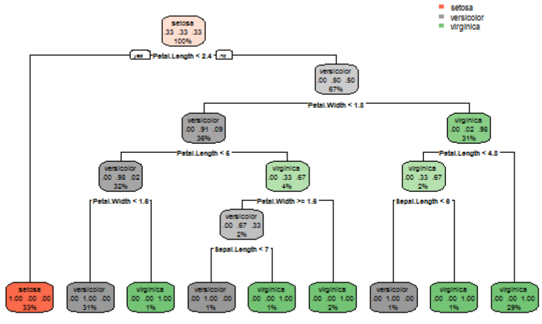
Wybór sposobu określenia niejednorodności

```
> model1<-rpart(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,
data=iris,parms=list(split="information"))

> model2<-rpart(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,
data=iris,parms=list(split="gini"))
```


Sterowanie wielkością drzewa

```
> model3<-rpart(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,
data=iris,parms=list(split="gini"),control=rpart.control(minsplit=1,cp=0.00001))
> rpart.plot(model3)
>
```



- ***minsplit*** – minimalna liczba elementów w węźle niezbędna do dokonania podziału
- ***cp*** (*complexity parameter*) – wymagana minimalna wartość polepszenia się miary jakości drzewa (o ile minimalnie musi się zmniejszyć miara niejednorodności)

Podział na zbiór uczący i testowy

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> nrow(iris)
[1] 150
>
```


Podział na zbiór uczący i testowy

```
> ind<-sample(150)
> ind
  [1]  62  41   8  73  66 100 149 101 135 130 117 140 133  29  68  65 115  63  30 144
 [21]  90 138 129  75  23  55  14  87  70  52  94  47  12 112  96  36  59  18  54  78
 [41] 102  50 103  19  35  76  31  16  17  38 137  69  82  97  80 104 122  27  72  57
 [61] 128  20 126  98  91  77  34  45 146  42  51  10  48 119  86 141  15  40  22  71
 [81] 134  95 118 116   7  56  25 107  61  67  33 124 109 114  49 105 143 111  92  93
[101] 147   4 125  89  43 139 127  99  26  37  44 121 113 136 132  60  83 106 131   9
[121]  88 150  58   1   2 123  85  64  39  74 110  46 142  21 120  79 148   5  32  13
[141]  53 108  81  24   3  11 145  28   6  84
>
```

Podział na zbiór uczący i testowy

```
> ind.ucz<-ind[1:120]
> ind.test<-ind[121:150]
> iris.ucz<-iris[ind.ucz,]
> iris.test<-iris[ind.test,]
> nrow(iris.ucz)
[1] 120
> nrow(iris.test)
[1] 30
>
```

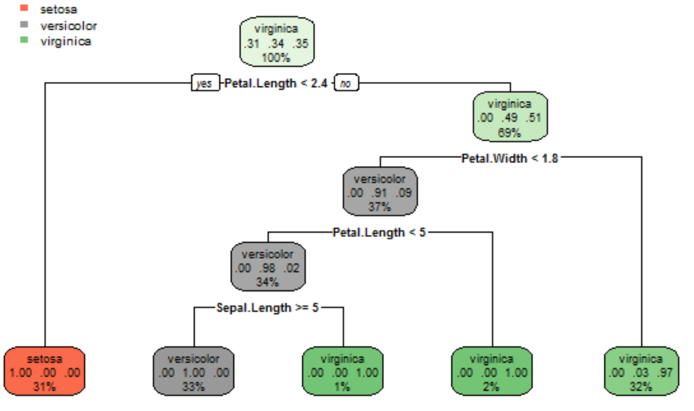

Budowa modelu w oparciu o zbiór uczacy

```
> library(rpart)
> library(rpart.plot)
> model4<-rpart(Species~Sepal.Length+Sepal.Width+Petal.Length+Petal.Width,
data=iris.ucz,parms=list(split="gini"),control=rpart.control(minsplit=3))
> model4
n= 120

node), split, n, loss, yval, (yprob)
      * denotes terminal node

 1) root 120 78 virginica (0.30833333 0.34166667 0.35000000)
   2) Petal.Length< 2.45 37  0 setosa (1.00000000 0.00000000 0.00000000) *
   3) Petal.Length>=2.45 83 41 virginica (0.00000000 0.49397590 0.50602410)
     6) Petal.Width< 1.75 44  4 versicolor (0.00000000 0.90909091 0.09090909)
      12) Petal.Length< 5.05 41  1 versicolor (0.00000000 0.97560976 0.02439024)
        24) Sepal.Length>=4.95 40  0 versicolor (0.00000000 1.00000000 0.00000000) *
        25) Sepal.Length< 4.95 1  0 virginica (0.00000000 0.00000000 1.00000000) *
      13) Petal.Length>=5.05 3  0 virginica (0.00000000 0.00000000 1.00000000) *
     7) Petal.Width>=1.75 39  1 virginica (0.00000000 0.02564103 0.97435897) *
>
```

Budowa modelu w oparciu o zbiór uczacy



Uruchomienie modelu dla zbioru uczącego

```
> res.ucz<-predict(model4,newdata=iris.ucz,type="class")
> tb.ucz<-table(iris.ucz$Species,res.ucz)
> tb.ucz
            res.ucz
             setosa versicolor virginica
  setosa         37          0         0
  versicolor      0         40         1
  virginica       0          0        42
> sum(diag(tb.ucz))/sum(tb.ucz)
[1] 0.9916667
>
```

Uruchomienie modelu dla zbioru testowego

```
> res.test<-predict(model4,newdata=iris.test,type="class")
> tb.test<-table(iris.test$Species,res.test)
> tb.test
            res.test
             setosa versicolor virginica
  setosa         13          0         0
  versicolor      0          7         2
  virginica       0          1         7
> sum(diag(tb.test))/sum(tb.test)
[1] 0.9
>
```


Wyznaczanie prawdopodobieństw przynależności do klas

```
> res.test.prob<-predict(model4,newdata=iris.test,type="prob")
> head(res.test.prob)
    setosa versicolor virginica
88       0 1.00000000  0.000000
150      0 0.02564103  0.974359
58       0 0.00000000  1.000000
1        1 0.00000000  0.000000
2        1 0.00000000  0.000000
123      0 0.02564103  0.974359
>
```

METODY TAKSONOMICZNE

Analiza skupień

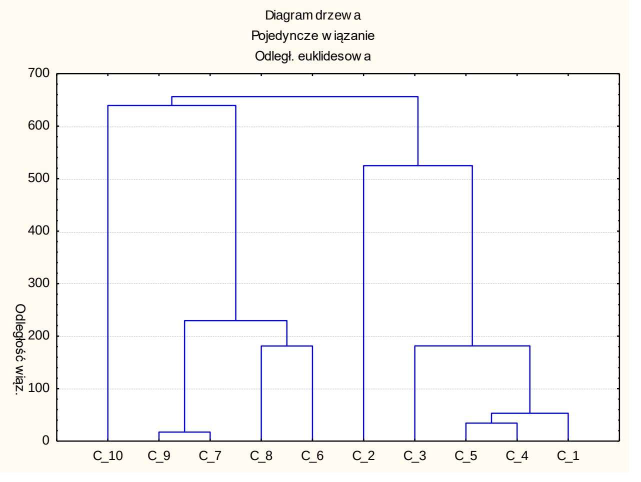
- Analiza skupień – analiza zbioru obiektów w celu określenia jego struktury (identyfikacji klas obiektów podobnych).
- W zależności od podejścia możliwe jest uzyskanie klas:
  - hierarchicznych (klasy dzielą się na podklasy),
  - wykluczających się (każdy obiekt należy do jednej klasy),
  - niewykluczających się (obiekt może należeć do kilku klas),
  - opisanych w kategoriach rozkładów prawdopodobieństwa (tworzone są modele klas).

METODY HIERARCHICZNE

Metody hierarchiczne i ich podział

- Metody hierarchiczne – metody pozwalające na odtworzenie hierarchii klas obiektów. Pokazują wszystkie stany pośrednie pomiędzy przypadkiem, w którym wszystkie obiekty tworzą jedną klasę i przypadkiem, w którym każdy z obiektów jest samodzielną klasą.
- Rodzaje metod hierarchicznych:
  – metody aglomeracyjne – w pierwszym kroku każdy z obiektów tworzy oddzielną klasę. Na każdym kolejnym dwie najbardziej podobne klasy są ze sobą łączone. Na ostatnim etapie wszystkie obiekty tworzą jedną klasę.
  – Metody podziałowe – w pierwszym kroku wszystkie obiekty tworzą jedną klasę. W trakcie każdego kolejnego kroku jedna klasa jest dzielona na dwie. W ostatnim kroku obiekty tworzą jednoelementowe klasy.

Dendrogram



Dendrogram jako wynik działania metod hierarchicznych.

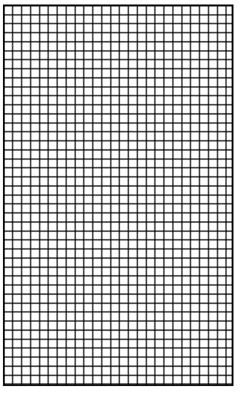
Etapy działania metod hierarchicznych

- określenie celu badań
- przygotowanie zbioru danych
- wstępne przetworzenie danych (np. standaryzacja)
- obliczenie macierzy odległości
- wykonanie obliczeń
- prezentacja wyników (drzewko połączeń)
- wybór podziału optymalnego

Struktura zbioru danych

zmienne

obiekty



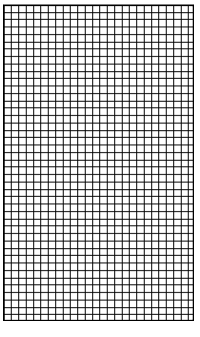
Cel badań:
- klasyfikacja obiektów,

lub
- klasyfikacja zmiennych.

Wstępne przetworzenie danych

zmienne

obiekty



$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{S_j}$$

$$z_{ij} = \frac{x_{ij}}{\max_i (x_{ij})}$$

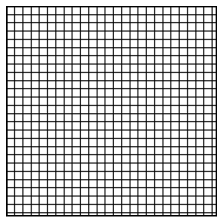
$$z_{ij} = \frac{x_{ij}}{\min_i (x_{ij})}$$

$$z_{ij} = \frac{x_{ij}}{\bar{x}_j}$$

Wyznaczenie macierzy odległości

obiekty

obiekty



Odległość miejska:

$$d_{ik} = \sum_{j=1}^{m} \left| z_{ij} - z_{kj} \right|$$

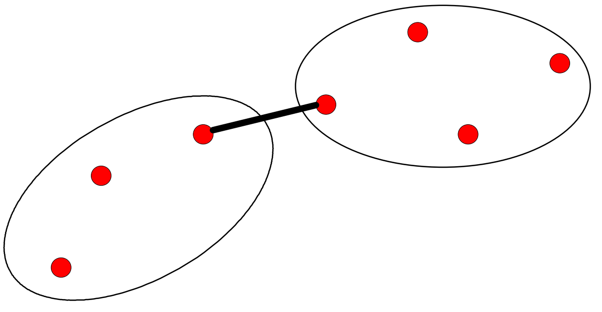
Odległość Euklidesa:

$$d_{ik} = \sqrt{\sum_{j=1}^{m} \left(z_{ij} - z_{kj} \right)^2}$$

Metody aglomeracyjne

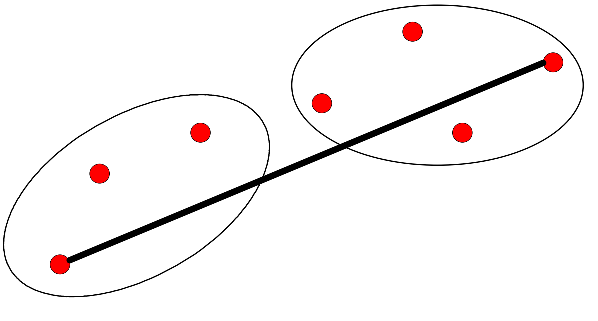
1. każdy obiekt tworzy oddzielne skupienie
2. **następuje łączenie dwóch <u>najbliższych</u> elementów**; połączone elementy tworzą grupę;
3. modyfikacja macierzy odległości – połączone elementy reprezentuje jeden wiersz (i jedna kolumna); aktualizacja elementów macierzy odległości;
4. jeżeli obiekty nie tworzą jednej grupy to przejście do kroku 2.

Metoda najbliższego sąsiedztwa



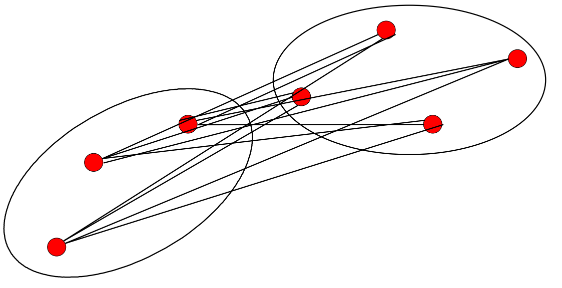
Single linkage method

Metoda najdalszego sąsiedztwa



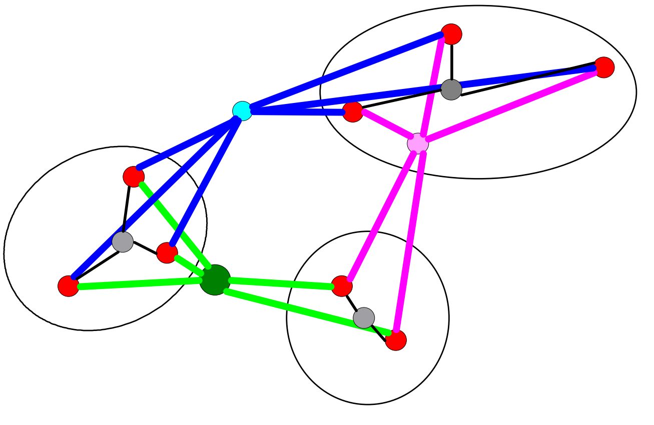
Complete linkage method

Metoda uśrednionego sąsiedztwa



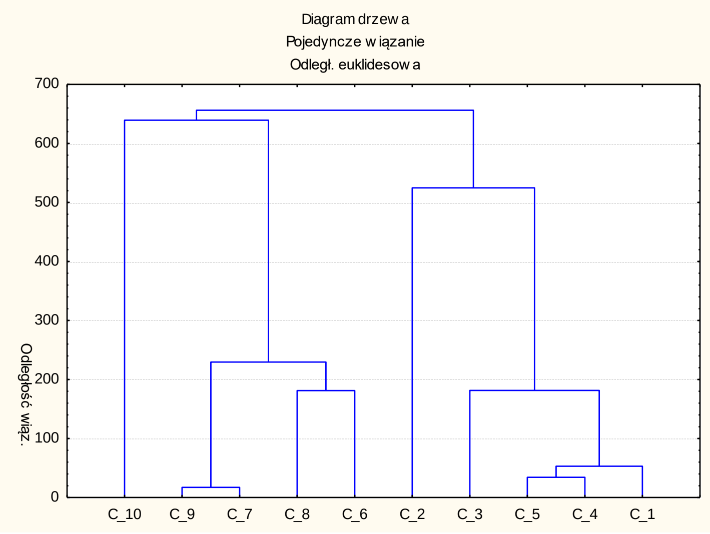
Average linkage method

Metoda Warda

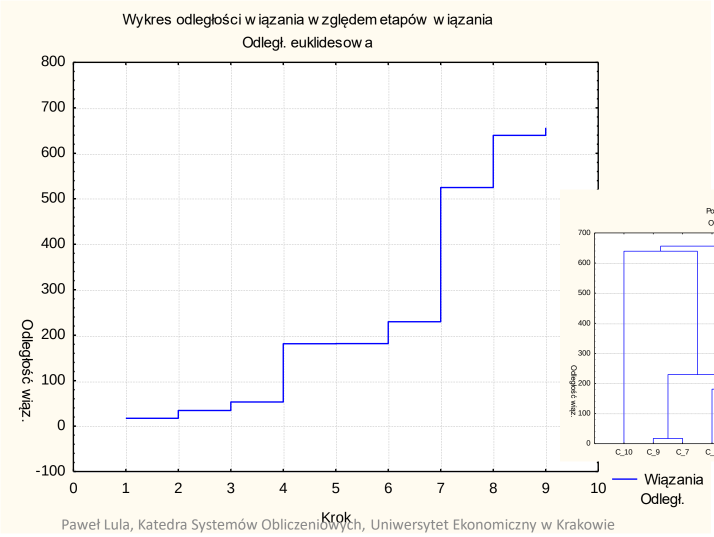


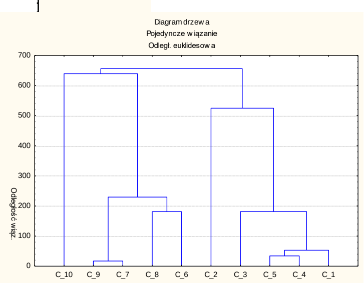
Łączenie dokonywane jest w taki sposób, aby w najmniejszym stopniu zwiększyć wariancję wewnątrzgrupową

Prezentacja wyników



Wybór podziału optymalnego





Przygotowanie danych

```
> names(iris)
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
> names(iris)=="Species"
[1] FALSE FALSE FALSE FALSE  TRUE
> !(names(iris)=="Species")
[1]  TRUE  TRUE  TRUE  TRUE FALSE
> iris.new<-iris[,!(names(iris)=="Species")]
> head(iris.new)
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1          5.1         3.5          1.4         0.2
2          4.9         3.0          1.4         0.2
3          4.7         3.2          1.3         0.2
4          4.6         3.1          1.5         0.2
5          5.0         3.6          1.4         0.2
6          5.4         3.9          1.7         0.4
>
```


Przygotowanie danych

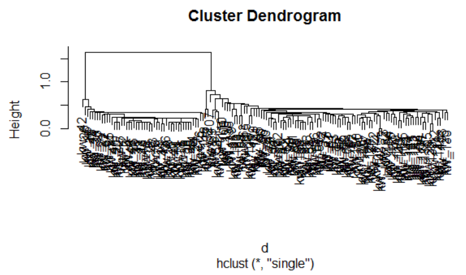
```
> rownames(iris.new)<-paste("kw_",rownames(iris.new),sep="")
> head(iris.new)
     Sepal.Length Sepal.Width Petal.Length Petal.Width
kw_1          5.1         3.5          1.4         0.2
kw_2          4.9         3.0          1.4         0.2
kw_3          4.7         3.2          1.3         0.2
kw_4          4.6         3.1          1.5         0.2
kw_5          5.0         3.6          1.4         0.2
kw_6          5.4         3.9          1.7         0.4
>
```

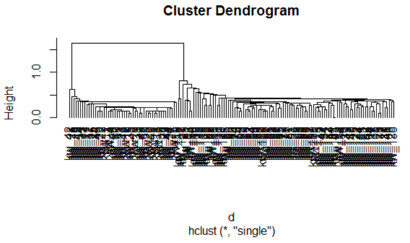
Wyznaczenie macierzy odległości

```
> rownames(iris.new)<-paste("kw_",rownames(iris.new),sep="")
> d<-dist(iris.new)
>
> dim(as.matrix(d))
[1] 150 150
> d<-dist(iris.new,method="euclidean")
> d<-dist(iris.new,method="manhattan")
> d<-dist(iris.new,method="canberra")
> d<-dist(iris.new,method="binary")
>
```

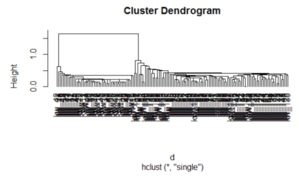

Realizacja podziału

```
> d<-dist(iris.new,method="euclidean")
> clust.res<-hclust(d,method="single")
> plot(clust.res)
>  plot(clust.res,hang = -1)
```

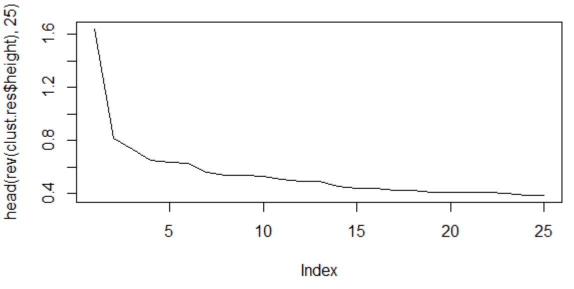




Wykres osuwiska




```
> plot(head(rev(clust.res$height),25),type="l")
>
```



Przecięcie drzewa

```
> div<-cutree(clust.res,h=1.2)
> head(div)
kw_1 kw_2 kw_3 kw_4 kw_5 kw_6
   1    1    1    1    1    1
> length(div)
[1] 150
> s<-c(rep(1,50),rep(2,50),rep(3,50))
> library(mclust)
> adjustedRandIndex(s,s)
[1] 1
> adjustedRandIndex(s,div)
[1] 0.5681159
> div<-cutree(clust.res,k=3)
> adjustedRandIndex(s,div)
[1] 0.563751
>
```

Indeks Randa:

- rozważamy dwie klasyfikacje i wszystkie pary obiektów,
- cztery możliwe przypadki:
    - a) K1: o1, o2 → w tej samej klasie, K2: o1, o2 → w tej samej klasie
    - b) K1: o1, o2 → w różnych klasach, K2: o1, o2 → w różnych klasach
    - c) K1: o1, o2 → w tej samej klasie, K2: o1, o2 → w różnych klasach
    - d) K1: o1, o2 → w różnych klasach, K2: o1, o2 → w tej samej klasie
- R = (#a+#b)/(#a+#b+#c+#d)

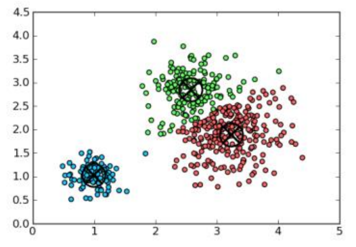
Zastosowanie metody Warda

```
> clust.res<-hclust(d,method="ward.D2")
> div<-cutree(clust.res,k=3)
> adjustedRandIndex(s,div)
[1] 0.7311986
> table(s,div)
   div
s    1  2  3
  1 50  0  0
  2  0 49  1
  3  0 15 35
>
```

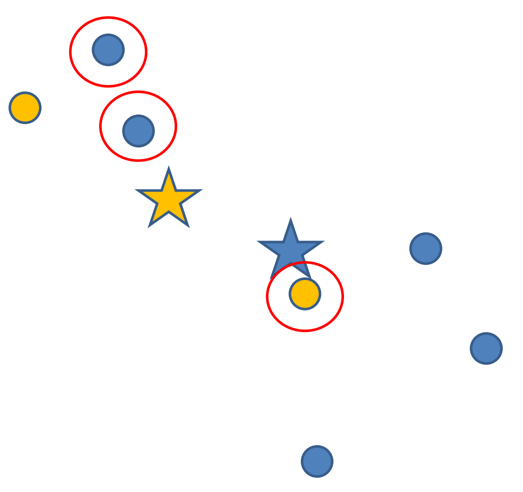

KLASYFIKACJA O KLASACH WYKLUCZAJĄCYCH SIĘ

Metoda k-średnich (k-means)

- Cel: podzielić obiekty na zadaną liczbę klas w taki sposób, aby suma odległości odległości poszczególnych obiektów od środków klas do których należą była minimalna.

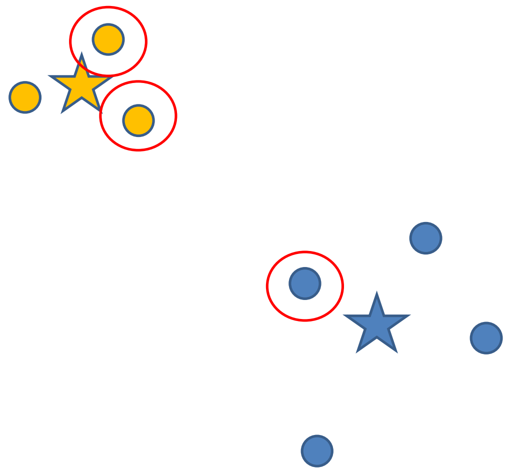


Metoda k-średnich



- Ustalana jest liczba klas
- Obiekty są losowo przypisywane do klas
- Dla każdej klasy wyznaczony jest środek
- Identyfikowane są obiekty błędnie zaklasyfikowane (położone bliżej środka innej klasy niż środka swojej własnej klasy)

Metoda k-średnich



- Obiekty są przesuwane do właściwych klas
- Aktualizowane są środki klas
- Weryfikowana jest poprawność klasyfikacji czy każdy obiekt jest we właściwej klasie). Jeśli nie to proces modyfikacji klas jest powtarzany.

Realizacja metody k-średnich

```
> km.res<-kmeans(iris.new,3)
> head(km.res$cluster)
kw_1 kw_2 kw_3 kw_4 kw_5 kw_6
   3    3    3    3    3    3
> adjustedRandIndex(s,km.res$cluster)
[1] 0.7302383
> table(s,km.res$cluster)

s    1  2  3
  1  0  0 50
  2  2 48  0
  3 36 14  0
>
```

DZIĘKUJĘ ZA UWAGĘ