

# Metody bioinformatyki (MBI)

Wykład 10 - analiza danych wielowymiarowych; grupowanie;  
algorytm PCA; drzewa filogenetyczne.

Robert Nowak

2025Z

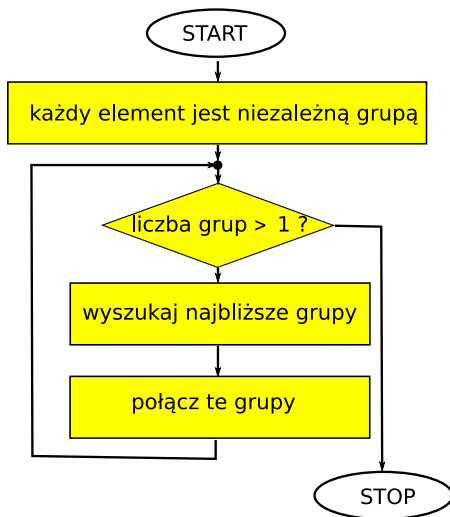
# *Grupowanie*

## grupowanie (ang. clustering)

Grupowanie to tworzenie klastrów (grup) obiektów o podobnych cechach.

- ▶ wymaga definicji, co to znaczy, że obiekty są „podobne”.
- ▶ obiekt jest „podobny” do dowolnego obiektu z tej samej grupy, i jest „niepodobny” do każdego obiektu z innych grup.
- ▶ istnieją miary jakości grupowania oceniające jednocześnie
  - ▶ czy klastry są zwarte („podobne” obiekty w grupach)
  - ▶ czy klastry są rozłączne („niepodobne” obiekty pomiędzy grupami)

# algorytm grupowania hierarchicznego



- ▶ wymaga definicji odległości pomiędzy elementami
- ▶ wymaga definicji odległości pomiędzy grupami

# definicja odległości

$x, y$  oznaczają punkty ze zbiorów danych, każdy punkt ma  $m$  cech (atrybutów)  $x = \langle x_1, x_2, \dots, x_m \rangle$

- ▶ odległość Euklidesowa

$$d_{xy} = \sqrt{\sum_{i=1}^m (x_i - y_i)^2}$$

- ▶ odległość Manhattan

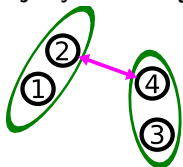
$$d_{xy} = \sum_{i=1}^m |x_i - y_i|$$

- ▶ korelacja

$$d_{xy} = \sum_{i=1}^m x_i y_i$$

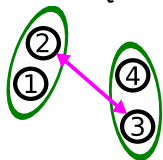
## odległości grup

pojedyncze wiązanie (najmniejsza odległość)



$$d_s(X, Y) = \min_{x \in X, y \in Y} d_{xy}$$

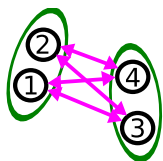
pełne wiązanie (największa odległość)



$$d_f(X, Y) = \max_{x \in X, y \in Y} d_{xy}$$

## odległości grup (2)

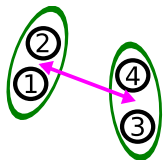
średnia odległość



$$d_a(X, Y) = \frac{1}{|X||Y|} \sum_{x \in X, y \in Y} d_{xy}$$

gdzie  $|X|$  to ilość elementów w  $X$ 

odległość pomiędzy środkami



$$d_c(X, Y) = d_{x'y'}$$

gdzie  $x'$  to średnia elementów  $x \in X$

# algorytm grupowania hierarchicznego, przykład

Tabela odległości pomiędzy obiektami A, B, C, D:

	A	B	C	D
A	0	2	6	11
B		0	4	9
C			0	5
D				0

Dla dwóch grup:

- ▶ jeżeli odległość między grupami to pojedyncze wiązanie (minimalna odległość pomiędzy elementami)?  **$\{A,B,C\}\{D\}$**
- ▶ jeżeli odległość między grupami to pełne wiązanie (maksymalna odległość pomiędzy elementami)?  **$\{A,B\}\{C,D\}$**



# algorytm grupowania hierarchicznego - złożoność

- ▶ liczba przykładów:  $n$ , liczba atrybutów:  $m$
- ▶ liczba kroków algorytmu  $n - 1$
- ▶ każda iteracja:
  - ▶  $n(n - 1)/2$  razy oblicza odległość
  - ▶ koszt obliczenia odległości  $O(m)$
  - ▶ koszt iteracji  $O(n^2m)$

$$O(n^3m)$$

# algorytm K-średnich (grupowanie niehierarchiczne)

zakłada podział na K grup

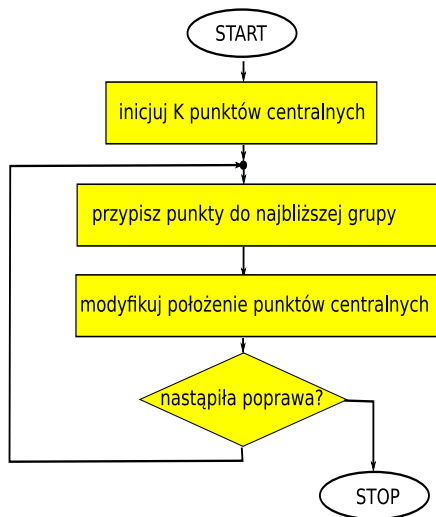
► odległość

$x = \langle x_1, x_2, \dots, x_m \rangle$  od  $c$

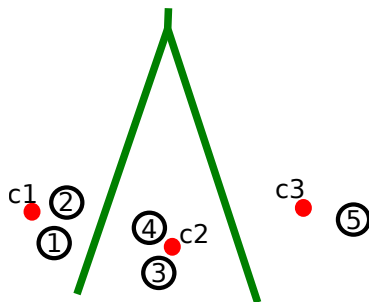
$$d_{xc} = \sum_{i=1}^m (x_i - c_i)^2$$

► funkcja błędu (którą minimalizujemy)

$$E = \sum_c \sum_{x \in c} d_{xc}$$



## algorytm K-średnich (2)

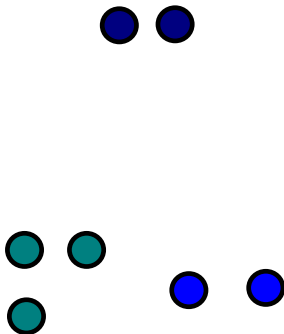


Algorytm optymalizacyjny:

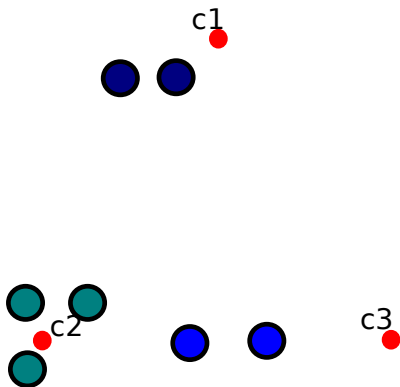
- ▶ inicjacja: losowa
- ▶ funkcja celu:

$$\arg \min_{C_1, C_2, \dots, C_k} \sum_{c=1}^k \sum_{x_{ij} \in C_c} \sum_{j=1}^m (x_{ij} - c_{cj})^2$$

## algorytm K-średnich (przykład)

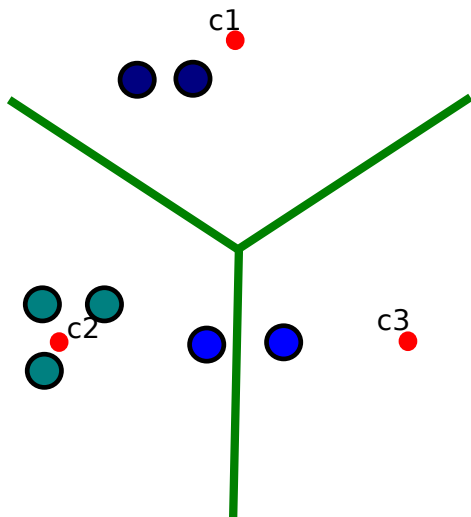


## algorytm K-średnich (przykład)



- ▶ poszukiwane 3 grupy
- ▶ inicjacja punktów centralnych

## algorytm K-średnich (przykład)

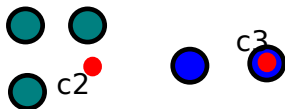


obliczenie przykładów  
należących do danej  
grupy

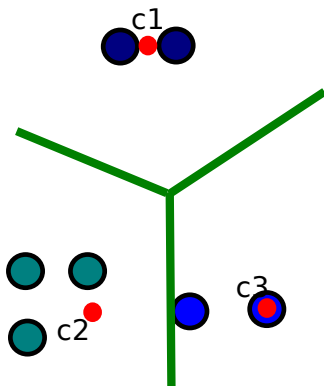
## algorytm K-średnich (przykład)



aktualizacja położenia  
punktów centralnych



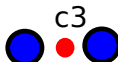
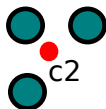
## algorytm K-średnich (przykład)



obliczenie przykładów  
należących do danej  
grupy



## algorytm K-średnich (przykład)



aktualizacja punktów  
centralnych

## algorytm K-średnich - złożoność

- ▶ liczba przykładów:  $n$ , liczba atrybutów  $m$ , liczba kroków algorytmu  $p \approx n$
- ▶ każda iteracja:
  - ▶ koszt obliczenia odległości  $O(m)$
  - ▶ koszt znalezienia grupy: dla  $n$  punktów  $k$  razy oblicza odległość od punktu centralnego, więc  $O(knm)$
  - ▶ koszt obliczenia nowego punktu środkowego, dla  $n$  punktów oblicza średnią  $O(nm)$
  - ▶ iteracja  $O(knm + nm) = O(knm)$

$$O(kn^2m)$$

algorytm znacznie wydajniejszy niż grupowanie hierarchiczne  $O(n^3m)$ , ponieważ  $k \ll n$ , ale problemy z właściwą inicjacją

# Miara jakości grupowania

Uwzg. odchylenie wewnątrzklastrowe (klastry zwarte)

$$wc = \sum_i^n \sum_{x \in C_i} d(x, r_i), \text{ } r_i \text{ to } \text{środek klastra } C_i$$

oraz odchylenie międzyklastrowe (klastry rozłączne)

$$bc = \sum_{i,j \neq i}^n d(r_i, r_j)$$

Przykłady miar:



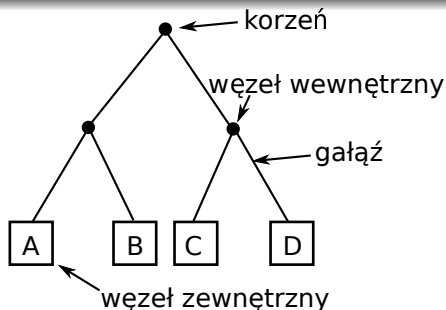
$$\frac{bc}{wc},$$

- ▶ indeks Daviesa-Bouldina,
- ▶ indeks Dunna.

# *Drzewa filogenetyczne*

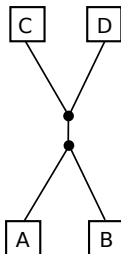
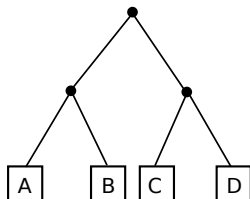
# Drzewa filogenetyczne

Drzewo filogenetyczne przedstawia zależności pomiędzy wieloma sekwencjami.



- ▶ drzewa ukorzenione i nieukorzenione
- ▶ drzewa binarne i inne
- ▶ krawędzie obrazują lub nie obrazują odległość

# Drzewa ukorzenione i nieukorzenione



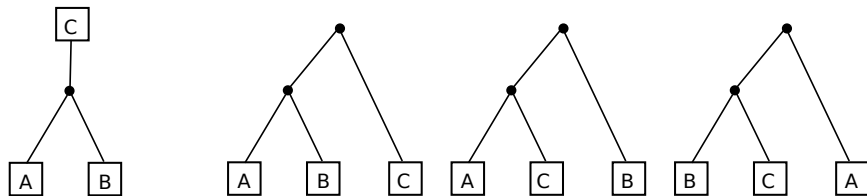
- liczba drzew ukorzenionych, gdzie  $n$  to liczba taksonów (węzłów zewnętrznych)

$$N_R = \frac{(2n - 3)!}{2^{n-2}(n - 2)!}$$

- liczba drzew nieukorzenionych

$$N_U = \frac{(2n - 5)!}{2^{n-3}(n - 3)!}$$

# Drzewa ukorzenione i nieukorzenione (2)



$n$	$N_U$	$N_R$
3	1	3
4	3	15
6	105	945
10	$2 * 10^6$	$34 * 10^6$

$n$  - liczba taksonów,  $N_U$  liczba drzew nieukorzenionych,  $N_R$  liczba drzew ukorzenionych

# Algorytmy do konstrukcji drzew

- ▶ metody bazujące na odległościach sekwencji
  - ▶ metoda średnich połączeń (UPGMA - unweighted pair group method with arithmetic mean)
  - ▶ metoda przyłączania sąsiadów (NJ - neighbour joining)
- ▶ metody bazujące na analizie symboli
  - ▶ metoda parsymonii (Parsimony)
  - ▶ metoda największej wiarygodności (Maximum likelihood)



# UPGMA - metoda grupowania parami ze średnią arytmetyczną

- ▶ bazuje na odległościach pomiędzy sekwencjami
- ▶ grupuje najbliższe taksony
- ▶ zakłada jednakową odległość taksonów od korzenia
- ▶ bardzo wydajna

Algorytm UPGMA identyczny jak algorytm grupowania hierarchicznego.



# UPGMA - przykład (1)

Przykładowa macierz odległości pomiędzy taksonami A, B, C, D:

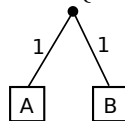
	B	C	D
A	2	6	11
B		4	9
C			4

Najbliższe taksony A i B

Zredukowana macierz odległości:

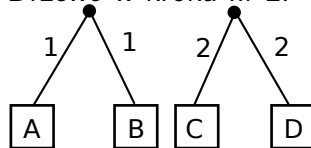
	C	D
{A, B}	5	10
C		4

Drzewo po utworzeniu taksonu {A, B}



Węzeł wewnętrzny umieszczamy w połowie odległości.

Drzewo w kroku nr 2.



## UPGMA - przykład (2)

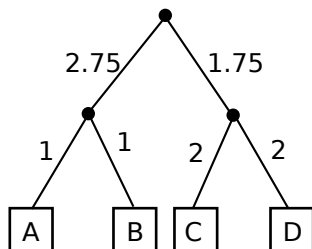


Tabela odległości odtworzona z drzewa:

	B	C	D
A	2	7.5	7.5
B		7.5	7.5
C			4

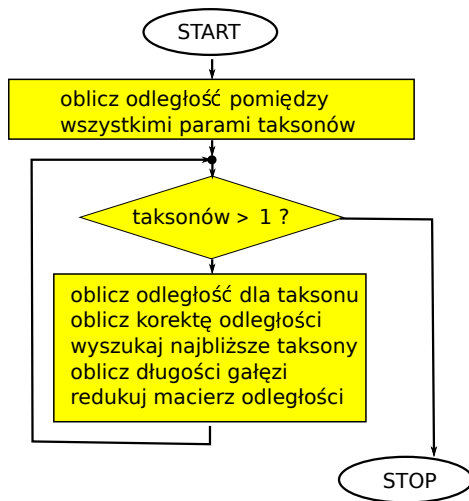
tabela pierwotna:

	B	C	D
A	2	6	11
B		4	9
C			4

# NJ - metoda przyłączania sąsiadów

- ▶ bazuje na odległościach pomiędzy sekwencjami
- ▶ grupuje najbliższe taksony
- ▶ nie zakłada jednakowej odległości taksonów od korzenia

Algorytm NJ podobny do UPGMA



## algorytm NJ - obliczenia

- ▶ korekta odległości pomiędzy taksonami

$$\begin{aligned}d_{ij} & \text{ odległość pomiędzy taksonami} \\ r_i & = \sum_{i \neq j} d_{ij} \text{ odległość dla taksonu} \\ d'_{ij} & = d_{ij} - \frac{r_i + r_j}{2} \text{ korekta odległości}\end{aligned}$$

- ▶ dodanie nowego węzła

$$\begin{aligned}r'_i & = \frac{r_i}{n-2}, \text{ gdzie } n \text{ to liczba taksonów} \\ d_{iu} & = \frac{d_{ij} + r'_i - r'_j}{2} \text{ nowy węzeł } u \\ d_{ju} & = \frac{d_{ij} + r'_i - r'_j}{2} \text{ nowy węzeł } u\end{aligned}$$

- ▶ redukcja macierzy (dodano takson  $u$  pomiędzy  $i$  i  $j$ )

$$d_{ku} = \frac{d_{ik} - d_{iu} + d_{jk} - d_{ju}}{2} \text{ dla każdego } k \neq i \wedge k \neq j$$

## algorytm NJ - przykład (1)

Przykładowa macierz odległości pomiędzy taksonami A, B, C, D:

	B	C	D
A	2	5	4
B		3	2
C			3

Macierz po korekcie:

	B	C	D
A	-7	-6	-6
B		-6	-6
C			-7

$$r_A = 2 + 5 + 4 = 11$$

$$r_B = 7$$

$$r_C = 11$$

$$r_D = 9$$

$$d'_{AB} = d_{AB} - \frac{r_A + r_B}{2} = -7$$

Minimalna odległość to  $d_{AB}$  lub  $d_{CD}$ , wybieramy dowolną z tych par, tutaj  $d_{AB}$

## Algorytm NJ - przykład (2)

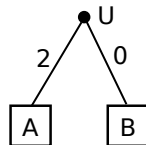
Nowy węzeł  $U$  pomiędzy  $A$  i  $B$ 

$$r'_A = \frac{r_A}{4-2} = 5.5$$

$$r'_B = 3.5$$

$$d_{AU} = \frac{d_{AB} + r'_A - r'_B}{2} = 2$$

$$d_{BU} = \frac{d_{AB} + r'_B - r'_A}{2} = 0$$



Redukcja:

$$d_{CU} = \frac{d_{AC} - d_{AU} + d_{BC} - d_{BU}}{2} = \frac{5 - 2 + 3 - 0}{2} = 3$$

$$d_{DU} = \frac{d_{AD} - d_{AU} + d_{BD} - d_{BU}}{2} = 2$$

Macierz po redukcji:

	C	D
U	3	2
C		3

## Algorytm NJ - przykład (3)

Macierz po redukcji:

	C	D
U	3	2
C		3

korekcje:

Macierz po

	C	D
U	-2.5	-3
C		-2.5

Nowy węzeł  $V$  pomiędzy  $U$  i  $D$ 

$$r'_U = \frac{r_U}{3-2} = 5$$

$$r'_D = 6$$

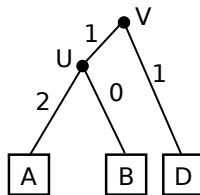
$$d_{UV} = \frac{d_{UD} + r'_U - r'_D}{2} = 1$$

$$d_{DV} = \frac{d_{UD} + r'_D - r'_U}{2} = 1$$

$$r_U = 3 + 2 = 5$$

$$r_C = 6$$

$$r_D = 5$$

Minimalna odległość to  $d_{UD}$ 



Po redukcji:

```

graph TD
    Root(( )) -- U --> UNode((U))
    Root -- V --> VNode((V))
    UNode -- 0 --> A[A]
    UNode -- 1 --> B[B]
    VNode -- 1 --> D[D]
    VNode -- 2 --> C[C]
  
```

Tabela odległości odtworzona z drzewa jest identyczna z tabelą pierwotną

# ***Analiza danych wielowymiarowych***

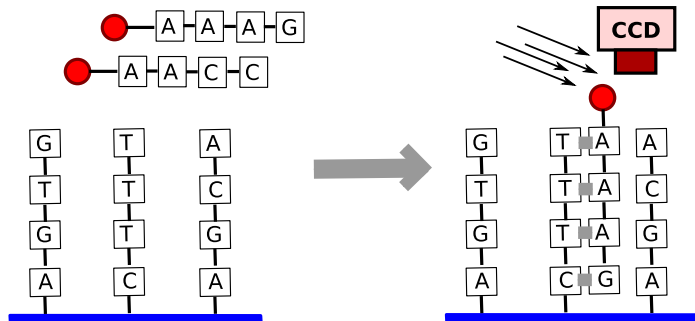
# techniki analizy danych wielowymiarowych

- ▶ nienadzorowane - nie uwzględniamy przypisania obiektów do klas
  - ▶ grupowanie
  - ▶ redukcja wymiarów
- ▶ nadzorowane - uwzględniamy przypisanie obiektów do klas
  - ▶ klasyfikacja
  - ▶ selekcja cech różnicujących

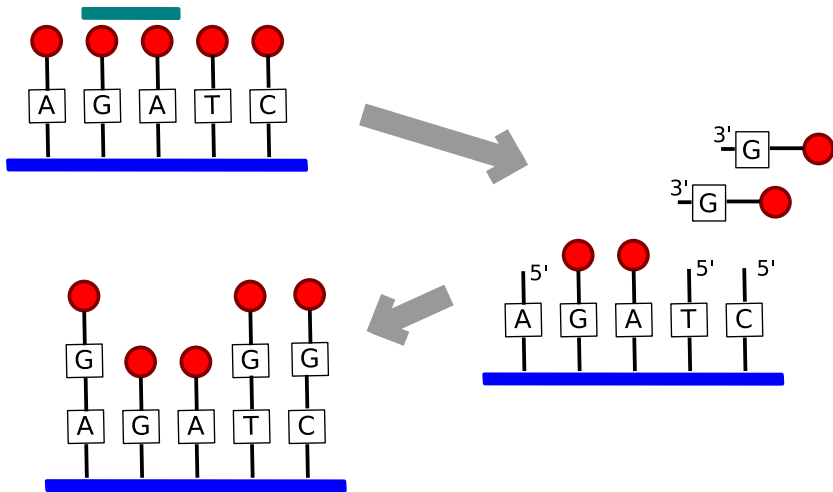
# mikromacierze DNA

Metoda badawcza, pozwalająca badać obecność wielu cząsteczek DNA lub RNA jednocześnie, utworzona do odczytywania sekwencji

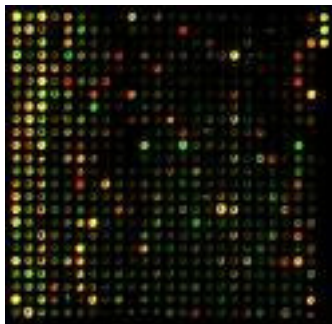
Badanie roztworów za pomocą mikromacierzy DNA:



## tworzenie mikromacierzy DNA



## wynik badania



Dla każdego doświadczenia:

- ▶  $m$  atrybutów ( $m \approx 10^5$ )
- ▶ wartości:
  - ▶ binarne: występuje, nie występuje
  - ▶ rzeczywiste: intensywność  $< 0, 1 >$

- ▶ zwykle przeprowadza się  $n \approx 100$  doświadczeń
- ▶ mamy macierz  $n \times m$ , gdzie  $n \ll m$  elementów  $x_{ij}$
- ▶ problem znalezienia atrybutów istotnych

# normalizacja danych

- ▶ usuwanie atrybutów, które mają tę samą wartość dla wszystkich przykładów
- ▶ obliczenie średniej i odchylenia standardowego

$$\mu_j = \frac{1}{n} \sum_{i=0}^n x_{ij} \text{ średnia}$$

$$\sigma_j = \sqrt{\frac{1}{n} \sum_{i=0}^n (x_{ij} - \mu_j)^2} \text{ odchylenie standardowe}$$

- ▶ przekształcenie danych, atrybut  $j$  ma  $\mu = 0$ ,  $\sigma = 1$

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

# *Redukcja wymiaru*



# algorytm analizy składowych głównych (PCA)

- ▶ pozwala na redukcję wymiaru problemu
- ▶ transformuje (liniowo) przestrzeń atrybutów dostarczając nowych współrzędnych

Dane wejściowe:

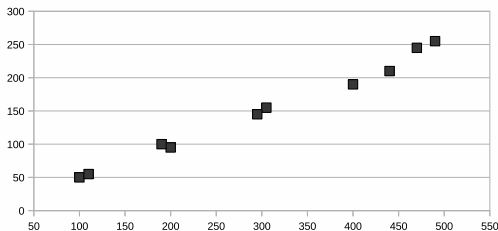
pomiar	atrybut			
	1	2	...	$m$
1	$x_{11}$	$x_{12}$	...	$x_{1m}$
2	$x_{21}$	$x_{22}$	...	$x_{2m}$
...	...	...	...	...
$n$	$x_{n1}$	$x_{n2}$	...	$x_{nm}$

- ▶ obliczenie  $\mu_j$  oraz  $\sigma_j$
- ▶ normalizacja danych

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

# algorytm PCA (2)

dane wejściowe (10 przykładów, 2 atrybuty):



$$\mu_1 = 300$$

$$\sigma_1 = 146.4$$

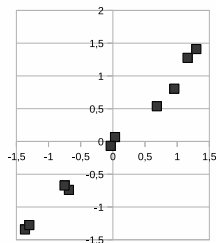
$$\mu_2 = 150$$

$$\sigma_2 = 74.4$$

normalizacja:

$$z_{i1} = \frac{x_{i1} - \mu_1}{\sigma_1}$$

$$z_{i2} = \frac{x_{i2} - \mu_2}{\sigma_2}$$



## algorytm PCA (3)

po normalizacji wszystkie atrybuty mają parametry:

►  $\mu = 0$

►  $\sigma = 1$

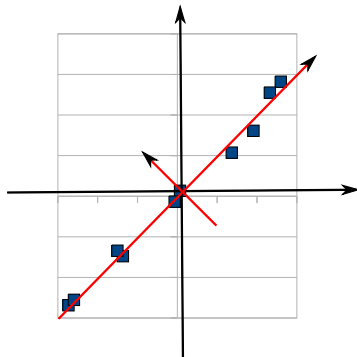
chcemy znaleźć nowy układ współrzędnych

- zakładamy tylko przekształcenia liniowe (obroty, odbicia)



## algorytm PCA (4)

Kierunki składowych dla rozpatrywanego przykładu:



Założenia algorytmu PCA:

- ▶ rozpatrywane przekształcenia liniowe
- ▶ maksymalizowana jest wariancja (wariancja - klasyczna miara zróżnicowania)
- ▶ nowe kierunki składowych są ortogonalne

## algorytm PCA (5) - kowariancja

$$\mathbf{Z} = \begin{bmatrix} Z_{11} & Z_{12} & \dots & Z_{1m} \\ Z_{21} & Z_{22} & \dots & Z_{2m} \\ \dots & \dots & \dots & \dots \\ Z_{n1} & Z_{n2} & \dots & Z_{nm} \end{bmatrix} \quad \mathbf{a}_i = \begin{bmatrix} Z_{1i} \\ Z_{2i} \\ \dots \\ Z_{ni} \end{bmatrix} \quad \mathbf{a}_j = \begin{bmatrix} Z_{1j} \\ Z_{2j} \\ \dots \\ Z_{nj} \end{bmatrix}$$

dane po normalizacji:  $\mu_{a_i} = \frac{1}{n} \sum_{k=0}^n z_{ki} = 0, \sigma_{a_i}^2 = \frac{1}{n} \sum_{k=0}^n z_{ki}^2 = 1$

Kowariancja - miarą liniowej zależności pomiędzy  $a_i$  i  $a_j$

$$\sigma_{a_i a_j} = \frac{1}{n} \sum_{k=0}^n z_{ik} z_{jk} = \frac{1}{n} \mathbf{a}_i \mathbf{a}_j^T \text{ gdzie } \mathbf{a}_j^T = \begin{bmatrix} z_{1j} & z_{2j} & \dots & z_{nj} \end{bmatrix}$$
$$-1 \leq \sigma_{a_i a_j} \leq 1$$

## algorytm PCA (6) - macierz kowariancji

$$\mathbf{C}_Z = \frac{1}{n} \mathbf{Z} \mathbf{Z}^T = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1m} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2m} \\ \dots & \dots & \dots & \dots \\ \sigma_{n1} & \sigma_{n2} & \dots & \sigma_m^2 \end{bmatrix} = \begin{bmatrix} 1 & \sigma_{12} & \dots & \sigma_{1m} \\ \sigma_{21} & 1 & \dots & \sigma_{2m} \\ \dots & \dots & \dots & \dots \\ \sigma_{n1} & \sigma_{n2} & \dots & 1 \end{bmatrix}$$

Ponieważ  $\sigma_{ij} = \sigma_{ji}$  macierz  $\mathbf{C}_Z$  jest symetryczna

- ▶ sumaryczna wariancja

$$\sum_{j=0}^m \sigma_i = m$$

- ▶ po zmianie (rotacja, odbicie) układu współrzędnych sumaryczna wariancja nie zmienia się

# algorytm PCA (7) - przekształcenie układu współrzędnych

$\mathbf{Y} = \mathbf{PZ}$ , gdzie  $\mathbf{P}$  jest macierzą przekształcenia

, macierz  $\mathbf{P}$  zawiera wektory, które są kierunkami składowych

$$\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m]$$

Wykorzystując algorytmy algebry liniowej przekształca się przestrzeń, aby macierz kowariancji była diagonalna

$$\mathbf{C}_Y = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & \lambda_m \end{bmatrix}$$

algorytm PCA (8) - jak znaleźć przekształcenie **P**

$$\mathbf{C}_Y = \frac{1}{n} \mathbf{Y} \mathbf{Y}^T = \frac{1}{n} (\mathbf{P} \mathbf{Z}) (\mathbf{P} \mathbf{Z})^T = \frac{1}{n} \mathbf{P} \mathbf{Z} \mathbf{Z}^T \mathbf{P}^T = \mathbf{P} \mathbf{C}_Z \mathbf{P}^T$$

dla macierzy symetrycznej **A**, macierzy jej wektorów własnych **E** zachodzi zależność:

$$\mathbf{A} = \mathbf{E} \mathbf{D} \mathbf{E}^T, \text{ gdzie } \mathbf{D} \text{ jest macierzą diagonalną}$$

więc: **P** jest macierzą wektorów własnych macierzy  $\mathbf{C}_Z$



## algorytm PCA (9) - przykład

Dla rozpatrywanego przykładu:

$$\mathbf{C}_Z = \begin{bmatrix} 1 & 0.994 \\ 0.994 & 1 \end{bmatrix}$$

po rozkładzie na wartości własne i wektory własne:

$$\begin{bmatrix} 1 & 0.994 \\ 0.994 & 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} 1.994 & 0 \\ 0 & 0.006 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix}$$

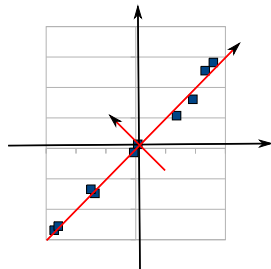
Nowe kierunki

$$\mathbf{p}_1 = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{bmatrix}, \mathbf{p}_2 = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} \end{bmatrix}, \lambda_1 = 1.994, \lambda_2 = 0.006$$

## algorytm PCA (10) - przykład

$$\lambda_1 = 1.994, \lambda_2 = 0.006$$

- ▶ Po zamianie współrzędnych i eliminacji drugiego wymiaru będziemy mieli 99.7% wariancji dla pierwszego wymiaru (utracimy tylko 0.3% wariancji)
- ▶ W ten sposób można wybrać tylko istotne atrybuty



## algorytm PCA (11) - podsumowanie

- ▶ algorytm nie posiada parametrów
- ▶ liniowo przekształca przestrzeń atrybutów
- ▶ wykorzystuje macierz korelacji, eliminuje kowariancję czyli liniowe zależności pomiędzy atrybutami
- ▶ pozwala na redukcję wymiarów, opuszczanie atrybutów o mniejszym znaczeniu (kompresja informacji)

Popularne kryterium dobierania ilości składowych (kryterium Kaisera-Guttmana)

należy zachować składowe, dla których wartości własne są większe od 1, czyli wkład składowej większy, niż wkład pojedynczej zmiennej

*Dziękuję*