Analiza skupień

Agnieszka Sołtys

Uczenie nienadzorowane

Cechy:

- Brak oczekiwanego wyjścia w danych uczących
- Odkrywanie wzorców w zbiorach danych

Najważniejsze przykłady:

- Transformacje danych, np. skalowanie, analiza składowych głównych (PCA)
- Analiza skupień

Uczenie nienadzorowane

Główne wyzwanie:

Ocena, czy algorytm nauczono czegoś pożytecznego!

Uczenie nienadzorowane

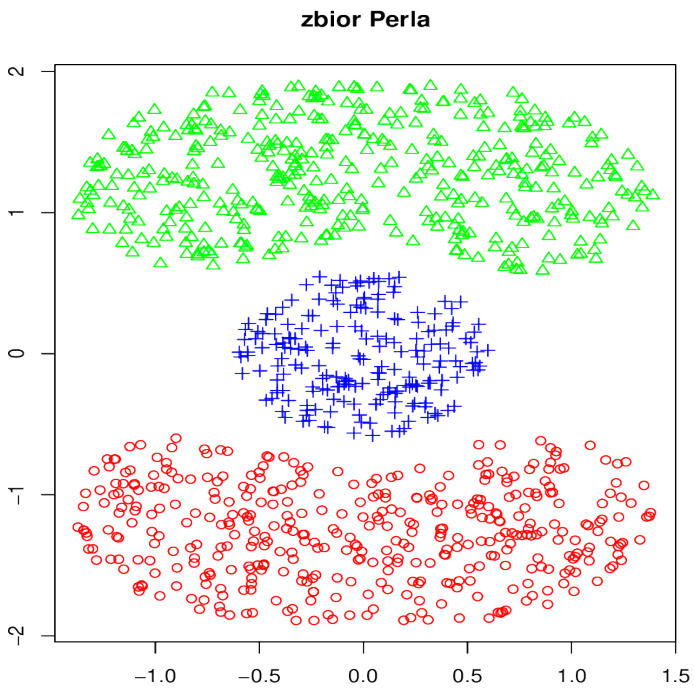
Główne wyzwanie:

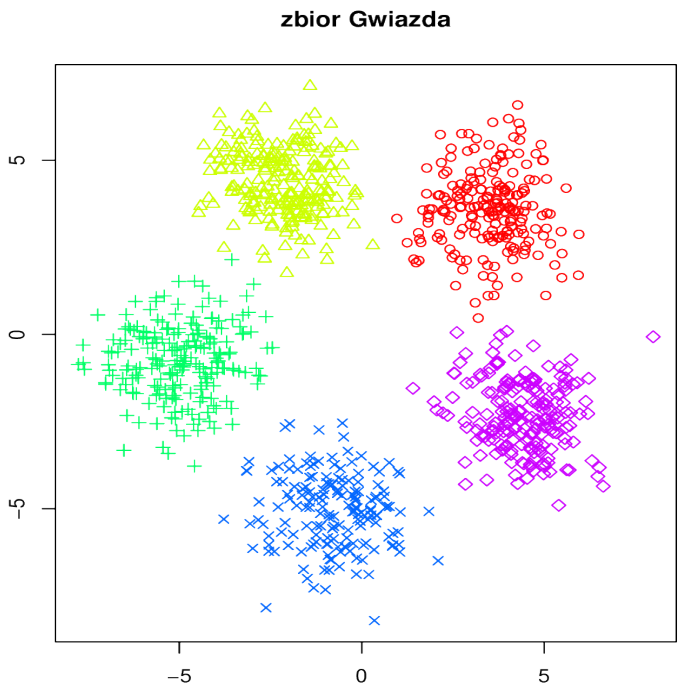
Ocena, czy algorytm nauczono czegoś pożytecznego!

Najczęściej inspekcja ręczna (sensowność i użyteczność)

Analiza skupień (ang. cluster analysis)

Podział zbioru danych na rozłączne grupy (klastry) podobnych elementów





1

[1] Źródło:https://pbiecek.github.io/NaPrzelajDataMiningR/part-3.html

Analiza skupień - przykładowe zastosowania

- Segmentacja klientów
  - dopasowanie odpowiedniej oferty/ udzielenie promocji
  - ze względu na zainteresowanie lub określone zachowanie
- Grupowanie tekstów
  - o zblizonej tematyce
- Grupowanie obrazów
  - podobnych twarzy w serwisie społecznościowym
  - segmentacja obrazu - podział obrazu na regiony homogeniczne pod względem pewnej własności (kolor, intensywność)
- Przetwarzanie wstępne danych
  - Eksploracja
  - Agregacja

`https://www.statology.org/cluster-analysis-real-life-examples/`

Plan na weekend

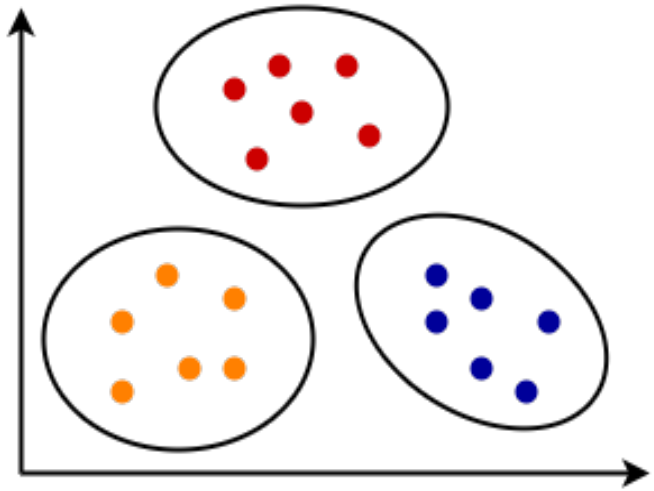
- Grupowanie k-średnich
- Grupowanie hierarchiczne aglomeracyjne
- Algorytm DBSCAN
- Porównanie i ocena algorytmów grupowania
- Grupowanie danych mieszanych
- Algorytm BIRCH dla dużych zbiorów danych
- Algorytm Mini-Batch K-Means dla dużych zbiorów danych

Grupowanie k-średnich

Grupowanie k-średnich

Pomysł: suma odległości obserwacji należących do klastra jest znacznie mniejsza od sumy odległości obserwacji pomiędzy klastrami.

Before K-Means



After K-Means

2

[2] Źródło: https://www.gatevidyalay.com/k-means-clustering-algorithm-example/

Grupowanie k-średnich

Znaleźć klastry (grupy):
$D = C_1 \cup C_2 \cup \ldots \cup C_k$ takie, że zmienność wewnątrzgrupowa (Within Sum of Squares) będzie minimalna:

$$WSS = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 = \sum_{i=1}^{k} |C_i| VarC_i$$

dla środków klastrów:

$$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$$

Grupowanie k-średnich

Algorytm zmniejszający WSS w każdym kroku:

- Dla danych środków klastrów przyporządkuj obserwacje do najbliższego środka
- Dla danego przyporządkowania do klastrów oblicz nowe środki jako średnie z obserwacji

`https://www.youtube.com/watch?v=5I3Ei69I40s`
`https://www.naftaliharris.com/blog/visualizing-k-means-clustering/`

Grupowanie k-średnich

Uwagi:

- Początkowe środki klastrów są często wybierane losowo (warto algorytm uruchomić wielokrotnie)
- Algorytm się zatrzymuje gdy kolejne kroki algorytmu nie zmieniają przyporządkowania do klastrów (minimum lokalne), bądź zmienność wewnątrzgrupowa przestaje się istotnie zmniejszać
- Algorytm zakłada konkretną liczbę klastrów
- Każdy klaster jest zdefiniowany wyłącznie przez jego środek

`https://www.youtube.com/watch?v=9nKfViAfajY`

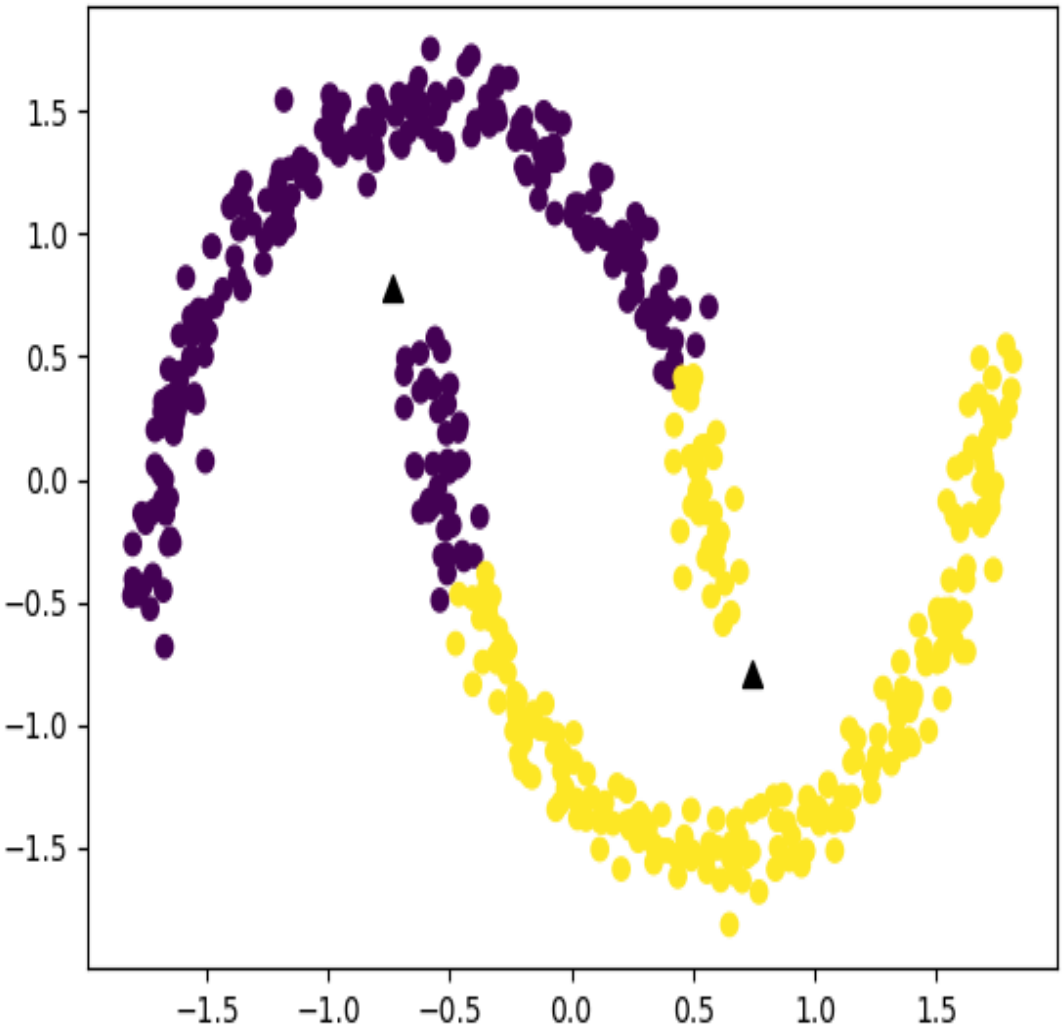
Grupowanie k-średnich dla jakich danych?

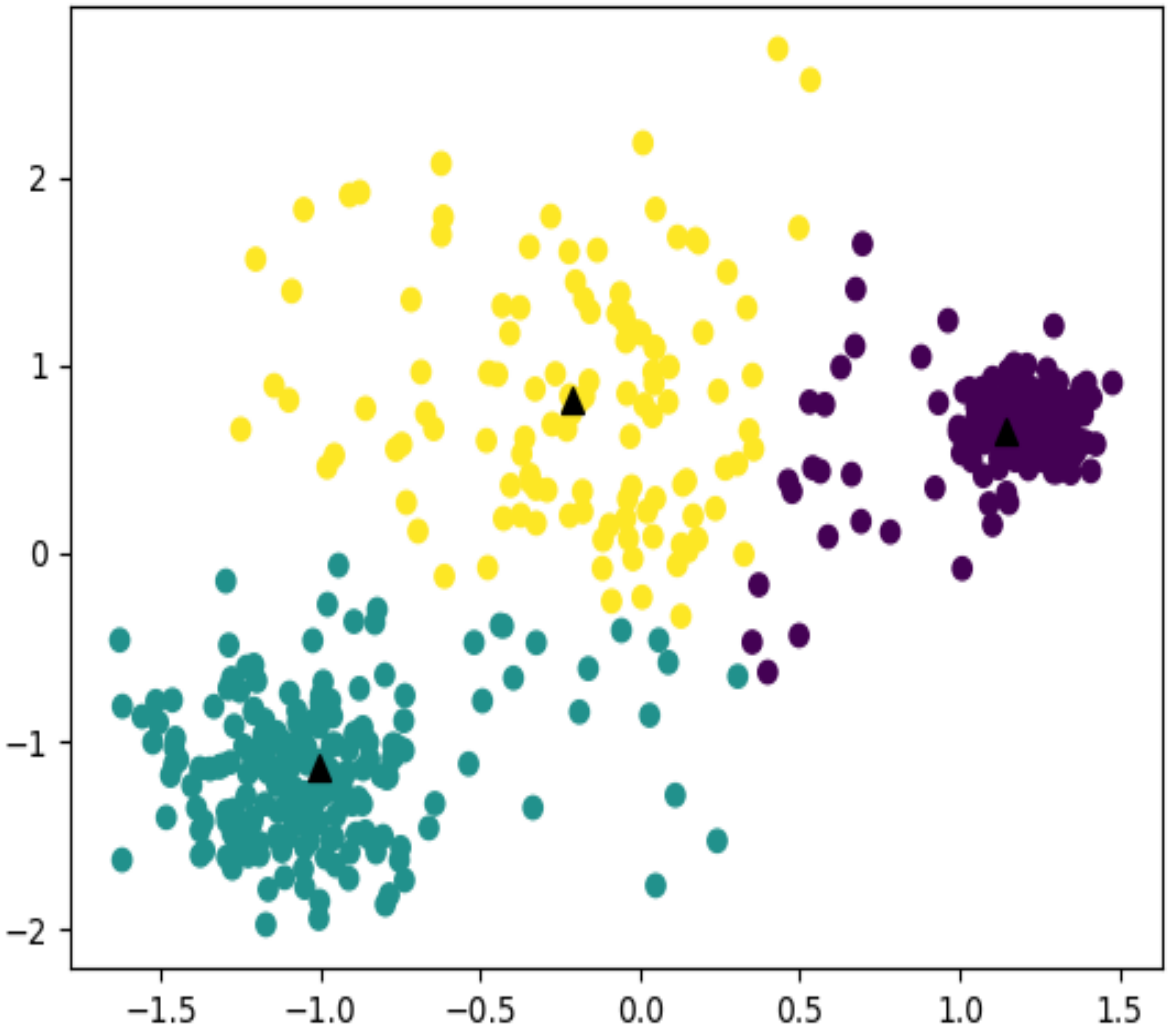
Zalety:

- Prosty, szybko zbiega nawet dla dużych zbiorów danych
- Popularny, zwykle używany jako "pierwszy wybór"

Wady:

- Działa dla konkretnej liczby klastrów
- Czuły na obserwacje odstające
- Nie radzi sobie ze złożonymi kształtami klastrów - zakłada wypukłe klastry ("kuliste") o tej samej wariancji ("średnicy"), każdy klaster jest zdefiniowany przez jego środek, granica pomiędzy dwoma klastrami zawsze znajduje się w równej odległości pomiędzy środkami klastrów


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


2
1
0
−1
−2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

Wybór k w algorytmie k-średnich

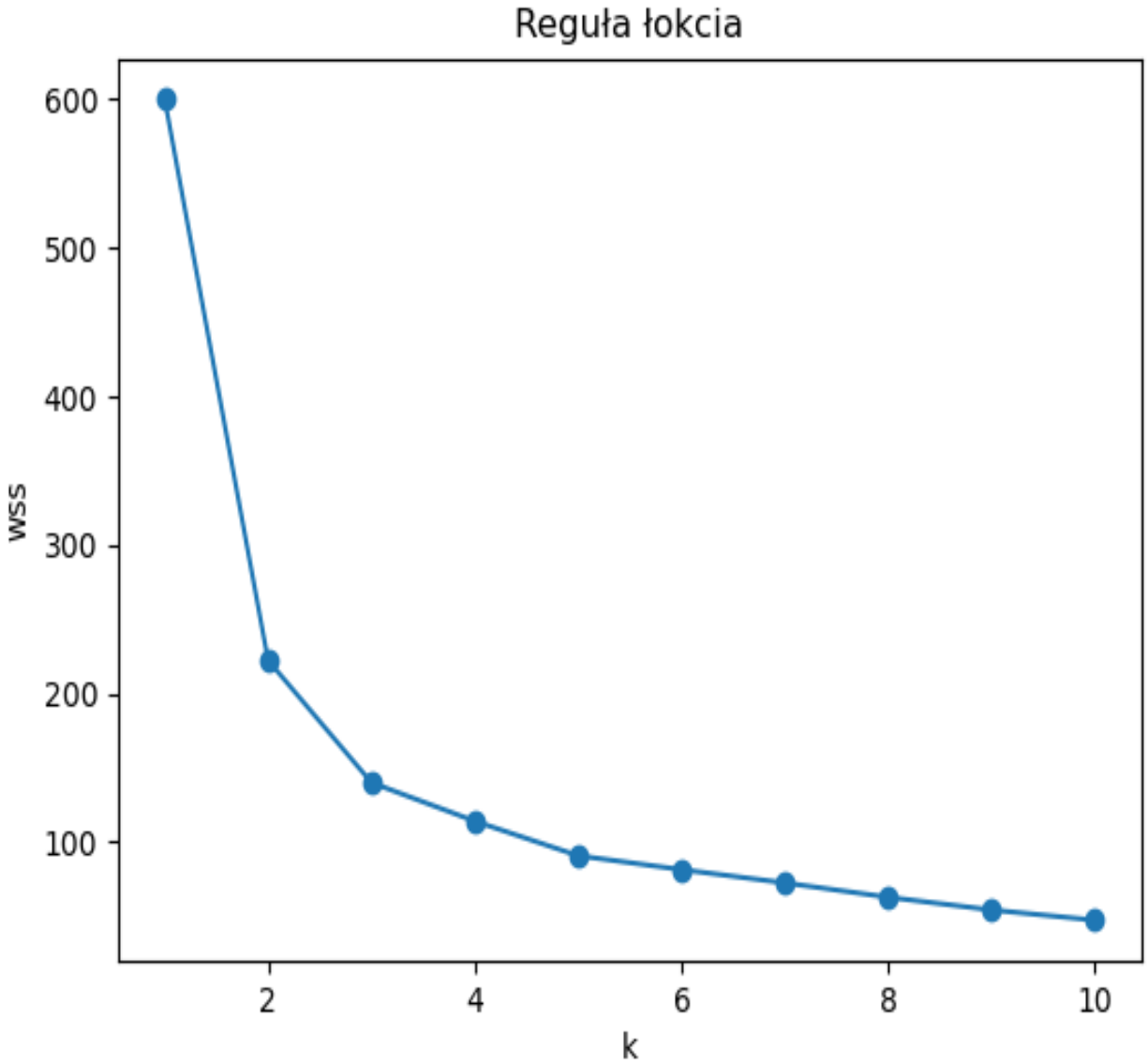
Ocena jakości grupowania jest trudna z powodu braku danych wyjściowych (zmiennej objaśnianej).

Metody wyboru k w k-średnich

1. Wiedza ekspercka
   - Jeżeli robimy segmentację klientów w celu skierowania do nich oferty, liczba ofert może być określona przez realia biznesowe lub należeć do jakiegoś przedziału
   - Często zdrowy rozsądek może nam podpowiedzieć, czy liczba szukanych klastrów to 10 czy 50, np. jeśli szukamy grup podobnych twarzy na serwisie społecznościowym.
2. Inspekcja ręczna - podczas wnikliwego oglądania danych możemy zauważyć, czy nie grupują się one w wyraźny sposób
3. Miary numeryczne:
   - Reguła łokcia dla wykresu zmienności wewnątrzgrupowej (WSS) w zależności od k
   - Maksymailizacja indeksu sylwetki (silhouette)

Reguła łokcia dla wykresu WSS w zależności od k

- WSS to miara, którą chcemy minimalizować podczas uczenia algorytmu k-średnich.
- Jeżeli narysujemy wykres zależności WSS od k, okaże się, że funkcja jest malejąca - dodając kolejny środek, suma odległości punktów od najbliżeszgo środka spada.
- Funkcja ta ma często punkt, w którym się wychyla - punkt zgięcia. Wartość k w tym punkcie jest zwykle rozsądnym kompromisem pomiędzy błędem a liczbą klastrów.


Reguła łokcia
600
500
400
300
200
100
wss
2
4
6
8
10
k

Indeks sylwetki (silhouette)

- Indeks sylwetki mierzy, jak obserwacja jest podobna do tych ze swojego klastra w porównaniu do tych z innych klastrów.
- Indeks sylwetki przyjmuje wartości od -1 do 1, gdzie wysoka dodatnia wartość oznacza dobre dopasowanie do swojego klastra i słabe do sąsiadujących klastrów. Wartość 0 oznacza, że obserwacja jest blisko granicy decyzyjnej pomiędzy dwoma sąsiadującymi klastrami, a wartość ujemna oznacza przyporządkowanie do złego klastra.
- Do wyboru k użyjemy średniego indeksu sylwetki dla wszystkich obserwacji w zbiorze danych.

Indeks sylwetki (silhouette)

$$a(i) = \frac{1}{|C_I| - 1} \sum_{j \in C_I, i \neq j} d(i,j),$$

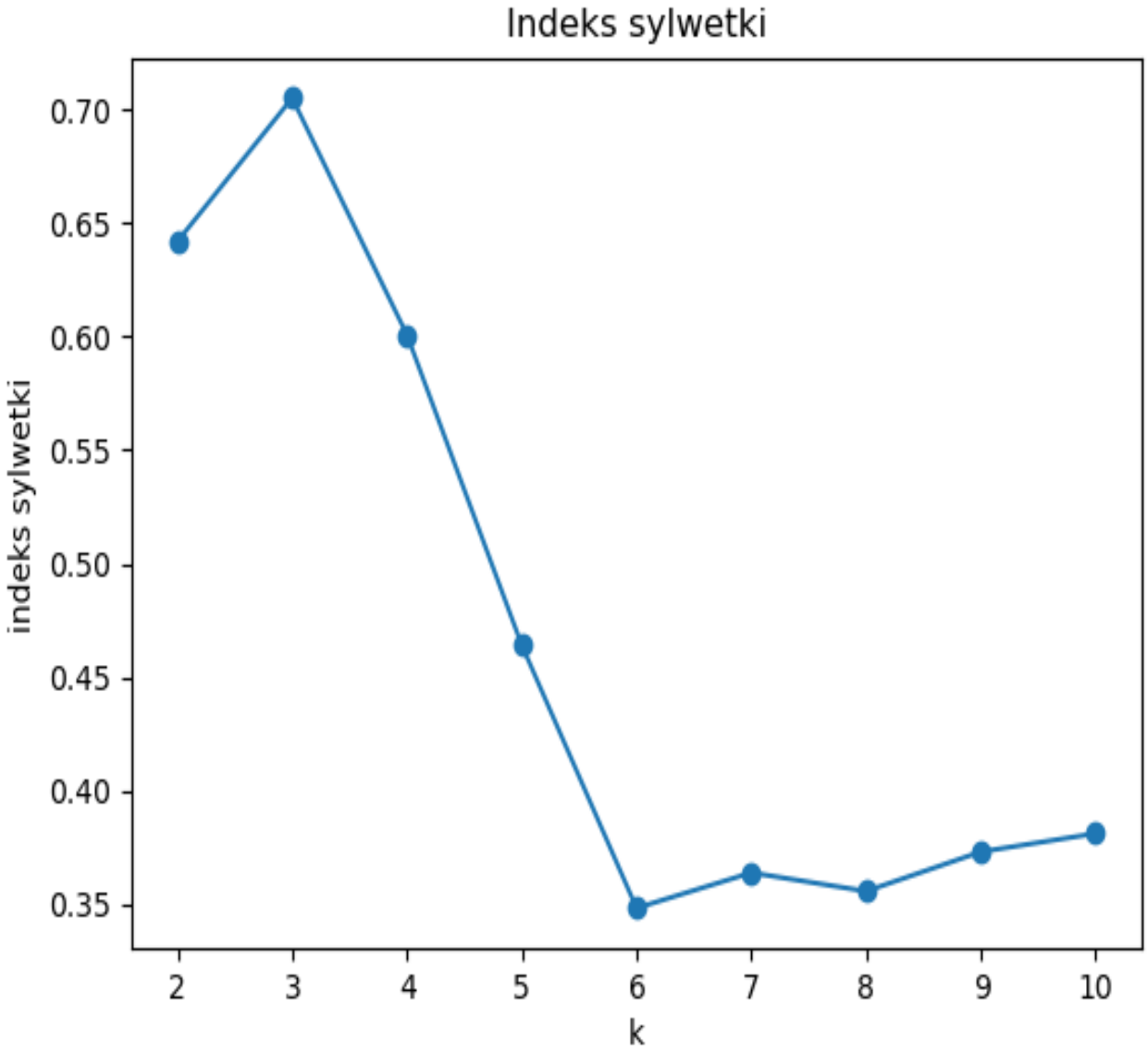
średnia odległość obserwacji i od obserwacji w tym samym klastrze I

$$b(i) = \min_{J \neq I} \frac{1}{|C_J|} \sum_{j \in C_J} d(i,j), i \in C_I$$

najmniejsza średnia odległość obserwacji i od wszystkich obserwacji w innym klastrze (od klastra sąsiada J)

$$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))}$$

jeżeli $|C_I| > 1$, $s(i) = 0$ jeżeli$|C_I| = 1$.


Indeks sylwetki
indeks sylwetki
0.70
0.65
0.60
0.55
0.50
0.45
0.40
0.35
2
3
4
5
6
7
8
9
10
k

Grupowanie aglomeracyjne (metoda hierarchiczna)

Grupowanie aglomeracyjne

Pomysł: zaczynając od najdrobniejszego podziału, gdzie każda obserwacja to osobny klaster, łączyć w każdym kroku dwa najbliższe klastry.

- Grupowanie aglomeracyjne buduje całą hierarchię grupowań, od najdrobniejszego do jednego dużego klastra zawierającego wszystkie obserwacje.
- Drzewo nazywane dendrogramem to narzędzie do wizualizacji hierarchii grupowań.
- Dendrogram pomaga w wyborze liczby klastrów.

Kroki algorytmu grupowania aglomeracyjnego

1. Każda obserwacja to oddzielny klaster
2. Oblicz macierz odległości dla obserwacji za pomocą wybranej miary (np. odległość euklidesowa)
3. Połącz ze sobą dwa najbliższe klastry
4. Przelicz macierz odległości dla nowych klastrów
5. Powtarzaj kroki 3 i 4 aż wszystkie obserwacje będą w jednym klastrze

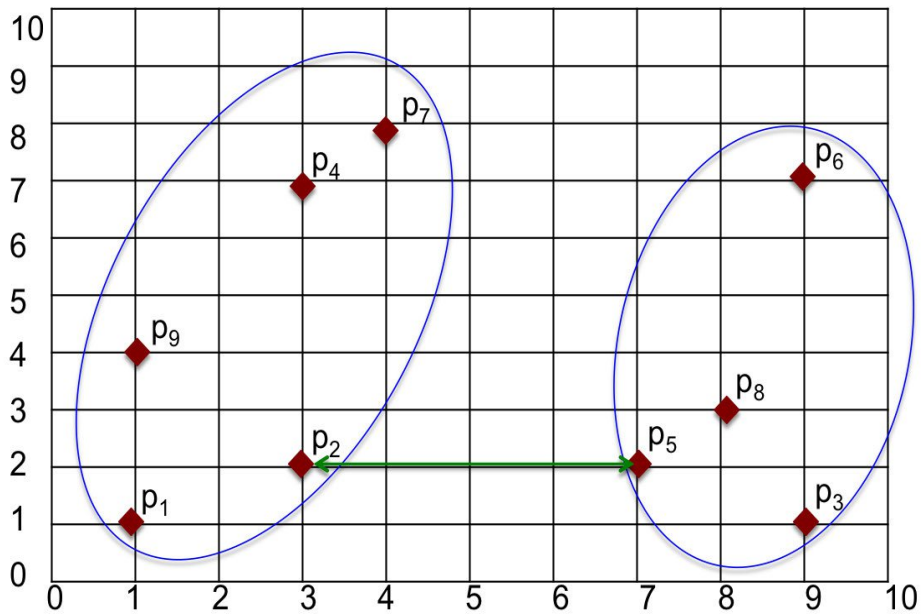
Metody obliczania odległości pomiędzy klastrami

Metoda Single linkage

Single linkage - minimalna odległość pomiędzy punktami w dwóch klastrach

$$d(C_i, C_j) = min_{x \in C_i, x' \in C_j} d(x, x')$$

np. rozprzestrzenianie się pożaru



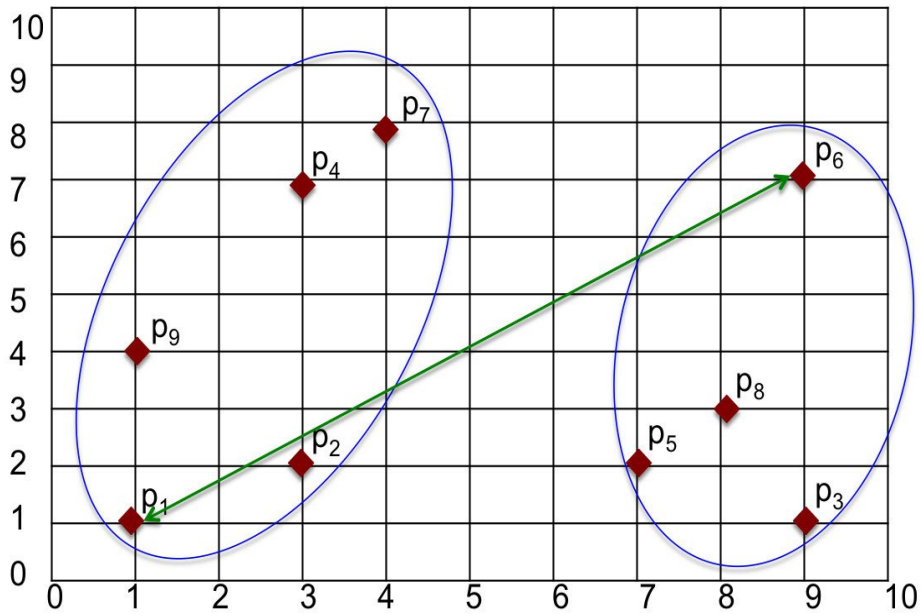
[3] Źródło: https://www.learndatasci.com/glossary/hierarchical-clustering/

Metoda complete linkage

Complete linkage - maksymalna odległość pomiędzy punktami w dwóch klastrach

$$d(C_i, C_j) = max_{x \in C_i, x' \in C_j} d(x, x')$$

np. partie polityczne

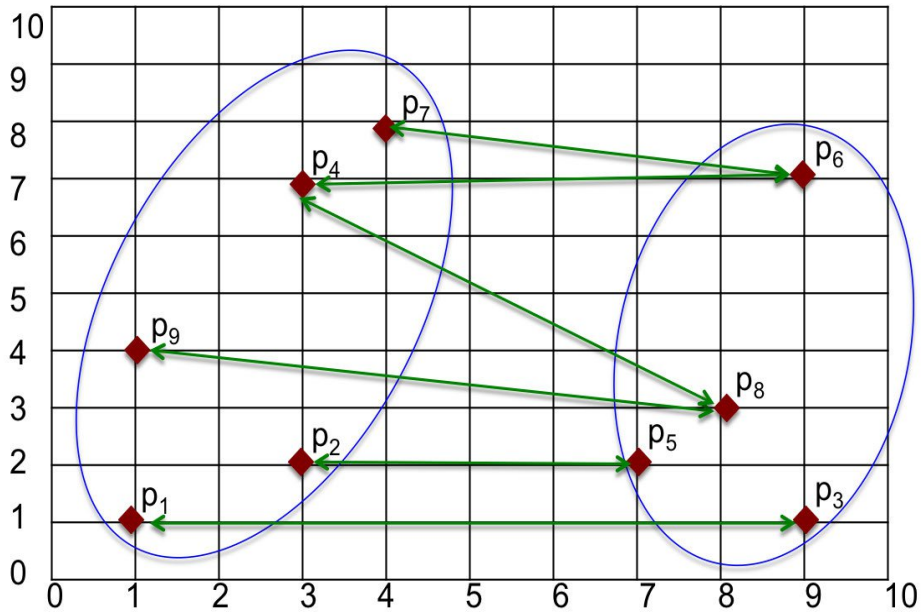


[4] Źródło: https://www.learndatasci.com/glossary/hierarchical-clustering/

Metoda average linkage

Average linkage - średnia odległość pomiędzy punktami w dwóch klastrach

$$d(C_i, C_j) = \frac{\sum_{x \in C_i, x' \in C_j} d(x, x')}{|C_i| \cdot |C_j|}$$



[5] Źródło: https://www.learndatasci.com/glossary/hierarchical-clustering/

Metoda Ward linkage

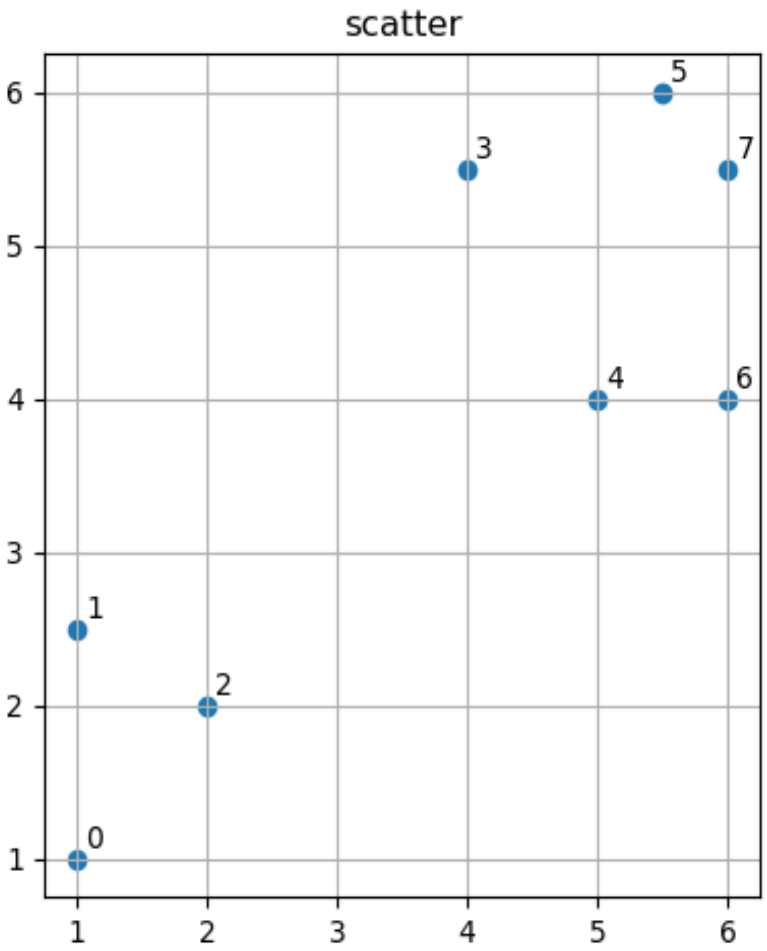
Ward linkage:

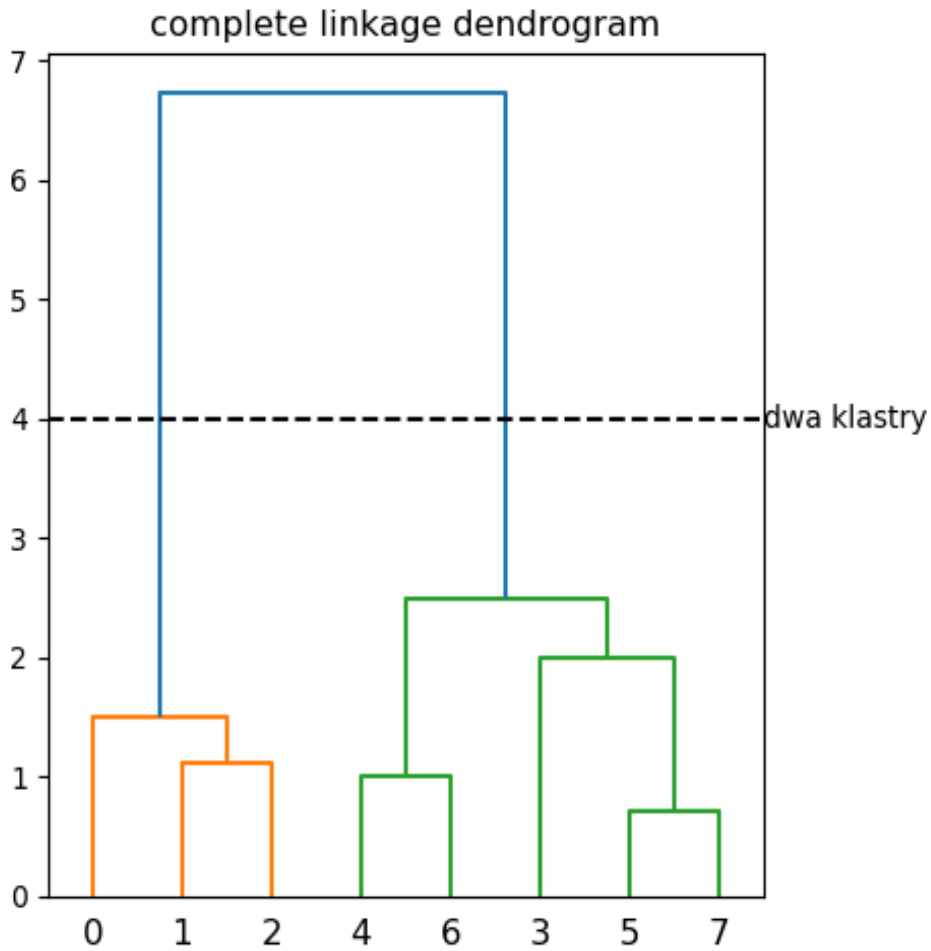
- zamiast mierzyć odległości pomiędzy punktami, analizuje wariancję klastrów.
- Nowa odległość pomiędzy klastrami to wartość, o jaką wzrośnie suma kwadratów (odległości od środków klastrów) jeżeli połączymy 2 klastry
- Podobieństwo do k-średnich: jak najmniejszy wzrost WSS w każdym kroku.

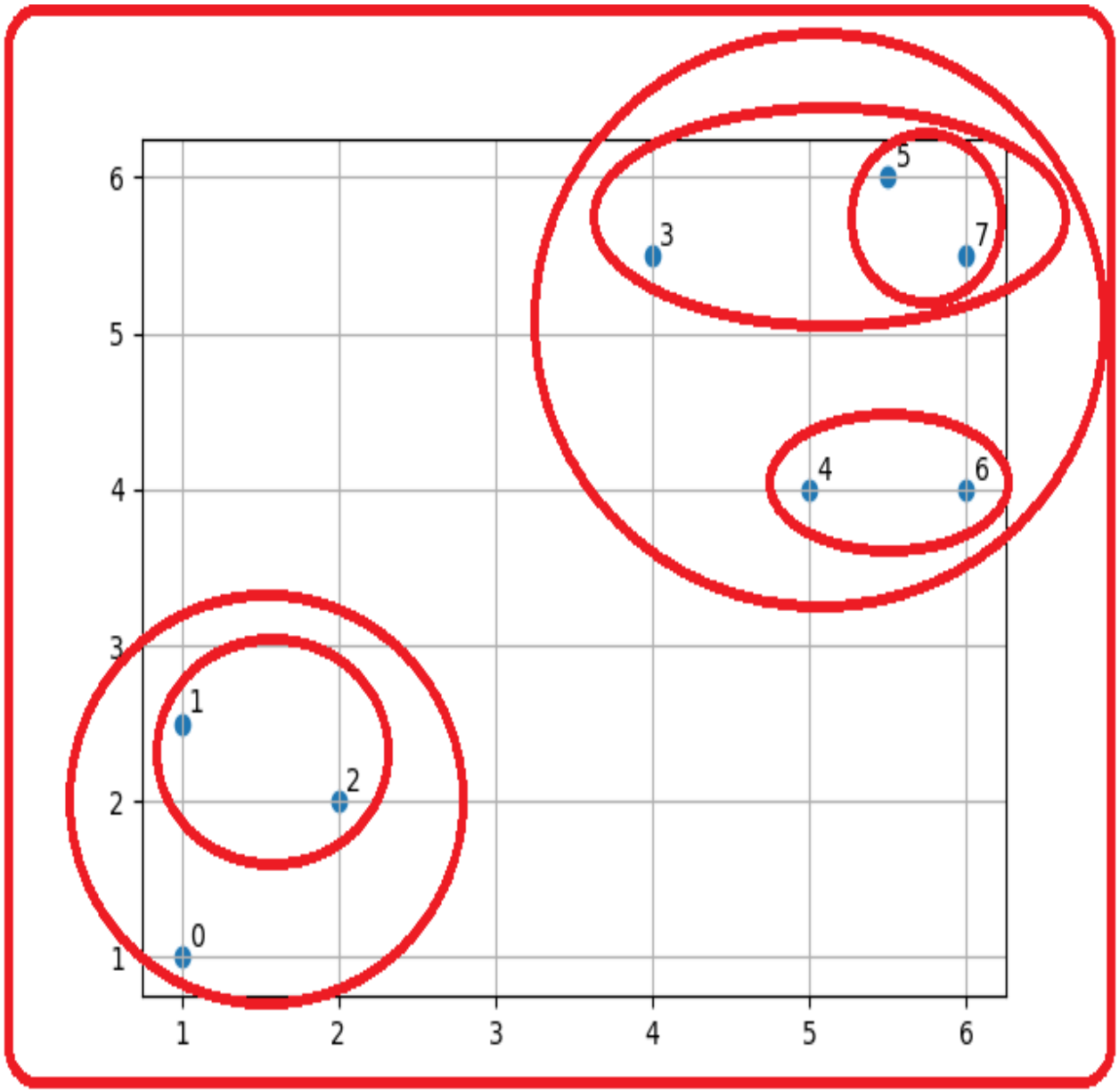
$$d(C_i, C_j) = \sum_{x \in C_i \cup C_j} ||x - \mu_{i \cup j}||^2 - \sum_{x \in C_i} ||x - \mu_i||^2 - \sum_{x \in C_j} ||x - \mu_j||^2$$

$$= |C_i + C_j| Var C_{i \cup j} - |C_i| Var C_i - |C_j| Var C_j,$$

$\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x.$


scatter
0
1
2
3
4
5
6
7


complete linkage dendrogram
dwa klastry
0
1
2
4
6
3
5
7


6
5
4
3
2
1
1
2
3
4
5
6
0
1
2
3
4
5
6
7

Wybór liczby klastrów

Do wyboru liczby klastrów używamy dendrogramu (jeżeli nie zakładamy konkretnej liczby klastrów):

- Patrząc na dendrogram oceniamy ile jest sensownych klastrów
- Wybieramy miejsce, gdzie mamy najdłuższy pionowy odcinek nieprzecięty poziomą linią.

Metoda **Single** linkage potrafi separować niesferyczne kształty, pod warunkiem, że nie ma pomiędzy nimi szumu. Produkuje wydłużone kształty klastrów
Metoda **Complete** dobrze radzi sobie z szumem, klastry mają kształty kul o podobnej średnicy, kiepsko radzi sobie z klastrami o znacznych różnicach w rozmiarach, często dzieli duże klastry
Metoda **Average** kompromis pomiędzy Single i Complete, dobrze radzi sobie z szumem, klastry mają kształty kul o podobnej średnicy
Metoda **Warda** dobrze radzi sobie z szumem, klastry mają kształty kul o podobnej średnicy (podobna w praktyce do Average)

Grupowanie z ograniczeniami

Klastrowanie hierarchiczne pozwala na wprowadzenie ograniczeń w grupowaniu

- Tylko sąsiednie klastry mogą być łączone ze sobą, gdzie sąsiedztwo jest zdefiniowane przez macierz połączeń.
- Macierz połączeń to rzadka macierz, w której niezerowe są miejsca dla przecięć kolumn i wierszy, o indeksach obserwacji, które mogą zostać zgrupowane.
- Np. klastrowanie stron internetowych poprzez łączenie tylko tych, które zawierają link przenoszący użytkownika z jednej strony na drugą.
- Można dyktować algorytmowi lokalną strukturę danych.
- Często przyśpiesza działanie algorytmu, jeżeli liczba obserwacji jest duża.

Algorytm DBSCAN

DBSCAN

Grupowanie oparte na gęstości, radzące sobie z szumem i obserwacjami odstającymi (ang. density-based spatial clustering of applications with noise).

DBSCAN

Algorytm DBSCAN ma 2 najważniejsze parametry: *min_samples* i *eps*.

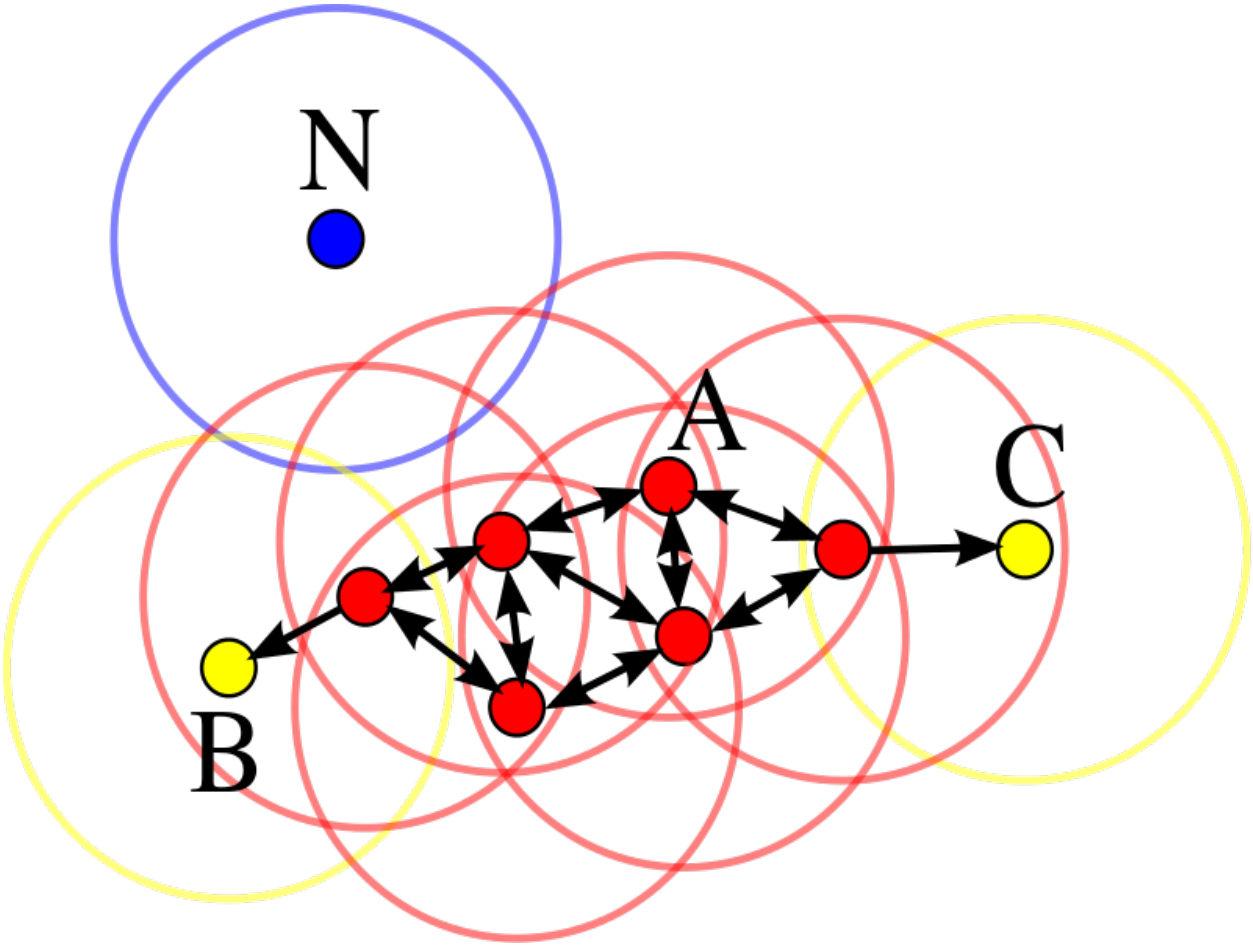
1. Wybierz dowolny punkt.
2. Znajdź wszystkie punkty w odległości $\leq$ *eps* od tego punktu. Jeżeli punktów jest $<$ *min_samples*, oznacz punkt jako szum. W.p.p. oznacz punkt jako punkt główny, przypisz punkt do nowej etykiety klastra.
3. Sprawdź wszystkie punkty sąsiednie (w obrębie *eps*) punktu głównego. Jeśli nie mają przypisanego klastra, przypisz właśnie utworzoną etykietę klastra.
4. Jeśli w punkcie 3 znaleziono punkty główne, sprawdź ich sąsiadów, itd. aż w odległości *eps* od klastra nie ma żadnych punktów głównych.
5. Wybierz inny punkt, który nie został sprawdzony, i wróć do punktu 2, aż wszystkie punkty będą w klastrach albo oznaczone jako szum.

DBSCAN

3 rodzaje punktów:

1. Punkty główne - znajdujące się w gęstym regionie: jeśli co najmniej *min_samples* punktów danych znajduje się w odległości *eps* od tego punktu. Punkty główne, które są bliżej siebie niż *eps*, zostają umieszczone w tym samym klastrze.
2. Punkty graniczne - w obrębie *eps* od punktów głównych (nie będące punktami głównymi), mogą się znajdować w sąsiedztwie więcej niż jednego klastra, ich przynależność do klastrów zależy od kolejności wyświetlania punktów. Rzadko wpływa to istotnie na całkowite grupowanie.
3. Szum.

min_samples = 4



https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

DBSCAN

- Nie wymaga ustawiania liczby klastrów (zaleta i wada)
- Identyfikuje obserwacje odstające, nienależące do żadnego klastra
- Algorytm identyfikuje gęste (zatłoczone) obszary danych, oddzielone regionami stosunkowo pustymi
- Klastry mogą mieć bardzo różne kształty, ale muszą być połączone "gęstą drogą"
- Wymaga starannego doboru parametrów *min_samples* i *eps*

Dobór parametrów

Wybór parametrów najlepiej oprzeć na wiedzy eksperckiej oraz znajomości zbioru danych. Istnieją jednak pewne wskazówki oparte na praktyce.

Dla wyboru *min_samples*:

- $min\_samples = 1$ nie ma sensu, trywializuje się pojęcie „gęstych" obszarów.
- Im większy zbiór danych, tym większa powinna być wartość parametru.
- Im więcej szumu w danych, tym większa powinna być wartość parametru.
- *min_samples* powinno być większe bądź równe wymiarowi danych.
- Dla danych 2-wymiarowych często dobrym wyborem jest $min\_samples = 4$.
- Dla danych o wymiarze większym niż 2, dobrym wyborem może być $min\_samples = 2 \cdot dim$, gdzie *dim* to wymiar danych.

Dobór parametrów

Dla wyboru *eps* istnieje następujaca metoda przeważnie dająca dobre wyniki:

- Dla każdego punktu w danych policz odległość do k-tego sąsiada, gdzie $k = min\_samples$.
- Narysuj obliczone posortowane odległości w kolejności rosnącej na wykresie k-odległości.
- Znajdź punkt zgięcia metodą łokcia (jak przy wyborze k w k-średnich).

Porównanie i ocena algorytmów grupowania

Porównanie i ocena algorytmów grupowania

Ocena jakości klasteryzacji i porównanie algorytmów jest bardzo trudne.

- **Ocenianie grupowania z prawdą podstawową:** *tabela krzyżowa*, *skorygowany indeks rand (ARI)* i *miara V (V-measure)*. Zwykle pomagają tylko w opracowaniu algorytmów, a nie w ocenie poprawności działania. Dla danych z prawdą podstawową można użyć informacji do zbudowania modelu nadzorowanego, jak klasyfikator.
- **Ocenianie grupowania bez prawdy podstawowej:** *współczynnik sylwetki* ocenia zwartość klastrów (im współczynnik bliższy 1 tym klastry bardziej zwarte), która nie zawsze jest pożądana.

Indeks rand (RI)

RI (Rand Index) to miara podobieństwa dwóch grupowań. Dla zbioru danych $X = \{X_1, \ldots, X_n\}$ mamy dwa grupowania: $G = \{G_1, \ldots, G_r\}$ z r klastrami i $P = \{P_1, \ldots, P_s\}$ z s klastrami. Zdefiniujmy:

- a - liczba par elementów z X będących w **tym samym** klastrze w G i w **tym samym** klastrze w P.
- b - liczba par elementów z X będących w **różnych** klastrach w G i w **różnych** klastrach w P.
- c - liczba par elementów z X będących w **tym samym** klastrze w G i w **różnych** klastrach w P.
- d - liczba par elementów z X będących w **różnych** klastrach w G i w **tym samym** klastrze w P.

$$RI = \frac{a+b}{a+b+c+d} = \frac{a+b}{\binom{n}{2}} = \frac{TP+TN}{TP+FP+FN+TN}$$

Skorygowany indeks rand (ARI)

Problem z *RI*: losowe przypisanie do klastrów może mieć różną wartość indeksu, nawet bliską 1. Skorygowany indeks rand (ARI) dla losowego przypisania jest skorygowany tak żeby zwracać wartość bliską 0:

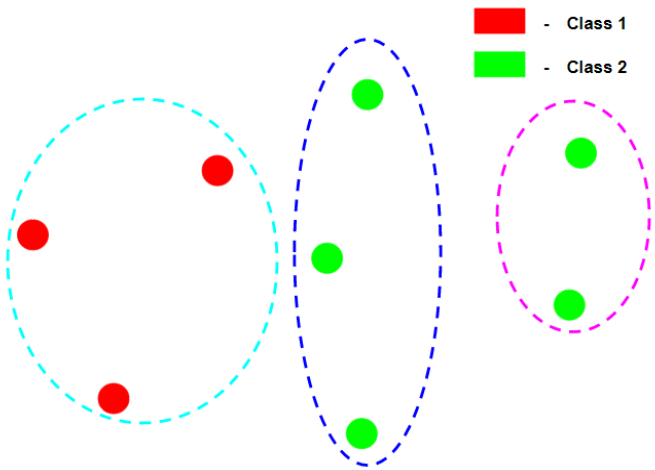
$$ARI = \frac{Index - ExpectedIndex}{MaxIndex - ExpectedIndex}$$

- *ARI* jest bliski 0 dla losowego przypisania do klas, a równy 1 dla identycznych grupowań (z dokładnością do permutacji).
- $ARI \geq -0.5$

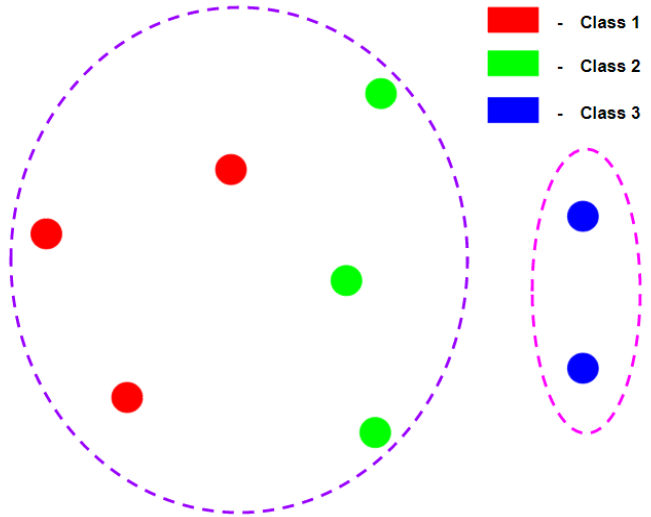
Miara V (V-measure)

Miara V mierzy podobieństwo grupowania do prawdy.

- homogeniczność h - perfekcyjnie homogeniczne grupowanie to takie, w którym każdy klaster zawiera obserwacje z tej samej prawdziwej grupy.
- kompletność c - perfekcyjnie kompletne grupowanie to takie, w którym wszystkie obserwacje zawarte w tej samej prawdziwej grupie znajdują się w tym samym klastrze.



h = 1



c = 1

Miara V (V-measure)

Miara V to średnia harmoniczna z homogeniczności i kompletności:

$$V = \frac{(1+\beta)hc}{\beta h + c}$$

`https://www.geeksforgeeks.org/ml-v-measure-for-evaluating-clustering-performance/`

- Miara V jest bliska 0 dla losowego przypisania do klas, a równa 1 dla identycznych grupowań (z dokładnością do permutacji).
- Miara V jest symetryczna, można ją traktować jako miarę podobieństwa dwóch grupowań.

`https://towardsdatascience.com/v-measure-an-homogeneous-and-complete-clustering-ab5b1823d0ad`

Grupowanie danych mieszanych

Grupowanie danych mieszanych (ciągłych i kategorialnych)

- Zakodowanie zmiennych jakościowych jako zmienne binarne
- Odległość Gowera
- Algorytm k-prototypów
- FAMD - factor analysis of mixed data

`https://medium.com/analytics-vidhya/the-ultimate-guide-for-clustering-mixed-data-1eefa0b4743b`

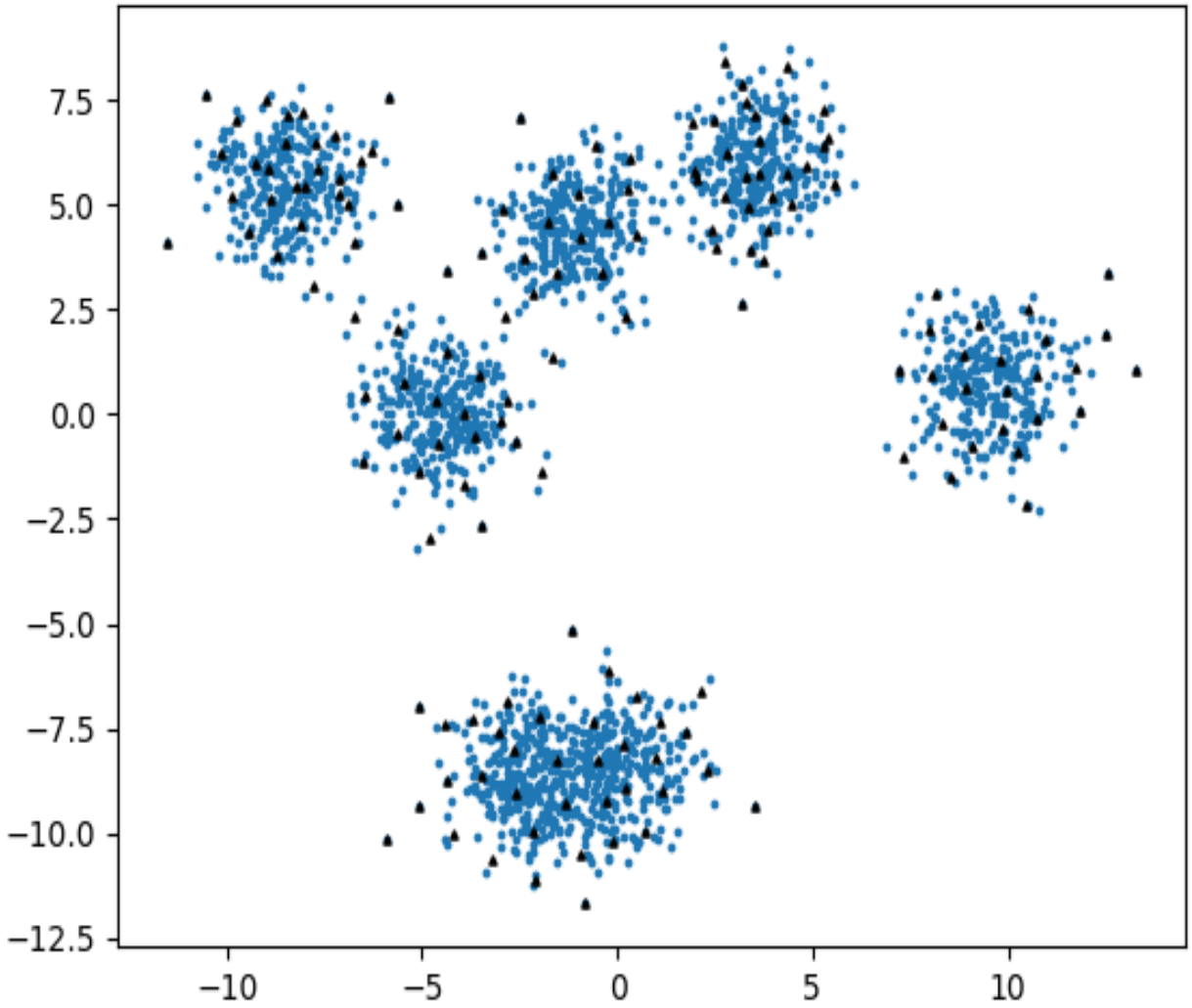
Algorytm BIRCH

Algorytm BIRCH

BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) - algorytm do klastrowania dużych zbiorów danych.

- W pierwszym kroku algorytm robi wstępne grupowanie obserwacji w małe, jednorodne klastry. Zachowuje przy tym tyle informacji o rozkładzie ile się da.
- W drugim kroku znalezione klastry grupuje dalej dowolnym algorytmem (np. k-średnich), gdzie obserwacjami są klastry znalezione w pierwszym kroku (ich środki).
- Algorytm potrzebuje tylko jednego przejścia po obserwacjach żeby znaleźć skupienia.

Algorytm BIRCH



CF - clustering feature

CF podsumowuje informacje o rozkładzie danych w klastrze tak, żeby zajmować mało pamięci, dobrze odtwarzać rozkład danych i łatwo się agregować dla sumy grup:

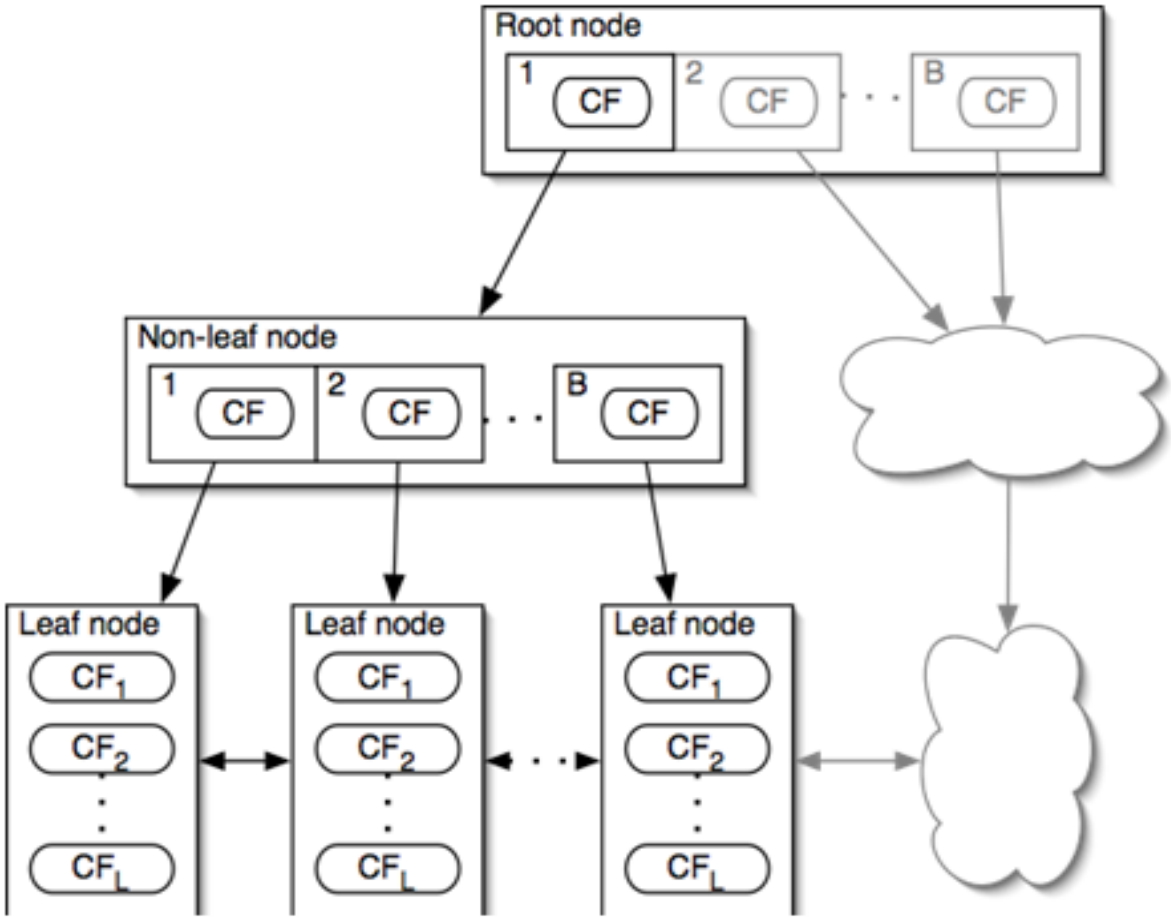
$$CF = (N, LS, SS) = (N, \sum_{i=1}^{N} X_i, \sum_{i=1}^{N} X_i^2),$$

gdzie N to liczba obserwacji w klastrze.
Twierdzenie o addytywności. Dla dwóch rozłącznych grup mamy:

$$CF_1 + CF_2 = (N_1 + N_2, LS_1 + LS_2, SS_1 + SS_2)$$

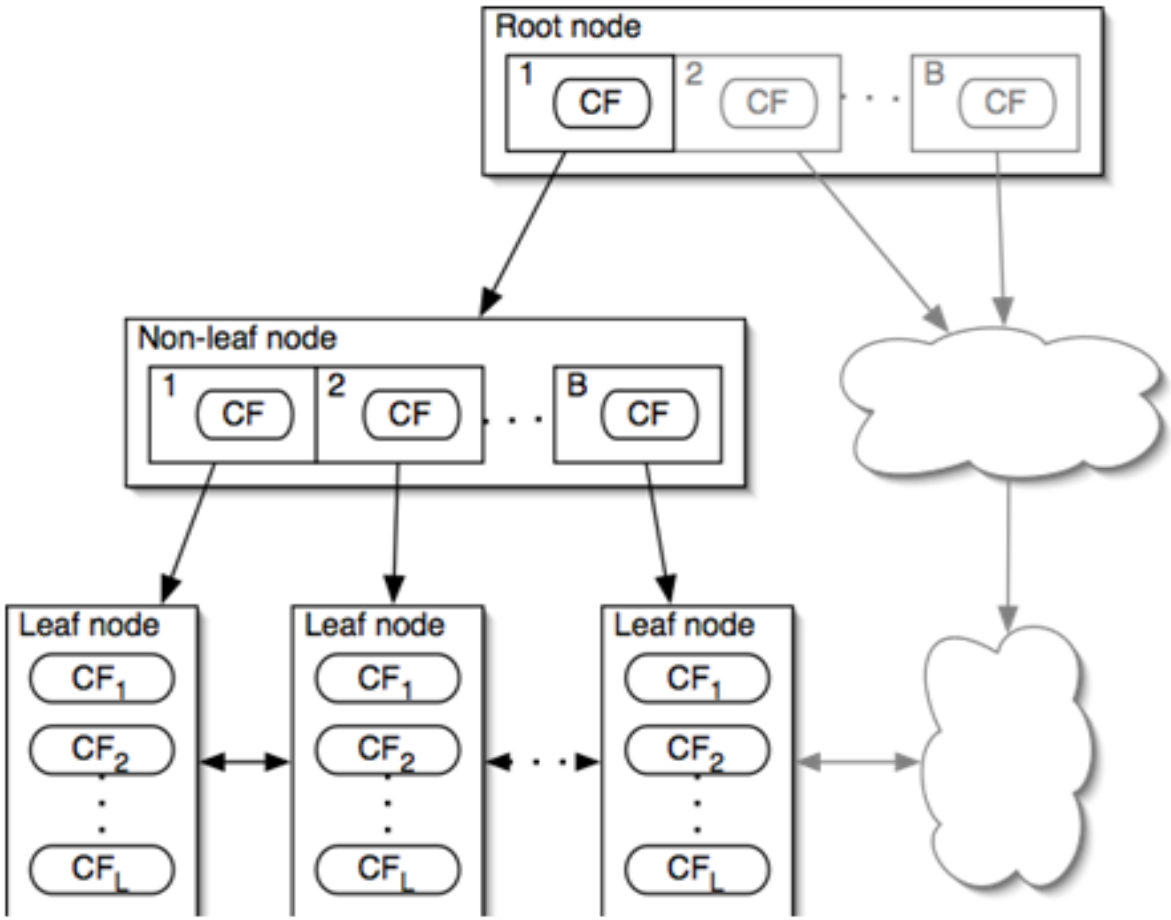
CF-tree zwarta reprezentacja zbioru danych

- Każdy element w liściu odpowiada klastrowi (CF-owi).
- Każdy liść zawiera co najwyżej L elementów będących CF-ami.



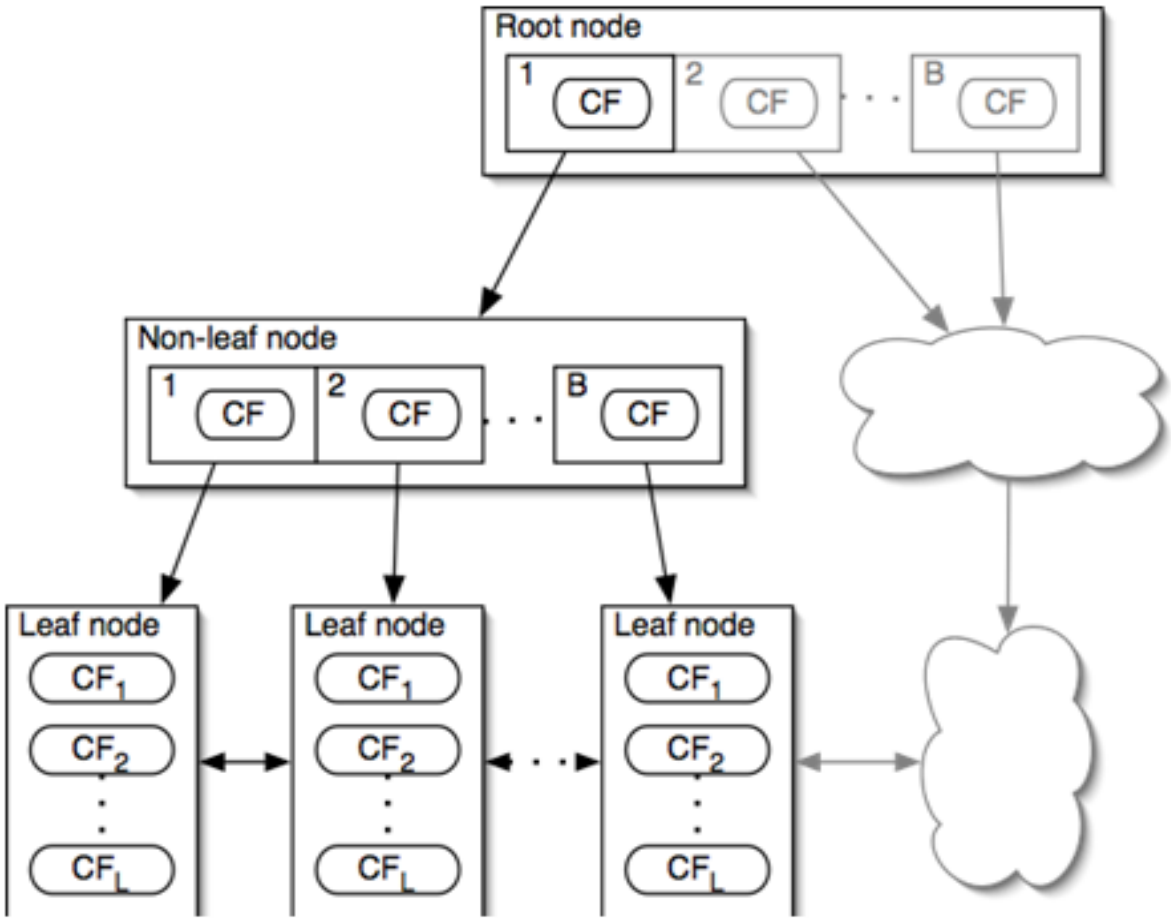
CF-tree

- Wszystkie elementy w liściu muszą spełniać warunek progowy: średnica klastra musi być mniejsza niż próg (threshold).
- Każdy liść ma 2 wskaźniki - prev i next, scalające wszystkie liście w celu efektywnego przeszukiwania.



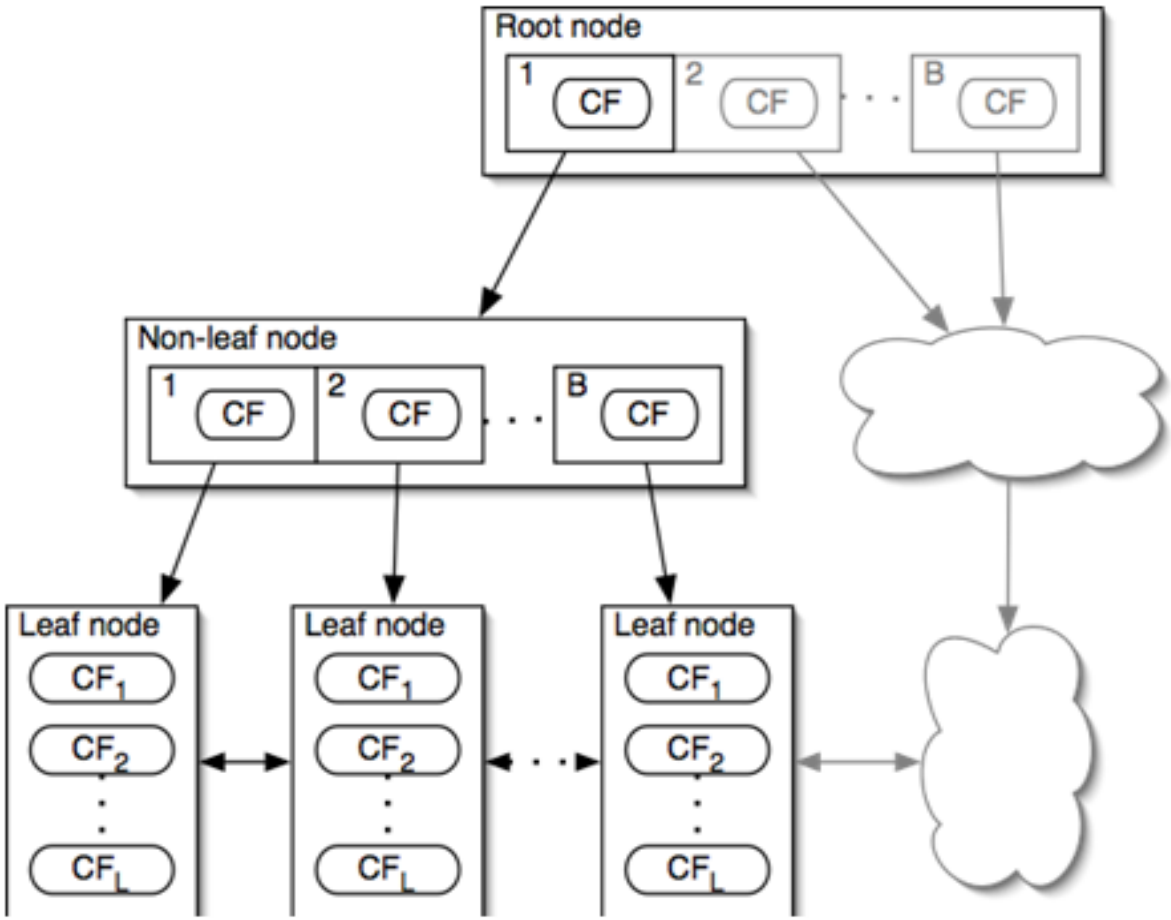
CF-tree

- Każdy węzeł nie będący liściem zawiera co najwyżej B (branching factor) elementów.
- Każdy z elementów zawiera wskaźnik do węzła potomnego oraz CF złożony z sumy CF-ów w węźle-dziecku.



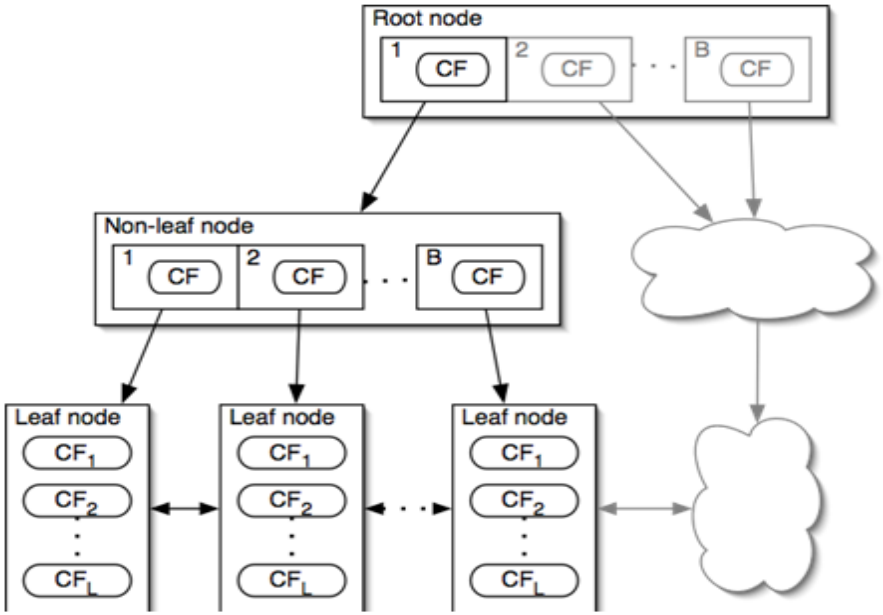
CF-tree, dodawanie punktu do drzewa

Dodawanie punktu (albo klastra) do drzewa:
1. Zidentyfikuj odpowiedni liść: zaczynając w korzeniu, rekurencyjnie schodź w dół wybierając najbliższe dziecko (w sensie wybranej metryki, u nas euklidesowej)



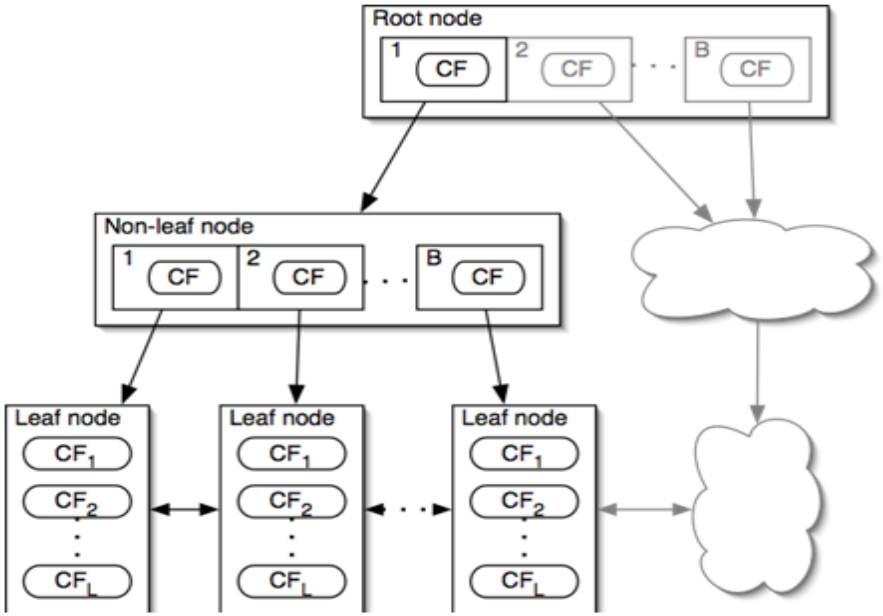
CF-tree, dodawanie punktu do drzewa

2. Zmodyfikuj liść: doszedłszy do liścia znajdź najbliższe CF i sprawdź, czy może wchłonąć nowy punkt (klaster) bez naruszenia warunku progowego. Jeżeli może, zaktualizuj CF. W przeciwnym przypadku dodaj nowe CF do liścia. Jeżeli w liściu nie ma już miejsca (liczba CF-ów $> L$), trzeba podzielić liść. Podział liścia polega na znalezieniu dwóch CF-ów położonych najdalej od siebie i rozdzielenie pozostałych CF-ów według odległości od wybranych dwóch.



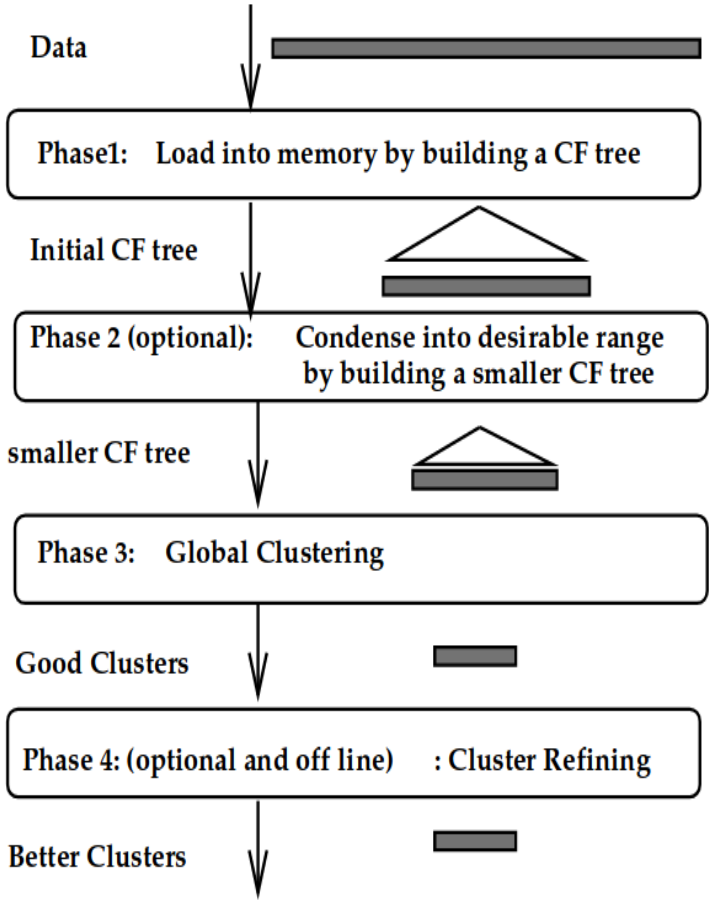
CF-tree, dodawanie punktu do drzewa

3. Zmodyfikuj ścieżkę do liścia. Każdy węzeł nie będący liściem jest CF-em złożonym z CF-ów dzieci. Po dodaniu punktu (klastra) do liścia trzeba zaktualizować informacje zawarte w węzłach znajdujących się po drodze do liścia. Jeżeli doszło do podziału liścia, musimy dodać nowy CF do węzła będącego rodzicem ze wskaźnikiem do nowo powstałego liścia. Jeżeli spowoduje to zwiększenie liczby CF-ów powyżej B, trzeba podzielić węzeł-rodzica, itd. aż do korzenia.



Kroki algorytmu BIRCH

Figure 2. BIRCH Overview



Źródło: https://www.coryjmaklin.com/2019-07-01_Machine-Learning--BIRCH-Clustering-Algorithm-Clearly-Explained-fb9838cbeed9/

BIRCH - wybór parametrów

- B - branching factor to parametr wpływający na szybkość algorytmu, im większy zbiór danych, tym jego wartość powinna być większa. Dobrym punktem wyjścia jest wartość domyślna równa 50.
- próg (threshold) wpływa na wielkość klastrów we wstępnym klastrowaniu, jego wartość powinna być jak najmniejsza żeby nie tracić niepotrzebnie informacji na temat rozkładu danych. Jednak na tyle duża żeby klastrowanie było możliwe do wykonania (ograniczenia pamięci i procesora).

Algorytm Mini-Batch K-Means

Algorytm k-średnich dla dużych zbiorów danych
Mini-Batch K-Means

- Algorytm k-średnich musi wczytać cały zbiór danych do pamięci, dlatego użycie go na dużych zbiorach często jest niemożliwe.
- Algorytm Mini-Batch K-Means: losowanie małych partii danych, żeby mieściły się w pamięci, i aktualizacja klastrów na ich podstawie.
- Dla kolejnych partii zdefiniowany jest parametr *learning_rate*, który z każdym krokiem (partią) maleje, powodując zmniejszający się wpływ kolejnych partii na klastrowanie i zbieżność algorytmu.
- Algorytm Mini-Batch K-Means jest istotnie szybszy od algorytmu K-Means za cenę niewielkich rozbieżności w wynikowych klastrach.

Algorithm 1 Mini-Batch K-Means

Wejście: zbiór danych X, wielkość partii (batch) b, liczba klastrów k, liczba iteracji J

1. Inicjalizacja środków klastrów: c_i, $i = 1, \ldots, k$ jak w algorytmie k-średnich

2. Inicjalizacja klastrów i ich liczności: $C_i = \emptyset$, $N_{C_i} = 0$, $i = 1, \ldots, k$

for $j = 1$ **to** J **do**

  wylosuj B (batch) podzbiór b elementów ze zbioru X

  for $x \in B$ **do**

    przyporządkuj x najbliższy klaster $C(x)$

  end for

  for $x \in B$ **do**

    znajdź środek klastra $c(x)$ odpowiadający x

    zaktualizuj liczbę elementów w klastrze $N(x) = N(x) + 1$

    oblicz *learning_rate* $\eta(x) = \frac{1}{N(x)}$

    zaktualizuj środek $c(x) = (1 - \eta(x))c(x) + \eta(x)x$

  end for

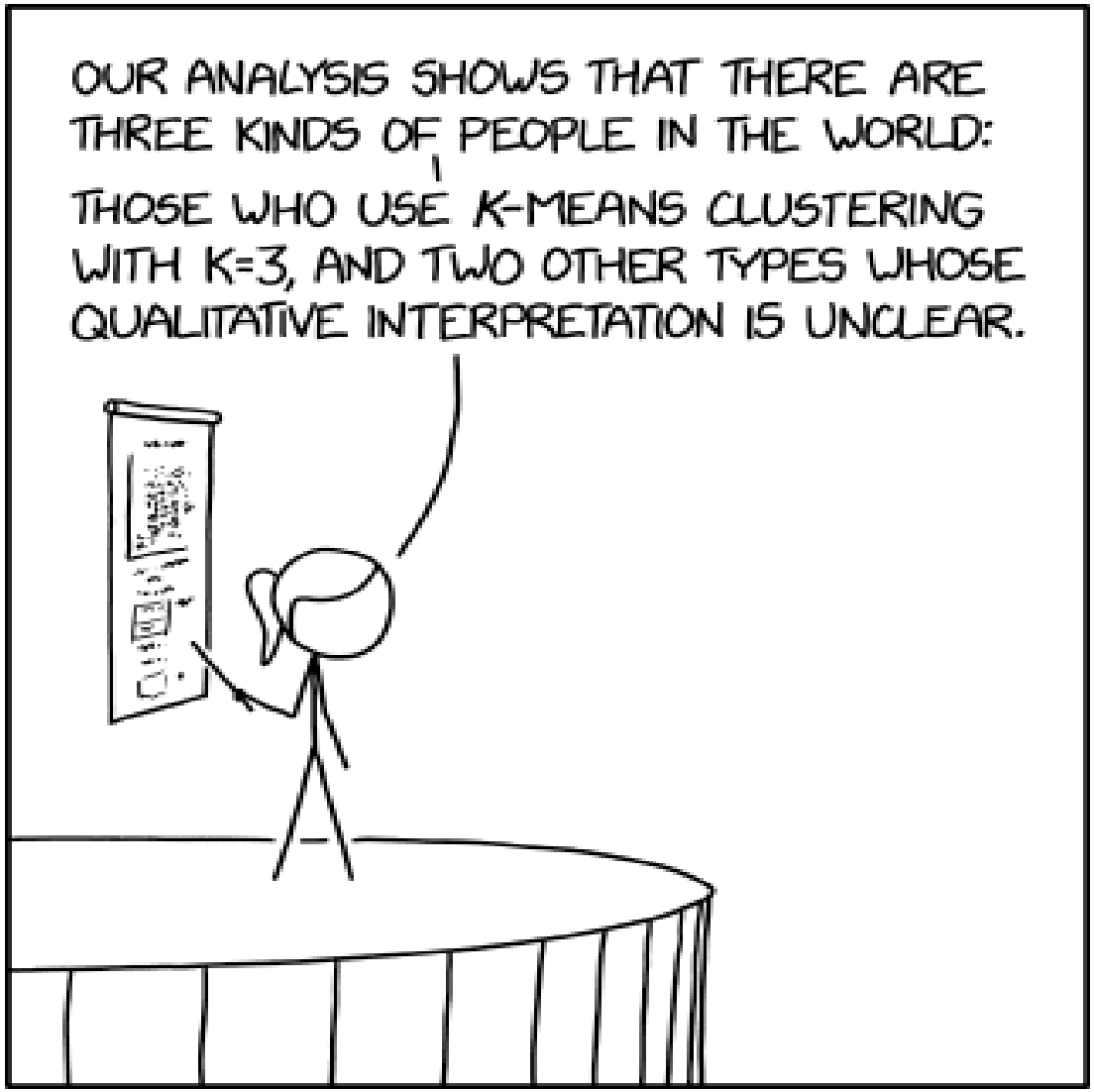
end for

Porównanie omówionych algorytmów

Porównanie omówionych algorytmów

- **Grupowanie k-średnich**:
  - Konkretna liczba klastrów
  - Kuliste kształty klastrów o takich samych średnicach
  - Klastry są reprezentowane przez średnie
- **Mini-Batch K-Means**: szybka wersja algorytmu k-średnich działająca na dużych zbiorach danych
- **Grupowanie hierarchiczne aglomerayjne**:
  - Cała hierarchia możliwych partycji danych,
  - Możliwość wizualizacji na dendrogramach
- **DBSCAN**:
  - Złożone kształty klastrów
  - Wykrywanie obserwacji odstających
  - Automatyczne określenie liczby klastrów
- **BIRCH**: dwukrokowa procedura działająca na dużych zbiorach danych

`https://scikit-learn.org/stable/modules/clustering.html`


OUR ANALYSIS SHOWS THAT THERE ARE THREE KINDS OF PEOPLE IN THE WORLD:
1
THOSE WHO USE K-MEANS CLUSTERING WITH K=3, AND TWO OTHER TYPES WHOSE QUALITATIVE INTERPRETATION IS UNCLEAR.