sages

Analiza skupień

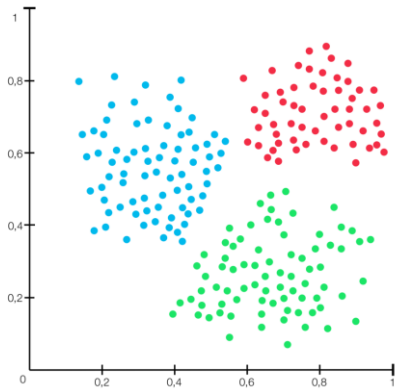
Bogdan Gliwa, PhD

Analiza skupień/klasteryzacja

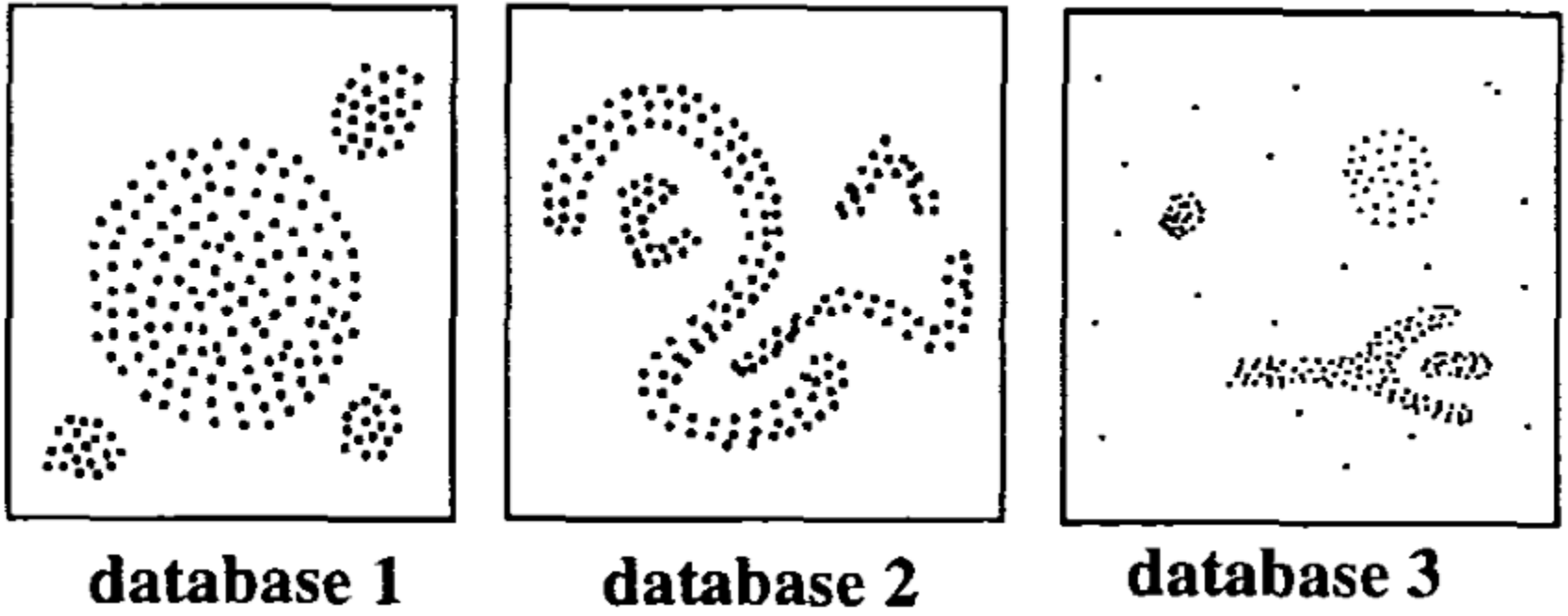
- Przykład uczenia nienadzorowanego (unsupervised learning)
  - znamy jakieś inne przykłady?

- Brak informacji o klasach w zbiorze

- Cel – pogrupowanie obiektów zbioru w klastry/grupy tak, aby wewnątrz jednej grupy obiekty były jak najbardziej do siebie podobne, a pomiędzy różnymi grupami – jak najmniej



Klaster



http://www.sthda.com/sthda/RDoc/images/dbscan-idea.png

Niejednoznaczność grupowania

How many clusters?

Niejednoznaczność grupowania

How many clusters?

Six Clusters

Two Clusters

Four Clusters

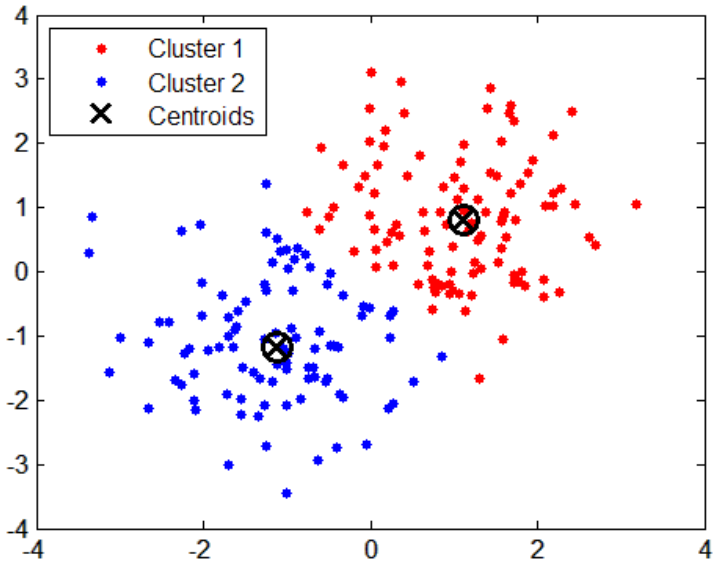
Zastosowania klasteryzacji

- Segmentacja rynku
- Systemy rekomendacyjne
- Detekcja anomalii
- Grupowanie tekstów pod kątem tematu
- Grupowanie obrazów pod kątem treści

K-means

- Podstawowa wersja działa tylko z danymi numerycznymi
- Musimy z góry podać liczbę klastrów (parametr k)
- Środek klastra jest nazywany centroidem

$$m = \frac{1}{C}\sum_{i=0}^{C} x_i$$



K-means

- Schemat działania
  1. wybierz losowo K punktów jako środki klastrów (centroidy)
  2. przyporządkuj każdy punkt do najbliższego centroidu
  3. przelicz centroidy na podstawie punktów, które należą do klastra danego centroida
  4. powtarzaj punkty 2-3 aż do uzyskania zbieżności

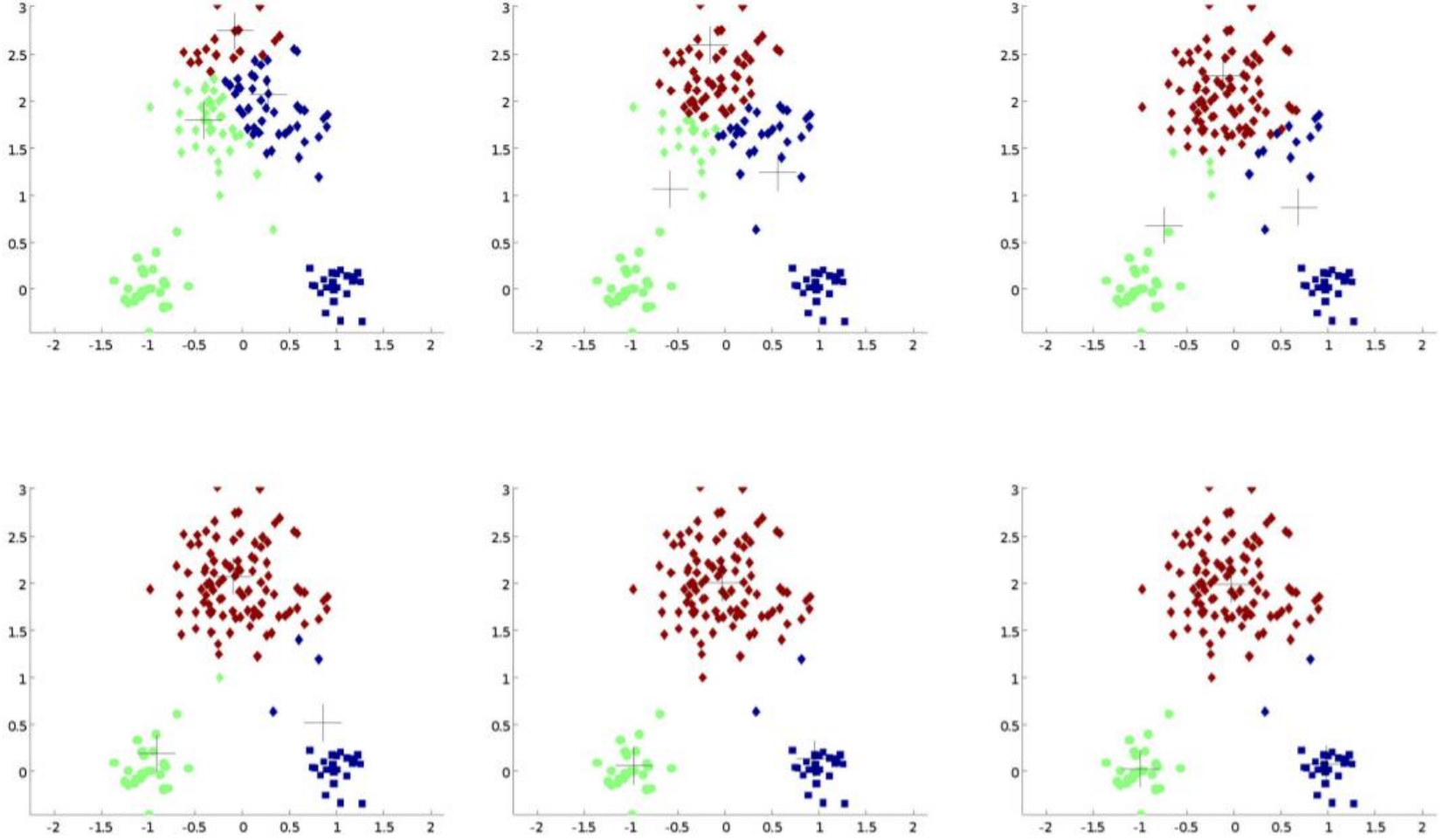
- Inercja (inertia) – funkcja, która jest optymalizowana

$$\sum_{i=0}^{n} min_{m_j \in C} \left(\|x_i - m_j\|^2 \right)$$

m_j – centroid

x_i – punkt danych

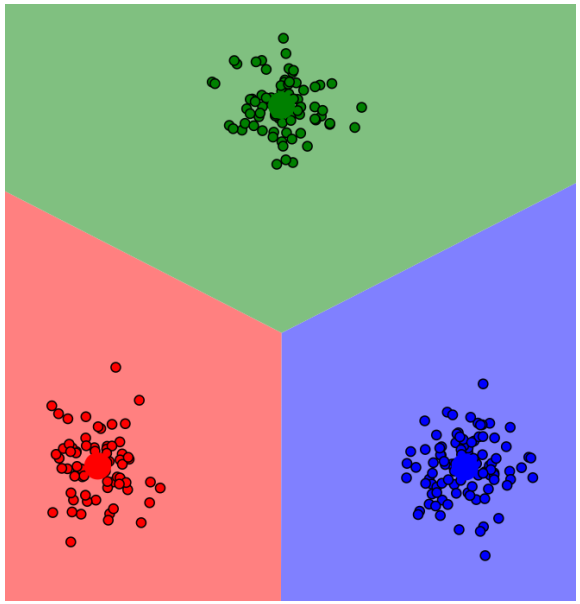
K-means



K-means

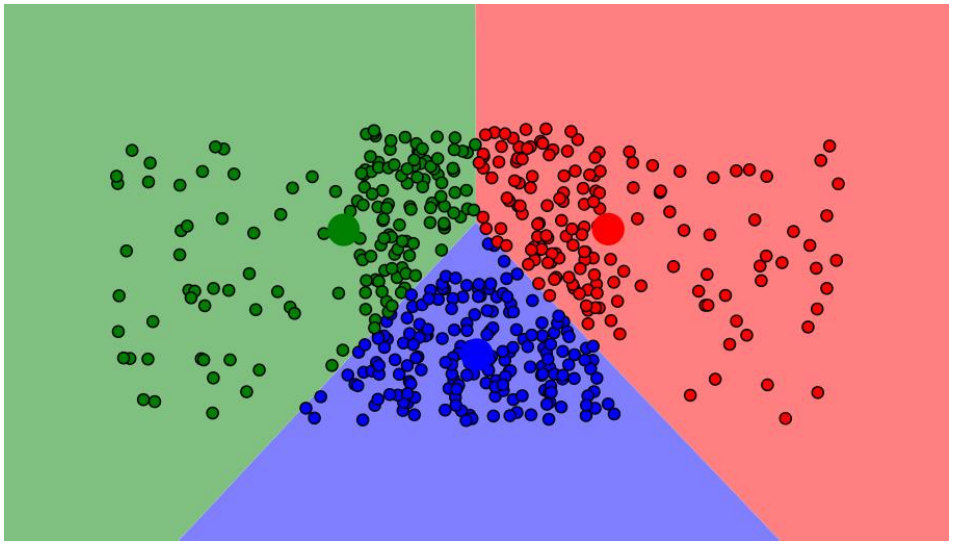
wizualizacja

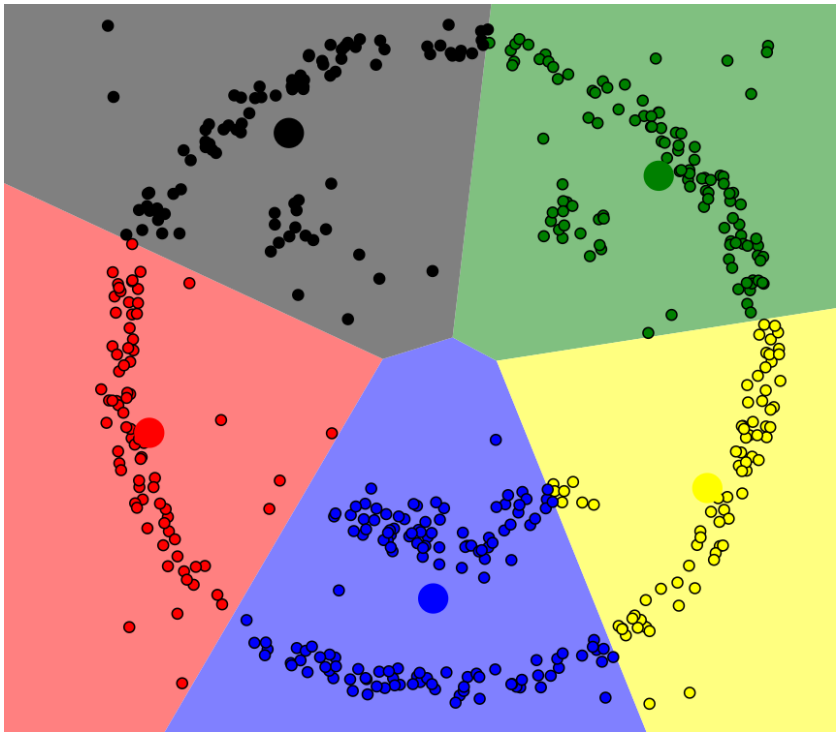
https://www.naftaliharris.com/blog/visualizing-k-means-clustering/



K-means

przykłady kiedy k-means nie sprawdza się





K-means

Kiedy sprawdza się algorytm?

- dane „wyspowe”, wyraźnie oddzielone klastry
- klastry mają jednorodną formę
- zbliżona liczność klastrów

K-means

- Rezultaty są zależne od początkowej inicjalizacji
  - często algorytm uruchamia się kilka razy i wybiera najlepszy wynik
- Trzeba wybrać z góry liczbę klastrów
- Każdy obiekt jest przyporządkowany do któregoś z klastrów
- K-means jest czułe na elementy odstające (outliers)

Jak wybrać parametr K?

- ewaluacja zewnętrzna (extrinsic)
  - gdy mamy w zbiorze informację o klasach
- ewaluacja wewnętrzna (intrinsic)
  - na podstawie wewnętrznych zależności w danych

Ocena jakości grupowania

- gdy mamy klastry ground-truth, klasy
  - Homogeneity
  - Completeness
  - V-measure

- gdy nie mamy informacji o klastrach ground-truth
  - Silhouette coefficient
  - Davies-Bouldin index

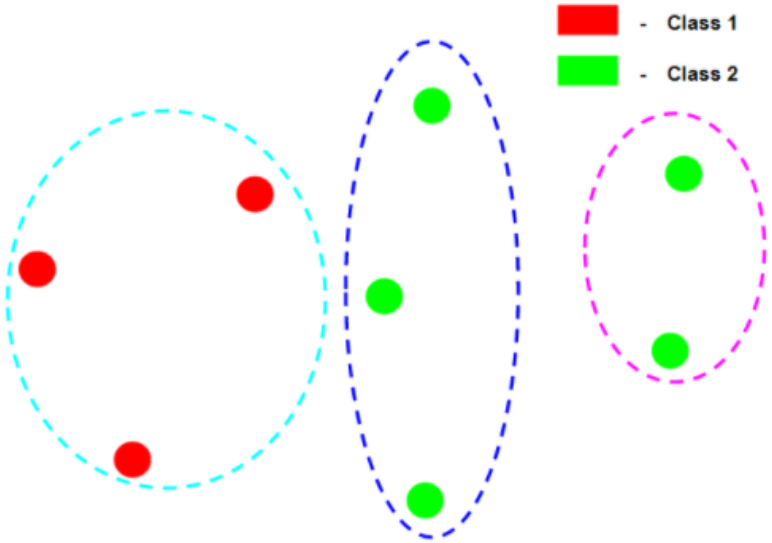
V-measure

- używana kiedy porównujemy się z klasami ground-truth
- homogenity (homogeniczność) – na ile każdy klaster zawiera punkty z tylko jednej klasy
- completeness (zupełność) – na ile wszystkie punkty z danej klasy są przyporządkowane do jednego klastra

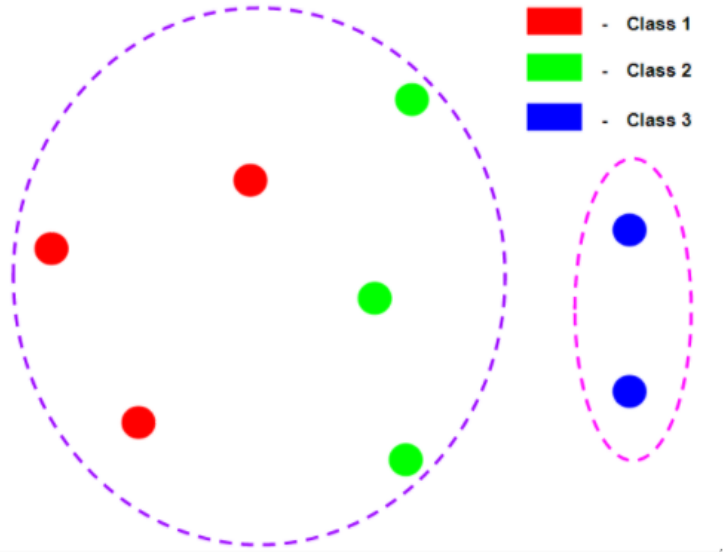
$$v = \frac{(1+\beta) \times homogenity \times completeness}{\beta \times homogenity + completeness}$$

V-measure

pełna homogeniczność,
ale nie zupełność



pełna zupełność,
ale nie homogeniczność

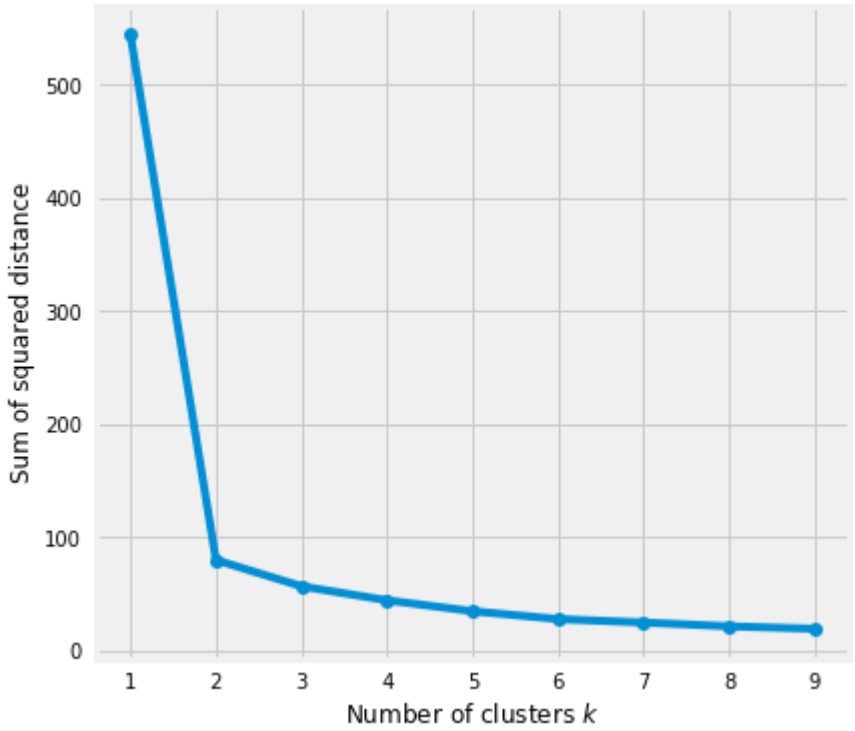


https://www.geeksforgeeks.org/ml-v-measure-for-evaluating-clustering-performance/

Dobieranie parametru k (ewaluacja wewnętrzna)

- metoda Elbow
- analiza Silhouette

Metoda łokcia (elbow method)



- Dobieramy parametr tak, aby krzywa SSE zaczęła się wypłaszczać

Analiza Silhouette

- Silhouette coefficient dla punktu x

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

a(x) – średnia odległość między x i wszystkimi innymi punktami w klastrze, do którego należy x

b(x) – minimum ze średnich odległości między x i punktami z pozostałych klastrów, do których nie należy x

Analiza Silhouette

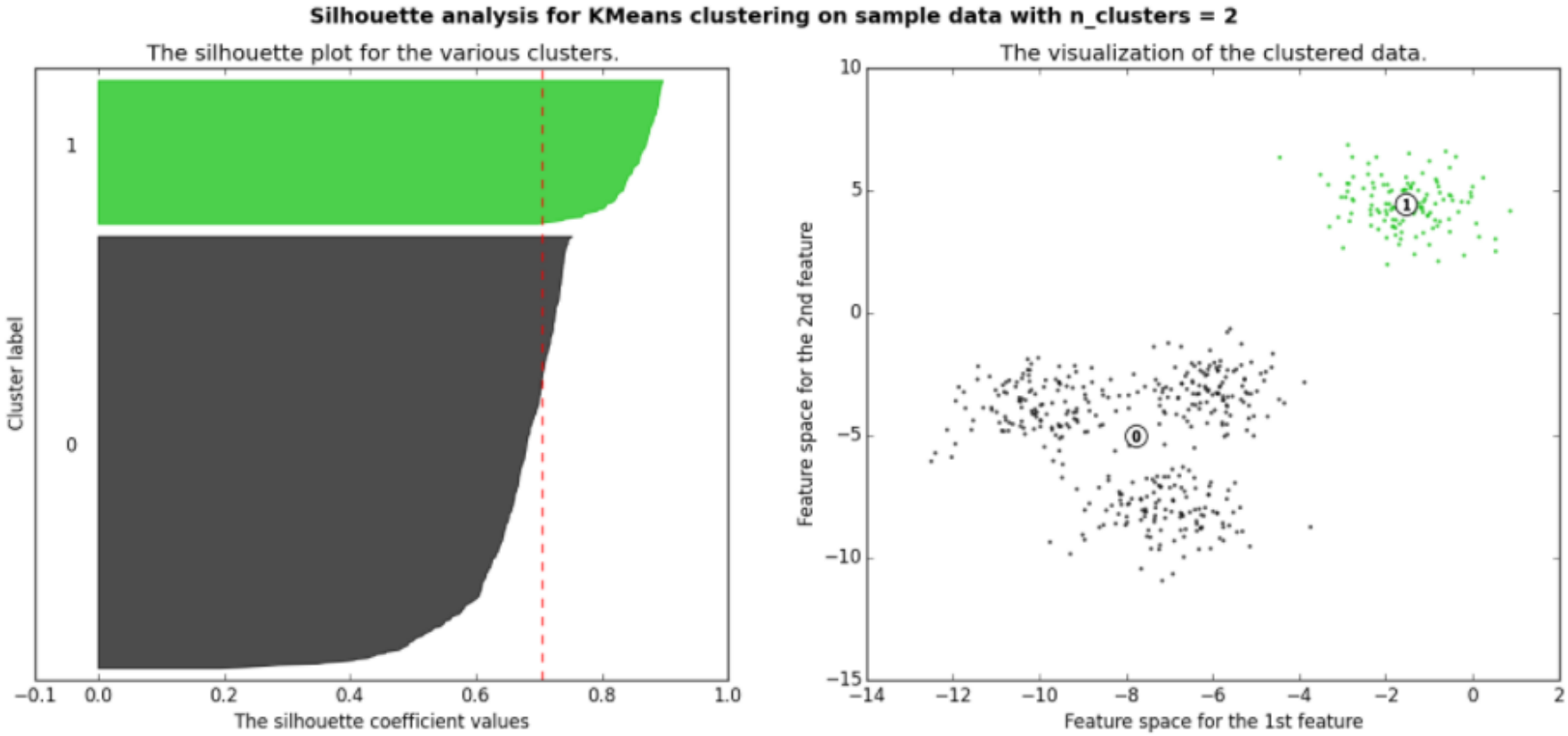
$$s(x) \in [-1,1]$$

s(x) = 1 oznacza, że punkt x jest blisko pozostałych punktów w swoim klastrze i daleko od innych klastrów

s(x) = 0 oznacza, że punkt nie jest mocno związany ze swoim klastrem i jest tak samo oddalony od innych, ew gdy są zachodzące na siebie klastry

s(x) = -1 oznacza, że punkt powinien należeć do innego klastra

Analiza Silhouette



Analiza Silhouette

Jak wybrać liczbę klastrów?

- średnia wartość współczynnika Silhouette powinna być jak najbliższa 1
- wykres każdego klastra powinien być powyżej średniej wartości w jak największym stopniu
- szerokość wykresu dla klastra powinna być jak najbardziej jednorodna

Davies-Bouldin index

- w klasteryzacji chcemy, aby odległości punktów w klastrze w stosunku do jego środka były jak najmniejsze, natomiast odległości pomiędzy klastrami jak największe

$$DBI = \frac{1}{k}\sum_{i=1}^{k} max_{j \neq i}\left[\frac{S_i + S_j}{M_{ij}}\right]$$

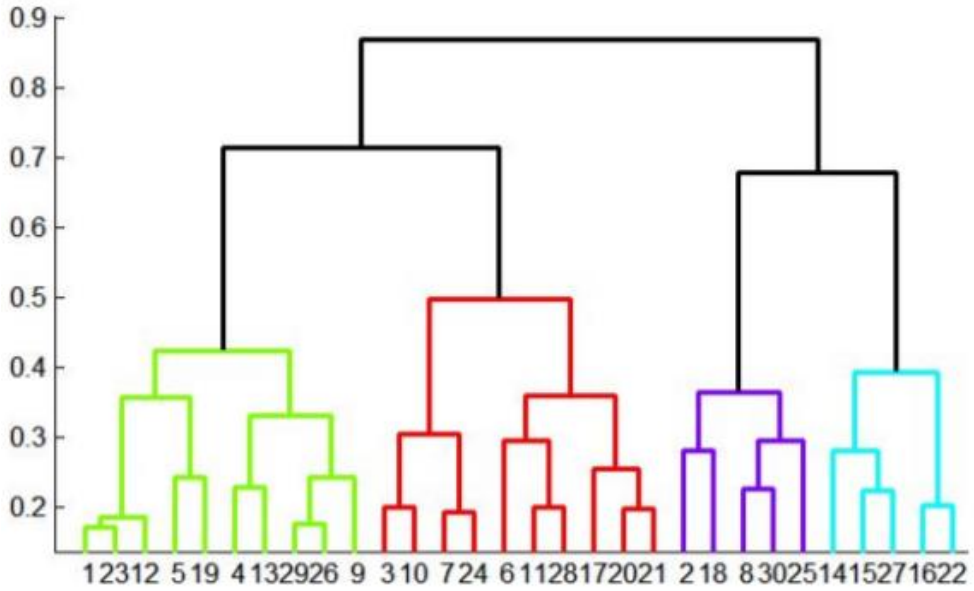
S_i oznacza średnią odległość punktów w klastrze *i* od jego środka

M_{ij} oznacza odległość między środkami klastra *i* oraz *j*

Im niższe DBI tym lepszy podział na klastry.

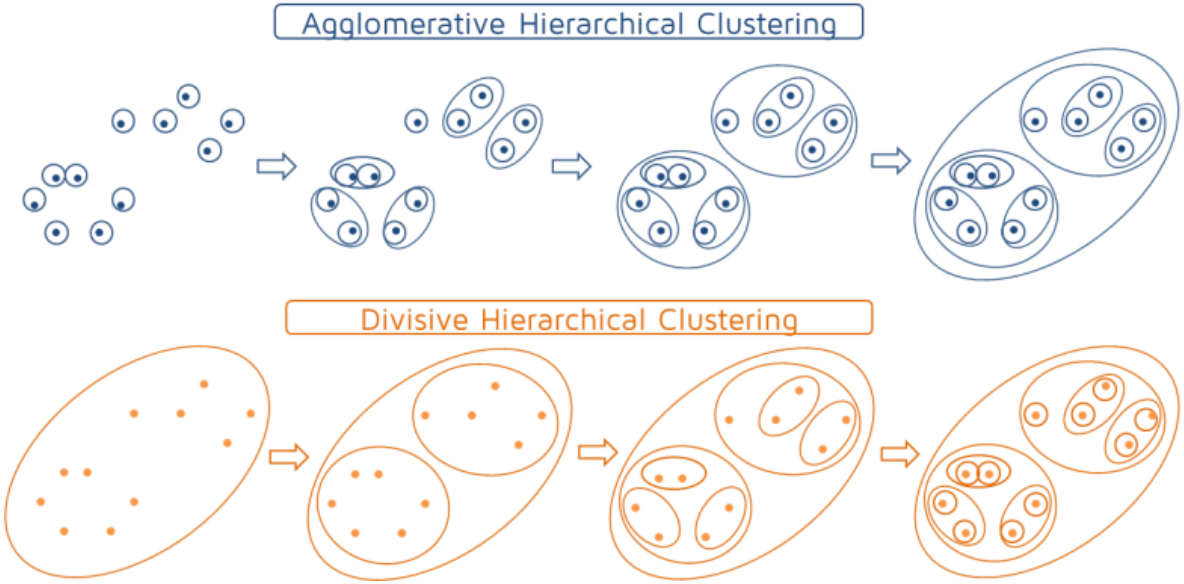
Grupowanie hierarchiczne

- Odkrywa zagnieżdżone klastry, które mogą być zwizualizowane za pomocą dendrogramu.



Grupowanie hierarchiczne

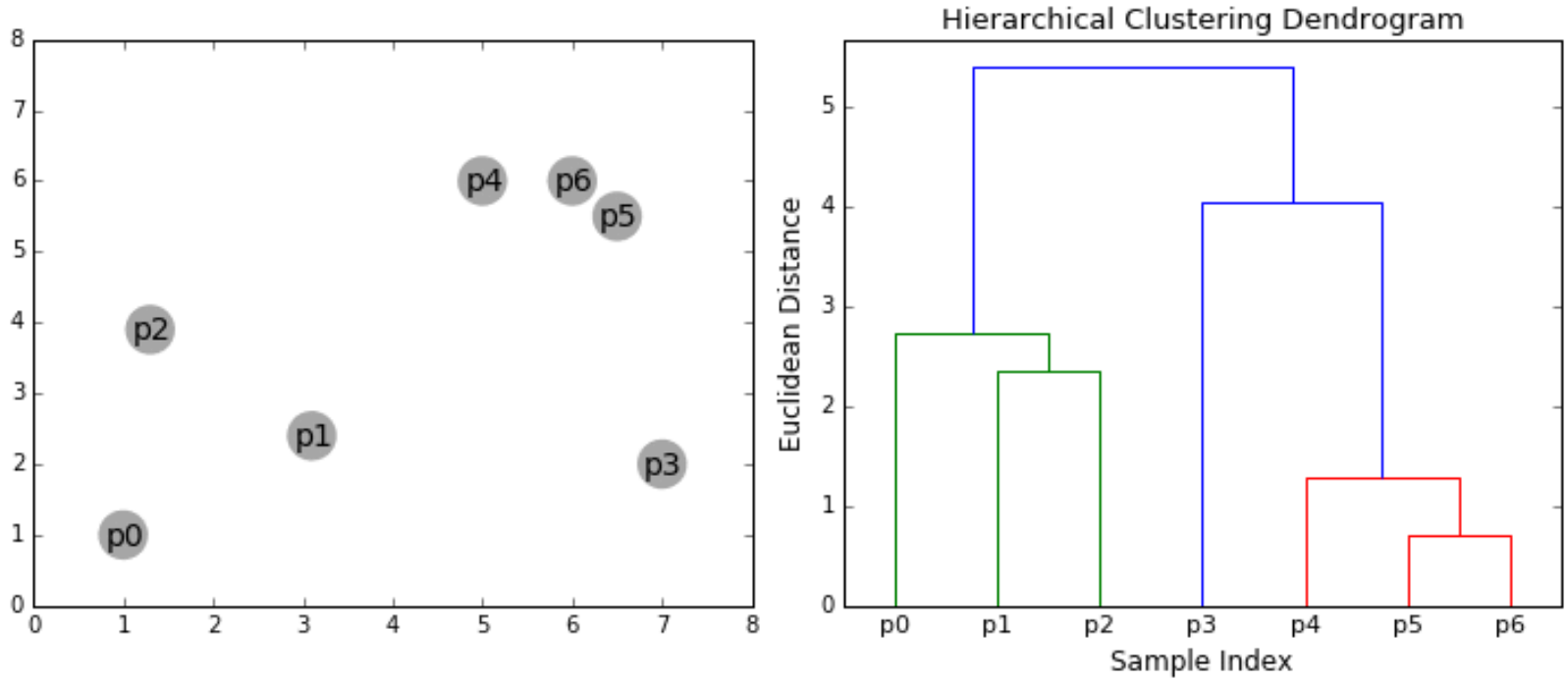
- 2 typy
  - ❑ aglomeratywne (agglomerative, bottom up)
    - na początku każdy element jest osobnym klastrem
    - iteracyjnie, łączymy najbliższe klastry w większe
  - ❑ podziałowe (divisive, top down)
    - zaczynamy od jednego dużego klastra i rekursywnie dzielimy go na mniejsze



Aglomeratywne grupowanie hierarchiczne

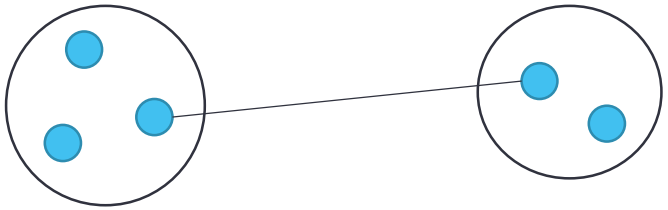
- Kroki algorytmu:
  - policz metrykę odległości dla wszystkich klastrów i pomiędzy nimi
  - połącz najbardziej podobne klastry w jeden większy klaster
  - powtarzaj 2 poprzednie kroki aż wszystkie obiekty są w jednym klastrze

Aglomeratywne grupowanie hierarchiczne

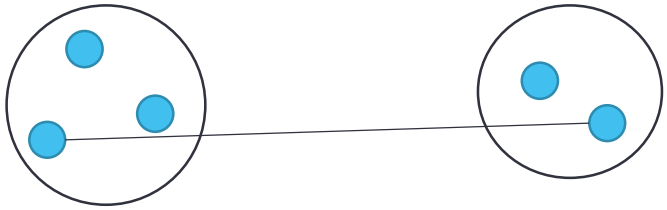


Strategie łączenia klastrów w aglomeratywnym grupowaniu hierarchicznym

- Single linkage – najmniejsza odległość między obiektami z 2 klastrów

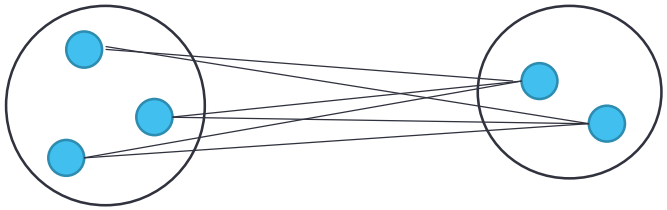


- Complete linkage – największa odległość między obiektami z 2 klastrów

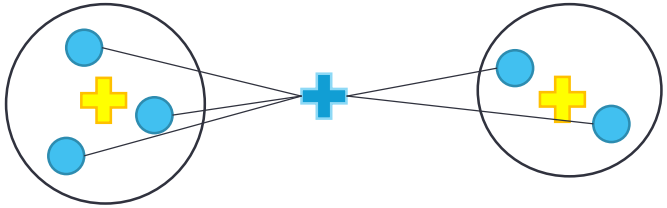


Strategie łączenia klastrów w aglomeratywnym grupowaniu hierarchicznym

- Average linkage – średnia z wszystkich odległości między obiektami w 2 klastrach



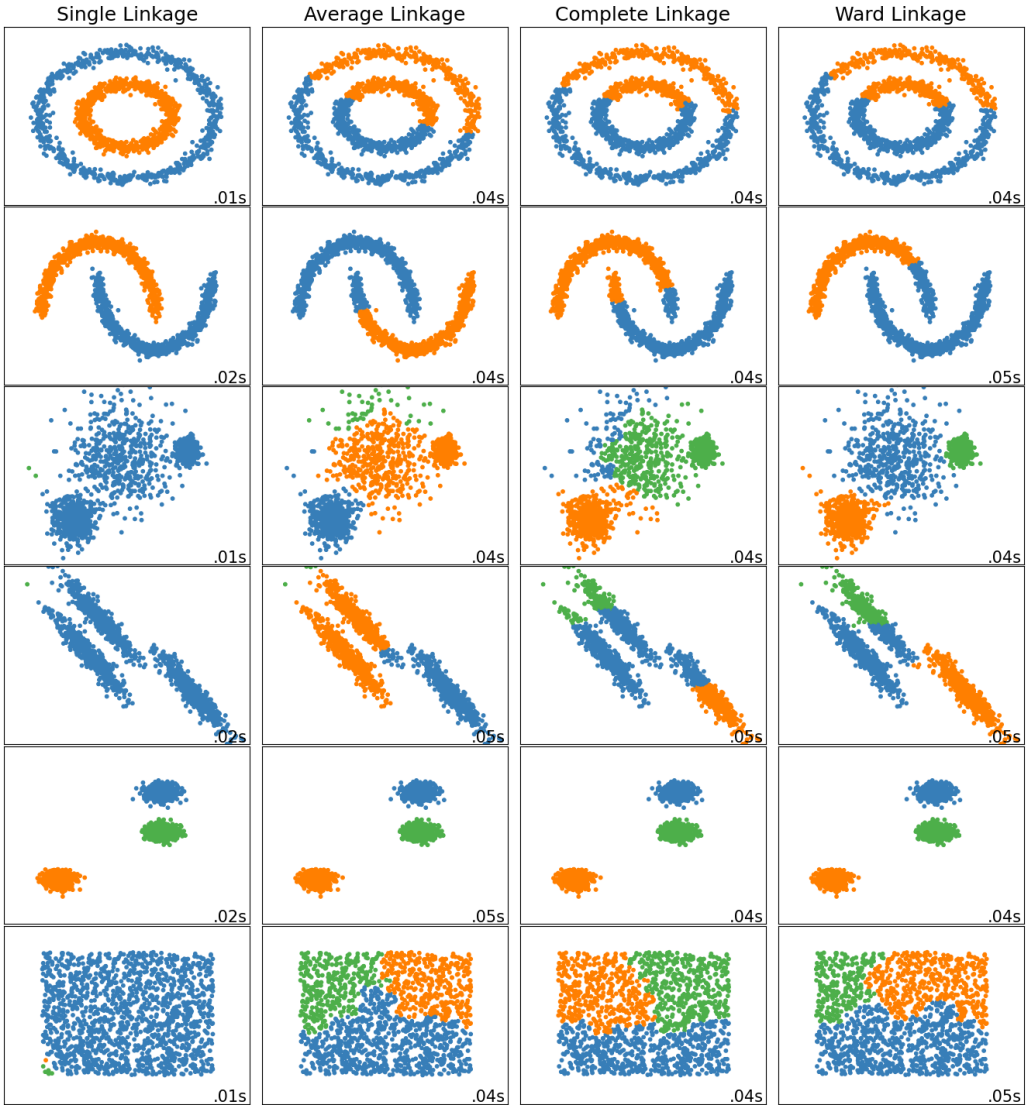
- Ward linkage (minimum variance) – najmniejszy wzrost w całkowitej wariancji wokół centroidów klastrów



$$d(i,j) = SS_{i \cup j} - (SS_i + SS_j)$$

$$SS(C) = \sum_{x \in C} (x - \mu_C)^2$$

Grupowanie hierarchiczne



Grupowanie hierarchiczne

- Nie zakłada z góry określonej liczby klastrów – pożądana liczba klastrów może być uzyskana przez odcięcie dendrogramu na odpowiednim poziomie

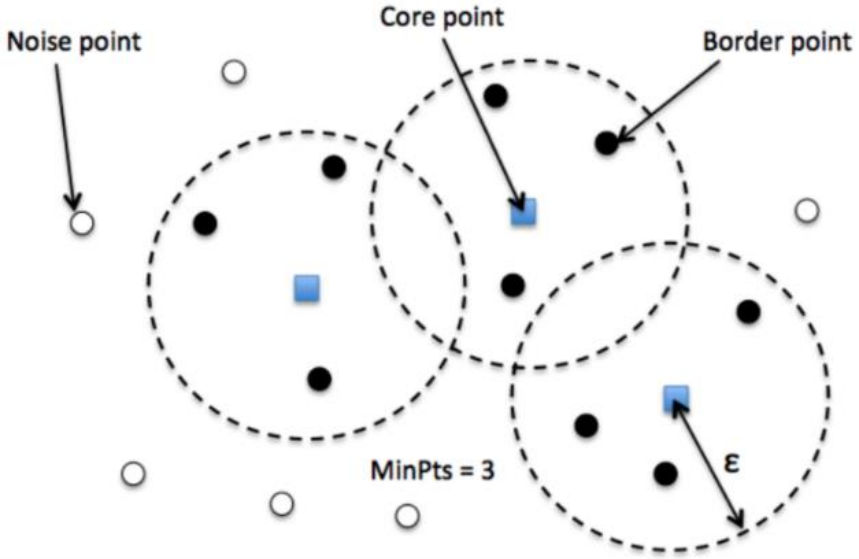
- Hierarchiczność klastrów może mieć odzwierciedlenie w taksonomii

DBSCAN

- **D**ensity-**b**ased **S**patial **C**lustering of **A**pplications with **N**oise
- Algorytm dla gęstościowej klasteryzacji (grupa składa się z sąsiednich punktów o wysokiej gęstości w otoczeniu)
- Nie wymaga określenia liczby grup.
- Pozwala znaleźć obserwacje odstające.
- Posiada 2 parametry:
  - minPts – minimalna liczba obiektów w otoczeniu
  - eps/epsilon – rozważane otoczenie obiektu

DBSCAN

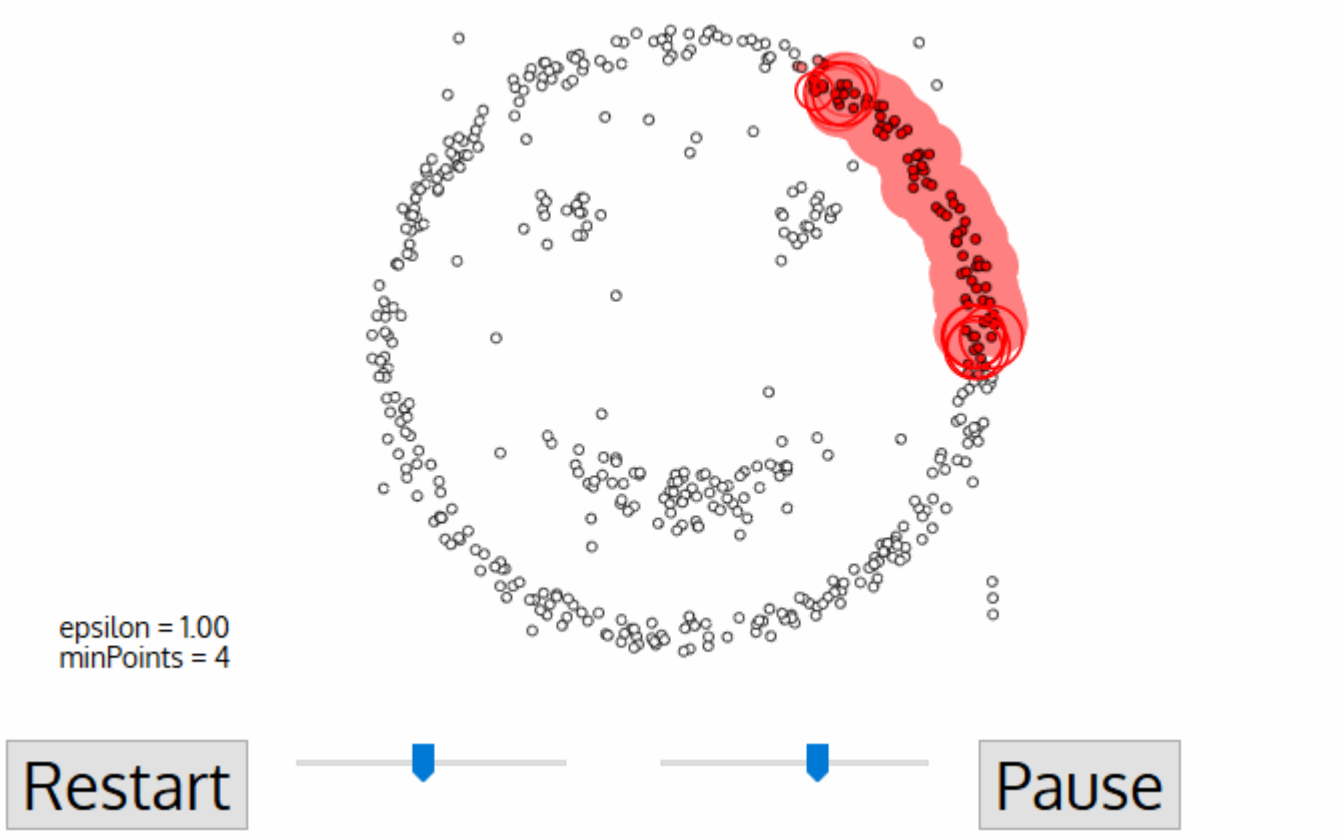
- 3 typy obiektów:
  - core – obiekty, które mają co najmniej minPts w promieniu eps (wnętrze klastra)
  - border – mają mniej niż minPts w promieniu eps, ale są w otoczeniu jakiegoś obiektu core
  - outlier – pozostałe obiekty



DBSCAN

- Algorytm:
  - znajdź punkty w promieniu eps dla każdego punktu i oznacz jako core points te, które mają co najmniej minPts sąsiadów
  - znajdź połączone komponenty dla core points
  - pozostałe punkty przypisz do najbliższego klastra jeśli klaster jest w sąsiedztwie eps, w przeciwnym wypadku przypisz jako szum (noise, outliers)

DBSCAN



https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

DBSCAN – dobór parametrów

- minPts
  - min 3 (2 -> klastrowanie hierarchiczne z metryką single linkage i dendrogramem odciętym na poziomie epsilon)
  - jeśli zbiór zawiera dużo szumu warto wybrać większą wartość tego parametru
  - dla dużych danych często wybiera się większe wartości
  - często używa się reguły minPts = 2 * dim

DBSCAN – dobór parametrów

- epsilon
  - empirycznie
  - korzystając z wykresu średnich odległości między każdym punktem danych a jego k najbliższymi sąsiadami (k=minPts)
    - ❑średnie k-odległości są rysowane rosnąco (k-distance graph)
    - ❑optymalna wartość epsilon jest na punkcie maksymalnej krzywizny

https://www.jstatsoft.org/article/view/v091i01

DBSCAN – dobór parametrów

- epsilon

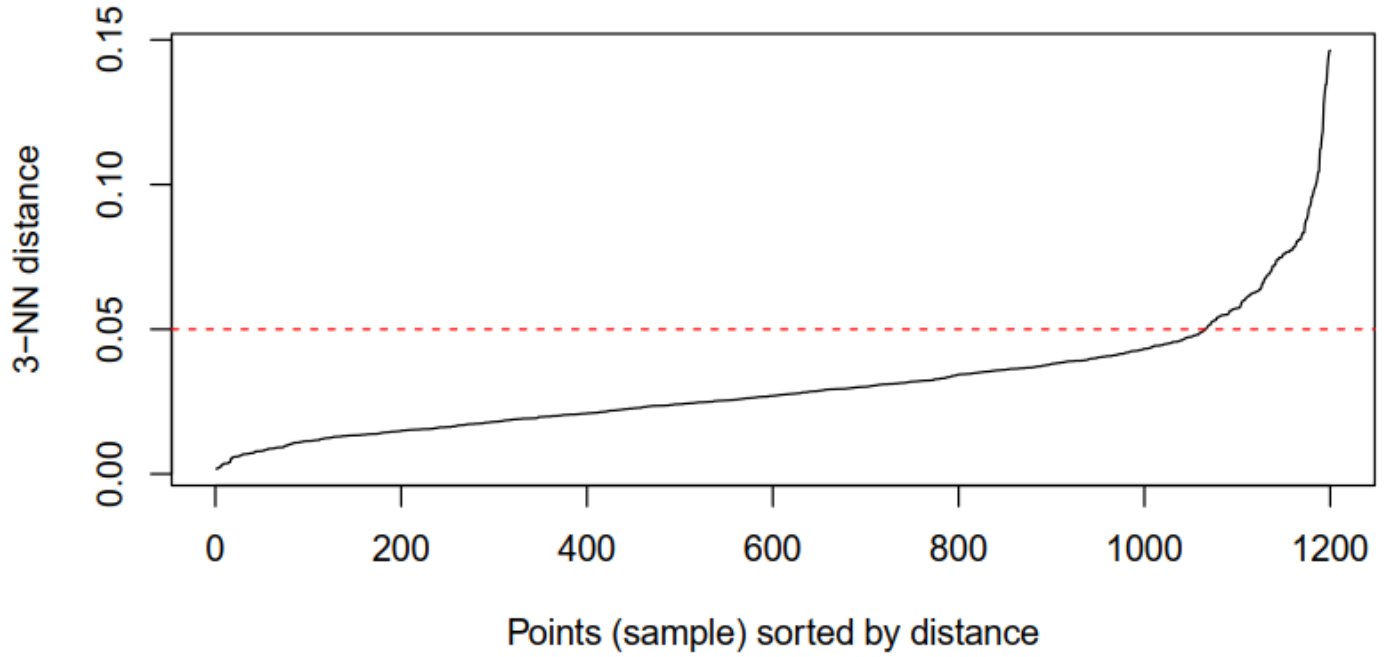


Figure 4: k-nearest neighbor distance plot.

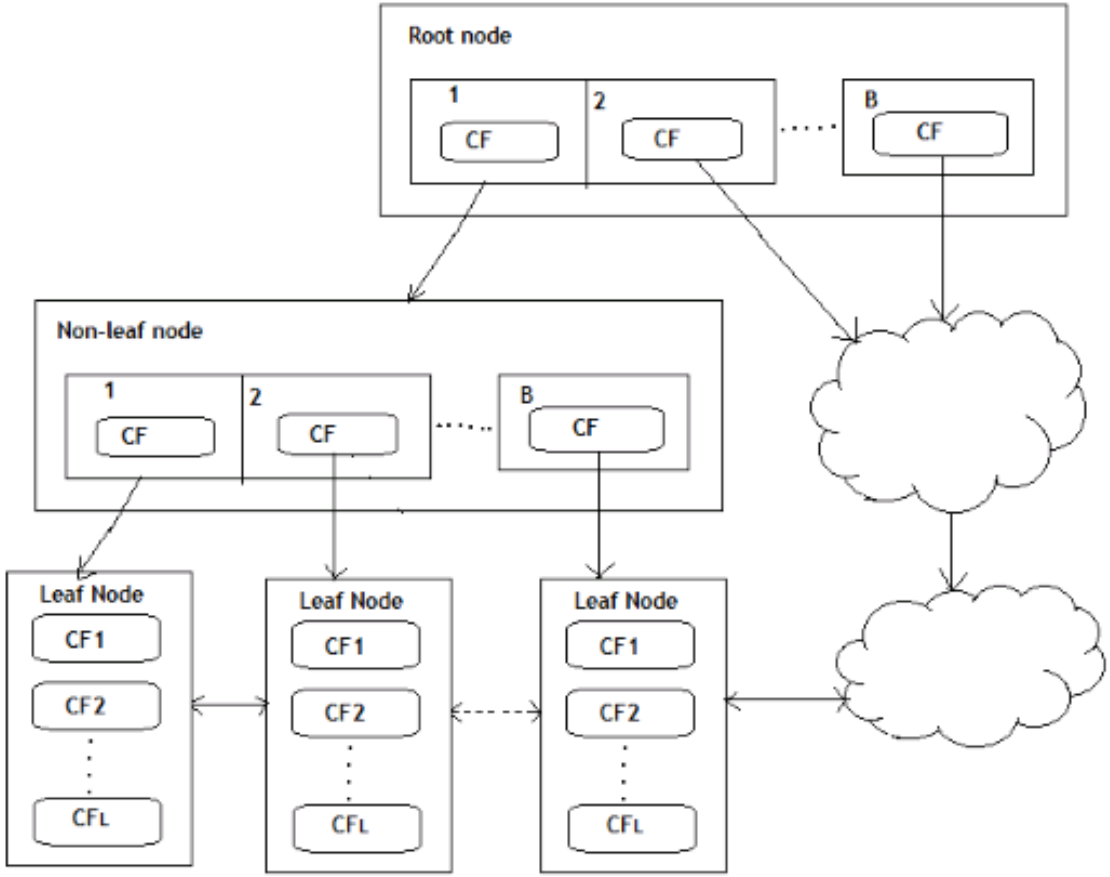
https://www.jstatsoft.org/article/view/v091i01

BIRCH

- **B**alanced **I**terative **R**educing and **C**lustering using **H**ierarchies
- używany do klasteryzacji głównie dużych danych
- jednokrotnie przegląda cały zbiór danych i buduje CF Tree
- używa struktury drzewa (Clustering Feature Tree), które w węzłach ma strukturę Clustering Features (CF)
- drzewo CF Tree jest używane, aby przechowywać istotne elementy używane w klasteryzacji bez potrzeby kolejnego przeglądania danych

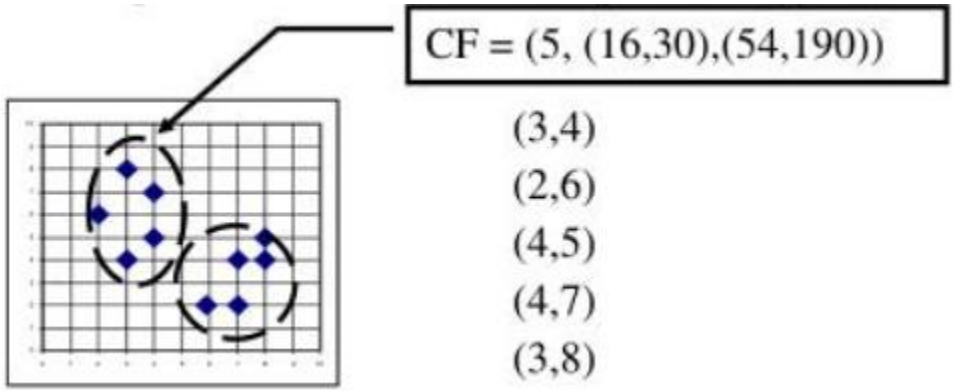
BIRCH

- Clustering Feature Tree



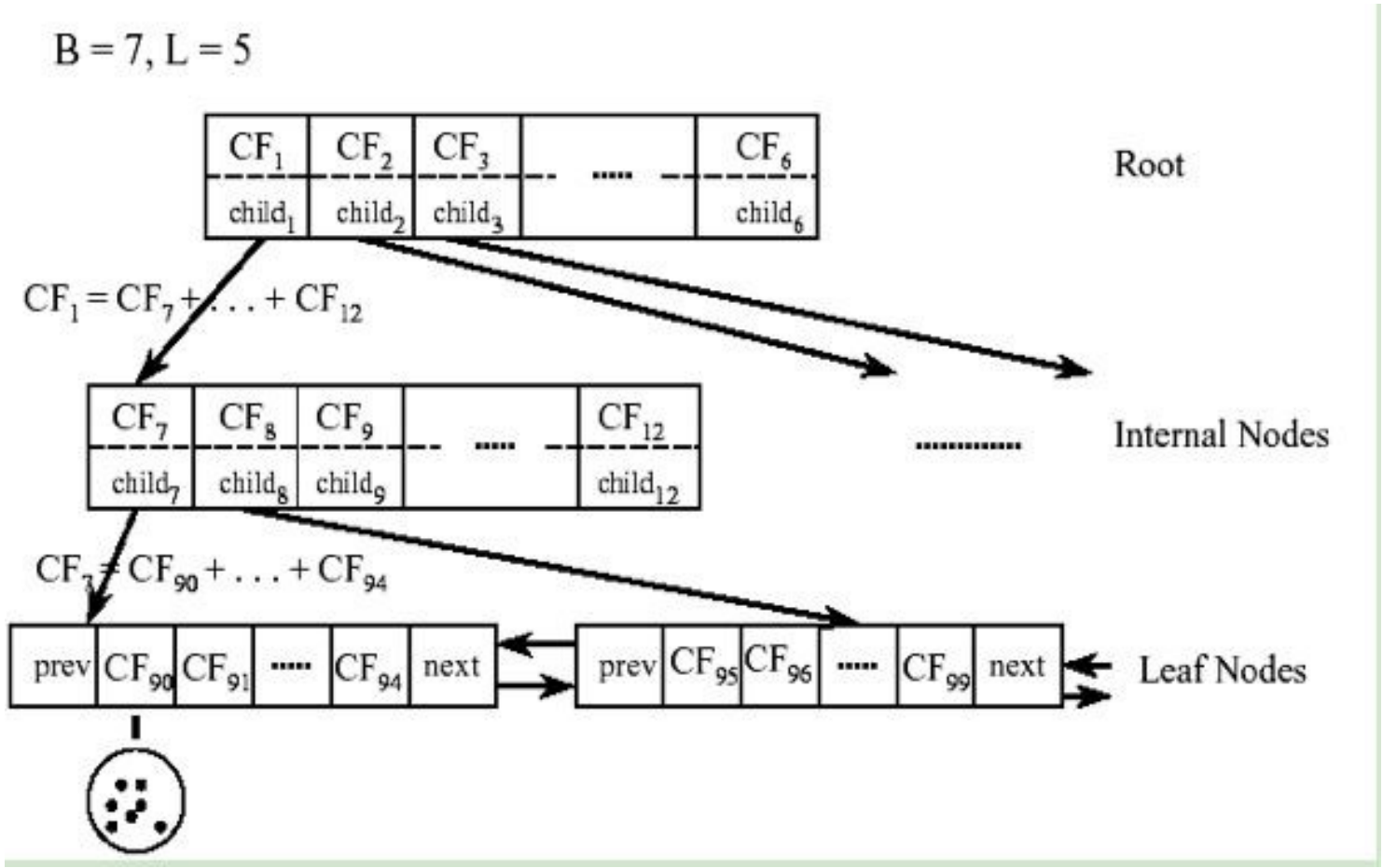
BIRCH

- Clustering Feature (CF) składa się z 3 elementów:
  - N – liczba punktów w grupie
  - LS – liniowa suma punktów $\overrightarrow{LS} = \sum_{i=1}^{N} \overrightarrow{X_i}$
  - SS – kwadratowa suma punktów $SS = \sum_{i=1}^{N} (\overrightarrow{X_i})^2$



BIRCH

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



BIRCH

- CF Tree ma 2 parametry:
  - branching factor – maksymalna liczba dzieci w drzewie
  - threshold – maksymalny promień „podgrupy” (subcluster) zapisanej w liściu węzła

- Węzły pośrednie drzewa przechowują sumy CF ze swoich dzieci

BIRCH

- Schemat działania
  - Faza 1: Zbudowanie CF Tree
  - Faza 2 (opcjonalna): Uproszczenie CF Tree przez łączenie najbliższych liści
  - Faza 3: Zastosowanie istniejących algorytmów grupowania dla obiektów w liściach
  - Faza 4 (opcjonalna): Poprawienie jakości grup przez przemieszczanie obiektów

BIRCH

- algorytm czuły na kolejność wstawiania punktów danych

- więcej szczegółów

https://www2.cs.sfu.ca/CourseCentral/459/han/papers/zhang96.pdf

Minibatch k-means

- zamiast przetwarzać cały zbiór w każdej iteracji k-means, przetwarzamy go w mini-batchach
- działa szybciej niż k-means
- zwłaszcza przydatny, gdy klasteryzujemy zbiory, które nie mieszczą się w pamięci

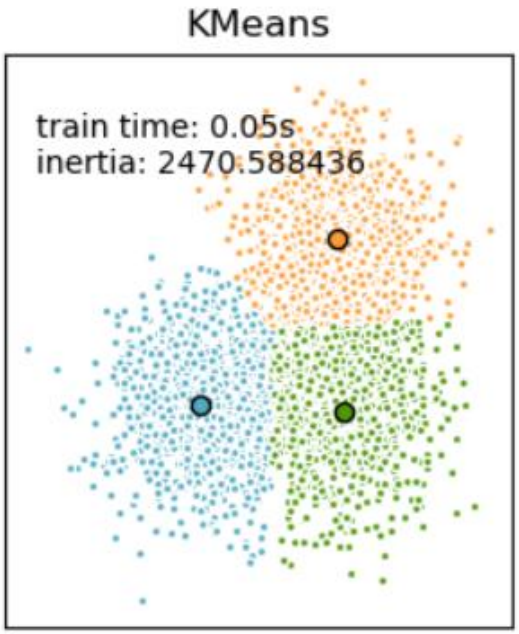
Minibatch k-means

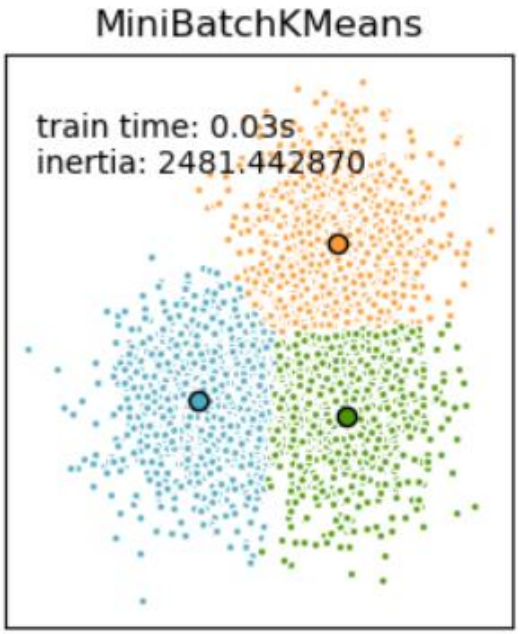
Algorithm 1 Mini-batch k-Means.

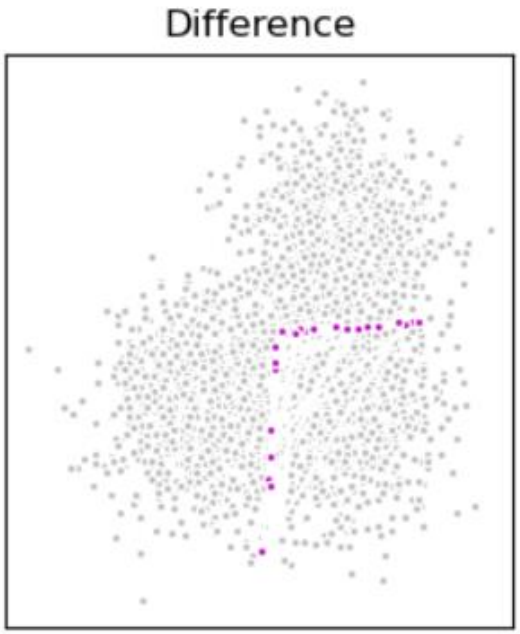
Given: k, mini-batch size b, iterations t, data set X
Initialize each $\mathbf{c} \in C$ with an $\mathbf{x}$ picked randomly from X
$\mathbf{v} \leftarrow 0$
for $i = 1$ to t **do**
  $M \leftarrow b$ examples picked randomly from X
  for $\mathbf{x} \in M$ **do**
    $\mathbf{d}[\mathbf{x}] \leftarrow f(C, \mathbf{x})$ // Cache the center nearest to $\mathbf{x}$
  end for
  for $\mathbf{x} \in M$ **do**
    $\mathbf{c} \leftarrow \mathbf{d}[\mathbf{x}]$ // Get cached center for this $\mathbf{x}$
    $\mathbf{v}[\mathbf{c}] \leftarrow \mathbf{v}[\mathbf{c}] + 1$ // Update per-center counts
    $\eta \leftarrow \frac{1}{\mathbf{v}[\mathbf{c}]}$ // Get per-center learning rate
    $\mathbf{c} \leftarrow (1 - \eta)\mathbf{c} + \eta\mathbf{x}$ // Take gradient step
  end for
end for

https://www.eecs.tufts.edu/~dsculley/papers/fastkmeans.pdf

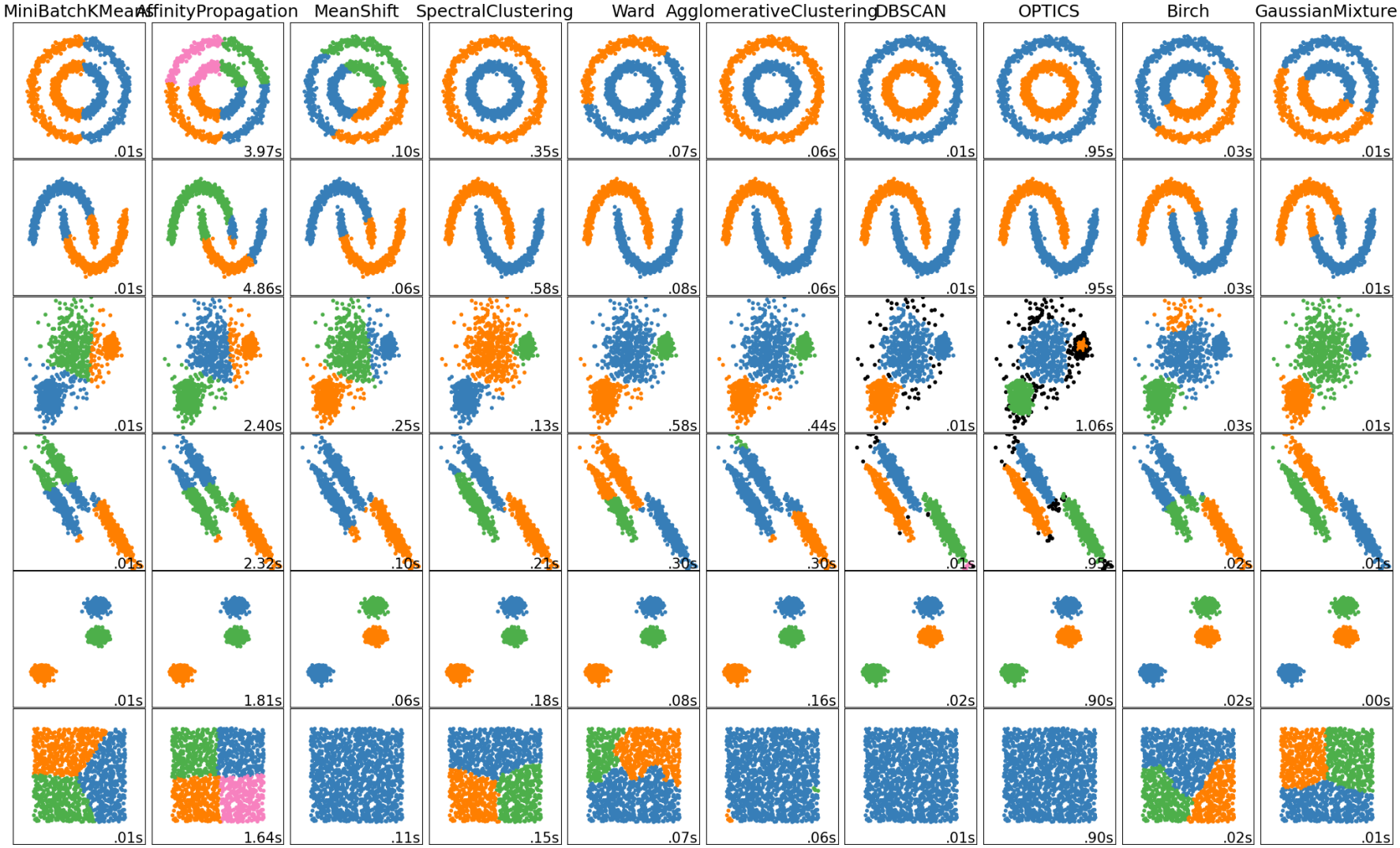
Minibatch k-means







Porównanie algorytmów



Dziękujemy za uwagę.