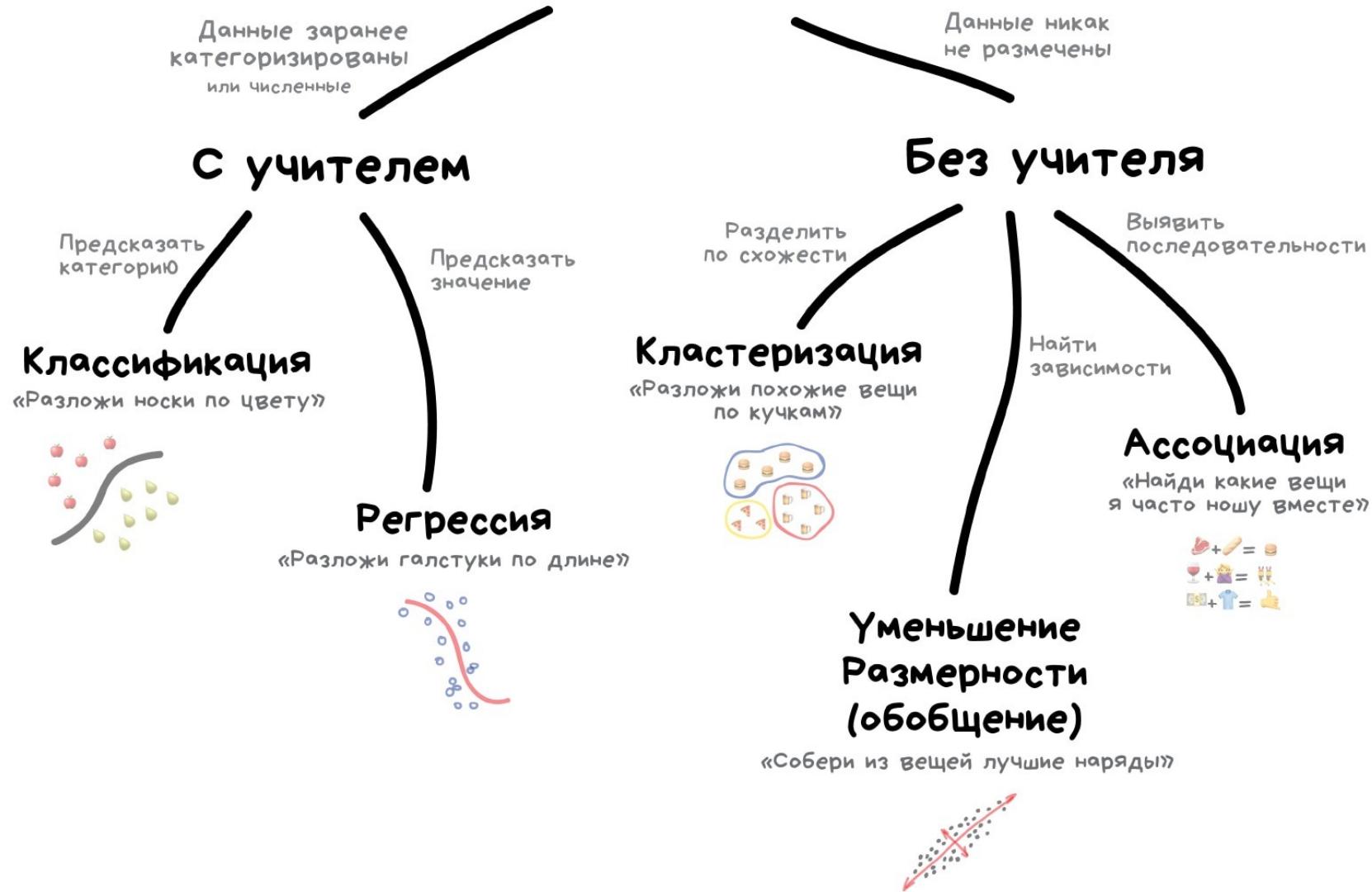


# Кластеризация

МАКСИМОВСКАЯ  
АНАСТАСИЯ

# Классическое Обучение



# Кластеризация

---

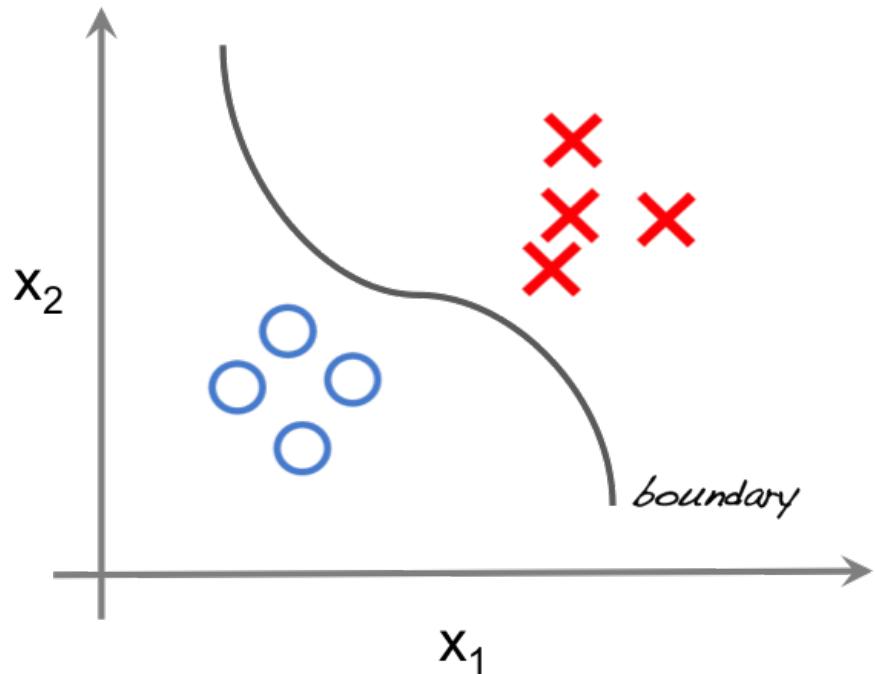
**Задача кластеризации** – выявить в данных  $K$  кластеров — таких областей, что объекты внутри одного кластера похожи друг на друга, а объекты из разных кластеров друг на друга не похожи.

Более формально, требуется построить алгоритм  $a : X \rightarrow \{1, \dots, K\}$ , определяющий для каждого объекта номер его кластера; число кластеров  $K$  может либо быть известно, либо являться параметром.

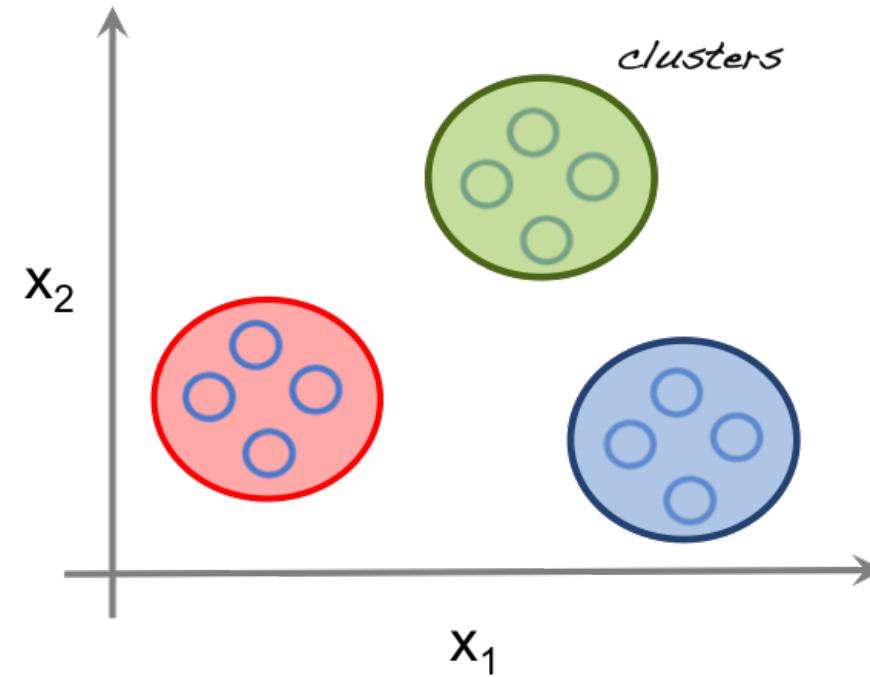
# Кластеризация

---

Supervised learning



Unsupervised learning



# Метрики качества

---

# Метрики качества

---

- Внешние метрики – используют информацию об истинных метках объектов
- Внутренние метрики – оценивают качество кластеризации, основываясь только на наборе данных
- Дальше поговорим о внутренних

# Внутрикластерное расстояние

---

$$\sum_{k=1}^K \sum_{i=1}^{\ell} [a(x_i) = k] \rho(x_i, c_k)$$

$\rho(x, c)$  – некоторая функция расстояния

- Минимизируем, поскольку в идеале все объекты кластера должны быть одинаковыми

# Межклusterное расстояние

---

$$\sum_{i,j=1}^{\ell} [a(x_i) \neq a(x_j)] \rho(x_i, x_j)$$

- Максимизируем, поскольку объекты из разных кластеров должны быть как можно менее похожими друг на друга

# Отношение расстояний

---

Минимизируем отношение внутрикластерного расстояния к среднему межклластерному:

$$\frac{F_0}{F_1} \rightarrow \min$$

Это будет достигаться, если расстояние между кластерами максимально, а внутри кластера — минимально

# Среднее расстояние до центра

---

$$\phi_0 = \sum_{y \in Y} \frac{1}{|K_y|} \sum_{i:y_i=y} \rho^2(x_i, \mu_y)$$

- Минимизируем, чтобы объекты располагались максимально близко к центру кластера

# Сумма межклusterных расстояний

---

$$\phi_1 = \sum_{y_i=y} \rho^2(\mu_i, \mu)$$

- Считаем от центра кластера до центра выборки
- Максимизируем, так как кластеры должны находиться друг от друга на как можно большем расстоянии.
- Также можно считать отношение среднего расстояния до центра к сумме межклusterных расстояний (минимизируем)

# Индекс Данна (Dunn Index)

---

$$\frac{\min_{1 \leq k < k' \leq K} d(k, k')}{\max_{1 \leq k \leq K} d(k)},$$

$d(k', k)$  – расстояние между кластерами  $k$  и  $k'$ ,  $d(k)$  – внутрикластерное расстояние для  $k$  – ого кластера

- Данный индекс максимизируем

# Коэффициент силуэта

---

Значение коэффициента силуэта показывает, насколько объект похож на свой кластер по сравнению с другими кластерами

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

а — среднее расстояние от данного объекта до объектов из того же кластера

б — среднее расстояние от данного объекта до объектов из ближайшего кластера

- silhouette\_score в sklearn

# Коэффициент силуэта

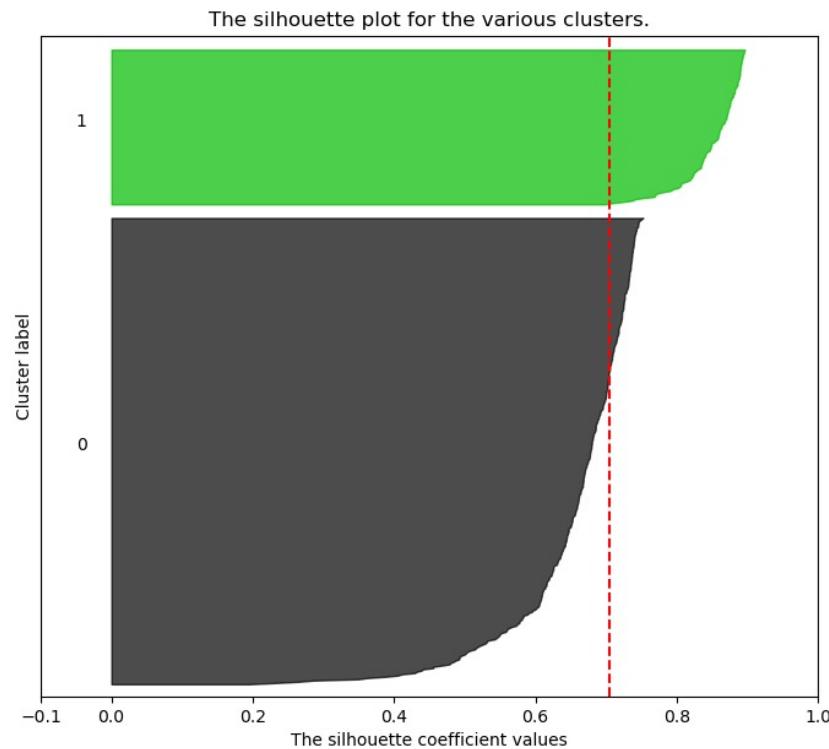
---

- Значение коэффициента находится в диапазоне от -1 до +1, где высокое значение указывает, что объект хорошо согласуется с кластером, которому он принадлежит, и плохо согласуется с «чужими» кластерами
- Если у подавляющего большинства объектов этот коэффициент высокий, то можно считать кластеризацию достаточно качественной
- Если же у большого числа объектов низкий или отрицательный коэффициент силуэта, то, возможно, кластеров слишком много, слишком мало или просто данные плохо поддаются разделению на кластеры

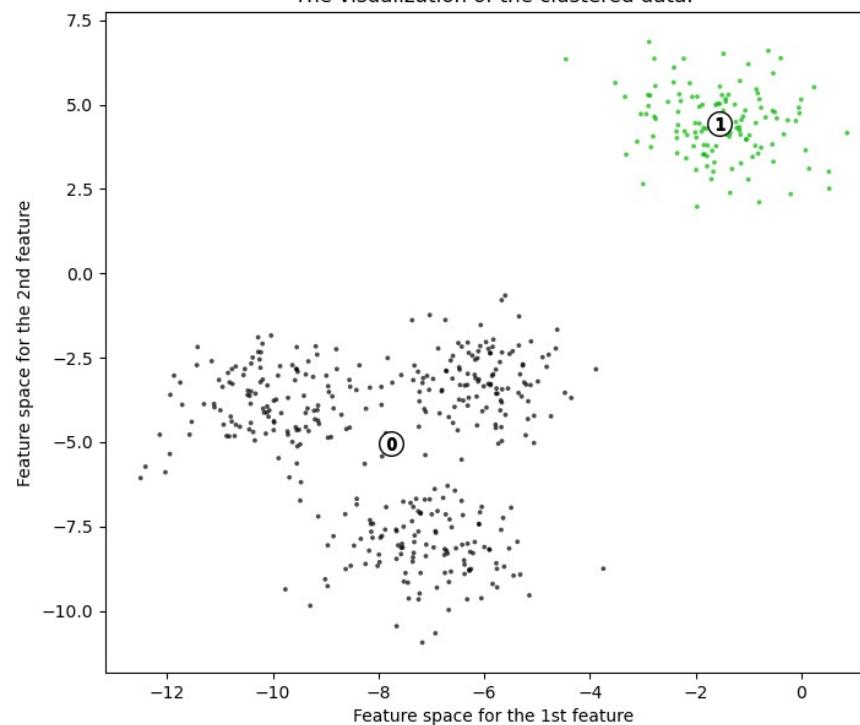
# Коэффициент силуэта

---

Silhouette analysis for KMeans clustering on sample data with n\_clusters = 2

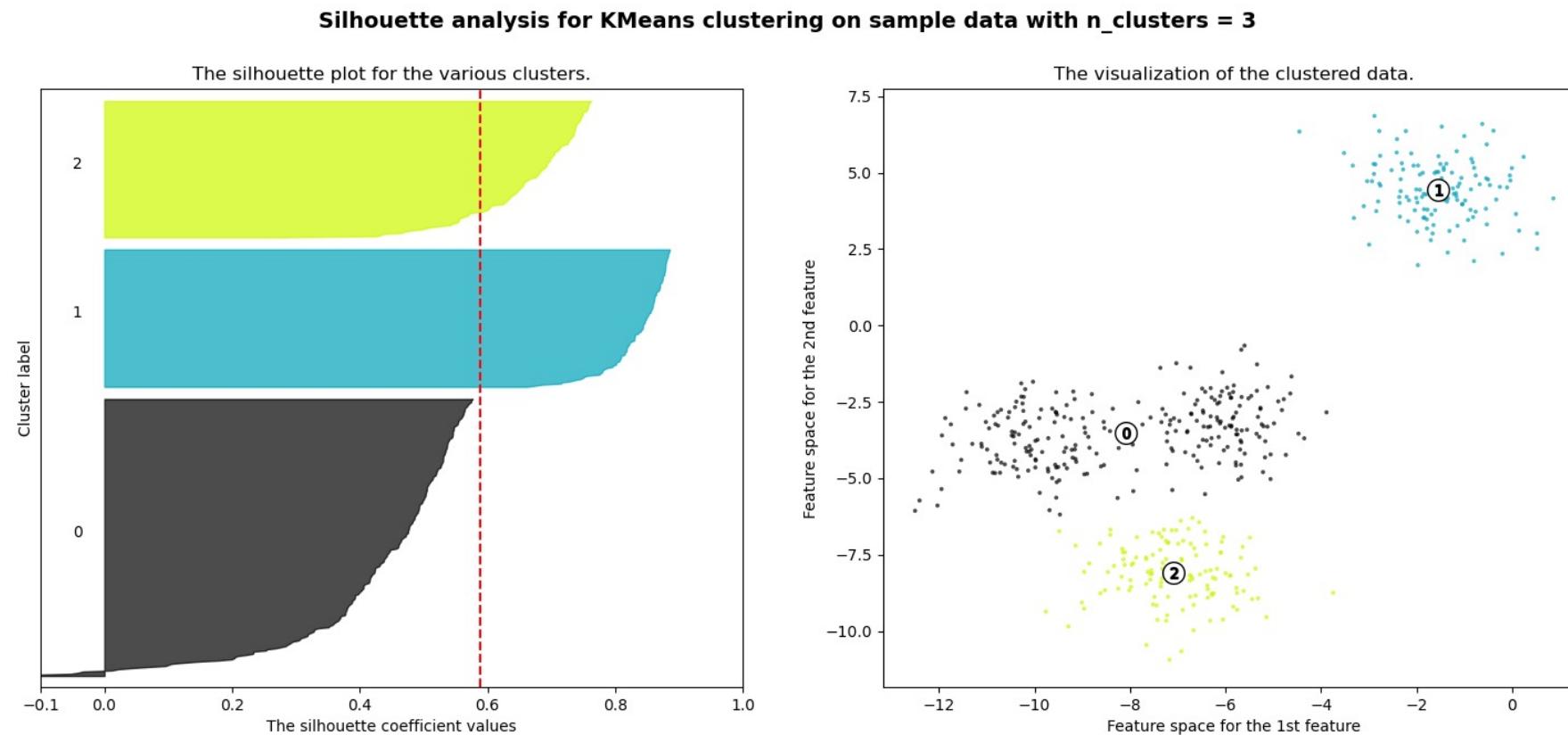


The visualization of the clustered data.



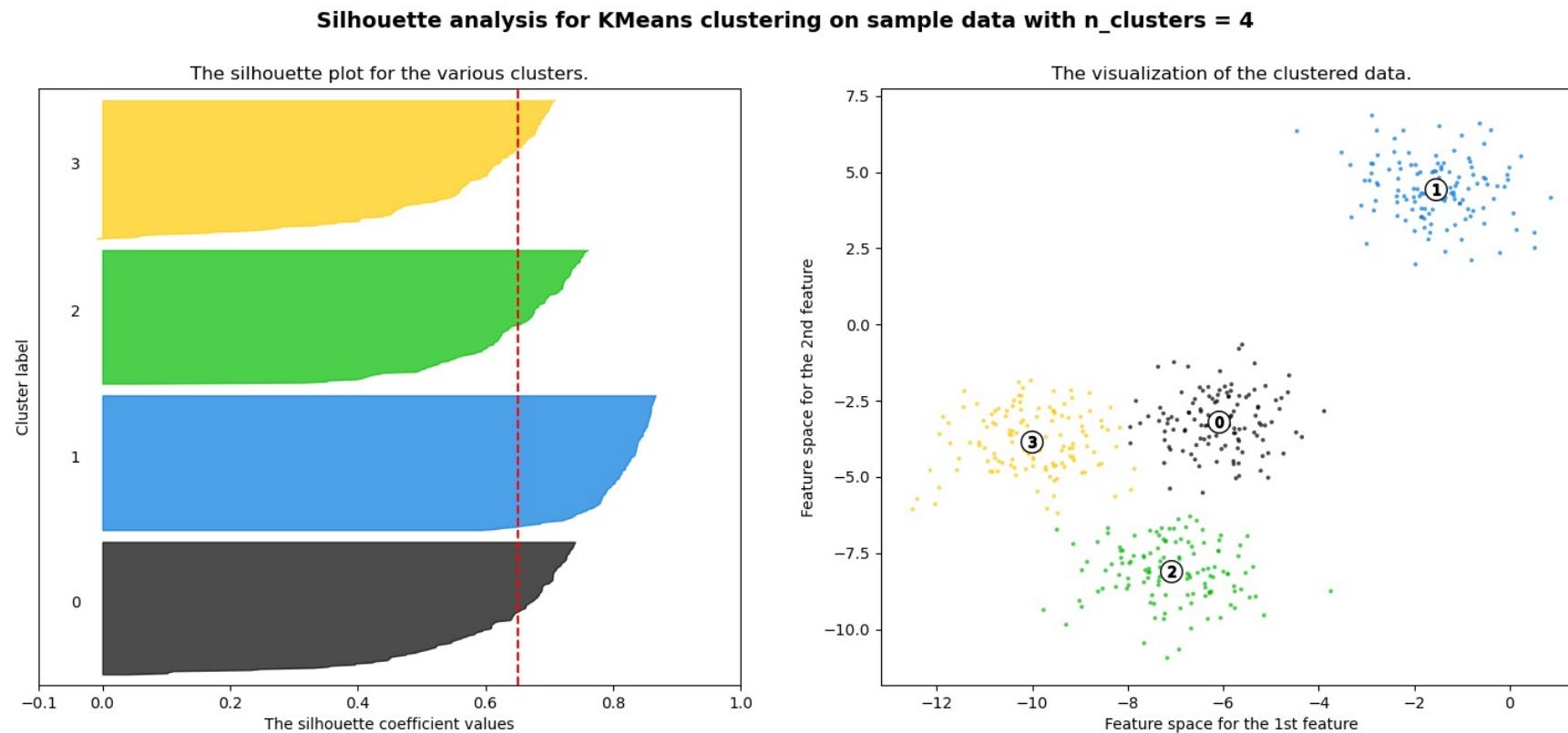
# Коэффициент силуэта

---



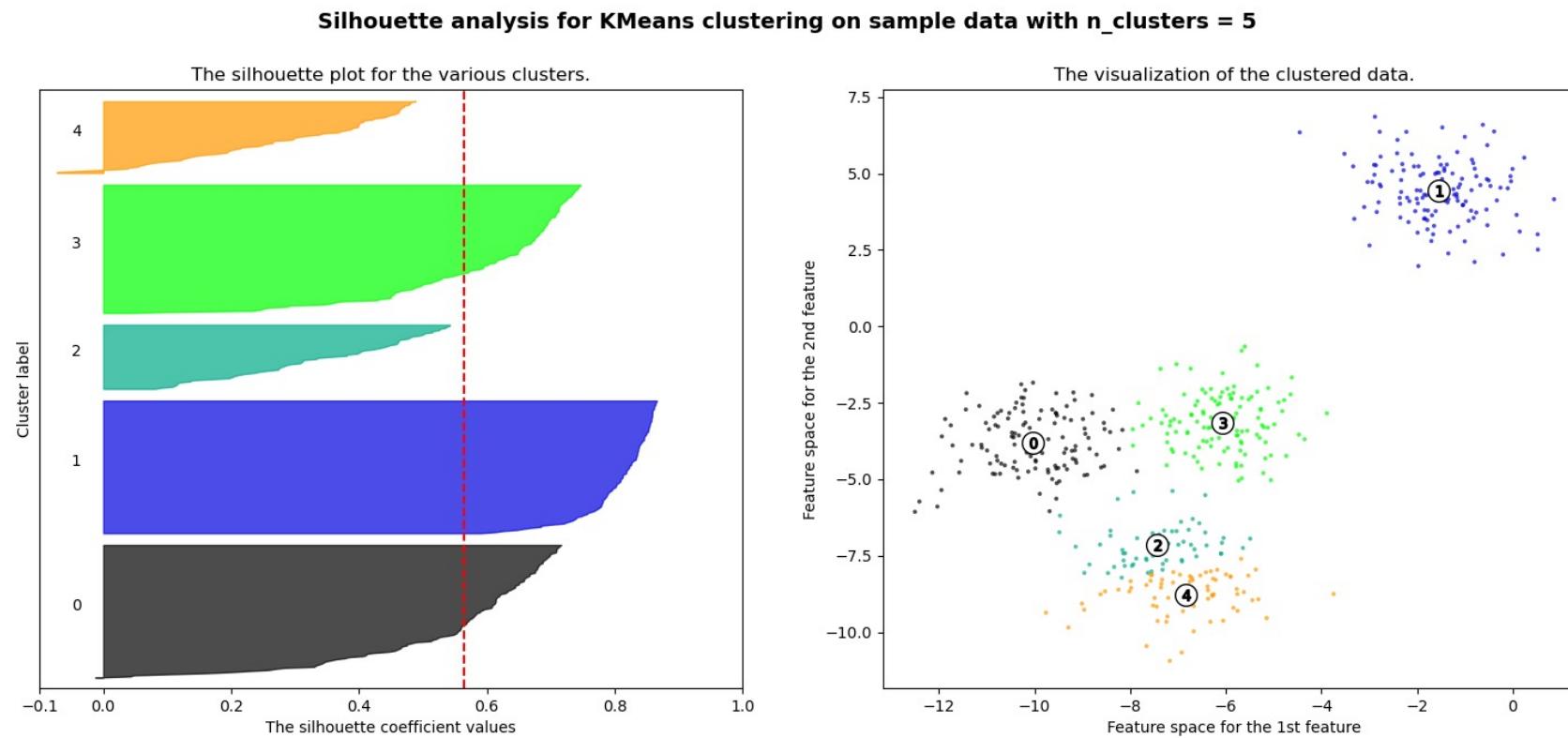
# Коэффициент силуэта

---



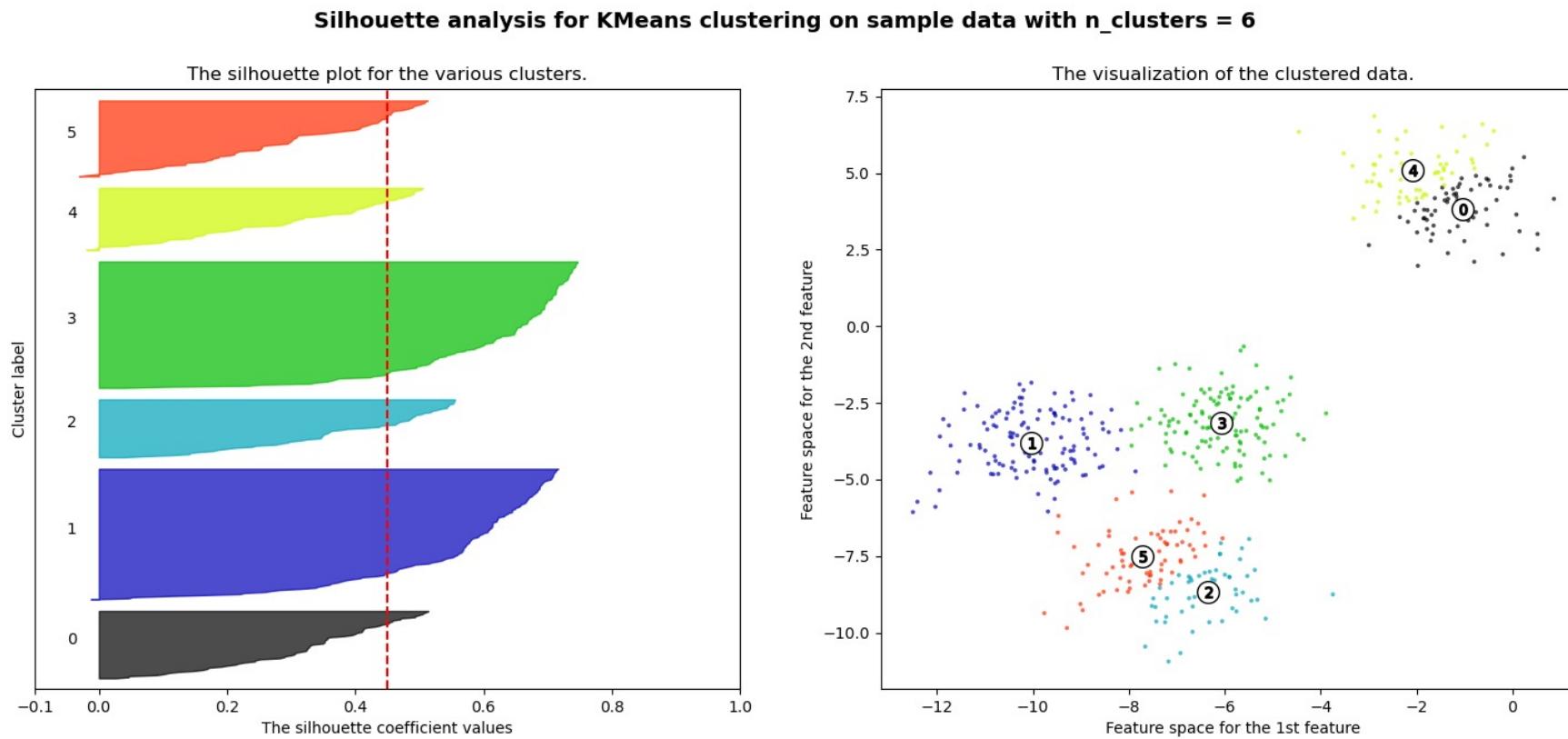
# Коэффициент силуэта

---



# Коэффициент силуэта

---



# Проверка наличия кластерной структуры

---

1. Генерируем  $p$  случайных точек из равномерного распределения.
2. Генерируем  $p$  случайных точек из обучающей выборки.
3. Вычисляем величину:

$$H_{ind} = \frac{\sum_n w_i}{\sum_n q_i + \sum_n w_i}$$

# Проверка наличия кластерной структуры

---

- Эта статистика является одним из **индикаторов тенденции к группированию**
- Для её расчета создается  $B$  псевдо-наборов данных, сгенерированных случайным образом на основе распределения с тем же стандартным отклонением, что и оригинальный набор данных
- Для каждого наблюдения  $i$  из  $n$  рассчитывается среднее расстояние до  $k$  ближайших соседей:  $w$  между реальными объектами и  $q$  между искусственными объектами и их ближайшими реальными соседями.

# Проверка наличия кластерной структуры

---

- Если статистика превышает значение 0.5, то мы оставляем нулевую гипотезу, которая заключается в том, что  $q$  и  $w$  подобны и группируемые объекты распределены однородно и случайно
- В случае, если статистика меньше чем 0.25, можно с 90% уверенностью отвергнуть нулевую гипотезу в пользу альтернативной, которая говорит о наличии тенденции к группировке данных.

# Внешние метрики

---

# Однородность

---

$$h = 1 - \frac{H(C|K)}{H(C)}$$

В данной формуле  $H(C|K)$  — энтропия класса при условии кластера, а  $H(C)$  — энтропия класса

Максимальное значение однородность достигает в том случае, если в кластере объекты одного класса

- `homogeneity_score` в `sklearn`

# Полнота

---

$$c = 1 - \frac{H(K|C)}{H(K)}$$

**Полнота** достигает максимальное значение в том случае, когда все объекты из класса принадлежат одному кластеру

В данной формуле  $H(K|C)$  — энтропия кластера при условии класса, а  $H(K)$  — энтропия кластера

➤ `completeness_score` в `sklearn`

# V-мера

---

Среднее гармоническое однородности и полноты, показывает насколько схожи 2 разбиения:

$$u = 2 \frac{hc}{h + c}$$

- `v_measure_score` в `sklearn`

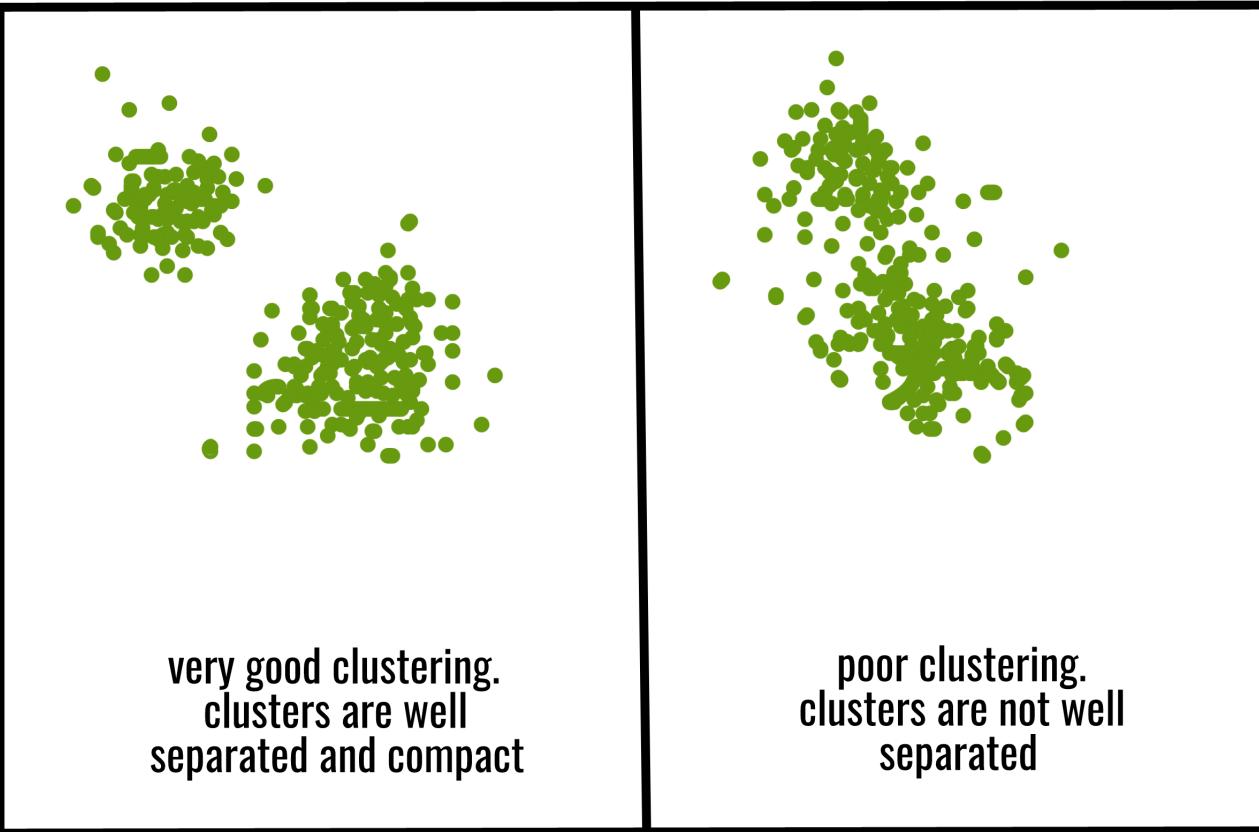
# Другие

---

- Внешние метрики возможно использовать, если известно истинное распределение объектов по кластерам
- В этом случае задачу кластеризации можно рассматривать как задачу многоклассовой классификации, и использовать любую метрику оттуда — F-меру с микро- или макроусреднением.

# Метрики качества

---



# Расстояние между объектами

---

**Евклидово расстояние** – расстояние между точками в общепринятом понимании, то есть геометрическое расстояние между двумя точками.

$$\rho(a, b) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

**Квадрат евклидова расстояния:**

$$\rho(a, b) = (x_1 - x_2)^2 + (y_1 - y_2)^2$$

# Расстояние между объектами

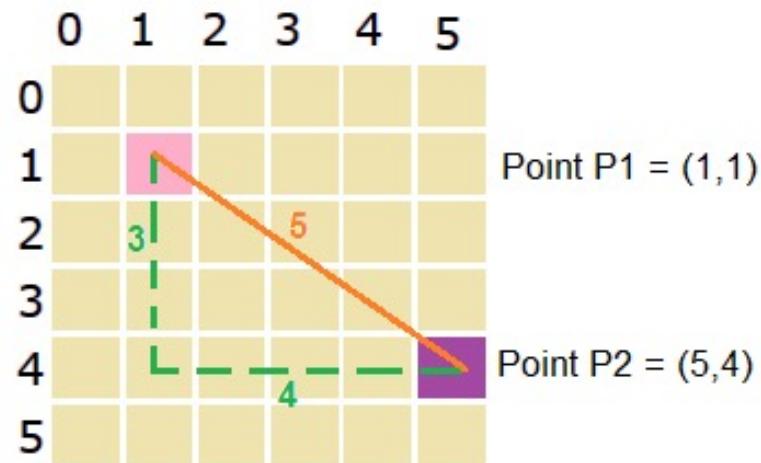
---

Расстояние городских кварталов (манхэттенское расстояние):

$$\rho(a, b) = |x_1 - x_2| + |y_1 - y_2|$$

# Расстояние между объектами

---

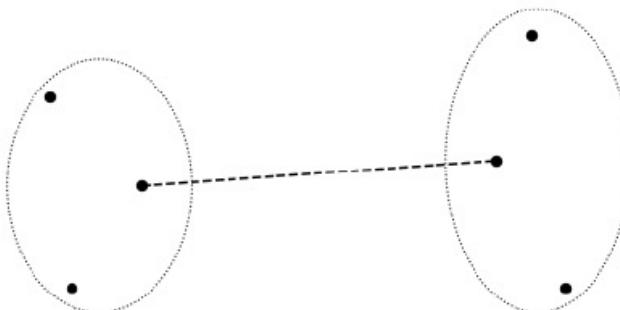


$$\text{Euclidean distance} = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

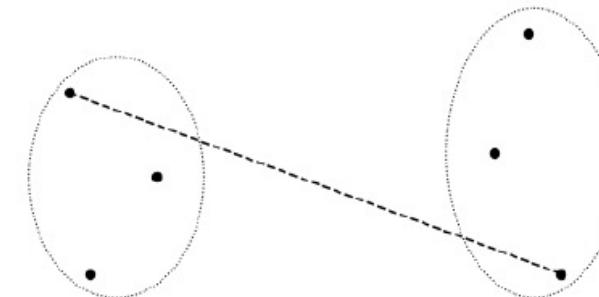
$$\text{Manhattan distance} = |5-1| + |4-1| = 7$$

# Расстояние между объектами

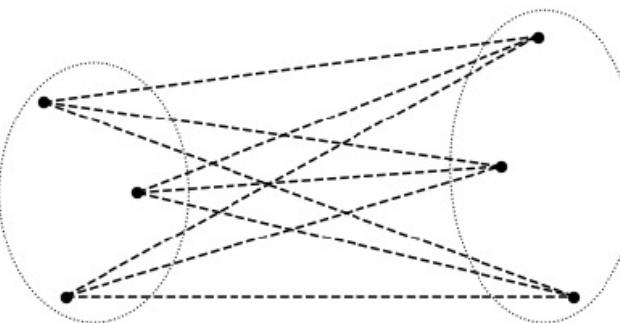
---



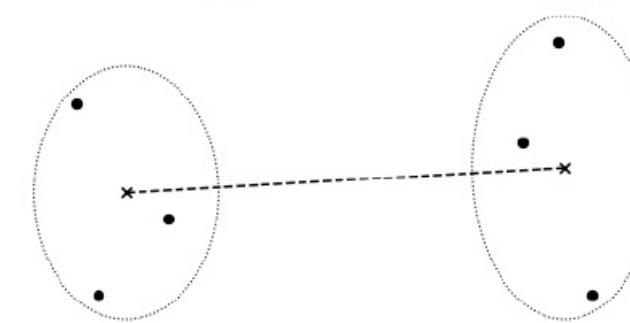
Расстояние ближнего соседа



Расстояние дальнего соседа



Групповое среднее расстояние



Расстояние между центрами

# Методы кластеризации

---

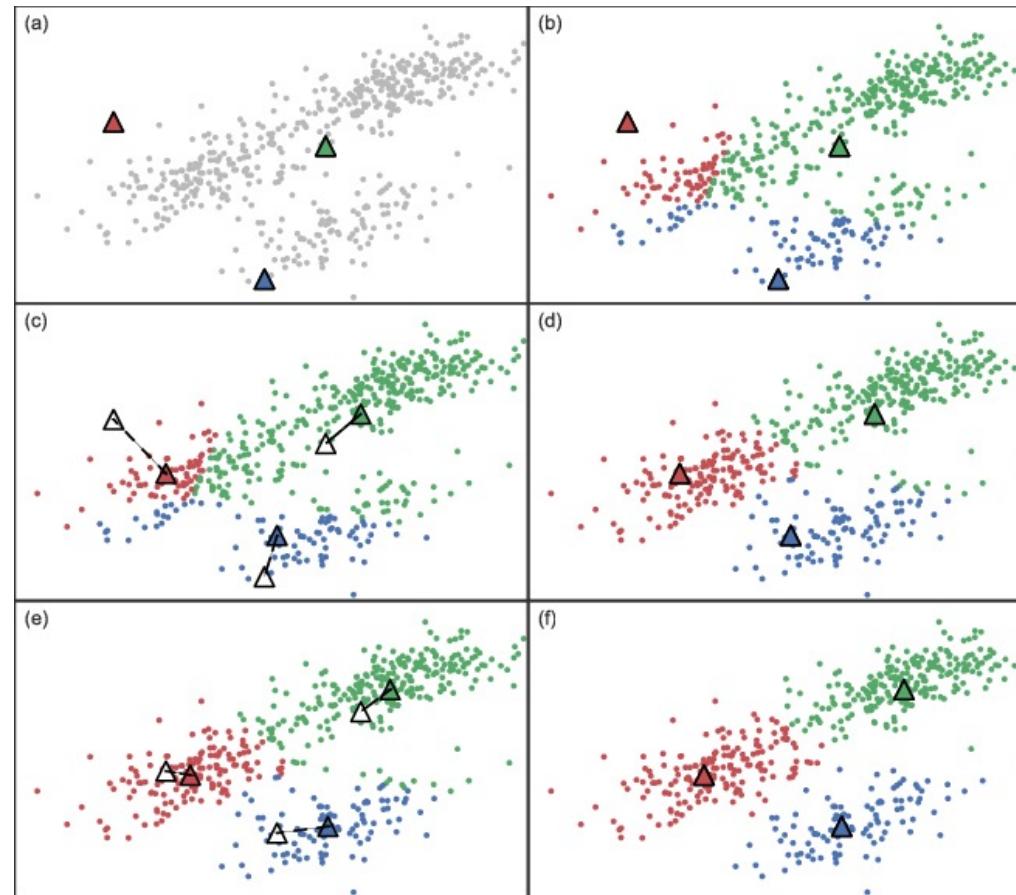
# K-Means (К-средних)

---

1. Выбрать количество кластеров, которое нам кажется оптимальным для наших данных.
2. Выбрать случайным образом в пространстве наших данных центроиды.
3. Для каждой точки набора данных посчитать, к какому центроиду она ближе.
4. Переместить каждый центроид в центр выборки, которую мы отнесли к этому центроиду. Каждый центроид на каждой итерации — вектор, элементы которого представляют собой средние значения признаков, вычисленные по всем записям кластера.
5. Повторять шаги 3-4 фиксированное число раз или до тех пор, пока центроиды не сойдутся.

# K-Means (К-средних)

---



# Недостатки K-Means

---

- Не гарантируется достижение глобального минимума суммарного квадратичного отклонения, а только одного из локальных минимумов.
- Результат зависит от выбора исходных центров кластеров, их оптимальный выбор неизвестен.
- Число кластеров надо знать заранее.

# K-Means++

---

1. Выбрать первый центроид случайным образом (среди всех точек)
2. Для каждой точки найти значение квадрата расстояния до ближайшего центроида (из тех, которые уже выбраны)
3. Выбрать из этих точек следующий центроид так, чтобы вероятность выбора точки была пропорциональна вычисленному для неё квадрату расстояния
4. Повторять шаги 2 и 3 до тех пор, пока не будут найдены все необходимые центроиды.

Далее выполняется основной алгоритм  $k$ -means.

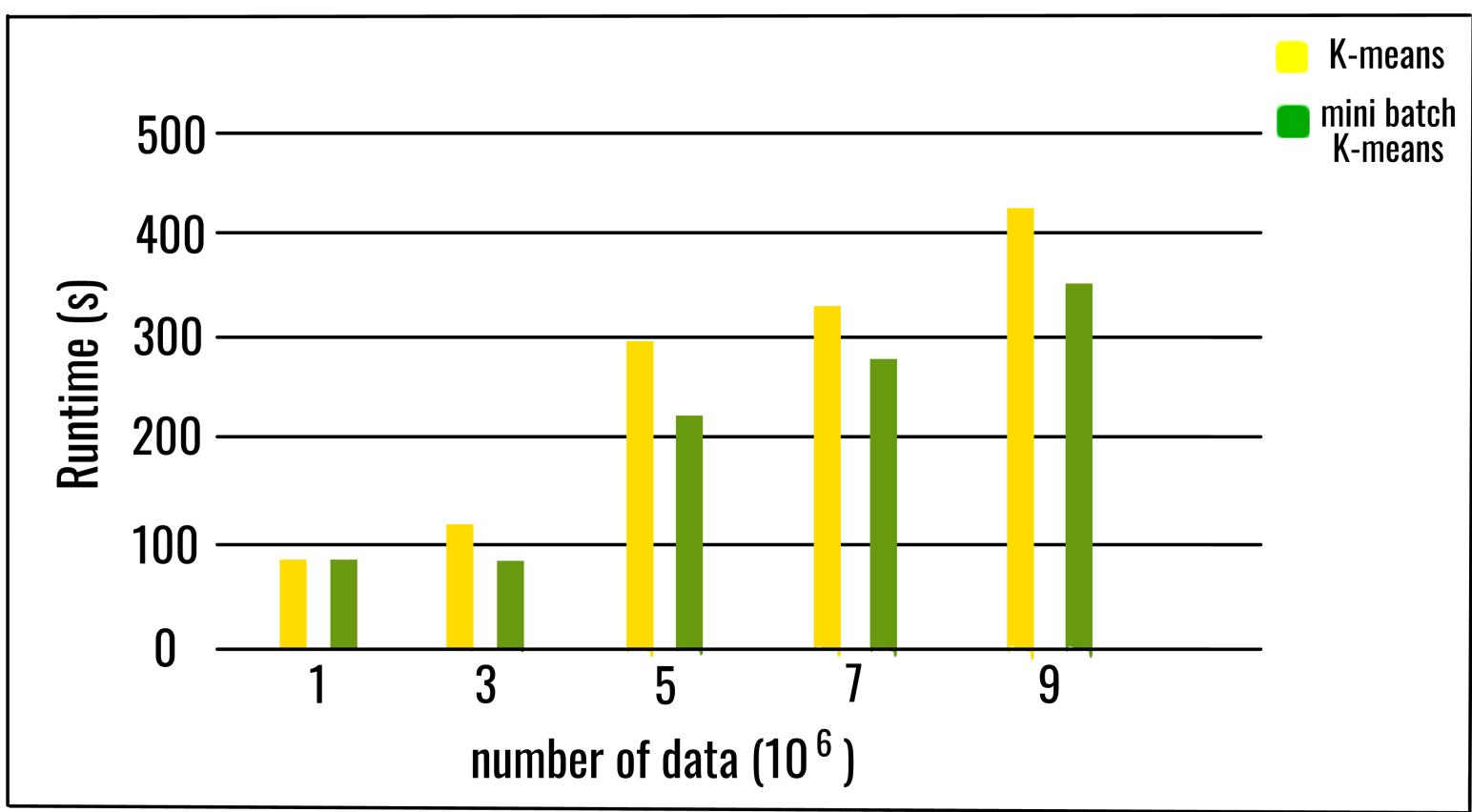
# Mini-Batch K-Means

---

- Данная вариация *k-means* используется в случае, если данных очень много.
- Из-за объема данных вычисление центров по всей выборке происходит долго.
- Решение проблемы: на каждом шаге *k-means* работать с небольшой **подвыборкой данных**.
- В общем случае упрощённый алгоритм должен сходиться к тому же результату, что и на полной выборке. Однако исследования показывают, что качество кластеров может ухудшаться по сравнению с классическим *k-means*.

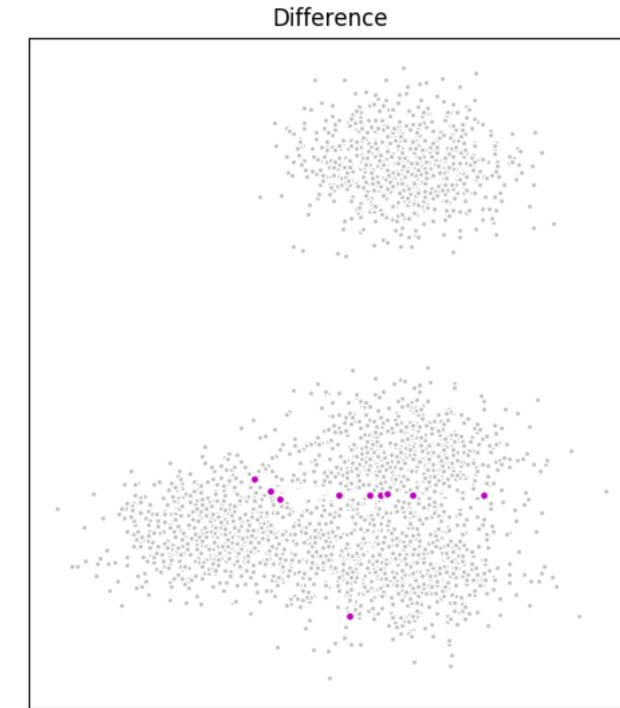
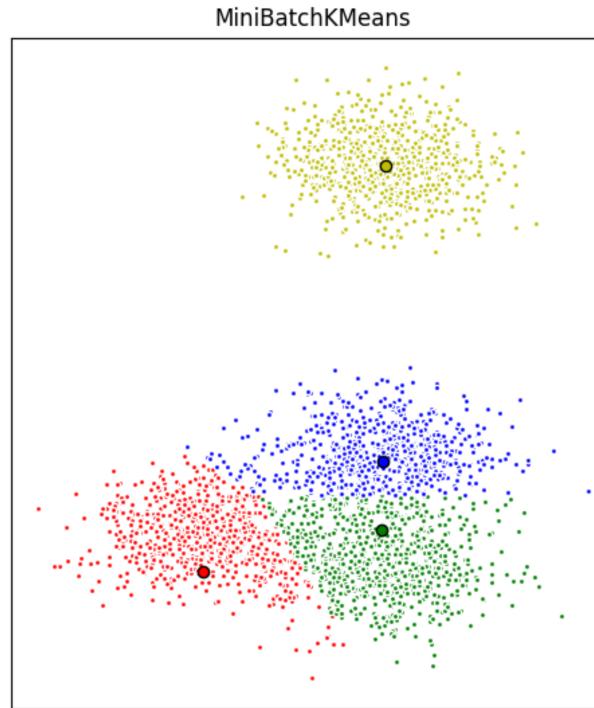
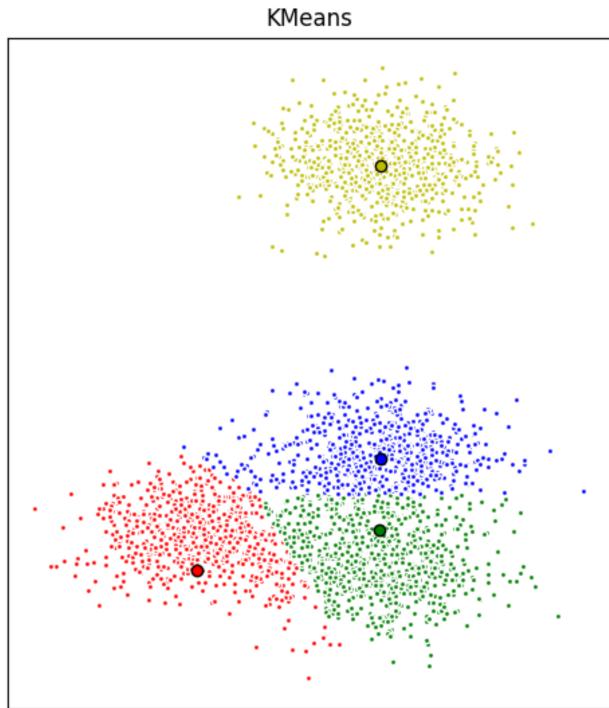
# Mini-Batch K-Means

---



# Mini-Batch K-Means

---



# Графовые методы

---

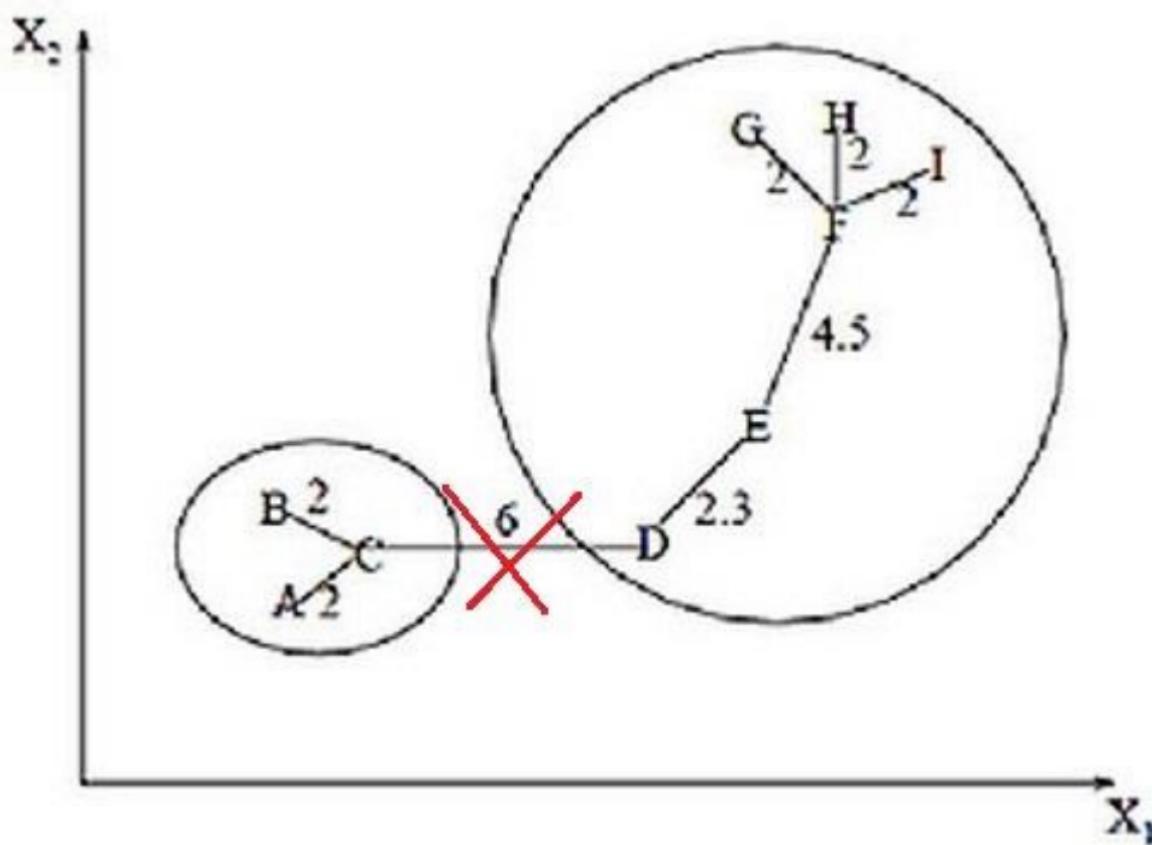
- Выборка представляется в виде графа, где в вершинах стоят объекты, а на рёбрах – расстояния между ними

Алгоритм выделения связных компонент:

1. Из графа удаляются все ребра, для которых расстояния больше некоторого значения  $R$
2. Кластеры – объекты, попадающие в одну компоненту связности

# Графовые методы

---



# Иерархическая кластеризация

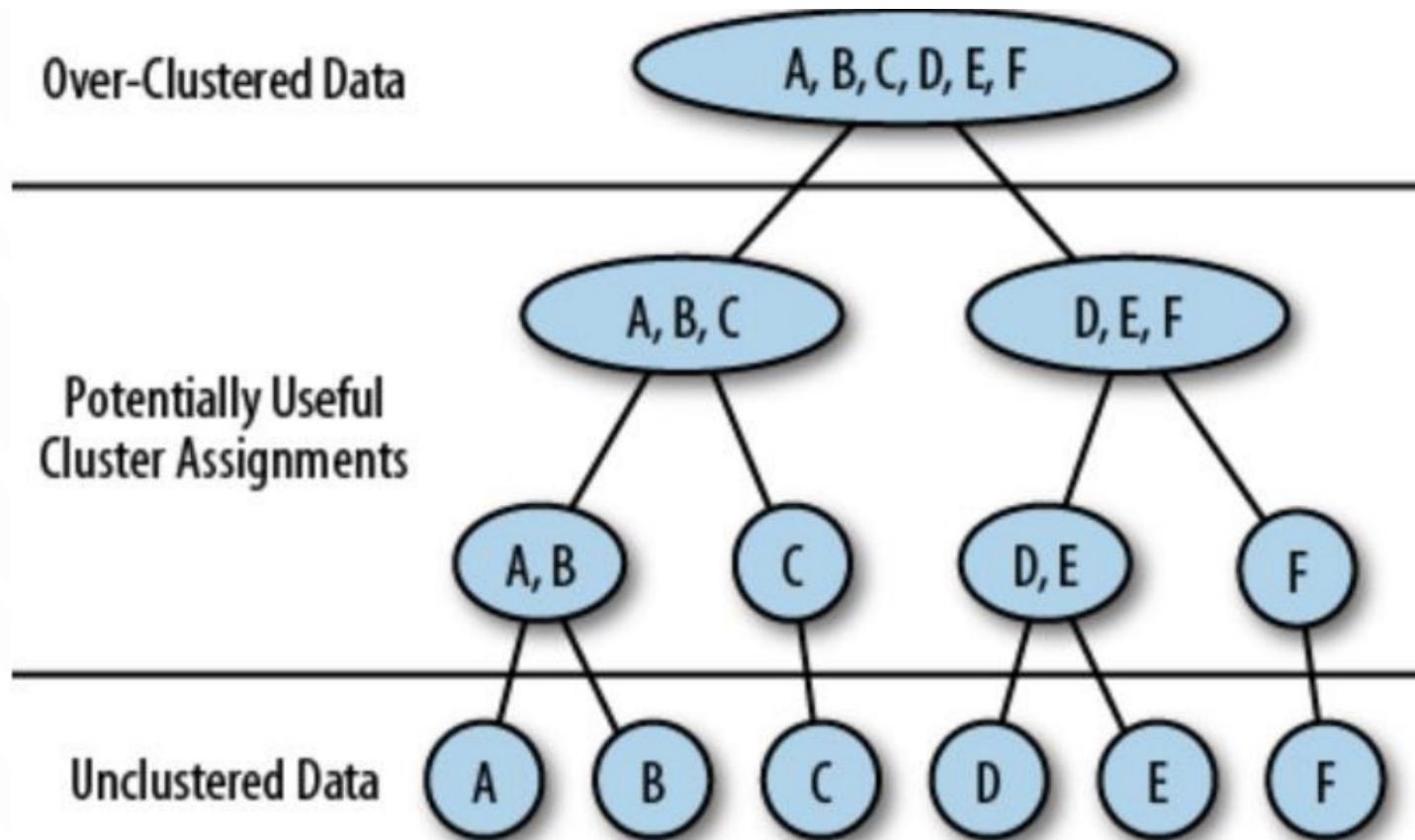
---

Иерархия кластеров:

- на верхнем уровне – один большой кластер
- на нижнем уровне -  $l$  кластеров, каждый из которых состоит из одного объекта

# Иерархическая кластеризация

---



# Иерархическая кластеризация

---

Иерархическая кластеризация делится на две стратегии:

- Агломеративная — снизу-вверх, объединяем точки в кластеры
- Дивизионная — сверху-вниз, разделяем один большой кластер на малые.

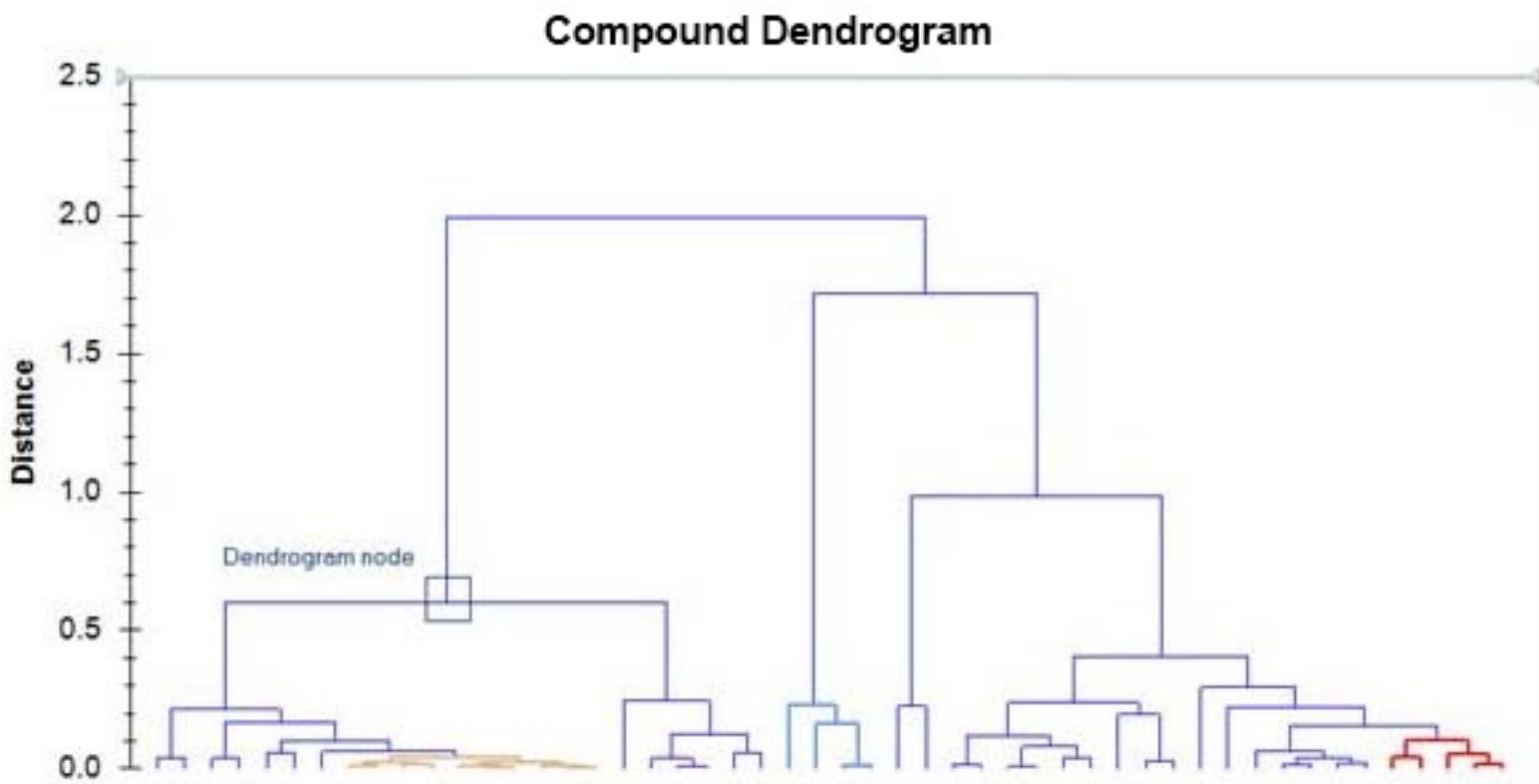
# Агломеративная кластеризация

---

1. Назначаем каждой точке свой кластер.
2. Сортируем попарные расстояния между центрами кластеров по возрастанию.
3. Берём пару ближайших кластеров, склеиваем их в один и пересчитываем центр кластера.
4. Повторяем шаги 2-3 до тех пор, пока все данные не склеятся в один кластер.

# Дендрограмма

---



# DBSCAN

---

- Расшифровывается как *Dense-based spatial clustering of applications with noise*.
- Это основанная на плотности **пространственная кластеризация** для приложений с шумами.

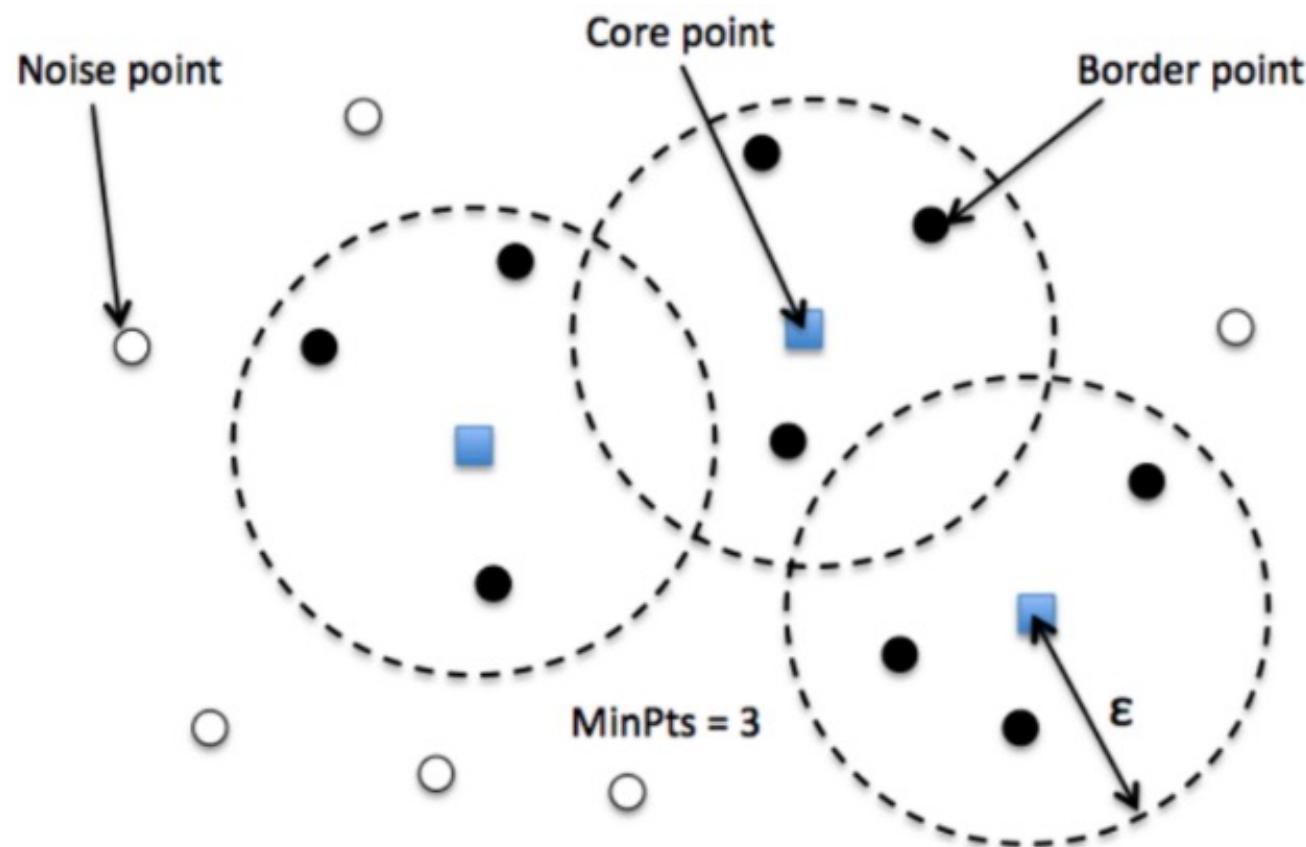
# DBSCAN

---

1. Случайно выбираем точку, которую не посещали. Окрестность точки извлекается с использованием расстояния  $\epsilon$ .
2. Если в этой окрестности точек  $\geq \text{minPoints}$ , тогда точка становится первой точкой в новом кластере. Иначе — помечаем точку как **шум**, она становится посещённой.
3. Точки из окрестности становятся частью кластера. Для каждой из них изучаем **окрестность**: если точек в окрестности  $< \text{minPoints}$ , то помечаем точку как **границную**.
4. Повторяем пункты 2 и 3, пока не определим все точки в кластере.
5. Повторяем пункты 1–4, пока все точки не станут просмотренными.

# DBSCAN

---



# DBSCAN

---

Достоинства алгоритма:

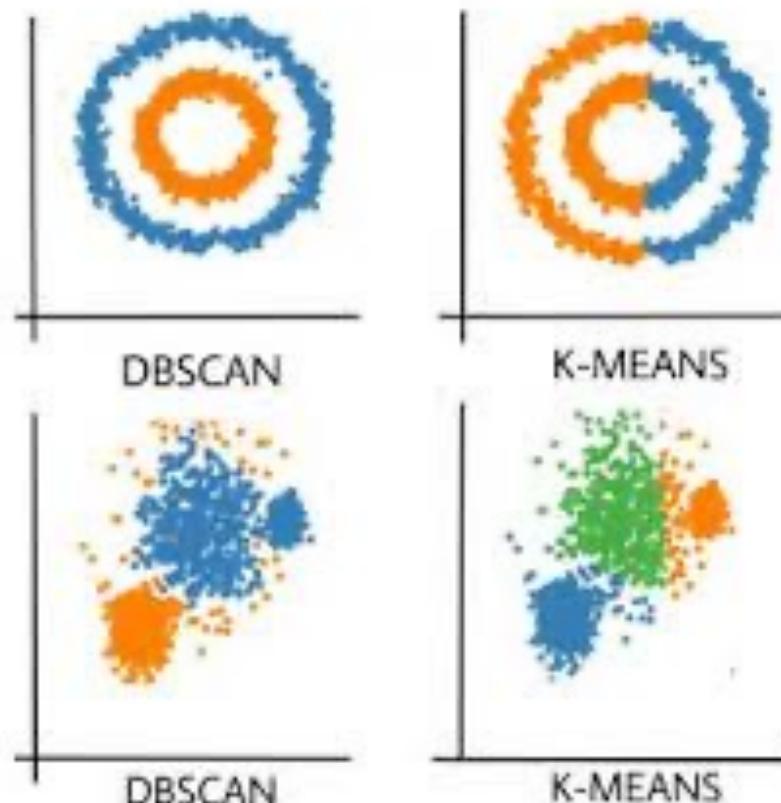
- не требуется число кластеров;
- определяем кластеры произвольной формы;
- определяет шум, устойчив к выбросам.

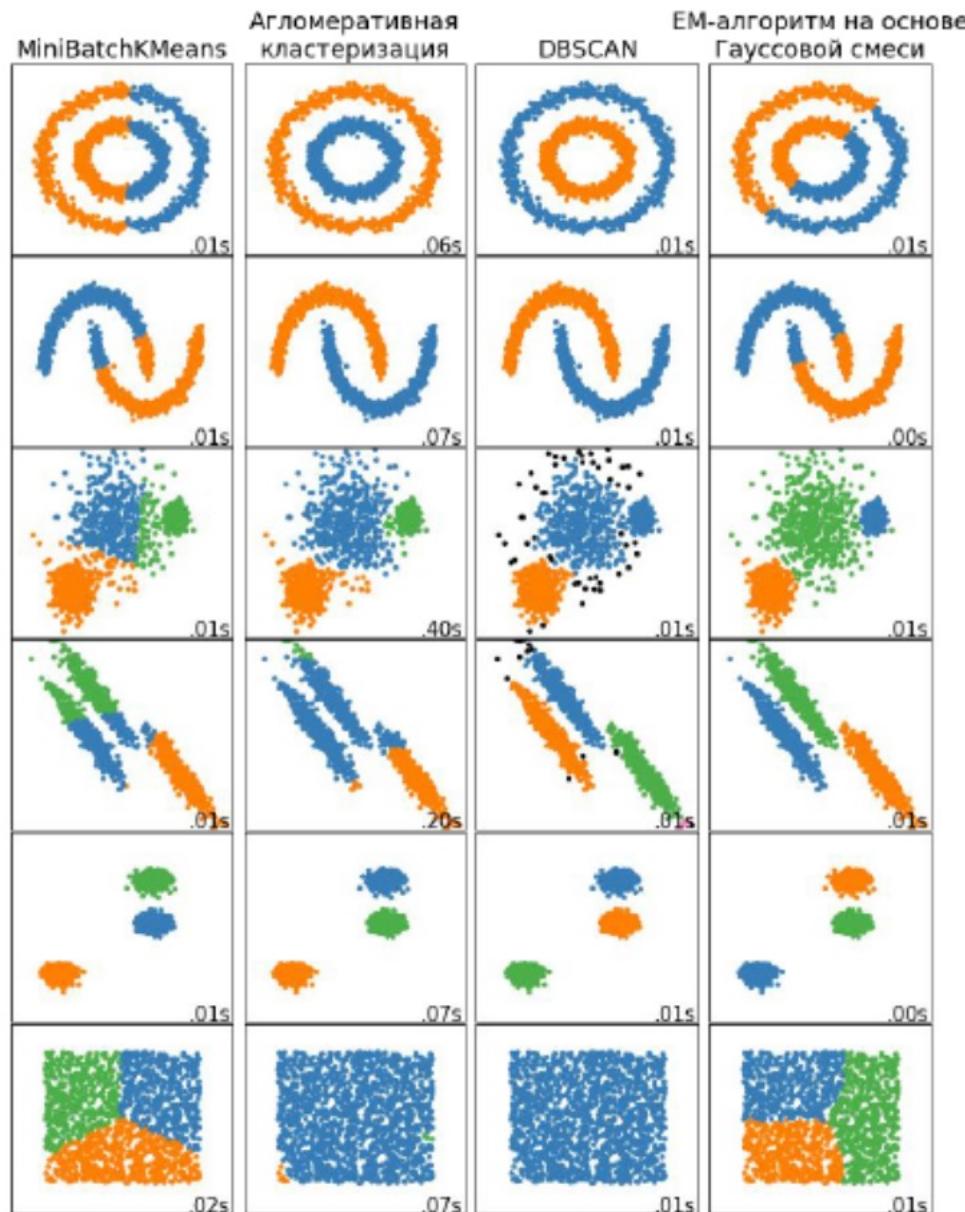
Недостатки алгоритма:

- не может выделять кластеры, имеющие разную плотность;
- результат зависит от используемой функции расстояния.

# DBSCAN vs K-Means

---





DBSCAN, Агломеративная кластеризация

DBSCAN, Агломеративная кластеризация

EM-алгоритм, Агломеративная кластеризация

EM-алгоритм, DBSCAN

Все алгоритмы нашли кластеры

DBSCAN

Метод	Параметры	Масштабируемость	Сценарий использования	Расстояние
K-Means	Число кластеров	Очень много объектов, среднее число кластеров	Выпуклые, соизмеримые кластеры	Евклидово
EM-алгоритм на основе Gaussian Mixture	Веса, векторы средних, матрицы ковариаций	—	Восстановление плотности, выпуклые кластеры	Обобщение Евклидова расстояния
Агglomerативная кластеризация	Число кластеров, связываемость, метрика	Много объектов, много кластеров	Много кластеров, нужно задавать функцию расстояния	Любое для Евклидова — Уорд
DBSCAN	Радиус окружности, число соседей	Много объектов, среднее число кластеров	Неравные, невыпуклые кластеры, выбросы	Евклидово

# Визуализация

---

# Визуализация

---

- Задача визуализации состоит в отображении объектов в 2х- или 3хмерное пространство с сохранением отношений между ними.

# MULTIDIMENSIONAL SCALING (MDS)

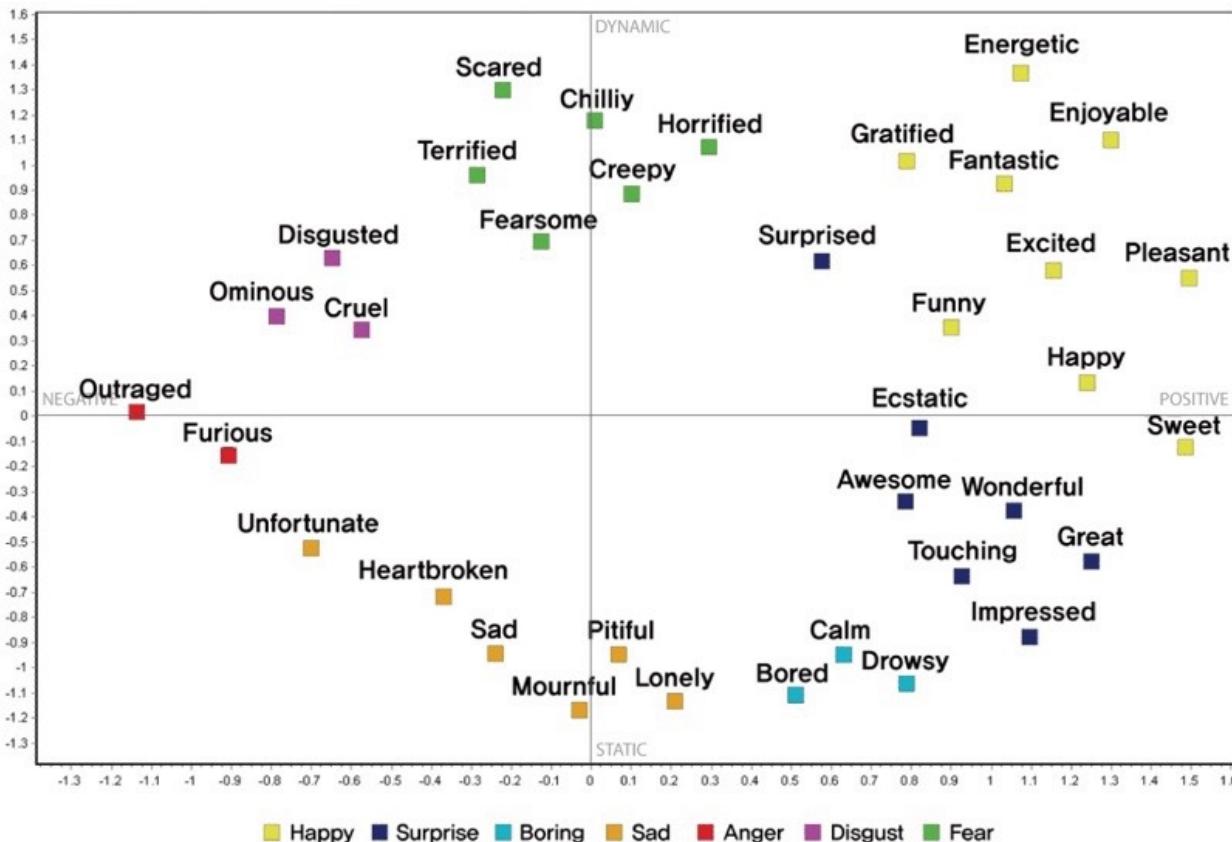
---

Идея метода – минимизация квадратов отклонений между исходными и новыми попарными расстояниями:

$$\sum_{i \neq j}^l (\rho(x_i, x_j) - \rho(z_i, z_j))^2 \rightarrow \min_{z_1, \dots, Z_l}$$

# MDS

---



# t-SNE

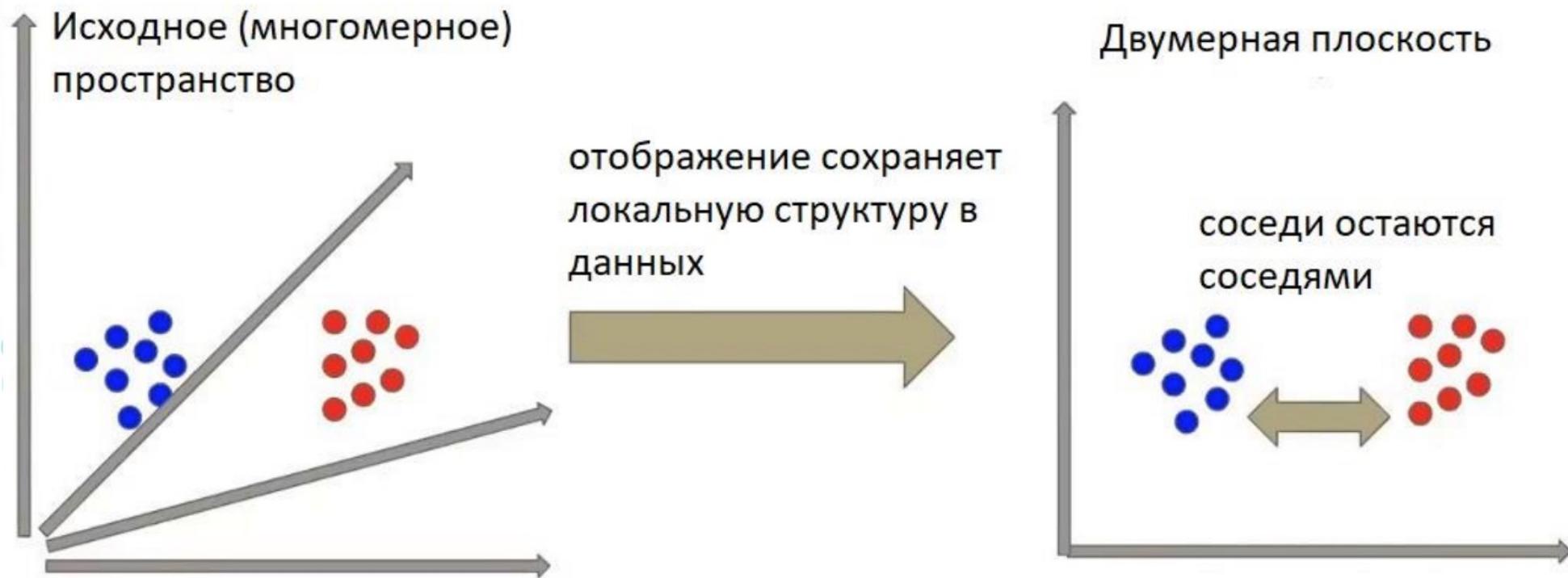
---

- t-SNE – t-distributed stochastic neighbor embedding
- При проекции нам важно не сохранение расстояний между объектами, а сохранение пропорций:

$$\rho(x_1, x_2) = \alpha \rho(x_1, x_3) \Rightarrow \rho(z_1, z_2) = \alpha \rho(z_1, z_3)$$

# t-SNE

---



# t-SNE

---

- Не метод многомерного шкалирования – полученные расстояния не будут соотноситься с исходными
- Пытаемся перенести «окрестность» точек из исходного пространства в пространство меньшей размерности

# t-SNE

---

Будем использовать нормальную плотность для измерения сходства объектов в исходном пространстве:

$$\rho(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Отнормируем эти близости так, чтобы получить вектор распределений расстояний от объекта  $x_j$  до всех остальных объектов:

$$p(i | j) = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_j^2)}{\sum_{k \neq j} \exp(-\|x_k - x_j\|^2/2\sigma_j^2)}$$

# t-SNE

---

$$p(i | j) = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_j^2)}{\sum_{k \neq j} \exp(-\|x_k - x_j\|^2 / 2\sigma_j^2)}$$

Вероятность встретить объект  $x_i$  при гауссовом распределении с центром в  $x_j$  и дисперсией  $\sigma_j^2$

# t-SNE

---

Данные величины не являются симметричными, что может добавить нам дополнительных сложностей при дальнейшей работе. Симметризуем их:

$$p_{ij} = \frac{p(i | j) + p(j | i)}{2\ell}.$$

# t-SNE

---

Схожесть в целевом пространстве:

$$q_{ij} = \frac{(1 + \|z_i - z_j\|^2)^{-1}}{\sum_{k \neq m} (1 + \|z_k - z_m\|^2)^{-1}}$$

# t-SNE

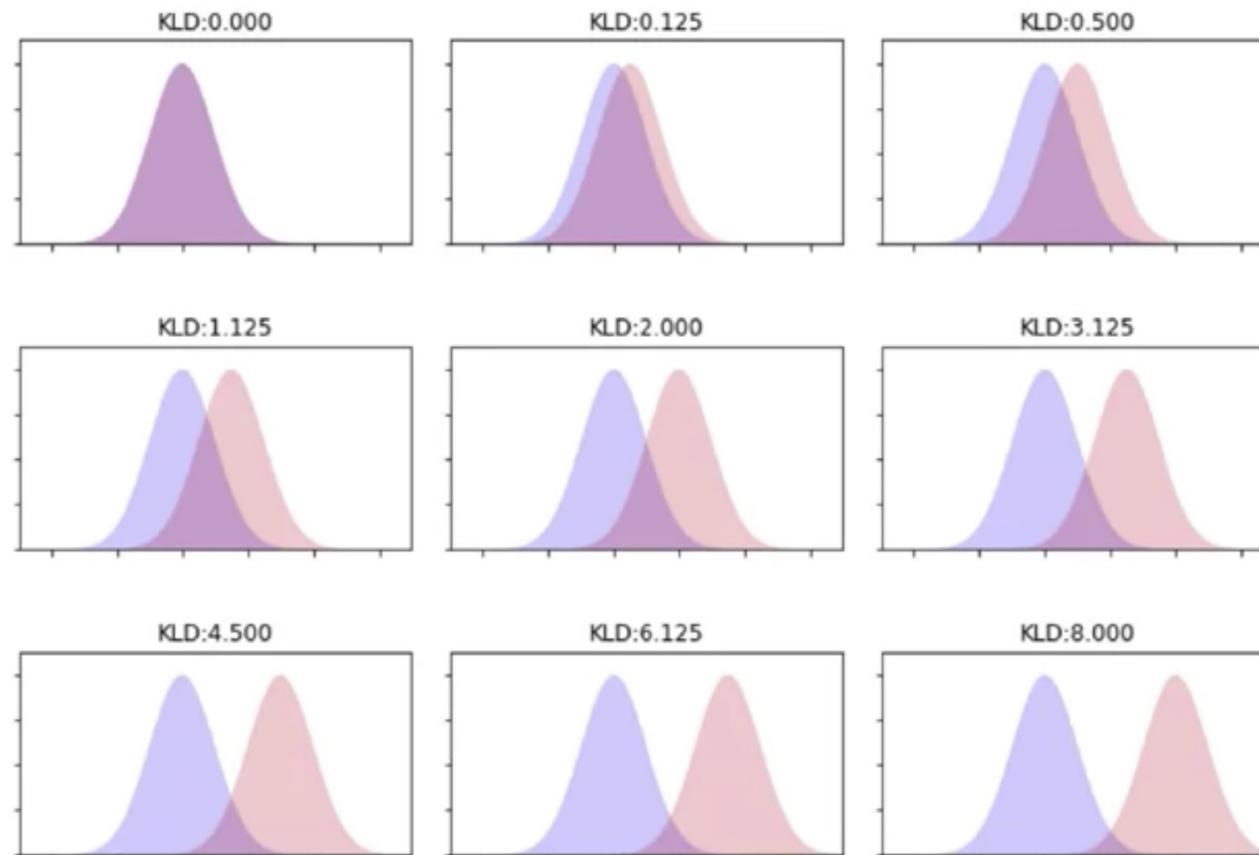
---

Будем измерять ошибку с помощью дивергенции Кульбака-Лейблера, которая часто используется для измерения расстояний между распределениями:

$$\text{KL}(p \parallel q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}} \rightarrow \min_{z_1, \dots, z_\ell}$$

# Дивергенция Кульбака-Лейблера

---



# Недостатки t-SNE

---

- Может быть нестабильным
- Размеры полученных кластеров могут ничего не значить
- Расстояния между кластерами могут ничего не значить
- Полностью шумовые данные могут выдать структуру