Лекция 4

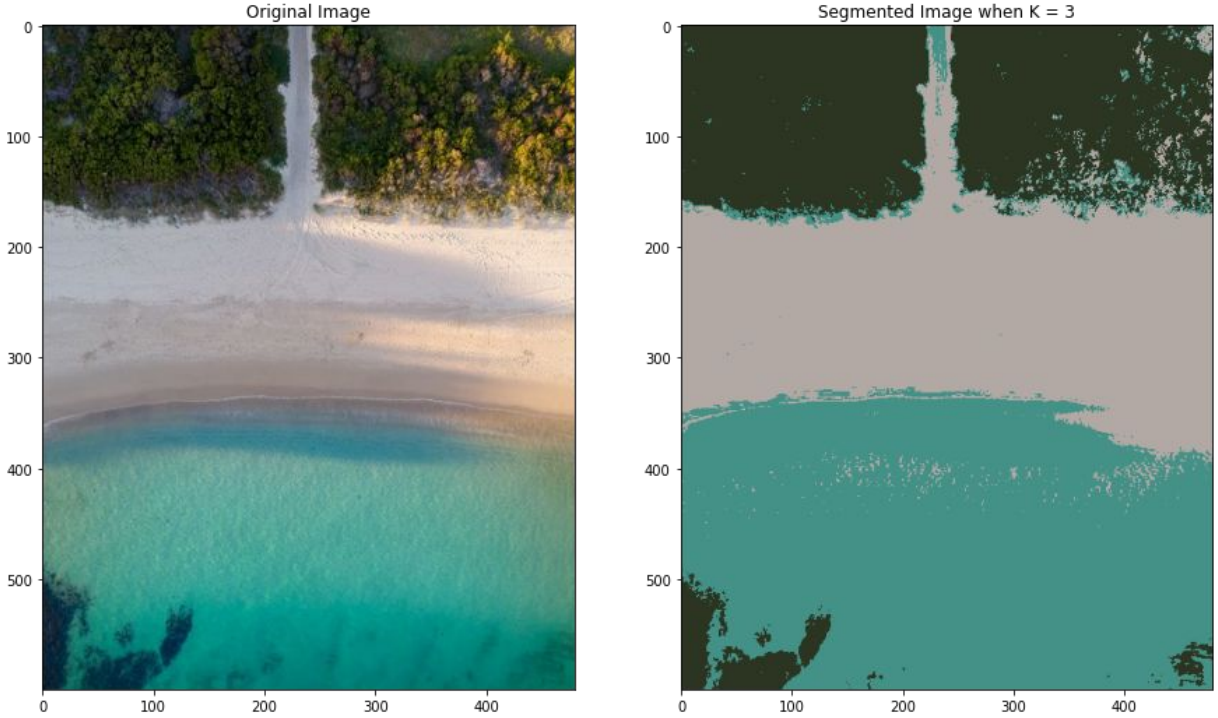
Кластеризация

Кластеризация

- Цель - группирование похожих наблюдений в кластеры
- Кластеризация является задачей обучения без учителя
- Кластеризация применяется для:
  - Анализа данных
  - Рекомендательных систем
  - Методика понижения размерности пространства
  - Обнаружения аномалий
  - Частичного обучения
  - Сегментирования изображения

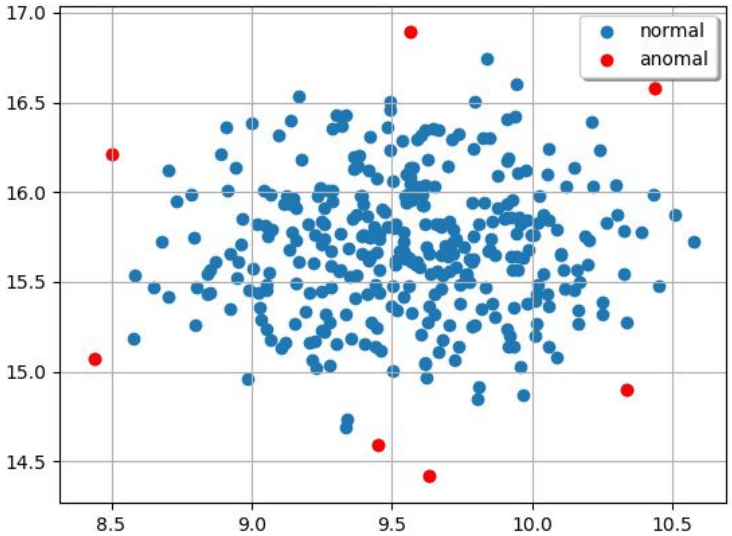
Кластеризация для сегментации изображения

- Сегментация - разбиение изображения на области “схожести”



Кластеризация для обнаружения аномалий

- Аномалии/выбросы - не всегда возможно определить заранее в силу размерности пространства или формы набора данных
- При кластеризации аномалии могут не попасть ни в один кластер или образовать кластеры сильно отличающийся от остальных



Кластеризация для понижения размерности

- После проведения кластеризации - есть возможность измерить “похожесть” каждого наблюдения с каждым кластером (в которые он не входит тоже).
- “похожесть” - мера того, насколько хорошо наблюдение подгоняется под кластер. То есть на наблюдения какого кластера и как сильно похоже.
- Каждый вектор наблюдение можно заменить вектором похожести к кластерам размерности k, где k количество кластеров.
- Чаще всего, размерность вектора похожести будет гораздо меньше размерности исходного пространства. И хоть характер данных меняется, но такой вектор способен предоставлять достаточно информации для последующей обработки.

Кластерный анализ

- Позволяет оценить весь набор данных на наличие ярко-выраженных групп наблюдений
- После разбиения данных на кластеры, каждый кластер можно анализировать по отдельности
- Из-за специфики данных или выбранного метода не гарантируется, что будет возможно разбить данные на “адекватные” кластеры

	Возраст	**З/п (тыс)**	**Кредит (тыс)**
Кластер 1	21.3	37.2	0.5
Кластер 2	28.2	62.5	4.7
Кластер 3	38.7	40.6	10.3

K-средних (K-means)

K-means

- Один из популярнейших и простых алгоритмов кластеризации
- В алгоритме кластер задается центром всех точек в кластере, называемом центроидом.
- Количество кластеров является гиперпараметром алгоритма
- Задача алгоритма - найти такие центроиды, чтобы минимизировать инерцию по всем кластерам
- Инерция - метрика, сумма квадратов отклонения всех точек в кластере до центра кластера

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

где k - кол-ва кластеров, S_i - i-й кластер, μ_i - центроид i-го кластера

Алгоритм K-means

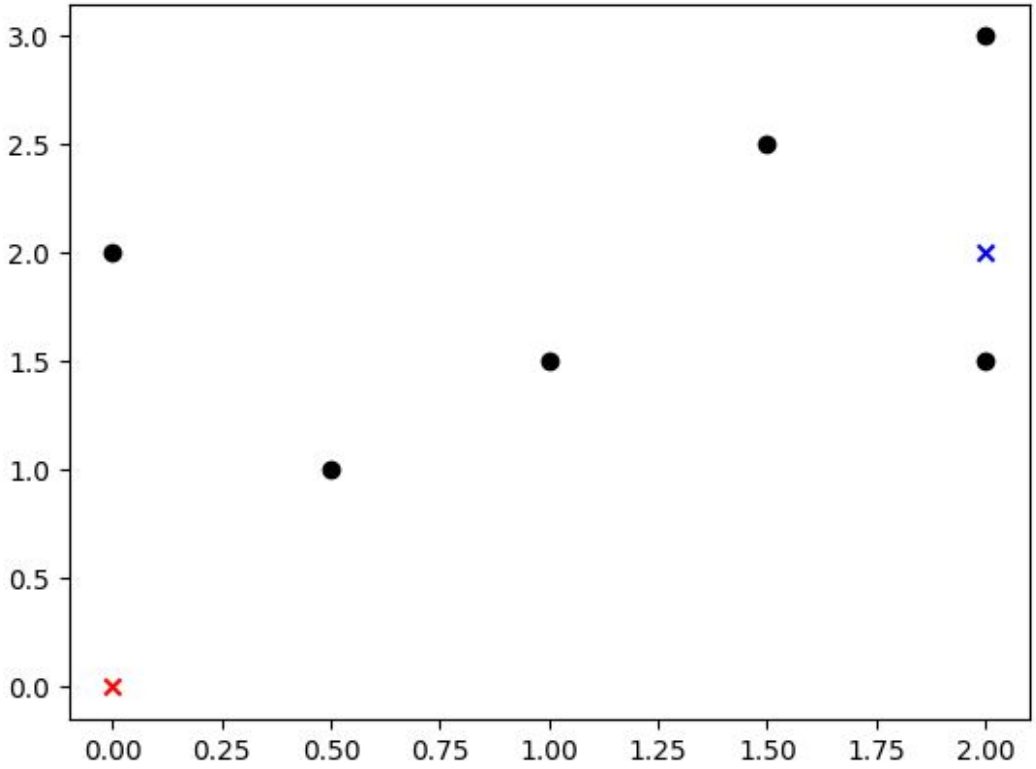
1. Выбрать случайным образом начальные центроиды кластеров. Самый простой способ, выбрать центроиды из имеющихся точек в данных.
2. Сформировать кластеры точек. Точка относится к тому кластеру, расстояние до центра которого минимально.
3. Рассчитать новые центроиды. Центроид рассчитывается как среднее значение по всем признакам всех точек в кластере.
4. Если центроиды сместились меньше порога или достигнуто максимальное количество итераций, то завершение алгоритма. Иначе, повторяется шаг 2.

Алгоритм K-means (шаг 1)

Выбор случайных центроидов

Точки	
0.5	1
1	1.5
2	1.5
2	3
1.5	1.5
0	2

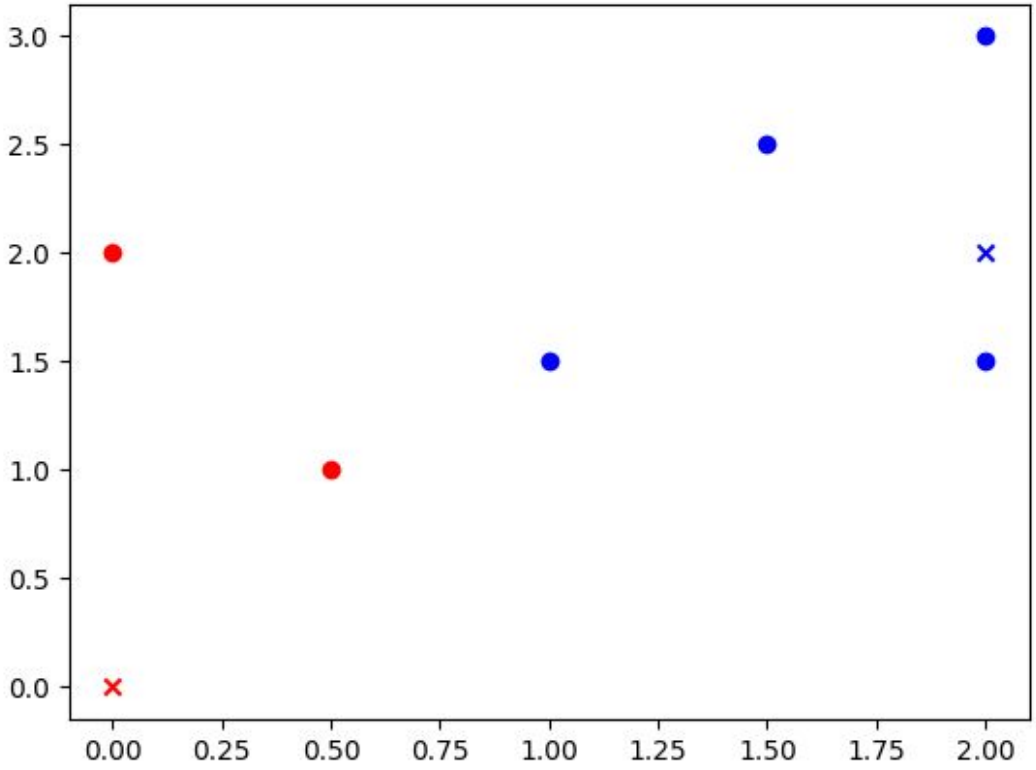
Центроиды	
0	0
2	2



Алгоритм K-means (шаг 2)

Формирование кластеров

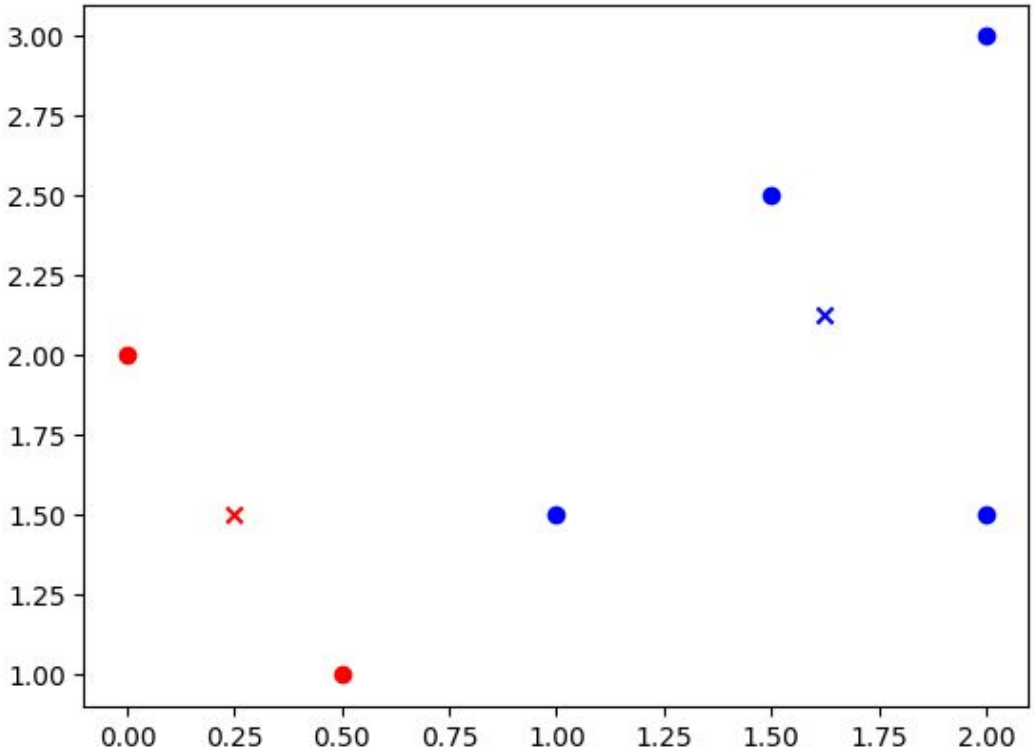
Расстояние	
K1	K2
1.25	3.25
3.25	1.25
6.25	0.25
13	1
8.5	0.5
4	4



Алгоритм K-means (шаг 3)

Перерасчет центроидов

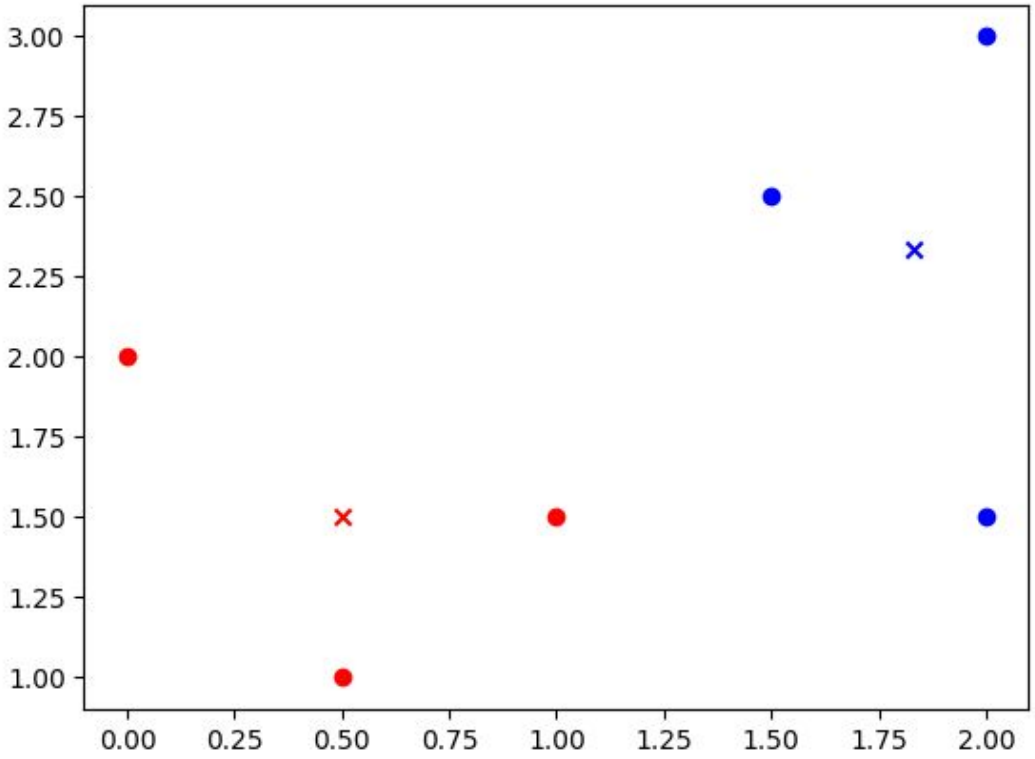
Центроиды	
0.25	1.5
1.625	2.125



Алгоритм K-means (шаг 4 [2 и 3])

Расчет новых кластеров

Расстояние	
K1	K2
0.3125	2.531
0.5625	0.781
3.0625	0.5312
5.312	0.906
2.5625	0.1562
0.312	2.656



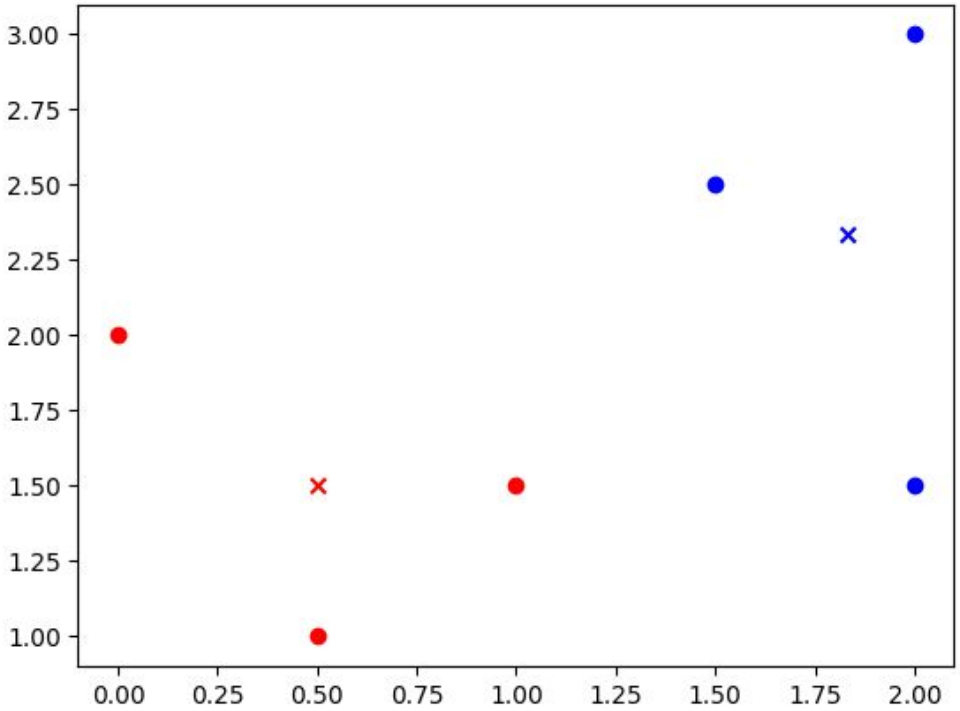
K-means в SKLearn

- Вызывается как `sklearn.cluster.KMeans`
- Параметры:
  - `n_clusters` - количество кластеров
  - `init` - способ инициализации центроидов: ‘random’ - случайным образом, ‘k-means++’ - специальный алгоритм выбора, задать значения вручную
  - `n_init (default = 10)` - количество прогонов алгоритма
  - `max_iter (default = 300)` - количество итераций алгоритма
  - `tol (default = 1e-4)` - порог для смещения центроид

- Атрибуты:
  - `cluster_centers_` - центроиды
  - `inertia_` - инерция. Метод `score` возвращает отрицательную инерцию

K-means в SKLearn - пример

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters = 2,
                init = [[0,0],[2,2]],
                n_init = 1)
clust_index = kmeans.fit_predict(data)
centers = kmeans.cluster_centers_
kmeans.inertia_#2.3333
```

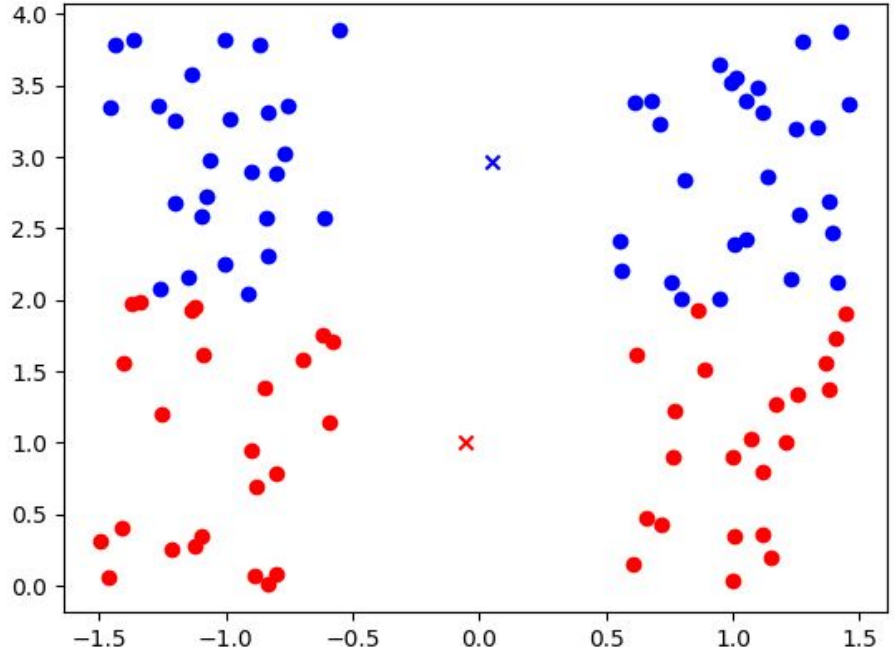


Недостатки K-means

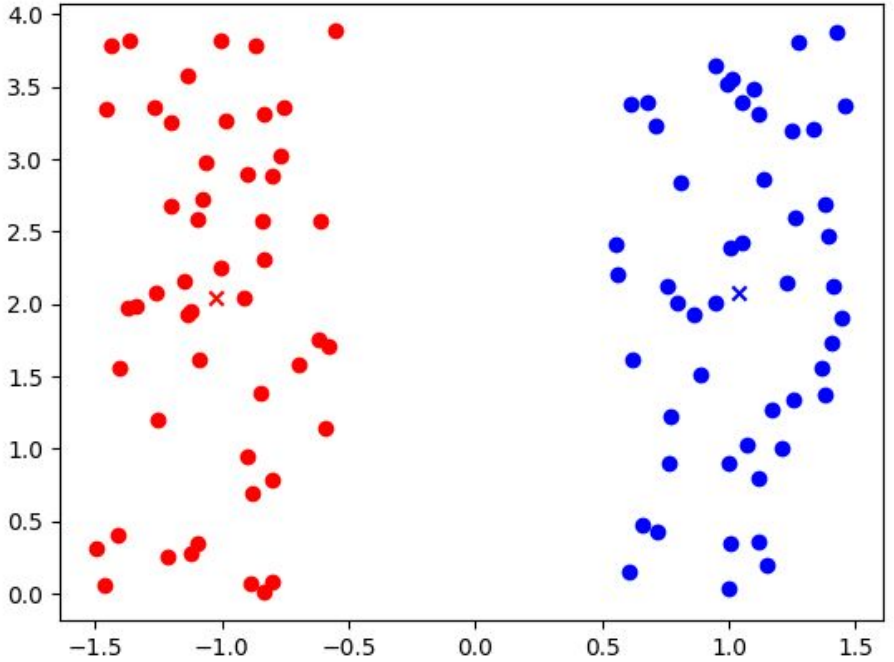
- Инерция - не нормализованная метрика
- Предполагается, что кластеры выпуклые и изотропные.
- Плохая устойчивость к вытянутым кластерам.
- Результат сильно зависит от начальной инициализации.
- Необходимо подбирать количество кластеров

Недостатки - пример

Начальные центроиды: (0,0) , (0,4)
Финальная инерция: 150.955



Начальные центроиды: (-1.5,2) , (1.5,2)
Финальная инерция: 140.252



Выбор центроидов K-means

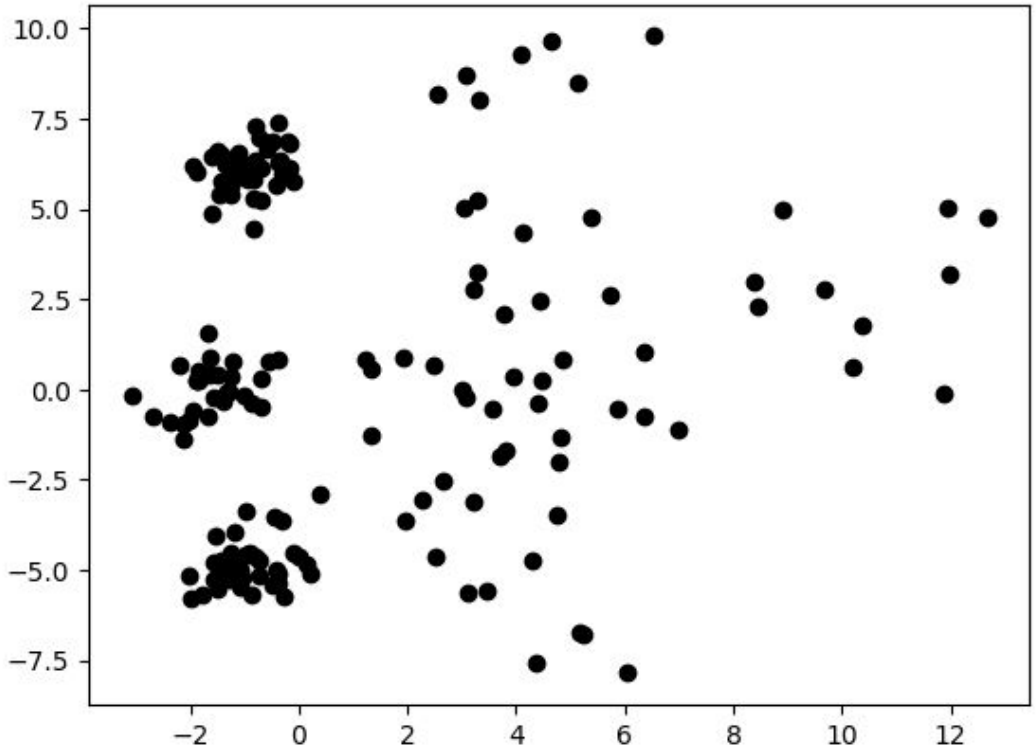
- Задать вручную - подходит в задачах, где данные простые и их структура явная
- `'random'` - алгоритм выполняется `n_init` раз. Каждый раз выбираются новые случайные центроиды. В результате выбираются центроиды, дающие минимальную инерцию.
- `'k-means++'` - специальный алгоритм выбора центроид. Сначала выбирается случайно один центроид $\mathbf{c}^{(1)}$из точек в наборе данных. Последующая центроида $\mathbf{c}^{(i)}$ выбирается, рассматривая образец $\mathbf{x}^{(i)}$ с вероятностью:

$$D(x^{(i)})^2 / \sum_{j=1}^{m} D(x^{(j)})^2$$

Выбор количества кластеров

Не всегда очевидно, сколько кластеров необходимо выбрать.

Задача выбора кластера легкая, если данные можно визуализировать, и явно прослеживаются группы точек.



Выбор количества кластеров (выбрали 3 кластера)

Визуально видно, что 3 кластера недостаточно.

Полученная инерция: *1636.733*

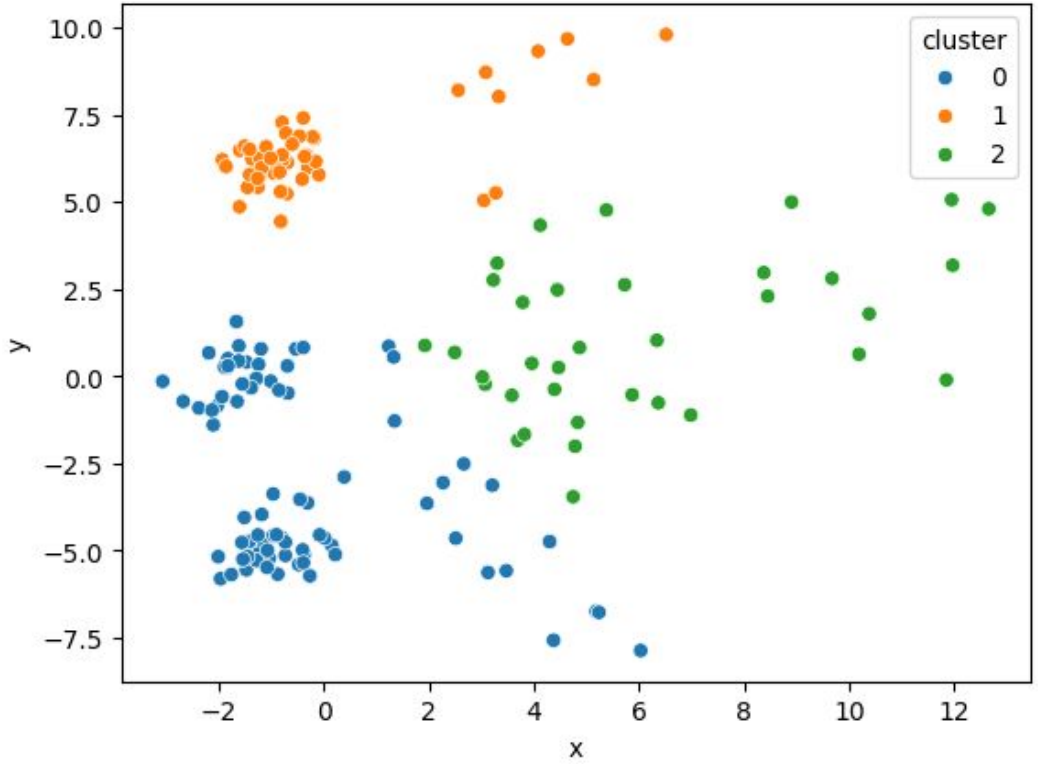
Центроиды:

[-0.4459258 , -3.03367569],

[-0.02980045, 6.46235075],

[6.1054111 , 1.16594025]

Что делать, если данные не визуализировать?

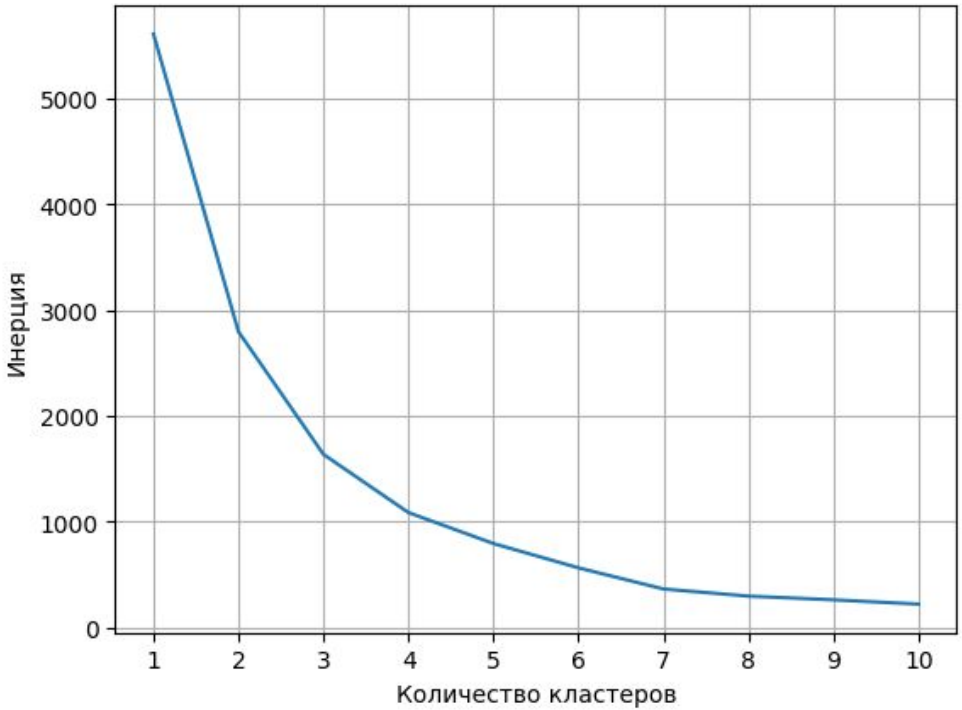


Метод локтя

Необходимо построить график зависимости инерции (либо другой метрики качества) от количества кластеров.

По графику выбрать точку, которая напоминает “локоть”, то есть такую точку, где скорость убывания графика уменьшается.

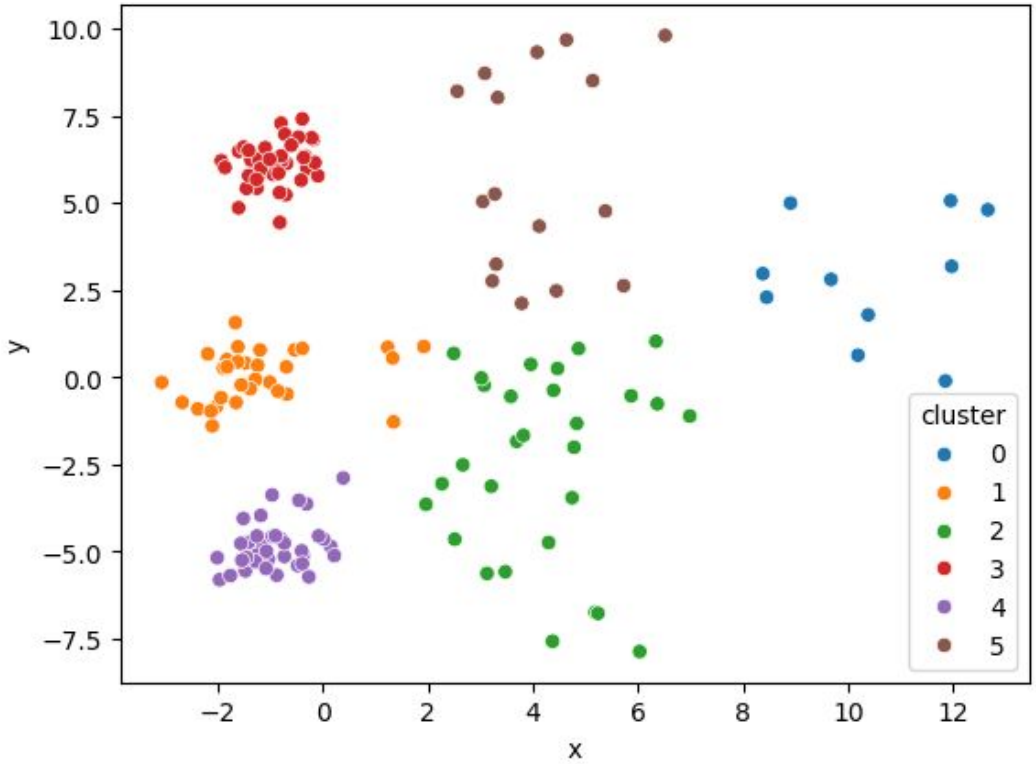
Для примера, эта точка находится в интервале [5-7]



Выбор количества кластеров (выбрали 6 кластеров)

Визуально видно, что для 6 кластеров, явно были выделены основные группы

Полученная инерция: *563.39*



Коэффициент силуэта

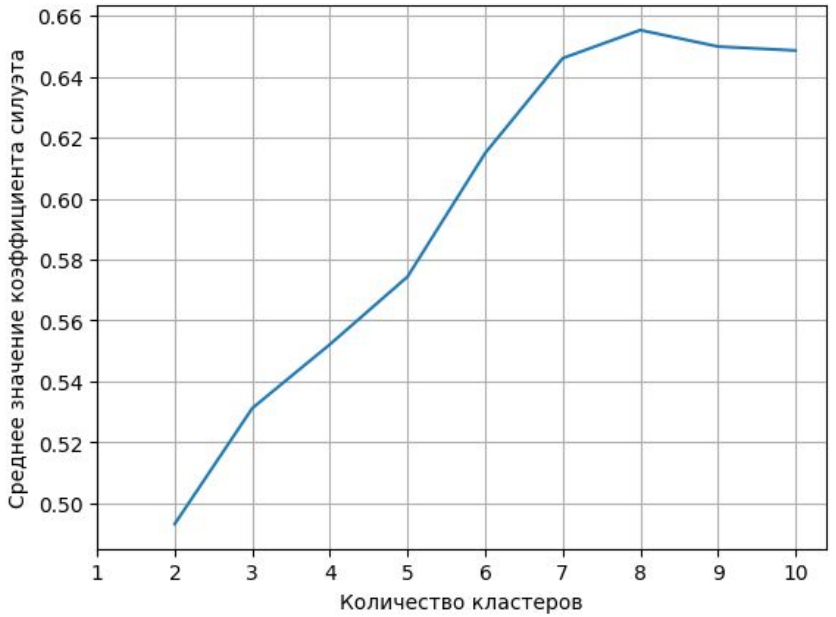
Можно рассчитать среднее внутриклассовое расстояние для i-го наблюдения в кластере: $a_i = \frac{1}{|S_I| - 1} \sum_{j \in S_I, i \neq j} d(i, j)$

Далее рассчитать ближайшее расстояние до точки из другого класса: $b_i = \min_{I \neq J} \frac{1}{|S_J|} \sum_{j \in S_j} d(i, j)$

Коэффициент силуэта: $\frac{b_i - a_i}{\max(a_i, b_i)}$

Коэффициент принимает значения от -1 до 1, и равен 1 если a_i гораздо меньше b_i, то есть в случае, когда наблюдение максимально подходит под кластер в котором находится.

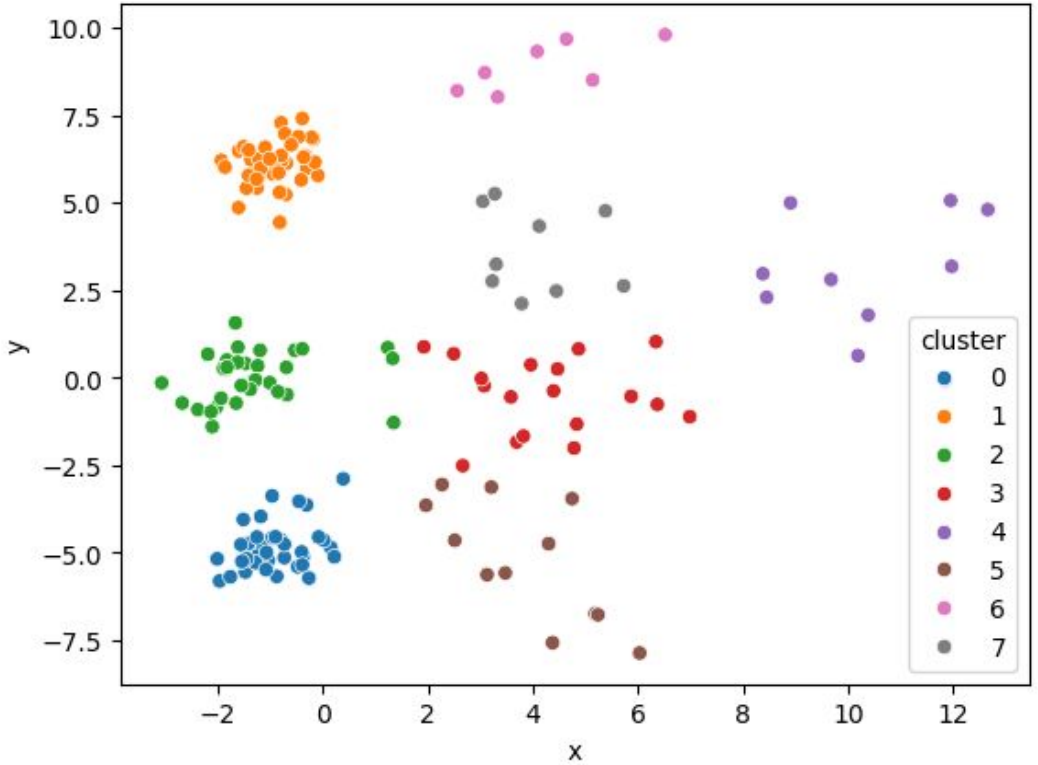
В sklearn можно рассчитать используя ***`sklearn.metrics.silhouette_score`***



Выбор количества кластеров (выбрали 8 кластеров)

При 8 кластерах все еще наблюдается слияние кластеров (зеленый, красный, коричневый)

Полученная инерция: 296.424



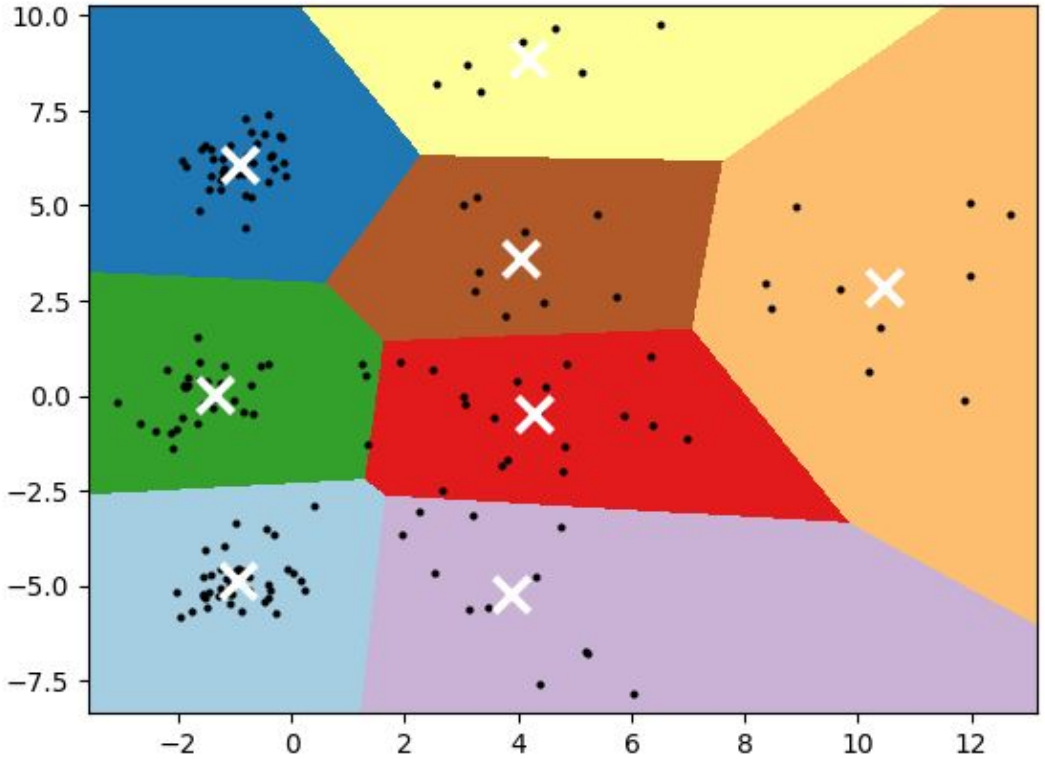
Визуализация K-mean через диаграмму Вороного

Диаграмма Вороного представляет разбиение плоскости таким образом, что в образуются множества с близкими точками к одному из элементов этого множества.

Если взять центроиды кластеров, то получатся области, в которых точки будут отнесены к ближайшему центроиду.

Для получения областей, необходимо построить сетку и кластеризовать каждую точку.

Построение диаграммы Вороного есть в модули ScyPi (scipy.spatial.Voronoi)



Улучшение K-means

- Так как на каждой итерации алгоритма происходит расчет расстояний для каждой точки и смещение центроидов, то при их большом количестве, алгоритм производит большое кол-во операций.
- Так как предполагается, что точки находятся в ограниченном пространстве, то для количества точек > 10000, предлагается использовать пакетные вычисления.
- Пакетное вычисление заключается в том, чтобы на каждой итерации брать случайную подвыборку из всех точек, и только на их основании рассчитывать новые центроиды.
- В SKlearn пакетный K-means реализован в виде `MiniBatchKMeans`. Для того, чтобы задать размер пакета, необходимо устанавливать параметр `batch_size`, который по умолчанию равен 1024.

Примечание: пакетные вычисления в машинном обучении, такие вычисления, когда используется только подвыборка из всей обучающей выборки.

Внешние метрики для оценки кластеризации

- В случае, если были известны метки для точек, то можно рассчитать следующие метрики:
  - Однородность (`homogeneity_score`) - показывает степень того (от 0 до 1), что весь кластер состоит из точек одного класса.
  - Полнота (`completeness_score`) - показывает степень того (от 0 до 1), что все точки класса были определены в один кластер.
  - V-мера (`v_measure_score`) - комбинация однородности и полноты.
    По умолчанию β = 1. Параметр β контролирует что сильнее влияет. При β < 1 влияет больше гомогенность, при β > 1 больше полнота.

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