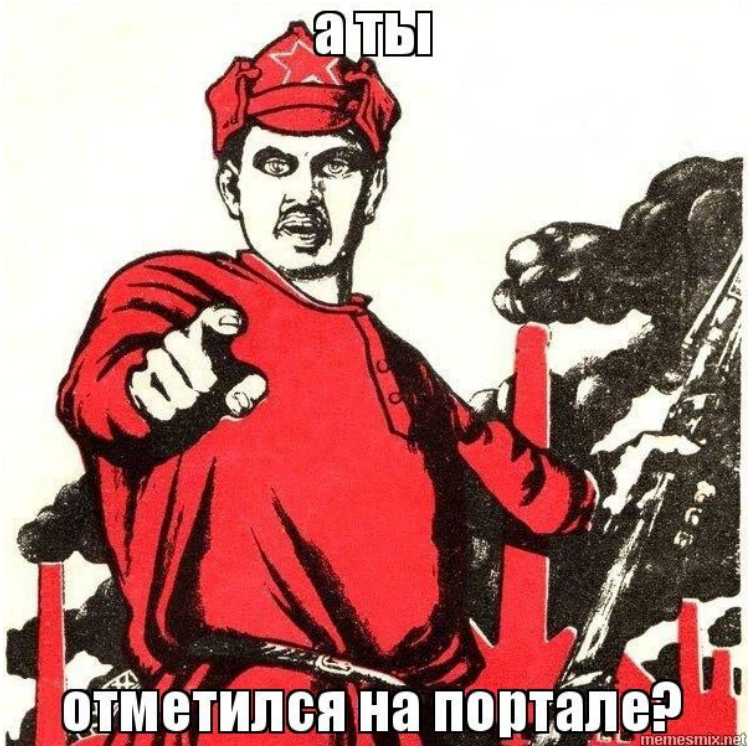
Поиск ближайших соседей. Кластеризация.

Отметься на портале!

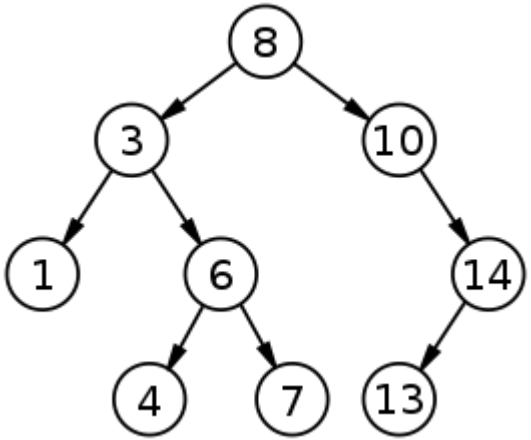


План занятия

- Деревья поиска (бинарные, AVL)
- KD-tree и ANN (Annoy, FAISS)
- Кластеризация (K-means, DBScan)
- Кластеризация на графах
- Union-Find

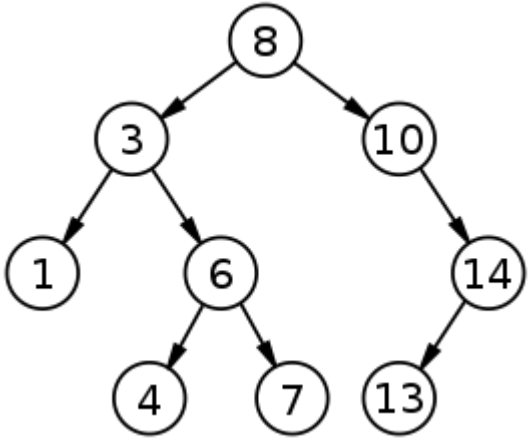
Двоичное дерево

Структура данных в которой каждый узел имеет не более 2х потомков



Двоичное дерево поиска

- у каждого узла есть ключ
- оба поддерева - двоичные деревья поиска
- у всех узлов левого поддерева некоторого узла значения ключей меньше, чем значение ключа у самого узла
- у всех узлов правого поддерева некоторого узла значения ключей не меньше, чем значение ключа у самого узла

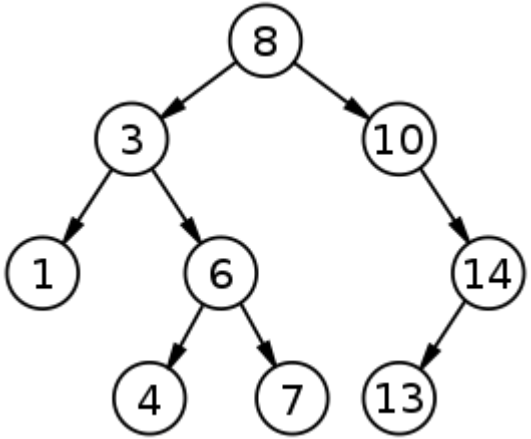


Используется для работы с упорядоченными множествами

Двоичное дерево поиска

Свойства:

- поиск элемента по ключу (O (log n))
- вставка (O (log n))
- удаление узла (O (log n))

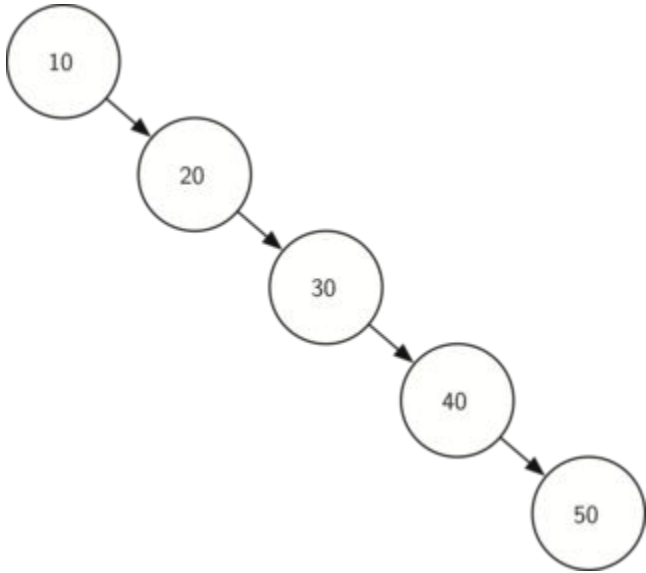


Двоичное дерево поиска

Какие проблемы?

- если добавление в новое дерево вырождено - т.е. добавляются элементы по возрастанию - то сложность поиска в таком дереве O(n)

Решается - балансировкой

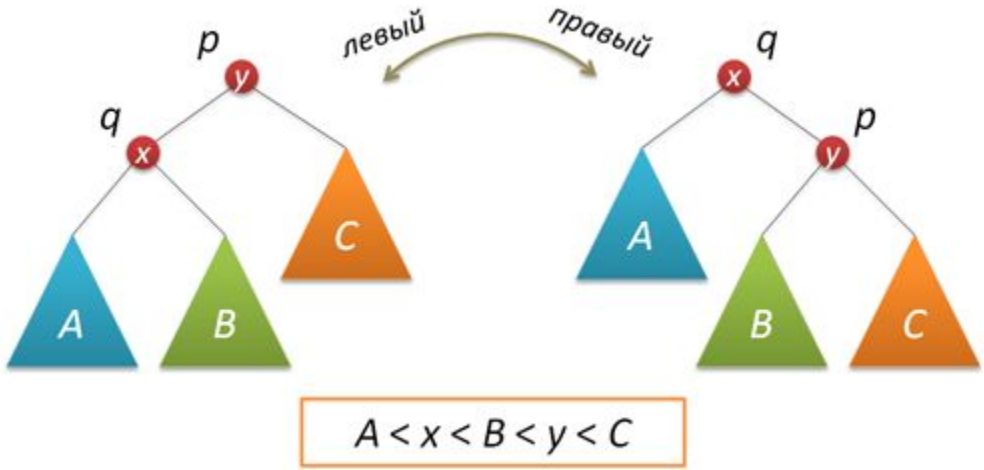


Двоичное дерево поиска

Какие проблемы?

- если добавление в новое дерево вырождено - т.е. добавляются элементы по возрастанию - то сложность поиска в таком дереве O(n)

Решается - балансировкой (поворот)



AVL дерево

сбалансированное по высоте двоичное дерево поиска:
для каждого узла высота поддеревьев различается не более чем на 1.

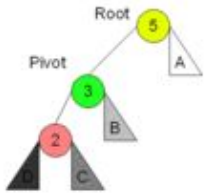
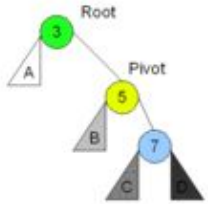
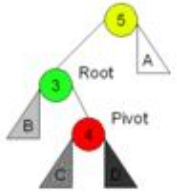
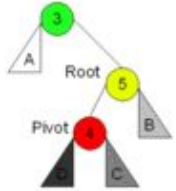
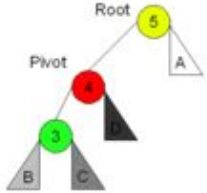
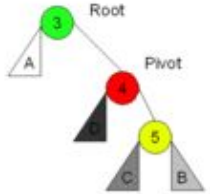
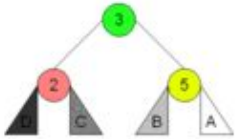
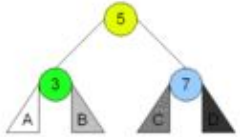
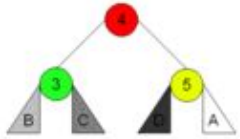
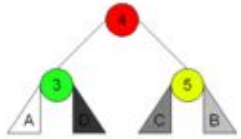
тутор по AVL

немного о красно-черных деревьях

AVL

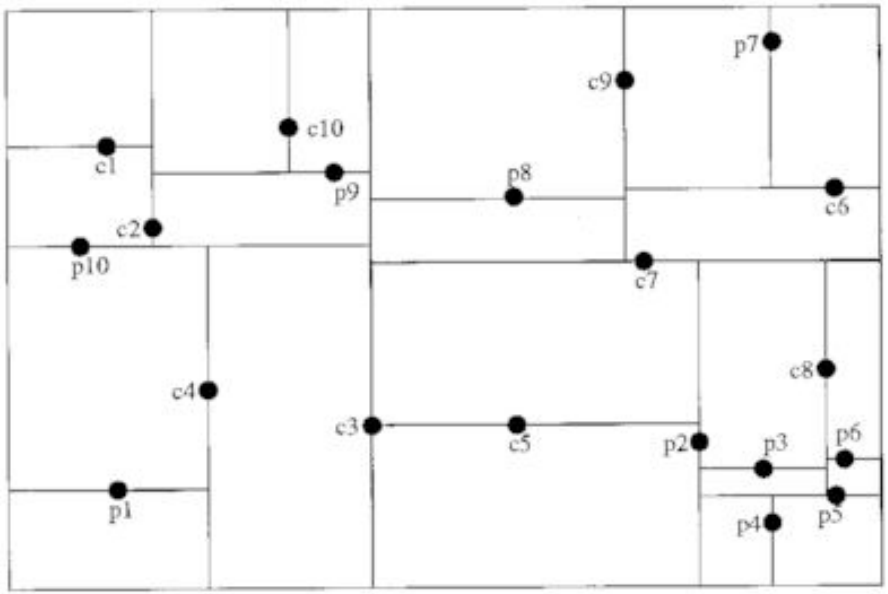
There are 4 cases in all, choosing which one is made by seeing the direction of the first 2 nodes from the unbalanced node to the newly inserted node and matching them to the top most row.

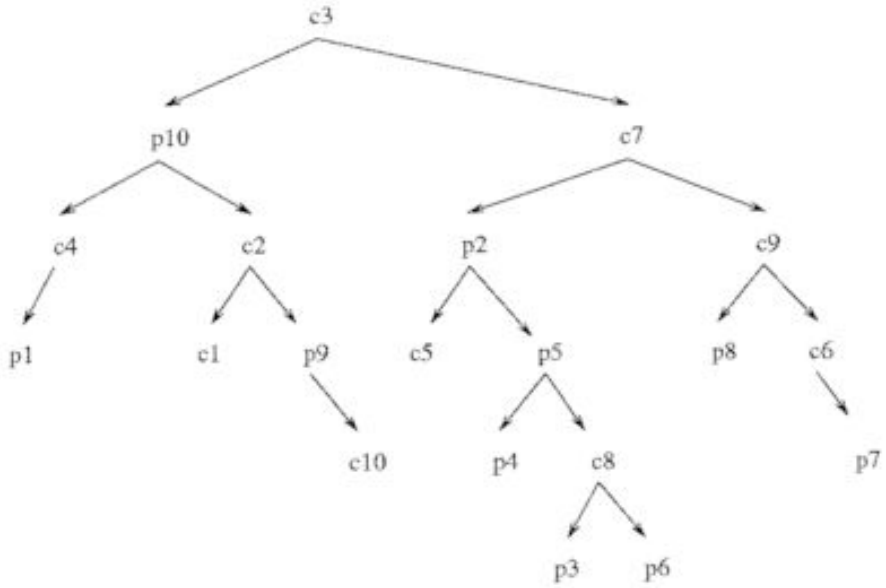
Root is the initial parent before a rotation and **Pivot** is the child to take the root's place.

Left Left Case	Right Right Case	Left Right Case	Right Left Case
			
Right Rotation	**Left Rotation**	**Left Rotation**	**Right Rotation**
			
		Right Rotation	**Left Rotation**
			

К-мерное дерево (kd-tree)

Структура данных с разбиением пространства для упорядочивания точек в к-мерном пространстве.

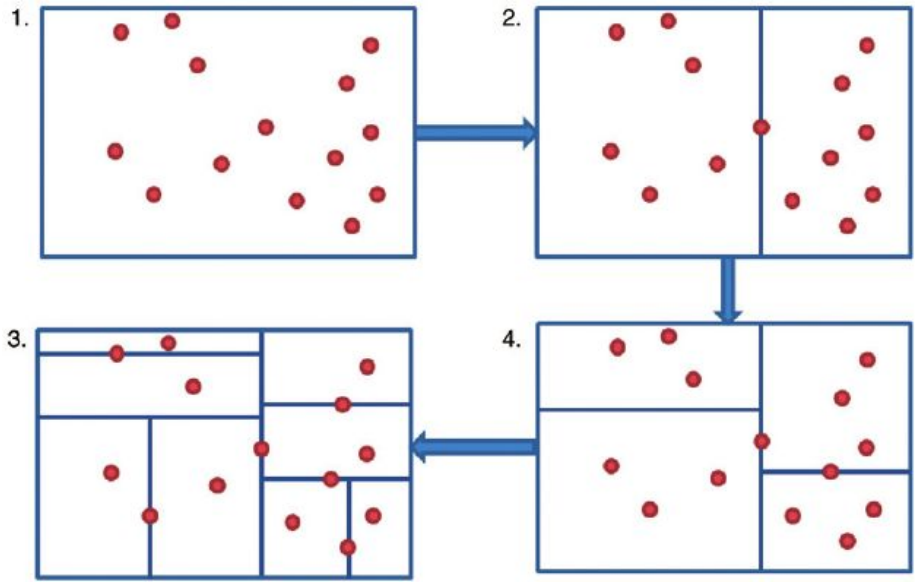




К-мерное дерево (kd-tree)

Построение

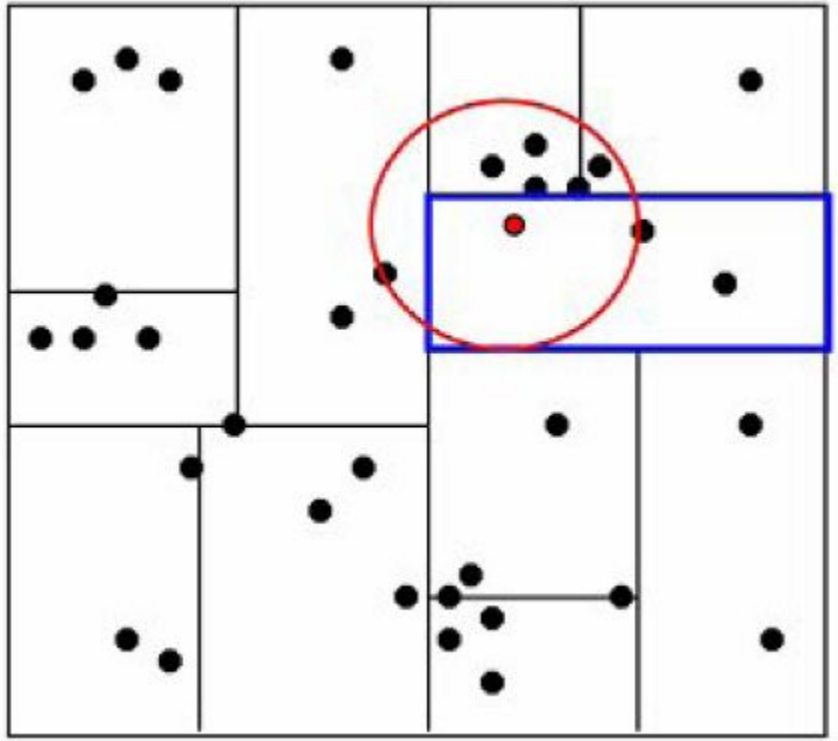
- случайно выбираем признак
- считаем значение медианы и делим по ней



К-мерное дерево (kd-tree)

Поиск в диапазоне

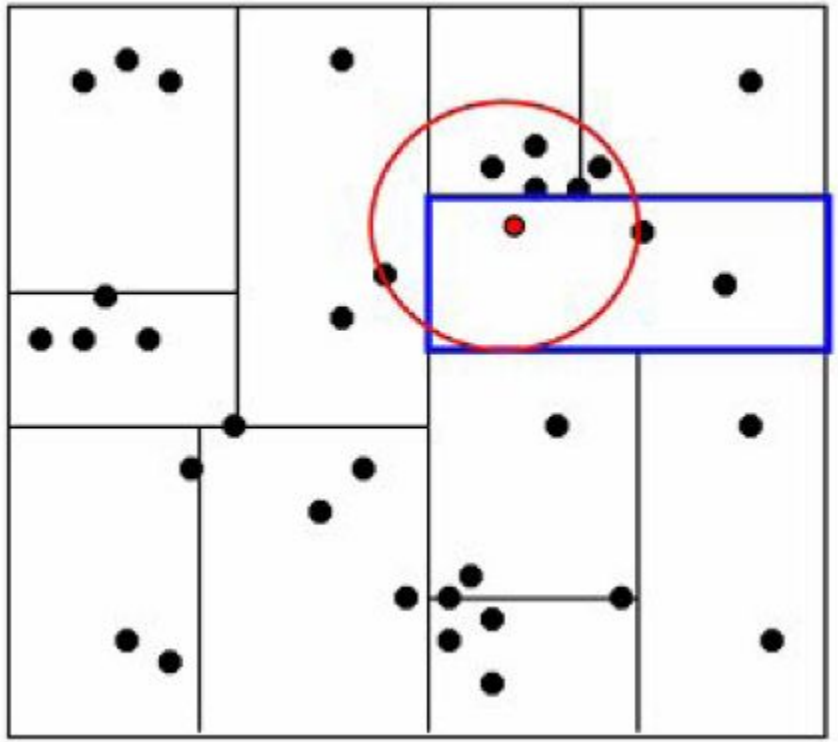
- сравниваем медиану в узле и значение диапазона
- на основе сравнения выбираем поддеревья куда спускаться



К-мерное дерево (kd-tree)

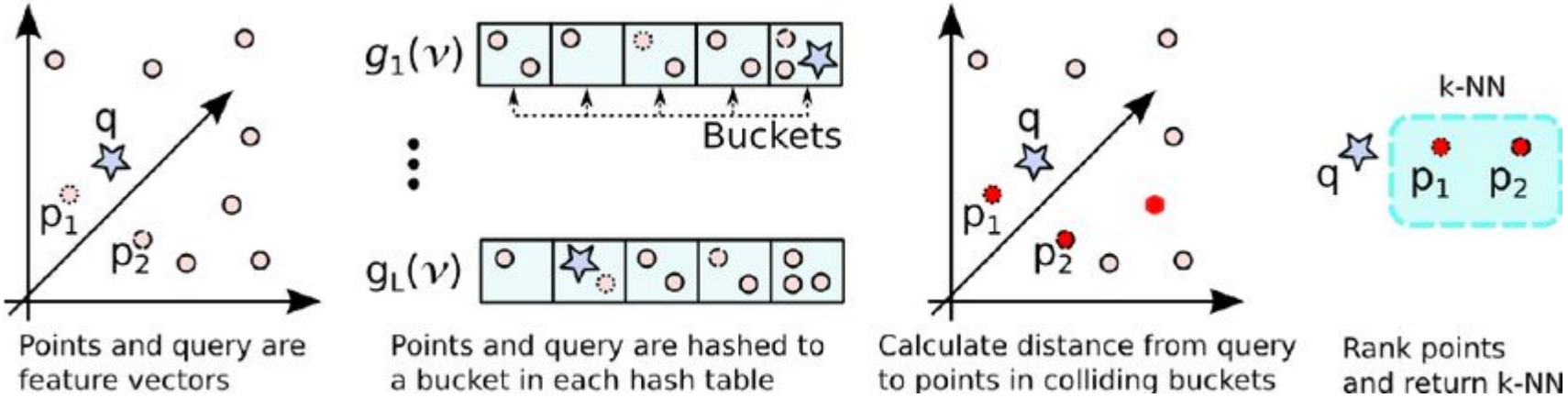
Поиск соседей

- определяем потенциального соседа и смотрим расстояние до него
- запускаем из корня поиск по диапазону



Поиск приближенных соседей

LSH (Locality-sensitive hashing) - идея: подберем такое семейство хеш-функций чтобы похожие объекты с бОльшей вероятностью попадали в один “бакет”, а перебирать в “бакете” - дешевле.



MinHash, SimHash

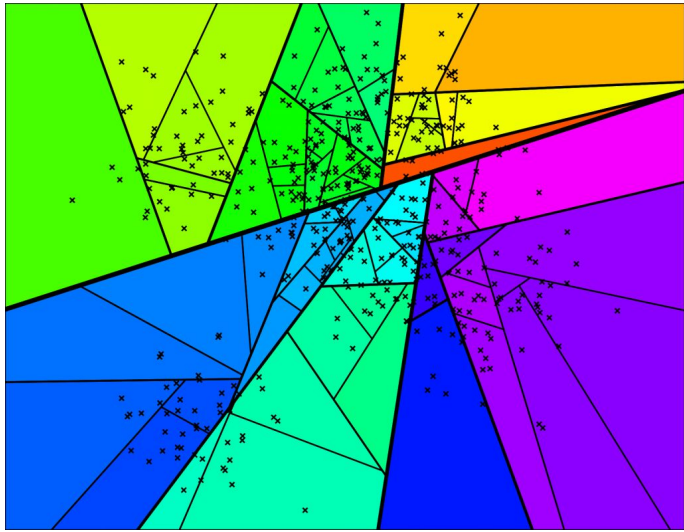
https://habr.com/company/mailru/blog/338360/ - обзор методов ANN

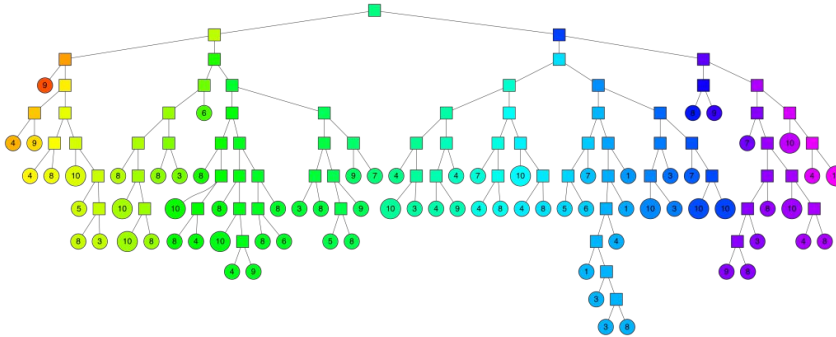
Approximate Nearest Neighbors Oh Yeah

Идея:

если точки близки в пространстве, то вероятно они будут близки в дереве

- берем две точки
- разделяем гиперплоскостью
- строим дерево
  (узлы - гиперплоскость, листья - точки)
- строим так несколько деревьев





описание в блоге автора

Approximate Nearest Neighbors Oh Yeah

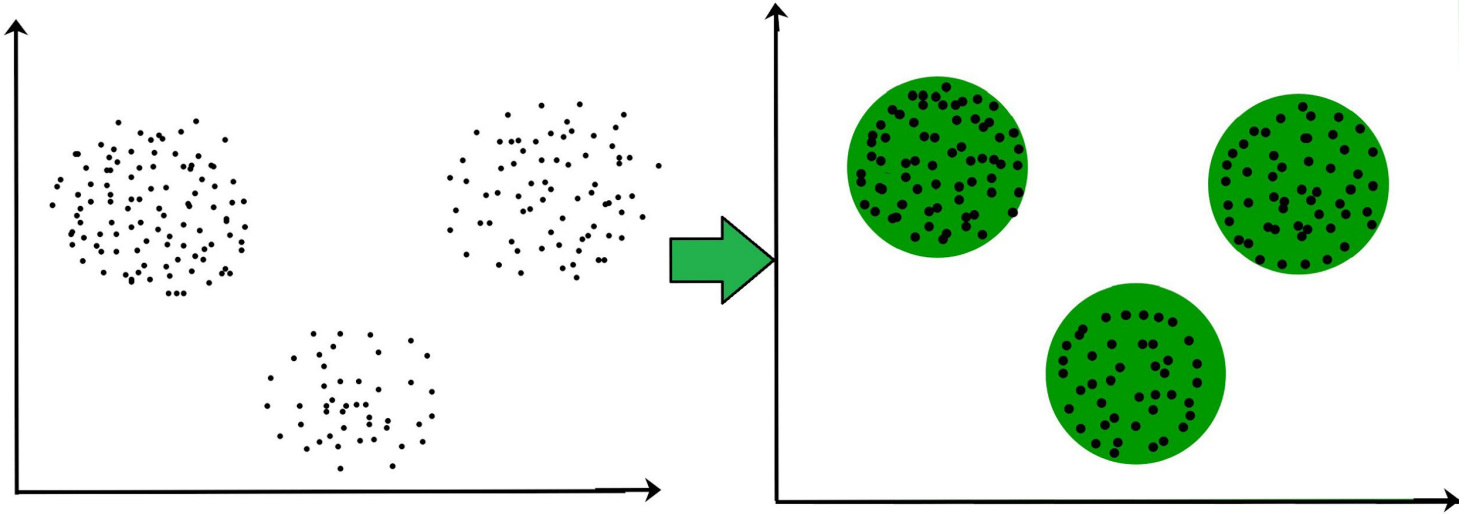
Поиск

- набираем потенциальных соседей из узлов спускаясь по M деревьям (исследуя “обе стороны сплита”)
  - спустились в узел - взяли всех как кандидатов
  - пошли в соседний - взяли и его кандидатов
- получили небольшое подмножество элементов в котором уже не так дорого считать расстояние до всех
- отсортировали по расстоянию и вернули топ_К ближайших

Кластеризация

Есть обучающая выборка x1, x2, … , xn.
Требуется расставить метки y1, .., yn таким образом, чтобы похожие друг на друга объекты имели одинаковую метку, а непохожие - разную.
Т.е. необходимо разбить объекты на несколько групп.

Кластеризация



Кластеризация

Метрики.

Среднее расстояние внутри кластера.

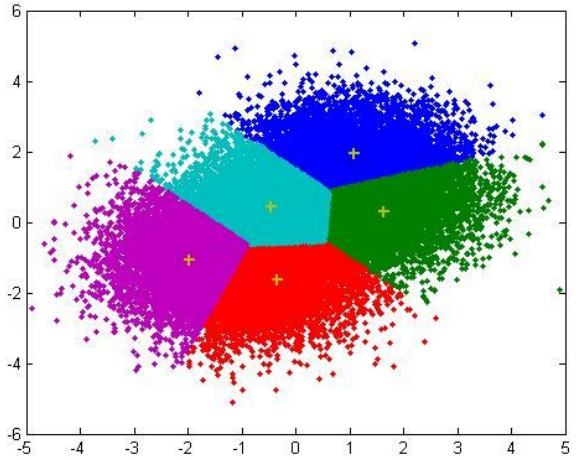
$$\frac{\sum_{i<j}[y_i = y_j]\rho(x_i, x_j)}{\sum_{i<j}[y_i = y_j]} \quad \to \quad \min$$

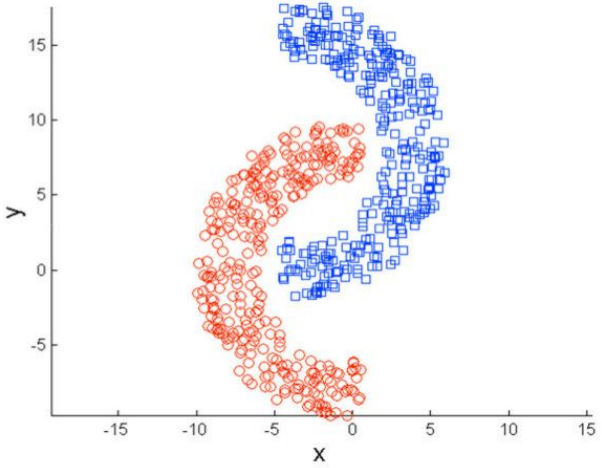
Среднее расстояние между кластерами.

$$\frac{\sum_{i<j}[y_i \neq y_j]\rho(x_i, x_j)}{\sum_{i<j}[y_i \neq y_j]} \quad \to \quad \max$$

Кластеризация

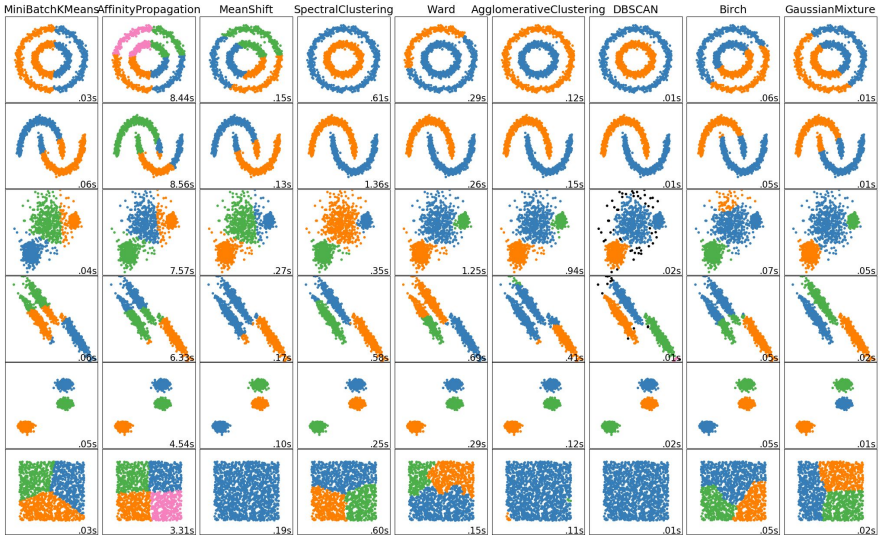
- разная форма кластеров
- наличие иерархии в данных
- разный размер кластеров

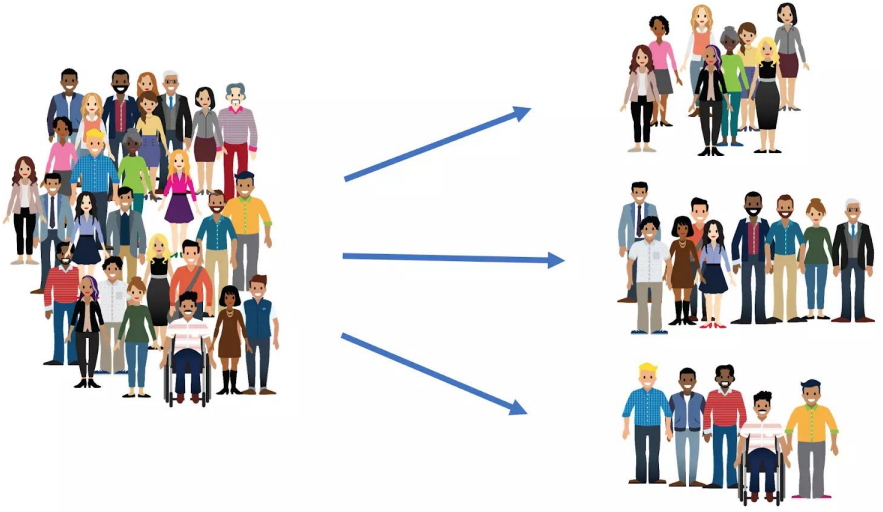




Кластеризация

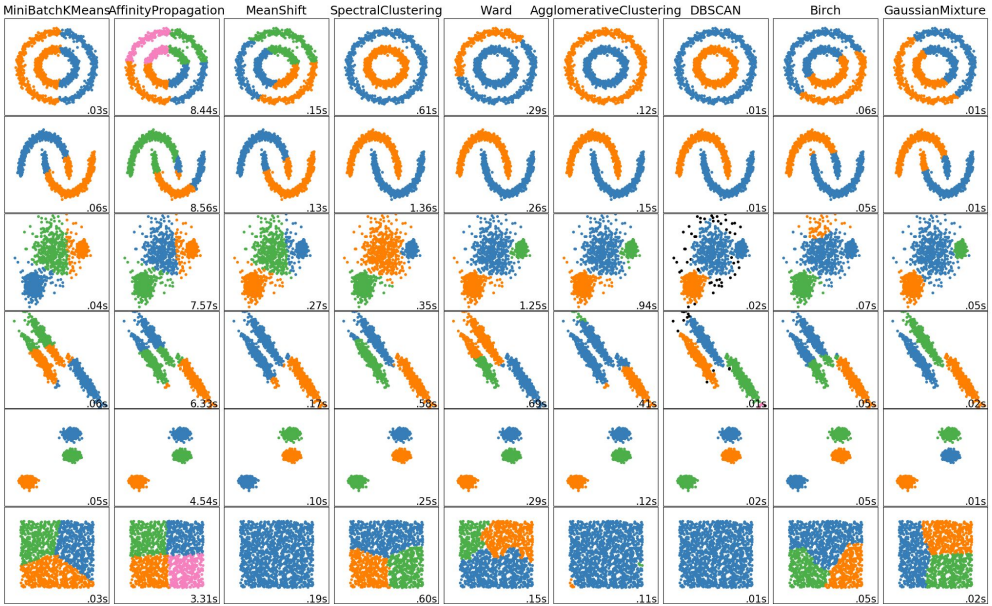
- основная задача (кластеризация новостей)
- вспомогательная задача (сегментация аудитории)



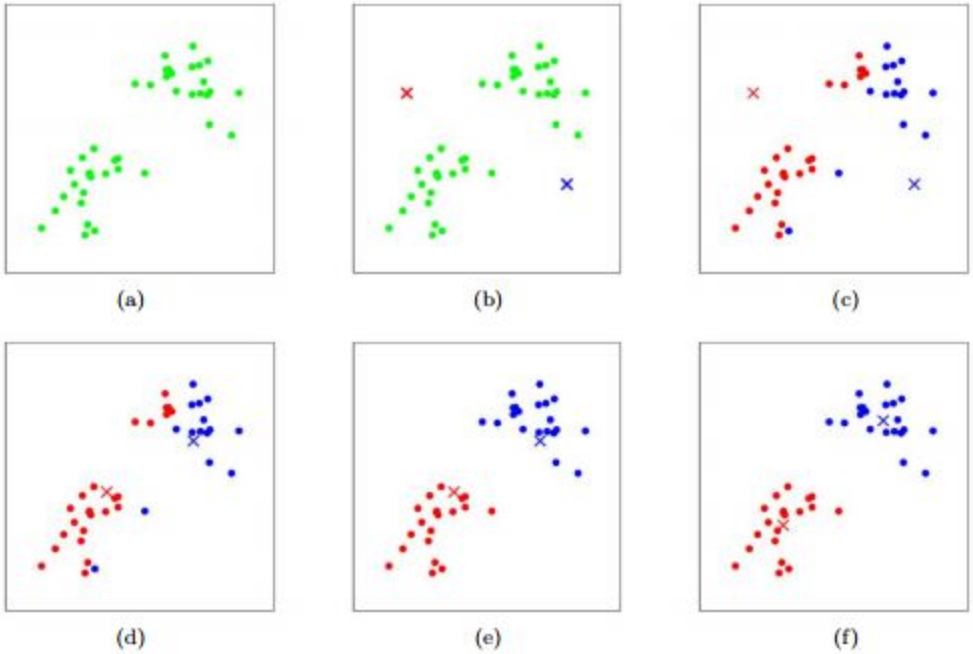


Кластеризация

универсального алгоритма кластеризации нет



Метод K-средних



Выбираем число кластеров (К), случайно выбираем К точек

- считаем для каждого элемента расстояние до каждого центроида и относим его к ближайшему
- пересчитываем для каждого кластера центроид - как средний элемент
- повторяем пока на итерации центроиды не останутся прежними

http://stanford.edu/~cpiech/cs221/handouts/kmeans.html

Метод K-средних

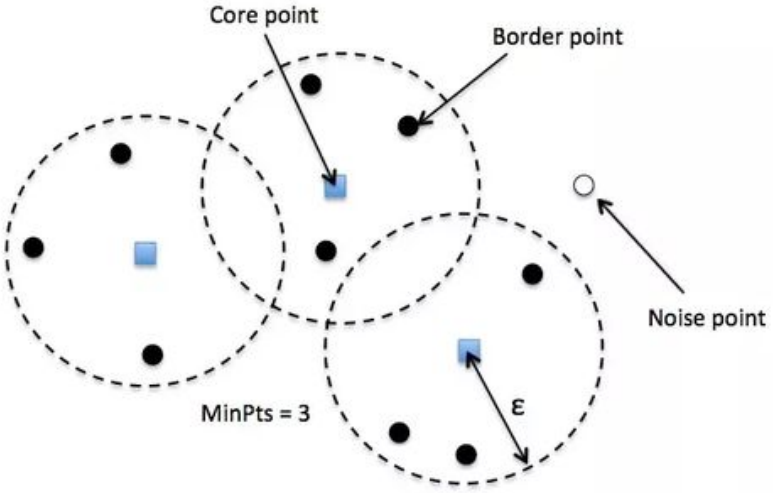
Недостатки:

- Попадает в локальный оптимум, а не глобальный. Лучше перезапускать несколько раз (или использовать умные инициализации наподобие k-means++).

- Нужно знать число кластеров заранее.

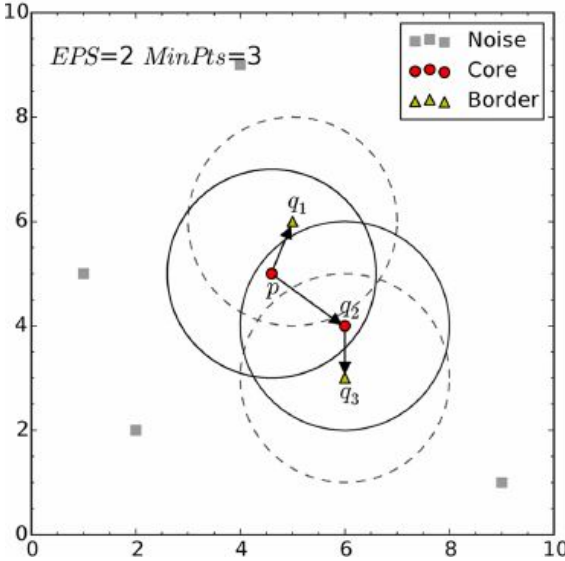
DBSCAN

плотностный алгоритм кластеризации
- плотность - количество объектов внутри сферы заданного радиуса eps
- core-объект (основной) - плотность вокруг него больше min_pts
- граничный объект - плотность меньше min_pts, но рядом есть core-объект
- шум - ни core, ни граничный



[тутор]
[оригинальная статья]

DBScan



1. Разделить точки на core (основные), пограничные и шумовые.

2. Отбросить шумовые точки.

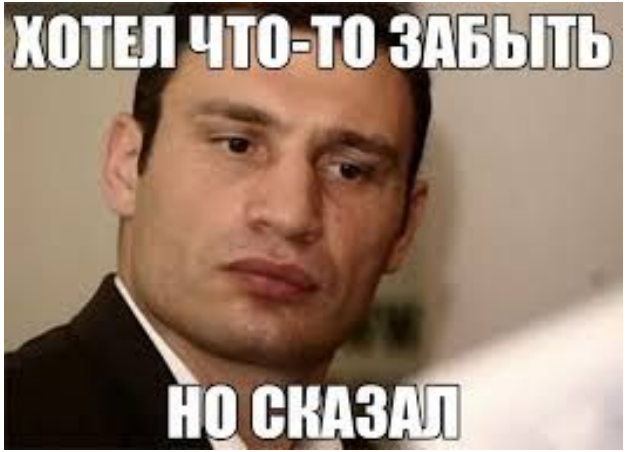
3. Соединить основные точки, которые находятся на расстоянии ε друг от друга.

4. Каждую группу соединенных основных точек объединить в свой кластер.

5. Отнести пограничные точки к соответствующим им кластерам.

http://scikit-learn.org/stable/modules/clustering.html#dbscan

Оставьте обратную связь!!!



Q?

Почитать

- https://ru.coursera.org/lecture/algorithms-part1/quick-union-ZgecU
- https://www.geeksforgeeks.org/union-find-algorithm-set-2-union-by-rank/

Про соседей

- https://habr.com/company/mailru/blog/338360/
- https://www.youtube.com/watch?v=UUm4MOyVTnE

Про кластеризацию

- https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68
- http://www.leonidzhukov.net/hse/2015/networks/papers/GraphClustering_Schaeffer07.pdf