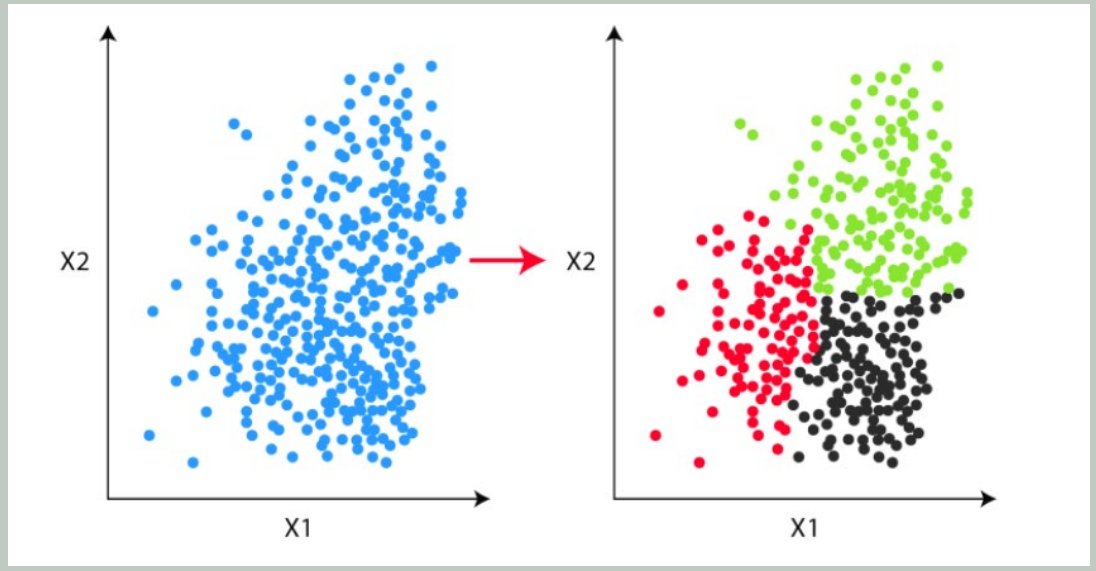
Кластеризация

Постановка задачи

- Даны объекты $x_1, x_2, \ldots, x_n \in X$.
- Требуется 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.
- Формализация задачи: необходимо построить алгоритм $a: X \to \{1, \ldots, K\}$, сопоставляющий каждому объекту x номер кластера.



Для чего используется

- Кластеризация делается на неразмеченных данных, следовательно, ее можно использовать для разведочного анализа данных
- Визуализация
- Детекция выбросов
- Сегментация изображений
- Группировка результатов поиска
- Анализ социальных сетей
- Тематическое моделирование
- Латентный семантический анализ
- Компьютерно-филологические штуки

Типы метрик качества

- Внешние метрики – используют информацию об истинных метках объектов (если у нас **есть разметка**, а мы хотим только добавить признаки)
- Внутренние метрики – оценивают качество кластеризации, основываясь только на наборе данных

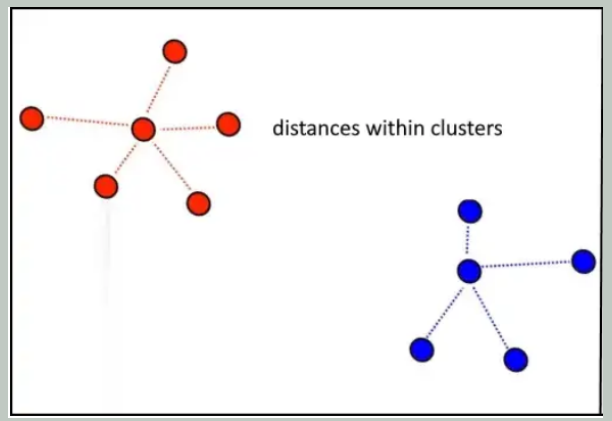
Оптимизация

Внутрикластерное расстояние

Пусть c_k - центр k-го кластера

Внутри кластера все объекты максимально похожи, поэтому наша цель – **минимизировать внутрикластерное расстояние:**

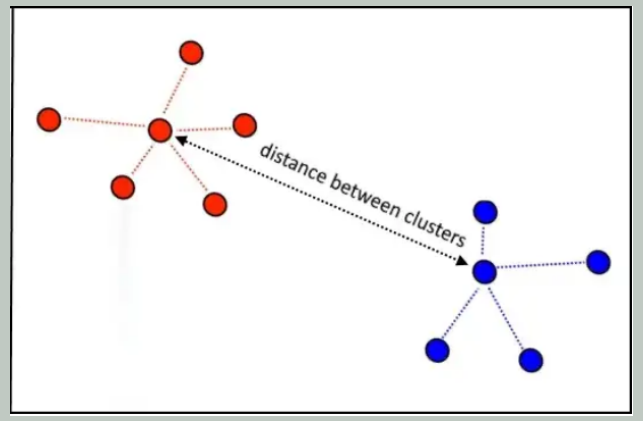
$$\sum_{k=1}^{K}\sum_{i=1}^{l}[a(x_i)=k]\rho(x_i,c_k)\to\min_a$$



Межкластерное расстояние

Объекты из разных кластеров должны быть как можно менее похожи друг на друга, поэтому мы **максимизируем межкластерное расстояние**:

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



Индекс Данна (Dunn Index)

Хотим минимизировать внутрикластерное расстояние и одновременно максимизировать межкластерное расстояние:

$$\frac{\min\limits_{1\le k<k\prime\le K} d(k,k')}{\max\limits_{1\le k\le K} d(k)} \to \max_{a}$$

$d(k,k')$ - расстояние между кластерами k и k';

$d(k)$ – внутрикластерное расстояние для k-го кластера.

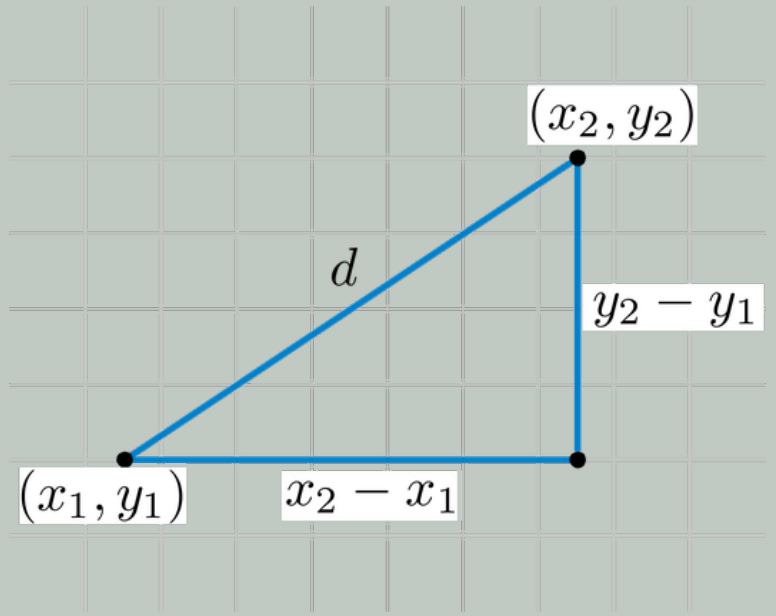
(Чем выше индекс Данна, тем лучше разделены и более компактны кластеры)

статья на тему

Виды расстояний между объектами

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

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



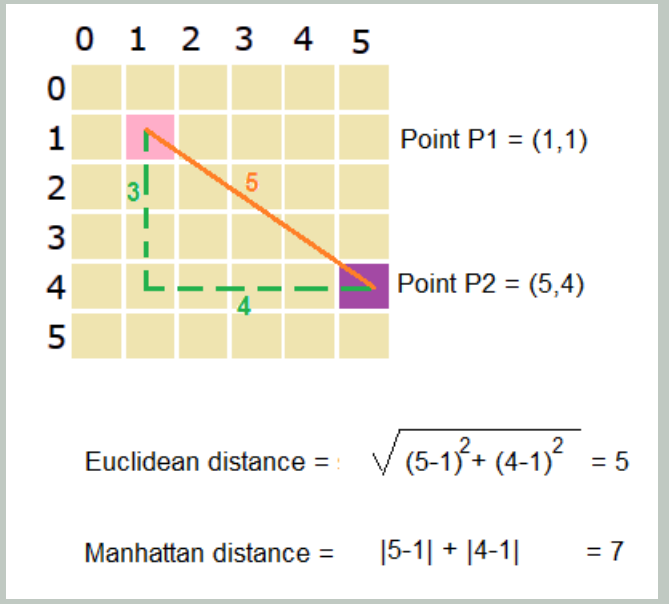
Виды расстояний между объектами

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

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

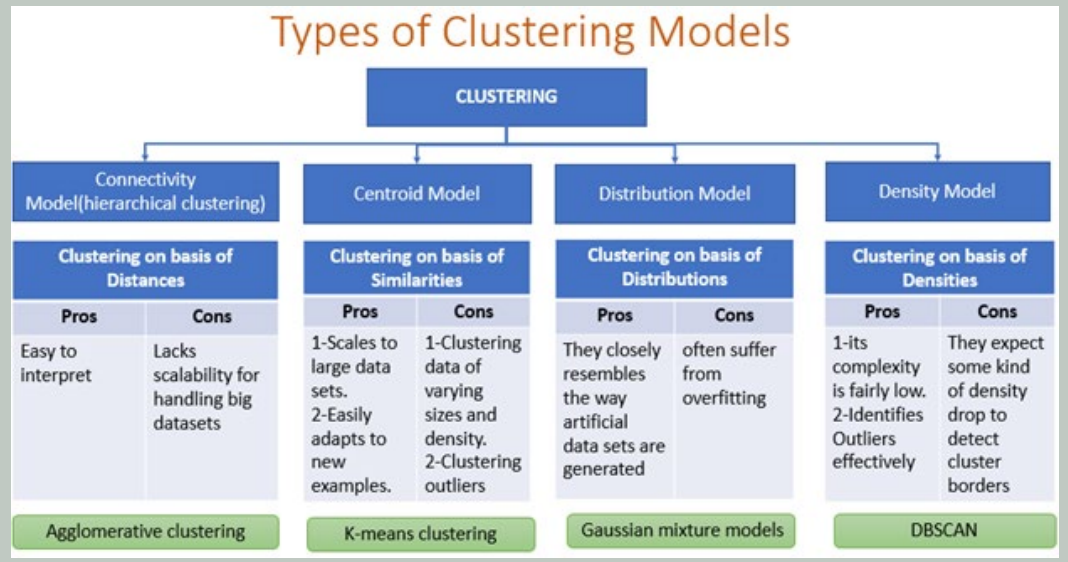
- Манхеттенское расстояние (расстояние городских кварталов):

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



Алгоритмы кластеризации

Виды алгоритмов



- Можно использовать точки-центроиды и кластеризовать по похожести;
- Можно использовать графы и считать расстояния;
- Можно смотреть плотность точек в выборке;
- Можно смотреть на распределение

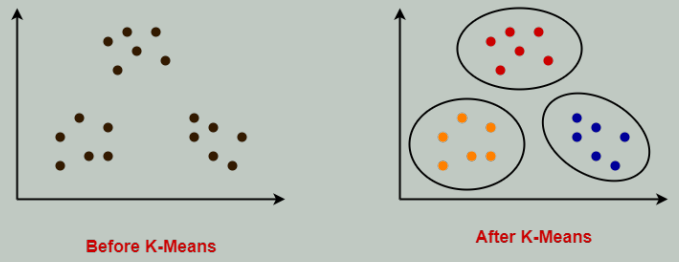
[статья с большим количеством алгоритмов]

K-Means (алгоритм Ллойда)

- Используем индекс Данна для оптимизации: максимизируем расстояние между кластерами и одновременно минимизируем расстояние внутри кластеров.
- Порядок действий:

1. Случайно выбрать центры кластеров $c_1, c_2, \ldots, c_K$
2. Каждый объект отнести к ближайшему к нему центру кластера
3. Пересчитать центры полученных кластеров
4. Повторять шаги 2 и 3 до стабилизации кластеров (когда центры перестанут сильно двигаться).

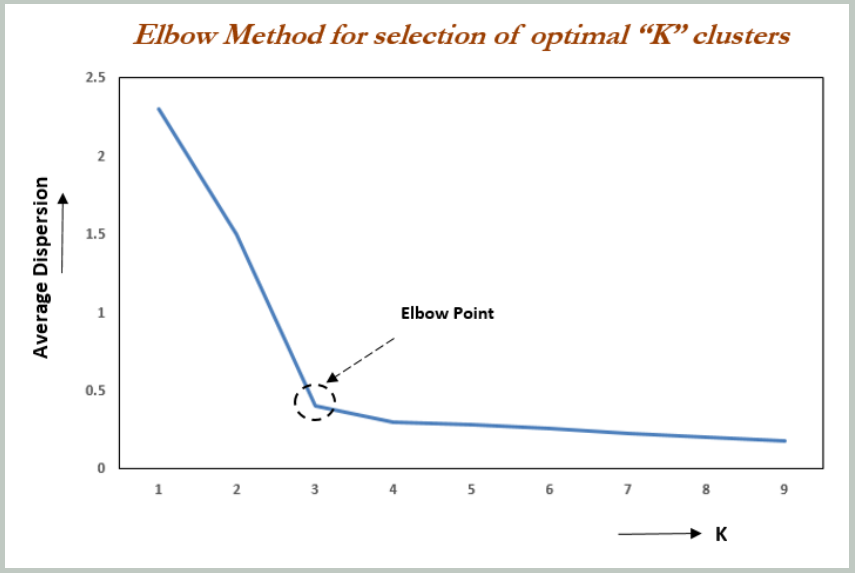
- Хорошо работает, если:
  - Разброс распределения каждого признака сферичный
  - Кластеры линейно разделимы
  - У кластеров примерно похожие объемы
  - Разброс у разных переменных похожий



визуализация

Метод локтя

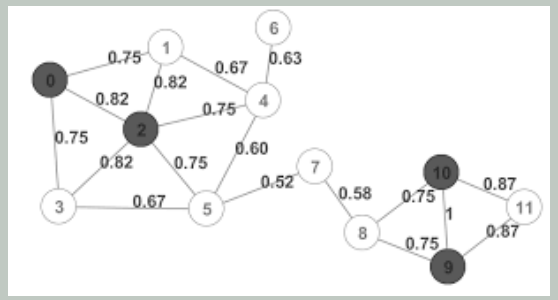
- Как выбрать количество кластеров K?
- Использовать elbow method: построить алгоритм для K = 1, 2, 3... и применить WCSS(Within-Cluster Sum of Squares), который вычисляет сумму расстояний между всеми членами кластера и его центроидом, чтобы минимизировать ее и достичь оптимального значения K.



Графовые методы кластеризации

- выборка представляется в виде графа, где в вершинах стоят объекты, а на рёбрах – расстояния между ними
- Алгоритм выделения связных компонент:

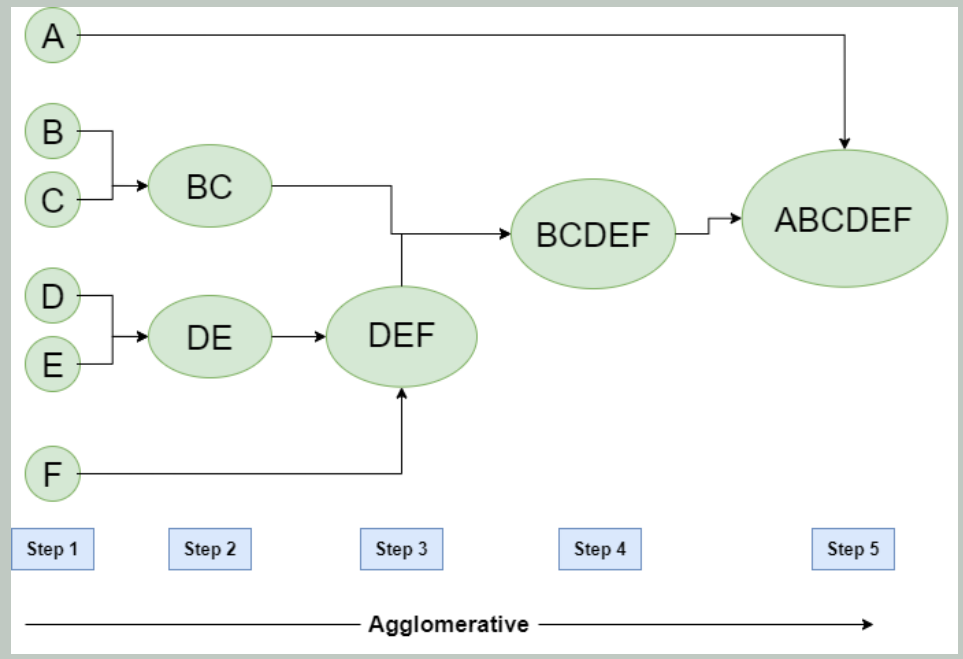
1. из графа удаляются все ребра, для которых расстояния больше некоторого значения R
2. Кластеры – объекты, попадающие в одну компоненту связности



[статья про графовые методы]

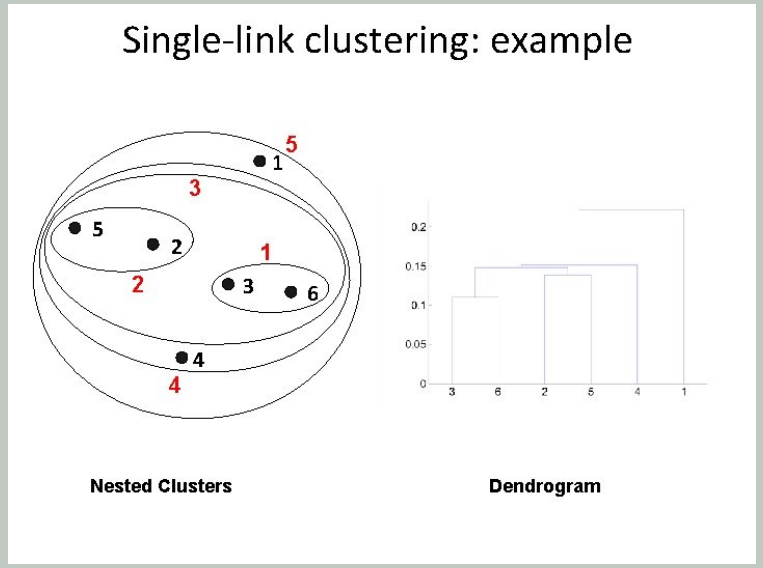
Иерархическая кластеризация

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



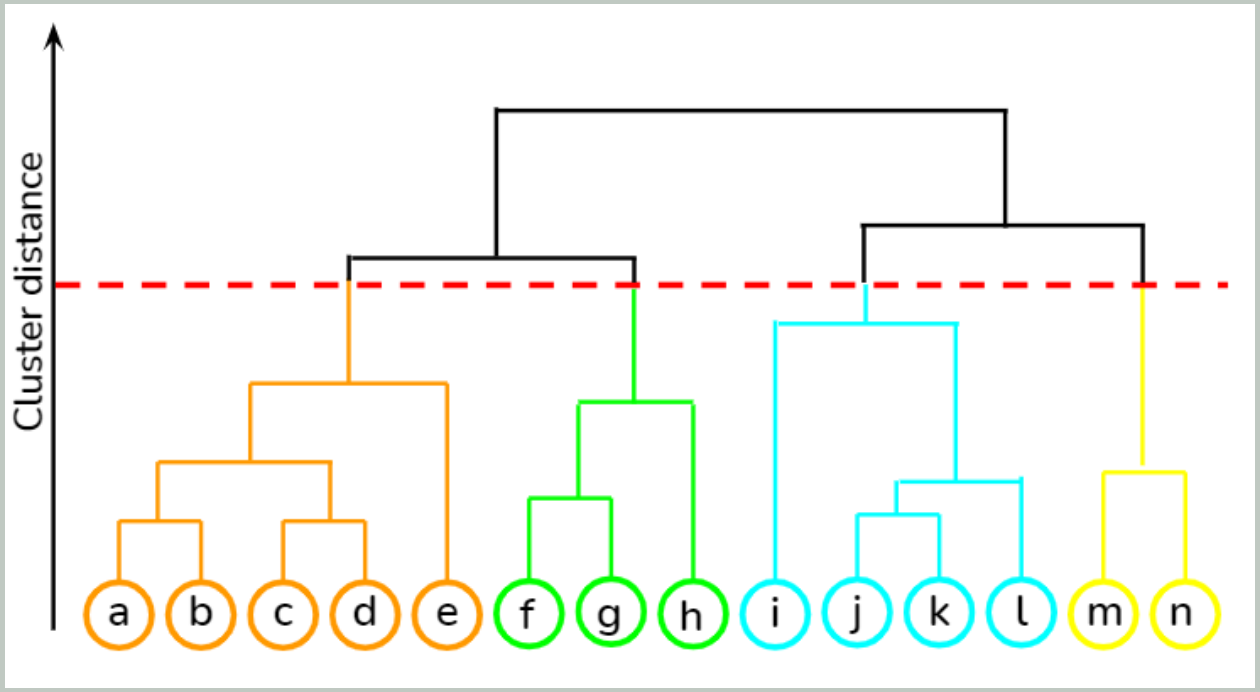
Иерархическая кластеризация

- Алгоритм Ланса-Уильямса:
  - первый шаг: один кластер = один объект
  - на каждом следующем шаге объединяем два наиболее похожих кластера (по некоторой мере схожести d) с предыдущего шага



Как выбирать количество кластеров

- Выбирать количество кластеров можно просто по картинке дерева: смотрим, где происходит много объединений, а где ничего не делится



Расстояние между кластерами

- **Single Linkage**

  $D(c_1,c_2) = \min D(x_1,x_2)$

  Minimum distance or distance between closest elements in clusters

- **Complete Linkage**

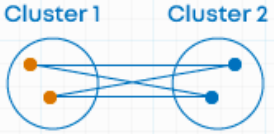
  $D(c_1,c_2) = \max D(x_1,x_2)$

  Maximum distance between elements in clusters

- **Average Linkage**

  $D(c_1,c_2) = \frac{1}{|c_1|}\frac{1}{|c_2|}\Sigma\Sigma D(x_1,x_2)$

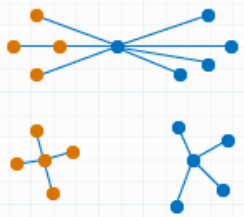
  Average of the distances of all pairs



- **Centroid Method**

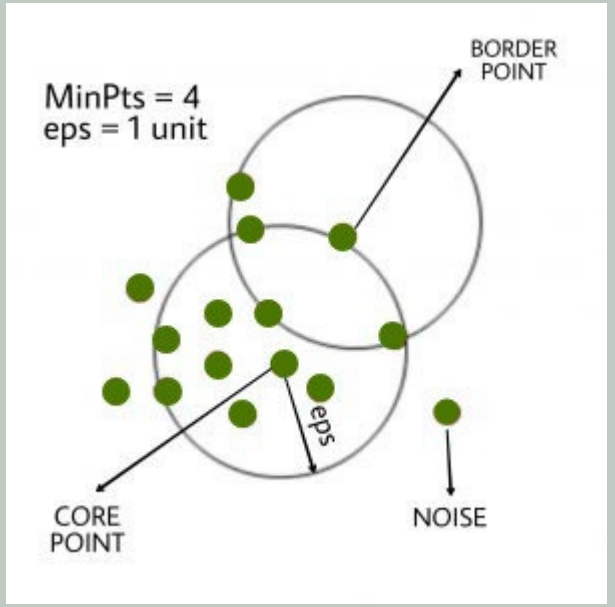
  Combining clusters with minimum distance between the centroids of the two clusters

- **Ward's Method**
  - Combining clusters where increase in within cluster variance is to the smallest degree.
  - Objective is to minimize the total within cluster vairance



Density-based clustering (DBSCAN)

- Объекты: основные, граничные, шумовые.
- Параметры метода:
  - eps – размер окрестности
  - min_samples – минимальное число объектов в окрестности (включая сам объект), для определения основных точек



Density-based clustering

Алгоритм DBSCAN

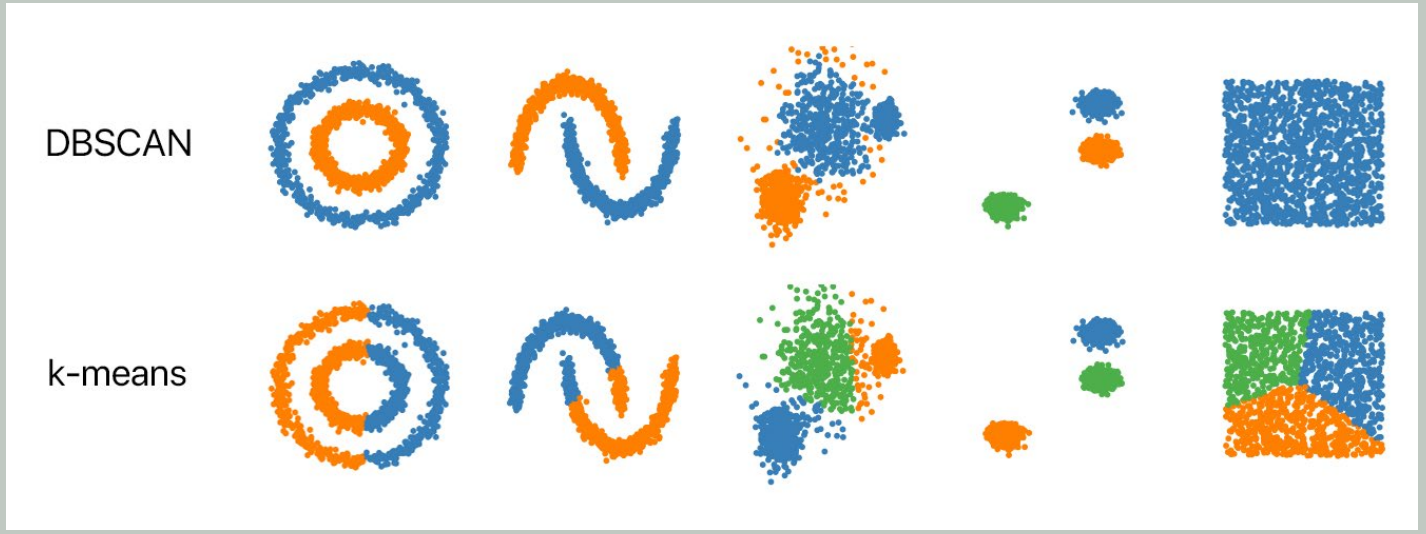
1. Выбрать точку без метки
2. Если в окрестности меньше, чем min_pts точек, то пометить её как шумовую
3. Создать кластер, поместить в него текущую точку (если это не шум, см. п.2)
4. Для всех точек из окрестности S:
   - если точка шумовая, то отнести к данному кластеру, но не использовать для расширения
   - если точка основная, то отнести к данному кластеру, а её окрестность добавить к S
5. Перейти к шагу 1.

визуализация

K-MEANS vs DBSCAN

K-Means
- плохо справляется с невыпуклыми кластерами - чувствителен к параметру K - плохо детектирует аномалии

DBSCAN
- Плохо работает с разреженными датасетами и для объектов с меняющейся плотностью.



Метрики качества

RAND INDEX

- RAND INDEX (RI)
  - a – число пар объектов с одинаковыми метками и находящихся в одном кластере, b – число пар объектов с различными метками и находящихся в разных кластерах, N – число объектов в выборке

$$RI = \frac{a+b}{C_N^2} = \frac{2(a+b)}{N(N-1)}$$

  - RI – доля объектов, для которых исходное и полученное разбиения согласованы. Выражает похожесть двух различных разбиений выборки.
- Adjusted RAND INDEX (ARI)
  - RI нормируется так, чтобы величина всегда принимала значения из отрезка $[-1; 1]$ независимо от числа объектов N и числа кластеров, получается ARI:

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]}$$

  - ARI > 0 – разбиения похожи (ARI = 1 – совпадают), A$RI \approx 0$ – случайные разбиения, ARI < 0 – непохожие разбиения

Mutual Information

- Mutual Information (MI)
  - Индекс MI – это взаимная информация для двух разбиений выборки на кластеры:

$$MI(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} P_{UV}(i,j)\frac{\log P_{UV}(i,j)}{P_U(i)\cdot P_V(j)}$$

  где
  - $P_{UV}(i,j)$ - вероятность, что объект принадлежит кластеру $U_i \subset U$ и кластеру $V_j \subset V$
  - $P_U(i)$ – вероятность, что объект принадлежит кластеру $U_i \subset U$
  - $P_V(i)$ – вероятность, что объект принадлежит кластеру $V_j \subset V$
- Adjusted Mutual Information (AMI)
  - Взаимная информация измеряет долю информации, общей для обоих разбиений: насколько информация об одном из них уменьшает неопределенность относительно другого.
  - $AMI \in [0;1]$ – нормировка MI; чем ближе к 1, тем более похожи разбиения.

Гомогенность, полнота, V-мера

- Пусть H - энтропия; $H = -\sum_{i=1}^{|U|} P(i) \log P(i)$. Тогда $h = 1 - \frac{H(C|K)}{H(C)}$, $c = \frac{H(K|C)}{H(K)}$, где K – результат кластеризации, C – истинное разбиение выборки на классы.
- $\boldsymbol{h}$ (гомогенность) измеряет, насколько каждый кластер состоит из объектов одного класса
- $\mathbf{c}$ (полнота) измеряет, насколько объекты одного класса относятся к одному кластеру
- Гомогенность и полнота принимают значения из отрезка [0; 1]. Большие значения соответствуют более точной кластеризации.

Эти метрики не нормализованы (как ARI и AMI), т.е. они зависят от числа кластеров!

- При большом числе кластеров и малом числе объектов лучше использовать ARI и AMI
- При более 1000 объектов и числе кластеров меньше 10 проблема не так сильно выражена, поэтому её можно игнорировать.
- V-мера – учитывает и гомогенность и полноту, это их среднее гармоническое:

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

- V-мера показывает, насколько два разбиения схожи между собой.

Силуэт

- Не требует знания истинных меток! (значит, это **внутренняя** метрика качества кластеризации)
- Пусть a – среднее расстояние от объекта до всех объектов из того же кластера, b – среднее расстояние от объекта до объектов из ближайшего (не содержащего объект) кластера. Тогда силуэт данного объекта:

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

- Силуэт выборки ($\boldsymbol{S}$) – средняя величина силуэта по объектам. Силуэт показывает, насколько среднее расстояние до объектов своего кластера отличается от среднего расстояния до объектов других кластеров.

$$S \in [-1; 1]$$

- S, близкий к -1 – плохие (разрозненные) кластеризации
- $S \approx 0$ – кластеры накладываются друг на друга
- S, близкий к 1 – четко выраженные кластеры С помощью силуэта можно выбирать число кластеров k (если оно заранее неизвестно) – выбирается k, для которого метрика максимальна.
- Силуэт зависит от формы кластеров и достигает больших значений на более выпуклых кластерах.

Визуализация

MULTIDIMENSIONAL SCALING (MDS)

- Это семейство методов;
- Идея – минимизация квадратов отклонений между исходными и новыми попарными расстояниями:

$$\sum_{i \neq j}^{l} \left(\rho(x_i, x_j) - \rho(z_i, z_j)\right)^2 \rightarrow \min_{z_1,\ldots,z_l}$$

- Как вычислять:
  - Припишем количество точек к координатам в n-мерном пространстве
  - Вычислим евклидовы расстояния для всех пар точек. Получим матрицу схожести.
  - После этого сравним эту матрицу и матрицу сходного инпута с помощью стресс-функции ($stress = \sqrt{\frac{\sum_i \sum_j (d_{ij} - \hat{d}_{(ij)})}{\sum_i \sum_j (d_{ij}^2)}}$).
  - Теперь поправим координаты, чтобы минимизировать стресс-функцию.

статья (с примерами на R, правда)

еще статья

PCA (Principal Component Analysis)

- Используется не только для визуализации, но и для нее тоже.
- Проекции объектов на первую главную компоненту c_1 имеют наибольшую выборочную дисперсию среди дисперсий проекций на всевозможные направления в R_m
- При j ≥ 2 можно показать, что c_j — направление с наибольшей выборочной дисперсией проекций объектов среди направлений, ортогональных векторам $c_1, \dots, c_{j-1}$
- Если составить **ковариационную матрицу** и посчитать ее **главные оси**, а потом записать их в матричном виде, то каждый столбец такой матрицы – новый обобщенный признак

(я не сама это придумала, это копипаста из лекций Лагутина, имейте в виду)

[статья на английском]

[статья на хабре]

t-SNE(t-distributed Stochastic Neighbor Embedding)

- При проекции нам важно не сохранение расстояний между объектами, а сохранение пропорций:

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

[статья]

PCA vs t-SNE

- t-SNE создан в первую очередь для визуализации, а PCA для снижения размерности, но если выбрать количество главных компонент = 2 или 3, можно построить проекцию.
- Хотя считается, что для визуализации лучше t-SNE, **не все так однозначно**

PCA	t-SNE
- линейный - пытается сохранить глобальную структуру данных - не имеет гиперпараметров - выбросы сильно влияют - может применяться к новым данным, если его обучить	- нелинейный - пытается сохранить локальную структуру данных - Гиперпараметры: perplexity, LR & количество шагов - повторно к другим данным применяться не может

визуализация w2v в PCA и t-SNE

UMAP

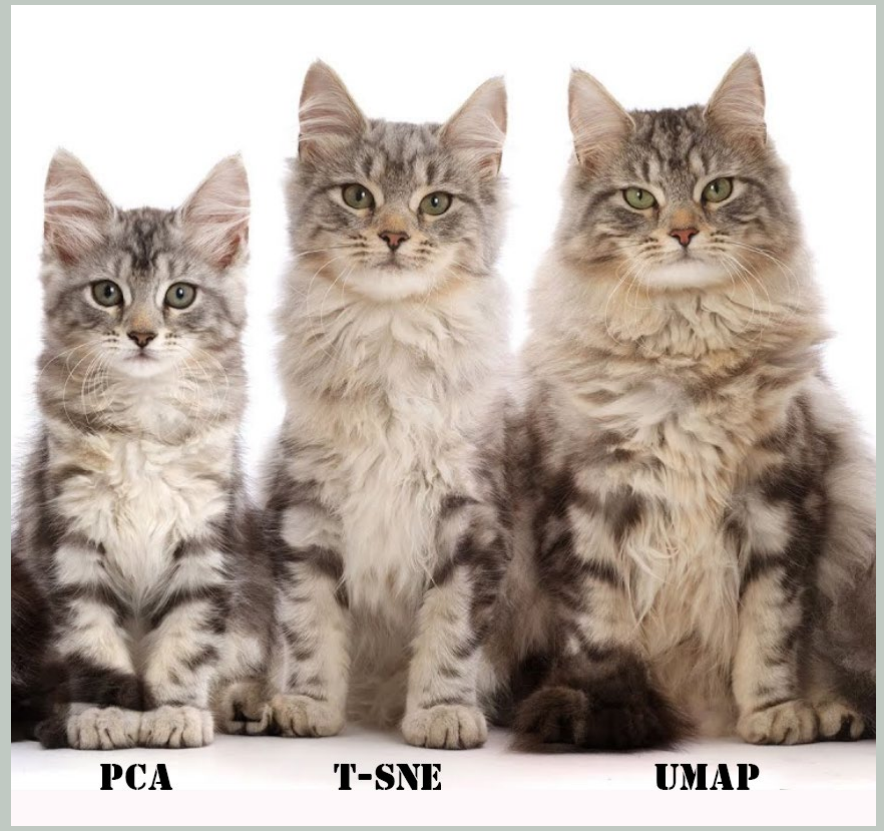
- Очень похож на t-SNE, но:
  - как и t-SNE, использует экспоненциальное распределение вероятностей, но **не обязательно евклидово расстояние, и его вероятности не нормализованы.**
  - Нет нормализации – быстрее вычисления!
  - Использует число ближайших соседей вместо preplexity
  - Использует другую функцию потерь (бинарная кросс-энтропия вместо KL-divergence)
  - Другая инициализация первоначальных точек
  - Сохраняет глобальную структуру (t-SNE только локальную) благодаря функции потерь
  - (да, там много математики в исходной статье)

сложная статья (нужен акк на medium)

еще статья

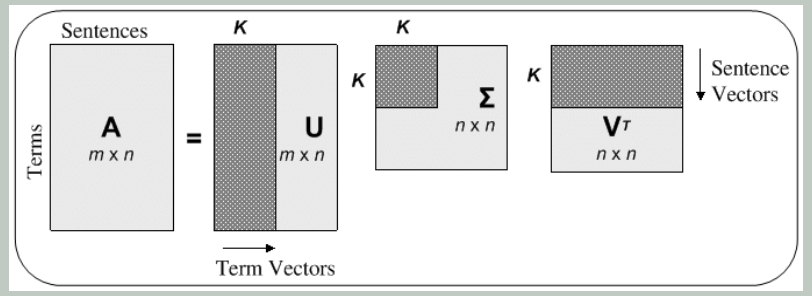
Уровни модности :)

- Судя по всему, UMAP очень полюбился биологам и генетикам
- В исследованиях по NLP иногда PCA показывает лучшие результаты, чем t-SNE (но обычно нет)
- чаще всего для визуализации используют несколько вариантов алгоритмов, а потом сравнивают



Кластеризация и текст

LSA (Latent Semantic Analysis)

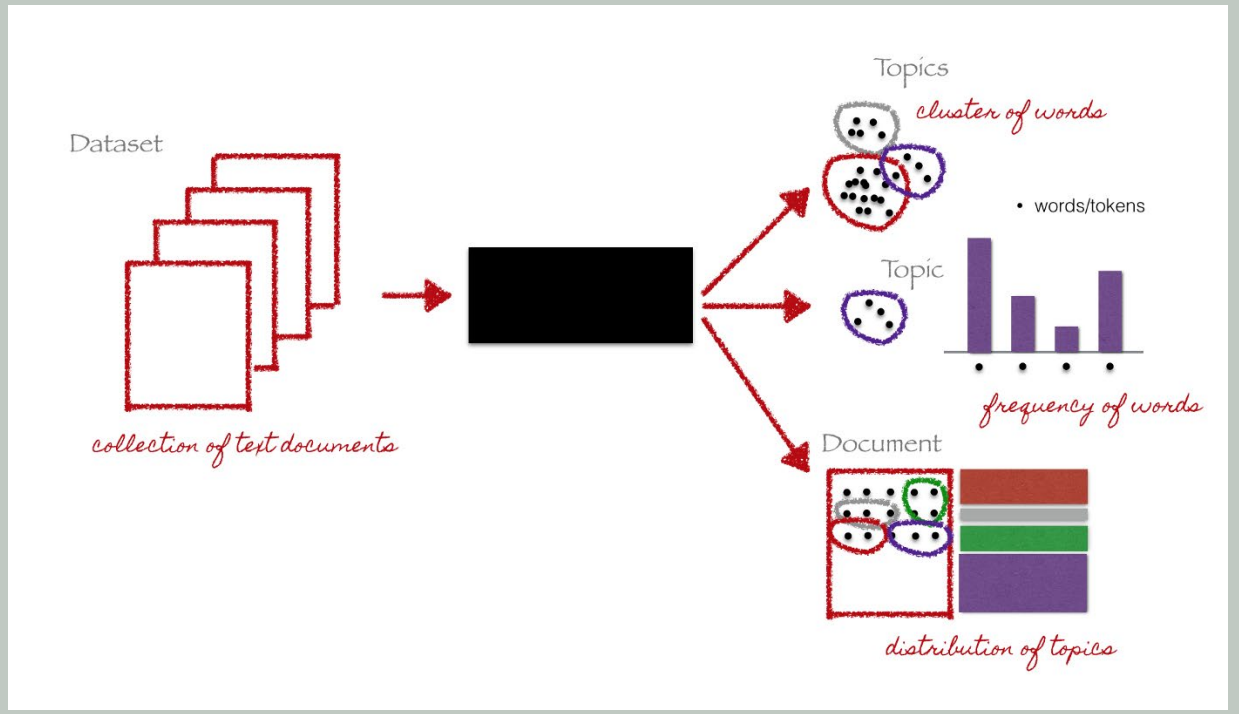


- составляем матрицу документ-слово
- раскладываем матрицу с помощью метода SVD
- получаем три матрицы U, Σ, V_T, где:
  - U – распределение слов по разным контекстам
  - Σ – диагональная матрица связи между контекстами
  - V_T - распределение контекстов по разным документам
- При этом в диагонали Σ оказываются веса важности контекстов, и если мы отбросим все, кроме K самых больших чисел в этой диагонали, то избавимся от редких контекстов (например, если слово «котик» редко встречается в сочетании с «сингулярное разложение матриц», мы такие контексты отбросим и осознаем, что их векторы перпендикулярны)
- В результате получаем в U векторы слов пониженной размерности.

LDA & Topic Modelling

- LDA (Latent Dirichlet Allocation) – алгоритм кластеризации
- Тематическое моделирование: если мы кластеризуем вектора слов, можем получить кластеры по темам
- Нам нужен будет корпус, поделенный на документы
- Количество тем мы задаем сами: это как K в K-Means

[статья]



LDA & Topic Modelling

- Алгоритм:
  - В каждом документе рандомно припишем всем словам номер k тем (k выбирается заранее)
  - Для каждого документа d и всех его слов вычислим:
  1. p(topic t | document d): пропорцию слов в документе d, которые приписаны теме t, с исключением текущего слова w, для которого считаем. Если много слов из документа d принадлежат t, вероятнее, что и слово w тоже принадлежит t. ($\frac{N \text{ слов в } d \text{ с темой } t + \alpha}{N \text{ слов в } d \text{ с любой темой } k + \alpha}$)
  2. p(word w | topic t): пропорцию того, что приписано теме t по всем документам по причине слова w. Как много слов в документе были тоже приписаны к t, потому что наше слово – этой темы?
  - Обновим вероятность того, что слово w принадлежит теме t, по формуле
    p(word w with topic t) = p(topic t | document d) * p(word w | topic t)