# Алгоритм DBSCAN

Петров Александр

15 декабря 2024 г.

## Постановка и актуальность задачи

Сейчас данных много. Модели стали сложными. В глубоком обучении также пытаются использовать неразмеченные данные. Частичное обучение является компромиссом между обучением без учителя (без каких-либо размеченных обучающих данных) и обучением с учителем (с полностью размеченным набором обучения).

Было замечено, что неразмеченные данные, будучи использованными совместно с небольшим количеством размеченных данных, могут обеспечить значительный прирост качества обучения.

Сбор размеченных данных для задачи обучения зачастую требует, чтобы квалифицированный эксперт вручную классифицировал объекты обучения. Затраты, связанные с процессом разметки, могут сделать построение полностью размеченного набора прецедентов невозможным, в то время как сбор неразмеченных данных сравнительно недорог. В подобных ситуациях ценность частичного обучения сложно переоценить.

Общий подход к данной задаче следующий:

- $U \subset X$ - большая неразмеченная выборка.
- $D \subset X \times Y$ - небольшая размеченная выборка.
- Требуется построить отображение $f : X \to Y$

## Алгоритм DBSCAN

В дополнение к определениям выше зададим определение эпсилон-окрестности элемента $x$ неразмеченной выборки $U$: $U_\varepsilon(x) = \{u \in U : \rho(x, u) \leq \varepsilon\}$. Разделим все объекты на 3 типа:

- корневой: имеющий плотную окрестность, $U_\varepsilon(x) \geq m$
- граничный: не корневой, но в окрестности корневого
- шумовой (выброс): не корневой и не граничный

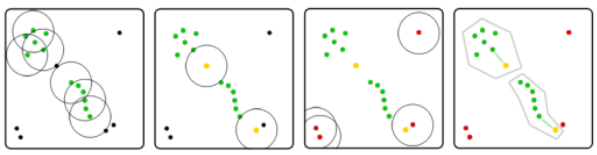

Рис. 1: Слева направо: корневые, граничные, шумовые объекты и кластеризация

Теперь рассмотрим, собственно, сам алгоритм кластеризации:

- Вход: выборка $X^l = \{x_1, ..., x_l\}$; параметры $\varepsilon$ и $m$.
- Выход: разбиение выборки на кластеры и шумовые выбросы; $U := X^l$ - непомеченные объекты, $a = 0$.
- Пока в выборке есть непомеченные точки, то есть $U \neq \emptyset$:
  - Берем случайную точку $x \in U$
  - Если $U_\varepsilon(x) < m$, то пометить $x$ как, возможно, шумовой.
  - Иначе:
    * Создаем новый кластер $K := U_\varepsilon(x)$; $a = a + 1$
    * Для всех $x' \in K$, не помеченных или шумовых:
      · Если $U_\varepsilon(x) \geq m$, то $K := K \cup U_\varepsilon(x')$
      · Иначе помечаем $x'$ как граничный кластера $K$
    * $a_i := a$ для всех $x_i \in K$
    * $U = U \backslash K$

# Преимущества и недостатки алгоритма DBSCAN

**Начнем с самых важных преимуществ алгоритма:**

- Быстрая кластеризация больших данных: асимптотика в худшем случае $O(l^2)$ и $O(l \log l)$ при эффективной реализации $U_\varepsilon(x)$;
- Кластеры могут иметь произвольную форму, что часто бывает полезно;
- Деление объектов на корневые, граничные и шумовые. Пример такого деления показан на рисунке 2.

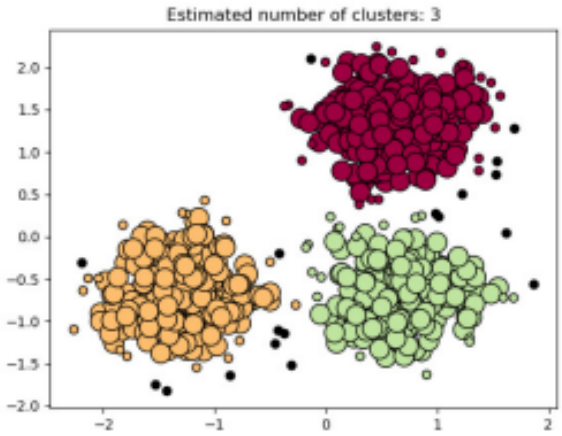


Рис. 2: Пример деления объектов на корневые, граничные и шумовые

**Однако, как и у любого алгоритма, у DBSCAN есть свои недостатки, а именно:**

- DBSCAN не полностью однозначен — краевые точки, которые могут быть достигнуты из более чем одного кластера, могут принадлежать любому из этих кластеров, что зависит от порядка просмотра точек. Однако для большинства наборов данных эти ситуации возникают редко и имеют малое влияние на результат кластеризации, так как основные точки и шум DBSCAN обрабатывает однозначно.

- Качество DBSCAN зависит от способа измерения расстояния между объектами. Наиболее часто используемой метрикой расстояний является евклидова метрика. Однако иногда, особенно для кластеризации данных высокой размерности, эта метрика может оказаться не очень хорошей ввиду экспоненциального роста необходимых экспериментальных данных в зависимости от размерности пространства
- DBSCAN не может хорошо кластеризовать наборы данных с большой разницей в плотности, так как в таком случае сложно подобрать параметр $m$;
- Если данные и масштаб не вполне хорошо поняты, выбор осмысленного параметра $\varepsilon$ может оказаться трудным.

# Задачи

## Задача 1

В каких случаях использование DBSCAN может быть предпочтительнее других алгоритмов кластеризации, таких как K-means?

*Решение:* DBSCAN предпочтительнее в случаях, когда данные содержат кластеры произвольной формы и/или когда данные содержат шум. В отличие от K-means, который требует заранее заданного числа кластеров и не учитывает выбросы, DBSCAN может обнаруживать кластеры любой формы (например, эллиптические или извилистые) и автоматически выделять шумовые точки (которые не принадлежат ни к одному кластеру). K-means также чувствителен к выбору начальных центров, а DBSCAN не зависит от инициализации и может успешно работать на данных с переменной плотностью.

## Задача 2

Как изменение параметров $\varepsilon$ и $m$ влияет на результаты кластеризации в DBSCAN?

*Решение:* Параметр $\varepsilon$ определяет радиус окрестности для поиска соседей. Если $\varepsilon$ слишком мал, то алгоритм будет воспринимать множество точек как шум и не сможет выделить кластеры, так как для большинства точек не будет достаточного количества соседей. Если $\varepsilon$ слишком велико, то кластеризация может стать слишком грубой, объединив слишком много точек в один кластер, включая выбросы.

Параметр $m$ контролирует минимальное количество соседей, необходимое для того, чтобы точка могла стать центром кластера. Если $m$ слишком велико, то алгоритм может не найти никакие кластеры, так как большинство точек не будут иметь достаточно соседей для формирования кластеров. Если $m$ слишком мало, DBSCAN может ошибочно выделить слишком много маленьких кластеров, что приведет к разбиению данных на множество неинформативных кластеров.

Таким образом, для достижения оптимальных результатов необходимо настроить оба параметра в зависимости от особенностей данных.

## Задача 3

Как изменение масштаба данных влияет на работу алгоритма DBSCAN? Почему важно нормализовать или стандартизировать данные перед применением DBSCAN?

*Решение:* Алгоритм DBSCAN использует расстояния между точками для определения их

близости, а также для вычисления плотности точек в окрестности (радиус $\varepsilon$). Если данные имеют разные масштабы по различным признакам, то алгоритм будет склонен учитывать признак с более крупным масштабом как более важный при расчете расстояний. Это приведет к искажению результатов кластеризации, так как DBSCAN будет ошибочно воспринимать различия в признаках с большими величинами как более значимые.

Для решения этой проблемы важно нормализовать или стандартизировать данные перед применением DBSCAN:

Нормализация приводит все признаки к одному масштабу, например, в диапазон [0, 1], что позволяет одинаково учитывать все признаки. Стандартизация изменяет данные так, чтобы каждый признак имел нулевое среднее значение и единичное стандартное отклонение, что также помогает привести данные к одинаковому масштабу. Если этого не сделать, то один из признаков может доминировать при расчете расстояний, что приведет к неверному определению плотности и кластеров. В результате DBSCAN может либо не обнаружить какие-то кластеры, либо объединить точки, которые не должны быть частью одного кластера.

Таким образом, перед применением DBSCAN важно привести данные к одному масштабу, чтобы алгоритм корректно учитывал все признаки и правильно разделял кластеры.