Алгоритм DBSCAN (Density-Based Spatial Clustering of Applications with Noise) — это метод кластеризации, который группирует точки на основе плотности их распределения. Он особенно полезен для выявления кластеров произвольной формы и для обработки шумов в данных. Вот основные аспекты алгоритма DBSCAN:

**Основные понятия**

**Параметры**:

**eps (ε)**: радиус окрестности, в пределах которого точки считаются соседями.

**minPts**: минимальное количество точек, необходимых для формирования кластера.

**Типы точек**:

**Ядровые точки**: точки, которые имеют по крайней мере minPts соседей в радиусе eps.

**Пограничные точки**: точки, которые находятся в радиусе eps от ядровой точки, но не имеют достаточного количества соседей, чтобы быть ядровыми.

**Шумовые точки**: точки, которые не являются ни ядровыми, ни пограничными.

**Алгоритм**

**Инициализация**: Начинаем с произвольной точки, которая еще не была обработана.

**Проверка соседей**: Находим все точки в радиусе eps от текущей точки.

**Формирование кластера**:

Если текущая точка является ядровой, создаем новый кластер и добавляем в него все ядровые точки и их пограничные точки.

Если текущая точка не является ядровой, помечаем ее как шумовую.

**Рекурсия**: Для каждой ядровой точки повторяем процесс, добавляя новые точки в кластер, пока не будут обработаны все точки.

**Завершение**: Процесс продолжается, пока не будут обработаны все точки в наборе данных.

import numpy as np

import matplotlib.pyplot as plt

from sklearn.cluster import DBSCAN

from sklearn.datasets import make\_moons

# Генерация тестовых данных (две луны)

X, \_ = make\_moons(n\_samples=300, noise=0.1)

# Применение алгоритма DBSCAN

dbscan = DBSCAN(eps=0.2, min\_samples=5)

labels = dbscan.fit\_predict(X)

# Визуализация результатов

plt.figure(figsize=(8, 6))

unique\_labels = set(labels)

colors = plt.cm.get\_cmap('Spectral', len(unique\_labels))

for k in unique\_labels:

class\_member\_mask = (labels == k)

xy = X[class\_member\_mask]

plt.scatter(xy[:, 0], xy[:, 1], s=30, color=colors(k), label=f'Cluster {k}')

plt.title('DBSCAN Clustering')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.legend()

plt.show()

**Импорт библиотек**: Мы импортируем необходимые библиотеки, такие как numpy для работы с массивами, matplotlib для визуализации и DBSCAN из sklearn.cluster.

**Генерация данных**: Используем функцию make\_moons для создания тестового набора данных с двумя лунообразными кластерами.

**Применение DBSCAN**:

Создаем объект DBSCAN с параметрами eps (радиус окрестности) и min\_samples (минимальное количество точек для формирования кластера).

Метод fit\_predict применяет алгоритм к данным и возвращает метки кластеров для каждой точки.

**Визуализация**: Мы используем matplotlib для визуализации кластеров. Каждому кластеру присваивается свой цвет, а шумовые точки (метка -1) отображаются отдельно.

**Примечания:**

Параметры eps и min\_samples могут быть настроены в зависимости от конкретных данных и желаемой плотности кластеров.

DBSCAN хорошо работает с данными, содержащими шум, и может находить кластеры произвольной формы, что делает его полезным для многих приложений.

…………………………………………………………………………..

Комидианная кластеризация (или медианная кластеризация) — это метод кластеризации, который использует медиану для определения центров кластеров. Этот подход может быть особенно полезен в ситуациях, когда данные содержат выбросы, так как медиана менее чувствительна к выбросам по сравнению со средним значением.

**Основные аспекты медианной кластеризации:**

**Определение центров кластеров**: В отличие от методов, таких как K-means, которые используют среднее значение для определения центров кластеров, медианная кластеризация использует медиану. Это позволяет лучше справляться с выбросами.

**Итеративный процесс**: Как и в K-means, медианная кластеризация включает итеративный процесс, в котором:

Каждая точка данных присваивается ближайшему кластеру на основе расстояния до медианы.

Центры кластеров обновляются, вычисляя медиану всех точек, принадлежащих к каждому кластеру.

**Параметры**: Как и в других методах кластеризации, необходимо задать количество кластеров, которое вы хотите получить.

**Применение медианной кластеризации:**

Медианная кластеризация может быть полезна в различных областях, таких как:

Обработка изображений

Анализ данных с выбросами

Социальные сети и анализ пользовательских данных

**Пример реализации на Python:**

Для реализации медианной кластеризации можно использовать библиотеку scikit-learn, которая предоставляет инструменты для работы с различными алгоритмами кластеризации. Однако, в scikit-learn нет встроенной реализации медианной кластеризации, поэтому можно использовать K-medoids, который является аналогом K-means, но использует медиану.

Вот пример использования K-medoids с библиотекой sklearn\_extra:

``python import numpy as np import matplotlib.pyplot as plt from sklearn\_extra.cluster import KMedoids from sklearn.datasets import make\_moons

**Генерация тестовых данных (две луны)**

X, \_ = make\_moons(n\_samples=300, noise=0.1)

**Применение K-medoids**

Комидианная кластеризация (или медианная кластеризация) — это метод кластеризации, который использует медиану для определения центров кластеров. Этот подход может быть особенно полезен в ситуациях, когда данные содержат выбросы или шум, так как медиана менее чувствительна к экстремальным значениям по сравнению со средним.

**Основные аспекты медианной кластеризации:**

**Определение центров кластеров**: В отличие от методов, таких как K-means, которые используют среднее значение для определения центров кластеров, медианная кластеризация использует медиану. Это позволяет лучше справляться с выбросами.

**Алгоритм**:

Инициализация: Выбираются начальные центры кластеров (медианы).

Присвоение кластеров: Каждая точка данных присваивается кластеру, чей центр (медиана) наиболее близок к ней.

Обновление центров: Центры кластеров пересчитываются как медианы точек, принадлежащих каждому кластеру.

Повторение: Процесс повторяется до тех пор, пока центры кластеров не перестанут изменяться или не будет достигнуто максимальное количество итераций.

**Применение:**

Медианная кластеризация может быть полезна в различных областях, таких как:

Обработка изображений

Анализ данных с выбросами

Социальные науки, где данные могут быть искажены экстремальными значениями

**Пример реализации:**

В Python можно использовать библиотеки, такие как scikit-learn`, для реализации медианной кластеризации, хотя в стандартной библиотеке нет прямой реализации медианной кластеризации. Однако можно создать свой собственный алгоритм, используя функции для вычисления медианы.

import numpy as np

import matplotlib.pyplot as plt

from sklearn\_extra.cluster import KMedoids

from sklearn.datasets import make\_moons

# Генерация тестовых данных (две луны)

X, \_ = make\_moons(n\_samples=300, noise=0.1)

# Применение K-medoids

n\_clusters = 2 # Задаем количество кластеров

kmedoids = KMedoids(n\_clusters=n\_clusters, random\_state=0)

labels = kmedoids.fit\_predict(X)

# Визуализация результатов

plt.figure(figsize=(8, 6))

unique\_labels = set(labels)

colors = plt.cm.get\_cmap('Spectral', len(unique\_labels))

for k in unique\_labels:

class\_member\_mask = (labels == k)

xy = X[class\_member\_mask]

plt.scatter(xy[:, 0], xy[:, 1], s=30, color=colors(k), label=f'Cluster {k}')

# Отображение центров кластеров

centers = kmedoids.cluster\_centers\_

plt.scatter(centers[:, 0], centers[:, 1], s=300, color='black', marker='X', label='Centers')

plt.title('K-Medoids Clustering')

plt.xlabel('Feature 1')

plt.ylabel('Feature 2')

plt.legend()

plt.show()

**Импорт библиотек**: Импортируем необходимые библиотеки, такие как numpy для работы с массивами, matplotlib для визуализации и KMedoids из sklearn\_extra.cluster.

**Генерация данных**: Используем функцию make\_moons для создания тестового набора данных с двумя лунообразными кластерами.

**Применение K-medoids**: Создаем объект KMedoids, указывая количество кластеров, и применяем метод fit\_predict для получения меток кластеров.

**Визуализация**: Используем matplotlib для визуализации кластеров и отображения центров кластеров.