Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Ульяновский государственный технический университет» Кафедра «Вычислительная техника»

Дисциплина «Системы искусственного интеллекта»

Лабораторная работа №5

«Алгоритмы кластеризации данных»

Вариант №15

Выполнил студент группы УПАСбд-41:

Пяткин И.А. Проверил: Хайруллин И.Д.

Ульяновск, 2025

# Задание:

1. Произвести масштабирование признаков (scaling).
2. С использованием библиотеки [scikit-learn](http://scikit-learn.org/stable/) написать программу с использованием алгоритмов кластеризации данных, позволяющую разделить исходную выборку на классы, соответствующие предложенной вариантом задаче (<http://scikit-learn.org/stable/modules/clustering.html>).
3. Провести эксперименты и определить наилучший алгоритм кластеризации, параметры алгоритма. Необходимо использовать не менее 3-х алгоритмов. Данные экспериментов необходимо представить в отчете (графики, ход проведения эксперимента, выводы).

Датасет по варианту: Wholesale customers

# Информация по датасету:

Датасет содержит информацию о существующих клиентах (магазинах), которые покупают определенные категории товара оптом. Содержит в себе 8 признаков.

# Описание реализации:

**Масштабирование данных:**

Масштабирование – важный шаг в моделировании алгоритмов с помощью наборов данных.

Для данной лабораторной работы была использована функция масшта- бирования StandartScaler() - масштабирует весь набор данных, чтобы среднее значение было равно 0, а стандартное отклонение было равно 1.

**Алгоритм кластеризации K-means (Центроидный метод)**

**K-means** — кластеризует данные, разделяя выборки на n групп с равной дисперсией, минимизируя критерий, известный как инерция или внутрикластерная сумма квадратов. Этот алгоритм требует указания количества кластеров. Он хорошо масштабируется для большого количества выборок и применяется в самых разных областях.

В общих чертах, алгоритм состоит из трёх этапов. На первом этапе выбираются начальные центроиды, а самый простой метод заключается в выборе образцов из набора данных. После инициализации алгоритм K-средних состоит из цикла, включающего два других шага. На первом шаге каждый образец назначается ближайшему к нему центроиду. На втором шаге создаются новые центроиды путем вычисления среднего значения всех образцов, назначенных каждому предыдущему центроиду. Вычисляется разница между старым и новым центроидами, и алгоритм повторяет эти два последних шага до тех пор, пока это значение не станет меньше порогового значения. Другими словами, алгоритм повторяется до тех пор, пока центроиды не перестанут существенно меняться.

В лабораторной работе используется функция K-means(), в ней необходимо указать количество кластеров. Далее выполнить анализ и провести кластеризацию данных.

**Алгоритм кластеризации DBSCAN (Плотностный метод)**

Алгоритм **DBSCAN** рассматривает кластеры как области высокой плотности, разделенные областями низкой плотности. Из-за этого довольно общего представления кластеры, найденные DBSCAN, могут иметь любую форму, в отличие от k-средних, которые предполагают, что кластеры имеют выпуклую форму. Центральным компонентом DBSCAN является концепция основных образцов , которые являются образцами, находящимися в областях высокой плотности. Таким образом, кластер представляет собой набор основных образцов, каждый из которых находится близко друг к другу (измеряемый некоторой мерой расстояния), и набор неосновных образцов, которые находятся близко к основному образцу (но сами не являются основными образцами). Алгоритм имеет два параметра min\_samples и eps.

В лабораторной работе используется функция DBSCAN(), в ней необходимо указать eps (параметр определяет радиус окрестности вокруг точки данных) и min\_samples (минимальное количество точек, необходимых в пределах фиксированного радиуса). Далее выполнить анализ и провести кластеризацию данных.

**Алгоритм кластеризации AgglomerativeClustering (Иерархический метод)**

Иерархическая кластеризация — это общее семейство алгоритмов кластеризации, которые строят вложенные кластеры путём их последовательного слияния или разделения. Эта иерархия кластеров представлена ​​в виде дерева (или дендрограммы). Корень дерева — это уникальный кластер, объединяющий все образцы, а листья — кластеры, содержащие только один образец. Объект **AgglomerativeClustering** выполняет иерархическую кластеризацию, используя подход «снизу вверх»: каждое наблюдение начинается в своём кластере, и кластеры последовательно объединяются.

В лабораторной работе используется функция AgglomerativeClustering(), в ней необходимо указать количество кластеров. Далее выполнить анализ и провести кластеризацию данных.

# Вывод:

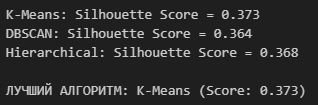


Рисунок 1 - Результаты экспериментов с алгоритмами кластеризации

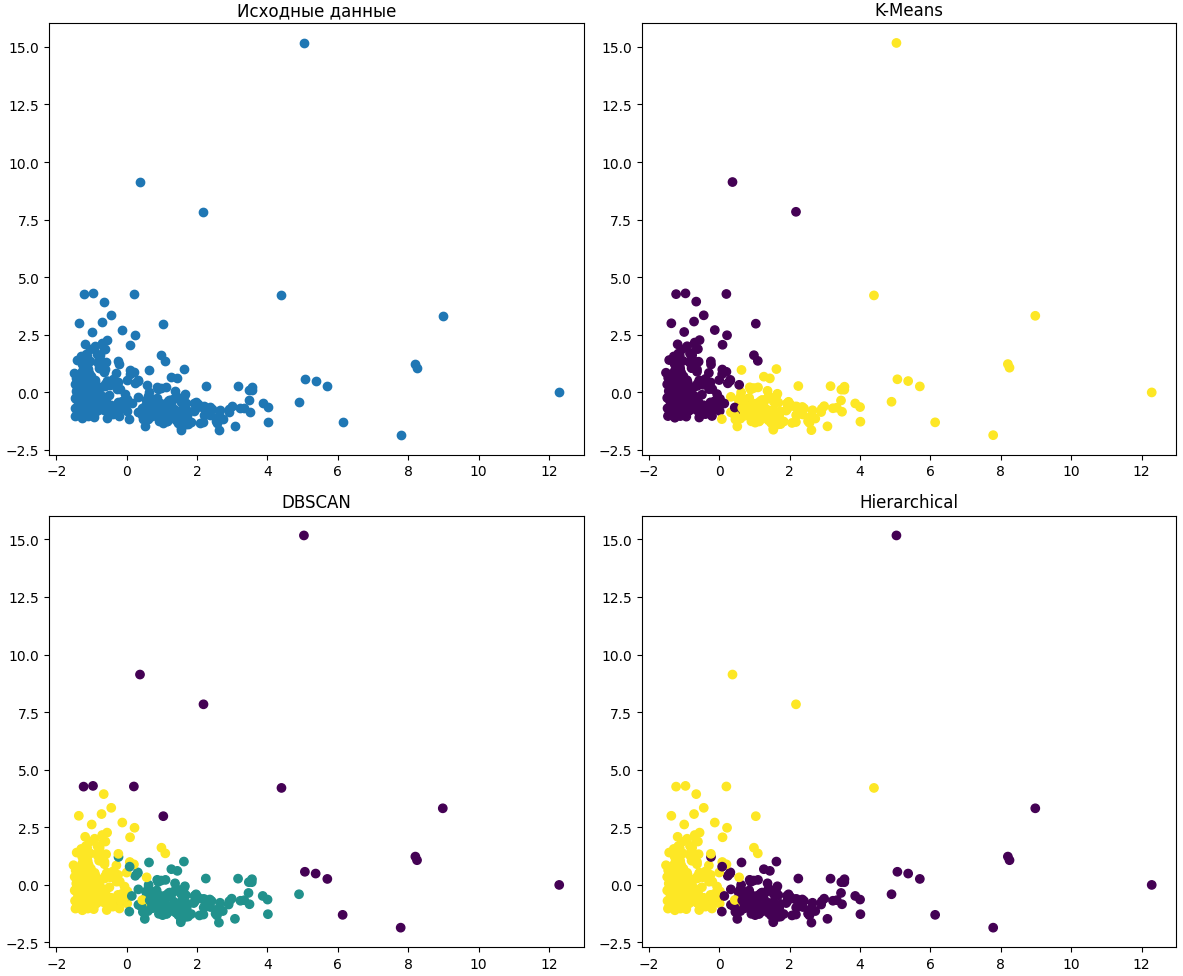


Рисунок 2 - Графики результатов распределения кластеризации

Вывод: Самое лучшее распределение получается при использовании 2 кластеров. Датасет хорошо разделяется на 2 группы (крупные и малые клиенты по покупке). Все алгоритмы показали результаты в примерно одинаковом диапазоне. Самым лучшим стал k-means с коэффициентом силуэта – 0,373.

# Приложение:

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.cluster import KMeans, DBSCAN, AgglomerativeClustering

from sklearn.metrics import silhouette\_score

from sklearn.decomposition import PCA

df = pd.read\_csv('Wholesale customers data.csv')

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(df)

results = {}

kmeans = KMeans(n\_clusters=2)

labels\_kmeans = kmeans.fit\_predict(X\_scaled)

results['K-Means'] = labels\_kmeans

dbscan = DBSCAN(eps=2.0, min\_samples=5)

labels\_dbscan = dbscan.fit\_predict(X\_scaled)

results['DBSCAN'] = labels\_dbscan

hierarchical = AgglomerativeClustering(n\_clusters=2)

labels\_hierarchical = hierarchical.fit\_predict(X\_scaled)

results['Hierarchical'] = labels\_hierarchical

best\_score = -1

best\_algorithm = None

for name, labels in results.items():

    if len(set(labels)) > 1:

        score = silhouette\_score(X\_scaled, labels)

        print(f"{name}: Silhouette Score = {score:.3f}")

        if score > best\_score:

            best\_score = score

            best\_algorithm = name

    else:

        print(f"{name}: не удалось создать кластеры")

print(f"\nЛУЧШИЙ АЛГОРИТМ: {best\_algorithm} (Score: {best\_score:.3f})")

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X\_scaled)

fig, axes = plt.subplots(2, 2, figsize=(12, 10))

axes[0,0].scatter(X\_pca[:, 0], X\_pca[:, 1])

axes[0,0].set\_title('Исходные данные')

axes[0,1].scatter(X\_pca[:, 0], X\_pca[:, 1], c=results['K-Means'])

axes[0,1].set\_title('K-Means')

axes[1,0].scatter(X\_pca[:, 0], X\_pca[:, 1], c=results['DBSCAN'])

axes[1,0].set\_title('DBSCAN')

axes[1,1].scatter(X\_pca[:, 0], X\_pca[:, 1], c=results['Hierarchical'])

axes[1,1].set\_title('Hierarchical')

plt.tight\_layout()

plt.show()