Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

Лабораторная работа № 5

По дисциплине: «Прикладные интеллектуальные системы и экспертные системы**»**

Кластеризация данных

Студент Александрук А.М.

Группа М-ИАП-23-1

Руководитель Кургасов В.В.
к.т.н. доцент

Липецк 2023г.

Цель работы:

Получить практические навыки решения задачи кластеризации фактографических данных в среде Jupiter Notebook. Научиться проводить настраивать параметры методов и оценивать точность полученного разбиения.

Задание кафедры

1) Загрузить выборки согласно варианту задания.

2) Отобразить данные на графике в пространстве признаков. Поскольку решается задача кластеризации, то подразумевается, что априорная информация о принадлежности каждого объекта истинному классу неизвестна, соответственно, на данном этапе все объекты на графике должны отображаться одним цветом, без привязки к классу.

3) Провести иерархическую кластеризацию выборки, используя разные способы вычисления расстояния между кластерами: расстояние ближайшего соседа (single), дальнего соседа (complete), Уорда (Ward). Построить дендрограммы для каждого способа. Размер графика должен быть подобран таким образом, чтобы дендрограмма хорошо читалась.

4) Исходя из дендрограмм выбрать лучший способ вычисления расстояния между кластерами.

5) Для выбранного способа, исходя из дендрограммы, определить количество кластеров в имеющейся выборке. Отобразить разбиение на кластеры и центроиды на графике в пространстве признаков (объекты одного кластера должны отображаться одним и тем же цветом, центроиды всех кластеров – также одним цветом, отличным от цвета кластеров)

6) Рассчитать среднюю сумму квадратов расстояний до центроида, среднюю сумму средних внутрикластерных расстояний и среднюю сумму межкластерных расстояний для данного разбиения. Сделать вывод о качестве разбиения.

7) Провести кластеризацию выборки методом k-средних. для k [1, 10].

8) Сформировать три графика: зависимость средней суммы квадратов расстояний до центроида, средней суммы средних внутрикластерных расстояний и средней суммы межкластерных расстояний от количества кластеров. Исходя из результатов, выбрать оптимальное количество кластеров.

9) Составить сравнительную таблицу результатов разбиения иерархическим методом и методом k-средних.

Вариант 1

make_blobs(n_samples=100,random_state = 34, cluster_std=2.1, centers= 7, n_features = 2).

Ход работы:

Подключаем все возможные библиотеки, которые потребуется для выполнения лабораторной работы. Все данные лабораторной работы и ее выполнение рассматриваются на рисунках. Загрузим выборки согласно варианту.

```
#Лабораторная работа 5
#Вариант 1
# Подключение библиотек
from sklearn.datasets import make_blobs
import numpy as np
import pandas as pd
from scipy.cluster.hierarchy import linkage, dendrogram, fcluster
from sklearn.cluster import KMeans
from scipy.cluster.hierarchy import fcluster
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.metrics import pairwise_distances
X, y = make_blobs(n_samples=100,random_state = 34, cluster_std=2.1, centers= 7, n_features = 2)
plt.scatter(X[:, 0], X[:, 1])
```

Рисунок 1 – Выборки для 1 варианта

На основе наших данных изобразим график.

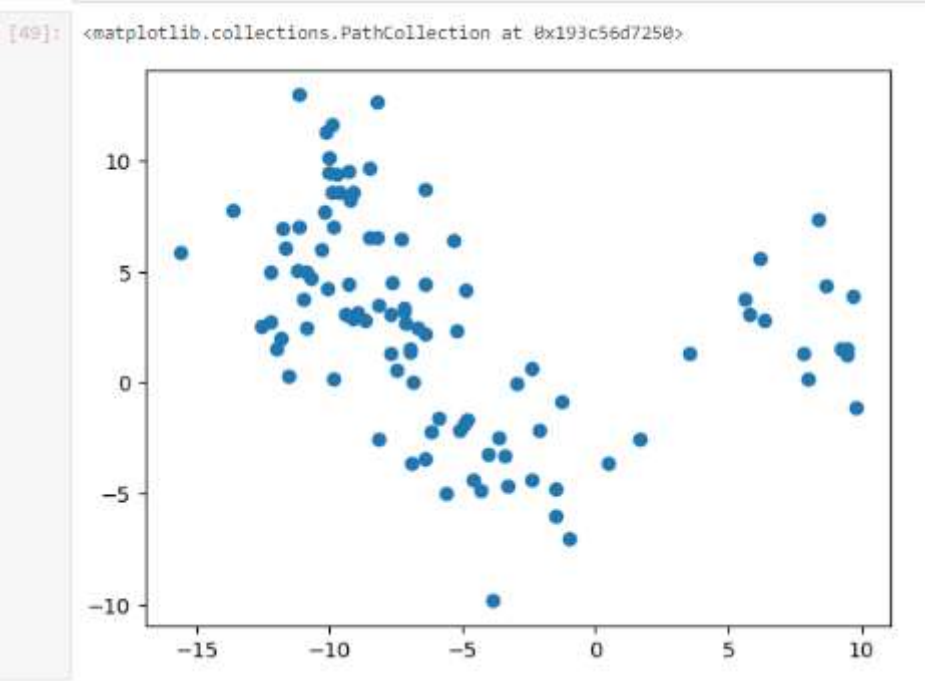


Рисунок 2 – График исходных данных

Через иерархическую кластеризации выборки воспользуемся методами (ближайших соседей, дальнего соседа, Уорда. Изобразим для каждого метода дендограмму.

```
[50]: # Иерархическая кластеризация выборки
      #Ближайшего соседа
      mergings_single = linkage(X, method='single')
      mergings_single
      dendrogram(mergings_single)
      plt.show()
```

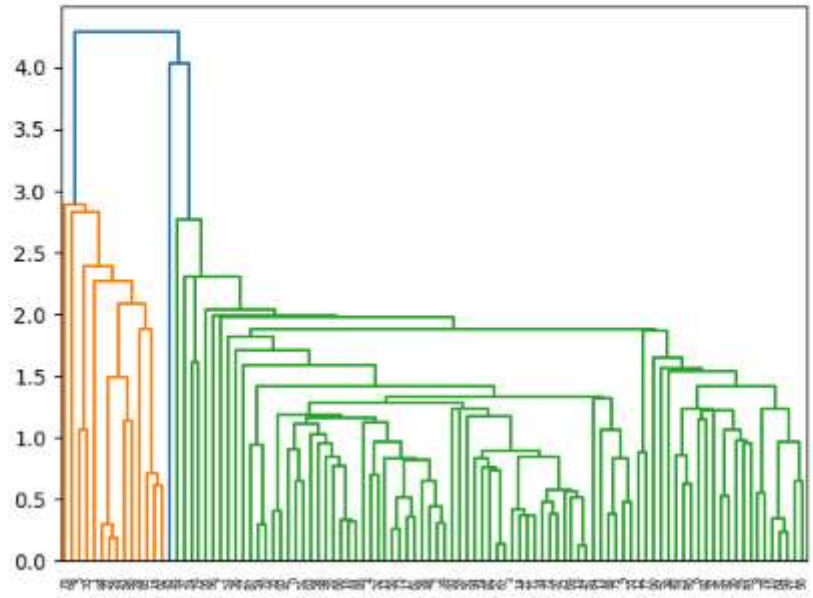


Рисунок 3 – Дендограмма метода ближайшего соседа

```
[51]: # Дальнего соседа
      mergings_complete = linkage(X, method='complete')
      mergings_complete
      dendrogram(mergings_complete)
```

```
 'C3',
 'C3',
 'C3',
 'C3',
 'C3']}
```

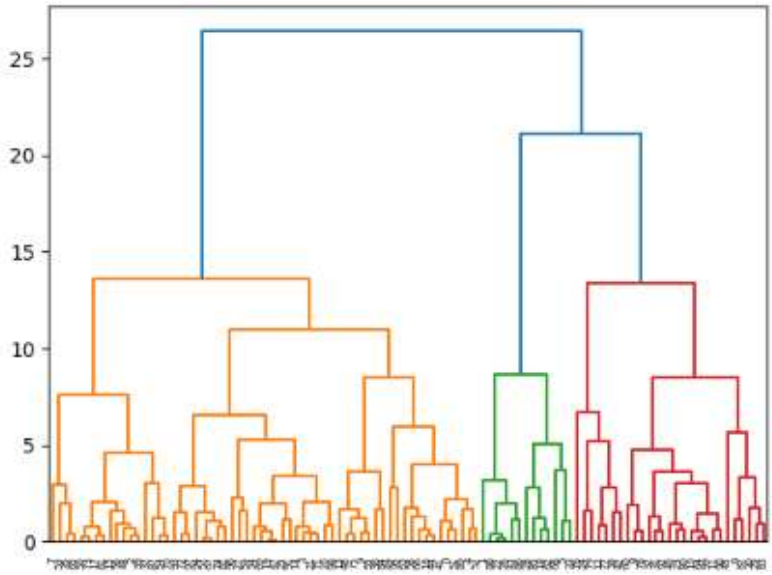


Рисунок 4 – Дендограмма метода дальнего соседа

```
[52]: # Метод Уорда
mergings_ward = linkage(X, method='ward')
mergings_ward
dendrogram(mergings_ward)
plt.show()
```

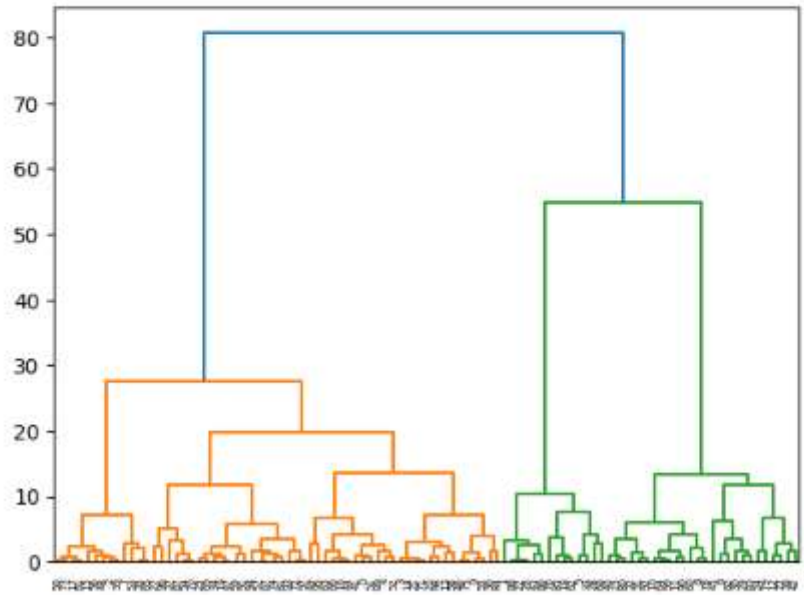


Рисунок 5 – Дендограмма метода Уорда

Выбирается из методов лучшая дендограмма.

```
[53]: # Выбор дендограммы лучшего разбиения
mergings_best = linkage(X, method='complete')
dendrogram(mergings_best)
plt.title("Дендограмма лучшая")
plt.show()
```

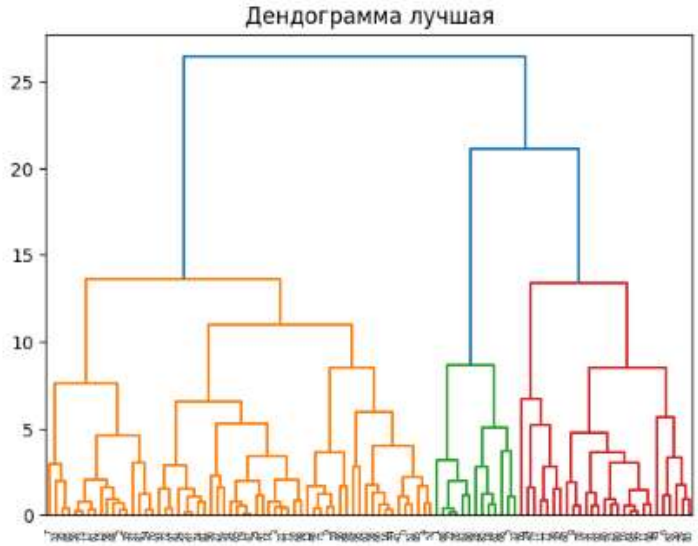


Рисунок 6 – Дендограмма лучшая метода дальнего соседа

Прописывается код для обновления центров кластеров и использование таких методов как k-средних и метода локтя, для определения оптимального количества кластеров.

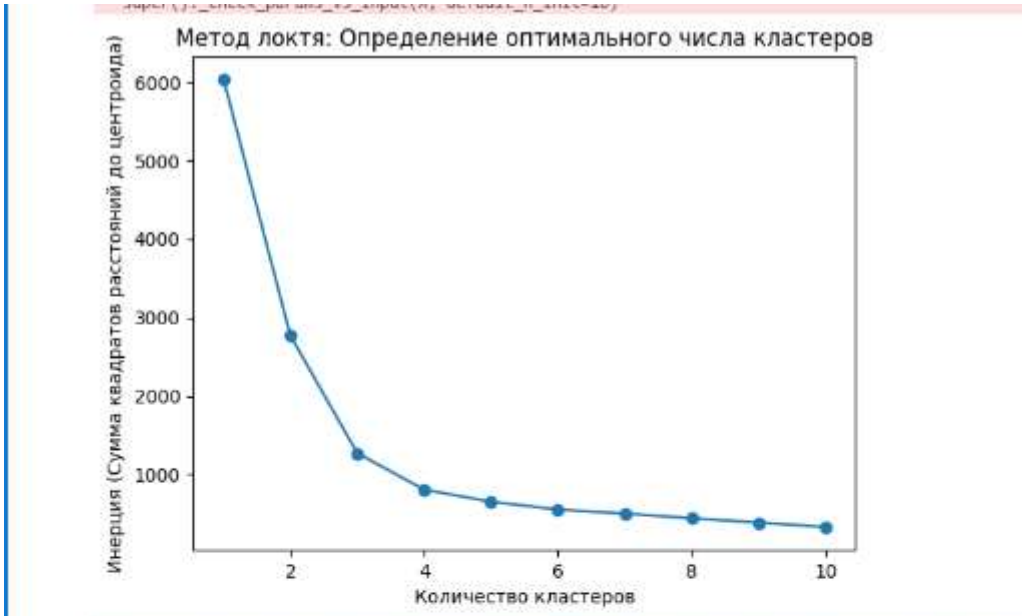
```
[54]: T = fcluster(mergings_best, 10, 'distance')
print("Назначения иерархической кластеризации:", T)
kmeans_optimal = KMeans(n_clusters=optimal_k, n_init=None)
# Функция обновления центров кластеров
def update_cluster_centers(X, c):
    mu = np.zeros((len(np.unique(c)), X.shape[1]))
    for cluster_id in np.unique(c):
        ix = np.where(c == cluster_id)
        mu[cluster_id - 1, :] = np.mean(X[ix, :], axis=(0, 1))
    return mu

# Обновление центров кластеров с использованием назначений иерархической кластеризации
mu = update_cluster_centers(X, T)
print("Обновленные центры кластеров:", mu)

inertia_values = []

#Метод локтя
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k)
    kmeans.fit(X)
    inertia_values.append(kmeans.inertia_)

# Построение графика метода локтя
plt.plot(range(1, 11), inertia_values, marker='o')
plt.title("Метод локтя: Определение оптимального числа кластеров")
plt.xlabel("Количество кластеров")
plt.ylabel("Инерция (Сумма квадратов расстояний до центроида)")
plt.show()
# Выбор оптимального числа кластеров
optimal_k = np.argmin(inertia_values) + 1
kmeans_optimal = KMeans(n_clusters=optimal_k, n_init=10)
# Кластеризация методом KMeans
kmeans_optimal = KMeans(n_clusters=optimal_k)
kmeans_optimal.fit(X)
predictions_optimal = kmeans_optimal.predict(X)

# Отображение разбиения на кластеры и центроиды
plt.scatter(X[:, 0], X[:, 1], c=predictions_optimal, cmap='plasma')
plt.scatter(kmeans_optimal.cluster_centers_[:, 0], np.zeros_like(kmeans_optimal.cluster_centers_[:, 0]))
plt.title(f"K-средние")
plt.show()
```

Рисунок 7 – Код программы для построения графиков различных методов

Вывод результатов на рисунках 8,9.

```
Назначения иерархической кластеризации: [3 4 1 2 3 4 6 1 6 3 6 6 5 3 2 1 3 1 4 2 2 2 4 6 5 2 4 2 4 6 6 6 6 4 2 1 3
 1 5 1 6 3 5 4 3 6 3 2 1 3 1 3 3 1 1 3 1 2 3 2 6 1 4 3 6 1 3 2 4 2 1 2 2 1
 2 3 5 5 6 5 6 2 6 6 3 4 4 1 4 3 6 2 6 2 2 3 2 2 2 6]
Обновленные центры кластеров: [[ -9.1233554    9.36040385]
 [ -7.44868675   2.94751449]
 [-11.54634727   4.48243435]
 [  8.03840886   2.74756109]
 [ -0.43753808  -1.04143661]
 [ -4.37436773  -3.93866267]]
```

Рисунок 8 – Вывод результата значений




```
C:\Users\rewaz\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\cluster\_kmeans
will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warni
  super()._check_params_vs_input(X, default_n_init=10)
```

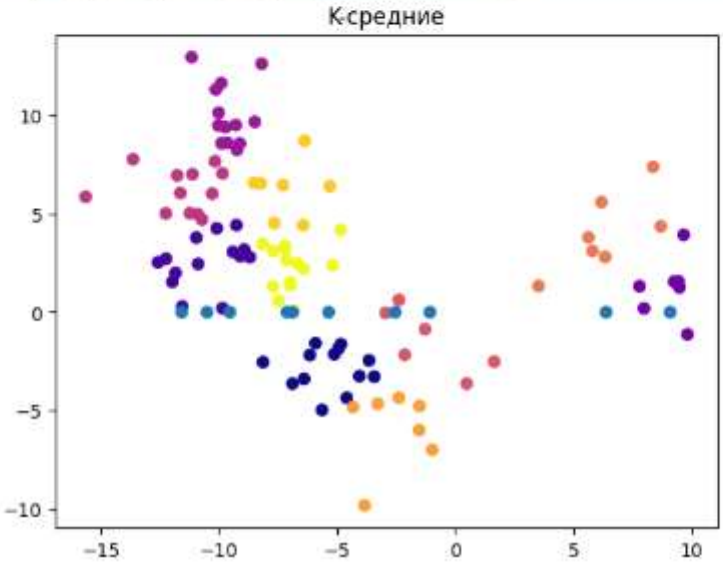


Рисунок 9 – Вывод результата в виде графиков

Далее записываем расчет сумм квадратов расстояний до центроида и суммы средних внутрикластерных и межкластерных расстояний.

```
[55]: def calculate_avg_intra_cluster_distance(X, labels, metric='euclidean'):
          total_distance = 0
          num_clusters = len(np.unique(labels))
          for cluster_id in np.unique(labels):
              cluster_points = X[labels == cluster_id]
              if len(cluster_points) > 1:
                  total_distance += np.sum(pairwise_distances(cluster_points, metric=metric))
          return total_distance / (len(X) - num_clusters)

      # Средняя сумма межкластерных расстояний
      def calculate_avg_inter_cluster_distance(X, labels, metric='euclidean'):
          cluster_centers = np.array([np.mean(X[labels == cluster_id], axis=0) for cluster_id in np.unique(labels)])
          return np.sum(pairwise_distances(cluster_centers, metric=metric))

      # Расчет средних сумм расстояний для различных значений k
      avg_intra_distances = []
      avg_inter_distances = []

      for k in range(1, 11):
          kmeans = KMeans(n_clusters=k)
          kmeans.fit(X)
          labels = kmeans.labels_

          avg_intra_distance = calculate_avg_intra_cluster_distance(X, labels)
          avg_inter_distance = calculate_avg_inter_cluster_distance(X, labels)

          avg_intra_distances.append(avg_intra_distance)
          avg_inter_distances.append(avg_inter_distance)


      columns = pd.MultiIndex.from_product([['Иерархический метод', 'Метод k-средних'],
                                            ['Сумма квадратов расстояний до центроида',
                                             'Сумма средних внутрикластерных расстояний',
                                             'Сумма межкластерных расстояний']])

      # Создание пустого DataFrame
      df = pd.DataFrame(columns=columns)

      # Добавление значений в таблицу
      df['Метод k-средних', 'Сумма квадратов расстояний до центроида'] = [inertia_values[optimal_k - 1]]
      df['Метод k-средних', 'Сумма средних внутрикластерных расстояний'] = [avg_intra_distances[optimal_k - 1]]
      df['Метод k-средних', 'Сумма межкластерных расстояний'] = [avg_inter_distances[optimal_k - 1]]

      df['Метод k-средних', 'Сумма квадратов расстояний до центроида'] = [inertia_values[optimal_k - 1]]
      df['Метод k-средних', 'Сумма средних внутрикластерных расстояний'] = [avg_intra_distances[optimal_k - 1]]
      df['Метод k-средних', 'Сумма межкластерных расстояний'] = [avg_inter_distances[optimal_k - 1]]

      # Вывод
      print(df)
```

Рисунок 10 – Код для расчета и вывода сумм

```
                    Иерархический метод  \
  Сумма квадратов расстояний до центроида
0                                   NaN

                                                      \
  Сумма средних внутрикластерных расстояний Сумма межкластерных расстояний
0                                       NaN                            NaN

                         Метод k-средних  \
  Сумма квадратов расстояний до центроида
0                              329.929481

  Сумма средних внутрикластерных расстояний Сумма межкластерных расстояний
0                                 30.336777                     1038.819314
```

Рисунок 11 – Составление таблицы сумм

Вывод

В ходе выполнения лабораторной работы были использованы методы кластеризации: иерархического метода (ближайших соседей, дальнего соседа, Уорда), более оптимальная дендограмма получилось у дальнего соседа метода. При методе средних провели кластеризацию данных и изобразили график оптимальной кластеризации, оптимальное количество от 3 до 4.