Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №4

по дисциплине «Прикладные интеллектуальные системы и экспертные системы»

Кластеризация данных

Студент Посаднев В.В.

Группа М-ИАП-22-1

Руководитель Кургасов В.В.

Доцент

Липецк 2022 г.

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 дендограммы для каждого способа. Размер графика должен быть подобран таким образом, чтобы дендограмма хорошо читалась.

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

5) Для выбранного способа, исходя из дендограммы, 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 для 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 №11

n_samples = 100

Вид классов: classification

Random_state = 5

class_sep = 1.5

n_features = 2

n_redundant = 0

n_informative = 2

n_cluster_per_class = 1

n_classes = 4

Ход работы

Загрузим выборки согласно варианту №11, код для генерации данных представлен на рисунке 1.

```
X, y = make_classification(n_samples=100,
                           n_features=2,
                           n_redundant=0,
                           n_informative=2,
                           n_clusters_per_class=1,
                           n_classes=4,
                           random_state=5,
                           class_sep=1.5)
```

Рисунок 1 – Код для генерации данных

Отобразим на графике сгенерированные данные, для этого воспользуемся библиотекой matplotlib.pyplot. Полученный график представлен на рисунке 2.

```
plt.scatter(X[:, 0], X[:, 1])
```

```
<matplotlib.collections.PathCollection at 0x23816fe7dc0>
```

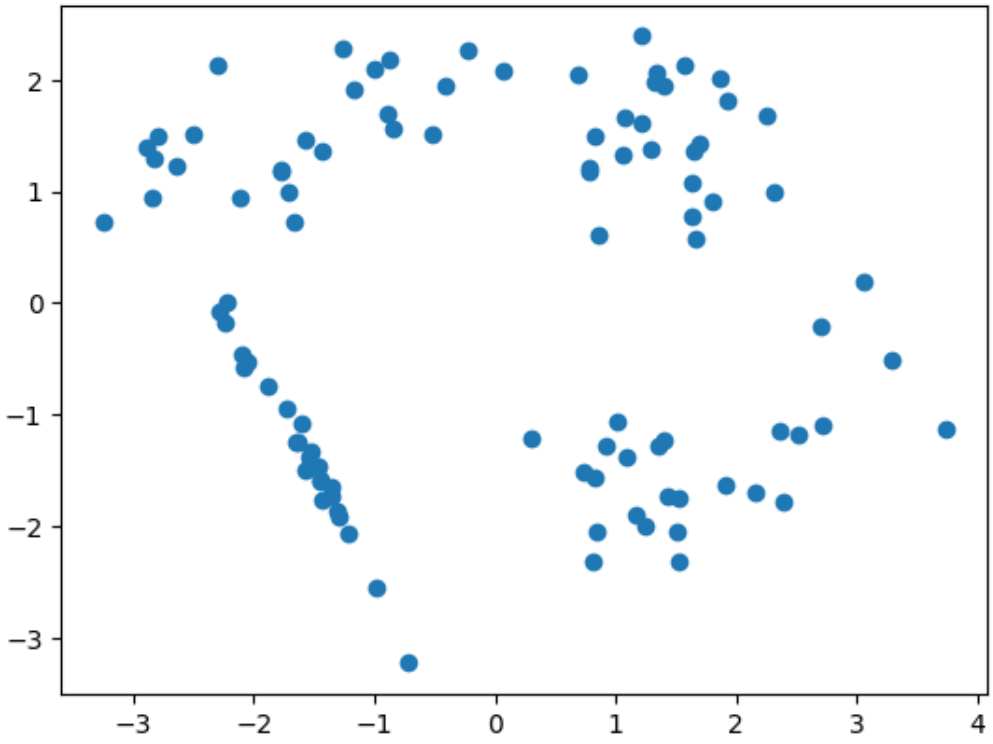

Рисунок 2 – Визуализация сгенерированных данных

Воспользуемся иерархической кластеризацией выборки с использованием различных методов вычисления расстояния.

Метод вычисления расстояния ближайшего соседа (single) и код для этого представлен на рисунке 3. Полученная дендограмма представлена на рисунке 4.

```
mergings_single = linkage(X, method='single')
mergings_single
```

```
array([[4.30000000e+01, 7.40000000e+01, 1.04010734e-02, 2.00000000e+00],
       [4.00000000e+01, 5.00000000e+01, 1.12542017e-02, 2.00000000e+00],
       [4.60000000e+01, 4.90000000e+01, 1.83059535e-02, 2.00000000e+00],
       [4.20000000e+01, 8.90000000e+01, 2.85527765e-02, 2.00000000e+00],
       [5.30000000e+01, 9.70000000e+01, 5.04282799e-02, 2.00000000e+00],
       [4.40000000e+01, 7.70000000e+01, 5.33827361e-02, 2.00000000e+00],
       [3.00000000e+00, 6.90000000e+01, 6.17673195e-02, 2.00000000e+00],
       [3.10000000e+01, 9.90000000e+01, 6.72403955e-02, 2.00000000e+00],
       [1.20000000e+01, 2.40000000e+01, 7.32365299e-02, 2.00000000e+00],
       [1.10000000e+01, 6.80000000e+01, 8.39490552e-02, 2.00000000e+00],
       [6.10000000e+01, 9.00000000e+01, 8.49616595e-02, 2.00000000e+00],
       [2.00000000e+00, 1.08000000e+02, 9.09643390e-02, 3.00000000e+00],
       [1.30000000e+01, 1.06000000e+02, 9.13774467e-02, 3.00000000e+00],
       [0.00000000e+00, 1.10000000e+02, 9.20819789e-02, 3.00000000e+00],
```

Рисунок 3 – Вычисленные расстояния ближайшего соседа (single)

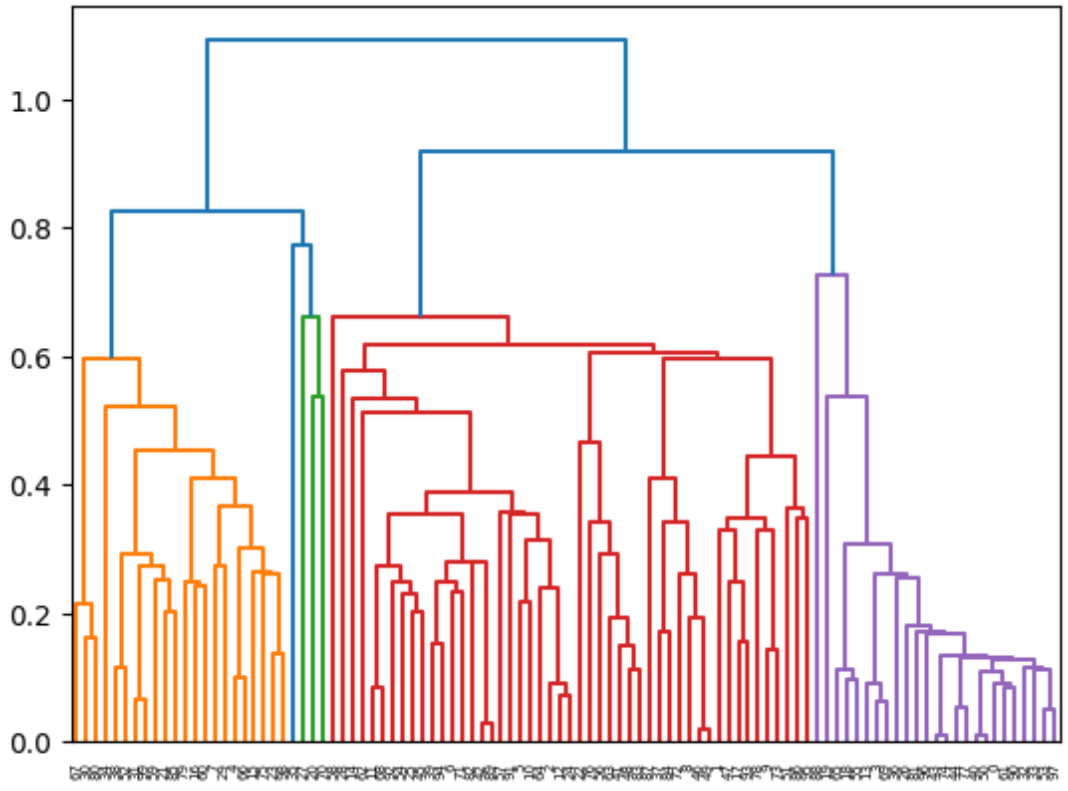

Рисунок 4 – Дендограмма для расстояния ближайшего соседа (single)

Метод вычисления расстояния дальнего соседа (complete) и код для этого представлен на рисунке 5. Полученная дендограмма представлена на рисунке 6.

```
mergings_complete = linkage(X, method='complete')
mergings_complete
```

```
array([[4.30000000e+01, 7.40000000e+01, 1.04010734e-02, 2.00000000e+00],
       [4.00000000e+01, 5.00000000e+01, 1.12542017e-02, 2.00000000e+00],
       [4.60000000e+01, 4.90000000e+01, 1.83059535e-02, 2.00000000e+00],
       [4.20000000e+01, 8.90000000e+01, 2.85527765e-02, 2.00000000e+00],
       [5.30000000e+01, 9.70000000e+01, 5.04282799e-02, 2.00000000e+00],
       [4.40000000e+01, 7.70000000e+01, 5.33827361e-02, 2.00000000e+00],
       [3.00000000e+00, 6.90000000e+01, 6.17673195e-02, 2.00000000e+00],
       [3.10000000e+01, 9.90000000e+01, 6.72403955e-02, 2.00000000e+00],
       [1.20000000e+01, 2.40000000e+01, 7.32365299e-02, 2.00000000e+00],
       [1.10000000e+01, 6.80000000e+01, 8.39490552e-02, 2.00000000e+00],
       [6.10000000e+01, 9.00000000e+01, 8.49616595e-02, 2.00000000e+00],
       [1.80000000e+01, 5.50000000e+01, 9.83732271e-02, 2.00000000e+00],
       [4.00000000e+00, 6.60000000e+01, 9.88681987e-02, 2.00000000e+00],
       [4.80000000e+01, 8.30000000e+01, 1.11845503e-01, 2.00000000e+00],
```

Рисунок 5 – Вычисленные расстояния дальнего соседа (complete)

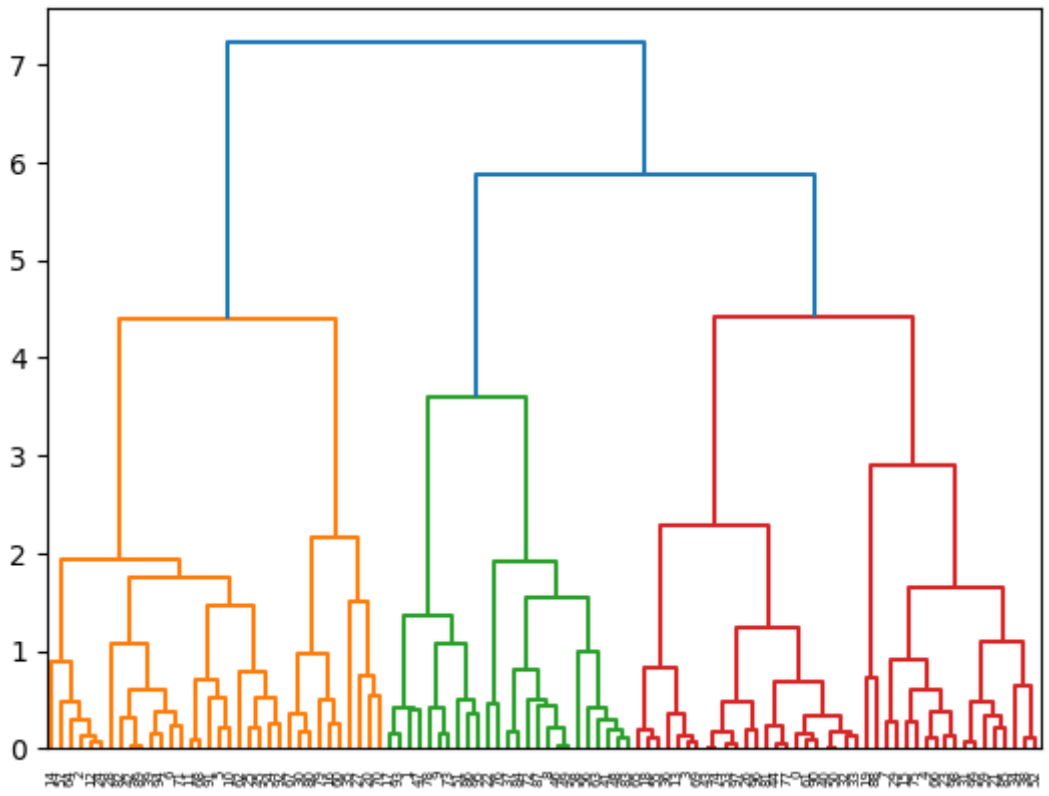

Рисунок 6 – Дендограмма для расстояния дальнего соседа (complete)

Метод вычисления расстояния Уорда (ward) и код для этого представлен на рисунке 7. Полученная дендограмма представлена на рисунке 8.

```
mergings_ward = linkage(X, method='ward')
mergings_ward
```

```
array([[4.30000000e+01, 7.40000000e+01, 1.04010734e-02, 2.00000000e+00],
       [4.00000000e+01, 5.00000000e+01, 1.12542017e-02, 2.00000000e+00],
       [4.60000000e+01, 4.90000000e+01, 1.83059535e-02, 2.00000000e+00],
       [4.20000000e+01, 8.90000000e+01, 2.85527765e-02, 2.00000000e+00],
       [5.30000000e+01, 9.70000000e+01, 5.04282799e-02, 2.00000000e+00],
       [4.40000000e+01, 7.70000000e+01, 5.33827361e-02, 2.00000000e+00],
       [3.00000000e+00, 6.90000000e+01, 6.17673195e-02, 2.00000000e+00],
       [3.10000000e+01, 9.90000000e+01, 6.72403955e-02, 2.00000000e+00],
       [1.20000000e+01, 2.40000000e+01, 7.32365299e-02, 2.00000000e+00],
       [1.10000000e+01, 6.80000000e+01, 8.39490552e-02, 2.00000000e+00],
       [6.10000000e+01, 9.00000000e+01, 8.49616595e-02, 2.00000000e+00],
       [1.80000000e+01, 5.50000000e+01, 9.83732271e-02, 2.00000000e+00],
       [4.00000000e+00, 6.60000000e+01, 9.88681987e-02, 2.00000000e+00],
       [4.80000000e+01, 8.30000000e+01, 1.11845503e-01, 2.00000000e+00],
```

Рисунок 7 – Вычисленные расстояния с помощью метода Уорда (ward)

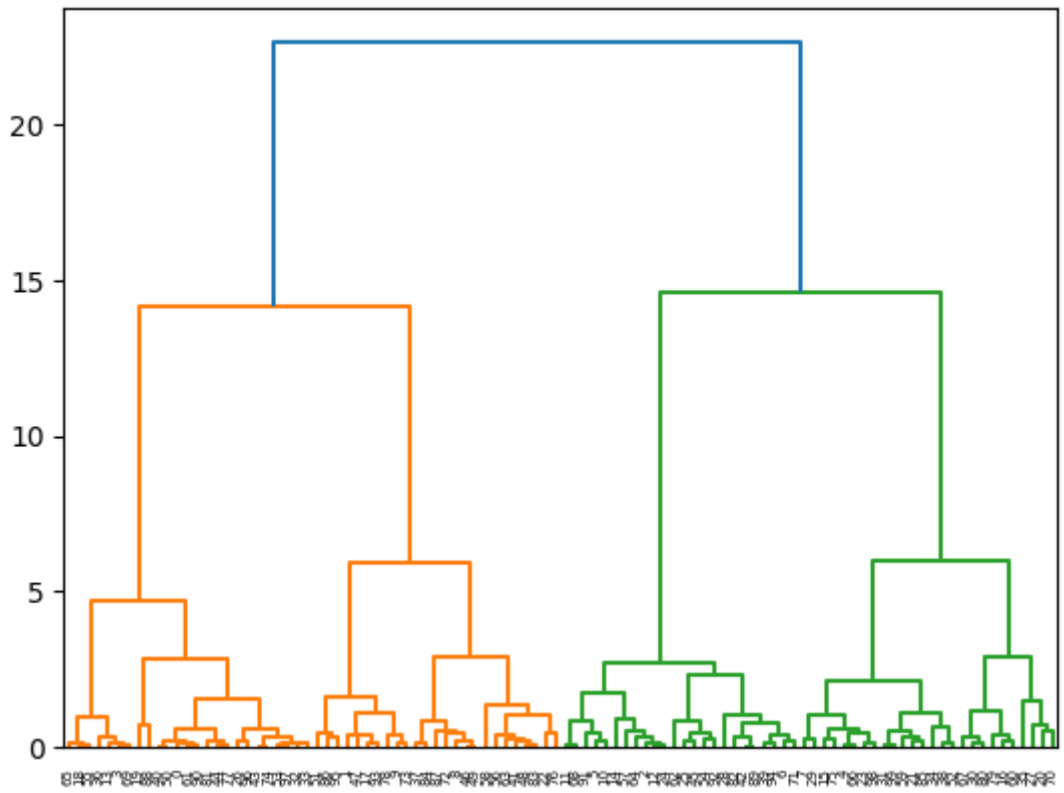

Рисунок 8 – Дендограмма для расстояния Уорда (ward)

Лучшим способом вычисления расстояния между кластерами является дальнего соседа (complete). Определим количество кластеров в имеющейся выборке с использованием данного способа и отобразим разбиение на кластеры и центроиды на графике в пространстве признаков. Полученное разбиение представлено на рисунке 9.

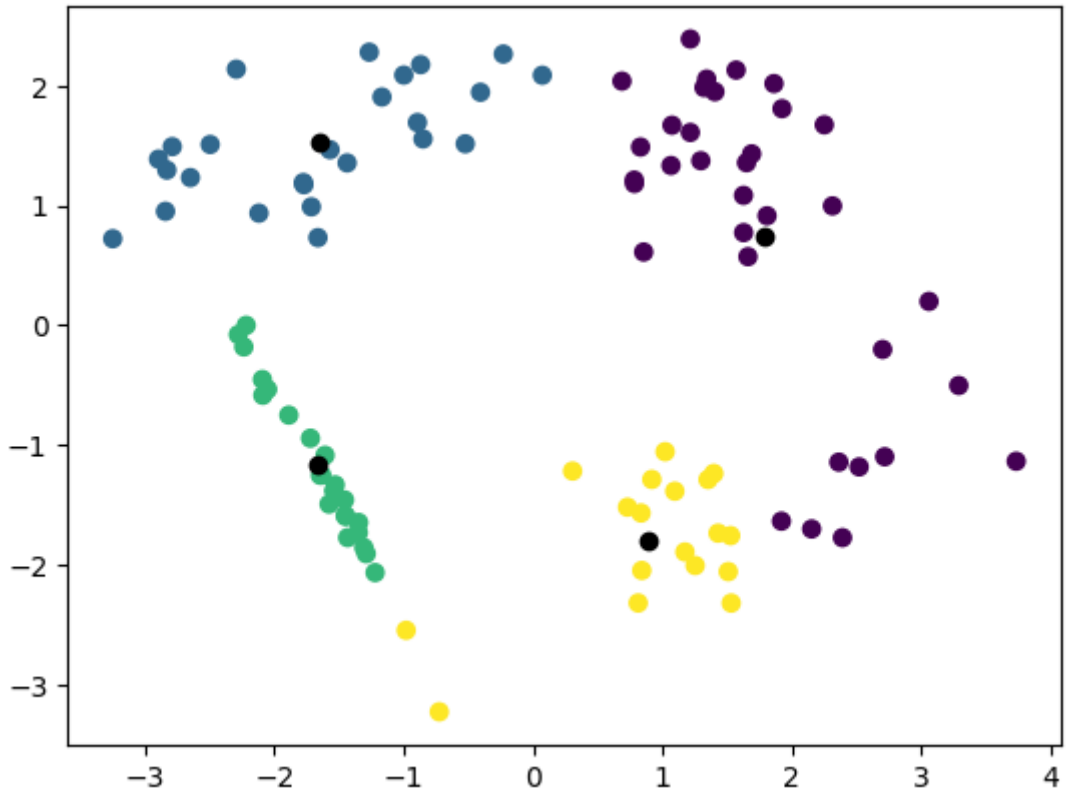

Рисунок 9 – График разбиения данных на кластеры

Рассчитаем среднюю сумму квадратов расстояний до центроида, среднюю сумму средних внутрикластерных расстояний и среднюю сумму межкластерных расстояний для данного разбиения.

Рассчитанное значение суммы квадратов расстояний до центроида и код для этого представлен на рисунке 10. Рассчитанное значение средних внутрикластерных расстояний и код для этого представлен на рисунке 11. Рассчитанное значение суммы межкластерных расстояний и код для этого представлен на рисунке 12.

```
# Сумма квадратов расстояний до центроида (inertia)
sum_sq_dist = np.zeros(4)
for i in range(1, 5):
    ix = np.where(T == i)
    sum_sq_dist[i - 1] = np.sum(euclidean_distances(*X[ix, :], [clusters[i - 1]]) ** 2)
sum_sq_dist = np.sum(sum_sq_dist) / 4
sum_sq_dist
```

```
31.087034614354266
```

Рисунок 10 – Сумма квадратов расстояний до центроида

```
# Сумма средних внутрикластерных расстояний
sum_avg_intercluster_dist = np.zeros(4)
for i in range(1, 5):
    ix = np.where(T == i)
    sum_avg_intercluster_dist[i - 1] = np.sum(euclidean_distances(*X[ix, :], [clusters[i - 1]]) ** 2) / len(*X[ix, :])
sum_avg_intercluster_dist = np.sum(sum_avg_intercluster_dist) / 4
sum_avg_intercluster_dist
```

```
1.1148321806219426
```

Рисунок 11 – Сумма средних внутрикластерных расстояний

```
# Сумма межкластерных расстояний
sum_intercluster_dist = np.sum(euclidean_distances(clusters, clusters))
sum_intercluster_dist
```

```
39.37587131636027
```

Рисунок 12 – Сумма межкластерных расстояний

Проведем кластеризацию выборки методом k-средних для k = [1, 10], а после построим три графика: зависимость средней суммы квадратов расстояний до центроида, средней суммы средних внутрикластерных расстояний и средней суммы межкластерных расстояний от количества кластеров.

Код для расчета средней суммы квадратов расстояний до центроида представлен на рисунке 13, а построенный график на рисунке 14.

```
# Средней суммы квадратов расстояний до центроида
sum_sq_dist_avg = []
for it, kmean in enumerate(models):
    sum_sq_dist_avg.append(kmean.inertia_ / (it + 1))
sum_sq_dist_avg
```

```
[545.7162442721157,
 144.93385251944756,
 54.50937795574206,
 20.56516779612909,
 12.702615089960599,
 7.573488639064396,
 5.197171687175967,
 4.034885804323963,
 3.1414549568038117,
 2.50427153253386]
```

Рисунок 13 – Код для вычисления средней суммы квадратов расстояний до центроида

```
plt.plot(range(1, 11), sum_sq_dist_avg, '-o')
```

```
[<matplotlib.lines.Line2D at 0x23817a7ef80>]
```

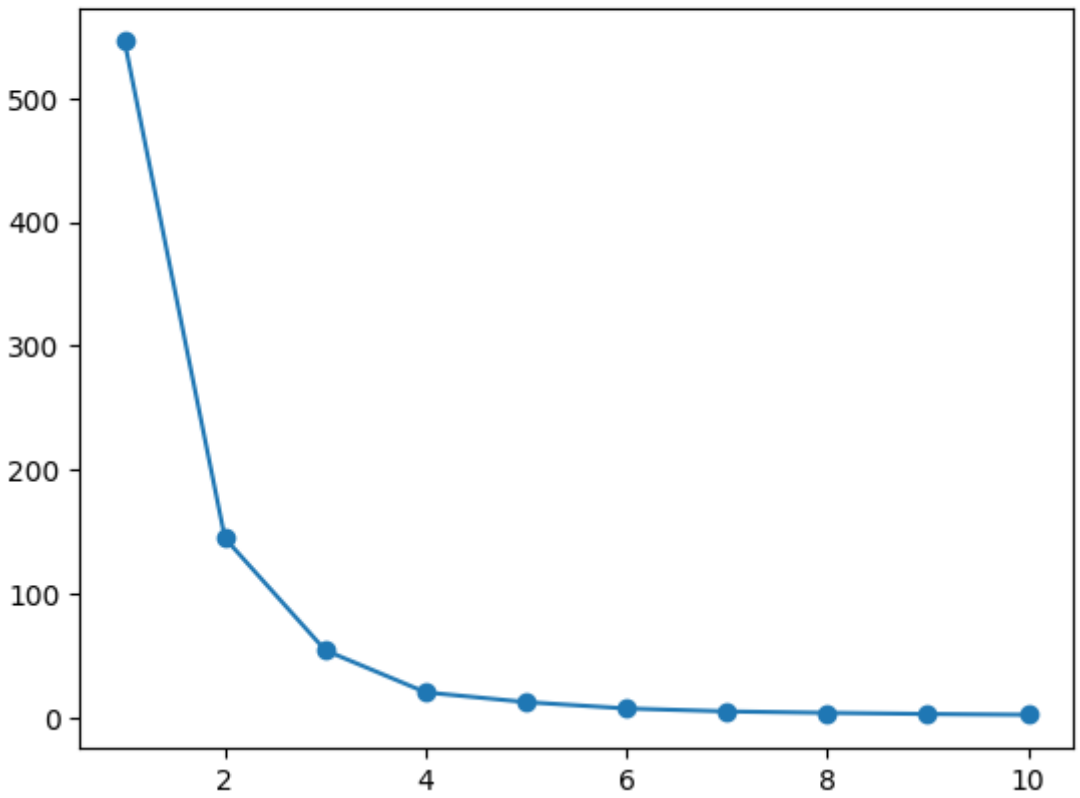

Рисунок 14 – График средней суммы квадратов расстояний до центроида

Код для расчета средней суммы средних внутрикластерных расстояний представлен на рисунке 15, а построенный график на рисунке 16.

```
# Средней суммы средних внутрикластерных расстояний
new_centers = [kmean.cluster_centers_ for kmean in models]

sum_avg_intercluster_dist_avg = []
for k, kmean in enumerate(models):
    intercluster_sum = np.zeros(4)
    for i in range(4):
        ix = np.where(predicted_values[k] == i)
        if len(ix[0]) == 0:
            intercluster_sum[i - 1] = 0
        else:
            intercluster_sum[i - 1] = np.sum(euclidean_distances(*X[ix, :], [kmean.cluster_centers_[i - 1]]) ** 2) / len(*X[ix, :])
    sum_avg_intercluster_dist_avg.append(np.sum(intercluster_sum) / (k + 1))
sum_avg_intercluster_dist_avg
```

```
[5.457162442721157,
 13.134942139552361,
 13.21168207036577,
 14.25587147571919,
 10.089408399483421,
 11.172896629070635,
 7.346492245325464,
 7.423014177427671,
 10.74275476303583,
 4.536319517477247]
```

Рисунок 15 – Код для вычисления средней суммы средних внутрикластерных расстояний

```
plt.plot(range(1, 11), sum_avg_intercluster_dist_avg, '-o')
```

```
[<matplotlib.lines.Line2D at 0x2381ac67dc0>]
```

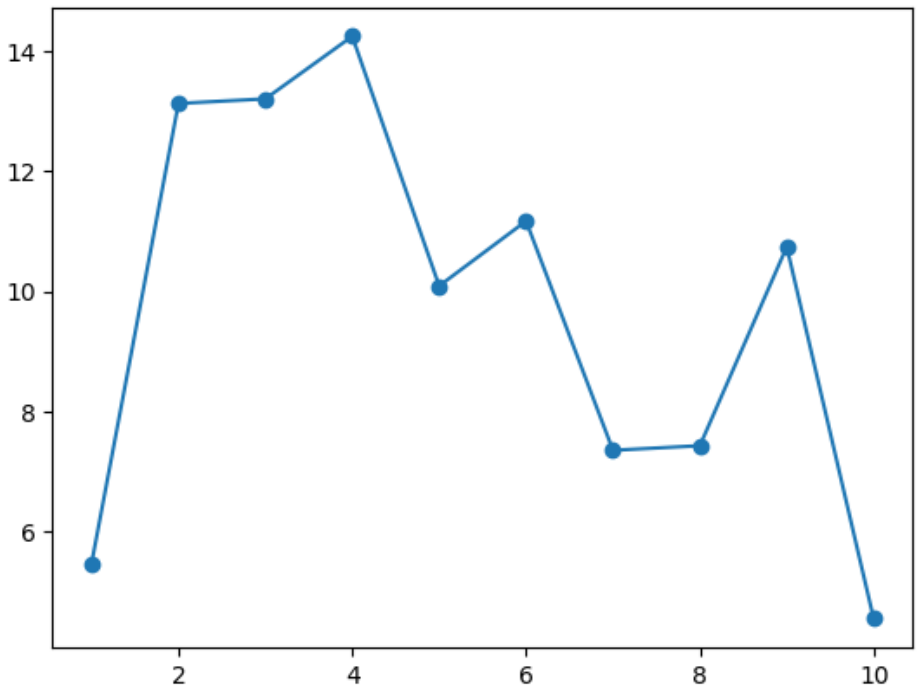

Рисунок 16 – График средней суммы средних внутрикластерных расстояний

Код для расчета средней суммы межкластерных расстояний от количества кластеров представлен на рисунке 17, а построенный график на рисунке 18.

```
# Средней суммы межкластерных расстояний от количества кластеров
sum_intercluster_dist_avg = []

for k, kmean in enumerate(models):
    value = np.sum(euclidean_distances(kmean.cluster_centers_, kmean.cluster_centers_))
    sum_intercluster_dist_avg.append(value / (k + 1))
sum_intercluster_dist_avg
```

```
[0.0,
 3.1996932328025496,
 6.801441493599005,
 10.388727979540153,
 13.252244338636618,
 17.165246982909057,
 20.174583394404824,
 22.952887415646607,
 26.05386642784076,
 28.95120558692425]
```

Рисунок 17 – Код для вычисления средней суммы межкластерных расстояний от количества кластеров

```
plt.plot(range(1, 11), sum_intercluster_dist_avg, '-o')
```

```
[<matplotlib.lines.Line2D at 0x2381acdbc70>]
```

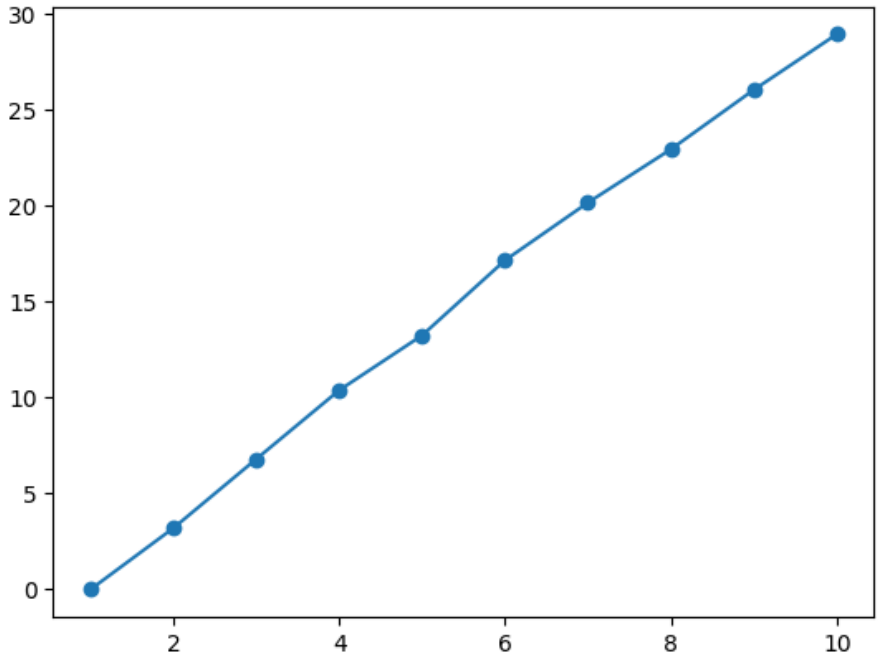

Рисунок 18 – График средней суммы межкластерных расстояний от количества кластеров

Как видно из рисунка 14 оптимальное количество кластеров составляет от двух до четырех.

Составим сравнительную таблицу для ранее описанных метрик качества моделей для иерархического метода и метода k-средних. Составленная таблица представлена на рисунках 19 – 21.

		Иерархический метод			Метод k-средних		
		Сумма квадратов расстояний до центроида	Сумма средних внутрикластерных расстояний	Сумма межкластерных расстояний	Сумма квадратов расстояний до центроида	Сумма средних внутрикластерных расстояний	Сумма межкластерных расстояний
0		31,08703461	1,114832181	39,37587132	545,7162443	5,457162443	0
1		31,08703461	1,114832181	39,37587132	144,9338525	13,13494214	3,199693233
2		31,08703461	1,114832181	39,37587132	54,50937796	13,21168207	6,801441494
3		31,08703461	1,114832181	39,37587132	20,5651678	14,25587148	10,38872798
4		31,08703461	1,114832181	39,37587132	12,70261509	10,0894084	13,25224434
5		31,08703461	1,114832181	39,37587132	7,573488639	11,17289663	17,16524698
6		31,08703461	1,114832181	39,37587132	5,197171687	7,346492245	20,17458339
7		31,08703461	1,114832181	39,37587132	4,034885804	7,423014177	22,95288742
8		31,08703461	1,114832181	39,37587132	3,141454957	10,74275476	26,05386643
9		31,08703461	1,114832181	39,37587132	2,504271533	4,536319517	28,95120559

Рисунок 19 – Сводная таблица сравнения

		Иерархический метод		
		Сумма квадратов расстояний до центроида	Сумма средних внутрикластерных расстояний	Сумма межкластерных расстояний
0		31,08703461	1,114832181	39,37587132
1		31,08703461	1,114832181	39,37587132
2		31,08703461	1,114832181	39,37587132
3		31,08703461	1,114832181	39,37587132
4		31,08703461	1,114832181	39,37587132
5		31,08703461	1,114832181	39,37587132
6		31,08703461	1,114832181	39,37587132
7		31,08703461	1,114832181	39,37587132
8		31,08703461	1,114832181	39,37587132
9		31,08703461	1,114832181	39,37587132

Рисунок 20 – Таблица данных для иерархического метода

Метод k-средних		
Сумма квадратов расстояний до центроида	Сумма средних внутрикластерных расстояний	Сумма межкластерных расстояний
545,7162443	5,457162443	0
144,9338525	13,13494214	3,199693233
54,50937796	13,21168207	6,801441494
20,5651678	14,25587148	10,38872798
12,70261509	10,0894084	13,25224434
7,573488639	11,17289663	17,16524698
5,197171687	7,346492245	20,17458339
4,034885804	7,423014177	22,95288742
3,141454957	10,74275476	26,05386643
2,504271533	4,536319517	28,95120559

Рисунок 21 – Таблица данных для метода k-средних

Заключение

В ходе выполнения данной лабораторной работы мною были получены навыки кластеризации данных. В рамках данной работы были применены различные методы кластеризации: иерархический метод и метод k-средних.

В ходе анализа метрик было определено, что оптимальное значение кластеров от двух до четырех. Также была составлена таблица сравнения метрик иерархическим методом кластеризации и методом k-средних.

Приложение А

Исходный код

```
#!/usr/bin/env python
# coding: utf-8

# # Вариант №11
# ---
# ### Вид классов: `classification`
# ### Random state: `5`
# ### Class sep: `1.5`
# ---
# ## Для всех:
# ### `n_features = 2`
# ### `n_redundant = 0`
# ### `n_informative = 2`
# ### `n_clusters_per_class = 1`
# ### `n_classes = 4`
# ### `n_samples = 100`

# In[1]:


from sklearn.datasets import make_classification


# # Загрузка выборки согласно варианту №11

# In[2]:


X, y = make_classification(n_samples=100,
                           n_features=2,
                           n_redundant=0,
                           n_informative=2,
                           n_clusters_per_class=1,
                           n_classes=4,
                           random_state=5,
                           class_sep=1.5)


# # Отображение выборки на графике

# In[3]:


import matplotlib.pyplot as plt


# In[4]:


plt.scatter(X[:, 0], X[:, 1])


# # Иерархическая кластеризация выборки

# In[5]:


from scipy.cluster.hierarchy import linkage, dendrogram
```

```
# ## Расстояние ближайшего соседа (single)

# In[6]:


mergings_single = linkage(X, method='single')
mergings_single


# In[7]:


dendrogram(mergings_single)
plt.show()


# ## Расстояние дальнего соседа (complete)

# In[8]:


mergings_complete = linkage(X, method='complete')
mergings_complete


# In[9]:


dendrogram(mergings_complete)
plt.show()


# ## Расстояние Уорда (Ward)

# In[10]:


mergings_ward = linkage(X, method='ward')
mergings_ward


# In[11]:


dendrogram(mergings_ward)
plt.show()


# # Выбор лучшего разбиения

# In[12]:


mergings_complete = linkage(X, method='complete')
mergings_complete


# In[13]:


dendrogram(mergings_complete)
plt.show()
```

```
# In[14]:


import numpy as np

def update_cluster_centers(X, c):
    centers = np.zeros((4, 2))
    for i in range(1, 5):
        ix = np.where(c == i)
        centers[i - 1, :] = np.mean(X[ix, :], axis=1)
    return centers


# In[15]:


from scipy.cluster.hierarchy import fcluster


# In[16]:


T = fcluster(mergings_complete, 4, criterion='maxclust')
clusters = update_cluster_centers(X, T)
clusters


# In[17]:


plt.scatter(X[:, 0], X[:, 1], c=T)
plt.scatter(clusters[:, 0], clusters[:, 1], c='black')


# # Вычисление характеристик

# In[18]:


from sklearn.metrics.pairwise import euclidean_distances


# In[19]:


# Сумма квадратов расстояний до центроида (inertia)
sum_sq_dist = np.zeros(4)
for i in range(1, 5):
    ix = np.where(T == i)
    sum_sq_dist[i - 1] = np.sum(euclidean_distances(*X[ix, :], [clusters[i -
1]]) ** 2)
sum_sq_dist = np.sum(sum_sq_dist) / 4
sum_sq_dist


# In[20]:


# Сумма средних внутрикластерных расстояний
sum_avg_intercluster_dist = np.zeros(4)
for i in range(1, 5):
```

```
    ix = np.where(T == i)
    sum_avg_intercluster_dist[i - 1] = np.sum(euclidean_distances(*X[ix, :],
[clusters[i - 1]]) ** 2) / len(*X[ix, :])
sum_avg_intercluster_dist = np.sum(sum_avg_intercluster_dist) / 4
sum_avg_intercluster_dist


# In[21]:


# Сумма межкластерных расстояний
sum_intercluster_dist = np.sum(euclidean_distances(clusters, clusters))
sum_intercluster_dist


# # Кластеризация выборки методом k-средних

# In[22]:


from sklearn.cluster import KMeans


# In[23]:


models = []
predicted_values = []

for k in range(1, 11):
    kmeans = KMeans(n_clusters=k)
    kmeans.fit(X)
    models.append(kmeans)
    predicted_values.append(kmeans.predict(X))


# In[24]:


# Средней суммы квадратов расстояний до центроида
sum_sq_dist_avg = []
for it, kmean in enumerate(models):
    sum_sq_dist_avg.append(kmean.inertia_ / (it + 1))
sum_sq_dist_avg


# In[25]:


plt.plot(range(1, 11), sum_sq_dist_avg, '-o')


# In[26]:


# Средней суммы средних внутрикластерных расстояний
new_centers = [kmean.cluster_centers_ for kmean in models]

sum_avg_intercluster_dist_avg = []
for k, kmean in enumerate(models):
    intercluster_sum = np.zeros(4)
    for i in range(4):
        ix = np.where(predicted_values[k] == i)
```

```
        if len(ix[0]) == 0:
            intercluster_sum[i - 1] = 0
        else:
            intercluster_sum[i - 1] = np.sum(euclidean_distances(*X[ix, :],
[kmean.cluster_centers_[i - 1]]) ** 2) / len(*X[ix, :])
    sum_avg_intercluster_dist_avg.append(np.sum(intercluster_sum) / (k + 1))
sum_avg_intercluster_dist_avg


# In[27]:


plt.plot(range(1, 11), sum_avg_intercluster_dist_avg, '-o')


# In[28]:


# Средней суммы межкластерных расстояний от количества кластеров
sum_intercluster_dist_avg = []

for k, kmean in enumerate(models):
    value = np.sum(euclidean_distances(kmean.cluster_centers_,
kmean.cluster_centers_))
    sum_intercluster_dist_avg.append(value / (k + 1))
sum_intercluster_dist_avg


# In[29]:


plt.plot(range(1, 11), sum_intercluster_dist_avg, '-o')


# # Составление сравнительной таблицы

# In[30]:


import pandas as pd


# In[31]:


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


# In[32]:


df['Иерархический метод', 'Сумма квадратов расстояний до центроида'] =
[sum_sq_dist for _ in range(len(sum_sq_dist_avg))]
df['Иерархический метод', 'Сумма средних внутрикластерных расстояний'] =
[sum_avg_intercluster_dist for _ in
range(len(sum_avg_intercluster_dist_avg))]
df['Иерархический метод', 'Сумма межкластерных расстояний'] =
```

```
[sum_intercluster_dist for _ in range(len(sum_intercluster_dist_avg))]

df['Метод k-средних', 'Сумма квадратов расстояний до центроида'] =
sum_sq_dist_avg
df['Метод k-средних', 'Сумма средних внутрикластерных расстояний'] =
sum_avg_intercluster_dist_avg
df['Метод k-средних', 'Сумма межкластерных расстояний'] =
sum_intercluster_dist_avg

df


# In[33]:


df.to_excel('result.xlsx')
```