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 Курдюков И.Ю.

Группы М-ИАП-23

Руководитель Кургасов В.В.

Доцент

Липецк 2023 г

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

Задание:

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 классунеизвестна, соответственно, на данном этапе все объекты на графике должныотображаться 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.

Ход работы

Вариант 10

n_features=2, n_redundant=0, n_informative=2, n_clusters_per_class=1, n_classes = 4

10
classification
68
1

Рисунок 1 - Вариант для выполнения

Генерация данных для варианта представлена на рисунке 2.

```
[2]  %load_ext rpy2.ipython
```

```
[3]  from sklearn.datasets import make_classification
```

```
[4]  # Генерация выборки с использованием make_classification
     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=68,
                                class_sep=1)
```

Рисунок 2 - Генерация данных

Отображение выборки на графике представлено на рисунке 3.

```
import matplotlib.pyplot as plt
```

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

```
<matplotlib.collections.PathCollection at 0x7f3f1a7d5960>
```

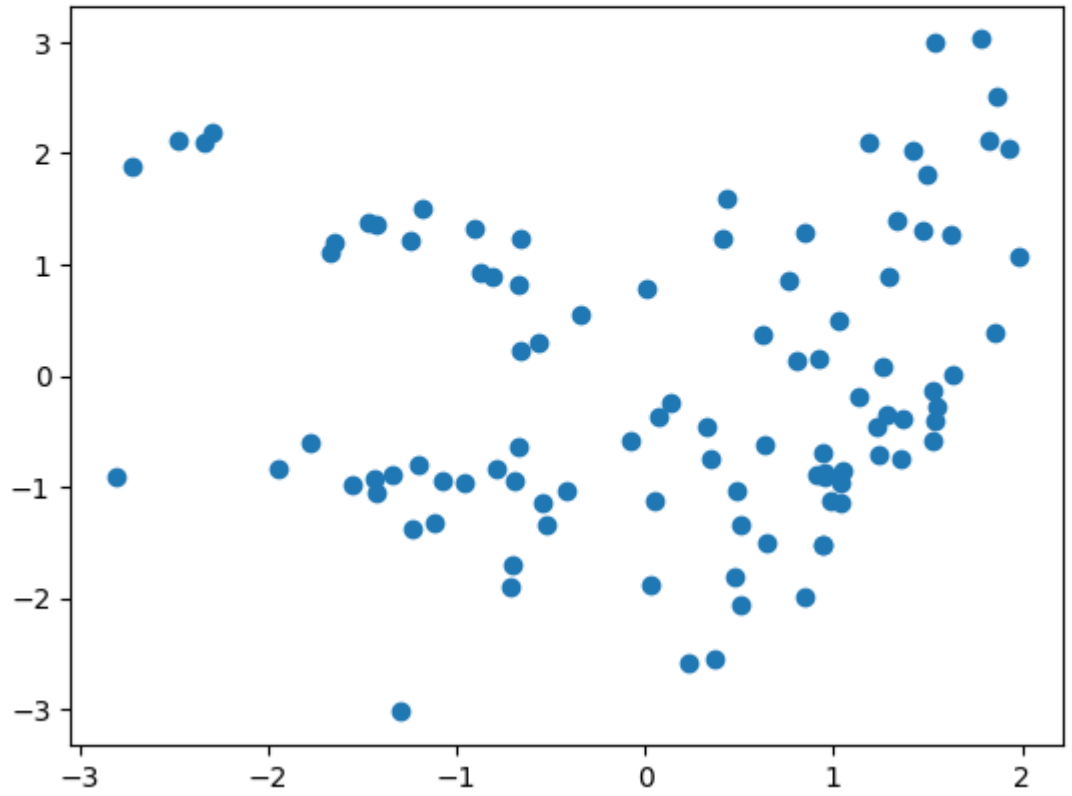

Рисунок 3 - Отображение выборки

```
from scipy.cluster.hierarchy import linkage, dendrogram
mergings_single = linkage(X, method='single')
mergings_complete = linkage(X, method='complete')
mergings_ward = linkage(X, method='ward')
# Расстояние ближайшего соседа (single)
fig, axes = plt.subplots(1, 3, figsize=(15, 5))
dendrogram(mergings_single, ax=axes[0])
axes[0].set_title('Расстояние ближайшего соседа')

# Расстояние дальнего соседа (complete)
dendrogram(mergings_complete, ax=axes[1])
axes[1].set_title('Расстояние дальнего соседа')

# Расстояние Уорда (Ward)
dendrogram(mergings_ward, ax=axes[2])
axes[2].set_title('Расстояние Уорда')
```

Графики иерархической кластеризации представлены на рисунке 4.

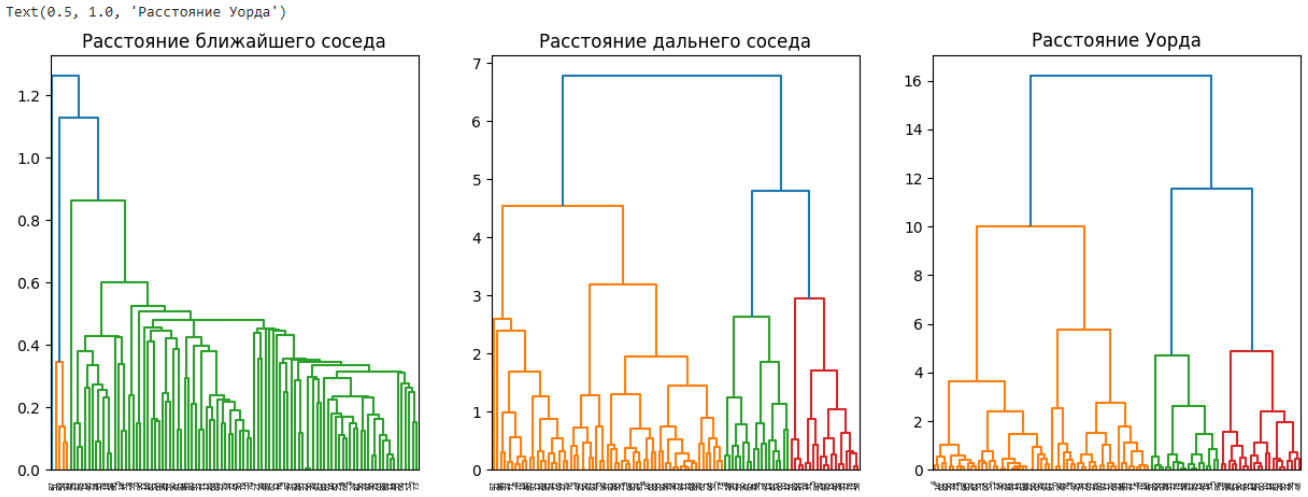

Рисунок 4 – Дендограммы

Выберем лучшее разбиение:

```
      mergings_ward = linkage(X, method='ward')
dendrogram(mergings_ward)
plt.show()
```

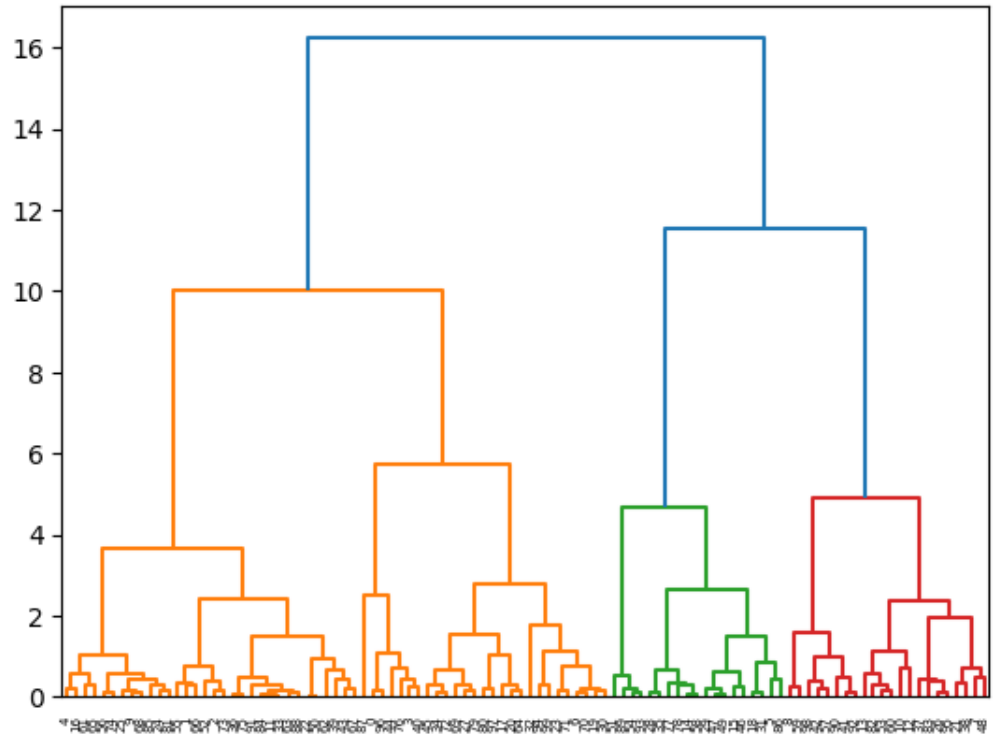

Рисунок 5 – Выбор лучшего разбиения

Лучшим способом вычисления расстояния между кластерами является расстояние Уорда (ward). 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 6.

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

```
<matplotlib.collections.PathCollection at 0x7f3f1a7d77f0>
```

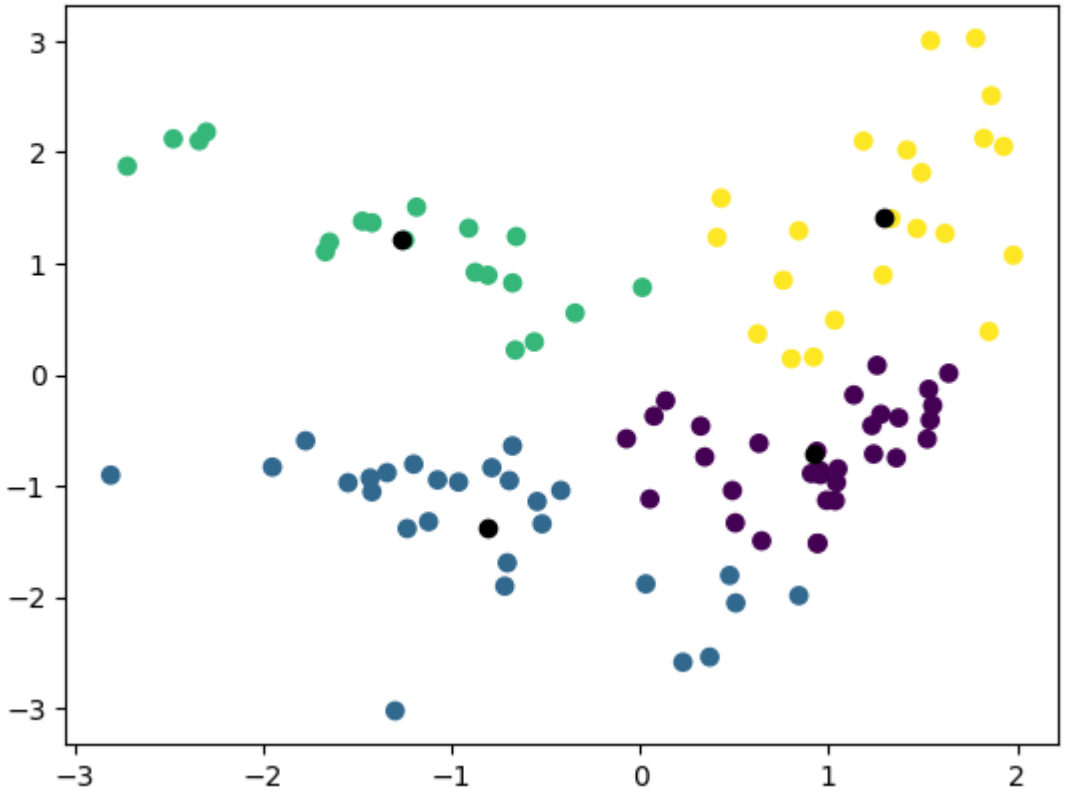

Рисунок 6 - График разбиения данных на кластеры

Рассчитаем среднюю сумму квадратов расстояний до центроида, среднюю сумму средних внутрикластерных расстояний и среднюю сумму межкластерных расстояний для данного разбиения. Расчеты представлены на рисунке 7.

```
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
```

```
19.849699014137087
```

```
[18] 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
```

```
0.8242211169214809
```

```
sum_intercluster_dist = np.sum(euclidean_distances(clusters, clusters))
sum_intercluster_dist
```

```
31.174884714925824
```

Рисунок 7 - Рассчитанные характеристики

Далее надо провести кластеризацию выборки методом k-средних. для k [1, 10]. Средняя сумма квадратов расстояний до центроида показана на рисунке 8.

```
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))
sum_sq_dist_avg = []
for it, kmean in enumerate(models):
    sum_sq_dist_avg.append(kmean.inertia_ / (it + 1))
sum_sq_dist_avg
plt.plot(range(1, 11), sum_sq_dist_avg, '-o')
```

```
sum_sq_dist_avg = []
for it, kmean in enumerate(models):
    sum_sq_dist_avg.append(kmean.inertia_ / (it + 1))
sum_sq_dist_avg
```

```
[328.13400628727373,
 98.02399884235895,
 40.42551871994463,
 18.262170939861747,
 11.321752642149097,
 7.758692865366876,
 5.5419216903797945,
 4.091394200764297,
 3.196492943973906,
 2.4775413679424085]
```

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

```
[<matplotlib.lines.Line2D at 0x7f3f14081540>]
```

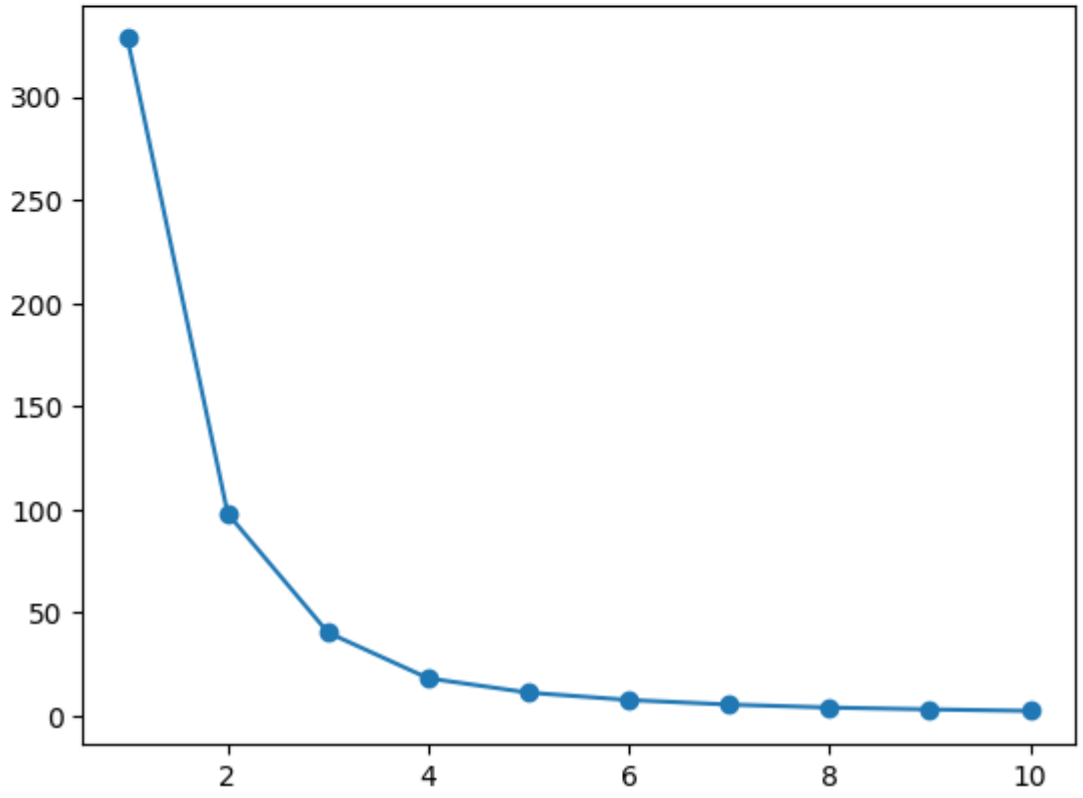

Рисунок 8 - Сумма квадратов расстояний до центроида

Средняя сумма средних внутрикластерных расстояний показана на рисунке 9.

```
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
```

```
[3.281340062872738,
 7.620718871931414,
 7.953389142975081,
 9.176142127523182,
 5.605547249205628,
 5.06629128136665,
 4.682204760635001,
 4.273150773273448,
 3.9870953479982703,
 1.872186966810793]
```

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

```
[<matplotlib.lines.Line2D at 0x7f3f140f2b60>]
```

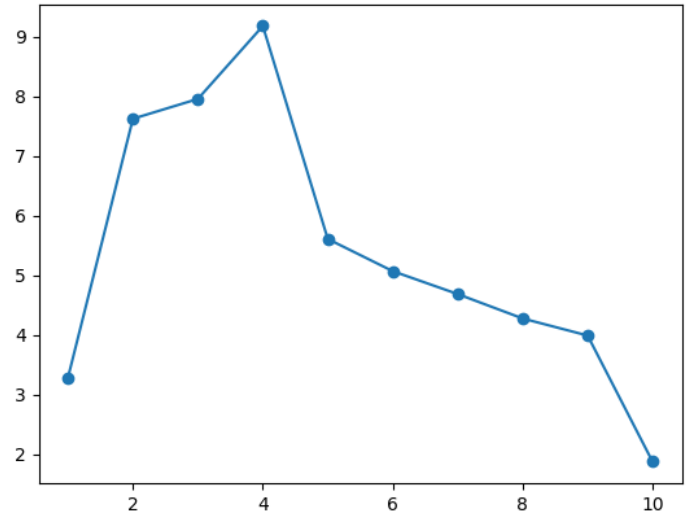

Рисунок 9 - Средняя сумма средних внутрикластерных расстояний

Средняя сумма средних межкластерных расстояний от количества кластеров показана на рисунке 10.

```
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,
 2.356303277746618,
 5.176504438492942,
 8.115744371552355,
 10.449064098026556,
 14.110567159282809,
 15.99648308206915,
 20.026943846332863,
 21.65471534655876,
 24.360636611553335]
```

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

```
[<matplotlib.lines.Line2D at 0x7f3f13f90b20>]
```

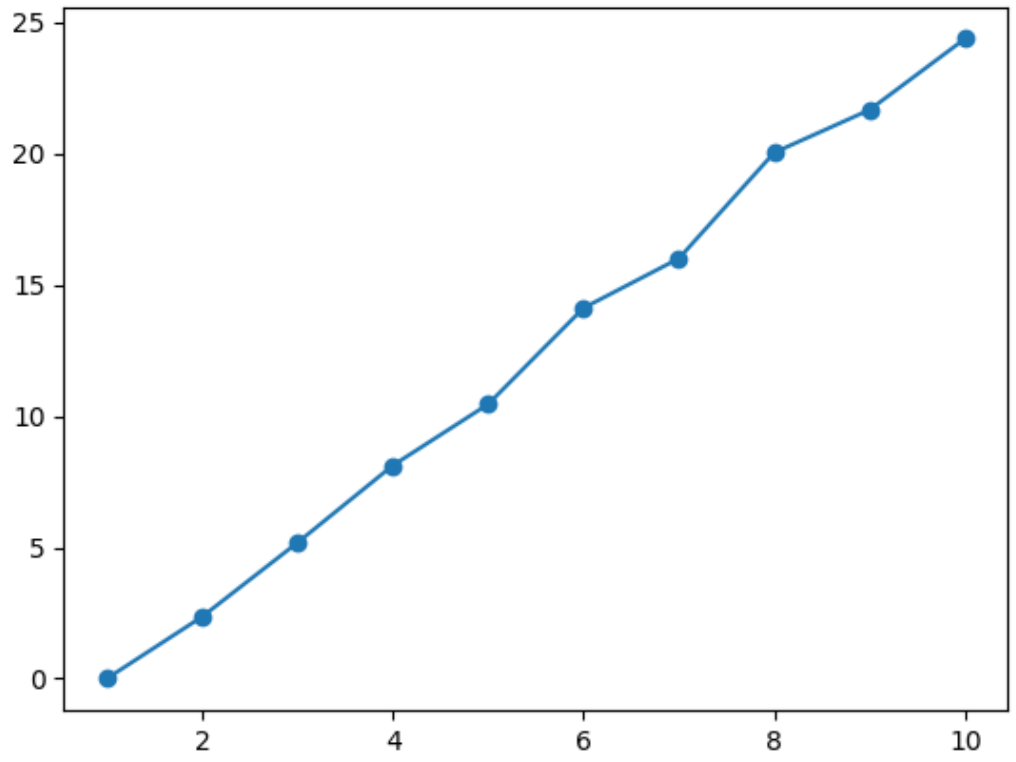

Рисунок 10 - Средняя сумма средних межкластерных расстояний от количества кластеров

Составим сравнительную таблицу результатов разбиения иерархическим методом и методом k-средних, показана на рисунке 11.

```
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
```

	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	19.849699	0.824221	31.174885	328.134006	3.281340	0.000000
1	19.849699	0.824221	31.174885	98.023999	7.620719	2.356303
2	19.849699	0.824221	31.174885	40.425519	7.953389	5.176504
3	19.849699	0.824221	31.174885	18.262171	9.176142	8.115744
4	19.849699	0.824221	31.174885	11.321753	5.605547	10.449064
5	19.849699	0.824221	31.174885	7.758693	5.066291	14.110567
6	19.849699	0.824221	31.174885	5.541922	4.682205	15.996483
7	19.849699	0.824221	31.174885	4.091394	4.273151	20.026944
8	19.849699	0.824221	31.174885	3.196493	3.987095	21.654715
9	19.849699	0.824221	31.174885	2.477541	1.872187	24.360637

Рисунок 11 - Сравнительная таблица

Вывод

В результате выполнения работы были получены практические навыки решения задачи кластеризации фактографических данных в среде Jupiter Notebook, были настроены параметры методов и оценена точность полученного разбиения.