МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №6
по дисциплине «Статистические методы обработки экспериментальных данных»
Тема: Кластерный анализ. Метод k-средних.

Студент гр. 8383 _____ Бабенко Н.С.

Студент гр. 8383 _____ Сахаров В.М.

Преподаватель _____ Середа А.-В.И.

Санкт-Петербург
2022

Цель работы

Освоение основных понятий и некоторых методов кластерного анализа, в частности, метода k-means.

Основные теоретические положения

Задача кластерного анализа заключается в том, чтобы разбить множества исследуемых объектов и признаков на однородные в соответствующем понимании группы. К характеристикам кластера относятся:

1. *Центр кластера* – это среднее геометрическое место точек, принадлежащих кластеру, в пространстве данных.
2. *Радиус кластера* – максимальное расстояние точек, принадлежащих кластеру, от центра кластера.
3. Кластеры могут быть *перекрывающимися*. В этом случае невозможно при помощи используемых процедур однозначно отнести объект к одному из двух или более кластеров. Такие объекты называют спорными.
4. *Спорный объект* – это объект, который по мере сходства может быть отнесен к более, чем одному кластеру.
5. *Размер кластера* может быть определен либо по радиусу кластера, либо по среднеквадратичному отклонению объектов для этого кластера. Объект относится к кластеру, если расстояние от объекта до центра кластера меньше радиуса кластера. Если это условие выполняется для двух и более кластеров, объект является спорным.

Существуют различные способы нормировки данных:

$$z = \frac{(x - \bar{x})}{\sigma}; z = \frac{x}{\bar{x}}; z = \frac{x}{x_{max}}; z = \frac{(x - \bar{x})}{x_{max} - x_{min}}$$

Расстоянием (метрикой) между объектами a и b пространстве параметров называется такая величина d_{ab}, которая удовлетворяет аксиомам:

$$1.\ d_{ab} > 0,\ если\ a \neq b, \quad 2.\ d_{ab} = 0,\ если\ a = b;$$
$$3.\ d_{ab} = d_{ba}; \quad 4.\ d_{ab} + d_{bc} \geq d_{ac}.$$

Мерой близости (сходства) называется величина μ_{ab}, имеющая предел и возрастающая с возрастанием близости объектов и удовлетворяющая условиям:

$$\mu_{ab} \text{ непрерывна; } \mu_{ab} = \mu_{ba}; 0 \le \mu_{ab} \le 1.$$

Суть метода k-средних заключается в том, что он стремится минимизировать суммарное квадратичное отклонение точек кластеров от центров этих кластеров:

$$V = \sum_{i=1}^{k} \sum_{x_j \in S_i} \left(x_j - \mu_i\right)^2$$

Центроиды выбираются в тех местах, где визуально скопление точек выше. Алгоритм разбивает множество элементов векторного пространства на заранее известное число кластеров k. Основная идея заключается в том, что на каждой итерации пересчитывается центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике. Алгоритм завершается, когда на какой-то итерации не происходит изменения центра масс кластеров.

Задание количества кластеров является сложным вопросом. Если нет разумных соображений на этот счет, рекомендуется первоначально создать 2 кластера, затем 3, 4, 5 и так далее, сравнивая полученные результаты.

Возможны две разновидности метода. Первая предполагает пересчет центра кластера после каждого изменения его состава, а вторая – лишь после завершения цикла.

После завершения многомерной классификации необходимо оценить полученные результаты. Для этой цели используются специальные характеристики – функционалы качества. Наилучшим разбиением считается такое, при котором достигается экстремальное (минимальное или максимальное) значение выбранного функционала качества.

В качестве таких функционалов могут быть использованы:

1. Сумма квадратов расстояний до центров кластеров

$$F_1 = \sum_{k=1}^{K} \sum_{i=1}^{N_k} d^2(X_i^{(k)}, X^{(k)}) \Rightarrow \min$$

2. Сумма внутрикластерных расстояний между объектами

$$F_2 = \sum_{k=1}^{K} \sum_{X_i, X_j \in S_k} d^2(X_i, X_j) \Rightarrow \min$$

3. Сумма внутрикластерных дисперсий

$$F_3 = \sum_{k=1}^{K} \sum_{i=1}^{N_k} \sigma_{ij}^2 \Rightarrow \min$$

Здесь σ - дисперсия *j*-й переменной в *k*-м кластере.

Оптимальным следует считать разбиение, при котором сумма внутрикластерных (внутригрупповых) дисперсий будет минимальной.

Постановка задачи

Дано конечное множество из объектов, представленных двумя признаками (в качестве этого множества принимаем исходную двумерную выборку, сформированную ранее в лабораторной работе №4). Выполнить разбиение исходного множества объектов на конечное число подмножеств (кластеров) с использованием метода k-means. Полученные результаты содержательно проинтерпретировать.

Порядок выполнения работы

a) Нормализовать множество точек, отобразить полученное множество.
b) Определить верхнюю оценку количества кластеров по формуле: $\bar{k} = \lfloor \sqrt{N/2} \rfloor$, где N – число точек.
c) Реализовать алгоритм k-means в двух вариантах:
    - Пересчет центра кластера осуществляется после каждого изменения его состава
    - Пересчет центра кластера осуществляется после того, как будет завершен шаг процедуры

d) На каждом шаге процедуры разбиения методом k-means вычислять функционалы качества полученного разбиения.

e) Отобразить полученные кластеры, выделить каждый кластер разным цветом, отметить центроиды.

f) Содержательно проинтерпретировать полученные результаты.

Выполнение работы

- Нормирование

Исходная выборка представлена в таблице 1.

Таблица 1

№	***nu***	***E***	№	***nu***	***E***	№	***nu***	***E***	№	***nu***	***E***	№	***nu***	***E***
1	481	135.2	***21***	418	131.4	***41***	513	159.3	***61***	450	122.3	***81***	475	143.6
2	445	124.7	***22***	378	103.8	***42***	489	149.8	***62***	468	128.9	***82***	518	144.4
3	550	147.9	***23***	521	154.9	***43***	474	132.5	***63***	441	122.8	***83***	566	175.7
4	465	140.9	***24***	394	117.7	***44***	379	94.6	***64***	460	140.7	***84***	464	131.3
5	566	168.5	***25***	504	145.3	***45***	472	135.6	***65***	480	117.7	***85***	394	112.1
6	497	147.3	***26***	440	126.7	***46***	544	169.6	***66***	429	112.9	***86***	480	146.1
7	478	136.6	***27***	465	114.8	***47***	507	142.4	***67***	457	126.4	***87***	321	86.1
8	521	139.6	***28***	418	109.3	***48***	409	116.7	***68***	464	143.2	***88***	502	132.5
9	352	84.9	***29***	418	118.6	***49***	498	164.0	***69***	431	125.0	***89***	460	122.4
10	422	117.9	***30***	465	127.7	***50***	468	142.0	***70***	424	119.0	***90***	458	104.7
11	506	153.5	***31***	447	117.5	***51***	593	187.4	***71***	502	137.2	***91***	362	111.7
12	443	122.9	***32***	433	131.5	***52***	523	152.6	***72***	465	140.7	***92***	503	148.5
13	434	140.4	***33***	460	136.8	***53***	478	126.6	***73***	492	137.5	***93***	446	144.0
14	422	108.6	***34***	382	98.8	***54***	438	122.2	***74***	446	128.4	***94***	421	115.1
15	569	157.4	***35***	532	160.6	***55***	423	115.9	***75***	482	136.4	***95***	407	110.5
16	439	119.2	***36***	482	148.2	***56***	408	110.0	***76***	510	140.6	***96***	448	137.7
17	437	129.4	***37***	472	122.6	***57***	386	105.8	***77***	434	122.3	***97***	490	139.9
18	461	138.6	***38***	532	158.7	***58***	428	130.3	***78***	623	195.7	***98***	482	141.2
19	351	89.0	***39***	473	137.9	***59***	560	169.8	***79***	468	141.2	***99***	463	129.2
20	390	91.4	***40***	525	148.3	***60***	483	130.3	***80***	471	119.7	***100***	459	145.4

Нормализация была выполнена по методу минимакс:

$$\mathrm{X}_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

После нормализации минимальное и максимальное масштабируемые значения равны 0 и 1 соответственно.

Нормализованная выборка представлена в табл. 2 и отображена на рис. 1.

Таблица 2

№	***nu***	***E***	№	***nu***	***E***	№	***nu***	***E***	№	***nu***	***E***	№	***nu***	***E***
1	0.53	0.454	***21***	0.321	0.42	***41***	0.636	0.671	***61***	0.427	0.338	***81***	0.51	0.53
2	0.411	0.359	***22***	0.189	0.171	***42***	0.556	0.586	***62***	0.487	0.397	***82***	0.652	0.537
3	0.758	0.569	***23***	0.662	0.632	***43***	0.507	0.43	***63***	0.397	0.342	***83***	0.811	0.819
4	0.477	0.505	***24***	0.242	0.296	***44***	0.192	0.088	***64***	0.46	0.504	***84***	0.474	0.419
5	0.811	0.755	***25***	0.606	0.545	***45***	0.5	0.458	***65***	0.526	0.296	***85***	0.242	0.245
6	0.583	0.563	***26***	0.394	0.377	***46***	0.738	0.764	***66***	0.358	0.253	***86***	0.526	0.552
7	0.52	0.467	***27***	0.477	0.27	***47***	0.616	0.519	***67***	0.45	0.375	***87***	0.0	0.011
8	0.662	0.494	***28***	0.321	0.22	***48***	0.291	0.287	***68***	0.474	0.526	***88***	0.599	0.43
9	0.103	0.0	***29***	0.321	0.304	***49***	0.586	0.714	***69***	0.364	0.362	***89***	0.46	0.338
10	0.334	0.298	***30***	0.477	0.386	***50***	0.487	0.515	***70***	0.341	0.308	***90***	0.454	0.179
11	0.613	0.619	***31***	0.417	0.294	***51***	0.901	0.925	***71***	0.599	0.472	***91***	0.136	0.242
12	0.404	0.343	***32***	0.371	0.421	***52***	0.669	0.611	***72***	0.477	0.504	***92***	0.603	0.574
13	0.374	0.501	***33***	0.46	0.468	***53***	0.52	0.376	***73***	0.566	0.475	***93***	0.414	0.533
14	0.334	0.214	***34***	0.202	0.125	***54***	0.387	0.337	***74***	0.414	0.393	***94***	0.331	0.273
15	0.821	0.654	***35***	0.699	0.683	***55***	0.338	0.28	***75***	0.533	0.465	***95***	0.285	0.231
16	0.391	0.31	***36***	0.533	0.571	***56***	0.288	0.227	***76***	0.626	0.503	***96***	0.421	0.477
17	0.384	0.402	***37***	0.5	0.34	***57***	0.215	0.189	***77***	0.374	0.338	***97***	0.56	0.496
18	0.464	0.485	***38***	0.699	0.666	***58***	0.354	0.41	***78***	1.0	1.0	***98***	0.533	0.508
19	0.099	0.037	***39***	0.503	0.478	***59***	0.791	0.766	***79***	0.487	0.508	***99***	0.47	0.4
20	0.228	0.059	***40***	0.675	0.572	***60***	0.536	0.41	***80***	0.497	0.314	***100***	0.457	0.546

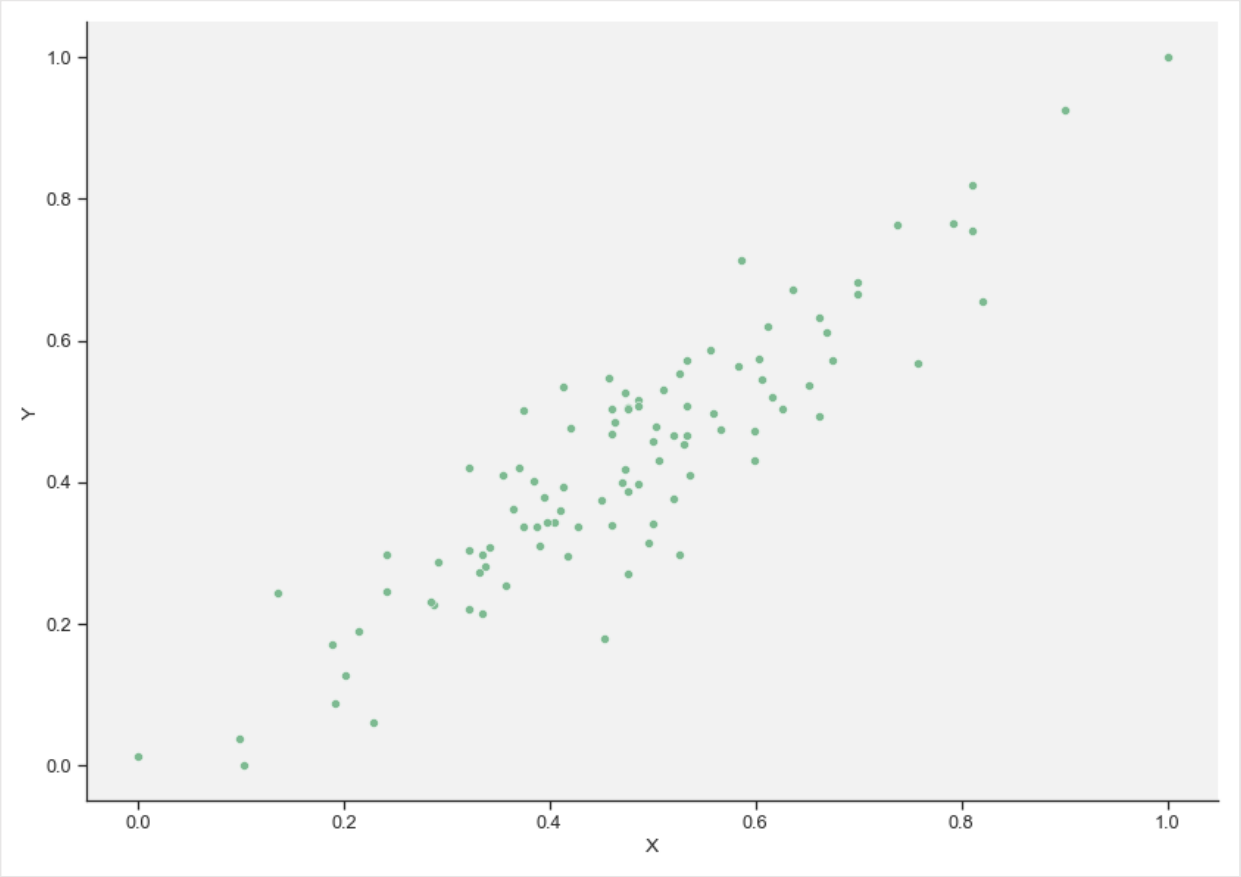

Рисунок 1 – Нормализованная выборка

- Оценка количества кластеров

  Была найдена верхняя оценка количества кластеров:

$$\bar{k} = \left\lfloor \sqrt{\frac{N}{2}} \right\rfloor = \left\lfloor \sqrt{\frac{104}{2}} \right\rfloor = 7$$

- Метод k-средних

Реализован метод k-средних для количества кластеров $[2;\ 7]$. Полученные кластеры были отображены, выделены разным цветом, были отмечены центроиды, вычислены функционалы качества разбиения. В таблицах представлены количество элементов в кластерах и их центры.

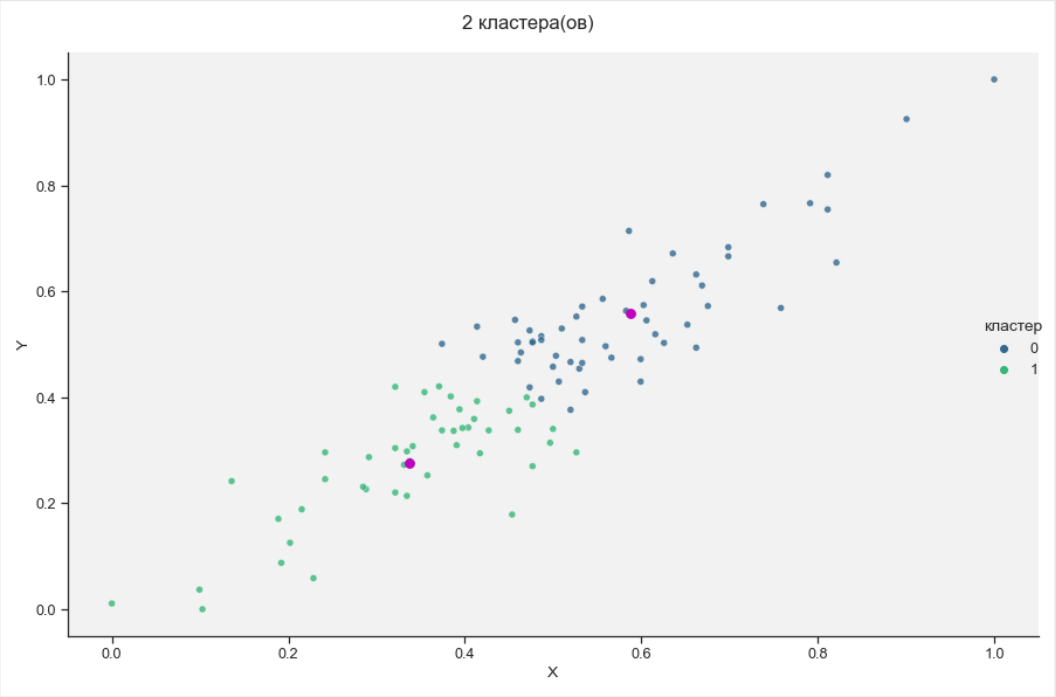

Рисунок 2 – 2 кластера (3 шага)

Таблица 3

Центр кластера		**Количество элементов**
0.5882	0.5593	54.0
0.3372	0.276	46.0

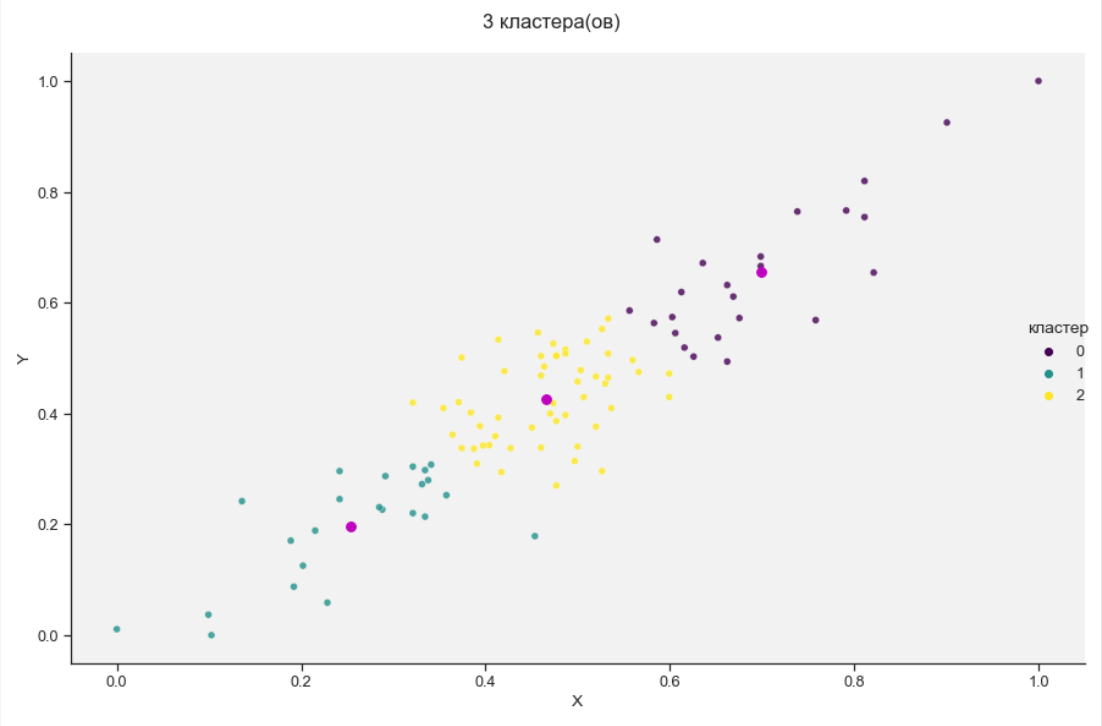

Рисунок 3 – 3 кластера (5 шагов)

Таблица 4

Центр кластера		Количество элементов
0.699	0.6559	24.0
0.2541	0.1971	23.0
0.4652	0.4268	53.0

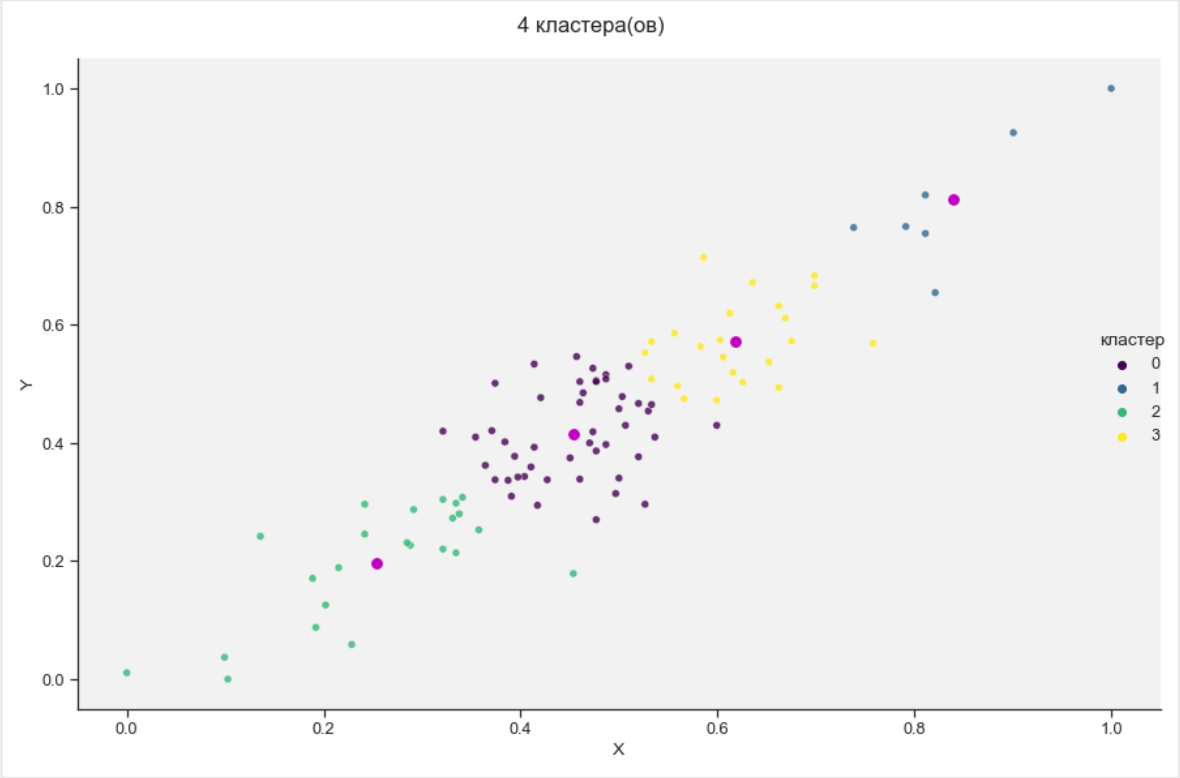

Рисунок 4 – 4 кластера (8 шагов)

Таблица 5

Центр кластера		Количество элементов
0.454	0.4159	47.0
0.8392	0.812	7.0
0.2541	0.1971	23.0
0.6182	0.571	23.0

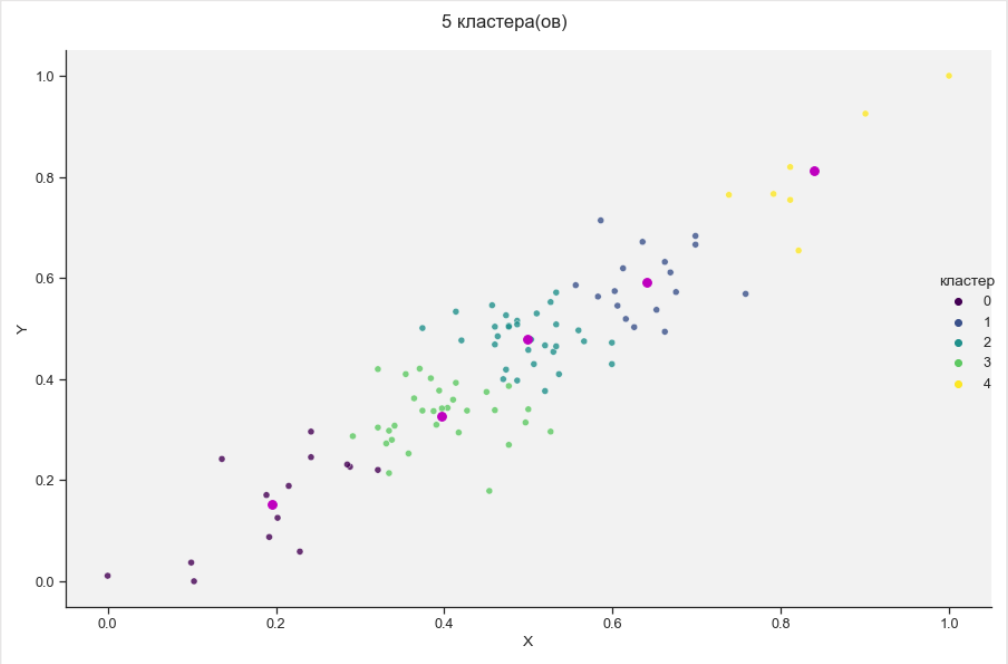

Рисунок 5 – 5 кластеров (10 шагов)

Таблица 6

Центр кластера		**Количество элементов**
0.1958	0.1528	14.0
0.6412	0.5916	17.0
0.4986	0.4793	31.0
0.3968	0.3276	31.0
0.8392	0.812	7.0

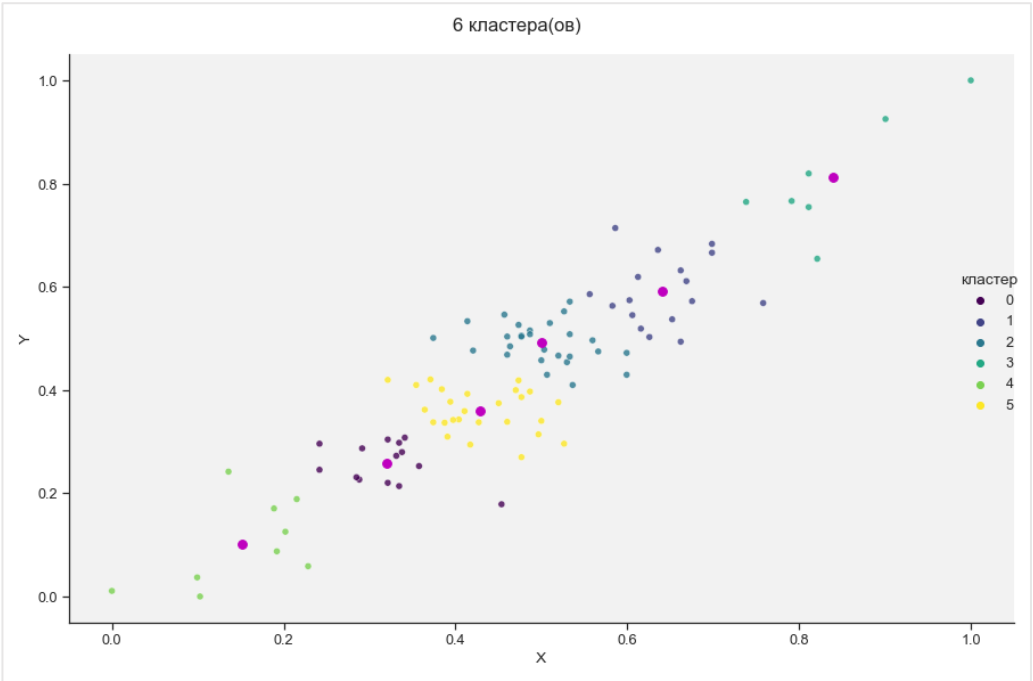

Рисунок 6 – 6 кластеров (11 шагов)

Таблица 7

Центр кластера		Количество элементов
0.32	0.2581	14.0
0.6412	0.5916	17.0
0.5002	0.4914	27.0
0.8392	0.812	7.0
0.1516	0.1023	9.0
0.4288	0.3598	26.0

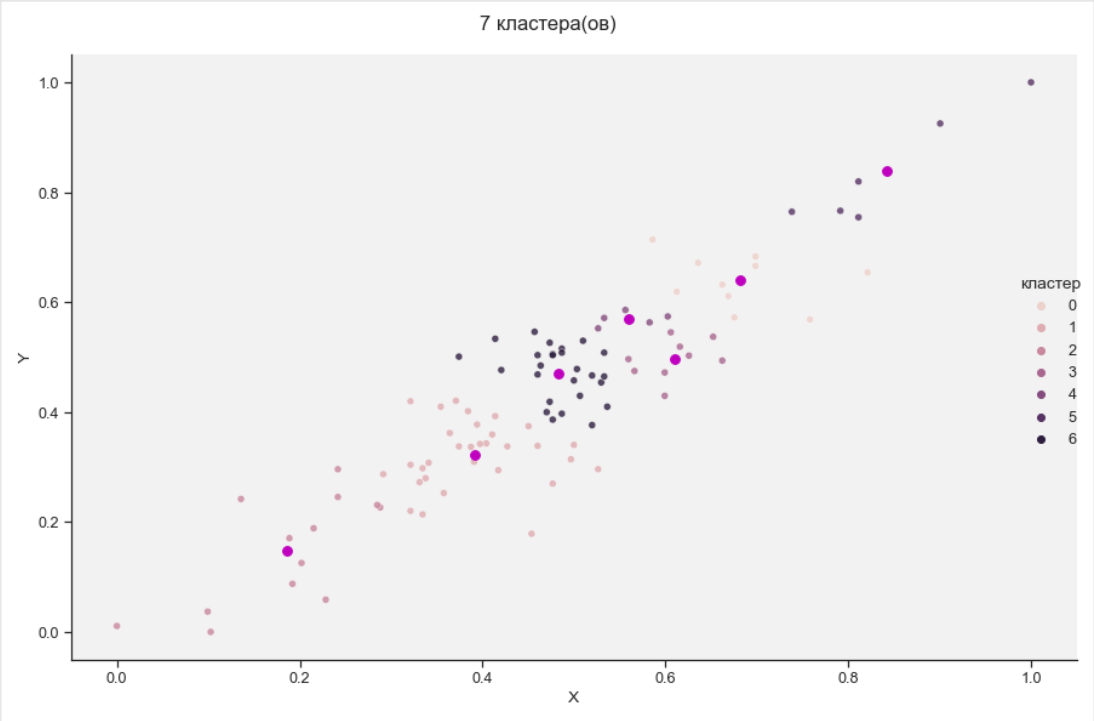

Рисунок 7 – 7 кластеров (16 шагов)

Таблица 8

Центр кластера		Количество элементов
0.6818	0.6392	10.0
0.3918	0.3223	31.0
0.1862	0.1477	13.0
0.6096	0.4967	9.0
0.5603	0.5693	5.0
0.8422	0.8383	6.0
0.4827	0.4711	26.0

- Оценка качества разбиения

Для каждого разбиения были вычислены функционалы качества F_1, F_2, F_3, которые были определены в начале. В таблице 9 приведены значения функционалов на первой и последней итерациях алгоритма для разного количества кластеров разбиения.

Таблица 9

Количество кластеров	**2**	**3**	**4**	**5**	**6**	**7**
F_1	2.92	1.948	2.045	1.791	1.134	1.27
F_1 **(конец)**	2.891	1.645	1.171	0.812	0.625	0.56
F_2	151.736	103.754	95.325	75.691	22.247	29.21
F_2 **(конец)**	146.915	54.414	34.918	16.851	10.607	7.662
F_3	0.058	0.061	0.07	0.066	0.065	0.063
F_3 **(конец)**	0.058	0.06	0.058	0.057	0.054	0.055

Из таблицы можно увидеть, что при увеличении числа кластеров, минимизируются все функционалы, а также, то насколько сильно они меняются в сравнении с первой итерацией.

Алгоритм был реализован в двух вариантах: в первом, который был представлен выше, центр пересчитывается только по завершении шага процедуры, второй же вариант предполагает изменение центра кластера после обработки каждого объекта. Сравнение алгоритмов представлено в таблице 10.

Таблица 10

Количество кластеров	**2**	**3**	**4**	**5**	**6**	**7**
Количество итераций (первый алгоритм)	3	5	8	10	11	16
Количество итераций (второй алгоритм)	2	2	3	4	3	3

Из таблицы видно, что при увеличении количества кластеров увеличивается число итераций, а также, что количество итераций второго алгоритма

меньше, чем первого, что связано с тем, что центр меняется после обработки каждого объекта.

Выводы

Освоены основные понятия кластерного анализа и метода k-means. Первоначальная двумерная выборка была нормализована методом минимакс, после которого минимальное и максимальное масштабируемые значения равны 0 и 1 соответственно. Нормализованная выборка была отображена на рисунке.

С помощью алгоритма k-средних было произведено разбиение выборки на 2, 3, …, 7 кластеров. Для каждого разбиения были выведены центры кластеров и количество элементов в кластерах. Было оценено качество разбиения с помощью функционалов качества. В сравнительной таблице можно заметить, что при увеличении числа кластеров, минимизируются все функционалы качества, а также, то насколько сильно они меняются при сравнении первой и последней итераций.

Алгоритм k-средних был реализован в двух вариантах. В первом, центр пересчитывается только по завершении шага процедуры, второй же вариант предполагает изменение центра кластера после обработки каждого объекта. Из сравнительной таблицы можно заметить, что при увеличении количества кластеров увеличивается число итераций, а также, что количество итераций второго варианта алгоритма меньше, чем первого, это связано с тем, что центр меняется после обработки каждого элемента.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from scipy.spatial import distance

df0 = pd.read_csv('data/main_data.csv')
X = df0['nu']
Y = df0['E']

X_norm = MinMaxScaler().fit_transform(df0)
df = pd.DataFrame(data=X_norm, columns=['nu','E'])
df.round(3).to_csv('data/df_norm.csv', index=False)

sns.set_theme(palette='crest', font_scale=1.15)
sns.set_style("ticks", {"axes.facecolor": ".95"})
ax = sns.relplot(data=df, x='nu', y='E', kind='scatter', height=8.27,
aspect=11.7/8.27)
ax.set_axis_labels('X', 'Y')
plt.tight_layout()
plt.savefig('pics/1.png')

N = len(df)
sup_k = np.floor(np.sqrt(N/2))
sup_k

def sc_plots(data, means, Ncl, step):
    if Ncl > 6:
        ax = sns.relplot(data=data, x='nu', y='E', hue='кластер',
kind='scatter', alpha=0.8,
                         height=8.27, aspect=11.7/8.27)
        plt.scatter(means[:,0],means[:,1], c='m', s=60)
    else:
        ax = sns.relplot(data=data, x='nu', y='E', hue='кластер',
kind='scatter', palette='viridis', alpha=0.8,
                         height=8.27, aspect=11.7/8.27)
        plt.scatter(means[:,0],means[:,1], c='m', s=60)

    print(f'кластеры = {Ncl}; шаги = {step}')
    ax.set_axis_labels('X', 'Y')
    ax.fig.suptitle(f'{Ncl} кластера(ов)')
    plt.tight_layout()
    plt.savefig(f'pics/2_{Ncl}.png')
    plt.close()
```

```
def nearest_center(data, cts):
    distl = np.array([], dtype=np.float64)
    for i in cts:
        distl = np.append(distl, np.linalg.norm(i[:-1]-data)) # евклидово
расстояние
    min_dist = np.argmin(distl)
    return min_dist


def Fs(data):
    curr_data = data.copy()
    cts = curr_data.groupby('кластер').mean()
    F1,F2,F3 = 0,0,0

    # F1 - сумма кв. расст. точек до центров соотв. кластеров
    for i in range(len(curr_data)):
        dist_F1 = np.linalg.norm(curr_data.iloc[i,:-1].values-
cts.values[curr_data.iloc[i,2]])
        F1 += dist_F1**2

    # F2 - сумма кв. расст. до всех точек соотв. кластеров
    for i in range(len(cts)):
        coords = curr_data[curr_data['кластер']==i].iloc[:,:2].values
        dist_F2 = distance.cdist(coords, coords, 'euclidean')
        F2 += (np.triu(dist_F2,0)**2).sum()

    # F3 - сумма внутрикластерных дисперсий
    F3 = curr_data.groupby('кластер').var().values.sum()

    return F1,F2,F3

def custKM(dataf, n_clusters, chng_ctr=1, max_iter=30, tol=0.01):
    data = dataf.copy()
    centers = data.sample(n_clusters) # рандомные центры
    data['кластер'] = -1 # нет принадлежности кластерам
    cts = np.array([], dtype=np.float64)
    F1,F2,F3 = 0,0,0
    df_Fs = pd.DataFrame(columns=['F1', 'F2', 'F3'])

    for i in range(n_clusters):
        data.loc[centers.index[i],'кластер'] = i # кластеры для центров
        cts = np.append(cts, [data.loc[centers.index[i]].values])
    centers = cts.reshape((n_clusters,3))

    for j in range(max_iter):
        for i in range(len(data)): # ближ. центр для каждой точки
            curr_clust = nearest_center(data.iloc[i,:-1].values, centers)
            data.loc[i,'кластер'] = curr_clust # соотносим кластер
```

```
            if chng_ctr: # пересчет центра при новой точке
                centers[curr_clust][:2] =
data[data['кластер']==curr_clust].iloc[:,:2].mean()

        if chng_ctr == 0: # пересчет центра на каждой итерации
            for i in range(n_clusters):
                centers[i][:2] = data[data['кластер']==i].iloc[:,:2].mean()

        cur_F1,cur_F2,cur_F3 = Fs(data) # функционалы
        df_Fs = df_Fs.append({'F1':cur_F1,'F2':cur_F2,'F3':cur_F3},
ignore_index=True)

        if np.abs(F1-cur_F1) < tol:
            data['кластер'].astype('int')
            sc_plots(data, centers, n_clusters, j+1)
            break
        F1,F2,F3 = cur_F1,cur_F2,cur_F3
        data['кластер'] = -1

    df_ctrs = pd.DataFrame(np.concatenate((centers[:,:2],

data.groupby('кластер')['nu'].count().values.reshape(-1,1)), axis=1),
                           columns=['nu_mean', 'E_mean', 'num'])
    silhouette_avg = silhouette_score(data.values[:,:2], data.values[:,2])

    return df_Fs, df_ctrs, silhouette_avg

for i in range(2,8):
    F, ctrs, sil = custKM(df, n_clusters=i, chng_ctr=1)
    F.round(3).to_csv(f'data/Fs_{i}c.csv', index=False)
    ctrs.round(4).to_csv(f'data/centers_{i}c.csv', index=False)
```