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.

Студентка гр. 7381 _____ Алясова А.Н.

Студент гр. 7381 _____ Кортев Ю.В.

Преподаватель _____ Середа А.-В.И.

Санкт-Петербург

2021

Цель работы.

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

Основные теоретические положения.

Кластерный анализ – многомерная статистическая процедура, выполняющая сбор данных, содержащих информацию о выборке объектов, и затем упорядочивающая объекты в сравнительно однородные группы.

К характеристикам кластера относятся в частности: центр, радиус; среднеквадратическое отклонение; размер кластера.

Центр кластера – это среднее геометрическое место точек, принадлежащих кластеру, в пространстве данных.

Радиус кластера – максимальное расстояние точек, принадлежащих кластеру, от центра кластера.

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.

Спорный объект - это объект, который по мере сходства может быть отнесен к более, чем одному кластеру.

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.

Большое значение в кластерном анализе имеет выбор масштаба. Пусть, например, значения переменной x превышают 100, а переменной y - в интервале от 0 до 1.

Тогда, при расчете расстояния между точками переменная x, будет практически полностью доминировать над переменной y. В результате практически невозможно корректно рассчитать расстояния между точками.

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

1. $d_{ab} > 0$, *если* $a \neq b$, 2. $d_{ab} = 0$, *если* $a = b$;
3. $d_{ab} = d_{ba}$; 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:

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

Существует возможность простого перехода от расстояния к мерам близости:

$$\mu = \frac{1}{1+d}.$$

Алгоритм k-means – это наиболее популярный метод кластеризации, который разделяет определенный набор данных на заданное пользователем число кластеров k. Алгоритм прост для реализации и запуска, относительно быстрый, легко адаптируется и распространен на практике. Это исторически один из самых важных алгоритмов интеллектуального анализа данных.

Суть алгоритма заключается в том, что он стремится минимизировать суммарное квадратичное отклонение точек кластеров от центров этих кластеров:

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

где k – это число кластеров, S_i – полученные кластеры, $i = 1, 2, \ldots, k$ и μ_i – центры масс.

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 перевычисляется 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. Это происходит за конечное число итераций, так как количество возможных разбиений конечного множества конечно, а на каждом шаге суммарное квадратичное отклонение V не увеличивается, поэтому зацикливание невозможно.

Возможны две разновидности метода k -средних.

Первая предполагает пересчет центра кластера после каждого изменения его состава, как рассмотрено выше, а вторая —лишь после завершения цикла.

В обоих случаях итеративный алгоритм этого метода минимизирует дисперсию внутри каждого кластера, хотя в явном виде такой критерий оптимизации не используется. Перед началом работы метода целесообразно нормировать характеристики объектов: $\hat{X} = \frac{x - \overline{x_в}}{S_x}$; $\hat{Y} = \frac{y - \overline{u_в}}{S_y}$.

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

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.

Судить о качестве разбиения позволяют и некоторые простейшие приемы. Например, можно сравнивать средние значения признаков в отдельных кластерах (группах) со средними значениями в целом по всей совокупности объектов. Если групповые средние существенно отличаются от общего среднего значения, то это может являться признаком хорошего разбиения.

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-средних. Полученные результаты содержательно проинтерпретировать.

Порядок выполнения работы.

1. Нормализовать множество точек, отобразить полученное множество.
2. Определить верхнюю оценку количества кластеров по формуле: $\bar{k} = \lfloor \sqrt{N/2} \rfloor$, где N – число точек.
3. Реализовать алгоритм k-means, отобразить полученные кластеры, выделить каждый кластер разным цветом, отметить центроиды.
4. Провести оценку качества разбиения для различных разбиений.
5. Содержательно проинтерпретировать полученные результаты.

Выполнение работы.

1) Нормализовать множество точек, отобразить полученное множество.

Исследуемая выборка представлена в таблице 1.

Таблица 1

№	nu	E	№	nu	E	№	nu	E	№	nu	E	№	nu	E
1	480	153.3	25	408	110.0	49	405	103.6	73	465	127.7	97	487	146.0
2	510	129.4	26	331	74.1	50	434	140.4	74	390	108.1	98	532	158.7
3	426	119.0	27	467	113.0	51	344	86.8	75	463	129.2	99	330	71.1
4	482	139.9	28	545	145.3	52	415	119.7	76	468	128.9	100	438	134.1
5	393	103.2	29	396	83.8	53	463	136.7	77	488	134.1	101	593	187.4
6	510	162.3	30	351	102.9	54	475	143.6	78	443	137.4	102	445	124.7
7	403	123.9	31	503	148.5	55	463	144.9	79	505	155.8	103	518	154.0
8	506	158.4	32	402	120.8	56	392	82.7	80	395	109.1	104	496	141.7
9	393	122.8	33	542	146.1	57	452	140.5	81	474	132.5	105	473	136.4
10	442	115.4	34	437	124.3	58	504	143.8	82	490	139.9	106	522	154.5
11	411	112.9	35	453	119.5	59	443	122.9	83	396	90.1	107	547	154.7
12	514	153.6	36	386	105.8	60	461	138.6	84	362	97.9	108	560	169.8
13	525	156.5	37	434	122.3	61	340	85.1	85	566	175.7	109	412	127.8
14	543	155.4	38	418	118.4	62	438	134.9	86	418	109.3	110	444	130.0
15	412	116.3	39	391	107.5	63	523	148.7	87	502	132.5	111	437	121.8
16	449	124.5	40	399	100.0	64	416	120.5	88	500	155.5	112	462	138.8
17	482	136.4	41	486	139.4	65	483	143.4	89	359	71.9	113	438	122.2
18	569	157.4	42	421	124.2	66	440	128.5	90	443	135.7	114	406	110.1
19	484	147.5	43	496	143.1	67	423	131.1	91	421	118.0	115	413	106.7
20	472	134.2	44	463	121.2	68	386	95.5	92	433	128.2	116	458	121.7
21	453	124.2	45	508	159.0	69	321	86.1	93	514	174.6	117	408	117.0
22	422	117.9	46	419	105.3	70	433	131.5	94	320	72.6			
23	320	64.5	47	434	108.7	71	351	89.0	95	406	113.8			
24	547	164.4	48	440	126.7	72	481	148.3	96	465	140.9			

Отображение исходной выборки представлено на рис. 1.

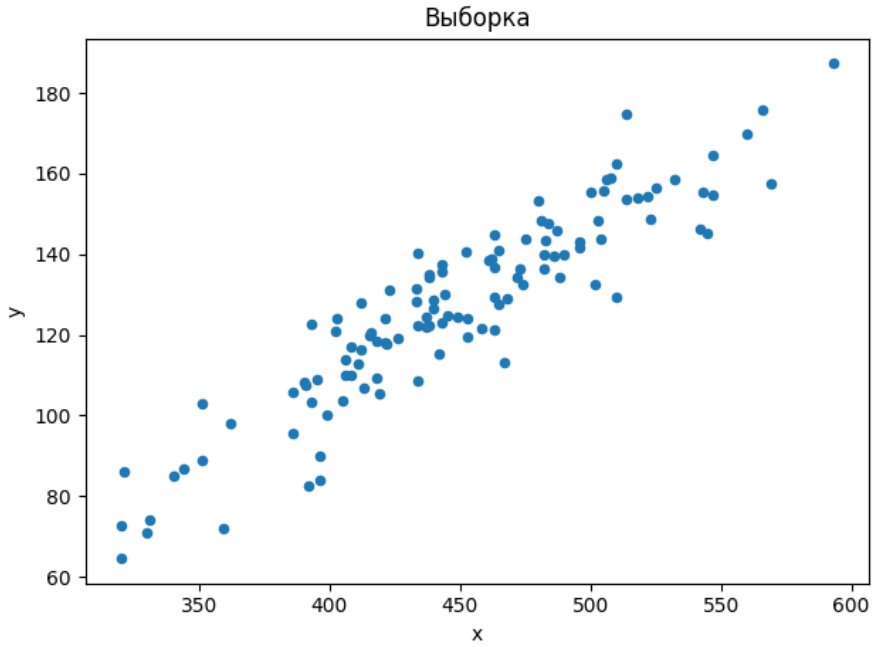

Рисунок 1 – Исходная выборка

Нормализация координат точек определяется по формулам:

$$x_i = \frac{x_i - x_{min}}{x_{max} - x_{min}}.$$

Отображение нормализованной выборки представлено на рис. 2.

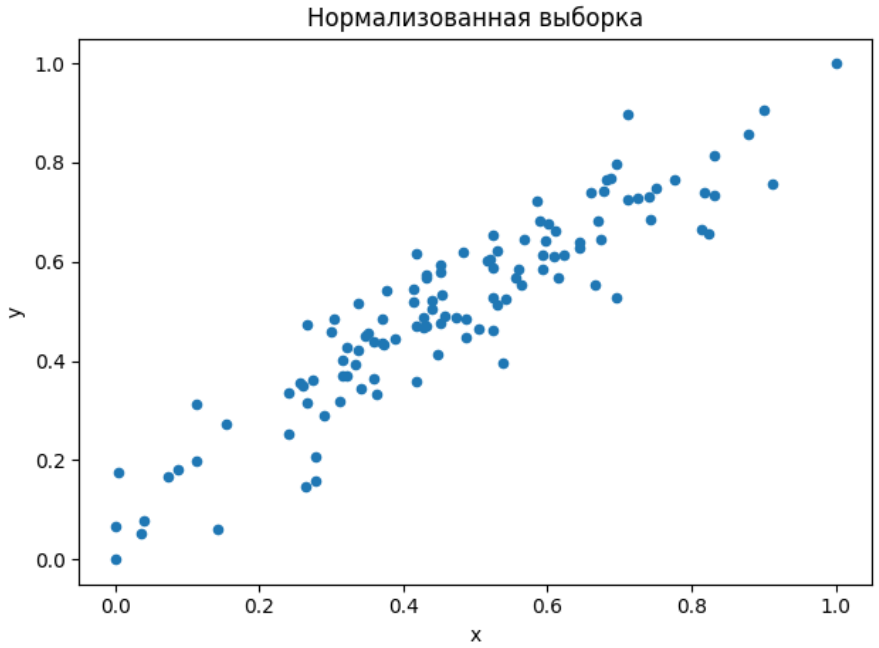

Рисунок 2 – Нормализованная выборка

2) Определим верхнюю оценку количества кластеров по формуле: $\bar{k} = \lfloor\sqrt{N/2}\rfloor$, где N – число точек.

Верхняя оценка количества кластеров:

$$\bar{k} = \lfloor\sqrt{N/2}\rfloor = \lfloor\sqrt{117/2}\rfloor = 7.$$

3) Реализовать алгоритм k-means, отобразить полученные кластеры, выделить каждый кластер разным цветом, отметить центроиды.

Реализуем алгоритм k-means. Отобразим полученные кластеры, выделим каждый кластер разным цветом, отметим центроиды.

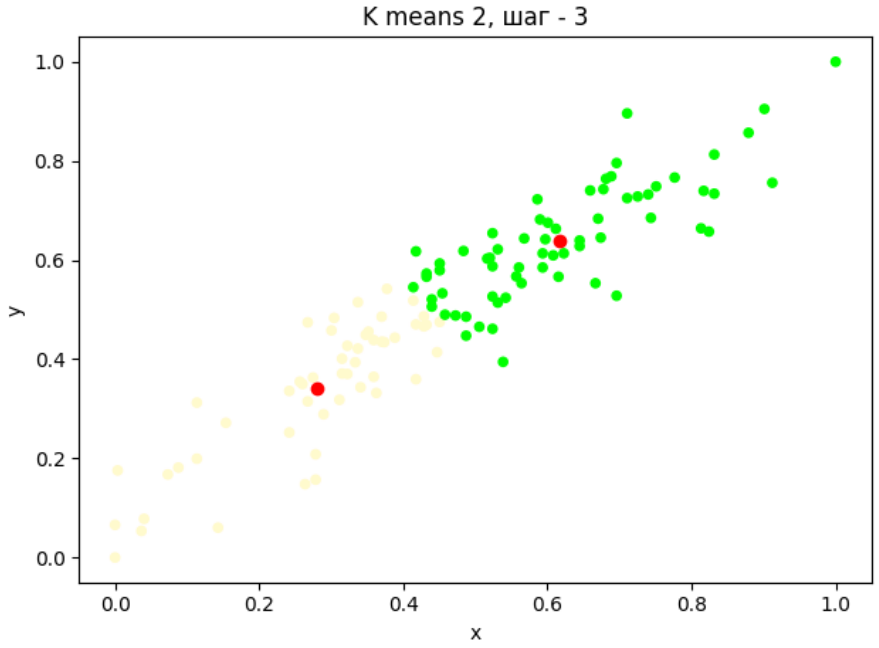

Рисунок 3 – Кластеризация алгоритмом k-means (2 кластера)

Таблица 2

Номер кластера	Центр кластера	Количество элементов в кластере
1	(0.28075845722904547; 0.34033727404712905)	51
2	(0.6171606171606171; 0.6384717804571344)	66

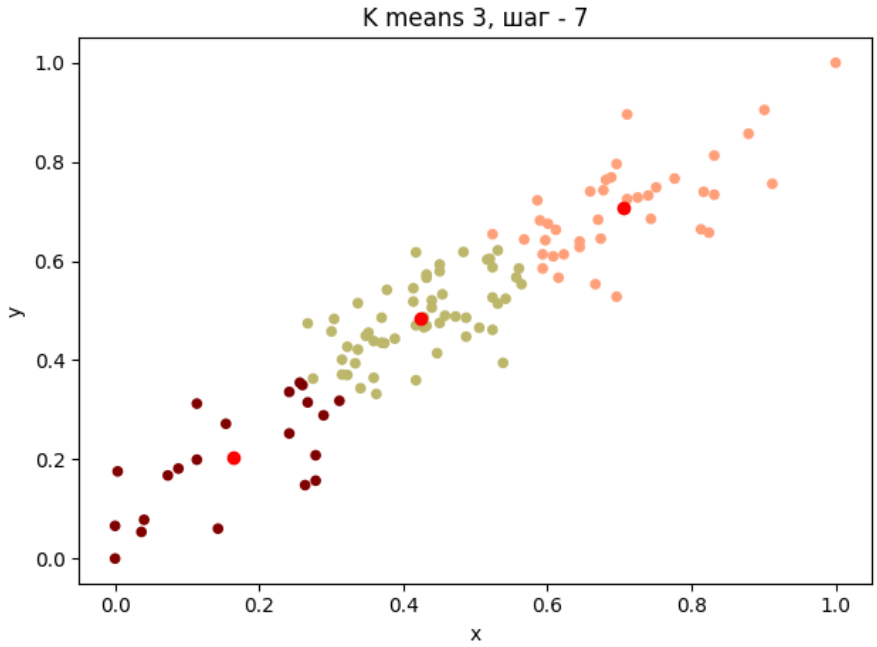

Рисунок 4 – Кластеризация алгоритмом k-means (3 кластера)

Таблица 3

Номер кластера	Центр кластера	Количество элементов в кластере
1	(0.7045177045177047; 0.7068494293880787)	39
2	(0.16448630734345018; 0.204502305397342)	21
3	(0.42317331791016; 0.48481863731745967)	57

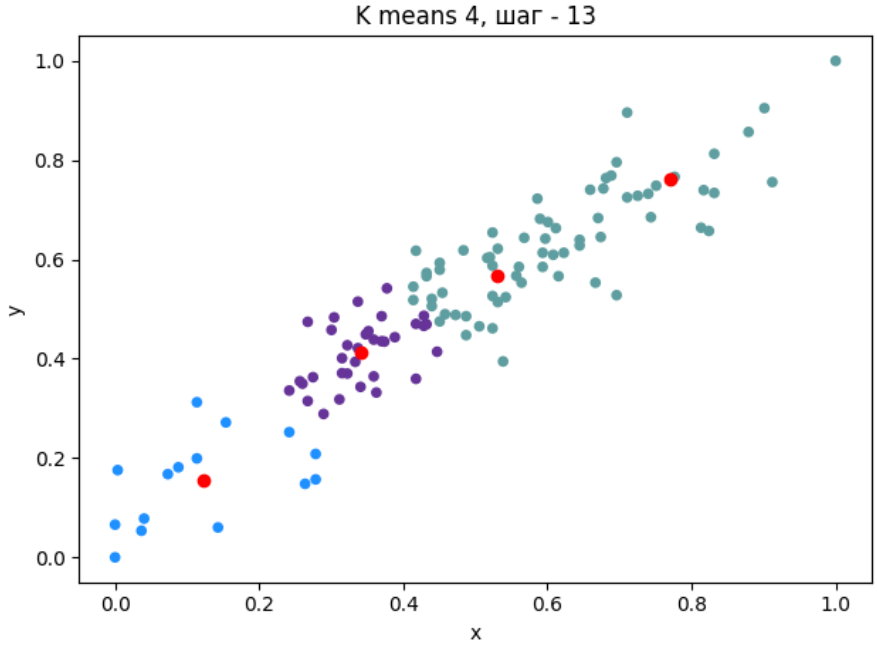

Рисунок 5 – Кластеризация алгоритмом k-means (4 кластера)

Таблица 4

Номер кластера	Центр кластера	Количество элементов в кластере
1	(0.7701863354037267; 0.7629391162840059)	23
2	(0.12185592185592185; 0.15546514781665308)	15
3	(0.5307285307285308; 0.5685561884097278)	45
4	(0.34195216548157736; 0.41269803283396345)	34

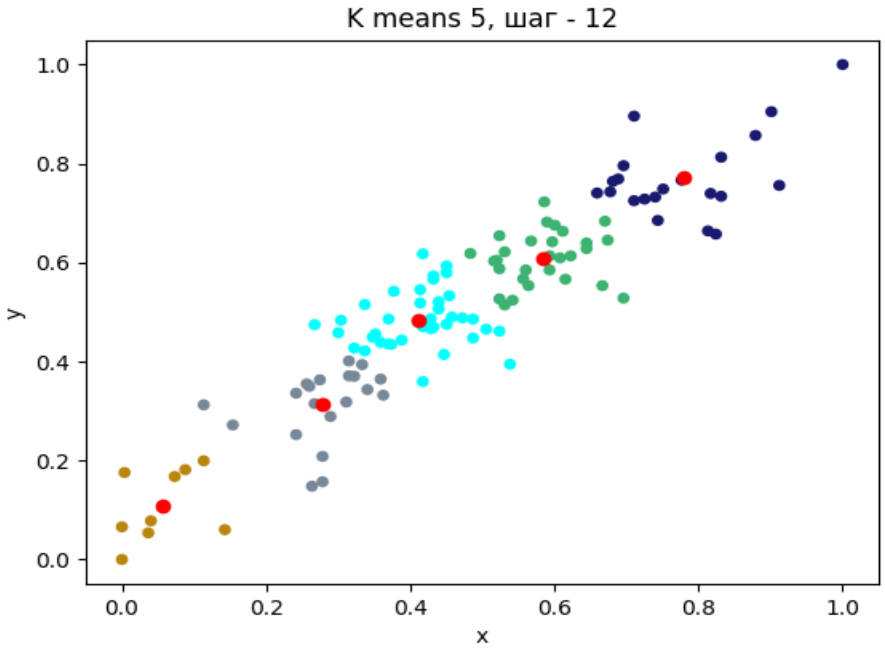

Рисунок 6 – Кластеризация алгоритмом k-means (5 кластеров)

Таблица 5

Номер кластера	Центр кластера	Количество элементов в кластере
1	(0.779522065236351; 0.7723274826610872)	21
2	(0.5848174813692055; 0.6087371285878621)	29
3	(0.05535205535205535; 0.10912214085525718)	9
4	(0.4117023327549644; 0.48396214294891016)	38
5	(0.2789377289377289; 0.31257119609438566)	20

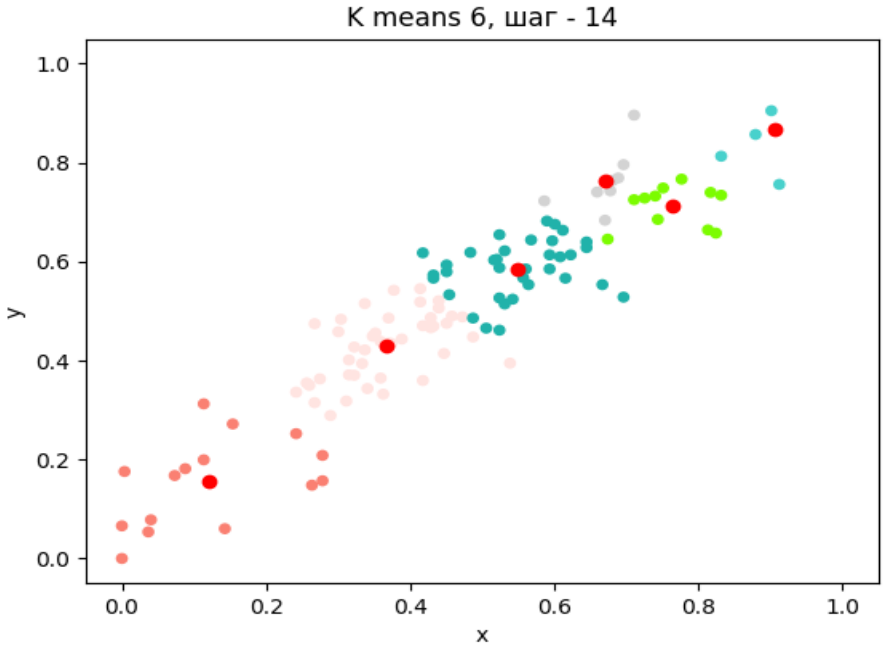

Рисунок 7 – Кластеризация алгоритмом k-means (6 кластеров)

Таблица 6

Номер кластера	Центр кластера	Количество элементов в кластере
1	(0.7642357642357642; 0.7114431540794436)	11
2	(0.12185592185592185; 0.15546514781665308)	15
3	(0.5480900052328623; 0.5850517261420435)	35
4	(0.9047619047619048; 0.8660699755899104)	5
5	(0.6712454212454213; 0.7642392188771359)	8
6	(0.36604480790527305; 0.42831191931424684)	43

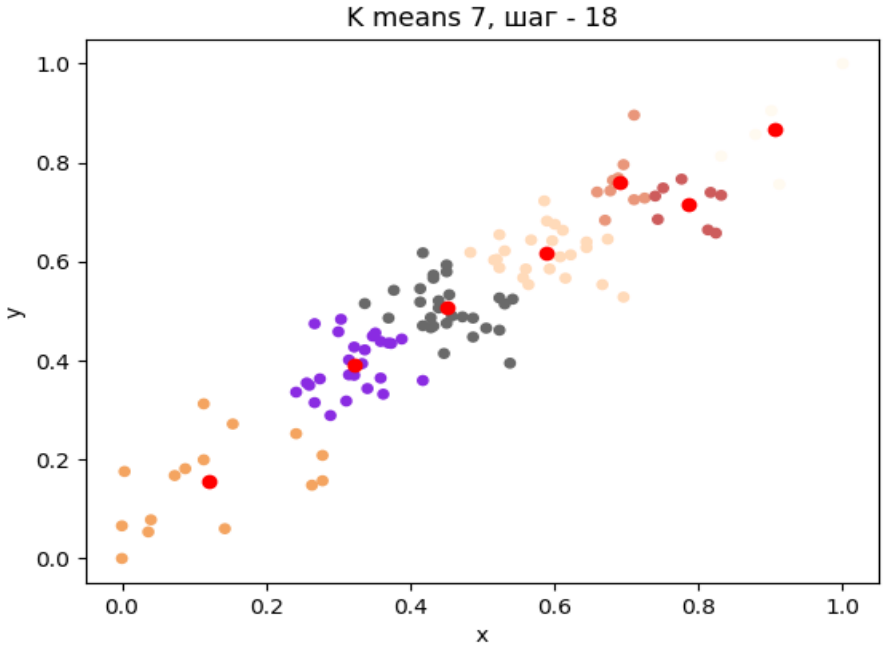

Рисунок 8 – Кластеризация алгоритмом k-means (7 кластеров)

Таблица 7

Номер кластера	Центр кластера	Количество элементов в кластере
1	(0.9047619047619048; 0.8660699755899104)	5
2	(0.6910866910866911; 0.7605098996474099)	9
3	(0.4518125552608312; 0.5060183496534889)	29
4	(0.3226260918568611; 0.3916254616010514)	26
5	(0.5876923076923076; 0.6162082994304312)	25
6	(0.7870879120879122; 0.7159275834011392)	8
7	(0.12185592185592185; 0.15546514781665308)	15

4) Провести оценку качества разбиения для различных разбиений.

Для проведения оценки качества разбиения для различных разбиений используются функционалы качества:

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.

Для различных значений k рассчитаем функционалы качества и результаты занесём в табл. 8.

Таблица 8

Количество кластеров k	F_1	F_2	F_3
2	3.9357	232.9588	0.03440
3	2.011	82.2782	0.0293
4	1.3016	40.6151	0.0259
5	0.9685	25.2083	0.0226
6	0.8843	18.7264	0.0291
7	0,7506	15,0071	0.0266

По полученным данным можно сделать вывод о том, что с увеличением числа кластеров, минимизируются значения перечисленных функционалов качества.

Выводы.

Таким образом, были освоены основные понятия кластерного анализа, в частности, метода k-средних. Верхняя оценка количества кластеров была посчитана по формуле: $\bar{k} = \left\lfloor \sqrt{\frac{N}{2}} \right\rfloor$ и равна 7. С помощью алгоритма k-means исходная выборка была разбита на различное количество кластеров: 2, 3, 4, 5, 6, 7. С увеличением числа кластеров, уменьшаются значения функционалов качества, используемых в работе. Было также замечено, что чем больше кластеров, тем больше шагов необходимо проделать алгоритму, чтобы на последнем шаге F_1, F_2 и F_3 имели минимальное значение. Из этого можно сделать выводы, что разбиение каждый раз улучшалось и в итоге получилось оптимальным.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.colors as colors
import random
from itertools import combinations

np.random.seed(10)
random.seed(11)
df = pd.read_csv('sample.csv', header=None)
df.columns = ['x', 'y']

ax = df.plot.scatter(x=0, y=1)
ax.set_title('Выборка')
plt.show()

df = (df - df.min(axis=0)) / (df.max(axis=0) - df.min(axis=0))
# df = (df - df.mean(axis=0)) / df.std(axis=0)

ax = df.plot.scatter(x=0, y=1)
ax.set_title('Нормализованная выборка')
plt.show()

up_limit = np.sqrt(len(df) / 2).astype(np.int64)
print('Верхняя граница: {}'.format(up_limit))


def f1():
    distances = df.apply(lambda x: np.min(dists_to_centroids(x,
centroids)) ** 2, axis=1)
    return distances.sum()


def get_metrics():
    f2 = []
    f3 = []
    for i, centroid in enumerate(centroids.to_numpy()):
        cluster_dists = []
        f3.append(df[cl_centroids == i].var().mean())
        for comb in combinations(df[cl_centroids == i].to_numpy(), 2):
            cluster_dists.append(np.linalg.norm(comb[0] - comb[1]) ** 2)
```

```
        f2.append(sum(cluster_dists))
    f2 = sum(f2)
    f3 = sum(f3)

    print('----\nF1 = {}\nF2 = {}\nF3 = {}\n----'.format(f1(), f2, f3))


def dists_to_centroids(point, cur_centroids):
    return cur_centroids.apply(lambda x: np.linalg.norm(x - point),
axis=1)


def get_closest_centroids(points, cur_centroids):
    return points.apply(lambda x: np.argmin(dists_to_centroids(x,
cur_centroids)), axis=1)


def move_centroids(points, closest_centroids, num_of_centroids):
    return np.array([points[closest_centroids == c].mean(axis=0) for c in
range(num_of_centroids)])


for N in range(2, up_limit+1):
    print(N)
    dict_colors = {i: name for i, (name, col) in
enumerate(random.choices(list(colors.CSS4_COLORS.items()), k=N))}
    list_colors = [name for name, col in
random.choices(list(colors.CSS4_COLORS.items()), k=N)]
    centroids = df.sample(N)

    i = 1
    while True:
        prev_centroids = centroids.copy()
        cl_centroids = get_closest_centroids(df, centroids)
        centroids[:] = move_centroids(df, cl_centroids, N)

        ax = df.plot.scatter(x=0, y=1, c=cl_centroids.apply(lambda x:
dict_colors[x]))
        ax.scatter(centroids.x, centroids.y, c='red')
        ax.set_title('K means {}, war - {}'.format(N, i))
        plt.show()
        # print('step {}'.format(i))
        i += 1
```

```
        if ((prev_centroids - centroids).mean(axis=0).abs() < [0.0001,
0.0001]).all():
            break

    cl_centroids = get_closest_centroids(df, centroids)
    get_metrics()
    rows = []
    for i, centroid in enumerate(centroids.to_numpy()):
        rows.append([i + 1, '({} : {})'.format(*centroid),
sum(cl_centroids == i)])

    res = pd.DataFrame(rows, columns=['Номер кластера', 'Центр кластера',
'Количество элементов в кластере'])
    res.to_csv('Таблица{}.csv'.format(N), index=False)
```