**Лабораторная работа № 3**

**КЛАСТЕРИЗАЦИЯ**

**Цель работы:** научится производить кластерный анализ данных на основе метода К-средних.

**Задача:** изучение возможностей языка Python для реализации методов кластерного анализа, изучение принципов определения оптимального количества кластеров в методах кластерного анализа.

Метод K-средних (K-Means) является популярным алгоритмом кластеризации без учителя, используемым для разделения набора данных на *k* заданных кластеров. Его основная цель — сгруппировать схожие точки данных вместе, минимизируя при этом внутрикластерный разброс. Анализ данных на основе метода K-средних позволяет выявить скрытые закономерности и структуры в неразмеченных данных.

Прежде всего подключаем библиотеки и загружаем имеющийся набор данных, имеющие только типы числовые (а именно возраст, баланс, длительность) (рисунок 1).

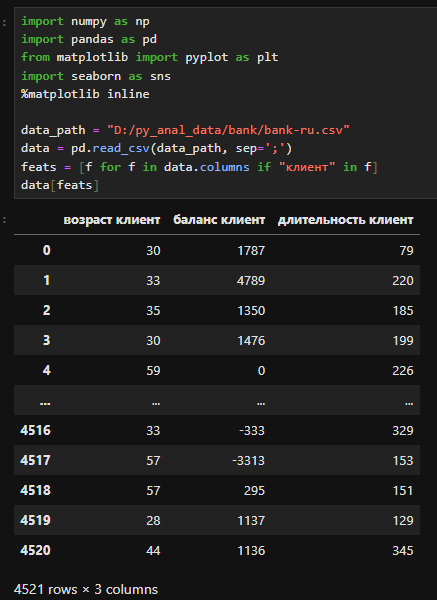


Рисунок 1 – Набор данных для задачи кластеризации.

Для разработки модели необходимо реализовать код, приведенный на рисунке 2 (x1 – возраст и баланс, х2 – возраст и длительность разговора).



Рисунок 2 – код необходимый для реализации разработки модели

Для визуализации был выбран Метод локтя (Elbow Method). Построение графика зависимости суммы квадратов расстояний точек до центроидов своих кластеров (WCSS - Within-Cluster Sum of Squares) от количества кластеров *k*. Оптимальное *k* часто находится в точке "изгиба" графика, где уменьшение WCSS начинает замедляться.

Результаты визуализация числа кластеров представлены на рисунке 3 и 4 (рисунок 3 – возраст и баланс, рисунок 3 – возраст и длительность разговора).

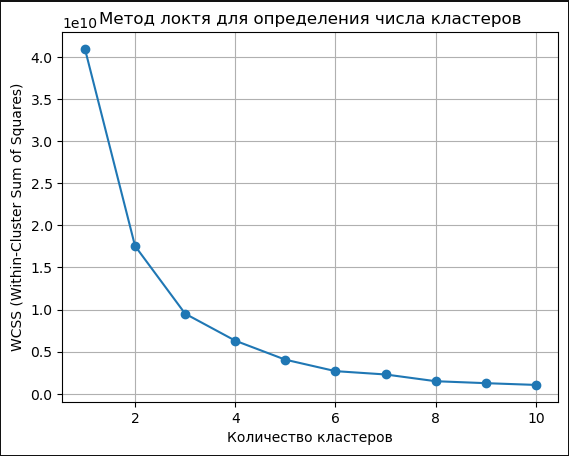


Рисунок 3 – визуализация числа кластеров возраста и баланса

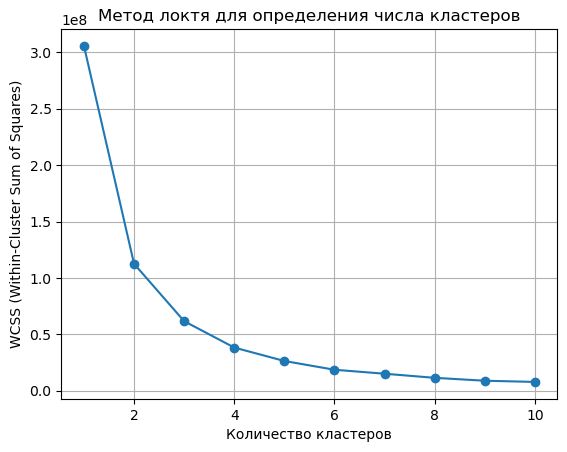


Рисунок 4 – визуализация числа кластеров возраста и длительности разговора

Судя по рисункам 3 и 4, оптимальное число кластеров для рисунков – 3.

Теперь нам надо обучить модель кластеризации для оптимального количества кластеров и визуализировать результаты. Обучим при помощи метода KMeans (из sklearn.cluster). Программная реализация обучения и визуализации на рисунке 5. На рисунках 6 и 7 визуализация.

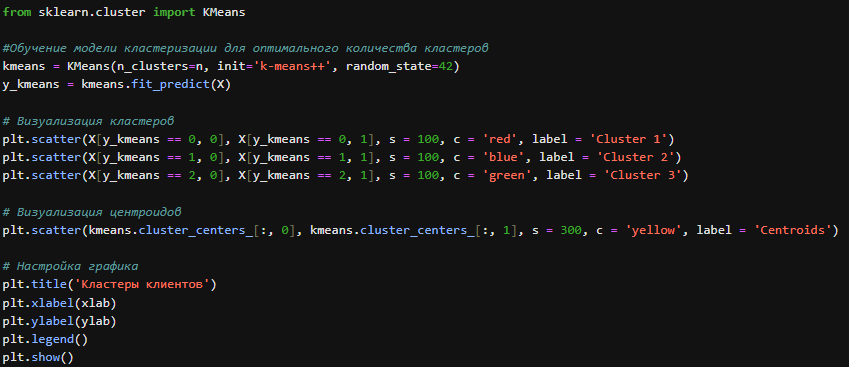


Рисунок 5 – Программная реализация обучения и визуализации

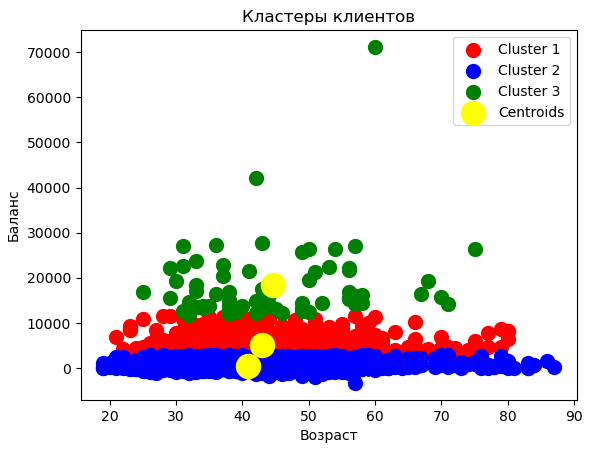


Рисунок 6 – Визуализация результатов расчета возраста и баланса

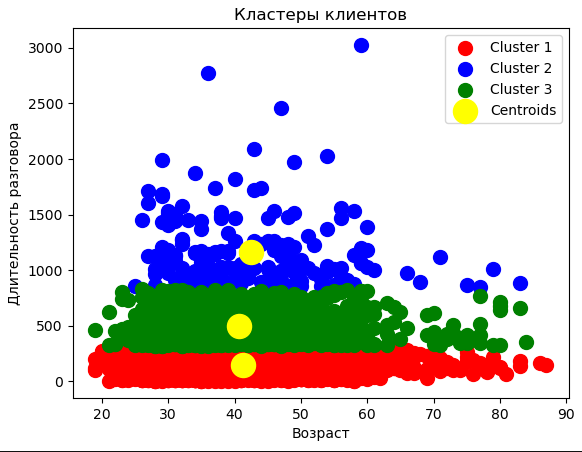


Рисунок 7 – Визуализация результатов расчета возраста и длительности разговора

**Приложение 1**

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

%matplotlib inline

data\_path = "D:/py\_anal\_data/bank/bank-ru.csv"

data = pd.read\_csv(data\_path, sep=';')

x1 = data.iloc[:, [0, 5]].values

x2 = data.iloc[:, [0, 11]].values

def k\_sr(x):

from sklearn.cluster import KMeans

wcss = []

for i in range(1, 11): # Проверяем от 1 до 10 кластеров

kmeans = KMeans(n\_clusters=i, init='k-means++', random\_state=42)

kmeans.fit(x)

wcss.append(kmeans.inertia\_) # Записываем WCSS

# График метода локтя

plt.plot(range(1, 11), wcss, marker='o')

plt.title('Метод локтя для определения числа кластеров')

plt.xlabel('Количество кластеров')

plt.ylabel('WCSS (Within-Cluster Sum of Squares)')

plt.grid()

plt.show()

k\_sr(x1)

k\_sr(x2)

def kl(X, n, xlab, ylab):

from sklearn.cluster import KMeans

#Обучение модели кластеризации для оптимального количества кластеров

kmeans = KMeans(n\_clusters=n, init='k-means++', random\_state=42)

y\_kmeans = kmeans.fit\_predict(X)

# Визуализация кластеров

plt.scatter(X[y\_kmeans == 0, 0], X[y\_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')

plt.scatter(X[y\_kmeans == 1, 0], X[y\_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')

plt.scatter(X[y\_kmeans == 2, 0], X[y\_kmeans == 2, 1], s = 100, c = 'green', label = 'Cluster 3')

# Визуализация центроидов

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], s = 300, c = 'yellow', label = 'Centroids')

# Настройка графика

plt.title('Кластеры клиентов')

plt.xlabel(xlab)

plt.ylabel(ylab)

plt.legend()

plt.show()

kl(x1, 3, 'Возраст', 'Баланс')

kl(x2, 3, 'Возраст', 'Длительность разговора')