Гузенко А.М. Группа 7.2. Вариант 4

Лабораторная работа № 1

Кластерный анализ

“Компьютерная грамотность”

**Цель**

Провести кластерный анализ двумя методами:

1. Методом k-средних.
2. Методом иерархической кластеризации.

**Описание данных**

Количество опрошенных: 1087

Количество признаков: 23

Описание признаков:

*1.* Насколько свободно вы можете работать в следующих приложениях?  
 *\** V1A Обработка текста  
 *\** V1B Графические программы, обработка звука или видео монтаж  
 *\** V1C Базы данных и табличные расчеты  
  
*2.* Насколько хорошо вы владеете следующими языками программирования?  
 \* V2A BASIC  
 \* V2B PASCAL  
 \* V2C C  
 \* V2D машинные языки  
 \* V2E программирование для Интернета (например, HTML)  
 \* V2F Java  
  
*3.* Насколько хорошо вы можете работать в следующих операционных системах?  
 \* V3A DOS  
 \* V3B Windows  
 \* V3C UNIX  
  
*4.* Насколько хорошо вы разбираетесь в возможностях Интернета?  
 \* V4A Email  
 \* V4B WWW  
 \* V4C Chat, IRC  
 \* V4D ICQ  
 \* V4E предложение собственных услуг (например, домашней страницы)  
  
*5.* Насколько хорошо вы разбираетесь в компьютерных играх?  
 \* V5A Как часто Вы играете в компьютерные игры?  
 \* V5B Насколько хорошо Вы разбираетесь в сценах компьютерных игр?  
  
*6.* SEX пол  
*7.* ALTER возраст  
*8.* HERKU 1 = Вост. Германия  
 2 = Зап. Германия  
 3 = Австрия  
*9.* FB код факультета, на котором обучается опрошенный.

**Выполнение работы**

1. Для большего удобства для начала нужно вывести отдельных балл, который будет обозначать компьютерную грамотность, из результатов опроса. Назовем его VALUE и поместим в общую таблицу.

input\_data = pd.read\_table(PATH + 'data\\computer.dat', sep='\t', encoding='utf-8')  
new\_column = []  
for i in range(0, len(input\_data[input\_data.columns[0]])):  
 temp = 0  
 for j in range(0, len(input\_data.columns)):  
 temp = temp + input\_data[input\_data.columns[j]][i]  
 new\_column.append(temp)  
input\_data['VALUE'] = new\_column  
input\_data = input\_data.drop(columns=input\_data.columns[0:19])

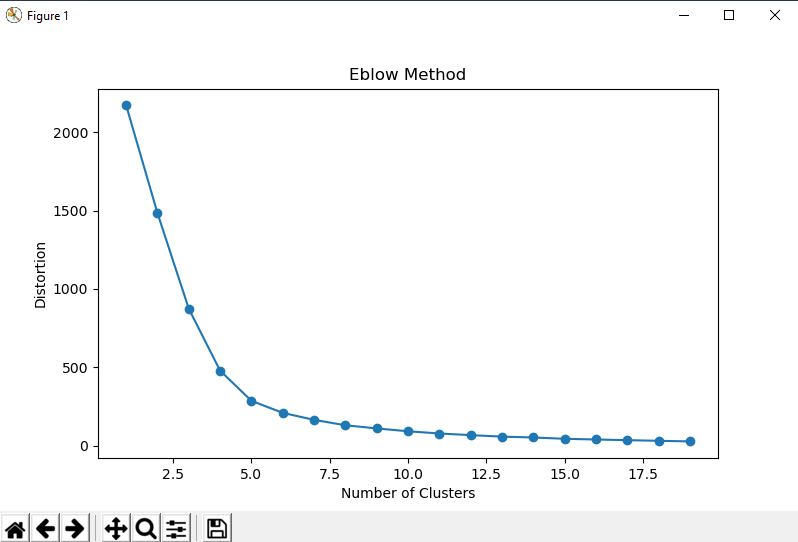
1. Для примера возьмем столбцы VALUE и HERKU. Остальные в ходе дальнейшего анализа будет отбрасывать.
2. Напишем функцию для метода локтя, при помощи которого будет определять оптимальное количество кластеров.

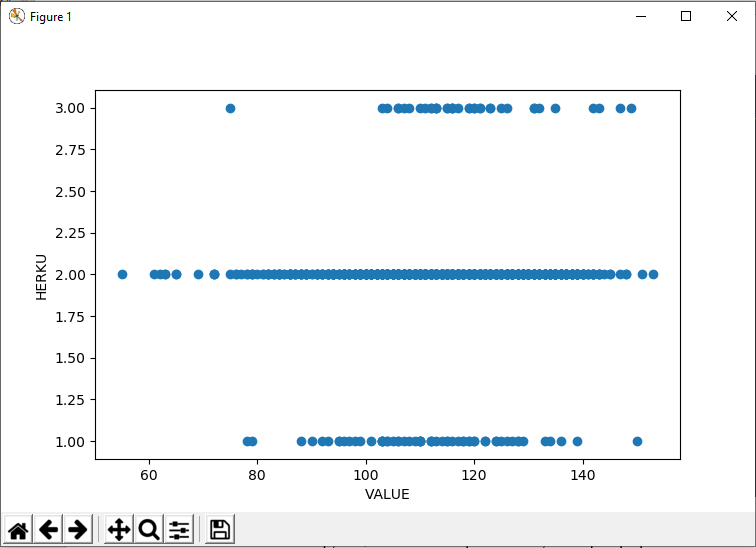
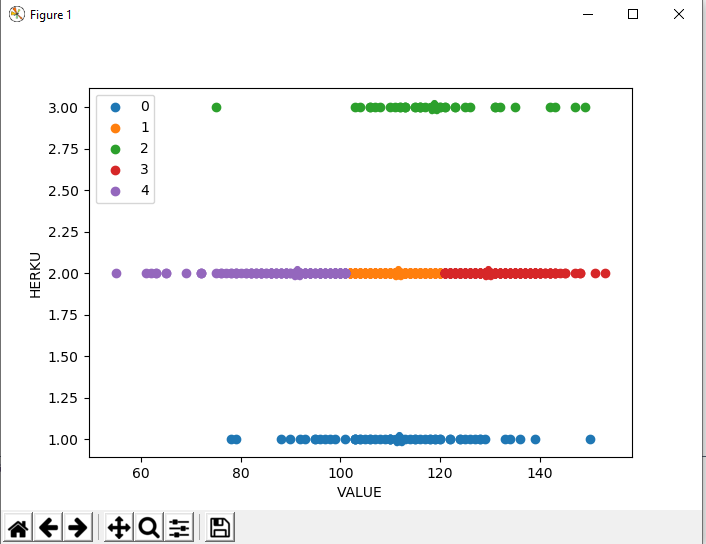
def elbow\_method(data\_elbow):  
 distortions = []  
 for i in range(1, 20):  
 km\_eblow = KMeans(n\_clusters=i,  
 n\_init=10,  
 max\_iter=300,  
 random\_state=0)  
 km\_eblow.fit(data\_elbow)  
 distortions.append(km\_eblow.inertia\_)  
 plt.plot(range(1, 20), distortions, marker='o')  
 plt.title('Eblow Method')  
 plt.xlabel('Number of Clusters')  
 plt.ylabel('Distortion')  
 plt.show()

1. Напишем функцию для кластерного анализа методом k-средних для наших данных. Данные будем стандартизировать перед обработкой.

def value\_herku(input\_data, cluster):  
 input\_data = input\_data.drop(columns=['SEX', 'ALTER', 'FB'])  
 data = pd.DataFrame(preprocessing.scale(input\_data), columns=['HERKU', 'VALUE'])  
 elbow\_method(data)  
 clusters = cluster  
 plt.xlabel('VALUE')  
 plt.ylabel('HERKU')  
 plt.scatter(input\_data['VALUE'], input\_data['HERKU'])  
 plt.show()  
 km = KMeans(n\_clusters=clusters)  
 data\_clusters = km.fit\_predict(data)  
 data = np.array(data)  
 input\_data = np.array(input\_data)  
 centr = km.cluster\_centers\_  
 mean\_of\_array = input\_data.mean(axis=0)  
 std\_of\_array = input\_data.std(axis=0)  
 centr = (centr \* std\_of\_array) + mean\_of\_array  
 plt.xlabel('VALUE')  
 plt.ylabel('HERKU')  
 scatter\_list = []  
 for i in range(0, clusters):  
 scatter\_list.append(plt.scatter(input\_data[data\_clusters == i, 1], input\_data[data\_clusters == i, 0], c=list(mcolors.TABLEAU\_COLORS.values())[i]))  
 plt.scatter(centr[i, 1], centr[i, 0], marker='\*', c=list(mcolors.TABLEAU\_COLORS.values())[i], linewidths=5)  
 plt.legend(scatter\_list, [i for i in range(0, clusters)])  
 plt.show()

1. Вызовем функцию с 5 кластерами. Получим три графика.
   1. Метод Локтя.



* 1. Данные на графике.
  2. Данные, разбитые на кластеры с центром кластеров.

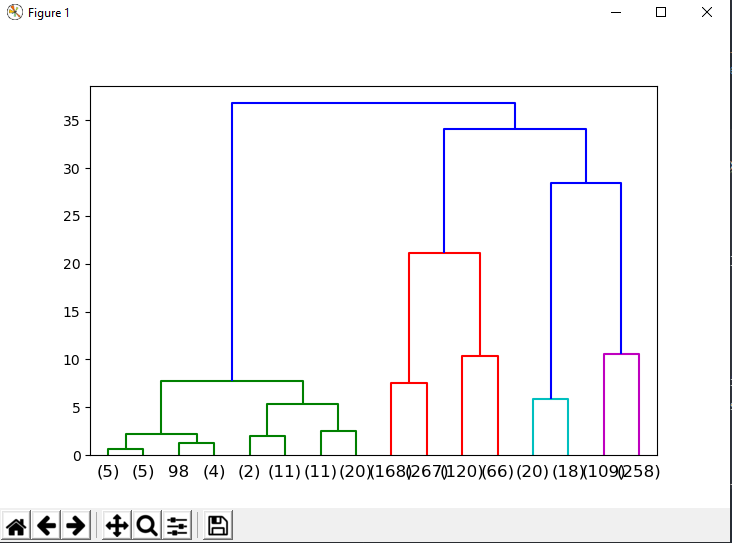
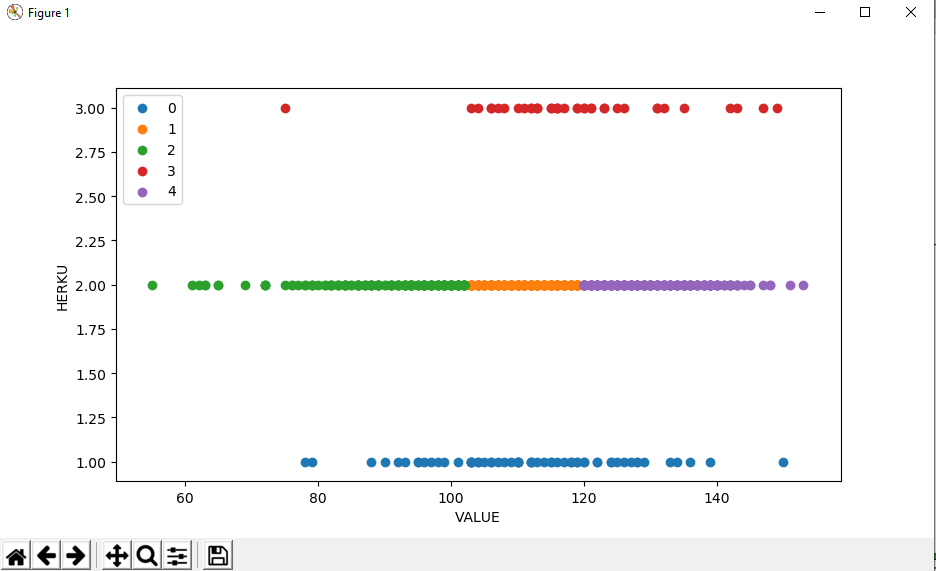
По графику 1.1. можно сделать вывод что 5 кластеров оптимальное количество кластеров для метода k-средних.

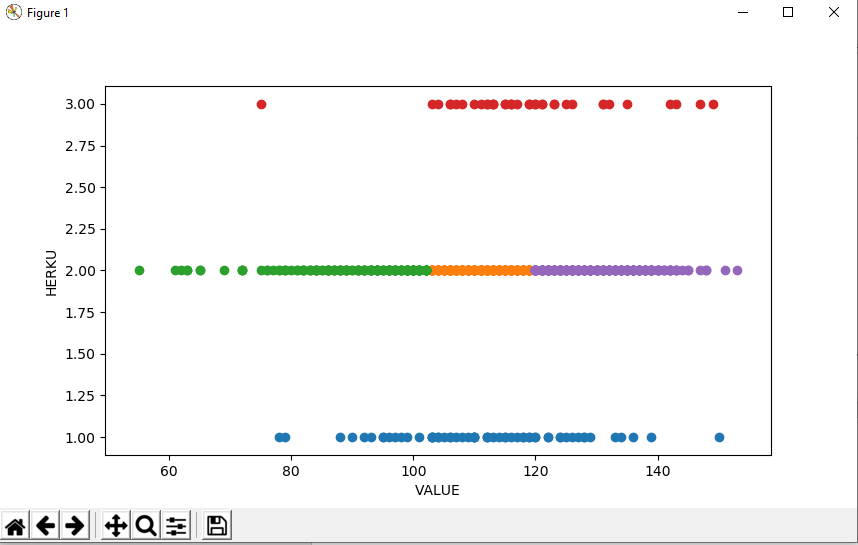
1. Напишем функцию для иерархического метода с расстояния между кластерами, так же внутри функции будем отображать дендрограмму. Данные будем стандартизировать перед обработкой.

def hierarchical\_number\_of\_cluster(input\_data, q):  
 input\_data = input\_data.drop(columns=['SEX', 'ALTER', 'FB'])  
 data = preprocessing.scale(input\_data)  
 dist = pdist(data, 'euclidean')  
 link = linkage(dist, 'ward')  
 dendrogram(link, p=3, truncate\_mode='level')  
 plt.show()  
 plt.xlabel('VALUE')  
 plt.ylabel('HERKU')  
 input\_data = np.array(input\_data)  
 clusters = fcluster(link, q, criterion='maxclust')  
 scatter\_list = []  
 for i in range(0, q):  
 scatter\_list.append(plt.scatter(input\_data[clusters == i+1, 1], input\_data[clusters == i+1, 0], c=list(mcolors.TABLEAU\_COLORS.values())[i]))  
 plt.legend(scatter\_list, [i for i in range(0, q)])  
 plt.show()

1. Напишем функцию для иерархического метода с заданием расстояния между объектами, необходимое количество кластеров метод будет определять сам в зависимости от минимального заданного расстояния, так же внутри функции будем отображать дендрограмму. Данные будем стандартизировать перед обработкой.

def hierarchical\_dist(input\_data, d):  
 input\_data = input\_data.drop(columns=['SEX', 'ALTER', 'FB'])  
 data = preprocessing.scale(input\_data)  
 dist = pdist(data, 'euclidean')  
 link = linkage(dist, 'ward')  
 dendrogram(link, p=3, truncate\_mode='level')  
 plt.show()  
 plt.xlabel('VALUE')  
 plt.ylabel('HERKU')  
 input\_data = np.array(input\_data)  
 clusters = fcluster(link, d, criterion='distance')  
 print('Number of Cluster = ', max(clusters))  
 scatter\_list = []  
 for i in range(0, max(clusters)):  
 scatter\_list.append(plt.scatter(input\_data[clusters == i+1, 1], input\_data[clusters == i+1, 0]))  
 plt.show()

1. Вызовем поочередно функцию из 6 и 7 пункта.
   1. Дендограмма (в двух функциях аналогичные)
   2. Кластеры с заданием минимального расстояния между кластерами. t = 5.
   3. Кластеры с заданием минимального расстояния между объектами. t = 15.



**Использованные библиотеки.**

import os  
import pandas as pd  
from sklearn import preprocessing  
from sklearn.cluster import KMeans  
from scipy.cluster.hierarchy import dendrogram, linkage  
from scipy.cluster.hierarchy import fcluster  
from scipy.spatial.distance import pdist  
import matplotlib.pyplot as plt  
import matplotlib.colors as mcolors  
import numpy as np

**Вывод**

При помощи метода k-средних, по центру кластеров, мы сразу поняли, что обучающиеся в Западной Германии лучше всех разбираются в компьютерных технологиях. На втором месте идет Австрия и на третьем Восточная Германия. Заметим, что при помощи иерархической кластеризации, кластеры определяются более четко и выделить группы проще. Аналогично можно провести анализ и для других данных аналогично HERKU. Для SEX получим, что пол под номером 1 лучше разбирается в компьютерных технологиях, чем пол под номер 2. Для ALTER получим, что люди от 30 лет очень хорошо разбираются в компьютерных технология, в то время как до 30 лет примерно одинаково. Для FB показал лучше всего себя показали факультеты с 16 по 23, а факультеты с 1 по 3 включительно показали себя с худшей стороны.