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])
```

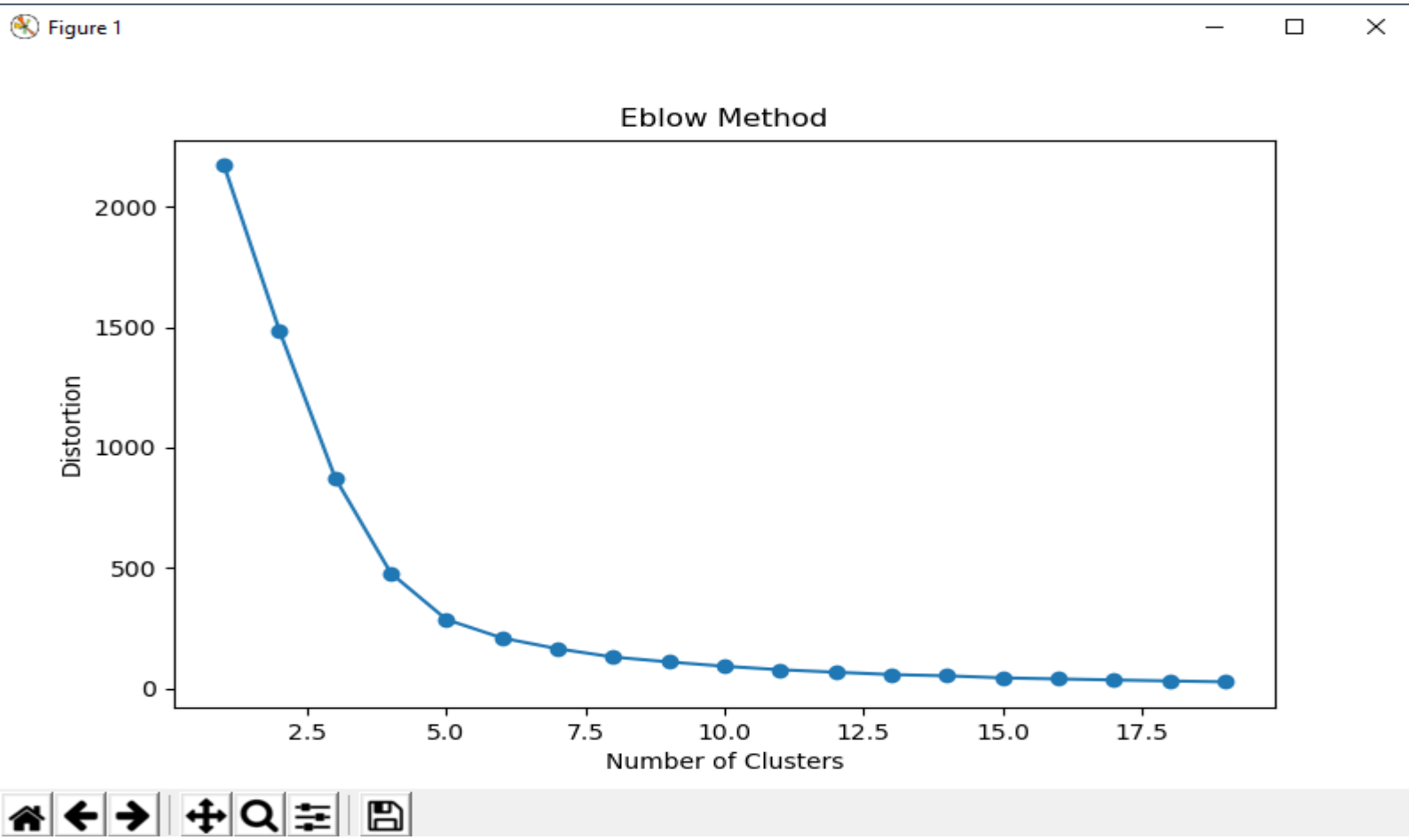
2. Для примера возьмем столбцы VALUE и HERKU. Остальные в ходе дальнейшего анализа будет отбрасывать.
3. Напишем функцию для метода локтя, при помощи которого будет определять оптимальное количество кластеров.

```
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()
```

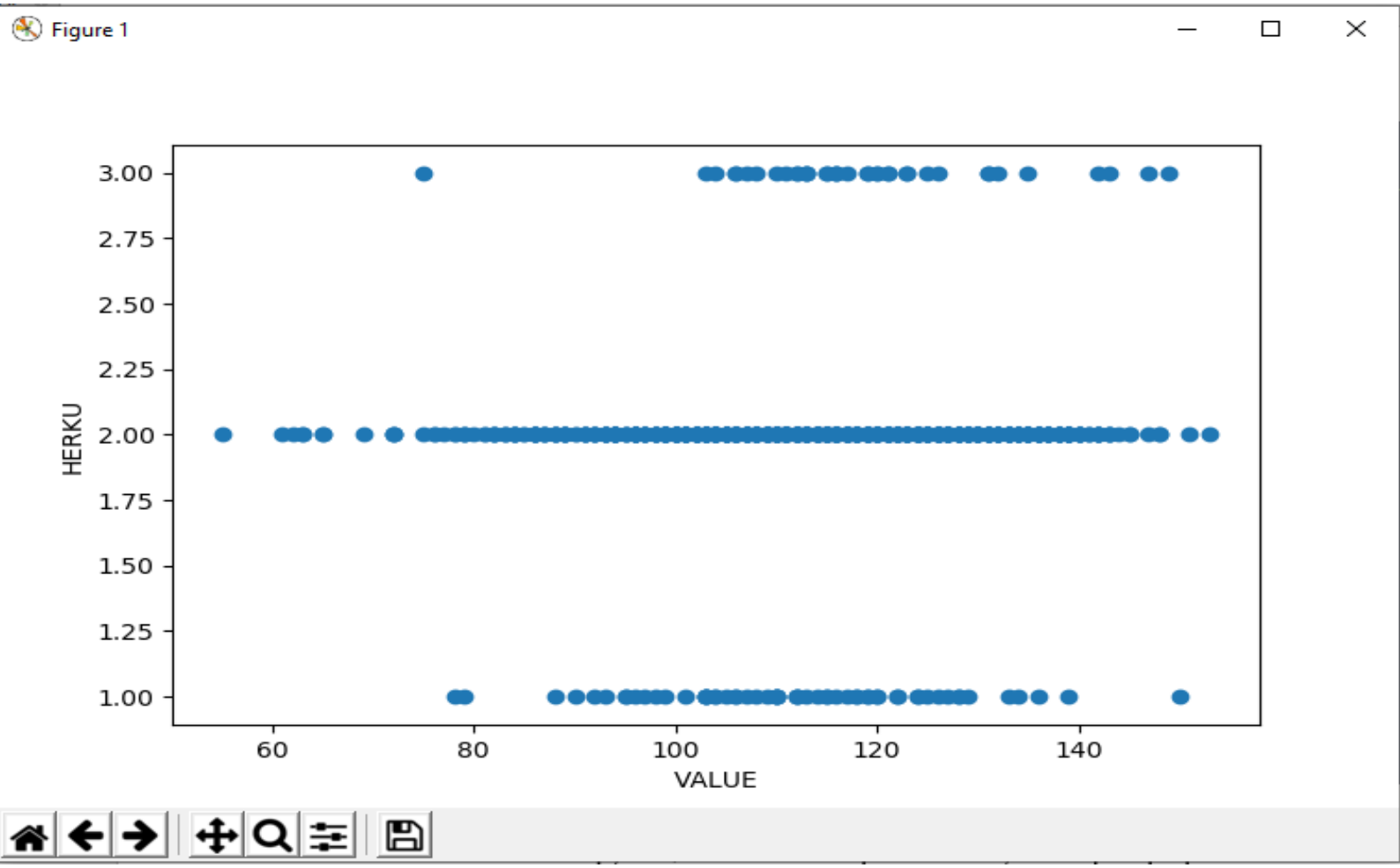
4. Напишем функцию для кластерного анализа методом k-средних для наших данных. Данные будем стандартизировать перед обработкой.

```
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()
```

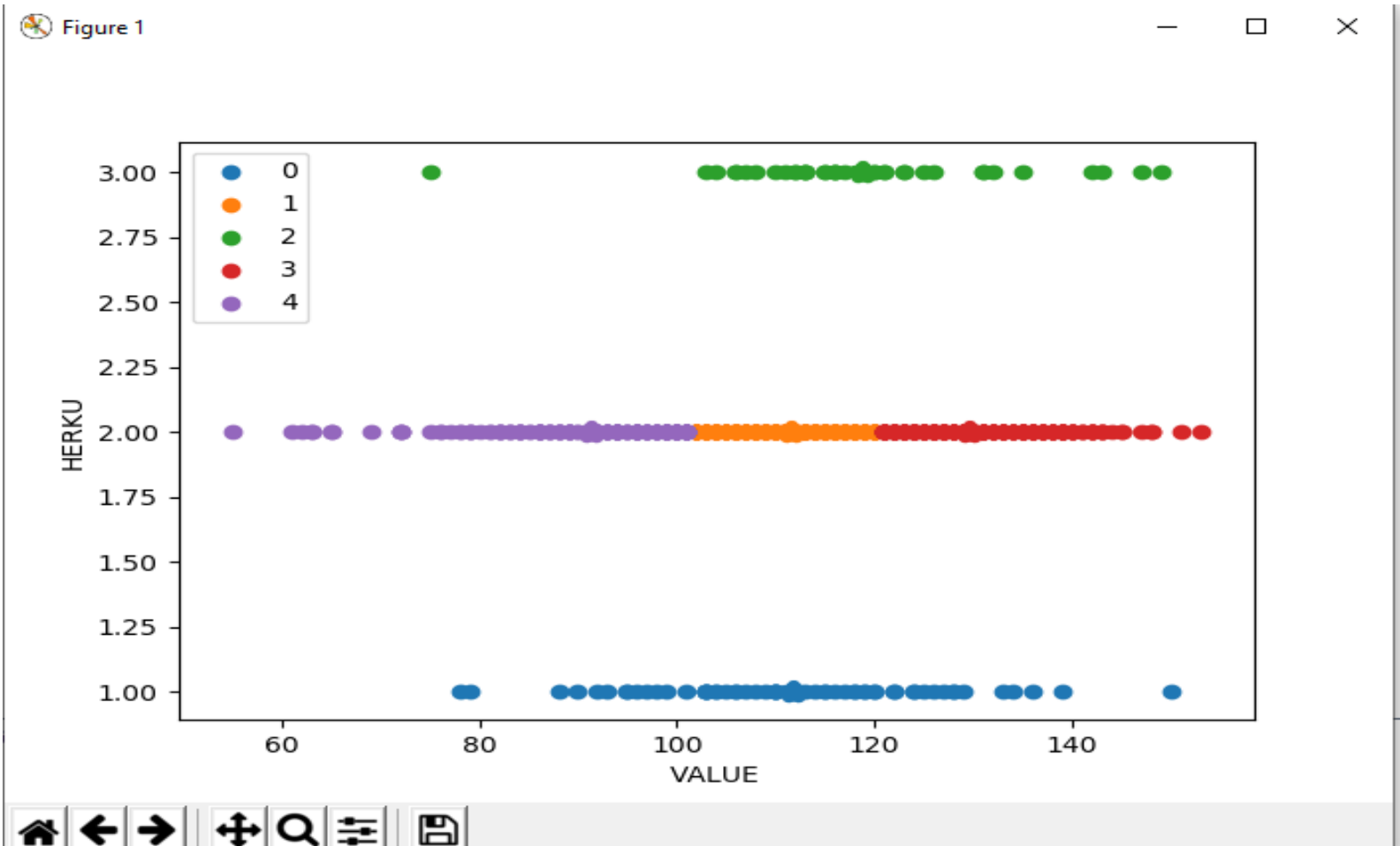
5. Вызовем функцию с 5 кластерами. Получим три графика.
1.1 Метод Локтя.



1.2 Данные на графике.



1.3 Данные, разбитые на кластеры с центром кластеров.



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

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

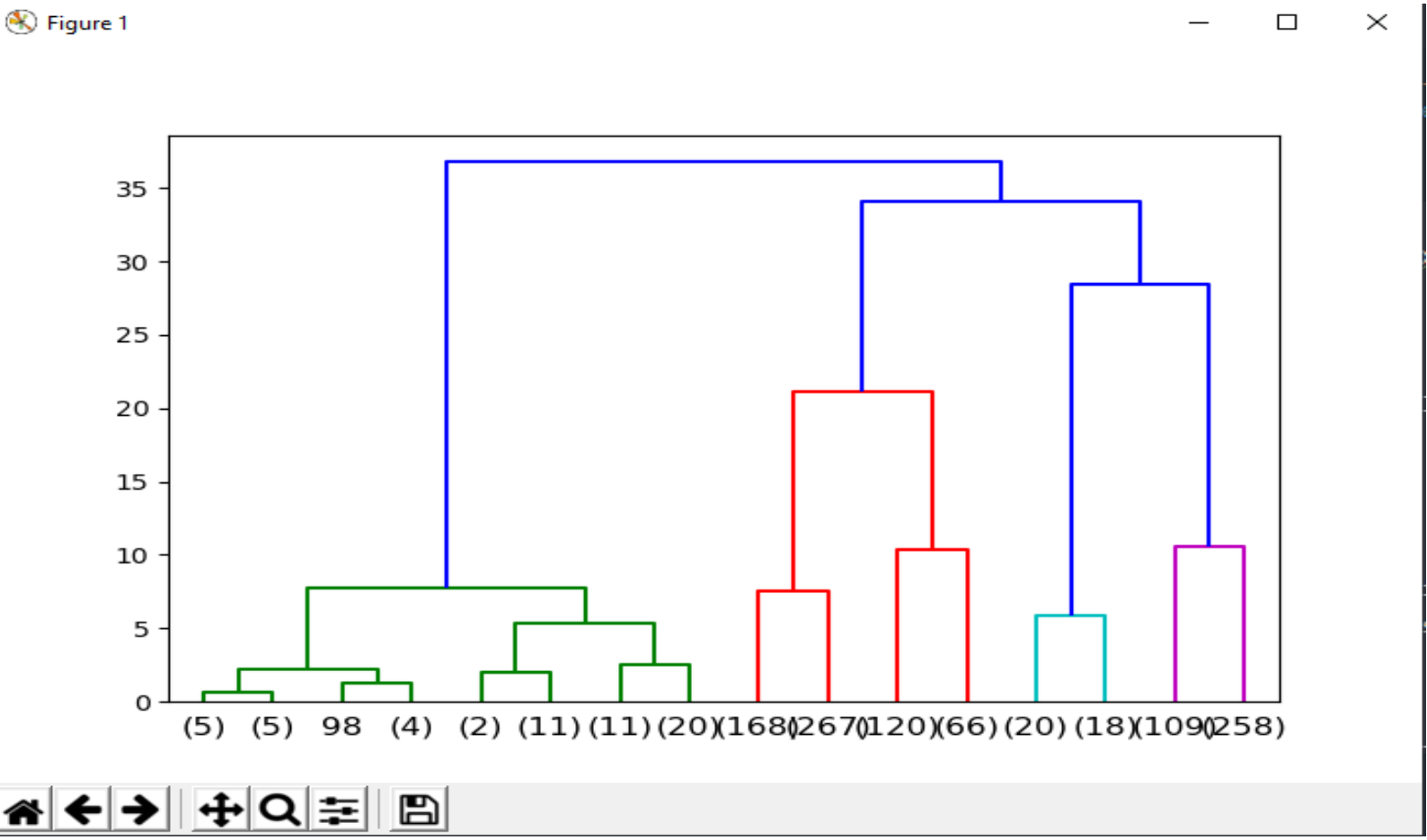
```
def hierarchical_number_of_cluster(input_data, q):
    input_data = input_data.drop(columns=['SEX', 'ALTER', 'FB'])
    data = preprocessing.scale(input_data)
    dist = pdist(data, 'euclidean')
    link = linkage(dist, 'ward')
    dendrogram(link, p=3, truncate_mode='level')
    plt.show()
    plt.xlabel('VALUE')
    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()
```

7. 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.

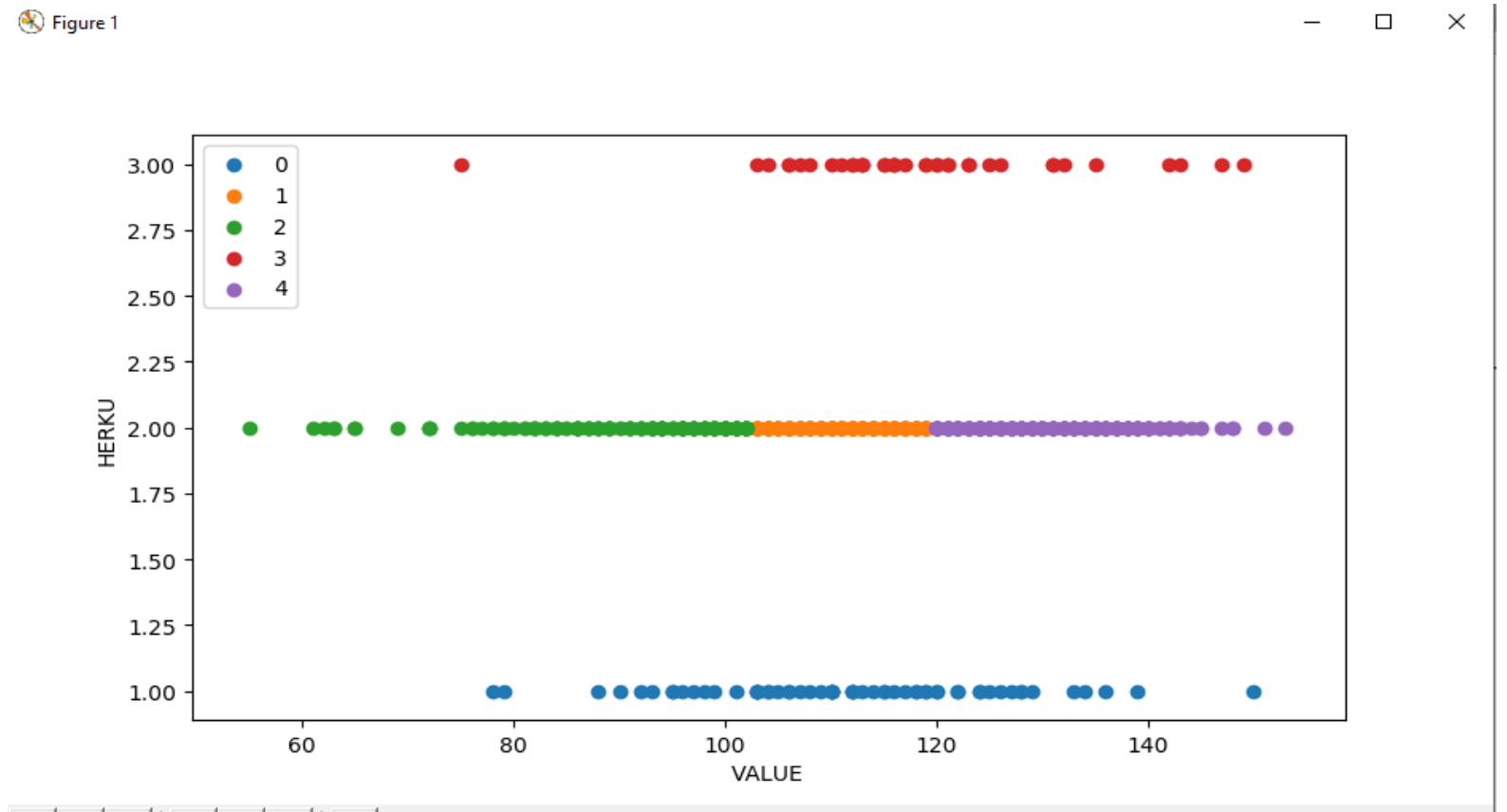
```
def hierarchical_dist(input_data, d):
    input_data = input_data.drop(columns=['SEX', 'ALTER', 'FB'])
    data = preprocessing.scale(input_data)
    dist = pdist(data, 'euclidean')
    link = linkage(dist, 'ward')
    dendrogram(link, p=3, truncate_mode='level')
    plt.show()
    plt.xlabel('VALUE')
    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()
```

8. Вызовем поочередно функцию из 6 и 7 пункта.

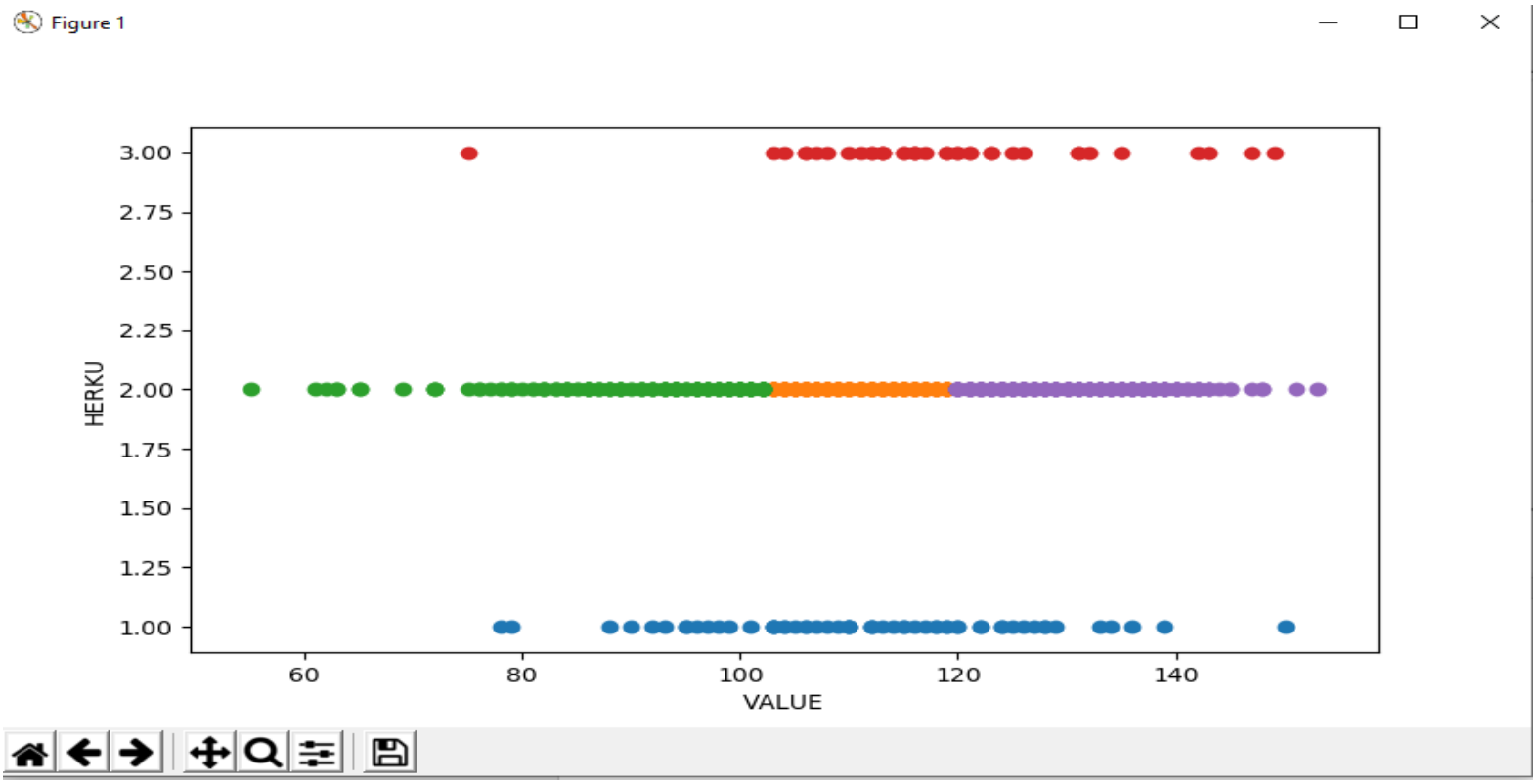
1.1. Дендограмма (в двух функциях аналогичные)



1.2. Кластеры с заданием минимального расстояния между кластерами. t = 5.



1.3. 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.