ГУАП

КАФЕДРА № 41

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

ассистент		В.В. Боженко
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

Отчет о лабораторной работе №3

Кластеризация данных

По курсу: Введение в анализ данных

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 4917		Е.А. Ясиновский
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2022

Цель работы: изучить алгоритмы и методы кластерного анализа на практике.

Вариант:

Содержит информацию о заболеваниях печени.

1. Возраст

2. Пол

3. Общий билирубин

4. Прямой билирубин

5. Щелочная фосфатаза

6. АЛТ

7. АСТ

8. Общее число белков

9. Альбумин

10. Отношение альбумина к глобулину

11.Диагноз(Болен\Здоров)

Выполнение работы:

Сначала загрузил данные и предварительно подготовил их к анализу: удалил явные и неявные дубликаты, проверил пропуски строк и устранил их

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
df = pd.read_csv("liver.csv")
df.duplicated().where(lambda x: x == True).dropna() ##  Явных дубликатов не найдено
df = df.dropna().reset_index()

### заккоментировал вывод строк, чтобы они не мешали отображению. Выводом воспользовался чтобы устранить неявные дубликаты
# for col in df.columns:
#     print(df[col].unique())

df["Gender"] = df["Gender"].replace("Mal","Male")
df["Gender"] = df["Gender"].replace(["Male","Female"],[0,1])
df["Dataset123"] = df["Dataset123"].replace(["yes","no"],["1","2"])
df["Dataset123"] = df["Dataset123"].replace(["1","2"],["1","0"])
df["Aspartate_Aminotransferase"] = df["Aspartate_Aminotransferase"].replace("3a4","34")
df = df.rename(columns={"Dataset123":"IsSick"})
df = df.astype({
    'Gender':'int64',
    "IsSick":"int64",
    "Alkaline_Phosphotase":"int64",
    "Aspartate_Aminotransferase":"int64"})
df.head()
```

✓ 4.3s

	index	Age	Gender	TotalBilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio	IsSick
0	0	65	1	0.7	0.1	187	16	18	6.8	3.3	0.90	1
1	1	62	0	10.9	5.5	699	64	100	7.5	3.2	0.74	1
2	2	62	0	7.3	4.1	490	60	68	7.0	3.3	0.89	1
3	3	58	0	1.0	0.4	182	14	20	6.8	3.4	1.00	1
4	4	72	0	3.9	2.0	195	27	59	7.3	2.4	0.40	1

Рисунок 1 – Загрузка датасета

Затем была выполнена кластеризация объектов иерархическим агломеративным методом. Для вычисления расстояния между кластерами был выбран метод "ward". Для вычисления расстояния между объектами выбрана "euclidean" метрика. Код для построения дендрограммы представлен на рисунке 2. Сама дендрограмма представлена на рисунке 3.

Стандартизируем данные с помощью объекта StandardScaler из модуля sklearn.preprocessing. Удалим целевой показатель из датафрейма.

```
IsSick = list(df.pop('IsSick'))
index = list(df.pop('index'))
from sklearn.preprocessing import StandardScaler, MinMaxScaler
sc = StandardScaler()
sc.fit(df)
df_sc = sc.fit_transform(df)
```

✓ 1.3s

Теперь из стандартизированных данных построим дендрограмму и определим оптимальное количество кластеров для кластеризации с помощью KMeans

```
from scipy.cluster.hierarchy import dendrogram, linkage
linked = linkage(df_sc,method='ward', metric="euclidean")
plt.figure(figsize=(15,15))
dendrogram(linked,orientation='top')
plt.title('Иерархическая агломерация')
plt.show()
```

✓ 3.6s

Рисунок 2 — Код нормализации датафрейма и построение дендрограммы.

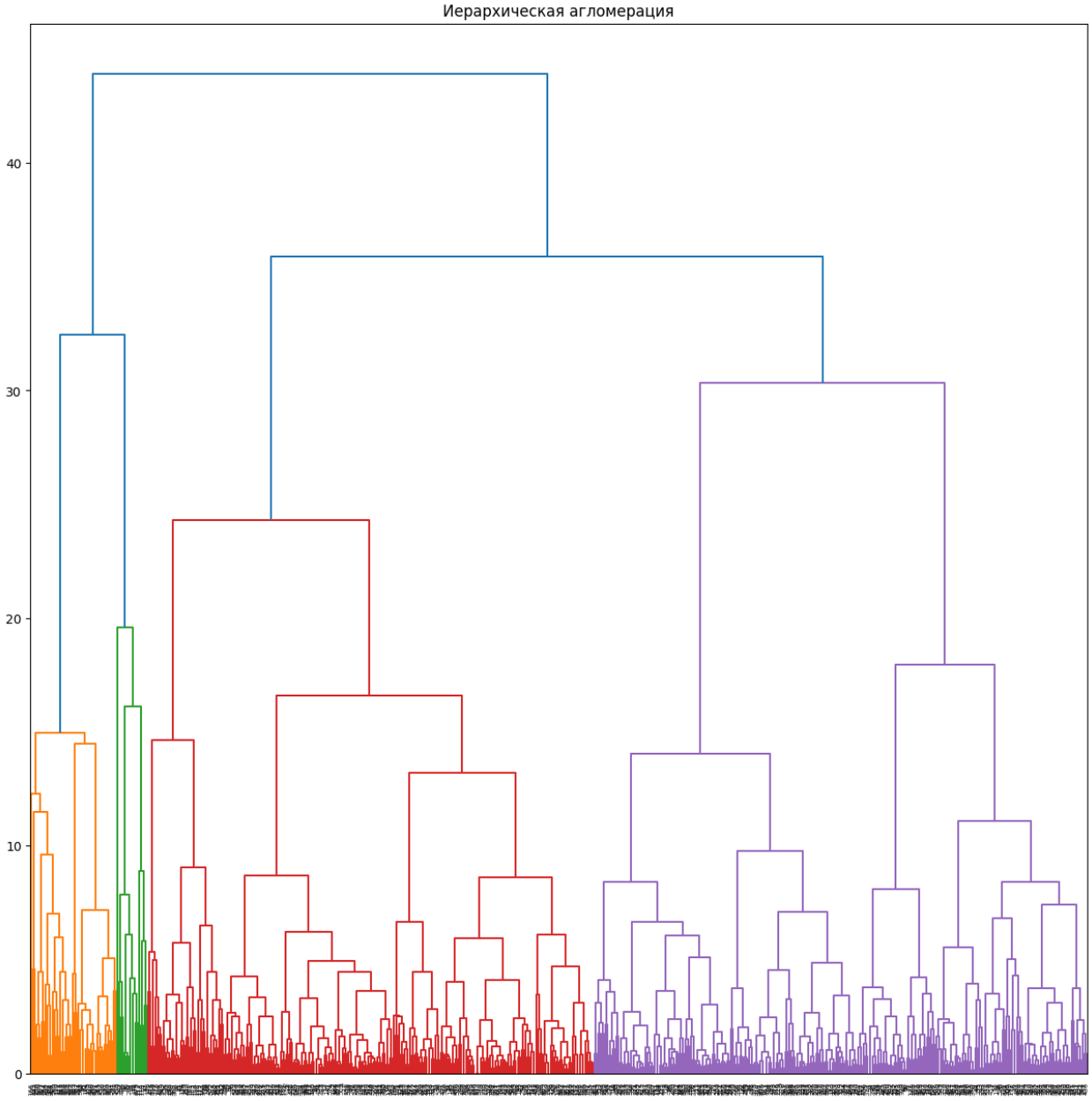

Рисунок 3 – Дендрограмма иерархической кластеризации

Из дендрограммы можно сделать вывод, что оптимальное число кластеров равно 4.ф

Далее с помощью таблицы средних значений в каждом кластере были определены признаки, которые оказали наибольшее влияние на выделение кластеров (рисунок 4).

```
from scipy.cluster.hierarchy import fcluster
df["cluster"] = fcluster(linked,4,criterion="maxclust")
df.groupby("cluster").mean()
```

✓ 0.3s

cluster	Age	Gender	TotalBilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio
1	42.148936	0.148936	19.631915	9.308511	364.063830	115.829787	200.808511	6.510638	2.644681	0.793191
2	40.764706	0.058824	6.958824	3.252941	306.470588	919.294118	1267.941176	5.929412	2.805882	0.892353
3	51.122449	0.024490	2.215918	0.982041	339.918367	49.910204	67.575510	5.960816	2.690612	0.815184
4	39.756458	0.464945	1.230258	0.481181	231.682657	50.523985	60.354244	6.980812	3.660148	1.102509

Рисунок 4 – Признаки, оказавшие наибольшее влияние на выделение

Из таблицы видно, что наибольшая дельта (в процентах) среди строк заметна у общего и прямого билирубина , АЛТ и АСТ. Общие количество белков и альбуминов а также их соотношение во всех кластерах примерно одинаковое, за исключением четвертого кластера, в которых эти значения повышены, а щелочная фосфатаза в нем понижена.

Затем была выполнена кластеризация объектов методом k-средних. Число кластеров n_clusters было задано 2(Больные и здоровые). Код кластеризации представлен на рисунке 5.

```
from sklearn.cluster import KMeans
kmens = KMeans(n_clusters=2,random_state=0)
kmens.fit(df_sc)
```

Рисунок 5 – Метод k-средних

Затем было рассчитано евклидово расстояние между кластерами. (рисунок 6).

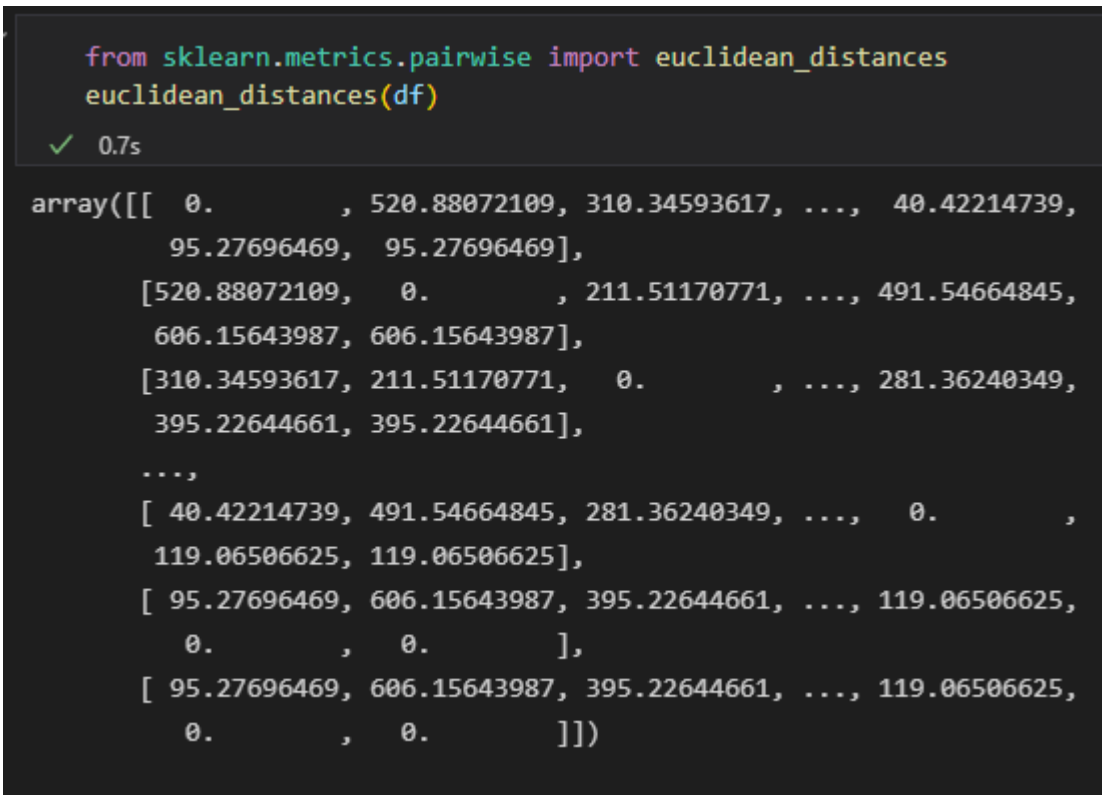

Рисунок 6 – Евклидово расстояние

После этого были определены объекты, относящиеся к каждому кластеру. (рисунок 7).

Рисунок 7 – Объекты, относящиеся к каждому кластеру

Далее были определены признаки, оказавшие наибольшее влияние на выделение кластеров. (рисунок 8).

```
df["cluster"] = kmens.labels_
df.groupby("cluster").mean()
```

✓ 6.7s

```
:\Users\Erop\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarni
n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
 warnings.warn(
```

cluster	Age	Gender	TotalBilirubin	Direct_Bilirubin	Alkaline_Phosphotase	Alamine_Aminotransferase	Aspartate_Aminotransferase	Total_Protiens	Albumin	Albumin_and_Globulin_Ratio
0	45.536232	0.159420	15.640580	7.279710	537.826087	315.565217	458.376812	6.286957	2.610145	0.740725
1	44.679061	0.252446	1.640117	0.707436	256.902153	49.350294	63.164384	6.507241	3.215264	0.978160

Рисунок 8 – Признаки, оказавшие наибольшее влияние на выделение кластеров

Здесь мы видим, что основными показателями для кластеров стали билирубин, щелочная фосфатаза, АМТ и АСТ

После этого было определено число кластеров с помощью Elbow метода. Код метода представлен на рисунке 9, результирующий график на рисунке 10.

```
models  = [KMeans(n_clusters=i, random_state=0).fit(df_sc) for i in range(1,20)]
dist = [model.inertia_ for model in models]
fig = plt.figure(figsize=(15,15))
plt.plot(range(1,20),dist,marker="o")
plt.xticks(range(1,20),range(1,20))
plt.xlabel("Кол-во кластеров")
plt.ylabel("Инерция")
plt.title("Elbow метод")
plt.show()
```

Рисунок 9 – Код для определения оптимального числа кластеров

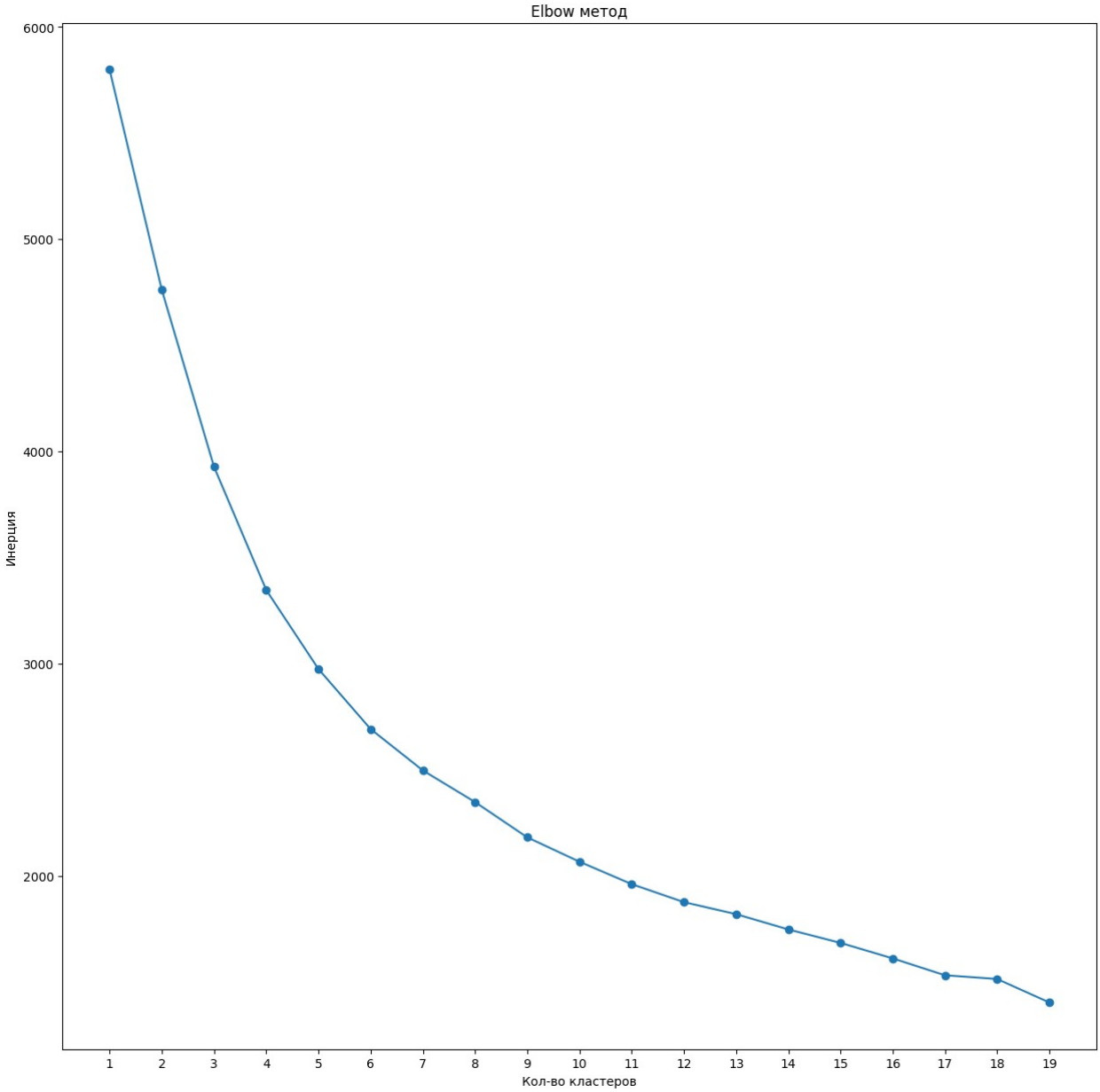

Рисунок 10 — График Elbow метода

По результату можно сделать вывод, что оптимальным числом кластеров скорее всего является 3.

Ссылка на Jupyter notebook:

hhttps://github.com/EgorYasinovskiy/Data-Analys.git

Вывод: во время выполнения данной лабораторной работы были получены практические навыки использования алгоритмов и методов кластерного анализа, таких как, иерархический алгомеративный метод и метод k-средних, что включало в себя нахождение оптимального числа

кластеров, построение дендрограмма, расчет евклидова расстояния между кластерами и определение объектов, относящихся к каждому кластеру. Характеристики, оказавшие наибольшее влияние на выделение кластеров, оказались одинаковыми для обоих методов, но назначения кластеров отличаются и интерпретировать их можно по-разному.