**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №1**

**по дисциплине «Машинное обучение»**

**Тема: Предобработка данных**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6307 |  | Давыдова Н. П. |
| Преподаватель |  | Жангиров Т. Р. |

Санкт-Петербург

2020

Содержание

[**Загрузка данных** 2](#_Toc54099122)

[**Стандартизация данных:** 5](#_Toc54099123)

[**Приведение к диапазону** 7](#_Toc54099124)

[**Нелинейные преобразования** 11](#_Toc54099125)

[**Дискретизация признаков** 14](#_Toc54099126)

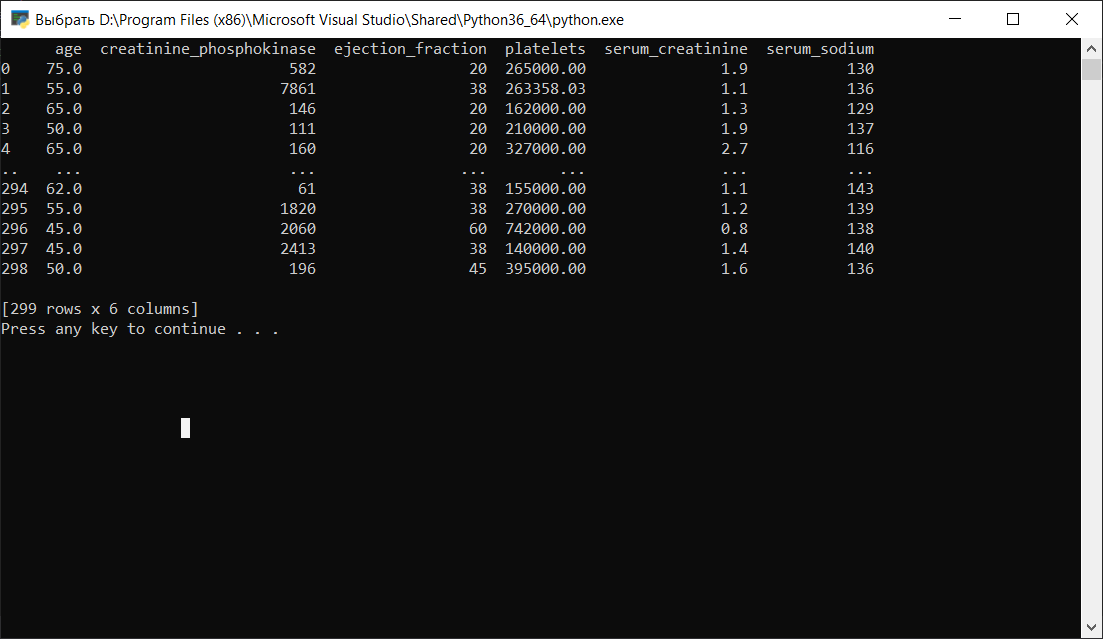
[**Вывод** 15](#_Toc54099127)

# **Загрузка данных**

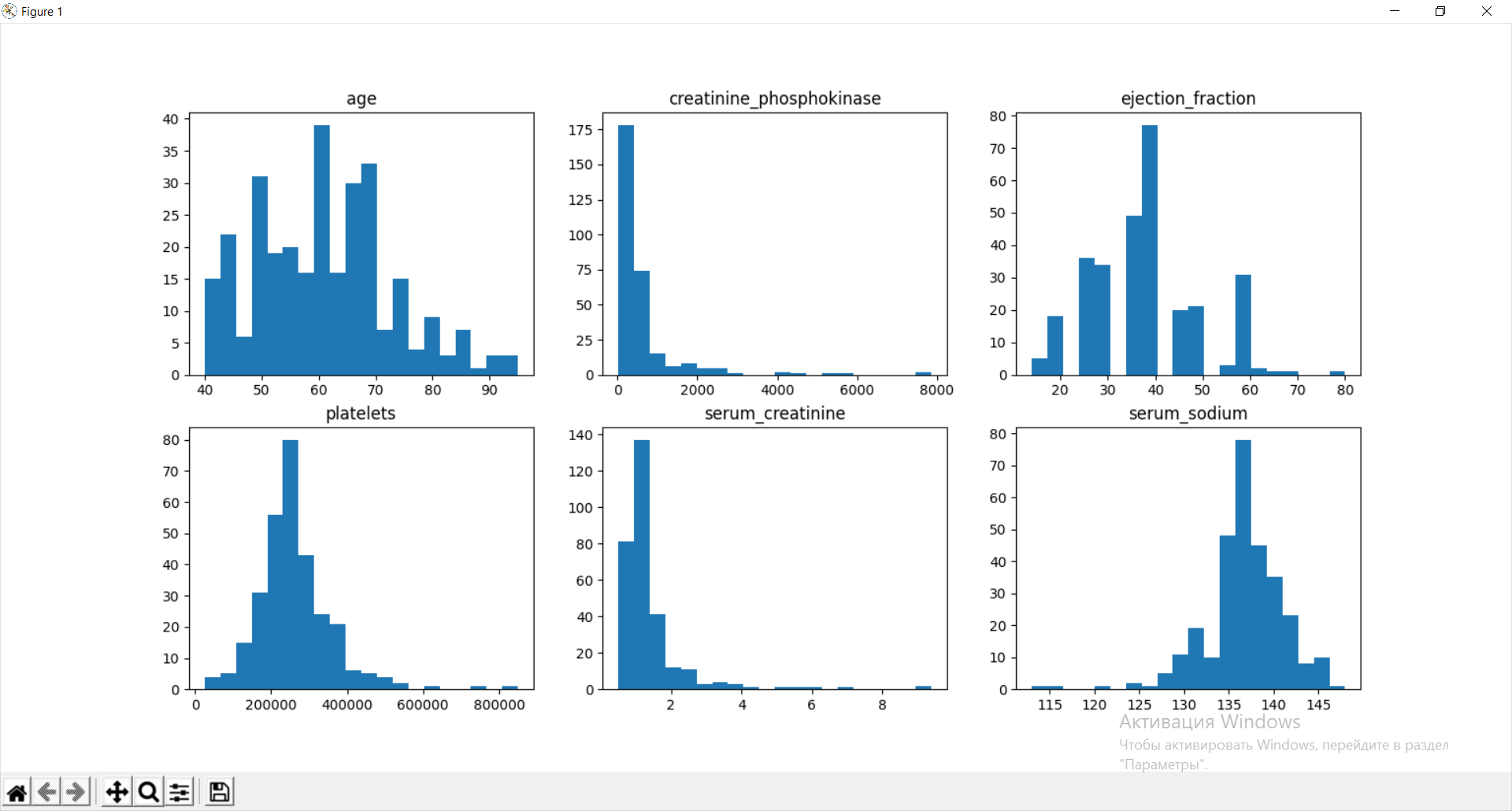
1. Загрузила датасет по ссылке: https://www.kaggle.com/andrewmvd/heart-failure-clinical-dat

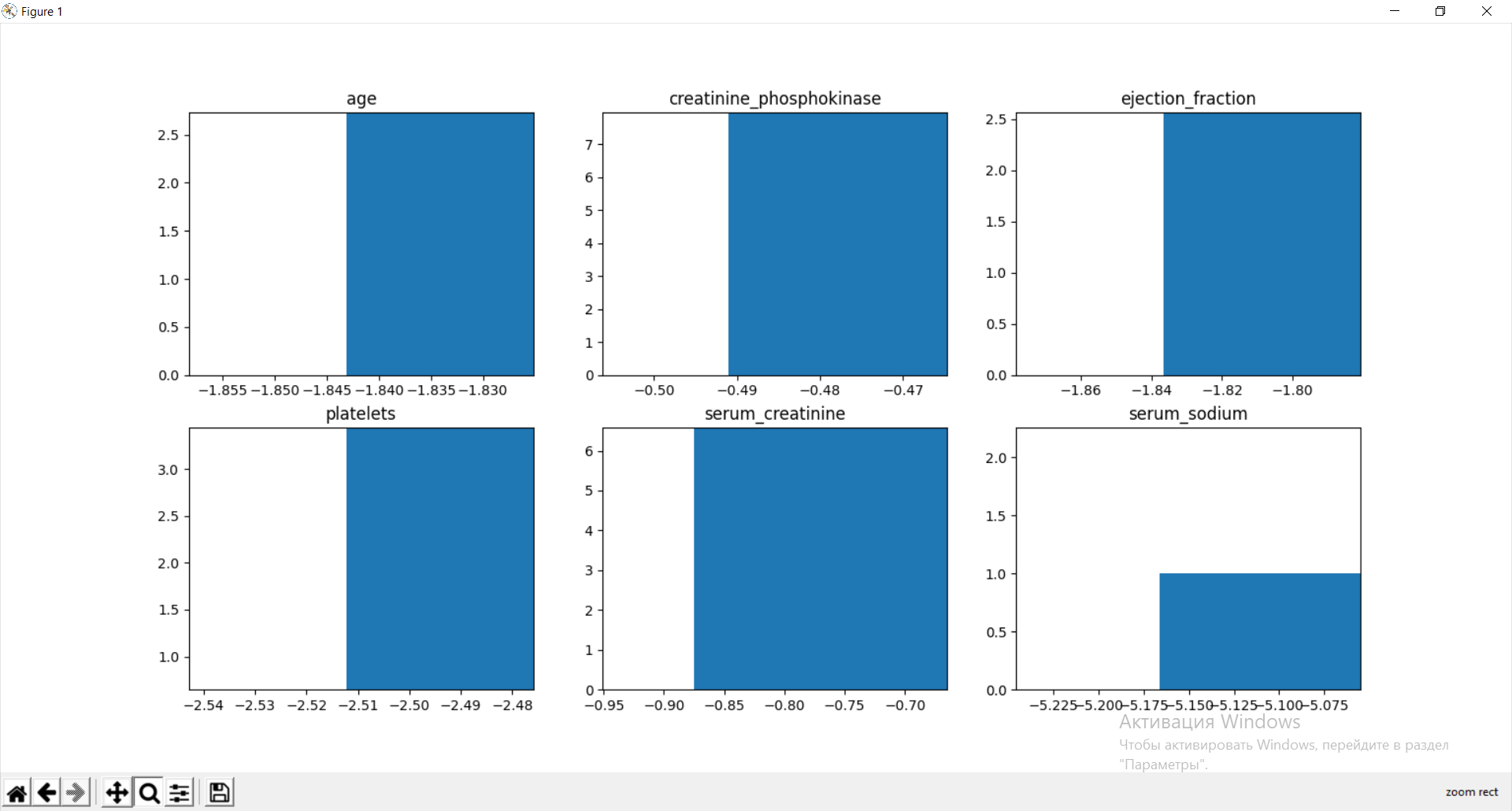
2. Загрузила датасет в датафрейм, и исключила бинарные признаки и

признак времени.

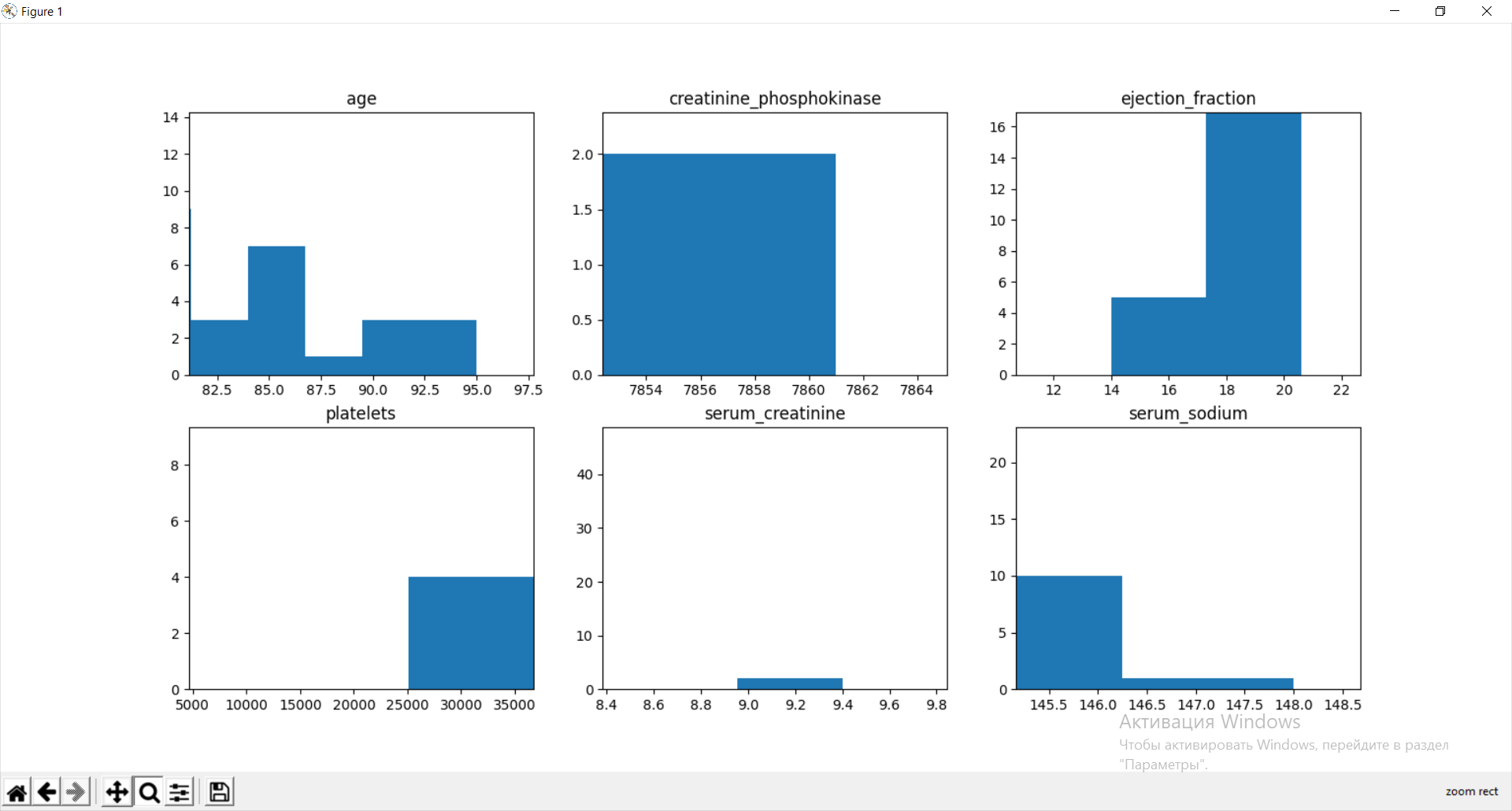


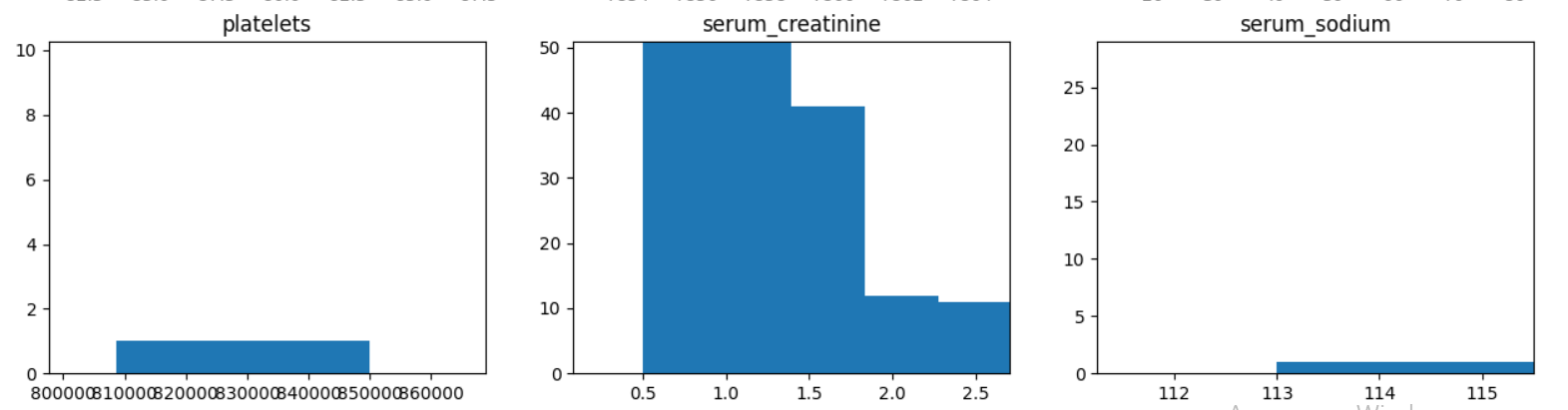
3. Построила гистограммы признаков



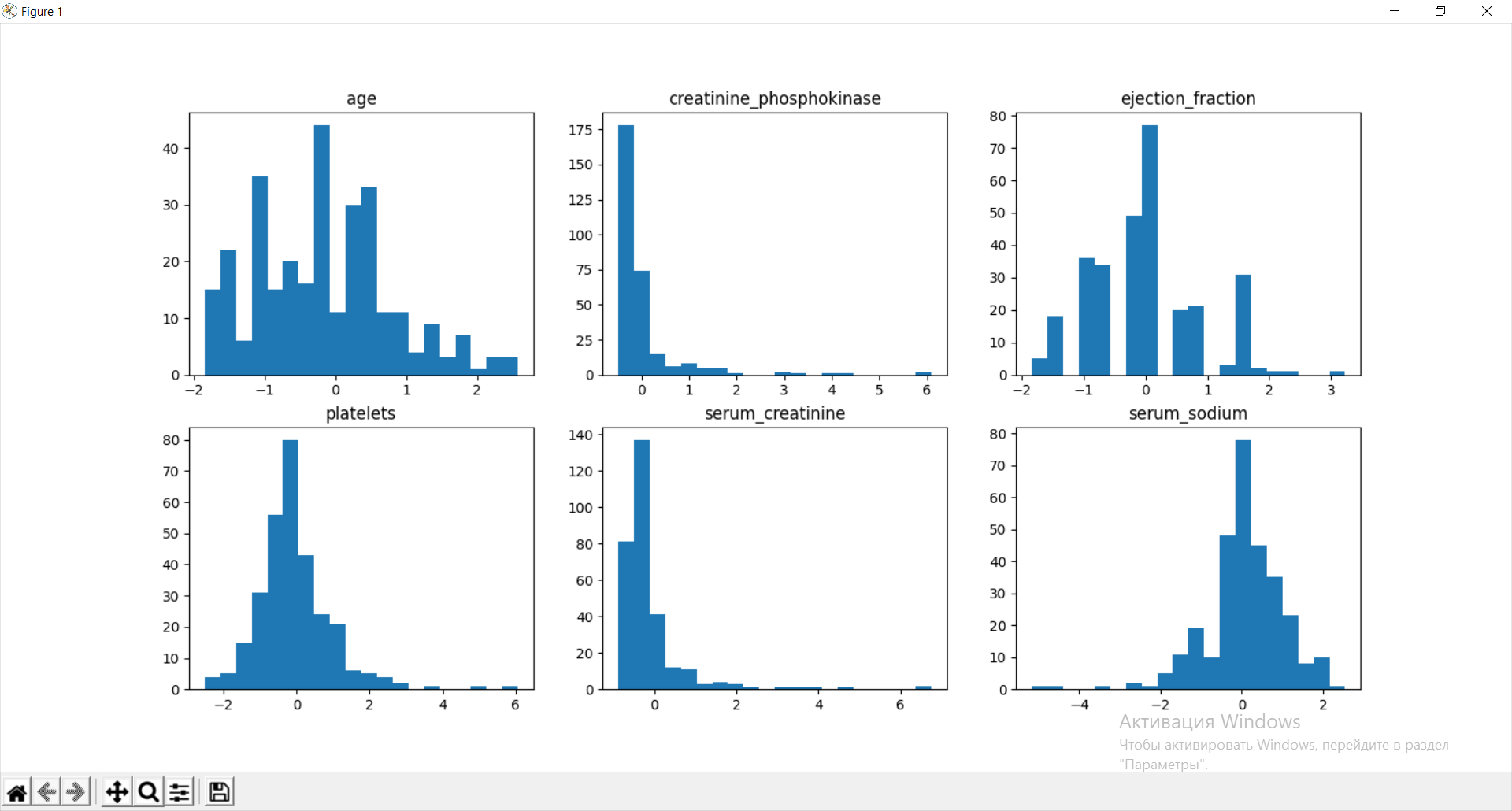


На основании гистограмм определила диапазоны значений для каждого из признаков, а также возле какого значения лежит наибольшее количество наблюдений. Приближения:





# **Стандартизация данных:**



5. Расcчитала мат. ожидание и СКО до и после стандартизации. На основании этих значений выведила для каждого признака формулы по которым они стандартизировались.

6. Сравнила значения из формул с полями mean\_ и var\_ объекта scaler

**До стандартизации:**

print(np.mean(data, axis=0))

[6.08338930e+01 5.81839465e+02 3.80836120e+01 2.63358029e+05

1.39387960e+00 1.36625418e+02]

print(np.var(data, axis=0))

[1.41013284e+02 9.38309881e+05 1.39595016e+02 9.53367655e+09

1.06663177e+00 1.94048389e+01]

**До стандартизации Scaler (получилось так же):**

print(scaler.mean\_)

[6.08338930e+01 5.81839465e+02 3.80836120e+01 2.63358029e+05

1.39387960e+00 1.36625418e+02]

print(scaler.var\_)

[1.41013284e+02 9.38309881e+05 1.39595016e+02 9.53367655e+09

1.06663177e+00 1.94048389e+01]

**После стандартизации:**

print(np.mean(data\_scaled, axis=0))

[ 5.70335306e-16 0.00000000e+00 -3.26754603e-17 7.72329061e-17

1.42583827e-16 -8.67384945e-16] Есть нулевое значение

print(np.var(data\_scaled, axis=0))

[1. 1. 1. 1. 1. 1.]

Формула:

**После стандартизации 150:**

print(np.mean(data\_scaled[:150,:], axis=0))

[ 1.30266168e-16 2.36847579e-17 1.18423789e-16 8.88178420e-17

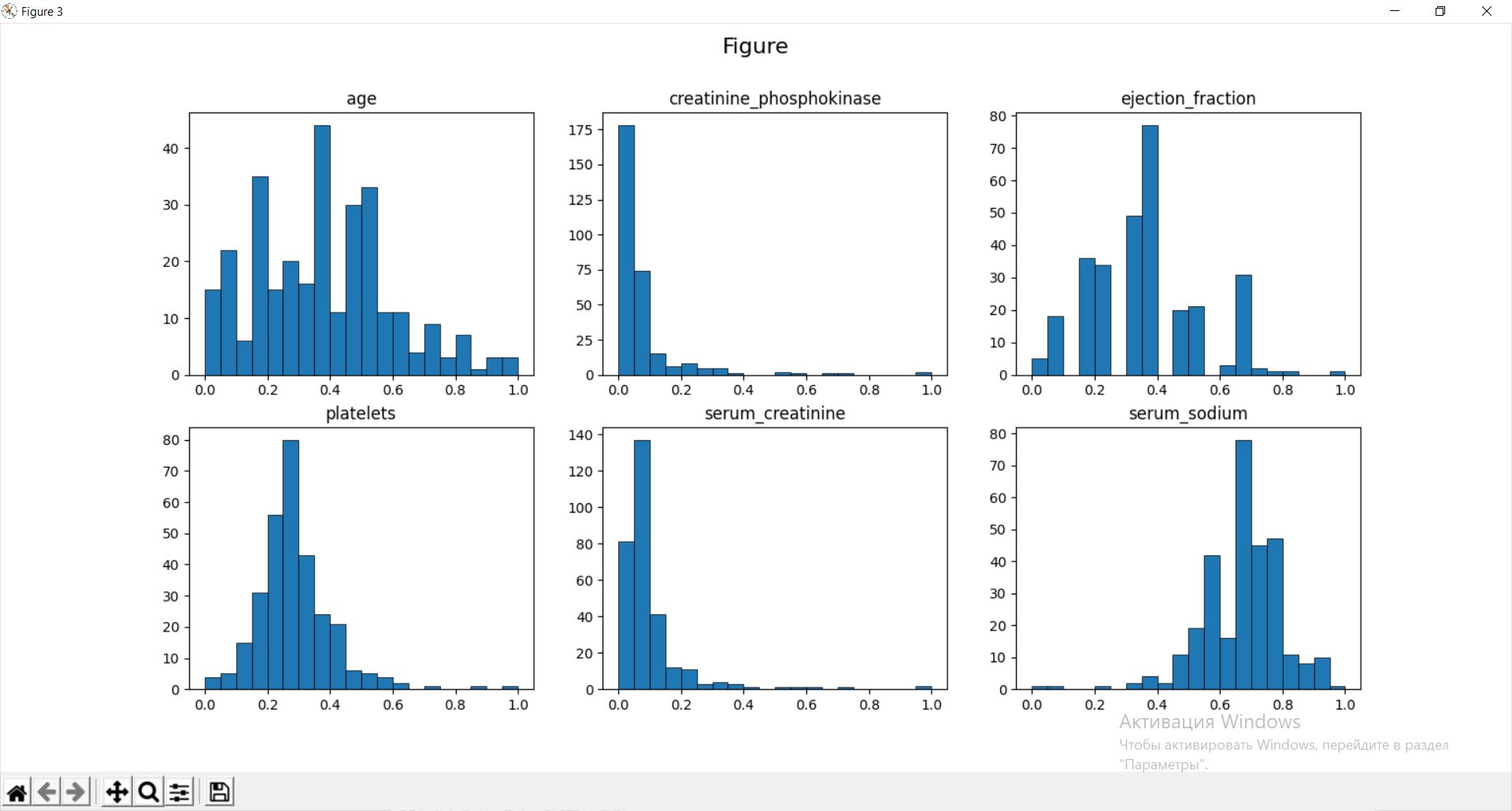
-1.30266168e-16 -3.06717614e-15]

print(np.var(data\_scaled[:150,:], axis=0))

[1. 1. 1. 1. 1. 1.]

**Приведение к диапазону**

1. Привела данные к диапазону используя MinMaxScaler
2. Построила гистограммы для признаков и сравнила с исходными данными



Диапазон стал [0..1]

1. Через параметры MinMaxScaler определила минимальное и максимальное значение данных для каждого признака

[4.00e+01 2.30e+01 1.40e+01 2.51e+04 5.00e-01 1.13e+02]

[9.500e+01 7.861e+03 8.000e+01 8.500e+05 9.400e+00 1.480e+02]

4. Аналогично трансформировала данные используя MaxAbsScaler и RobustScaler. Построила гистограммы. Определила к какому диапазону приводятся данные.

max\_abs\_scaler = preprocessing.MaxAbsScaler().fit(data)

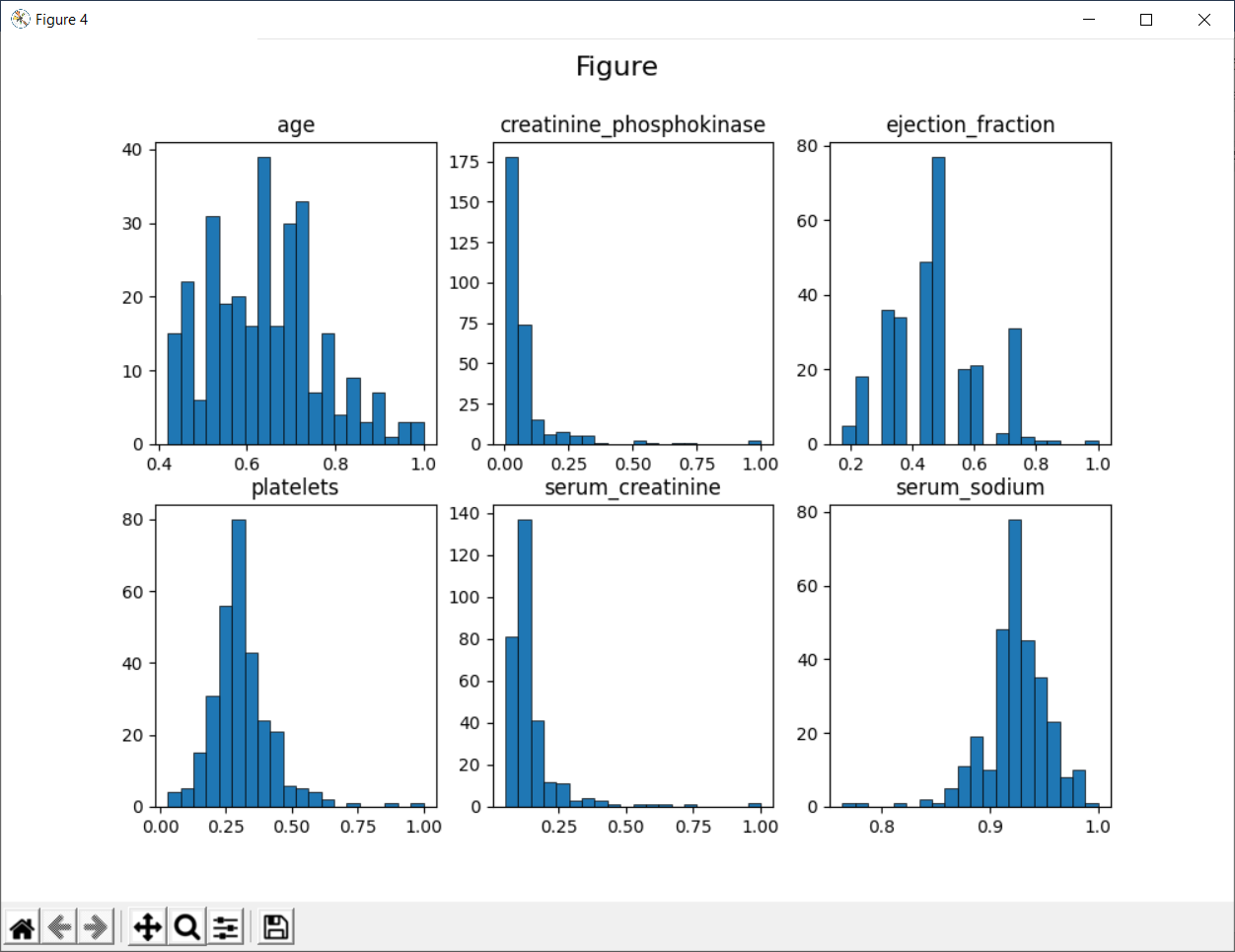
data\_max\_abs\_scaler = max\_abs\_scaler.transform(data)

robust\_scaler = preprocessing.RobustScaler().fit(data)

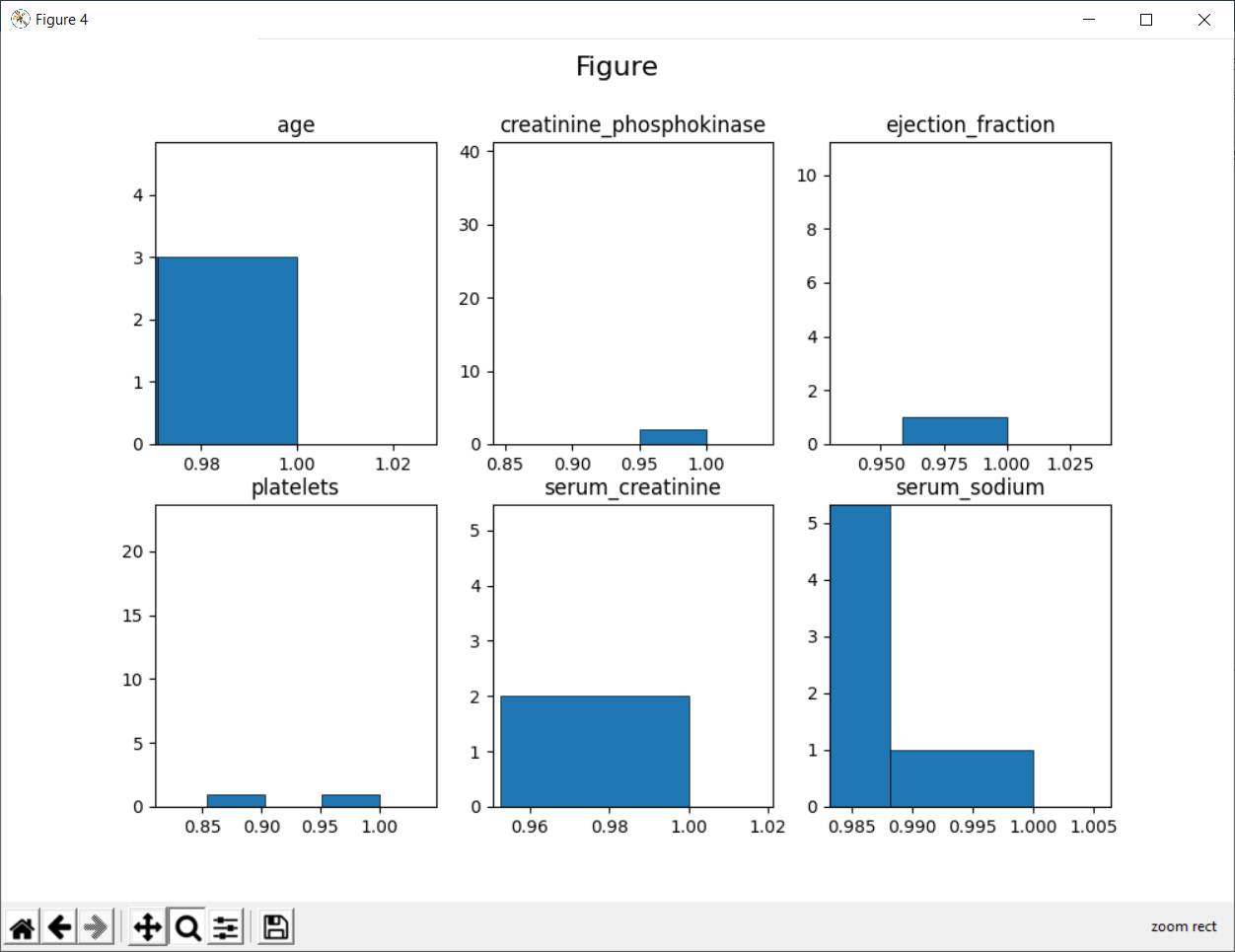
data\_robust\_scaler = robust\_scaler.transform(data)

plot\_hists(data\_max\_abs\_scaler)

plot\_hists(data\_robust\_scaler)

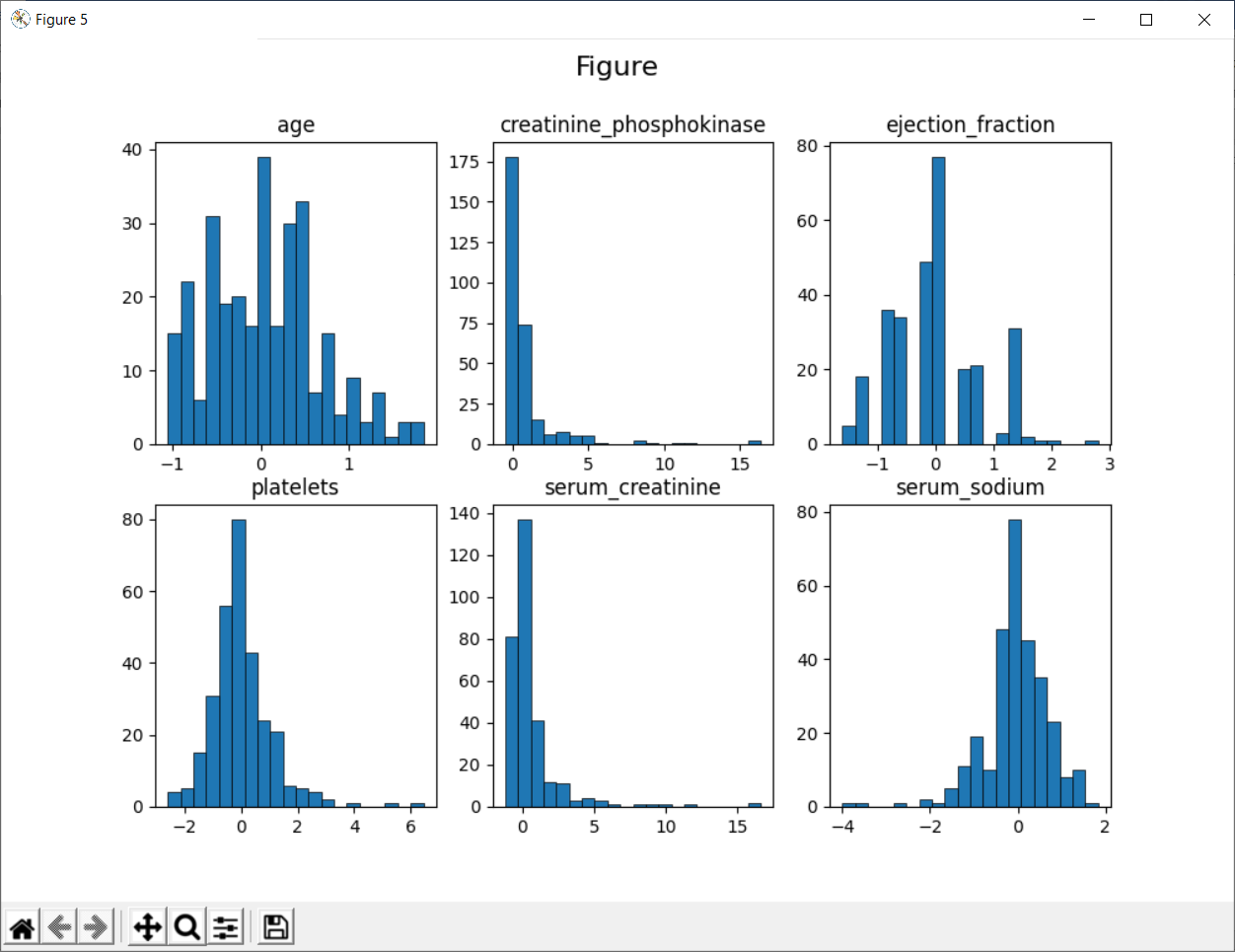


MaxAbsScaler



MaxAbsScaler

По гистограммам видно, что он приводит верхнюю границу диапазона к 1



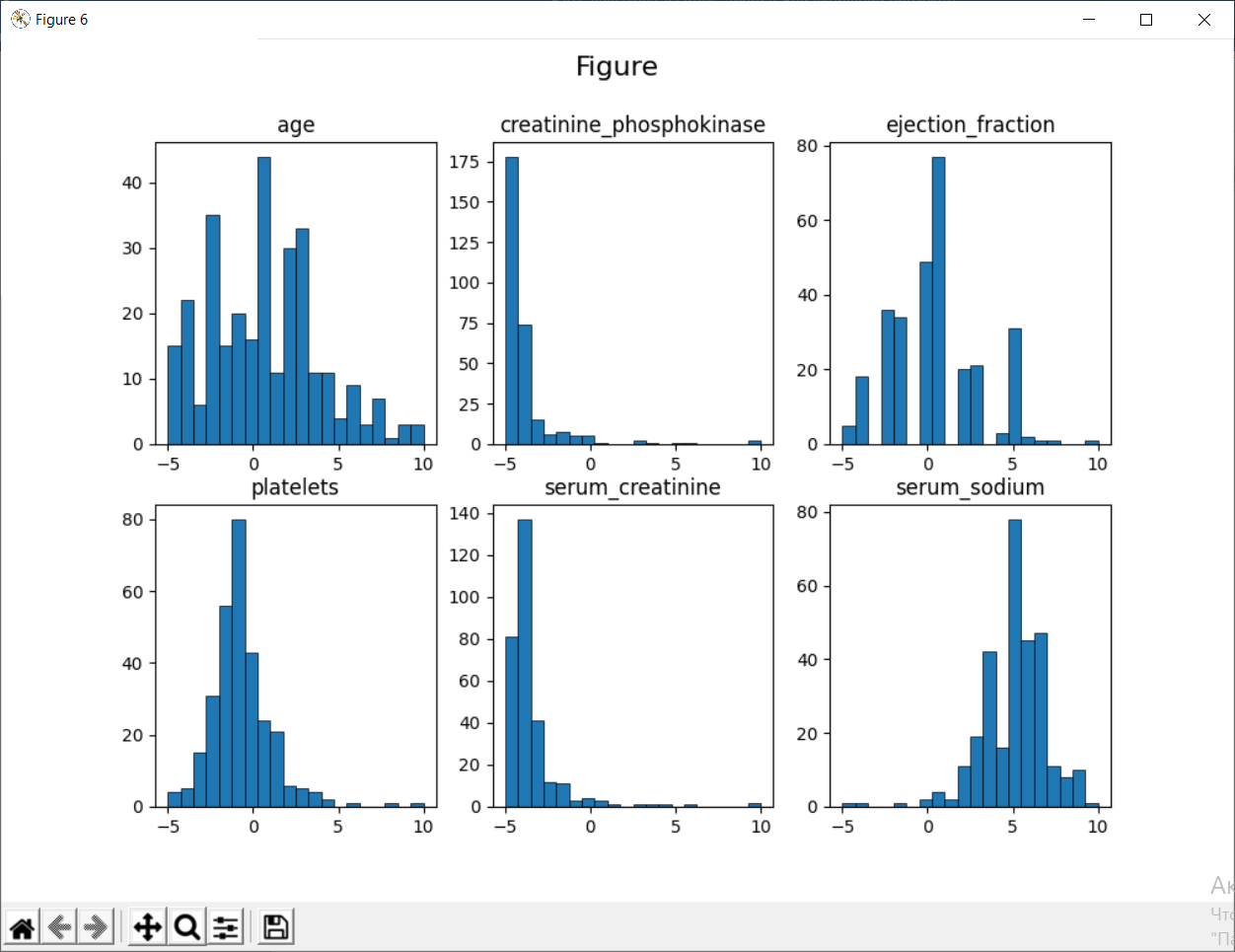
RobustScaler

Приводит среднее и медиану к 0, а среднее отклонение – к 1.

1. Напишите функцию, которая приводит все данные к диапазону [-5 10]

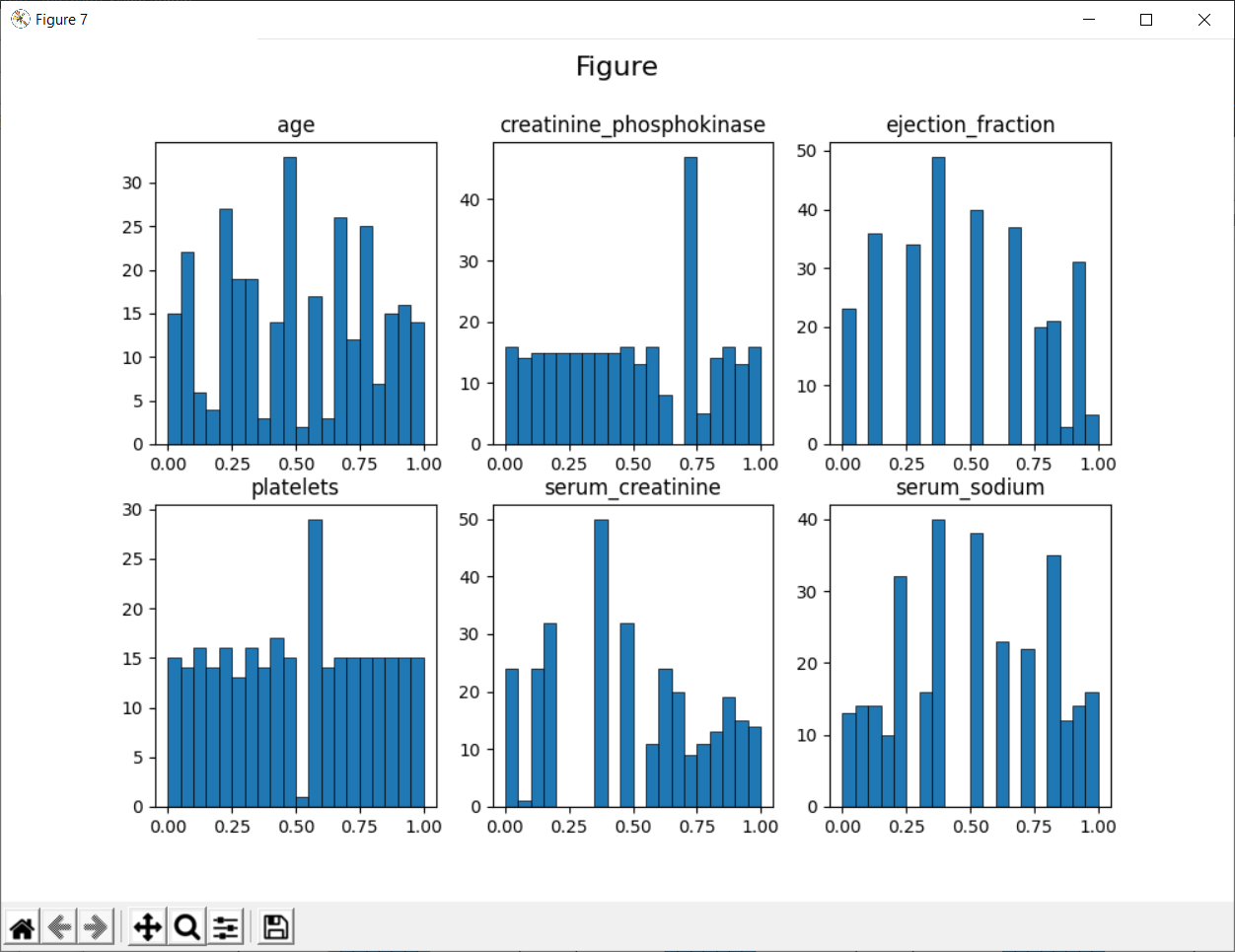
range\_scaler = preprocessing.MinMaxScaler().fit(data)

data\_range\_scaler = range\_scaler.transform(data)\*15-5



**Нелинейные преобразования**

1. Приведите данные к равномерному распределению используя QuantileTransformer
2. Постройте гистограммы и сравните с исходными данными



1. Определите, как и на что влияет значение параметра n\_quantiles

Влияет на частоту дискретизации -> приближение к равномерному распределению.

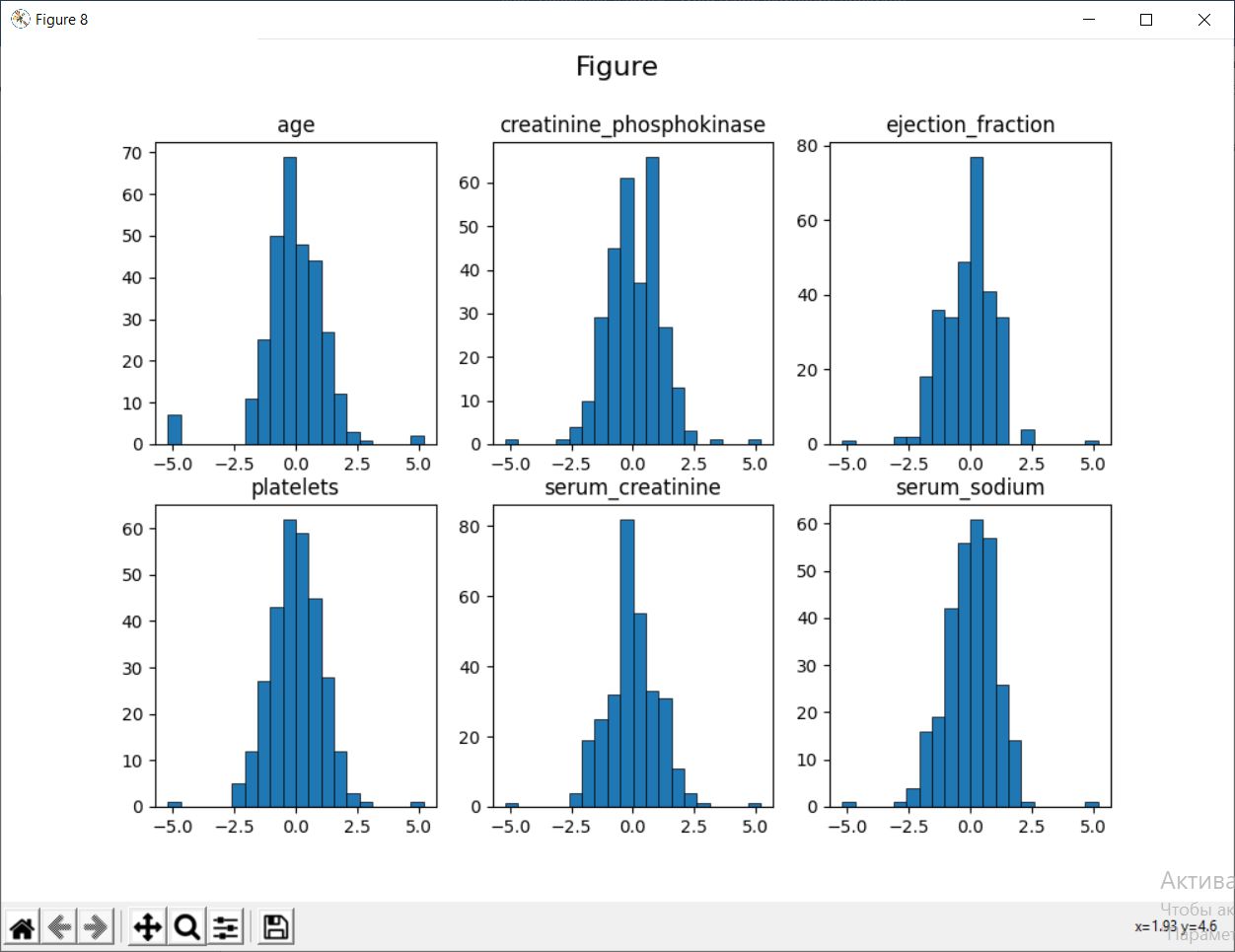
1. Приведите данные к нормальному распределению передав в QuantileTransformer параметр output\_distribution=‘normal’

quantile\_transformer = preprocessing.QuantileTransformer(n\_quantiles = 100,

random\_state=0, output\_distribution='normal').fit(data)

data\_quantile\_scaled2 = quantile\_transformer.transform(data)

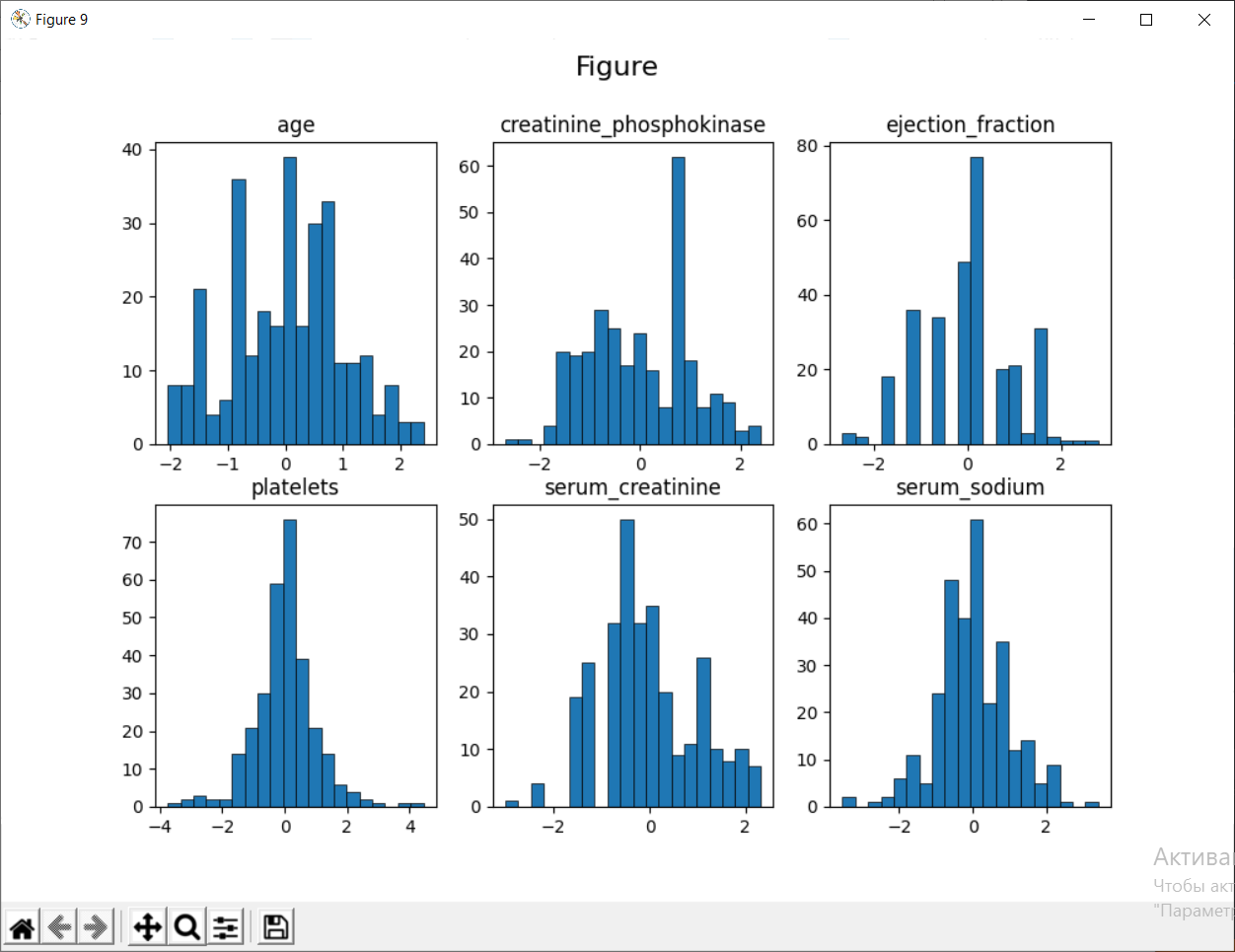
1. Постройте гистограммы и сравните с исходными данными



1. Самостоятельно приведите данные к нормальному распределению используя PowerTransformer

power\_scaled = preprocessing.PowerTransformer().fit(data)

data\_power\_scaled = power\_scaled.transform(data)



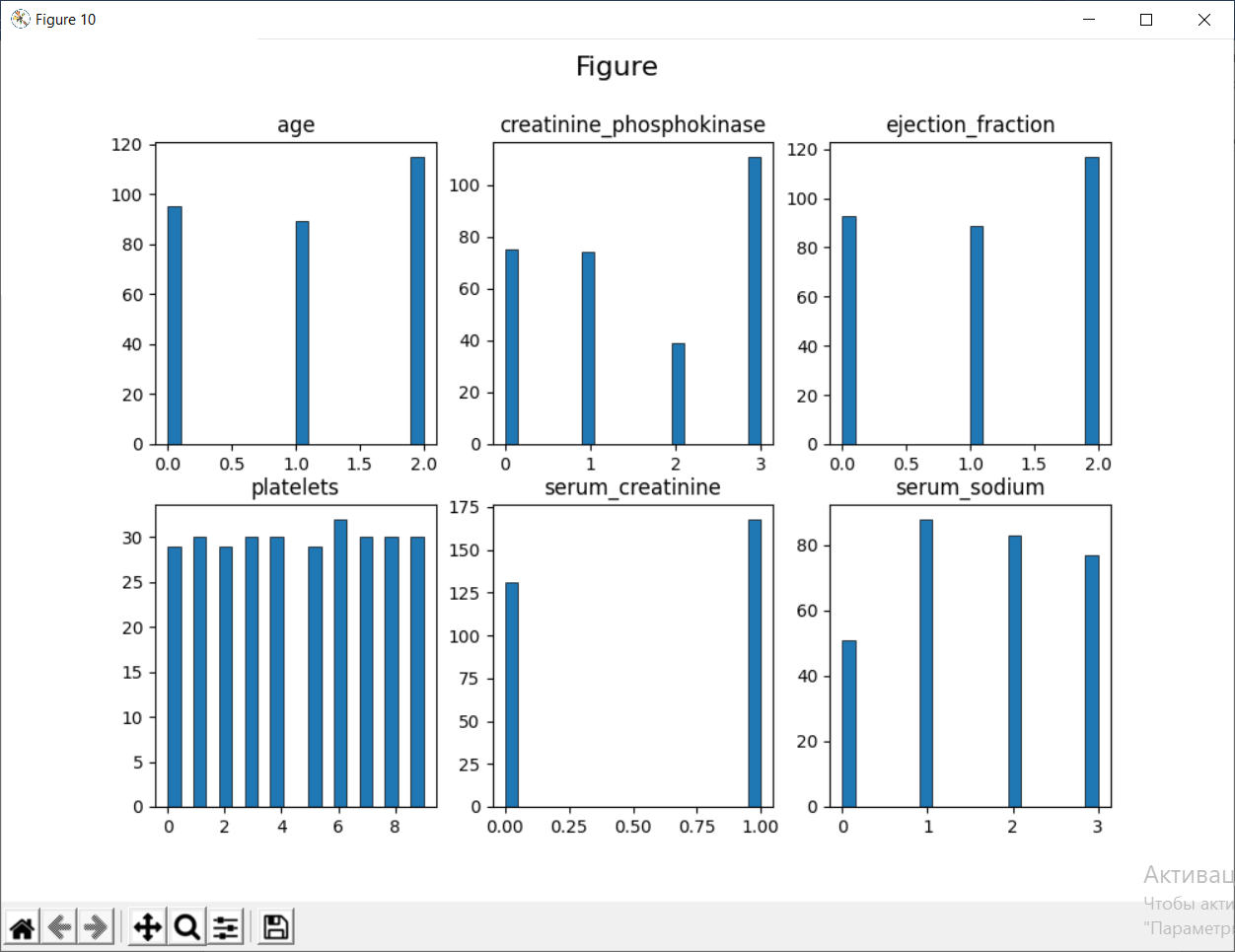
**Дискретизация признаков**

1. Проведите дискретизацию признаков, используя KBinsDiscretizer

2. Постройте гистограммы. Объясните полученные результаты

discret = preprocessing.KBinsDiscretizer(n\_bins=[3, 4, 3, 10, 2, 4], encode='ordinal').fit(data)

data\_discret = discret.transform(data)



3. Через параметр bin\_edges\_ выведите диапазоны каждого интервала для каждого признака

print(discret.bin\_edges\_)

[array([40., 55., 65., 95.])

array([ 23. , 116.5, 250. , 582. , 7861. ])

array([14., 35., 40., 80.])

array([ 25100., 153000., 196000., 221000., 237000., 262000., 265000.,

285200., 319800., 374600., 850000.])

array([0.5, 1.1, 9.4]) array([113., 134., 137., 140., 148.])]

# **Вывод**

Ознакомилась с методами предобработки данных из библиотеки Scikit Learn, а именно стандартизацией, приведением к диапазону, нелинейными преобразованиями и дискретизацией данных.