МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Машинное обучение»

Тема: Предобработка данных.

Студент гр. 6302 _____ Барбарич И.Г.

Руководитель _____ Жангиров Т. Р.

Санкт-Петербург
2020

Цель работы

Ознакомиться с методами предобработки данных из библиотеки Scikit Learn

1. **Загрузить датасет по ссылке**

Рисунок 1. Загруженный датасет.

2. Загрузить датасет в датафрейм, и исключить бинарные признаки и признак времени.

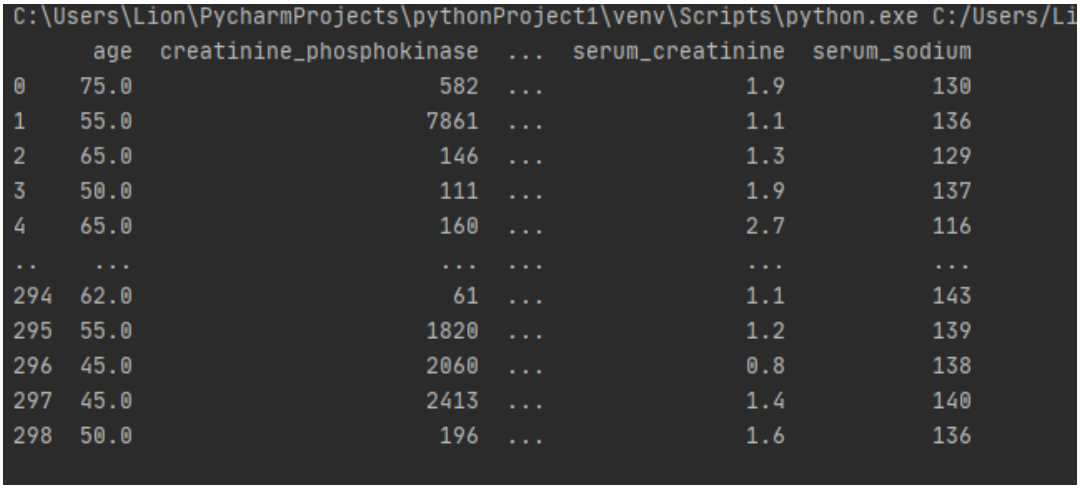

Рисунок 2. Загруженный датасет в датафрейм.

3. Построить гистограмму признаков

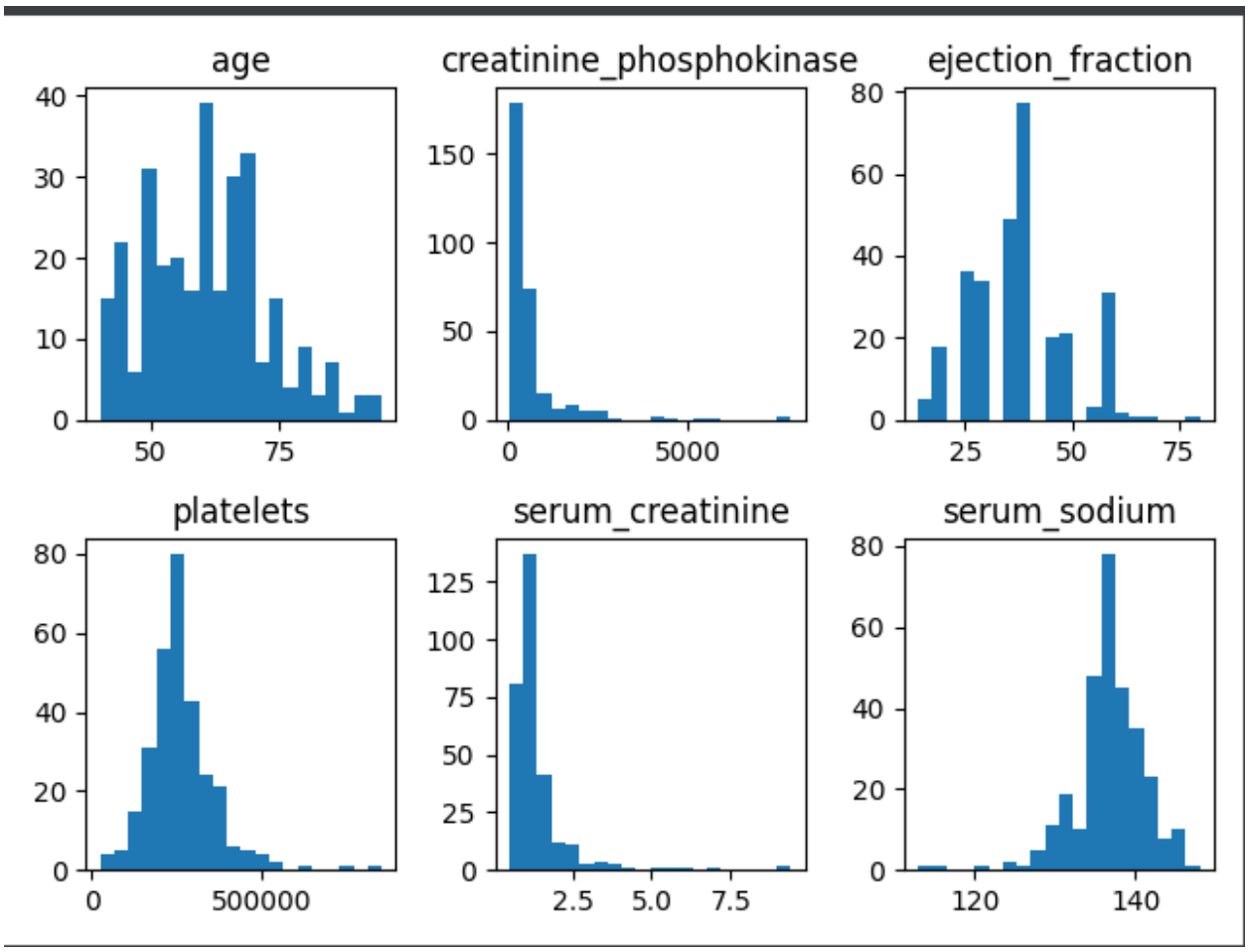

Рисунок 3. Гистограмма признаков.

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

Код программы:

```
n_bins = 20

fig, axs = plt.subplots(2,3)

axs[0, 0].hist(df['age'].values, bins = n_bins)
axs[0, 0].set_title('age')
print('min {}'.format(df['age'].values.min()),
    'max {}'.format(df['age'].values.max()))

axs[0, 1].hist(df['creatinine_phosphokinase'].values, bins = n_bins)
axs[0, 1].set_title('creatinine_phosphokinase')
print('min {}'.format(df['creatinine_phosphokinase'].values.min()),
    'max {}'.format(df['creatinine_phosphokinase'].values.max()))
```

```
axs[0, 2].hist(df['ejection_fraction'].values, bins = n_bins)
axs[0, 2].set_title('ejection_fraction')
print('min {}'.format(df['ejection_fraction'].values.min()),
    'max {}'.format(df['ejection_fraction'].values.max()))

axs[1, 0].hist(df['platelets'].values, bins = n_bins)
axs[1, 0].set_title('platelets')
print('min {}'.format(df['platelets'].values.min()),
    'max {}'.format(df['platelets'].values.max()))

axs[1, 1].hist(df['serum_creatinine'].values, bins = n_bins)
axs[1, 1].set_title('serum_creatinine')
print('min {}'.format(df['serum_creatinine'].values.min()),
    'max {}'.format(df['serum_creatinine'].values.max()))

axs[1, 2].hist(df['serum_sodium'].values, bins = n_bins)
axs[1, 2].set_title('serum_sodium')
print('min {}'.format(df['serum_sodium'].values.min()),
    'max {}'.format(df['serum_sodium'].values.max()))

plt.show()
```

Результат:

min 40.0 max 95.0

min 23 max 7861

min 14 max 80

min 25100.0 max 850000.0

min 0.5 max 9.4

min 113 max 148

Таблица 1

Признак	**диапазоны**	**Наибольшее кол-во наблюдений**
age	40 – 95	60
creatinine_phosphokinase	23 – 7861	50
ejection_fraction	14 – 80	35
platelets	25100 – 850 000	250000

serum_creatinine	0.5 – 9.4	1,25
serum_sodium	113 - 148	136

5. Преобразовал *датафрейм* к двумерному массиву *NumPy*, где строка соответствует наблюдению, а столбец признаку.

Стандартизация данных.

1. Была настроена стандартизация данных на основе 150 наблюдений, используя StandartScaler и построена гистограмма на стандартизированных данных.

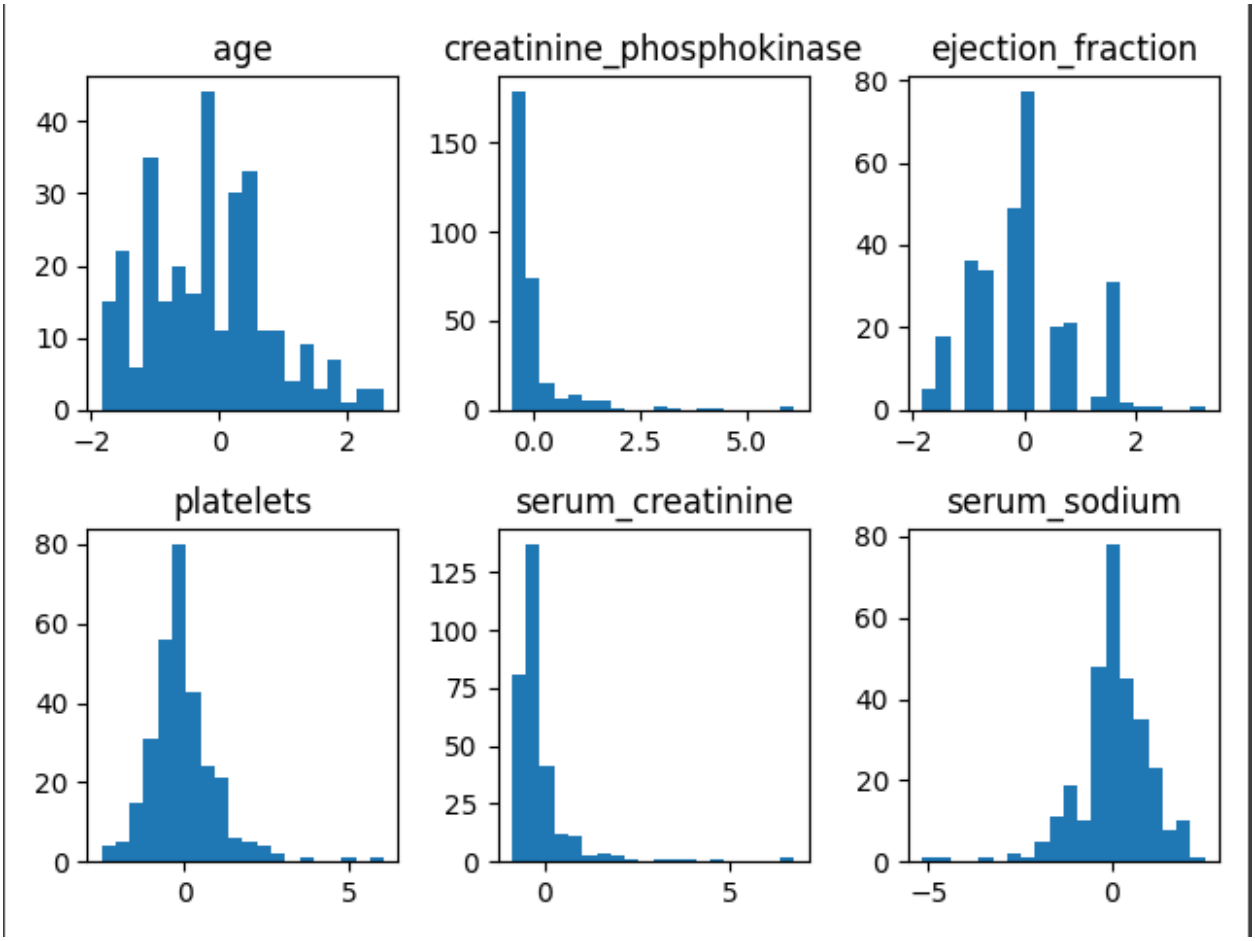

Рисунок 4. Гистограмма данных первых 150 наблюдений, используя StandartScaler.

2. Сравнение данных до и после стандартизации.
   - Объекты после стандартизации центрированы вокруг 0.
   - Масштаб изменился до единичной дисперсии.
3. Рассчитайте мат. ожидание и СКО до и после стандартизации.

Код программы:

```
print('до стандартизации')
print (np.mean(data[:150, :], axis=0))
print (np.std(data[:150, :], axis=0))
…
print('после стандартизации')
print (np.mean(data[:150, :], axis=0))
print (np.std(data[:150, :], axis=0))
```

Результат:

до стандартизации

мат ожидание: [6.29466667e+01 6.07153333e+02 3.79466667e+01 2.66746749e+05 1.52060000e+00 1.36453333e+02]

СКО: [1.24497854e+01 1.18974318e+03 1.30393183e+01 9.61917902e+04 1.16641630e+00 4.53958393e+00]

после стандартизации

мат ожидание: [-0.16970362 -0.02127675 0.01050249 -0.03522879 -0.1086408 0.0379076]

СКО: [0.95382379 0.81417905 0.90610822 1.01506113 0.88542887 0.9703736]

4. На основе значений можно вывести формулу

$$Y = \frac{X - mean(X)}{std(X)};$$

5. Сравните значений из формул с полями mean_ и var_ объекта scaler

Код программы:

```
print ('mean scaler', scaler.mean_)
print ('var scaler', scaler.var_)
```

Результат:

mean scaler [6.29466667e+01 6.07153333e+02 3.79466667e+01 2.66746749e+05 1.52060000e+00 1.36453333e+02]

var scaler [1.54997156e+02 1.41548882e+06 1.70023822e+02 9.25286050e+09
1.36052697e+00 2.06078222e+01]

Mean_ scaler соответствует мат ожиданию

6. Проведите настройку стандартизации на всех данных и сравните с результатами настройки на основании 150 наблюдений.

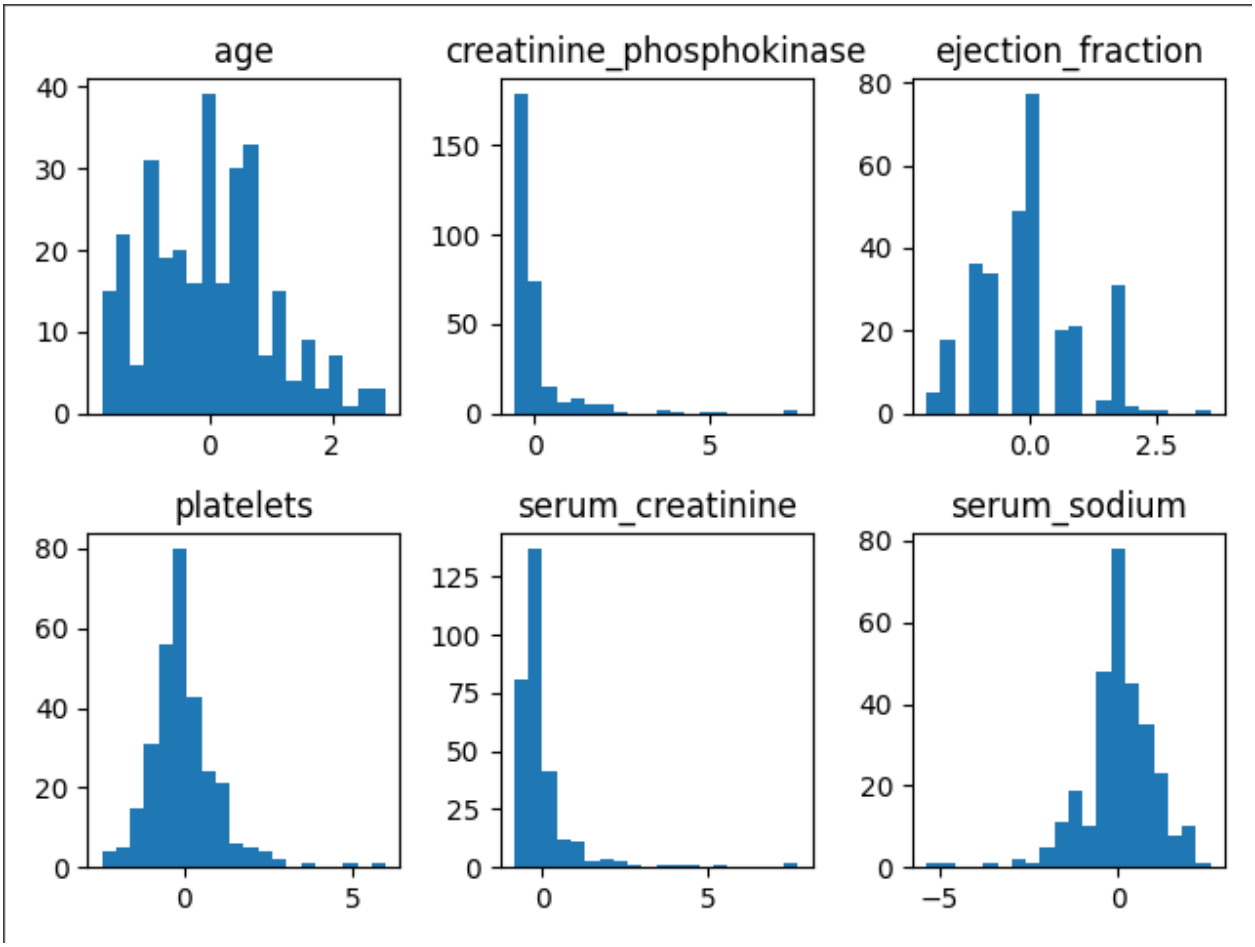

Рисунок 5. Гистограмма данных всех наблюдений, используя StandartScaler.

мат ожидание: [5.70335306e-16 0.00000000e+00 -3.26754603e-17
7.72329061e-17
 1.42583827e-16 -8.67384945e-16]

СКО: [1. 1. 1. 1. 1. 1.]

Приведение к диапазону

1. Приведите данные к диапазону, используя MinMaxScaler.

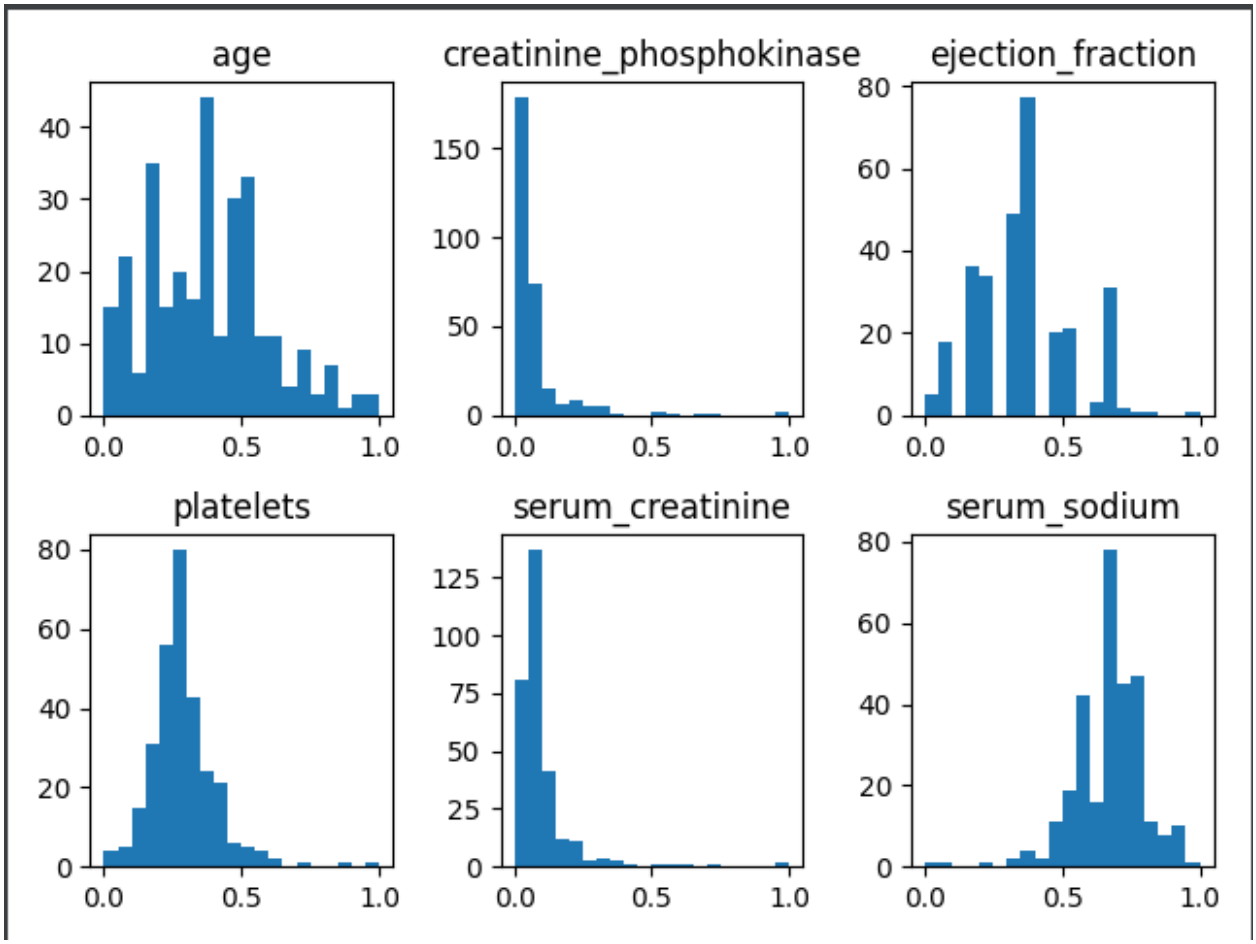

Рисунок 6. Приведение данные к диапазону, используя MinMaxScaler.

2. Через параметры MinMaxScaler определите минимальное и максимальное значение в данных для каждого признака.

Код программы:

```
print ('min', min_max_scaler.data_min_)
print ('max', min_max_scaler.data_max_)
```

Результат:

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

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

3. Аналогично трансформируйте данные используя MaxAbsScaler и RobustScaler. Постройте гистограммы. Определите к какому диапазону приводятся данные.

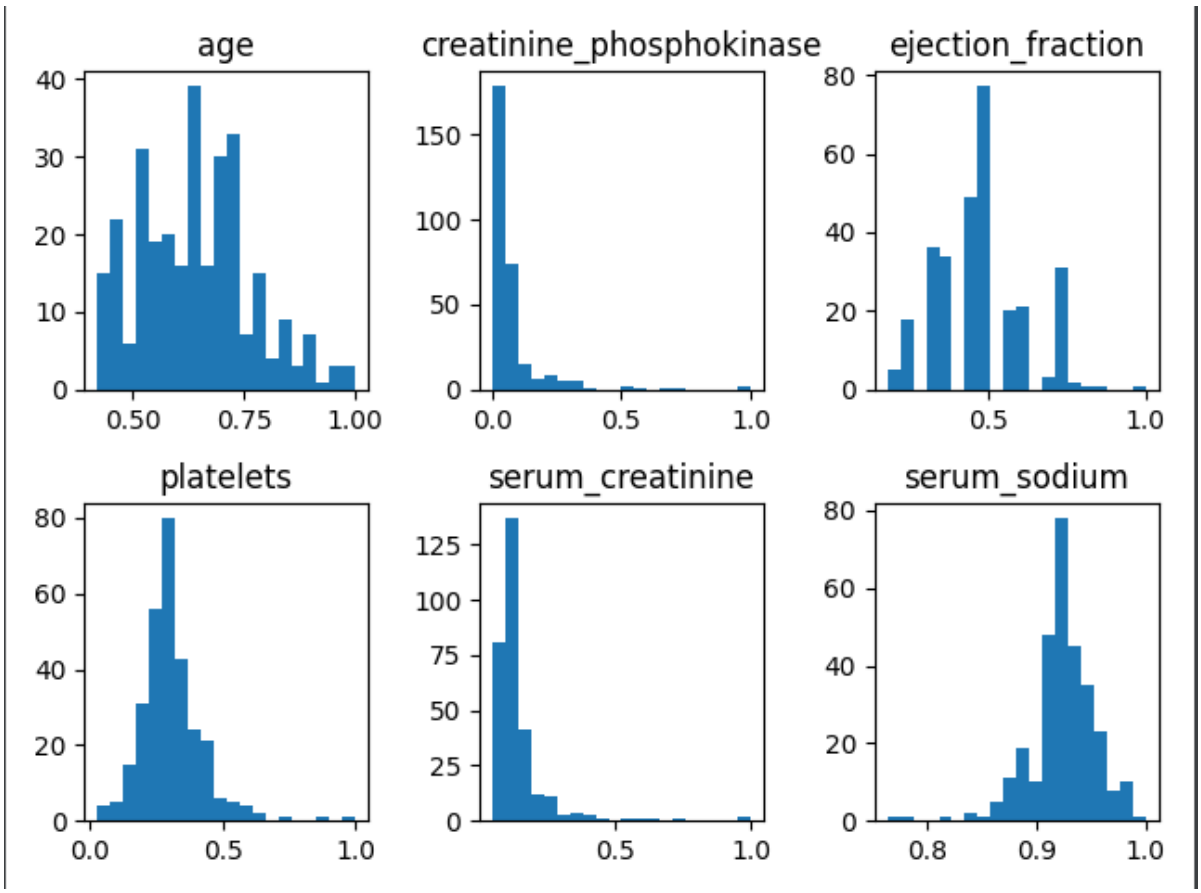

Рисунок 7. Приведение данные к диапазону, используя MaxAbsScaler.

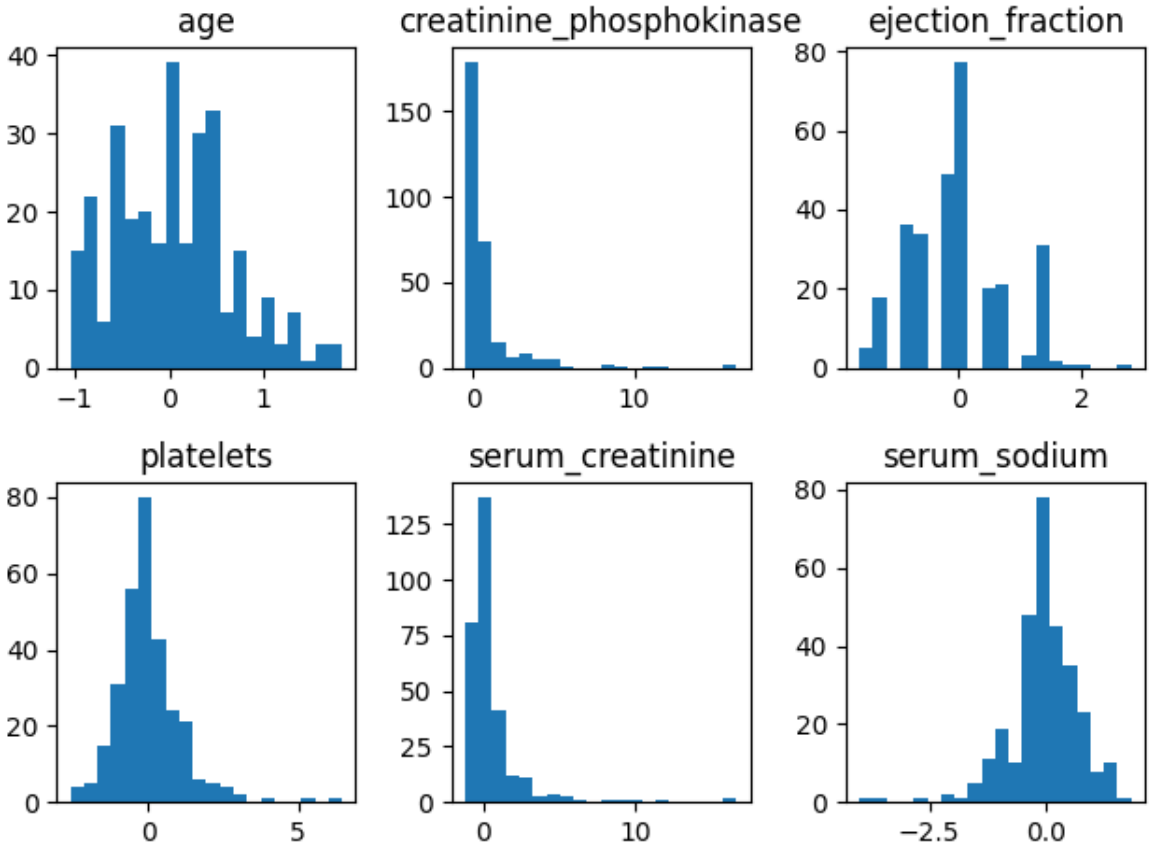

Рисунок 8. Приведение данные к диапазону, используя и RobustScaler.

MaxAbsScaler масштабирует так, что максимальное значение равно 1, а RobustScaler центрирует относительно 0.

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

Трансформируется по следующей формуле.

X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))

X_scaled = X_std * (max - min) + min

В результате, получаем:

$$X\_scaled = 15 * \frac{(X - X.min(\ \ ))}{X.max(\ \ ) - X.min(\ \ )} - 5;$$

Код программы:

```
min_max_scaler = preprocessing.MinMaxScaler().fit(data)
data_min_max_scaled = min_max_scaler.transform(data)*15-5
```

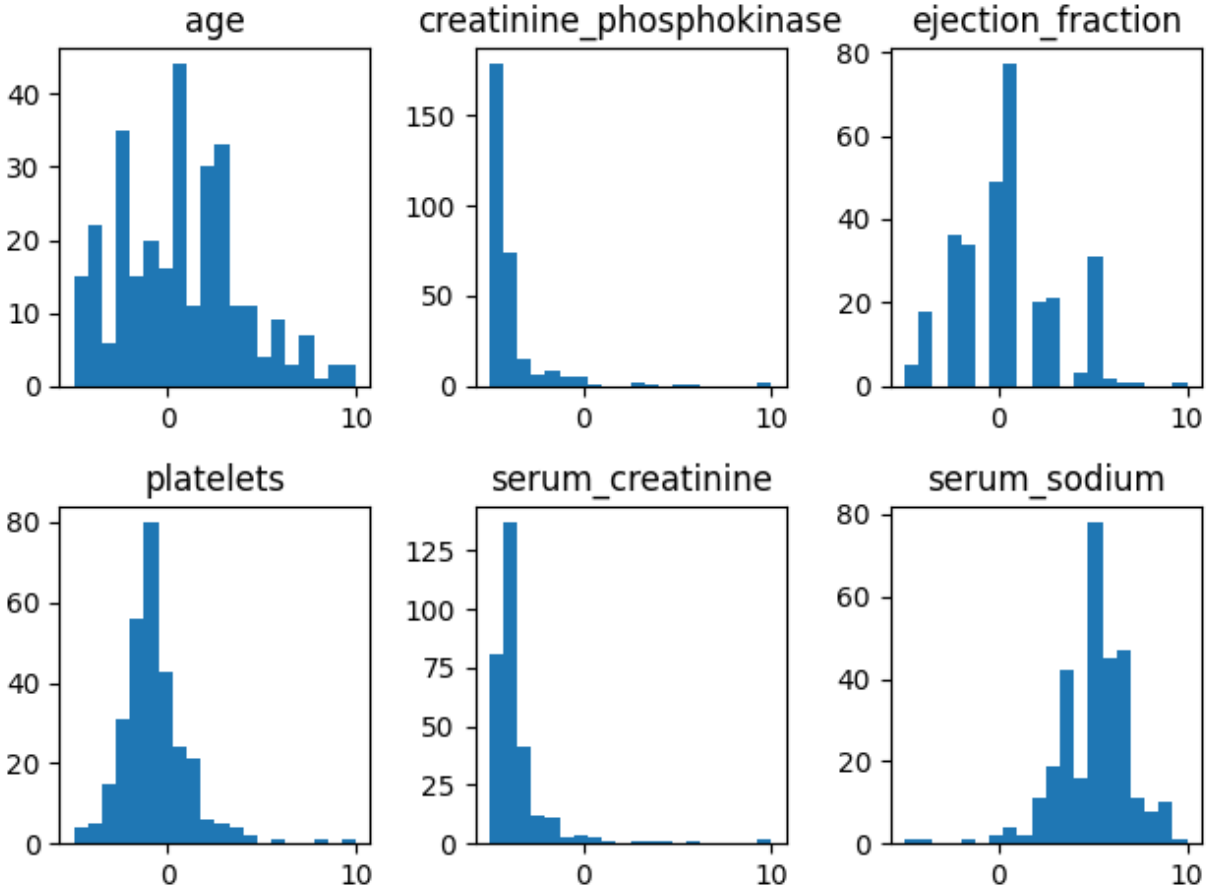

Рисунок 9. Приведение данные к диапазону [-5 10].

Нелинейные преобразования

1. Приведение данных к равномерному распределению используя QuantileTransformer.

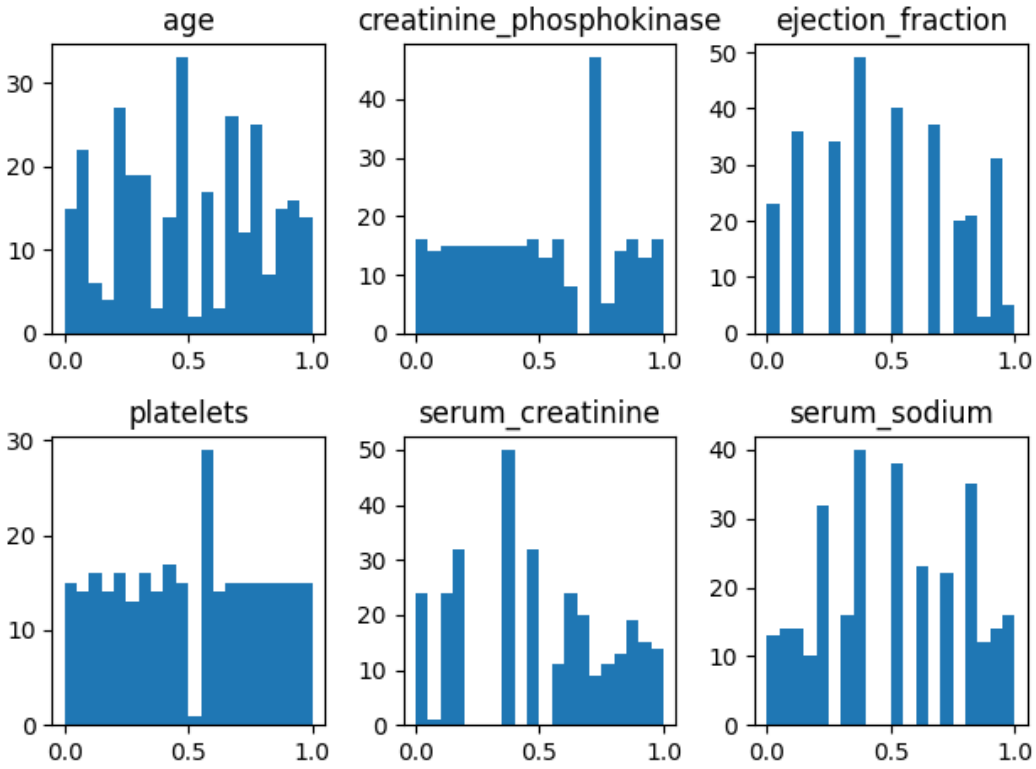

Рисунок 8. Приведение к равномерному распределению с помощью QuantileTransformer при 100 квантелях.

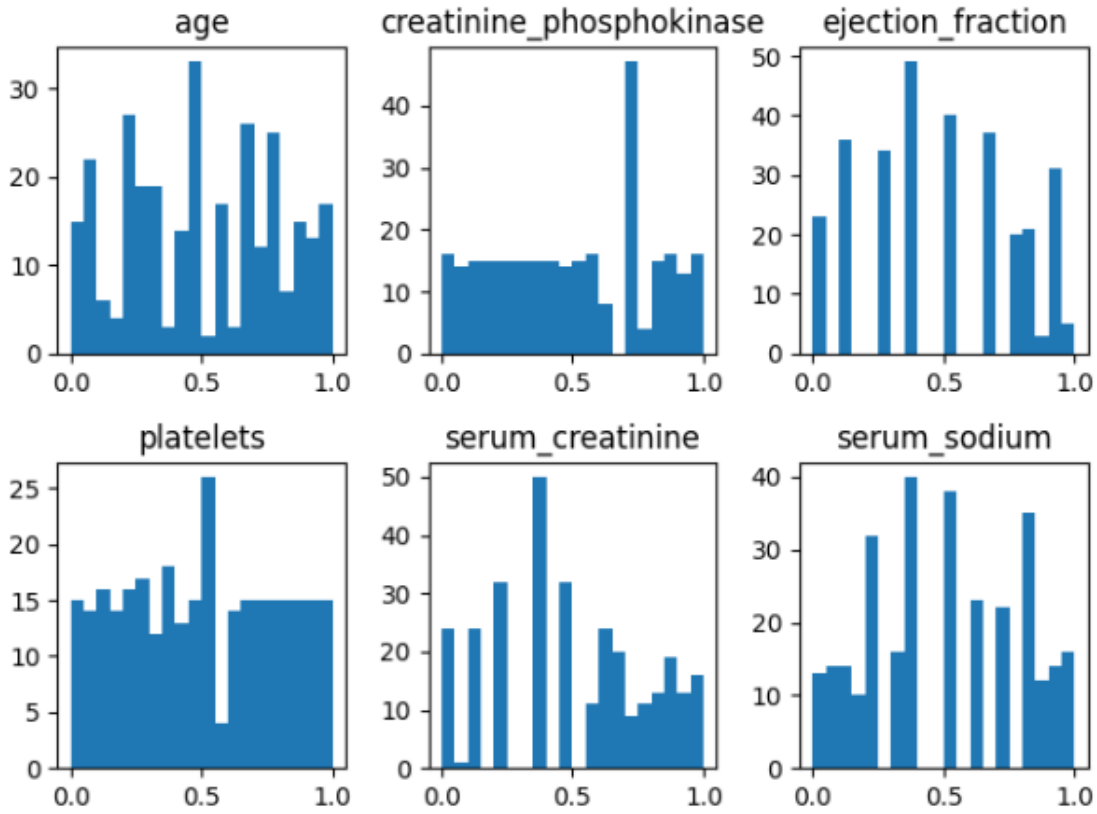

Рисунок 8. Приведение к равномерному распределению с помощью QuantileTransformer при 150 квантелях.

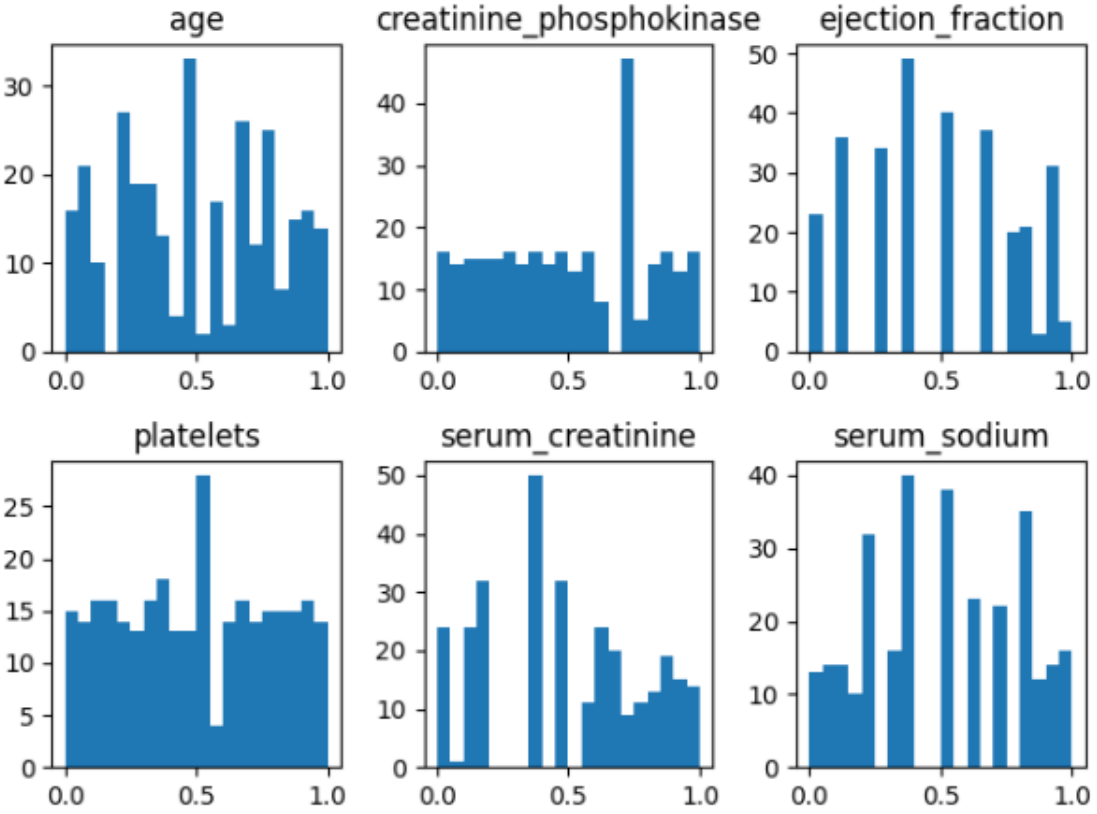

Рисунок 9. Приведение к равномерному распределению с помощью QuantileTransformer при 50 квантелях.

Чем больше квантелей, тем лучше приближение к требуемому распределению.

2. Приведите данные к нормальному распределению передав в QuantileTransformer параметр output_distribution=‘normal

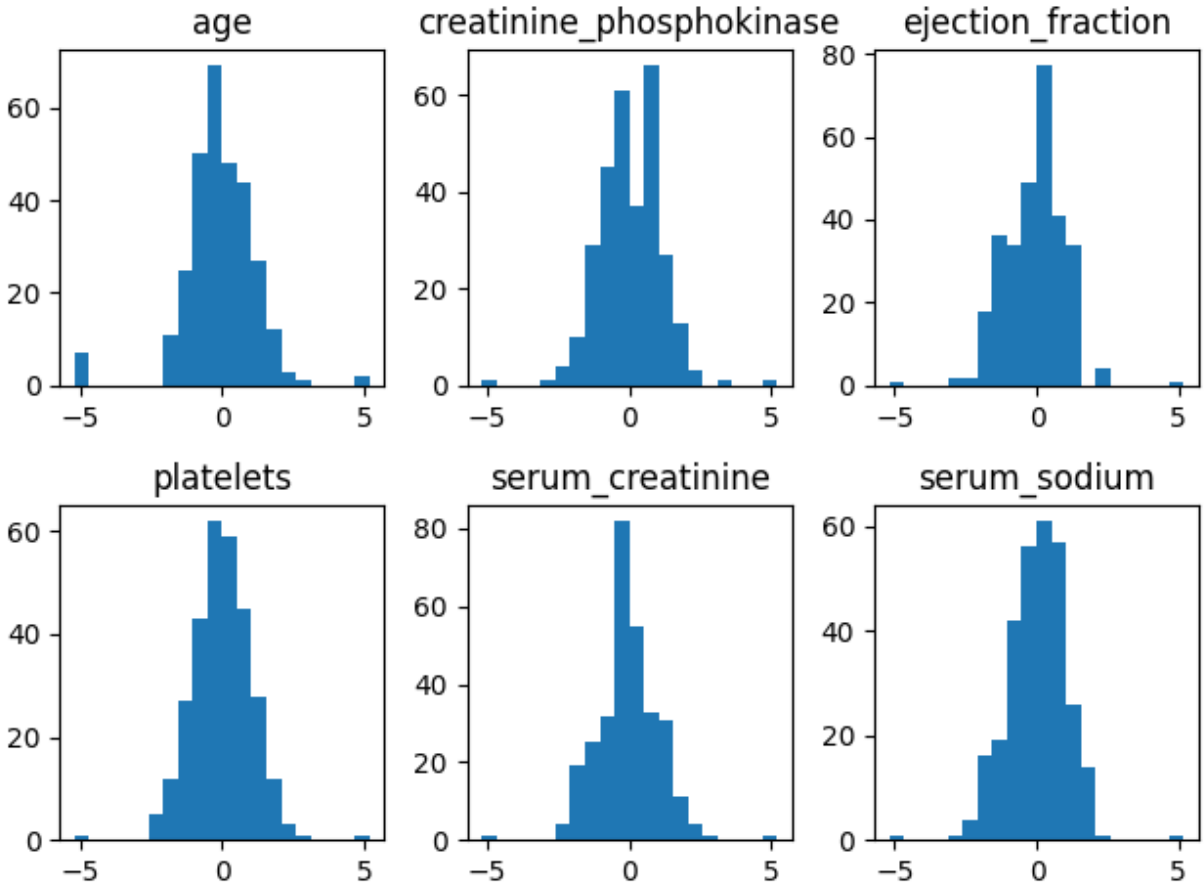

Рисунок 10. Приведение к нормальному распределению с использованием параметра output_distribution=’normal’.

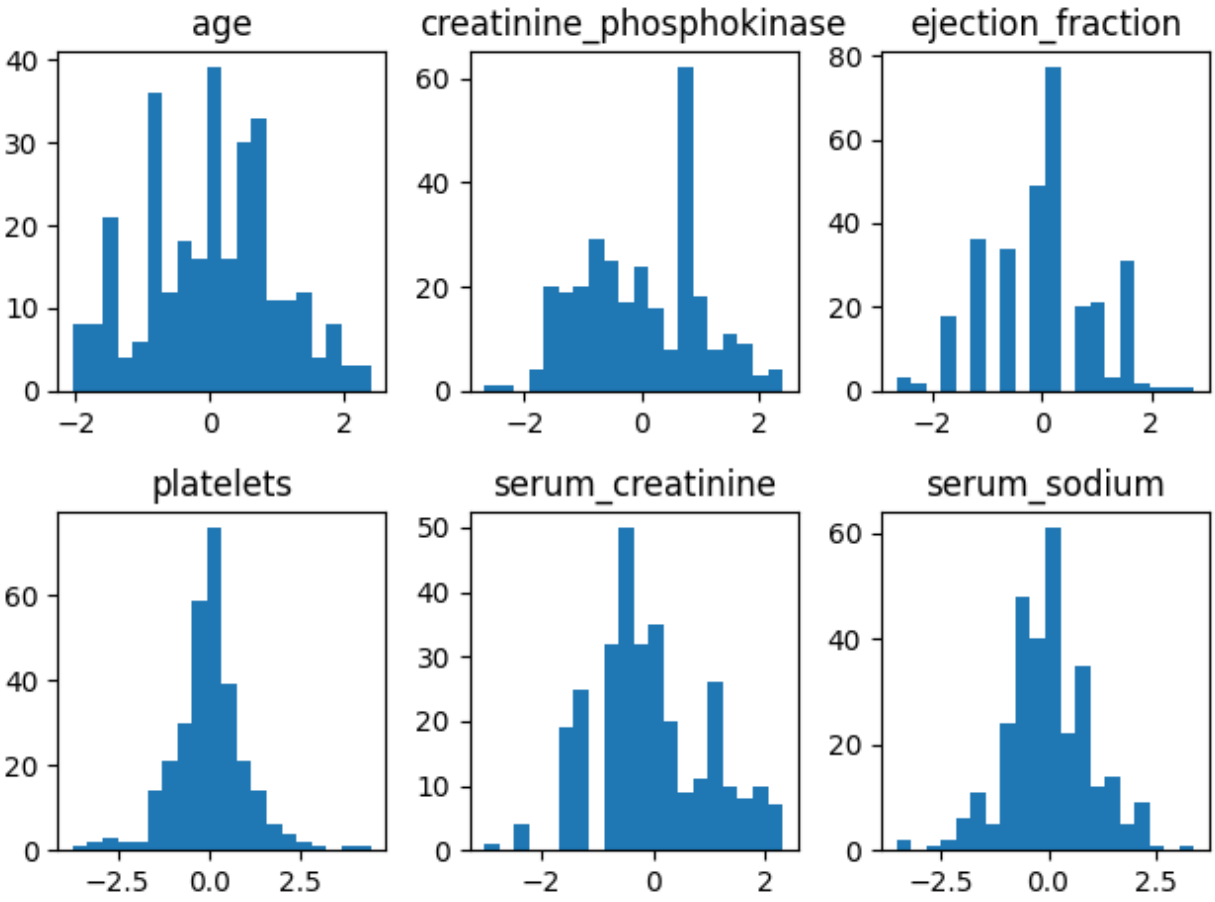

Рисунок 11. Приведение данных к нормальному распределению используя PowerTransformer

Дискретизация признаков

1. Проведите дискретизацию признаков, используя KBinsDiscretizer, на следующее количество диапазонов:
   - age - 3
   - creatinine_phosphokinase - 4
   - ejection_fraction - 3
   - platelets – 10
   - serum_creatinine - 2 s
   - erum_sodium - 4 2

2. Постройте гистограммы. Объясните полученные результаты

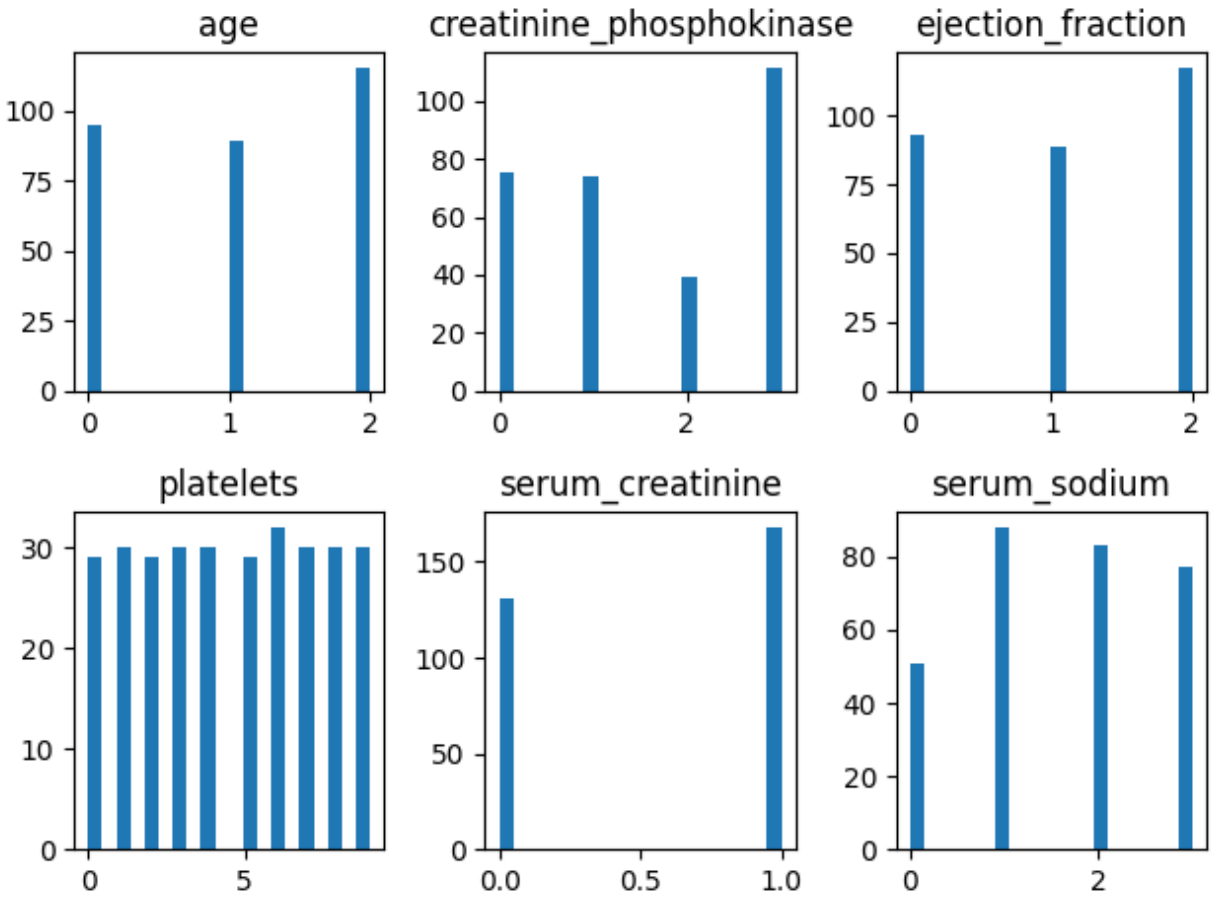

Рисунок 12. Приведение дискретизацию признаков, используя KBinsDiscretizer.

3. Через параметр bin_edges_ выведите диапазоны каждого интервала для каждого признака

[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

В результате работы были получены навыки с различными методами предобработки данных.