Рубежный контроль №2

Воронова О. А. ИУ5-63Б

Вариант №8

Задание. Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2 (по варианту для Вашей группы). Оцените качество моделей на основе подходящих метрик качества (не менее двух метрик). Какие метрики качества Вы использовали и почему? Какие выводы Вы можете сделать о качестве построенных моделей? Для построения моделей необходимо выполнить требуемую предобработку данных: заполнение пропусков, кодирование категориальных признаков, и т.д.

Методы ИУ5-63 - Дерево решений и Случайный лес

Набор данных: https://www.kaggle.com/lava18/google-play-store-apps

Библиотеки

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

Загрузка и первичная обработка данных

```
data=pd.read_csv('googleplaystore.csv', sep=",")
# размер набора данных
data.shape
```

```
(10841, 13)
```

```
data.head()
```

```
                                                 App        Category
Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN
4.1
1                                Coloring book moana  ART_AND_DESIGN
3.9
2  U Launcher Lite – FREE Live Cool Themes, Hide ...  ART_AND_DESIGN
4.7
3                            Sketch - Draw & Paint  ART_AND_DESIGN
```

```
4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN
4.3

  Reviews  Size     Installs  Type Price Content Rating  \
0     159   19M      10,000+  Free     0       Everyone
1     967   14M     500,000+  Free     0       Everyone
2   87510  8.7M   5,000,000+  Free     0       Everyone
3  215644   25M  50,000,000+  Free     0           Teen
4     967  2.8M     100,000+  Free     0       Everyone

                      Genres      Last Updated         Current Ver  \
0               Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
2               Art & Design    August 1, 2018               1.2.4
3               Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up

data.head()

                                                 App        Category
Rating  \
0     Photo Editor & Candy Camera & Grid & ScrapBook  ART_AND_DESIGN
4.1
1                                Coloring book moana  ART_AND_DESIGN
3.9
2  U Launcher Lite – FREE Live Cool Themes, Hide ...  ART_AND_DESIGN
4.7
3                               Sketch - Draw & Paint  ART_AND_DESIGN
4.5
4              Pixel Draw - Number Art Coloring Book  ART_AND_DESIGN
4.3

  Reviews  Size     Installs  Type Price Content Rating  \
0     159   19M      10,000+  Free     0       Everyone
1     967   14M     500,000+  Free     0       Everyone
2   87510  8.7M   5,000,000+  Free     0       Everyone
3  215644   25M  50,000,000+  Free     0           Teen
4     967  2.8M     100,000+  Free     0       Everyone

                      Genres      Last Updated         Current Ver  \
0               Art & Design   January 7, 2018               1.0.0
1  Art & Design;Pretend Play  January 15, 2018               2.0.0
```

```
2               Art & Design    August 1, 2018               1.2.4
3               Art & Design      June 8, 2018  Varies with device
4    Art & Design;Creativity     June 20, 2018                 1.1

    Android Ver
0  4.0.3 and up
1  4.0.3 and up
2  4.0.3 and up
3    4.2 and up
4    4.4 and up
```

```
data.isnull().sum()
```

```
App                  0
Category             0
Rating            1474
Reviews              0
Size                 0
Installs             0
Type                 1
Price                0
Content Rating       1
Genres               0
Last Updated         0
Current Ver          8
Android Ver          3
dtype: int64
```

```
data_new = data.dropna(axis=0, how='any')
(data.shape, data_new.shape)
```

```
((10841, 13), (9360, 13))
```

```
data.dtypes
```

```
App                object
Category           object
Rating            float64
Reviews            object
Size               object
Installs           object
Type               object
Price              object
Content Rating     object
Genres             object
Last Updated       object
Current Ver        object
Android Ver        object
dtype: object
```

```
#Price не нужен потому что, почти все значение 0-ые
data.drop(['Current Ver', 'Android Ver', 'Type', 'Genres', 'Last
Updated', 'App', 'Content Rating'], axis = 1, inplace = True)
```

```
data_new.isnull().sum()
```

```
App               0
Category          0
Rating            0
Reviews           0
Size              0
Installs          0
Type              0
Price             0
Content Rating    0
Genres            0
Last Updated      0
Current Ver       0
Android Ver       0
dtype: int64
```

```
#Ограничение в 500 элементов
parts = np.split(data, [500], axis=0)
data_new = parts[0]
```

```
data_new.shape
```

```
(500, 6)
```

```
le = LabelEncoder()
le.fit(data_new.Category)
data_new['Category'] = le.transform(data_new.Category)
data_new
data_new['Size'] = data_new['Size'].map(lambda x: str(x)[:-1])
data_new['Installs'] = data_new['Installs'].map(lambda x: str(x)[:-1])
data_new['Installs'] = data_new['Installs'].replace(',','',
regex=True)
data_new['Size'] = data_new['Size'].replace(['Varies with devic'],
np.nan)
data_new['Price'] = data_new.Price.str.replace('$','')
```

```
data_new.head()
data_new['Price'].unique()
```

```
array(['0', '4.99', '3.99', '6.99', '1.49', '2.99', '7.99'],
dtype=object)
```

```
data_new = data_new.dropna(axis=0, how='any')
data_new.isnull().sum()
```

```
Category    0
Rating      0
Reviews     0
Size        0
Installs    0
Price       0
dtype: int64
```

```
data_new['Reviews'].unique()
```

```
array(['159', '967', '87510', '215644', '167', '178', '36815',
'13791',
       '121', '13880', '8788', '44829', '4326', '1518', '55', '3632',
       '27', '194216', '224399', '450', '654', '7699', '118', '192',
       '20260', '203', '136', '223', '1120', '227', '5035', '1015',
'353',
       '564', '8145', '158', '591', '117', '176', '2206', '26',
'174531',
       '1070', '85', '845', '367', '1598', '284', '129', '542',
'10479',
       '805', '1403', '3971', '534', '7774', '38846', '2431', '6090',
       '295', '190', '52530', '116986', '1379', '271920', '7021',
'197',
       '737', '3574', '994', '197136', '142', '15168', '2155', '138',
       '5414', '348', '250', '3617', '4806', '31433', '5097', '1754',
       '2680', '1288', '18900', '49790', '1150', '1739', '2225',
'4369',
       '8572', '964', '104', '601', '36', '187', '30', '134', '74',
       '9315', '75', '38', '26834', '2277', '2280', '184', '9', '364',
       '18', '473', '66', '3871', '257', '62', '1857', '4478', '418',
       '22486', '1435', '116507', '90468', '860', '363934', '17506',
       '1862', '2084', '47303', '85842', '7831', '91615', '4620',
'21336',
       '26875', '1778', '2709', '527', '1322', '1680', '2739', '1065',
       '51269', '30105', '156', '341157', '85185', '1002861', '16589',
       '148945', '4458', '62272', '8941', '46353', '30847', '188841',
       '4034', '45964', '6903', '31614', '207372', '1225', '380837',
       '10600', '74359', '822', '2287', '4162', '14760', '286897',
       '103755', '46505', '11442', '10295', '296', '29313', '1802',
       '1383', '23175', '5868', '5448', '4159', '20815', '78662',
'7149',
       '3079', '5800', '16422', '108741', '624', '1661', '308',
'5211',
       '1058', '1002859', '413', '24005', '57106', '2249', '516',
'834',
       '1010', '238970', '302', '438', '73', '39', '144', '2181',
'93965',
       '1446', '12088', '314', '15194', '22551', '29839', '279',
'1677',
       '757', '115', '125', '9952', '21', '15', '3596', '1006',
'5968',
```

```
       '4895', '125257', '158679', '4187998', '659395', '4785892',
       '30209', '36901', '192948', '13698', '20769', '36880',
'615381',
       '33053', '3648120', '136662', '42370', '2939', '40751',
'17712922',
       '25021', '27187', '4785988', '122498', '132014', '83239',
       '2876500', '28238', '335646', '349384', '37320', '36893',
'15880',
       '2264916', '42925', '3648480', '17714850', '13100', '55098',
       '1133501', '12578', '10965', '190613', '125232', '72065',
'27540',
       '104990', '177703', '17529', '177263', '237468', '32254',
'483565',
       '552441', '93825', '15287', '205739', '9498', '4188142',
'88427',
       '183374', '20901', '122595', '124346', '255', '41420', '29208',
       '191032', '57', '2', '285726', '2556', '7779', '61637',
'12632',
       '48845', '31320', '172460', '4195', '11633', '10212', '37053',
       '667', '13202'], dtype=object)
```

Масштабирование данных

MinMax масштабирование

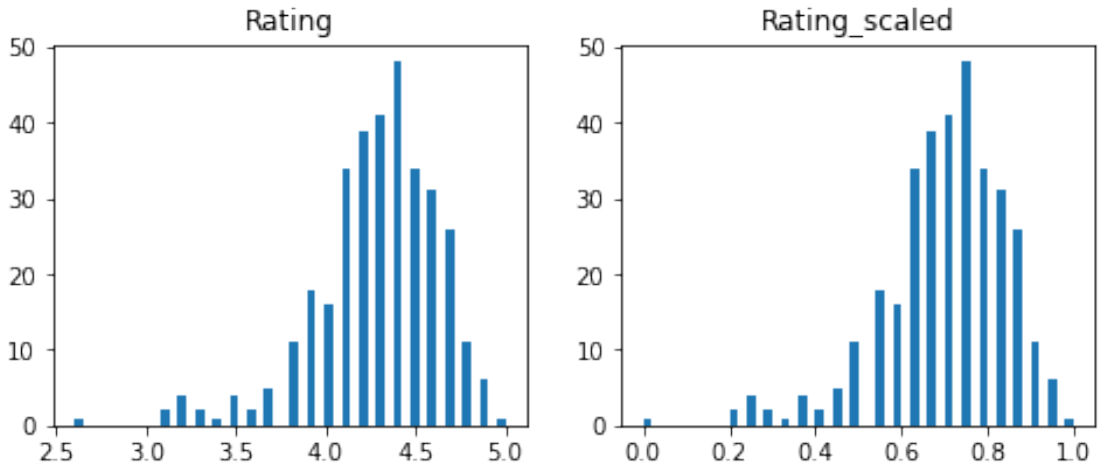
```
from sklearn.preprocessing import MinMaxScaler, StandardScaler,
Normalizer

# Числовые колонки для масштабирования
num_cols = ['Rating','Reviews', 'Installs', 'Category', 'Price']
scale_cols = num_cols

sc1 = MinMaxScaler()
sc1_data_new = sc1.fit_transform(data_new[scale_cols])

for i in range(len(scale_cols)):
    col = scale_cols[i]
    new_col_name = col + '_scaled'
    data_new[new_col_name] = sc1_data_new[:,i]
```

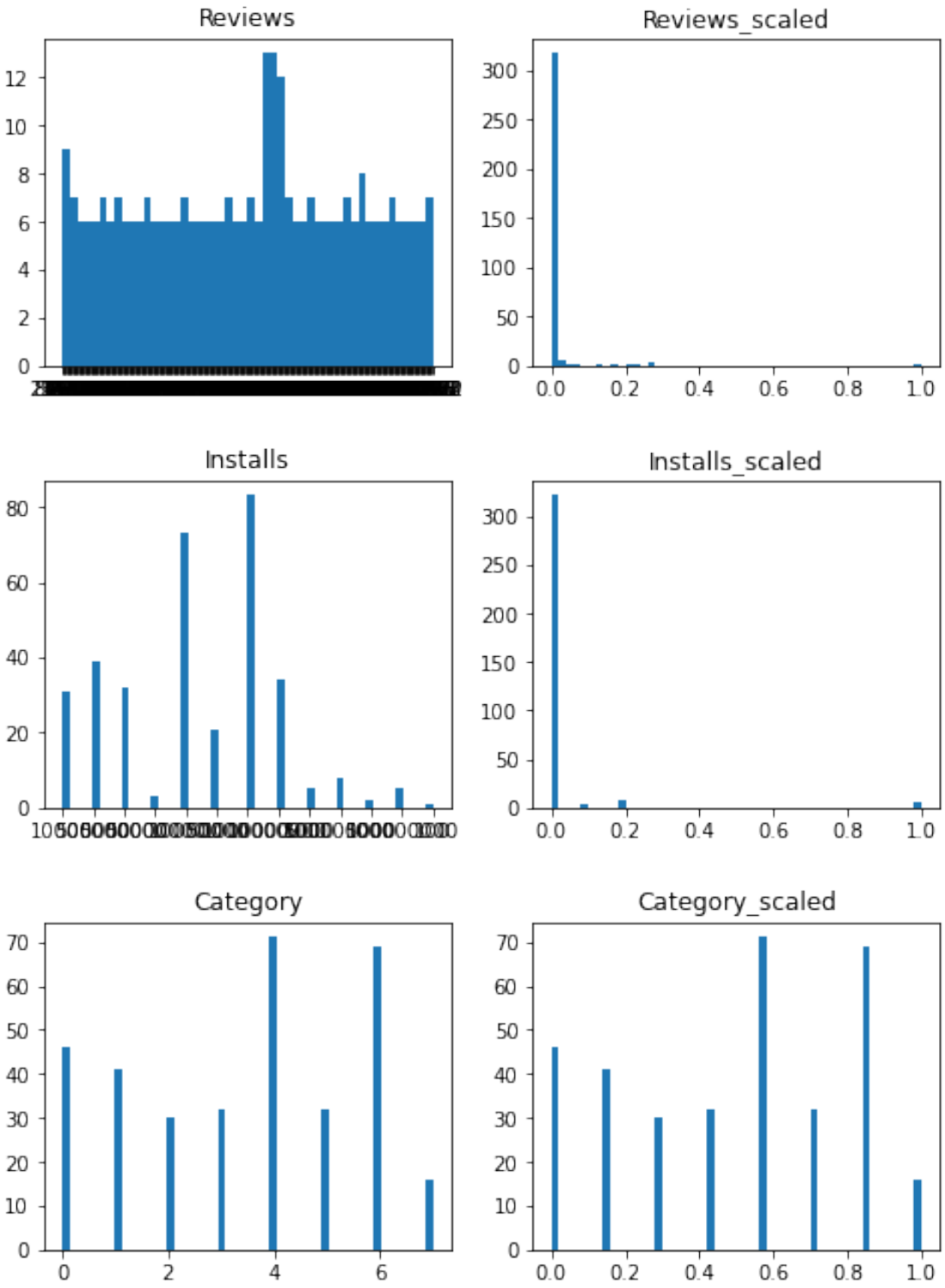
```
<ipython-input-613-ed1cbef29b17>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  data_new[new_col_name] = sc1_data_new[:,i]
```

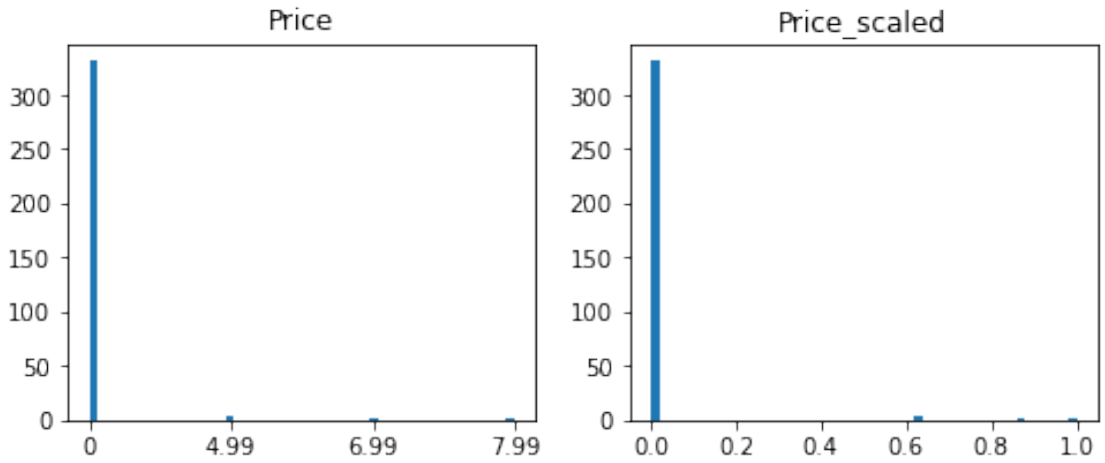
```
data_new.head()
```

```
   Category  Rating Reviews Size  Installs Price  Rating_scaled  \
0         0     4.1     159   19     10000     0       0.625000
1         0     3.9     967   14    500000     0       0.541667
2         0     4.7   87510  8.7   5000000     0       0.875000
3         0     4.5  215644   25  50000000     0       0.791667
4         0     4.3     967  2.8    100000     0       0.708333

   Reviews_scaled  Installs_scaled  Category_scaled  Price_scaled
0        0.000009          0.00002              0.0           0.0
1        0.000054          0.00100              0.0           0.0
2        0.004940          0.01000              0.0           0.0
3        0.012173          0.10000              0.0           0.0
4        0.000054          0.00020              0.0           0.0
```

```
for col in scale_cols:
    col_scaled = col + '_scaled'

    fig, ax = plt.subplots(1, 2, figsize=(8,3))
    ax[0].hist(data_new[col], 50)
    ax[1].hist(data_new[col_scaled], 50)
    ax[0].title.set_text(col)
    ax[1].title.set_text(col_scaled)
    plt.show()
```




Reviews
Reviews_scaled
Installs
Installs_scaled
Category
Category_scaled




```
data_new.drop(['Rating', 'Reviews', 'Installs', 'Category', 'Price'],
axis = 1, inplace = True)
```

```
/Users/egorzhidkov/opt/anaconda3/lib/python3.8/site-packages/pandas/
core/frame.py:4163: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  return super().drop(
```

Построение моделей

```
X = data_new.drop(['Rating_scaled'], axis = 1)
Y = data_new.Rating_scaled
print('Входные данные:\n\n', X.head(), '\n\nВыходные данные:\n\n',
Y.head())
```

```
Входные данные:

   Size  Reviews_scaled  Installs_scaled  Category_scaled
Price_scaled
0   19        0.000009          0.00002              0.0           0.0
1   14        0.000054          0.00100              0.0           0.0
2  8.7        0.004940          0.01000              0.0           0.0
3   25        0.012173          0.10000              0.0           0.0
4  2.8        0.000054          0.00020              0.0           0.0


Выходные данные:

 0    0.625000
1    0.541667
2    0.875000
```

```
3    0.791667
4    0.708333
Name: Rating_scaled, dtype: float64
```

```
X_train,  X_test,  Y_train,  Y_test = train_test_split(X,  Y,
random_state = 0, test_size = 0.30)
print('Входные параметры обучающей выборки:\n\n',X_train.head(), \
      '\n\nВходные параметры тестовой выборки:\n\n', X_test.head(), \
      '\n\nВыходные параметры обучающей выборки:\n\n', Y_train.head(),
\
      '\n\nВыходные параметры тестовой выборки:\n\n', Y_test.head())
```

```
Входные параметры обучающей выборки:

     Size  Reviews_scaled  Installs_scaled  Category_scaled
Price_scaled
48   7.9        0.000048           0.0002         0.000000
0.0
270   26        0.000102           0.0002         0.571429
0.0
281  8.6        0.000327           0.0002         0.571429
0.0
302   15        0.000047           0.0001         0.714286
0.0
197  3.9        0.000252           0.0010         0.571429
0.0

Входные параметры тестовой выборки:

     Size  Reviews_scaled  Installs_scaled  Category_scaled
Price_scaled
58   201        0.000079           0.0002         0.142857
0.0
56   5.6        0.000045           0.0001         0.142857
0.0
387   25        0.007452           0.0200         0.857143
0.0
379   61        0.001412           0.0100         0.857143
0.0
267   20        0.001655           0.0100         0.571429
0.0

Выходные параметры обучающей выборки:

 48     0.708333
270    0.625000
281    0.833333
302    0.541667
197    0.791667
Name: Rating_scaled, dtype: float64
```

```
Выходные параметры тестовой выборки:

 58     0.583333
56     0.750000
387    0.625000
379    0.458333
267    0.708333
Name: Rating_scaled, dtype: float64
```

Модель "Дерево решений"

```
from sklearn.tree import DecisionTreeRegressor

data_new['Size'] = data_new['Size'].replace(['Varies with devic'],
np.nan)

<ipython-input-620-fc04eac2b934>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  data_new['Size'] = data_new['Size'].replace(['Varies with devic'],
np.nan)

data_new.isnull().sum()

Size               0
Rating_scaled      0
Reviews_scaled     0
Installs_scaled    0
Category_scaled    0
Price_scaled       0
dtype: int64

dtc = DecisionTreeRegressor(random_state=1).fit(X_train, Y_train)
data_test_predicted_dtc = dtc.predict(X_test)
```

Модель "Случайный лес"

```
from sklearn.ensemble import RandomForestRegressor

RF = RandomForestRegressor(random_state=1).fit(X_train, Y_train)
data_test_predicted_rf = RF.predict(X_test)

from sklearn.metrics import mean_squared_error, r2_score
```

Оценка качества моделей:

В качестве метрик для оценки качества моделей я использую Mean squared error (средняя квадратичная ошибка), как наиболее часто используемую метрику для оценки качества регрессии, и метрику R^2 (коэффициент детерминации), потому что эта метрика является нормированной.

```
#Mean squared error - средняя квадратичная ошибка
print('Метрика MSE:\nДерево решений: {}\nСлучайный лес:
{}'.format(mean_squared_error(Y_test, data_test_predicted_dtc),
mean_squared_error(Y_test, data_test_predicted_rf)))
```

```
Метрика MSE:
Дерево решений: 0.03557325708061003
Случайный лес: 0.01845231992102397
```

```
# 4) Метрика R2 или коэффициент детерминации
print('Метрика R\u00B2:\nДерево решений: {}\nСлучайный лес:
{}'.format(r2_score(Y_test, data_test_predicted_dtc), r2_score(Y_test,
data_test_predicted_rf)))
```

```
Метрика R²:
Дерево решений: -0.7460031450661775
Случайный лес: 0.09432502784693997
```

Вывод

Исходя из оценки качества построенных моделей можно увидеть, что набор данных не подоходит для данных методов, потому что выборка значений слишком маленькая, например в Price из 500 значений 480 - это нули, из-за этого не получается сделать возможным нормальное масштабирование.