Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

Отчет по рубежному контролю №2

По дисциплине:
«Технологии машинного обучения»

Выполнил:

Студент группы ИУ5 _____ **Перлин Л.В.**

(Подпись, дата) (Фамилия И.О.)

Проверил:

_____ **Гапанюк Ю. Е.**

(Подпись, дата) (Фамилия И.О.)

Москва, 2021

Задание

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

Вариант № 17

Данные: https://www.kaggle.com/mathan/fifa-2018-match-statistics

ИУ5-63Б	Дерево решений	Случайный лес

Рубежный контроль №2

Перлин Леонид ИУ5-63Б

Импорт библиотек

In [4]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from pandas.plotting import scatter_matrix
import warnings
warnings.filterwarnings('ignore')
sns.set(style="ticks")
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

In [6]:

```
data = pd.read_csv('FIFA 2018 Statistics.csv')
```

In [7]:

```
data.head()
```

Out[7]:

5 rows × 27 columns

In [8]:

```
data.dtypes
```

Out[8]:

```
Date                        object
Team                        object
Opponent                    object
Goal Scored                  int64
Ball Possession %            int64
Attempts                     int64
On-Target                    int64
Off-Target                   int64
Blocked                      int64
Corners                      int64
Offsides                     int64
Free Kicks                   int64
Saves                        int64
Pass Accuracy %              int64
Passes                       int64
Distance Covered (Kms)       int64
Fouls Committed              int64
Yellow Card                  int64
Yellow & Red                 int64
Red                          int64
Man of the Match            object
1st Goal                   float64
Round                       object
PSO                         object
Goals in PSO                 int64
Own goals                  float64
Own goal Time              float64
dtype: object
```

In [9]:

```
data.isnull().sum()
# проверим есть ли пропущенные значения
```

Out[9]:

```
Date                        0
Team                        0
Opponent                    0
Goal Scored                 0
Ball Possession %           0
Attempts                    0
On-Target                   0
Off-Target                  0
Blocked                     0
Corners                     0
Offsides                    0
Free Kicks                  0
Saves                       0
Pass Accuracy %             0
Passes                      0
```

```
Distance Covered (Kms)      0
Fouls Committed             0
Yellow Card                 0
Yellow & Red                0
Red                         0
Man of the Match            0
1st Goal                   34
Round                       0
PSO                         0
Goals in PSO                0
Own goals                 116
Own goal Time             116
dtype: int64
```

In [10]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 128 entries, 0 to 127
Data columns (total 27 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Date                    128 non-null    object 
 1   Team                    128 non-null    object 
 2   Opponent                128 non-null    object 
 3   Goal Scored             128 non-null    int64  
 4   Ball Possession %       128 non-null    int64  
 5   Attempts                128 non-null    int64  
 6   On-Target               128 non-null    int64  
 7   Off-Target              128 non-null    int64  
 8   Blocked                 128 non-null    int64  
 9   Corners                 128 non-null    int64  
 10  Offsides                128 non-null    int64  
 11  Free Kicks              128 non-null    int64  
 12  Saves                   128 non-null    int64  
 13  Pass Accuracy %         128 non-null    int64  
 14  Passes                  128 non-null    int64  
 15  Distance Covered (Kms)  128 non-null    int64  
 16  Fouls Committed         128 non-null    int64  
 17  Yellow Card             128 non-null    int64  
 18  Yellow & Red            128 non-null    int64  
 19  Red                     128 non-null    int64  
 20  Man of the Match        128 non-null    object 
 21  1st Goal                94 non-null     float64
 22  Round                   128 non-null    object 
 23  PSO                     128 non-null    object 
 24  Goals in PSO            128 non-null    int64  
 25  Own goals               12 non-null     float64
 26  Own goal Time           12 non-null     float64
dtypes: float64(3), int64(18), object(6)
memory usage: 27.1+ KB
```

In [11]:

```
data.head()
```

Out[11]:

5 rows × 27 columns

In [12]:

```
#Построим корреляционную матрицу
fig, ax = plt.subplots(figsize=(15,7))
sns.heatmap(data.corr(method='pearson'), ax=ax, annot=True, fmt='.2f')
```

Out[12]:

```
<AxesSubplot:>
```

In [13]:

```
X = data[["On-Target","Off-Target","Blocked"]]
Y = data.Attempts
print('Входные данные:\n\n', X.head(), '\n\nВыходные данные:\n\n', Y.head())
```

```
Входные данные:

    On-Target  Off-Target  Blocked
0          7           3        3
1          0           3        3
2          3           3        2
3          4           6        4
4          3           6        4

Выходные данные:

 0    13
1     6
2     8
3    14
4    13
Name: Attempts, dtype: int64
```

In [14]:

```
X_train,  X_test,  Y_train,  Y_test = train_test_split(X,  Y, random_state =
0, test_size = 0.1)
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:

     On-Target  Off-Target  Blocked
94         1           7        5
30         4           5        1
33         1           8        4
2          3           3        2
59         2           5        1
```

Входные параметры тестовой выборки:

```
     On-Target  Off-Target  Blocked
40           5           5        0
24           6           7        2
86           3           5        4
51           5           8        7
8            5           4        3
```

Выходные параметры обучающей выборки:

```
 94    13
30    10
33    13
2      8
59     8
Name: Attempts, dtype: int64
```

Выходные параметры тестовой выборки:

```
 40    10
24    15
86    12
51    20
8     12
Name: Attempts, dtype: int64
```

In [23]:

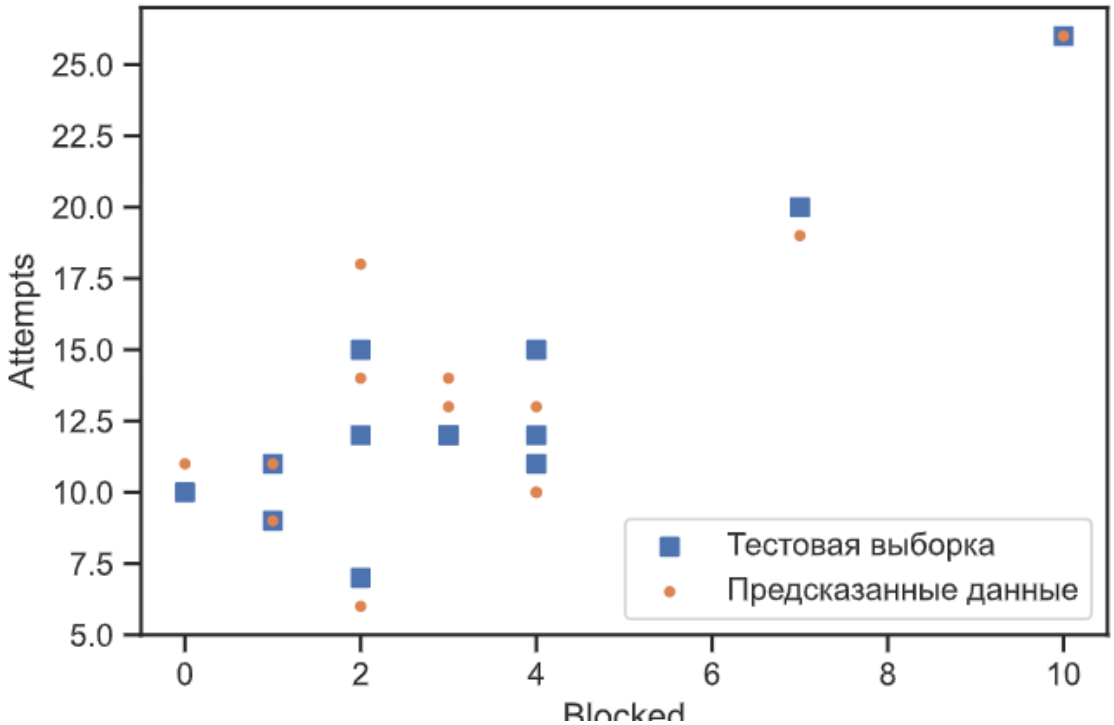
```
from sklearn.tree import DecisionTreeRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error,  median_
absolute_error, r2_score
```

In [24]:

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

In [25]:

```
plt.scatter(X_test.Blocked, Y_test,    marker = 's', label = 'Тестовая выборк
a')
plt.scatter(X_test.Blocked, data_test_predicted_dtc, marker = '.', label = 'П
редсказанные данные')
plt.legend (loc = 'lower right')
plt.xlabel ('Blocked')
plt.ylabel ('Attempts')
plt.show()
```



In [19]:

```
from sklearn.ensemble import RandomForestRegressor
```

In [20]:

```
forest_1 = RandomForestRegressor(n_estimators=5, oob_score=True, random_state
=10)
forest_1.fit(X, Y)
```

Out[20]:

```
RandomForestRegressor(n_estimators=5, oob_score=True, random_state=10)
```

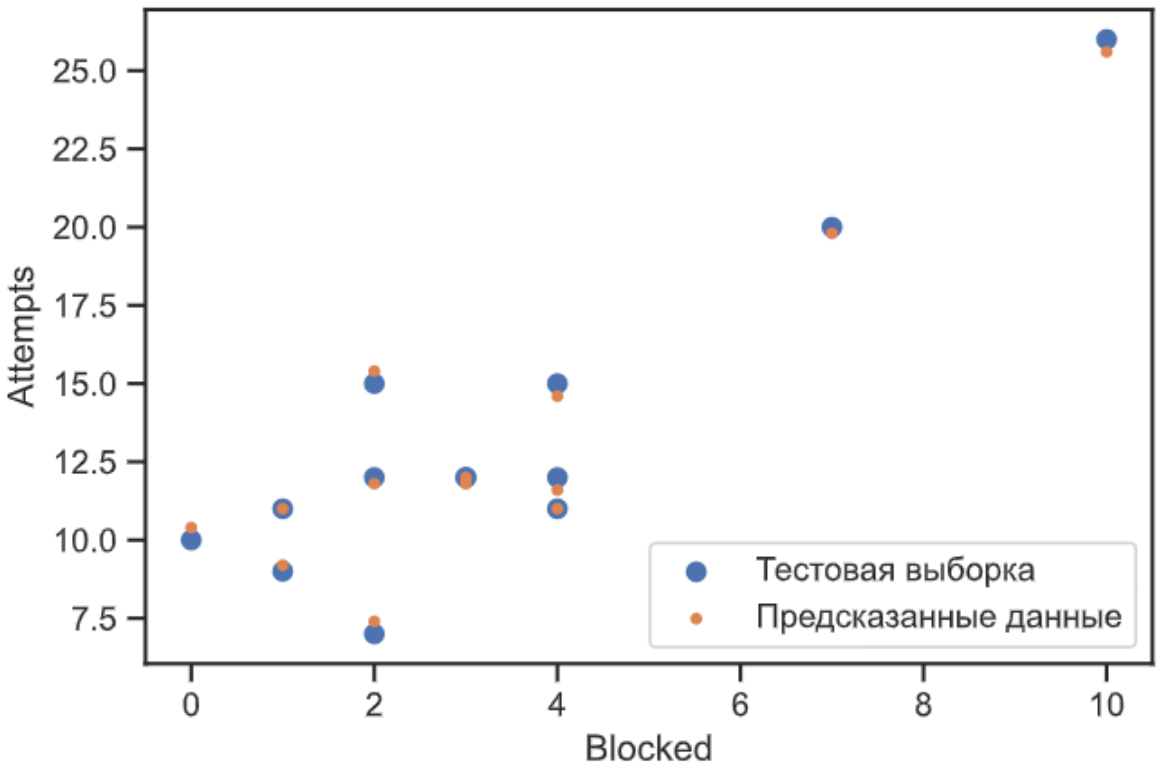
In [21]:

```
Y_predict = forest_1.predict(X_test)
print('Средняя абсолютная ошибка:',    mean_absolute_error(Y_test, Y_predict))
print('Средняя квадратичная ошибка:', mean_squared_error(Y_test, Y_predict))
print('Median absolute error:',        median_absolute_error(Y_test, Y_predict
))
print('Коэффициент детерминации:',     r2_score(Y_test, Y_predict))
```

```
Средняя абсолютная ошибка: 0.24615384615384597
Средняя квадратичная ошибка: 0.08615384615384608
Median absolute error: 0.1999999999999993
Коэффициент детерминации: 0.9962454873646209
```

In [22]:

```
plt.scatter(X_test.Blocked, Y_test,    marker = 'o', label = 'Тестовая выборк
а')
plt.scatter(X_test.Blocked, Y_predict, marker = '.', label = 'Предсказанные д
анные')
plt.legend(loc = 'lower right')
plt.xlabel('Blocked')
plt.ylabel('Attempts')
plt.show()
```


25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
Attempts
0
2
4
6
8
10
Blocked
Тестовая выборка
Предсказанные данные