Светашева Юлия ИУ5-64Б

17 вариант РК-2

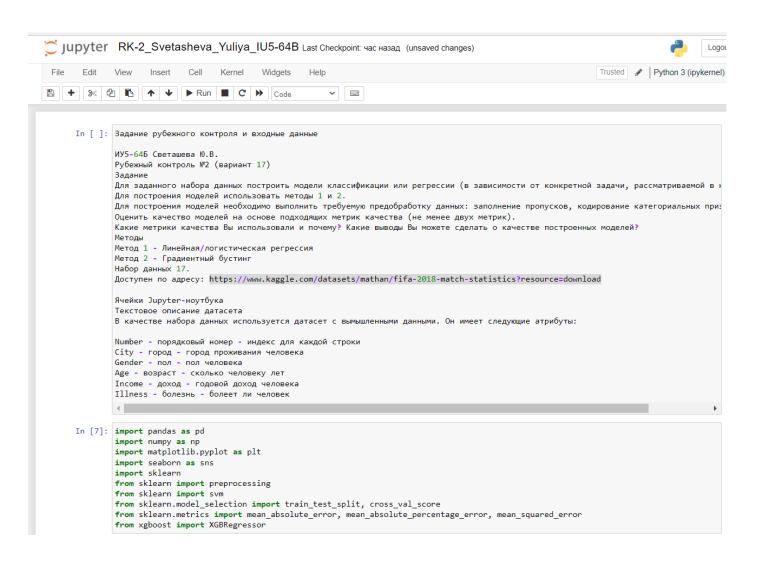
Задание. Для заданного набора данных (по Вашему варианту) постройте модели классификации или регрессии (в зависимости от конкретной задачи, рассматриваемой в наборе данных). Для построения моделей используйте методы 1 и 2 (по варианту для Вашей группы). Оцените качество моделей на основе подходящих метрик 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.

• Для студентов групп ИУ5-61Б, ИУ5-62Б, ИУ5-63Б, ИУ5-64Б, ИУ5-65Б, РТ5-61Б номер варианта = номер в списке группы.

Группа	Метод №1	Метод №2			
ИУ5-64Б, ИУ5Ц-	Линейная/логистическая	Градиентный			
84Б	регрессия	бустинг			

Используемый набор данных: Predict FIFA 2018 Man of the Match | Kaggle

Результат:





Trusted / Python 3 (ipykernel)



In [8]: data = pd.read_csv("FIFA 2018 Statistics.csv")

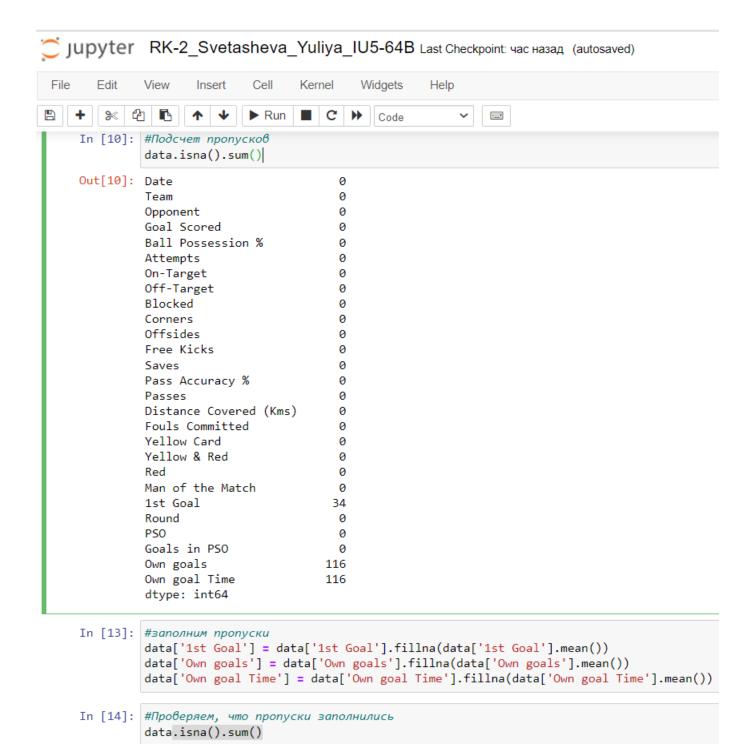
Out[8]:

	Date	Team	Opponent	Goal Scored	Ball Possession %	Attempts	On- Target	Off- Target	Blocked	Corners	 Yellow Card	Yellow & Red	Red	Man of the Match	1st Goal	Round	PSO	Goals in PSO
0	14- 06- 2018	Russia	Saudi Arabia	5	40	13	7	3	3	6	 0	0	0	Yes	12.0	Group Stage	No	0
1	14- 06- 2018	Saudi Arabia	Russia	0	60	6	0	3	3	2	 0	0	0	No	NaN	Group Stage	No	0
2	15- 06- 2018	Egypt	Uruguay	0	43	8	3	3	2	0	 2	0	0	No	NaN	Group Stage	No	0
3	15- 06- 2018	Uruguay	Egypt	1	57	14	4	6	4	5	 0	0	0	Yes	89.0	Group Stage	No	0
4	15- 06- 2018	Morocco	Iran	0	64	13	3	6	4	5	 1	0	0	No	NaN	Group Stage	No	0
123	11- 07- 2018	England	Croatia	1	46	11	1	6	4	4	 1	0	0	No	5.0	Semi- Finals	No	0
124	14- 07- 2018	Belgium	England	2	43	12	4	3	5	4	 1	0	0	Yes	4.0	3rd Place	No	0
125	14- 07- 2018	England	Belgium	0	57	15	5	7	3	5	 2	0	0	No	NaN	3rd Place	No	0
126	15- 07- 2018	France	Croatia	4	39	8	6	1	1	2	 2	0	0	Yes	18.0	Final	No	0
127	15- 07- 2018	Croatia	France	2	61	15	3	8	4	6	 1	0	0	No	28.0	Final	No	0

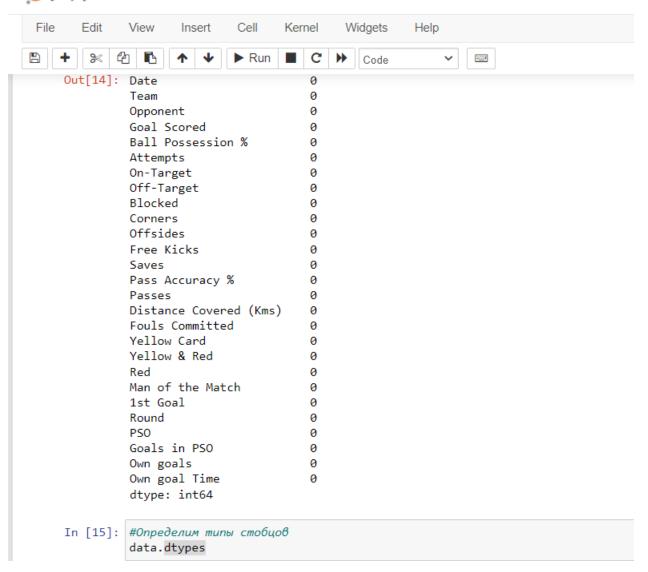
128 rows × 27 columns

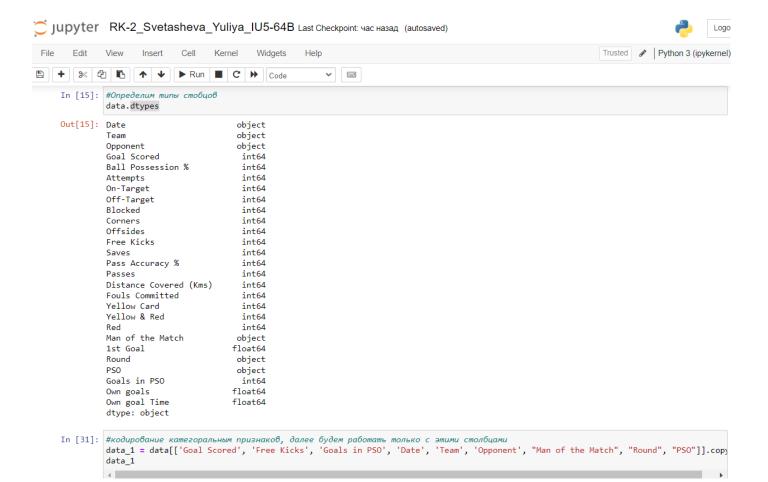
Jupyter RK-2_Svetasheva_Yuliya_IU5-64B Last Checkpoint: час назад (autosaved) File Edit View Insert Cell Kernel Widgets Help ▶ Run ■ C → Code [23332] In [9]: data.keys().to_list() Out[9]: ['Date', 'Team', 'Opponent', 'Goal Scored', 'Ball Possession %', 'Attempts', 'On-Target', 'Off-Target', 'Blocked', 'Corners', 'Offsides', 'Free Kicks', 'Saves', 'Pass Accuracy %', 'Passes', 'Distance Covered (Kms)', 'Fouls Committed', 'Yellow Card', 'Yellow & Red', 'Red', 'Man of the Match', '1st Goal', 'Round', 'PSO', 'Goals in PSO', 'Own goals', 'Own goal Time'] In [10]: #Подсчет пропусков

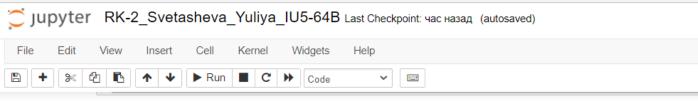
data.isna().sum()



◯ Jupyter RK-2_Svetasheva_Yuliya_IU5-64B Last Checkpoint: час назад (autosaved)







Out[31]:

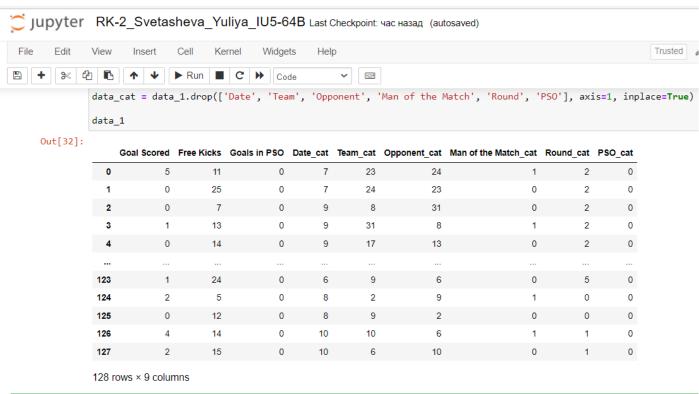
	Goal Scored	Free Kicks	Goals in PSO	Date	Team	Opponent	Man of the Match	Round	PSO
0	5	11	0	14-06-2018	Russia	Saudi Arabia	Yes	Group Stage	No
1	0	25	0	14-06-2018	Saudi Arabia	Russia	No	Group Stage	No
2	0	7	0	15-06-2018	Egypt	Uruguay	No	Group Stage	No
3	1	13	0	15-06-2018	Uruguay	Egypt	Yes	Group Stage	No
4	0	14	0	15-06-2018	Morocco	Iran	No	Group Stage	No
123	1	24	0	11-07-2018	England	Croatia	No	Semi- Finals	No
124	2	5	0	14-07-2018	Belgium	England	Yes	3rd Place	No
125	0	12	0	14-07-2018	England	Belgium	No	3rd Place	No
126	4	14	0	15-07-2018	France	Croatia	Yes	Final	No
127	2	15	0	15-07-2018	Croatia	France	No	Final	No

128 rows × 9 columns

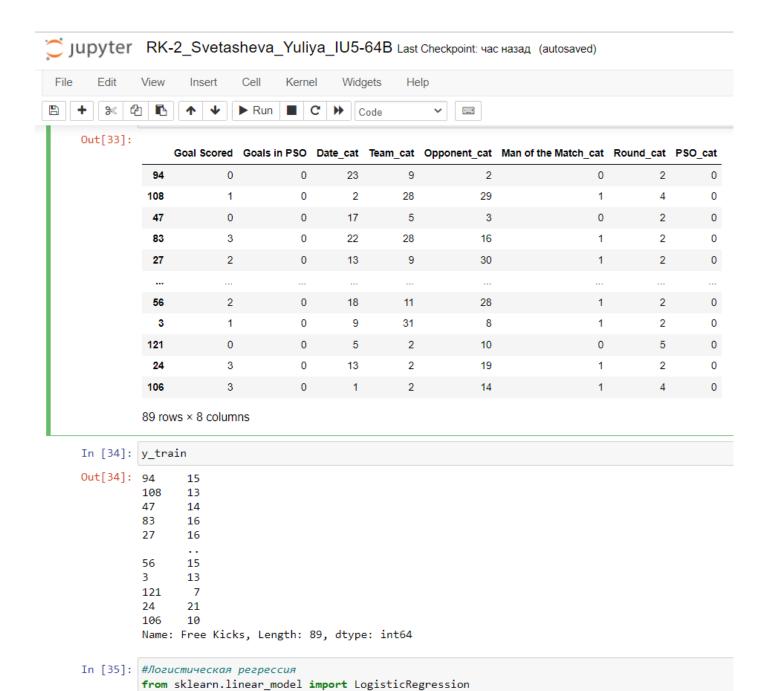
```
In [32]: #Κοδυροβαниε καπεεορυαльных πρυ3Ηακοβ
data_1["Date"].value_counts()
data_1["Date"] = data_1["Date"].astype('category')

data_1["Team"] = data_1["Team"].astype('category')
data_1["Opponent"] = data_1["Opponent"].astype('category')
data_1["Man of the Match"] = data_1["Man of the Match"].astype('category')
data_1["Round"] = data_1["Round"].astype('category')
data_1["PSO"] = data_1["PSO"].astype('category')

#Hазначить закодированную переменную новосу столбцу с помощью метода доступа
data_1["Date_cat"] = data_1["Date"].cat.codes
data_1["Team_cat"] = data_1["Team"].cat.codes
data_1["Man of the Match_cat"] = data_1["Man of the Match"].cat.codes
data_1["Round_cat"] = data_1["Round"].cat.codes
data_1["Round_cat"] = data_1["Round"].cat.codes
data_1["PSO_cat"] = data_1["PSO"].cat.codes
```



```
In [33]: #разделение выборки
from sklearn.model_selection import train_test_split
y = data_1['Free Kicks']
X = data_1.drop('Free Kicks', axis=1)|
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=3)
x_train
```



```
Jupyter RK-2 Svetasheva Yuliya IU5-64B Last Checkpoint: час назад (autosaved)
File
       Edit
              View
                     Insert
                             Cell
                                    Kernel
                                           Widgets
                                                       Help
                                                                                                                    Trusted
A code
A code
A code
A code
                                                           ~
     In [47]: def print_metrics(y_test, y_pred):
                  print(f"R^2: {r2_score(y_test, y_pred)}")
                  print(f"MSE: {mean_squared_error(y_test, y_pred)}")
                  print(f"MAE: {mean_absolute_error(y_test, y_pred)}")
     In [41]: import warnings
              warnings.filterwarnings('ignore')
              model_logistic = LogisticRegression()
              model_logistic.fit(x_train, y_train)
    Out[41]: LogisticRegression()
     In [43]: targ logistic = model logistic.predict(x test)
     In [44]: | mae = mean_absolute_error(y_test,targ_logistic)
              mape = mean_absolute_percentage_error(y_test,targ_logistic)
              mse = mean_squared_error(y_test,targ_logistic)
              print('MAE:' + str(round(mae,3)) + ' MAPE:' + str(round(mape,3)) + ' MSE:' + str(round(mse,3)))
              MAE:5.564 MAPE:0.43 MSE:45.513
     In [49]: #Градиентный брустинг
              XGB_model = XGBRegressor()
              mape = -cross_val_score(XGB_model,x_train,y_train,cv=4,scoring='neg_mean_absolute_percentage_error').mean()
              mae = -cross_val_score(XGB_model,x_train,y_train,cv=4,scoring='neg_mean_absolute_error').mean()
              mse = -cross_val_score(XGB_model,x_train,y_train,cv=4,scoring='neg_mean_squared_error').mean()
              print('SVM Errors')
              print('MAE:' + str(round(mae,3)) + ' MAPE:' + str(round(mape,3)) + ' MSE:' + str(round(mse,3)))
              SVM Errors
              MAE:4.361 MAPE:0.355 MSE:30.424
     In [50]: XGB_model.fit(x_train,y_train)
              mae = mean_absolute_error(y_test,XGB_model.predict(x_test))
              mape = mean_absolute_percentage_error(y_test,XGB_model.predict(x_test))
              mse = mean_squared_error(y_test,XGB_model.predict(x_test))
              print('MAE:' + str(round(mae,3)) + ' MAPE:' + str(round(mape,3)) + ' MSE:' + str(round(mse,3)))
              MAE:5.732 MAPE:0.416 MSE:45.084
 In [62]: #Сравнение моделей
```

Видим, что модель Градиентный брустинг показал себя лучше, чем логистическая регрессия

Кодируем категориальные признаки

Разделяем выборки