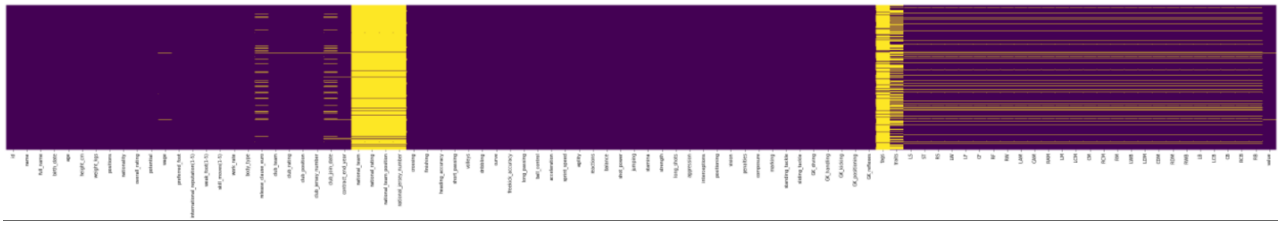
Preprocessing:

1- Handling nulls:

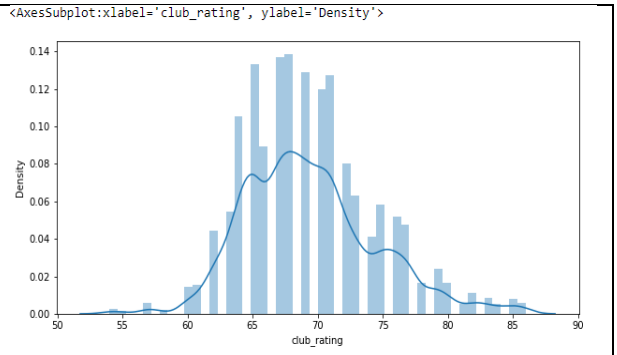
- Representing the nulls in columns using:
  sns.heatmap(player_data.isnull(),yticklabels=False,cbar=False,cmap="viridis",ax=ax)



- Filling nulls with Zeros:
  Columns → club_join_date, contract_end_year
  Reason → we considered, that if we found null in these columns means that player didn't join the club

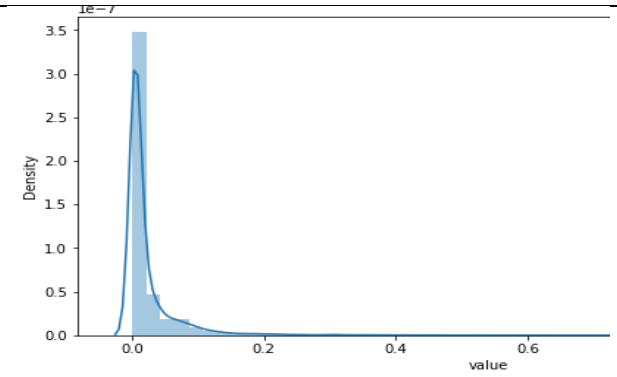
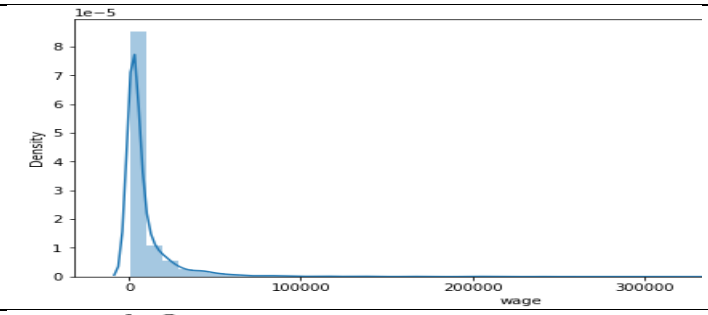
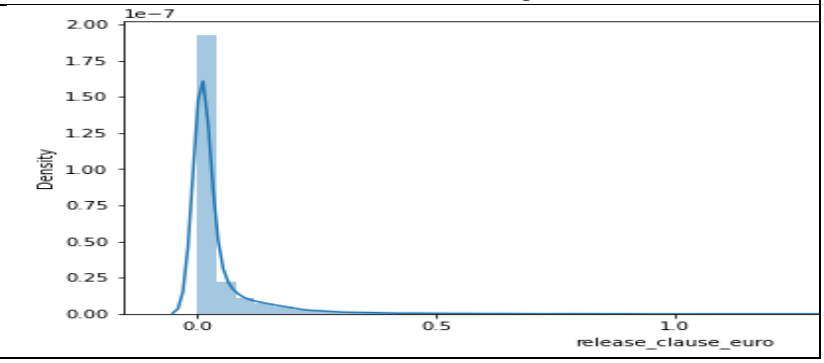
  Columns → national_rating
  Reason → Null values are greater than 90% in column, but non missing values may help us in prediction of value so we replaced it with zeros

- Replace nulls with mean:

Club_rating	

Reason → Data is normally distributed as shown in histogram plot, so mean suitable for it

- Replace nulls with mode:

value	
wage	
release_clause_euro	

Reason → Data is skewed as shown in histogram plot, so mean not suitable and we found the median equal null

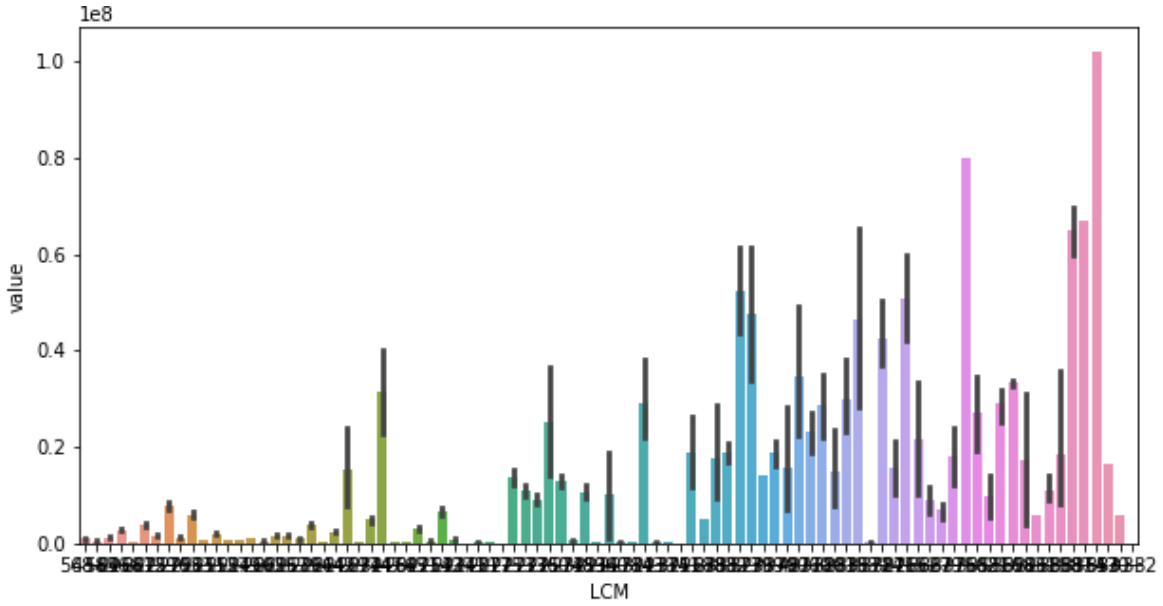
Columns → traits

Reason → replace with mode because it is categorical column

- Drop nulls:
  Columns → national_team, national_team_position, tags, national_jersey_number, club_jersey_number
  Reason → Null values are greater than 95% from each column

- Predict nulls:

Bar plot of (LCM with VALUE)



Columns → From column number 65 to 91 (zero based)
Reason →we found that the last 27 columns are dependent on **value** as shown in figure for example**,** so we predict nulls of each column with respect to **value column.**
Algorithm → predict using KNeighborsClassifier, with X equal "one of the columns, that we want to predict its value" and Y equal "value"

2- Handling categories:

body_type	○ First : we found data not true ,so we replaced it with mean ○ Then: apply Label encoding (as it has ordinal values)
work_rate	○ First: Ranking its categories (Ex:"High/ High": 9, "High/ Medium": 8 , "High/ Low": 7)
positions	○ First: Split string with(",") ○ Then: Generate 4 columns (1st position,2nd position,3rd position, 4th position). ○ Get correlation between 4 columns & y ,found it low so we drop column
preferred_foot	○ Apply One hot encoding ○ Reason → It has only 2 values (Right and Left) and they are not ordinal
traits	○ Try to tokenize it and find 37 unique value, so one hot encoding will not be applicable, and its values are not ordinal, so we drop it
From column number 65 to 91 (Zero based)	○ Apply Target Encoding ○ Reason → it has 89 unique category, so one hot encoding will not be applicable, and its values are not ordinal, so we choose target encoding to rank these categories with respect to target column

3- Feature selection:

- Joining 2 columns into one column:
  Columns → club_join_date, contract_end_year
  Reason → Subtracting these 2 columns (after converting contract_end_year from objects to integers → years only) to get one column called years_player_club which contains the total amount of years that the player spends in this club. Because the dependency between club_join_date & contract_end_year is very high.

- Dropping columns:
  Columns → GK_handling, GK_kicking, GK_positioning, GK_reflexes, agility, ball_control, curve, dribbling, freekick_accuracy, long_passing, long_shots, marking, penalties, positioning, reactions, release_clause_euro, short_passing, shot_power, sliding_tackle, sprint_speed, standing_tackle, volleys
  CAM, CB, CDM, CF, CM, RAM, RB, RCB,RCM, RDM,RF,RM,RS,RW, RWB , ST

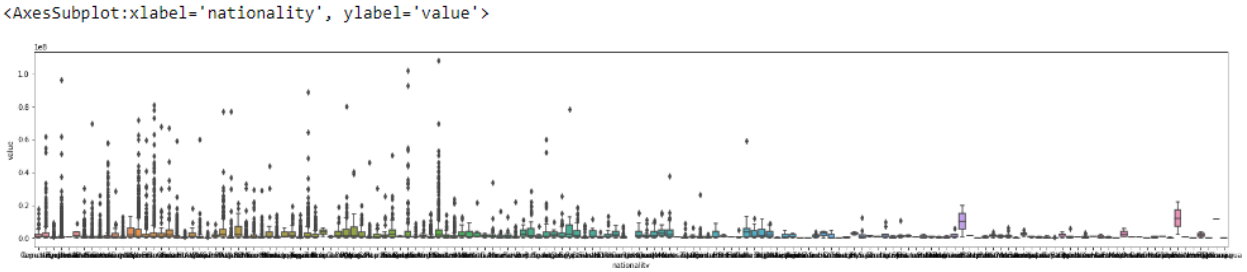
  Reason → High correlation with other features

  Columns → birth_date, club_position, club_team
  Reason → Has high correlation with other features. Using ANOVA algorithm

  Columns → Id, name, height_cm, full_name
  Reason → Low correlation with target column

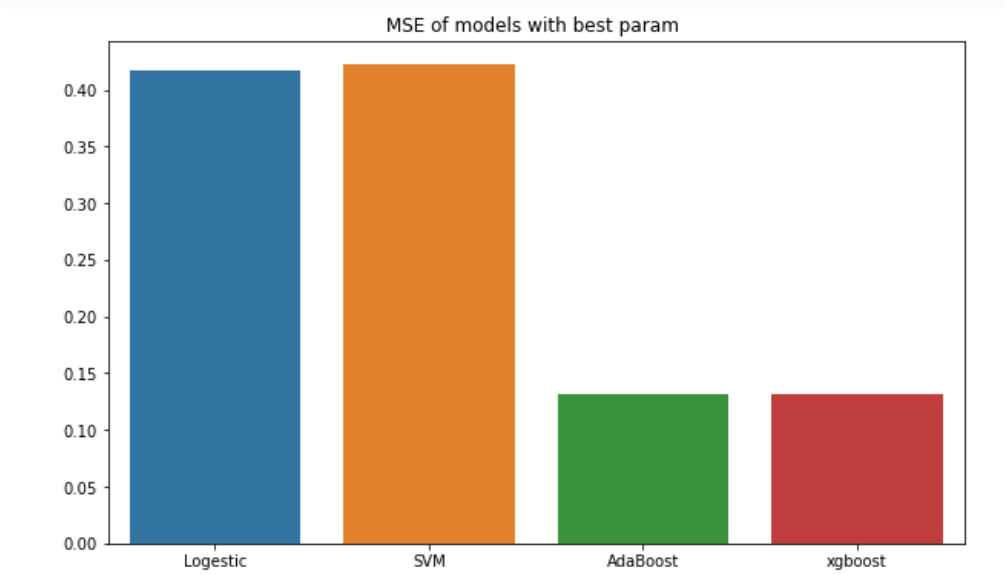
Columns → nationality

Reason →it has a lot of outliers and low correlation with value as shown in the figure below.

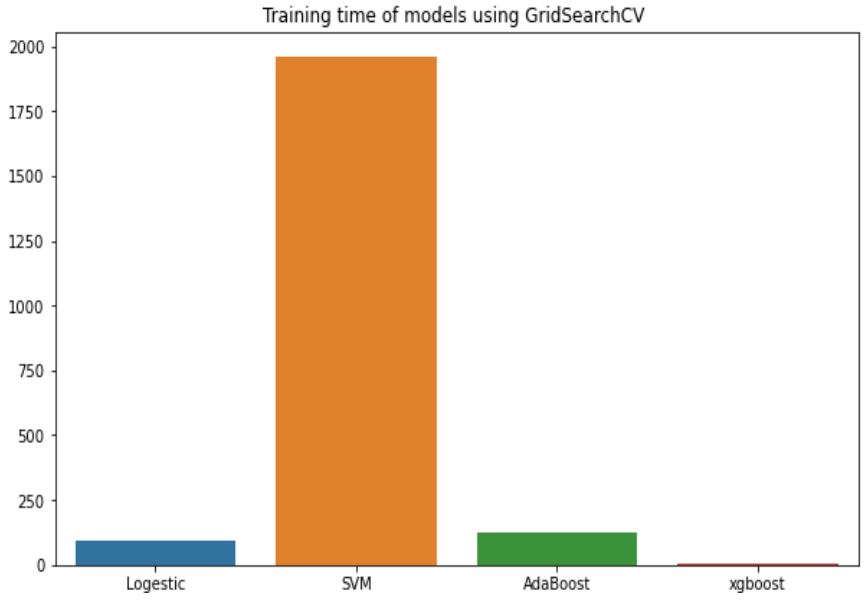


Box plot of (nationality with value)

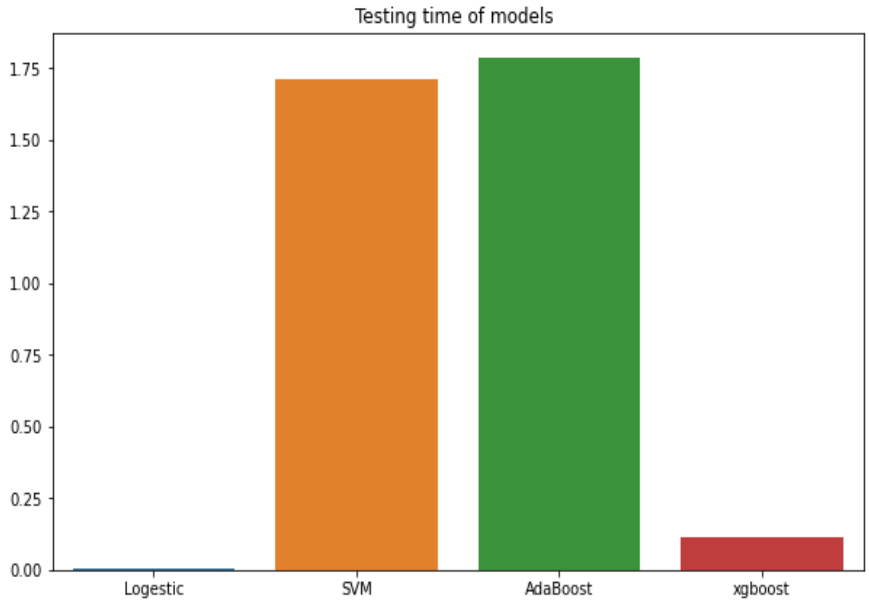
Classification (MSE)



Training time



Testing time



Preprocessing (differ from previous milestone)

Columns → From column number 65 to 91 (zero based),

Fill nulls with mode per each category in PlayerLevel column

Hyperparameters tuning

- AdaBoost classifier:

- We try to tune 2 hyperparameters (max_depth & n_estemators).

- First set max_depth=4 and tune n_estemators (using GridSearchCV)
  , then try with max_depth=5 tune n_estemators (using GridSearchCV).
  And I stopped training at max_depth=6 because training accuracy increasing, and testing accuracy doesn't change.

- Best parameters are:
  max_depth=6, n_estemators=500(git it from GridSearchCV)

- SVM
  - We try to tune 4 parameters (Kernel,C,gamma,decision_function_shape).

  - Tune the parameters using "GridSearchCV" ,by giving it a list of values per each hyperparameters.
    Then "GridSearchCV" uses a different combination of all the specified hyperparameters and their values and calculates the performance for each combination and selects the best value for the hyperparameters
  - Best parameters are:
    (C=100, decision_function_shape='ovo', gamma=0.1,kernel=rbf)

- Logistic Regression
  - We try to tune 2 parameters (max_iter,C).

- Tune the parameters using “GridSearchCV”, by giving it a list of values per each hyperparameters. Then “GridSearchCV” uses a different combination of all the specified hyperparameters and their values and calculates the performance for each combination and selects the best value for the hyperparameters

- Best parameters are:
  (C=200, max_iter=1000, multi_class='multinomial')

Conclusion:

At this phase we use 4 different models (SVM , Logistic Regression , AdaBoost,XGBoost) and tune hyperparameters per each model using “GridSearchCV” ,trying to get best model with best hyperparameters and avoid overfitting.

After ,that we choose XGBoost model that has highest testing accuracy with value = 0.9613644274277758 & training accuracy=1.0 and save it using joblib library to use it to predict unseen data