# FIFA 22 Players

# Phase 1

The objective of this project is to mine the players dataset from FIFA 22 and create a model for determining the best position for a player depending on the best features.

Also, as a future extension to this, we can create a model for predicting growth of a player based on all the current values of his attributes/ ratings.

The dataset was downloaded from: <https://www.kaggle.com/cashncarry/fifa-22-complete-player-dataset>

The dataset has 89 attributes and 19260 rows (One row for each FIFA registered player). So, we can consider this dataset as the whole universe of data for this domain.

The dataset contains attributes related to different soccer related abilities as well as some other features, i.e. value, wage, nationality, club name, playing position in club team, national team name, playing position in national team, jersey number in national team and club, release clause for clubs, contract info with clubs etc. The scores to the soccer related abilities are given out by FIFA ratings division on a scale of 0 to 100. TotalStats is sum of scores from all soccer abilities related attributes and BaseStats is sum of scores from position related attributes.

# Import libraries

# Necessary libraries  
import matplotlib.pyplot as plt  
import numpy as np  
import pandas as pd  
import seaborn as sns   
pd.set\_option('display.max\_colwidth', None)  
pd.set\_option('display.max\_columns', None)  
pd.set\_option('display.max\_rows', None)  
from sklearn.neighbors import NearestNeighbors  
from sklearn.preprocessing import StandardScaler  
import plotly.express as px  
from sklearn.cluster import KMeans  
from sklearn import tree  
from sklearn.model\_selection import KFold  
from sklearn.metrics import confusion\_matrix, classification\_report  
from sklearn.model\_selection import train\_test\_split  
from sklearn.feature\_selection import SelectKBest  
from sklearn.feature\_selection import f\_classif  
from numpy import set\_printoptions  
from sklearn.ensemble import RandomForestClassifier  
from sklearn.model\_selection import GridSearchCV

# Import CSV file

# Read the csv file into a pandas DataFrame  
df = pd.read\_csv('players\_fifa22.csv')#, dtype= str)  
df.head()

ID Name FullName Age Height \  
0 158023 L. Messi Lionel Messi 34 170   
1 188545 R. Lewandowski Robert Lewandowski 32 185   
2 20801 Cristiano Ronaldo C. Ronaldo dos Santos Aveiro 36 187   
3 231747 K. Mbappé Kylian Mbappé 22 182   
4 200389 J. Oblak Jan Oblak 28 188   
  
 Weight PhotoUrl Nationality \  
0 72 https://cdn.sofifa.com/players/158/023/22\_60.png Argentina   
1 81 https://cdn.sofifa.com/players/188/545/22\_60.png Poland   
2 83 https://cdn.sofifa.com/players/020/801/22\_60.png Portugal   
3 73 https://cdn.sofifa.com/players/231/747/22\_60.png France   
4 87 https://cdn.sofifa.com/players/200/389/22\_60.png Slovenia   
  
 Overall Potential Growth TotalStats BaseStats Positions BestPosition \  
0 93 93 0 2219 462 RW,ST,CF RW   
1 92 92 0 2212 460 ST ST   
2 91 91 0 2208 457 ST,LW ST   
3 91 95 4 2175 470 ST,LW ST   
4 91 93 2 1413 489 GK GK   
  
 Club ValueEUR WageEUR ReleaseClause ClubPosition \  
0 Paris Saint-Germain 78000000 320000 144300000 RW   
1 FC Bayern München 119500000 270000 197200000 ST   
2 Manchester United 45000000 270000 83300000 ST   
3 Paris Saint-Germain 194000000 230000 373500000 ST   
4 Atlético de Madrid 112000000 130000 238000000 GK   
  
 ContractUntil ClubNumber ClubJoined OnLoad NationalTeam \  
0 2023.0 30.0 2021 False Argentina   
1 2023.0 9.0 2014 False Poland   
2 2023.0 7.0 2021 False Portugal   
3 2022.0 7.0 2018 False France   
4 2023.0 13.0 2014 False Not in team   
  
 NationalPosition NationalNumber PreferredFoot IntReputation WeakFoot \  
0 RW 10.0 Left 5 4   
1 ST 9.0 Right 5 4   
2 ST 7.0 Right 5 4   
3 LW 10.0 Right 4 4   
4 NaN NaN Right 5 3   
  
 SkillMoves AttackingWorkRate DefensiveWorkRate PaceTotal ShootingTotal \  
0 4 Medium Low 85 92   
1 4 High Medium 78 92   
2 5 High Low 87 94   
3 5 High Low 97 88   
4 1 Medium Medium 87 92   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal Crossing \  
0 91 95 34 65 85   
1 79 85 44 82 71   
2 80 87 34 75 87   
3 80 92 36 77 78   
4 78 90 52 90 13   
  
 Finishing HeadingAccuracy ShortPassing Volleys Dribbling Curve \  
0 95 70 91 88 96 93   
1 95 90 85 89 85 79   
2 95 90 80 86 88 81   
3 93 72 85 83 93 80   
4 11 15 43 13 12 13   
  
 FKAccuracy LongPassing BallControl Acceleration SprintSpeed Agility \  
0 94 91 96 91 80 91   
1 85 70 88 77 79 77   
2 84 77 88 85 88 86   
3 69 71 91 97 97 92   
4 14 40 30 43 60 67   
  
 Reactions Balance ShotPower Jumping Stamina Strength LongShots \  
0 94 95 86 68 72 69 94   
1 93 82 90 85 76 86 87   
2 94 74 94 95 77 77 93   
3 93 83 86 78 88 77 82   
4 88 49 59 78 41 78 12   
  
 Aggression Interceptions Positioning Vision Penalties Composure \  
0 44 40 93 95 75 96   
1 81 49 95 81 90 88   
2 63 29 95 76 88 95   
3 62 38 92 82 79 88   
4 34 19 11 65 11 68   
  
 Marking StandingTackle SlidingTackle GKDiving GKHandling GKKicking \  
0 20 35 24 6 11 15   
1 35 42 19 15 6 12   
2 24 32 24 7 11 15   
3 26 34 32 13 5 7   
4 27 12 18 87 92 78   
  
 GKPositioning GKReflexes STRating LWRating LFRating CFRating \  
0 14 8 92 92 93 93   
1 8 10 92 85 88 88   
2 14 11 91 88 89 89   
3 11 6 91 90 90 90   
4 90 90 36 32 35 35   
  
 RFRating RWRating CAMRating LMRating CMRating RMRating LWBRating \  
0 93 92 93 93 90 93 69   
1 88 85 89 87 83 87 67   
2 89 88 89 89 81 89 66   
3 90 90 92 92 84 92 70   
4 35 32 41 38 41 38 35   
  
 CDMRating RWBRating LBRating CBRating RBRating GKRating   
0 67 69 64 53 64 22   
1 69 67 64 63 64 22   
2 62 66 63 56 63 23   
3 66 70 66 57 66 21   
4 39 35 35 36 35 92

# Checking the data types of each column  
df.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 19260 entries, 0 to 19259  
Data columns (total 90 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 ID 19260 non-null int64   
 1 Name 19260 non-null object   
 2 FullName 19260 non-null object   
 3 Age 19260 non-null int64   
 4 Height 19260 non-null int64   
 5 Weight 19260 non-null int64   
 6 PhotoUrl 19260 non-null object   
 7 Nationality 19260 non-null object   
 8 Overall 19260 non-null int64   
 9 Potential 19260 non-null int64   
 10 Growth 19260 non-null int64   
 11 TotalStats 19260 non-null int64   
 12 BaseStats 19260 non-null int64   
 13 Positions 19260 non-null object   
 14 BestPosition 19260 non-null object   
 15 Club 19260 non-null object   
 16 ValueEUR 19260 non-null int64   
 17 WageEUR 19260 non-null int64   
 18 ReleaseClause 19260 non-null int64   
 19 ClubPosition 19184 non-null object   
 20 ContractUntil 19184 non-null float64  
 21 ClubNumber 19184 non-null float64  
 22 ClubJoined 19260 non-null int64   
 23 OnLoad 19260 non-null bool   
 24 NationalTeam 19260 non-null object   
 25 NationalPosition 757 non-null object   
 26 NationalNumber 757 non-null float64  
 27 PreferredFoot 19260 non-null object   
 28 IntReputation 19260 non-null int64   
 29 WeakFoot 19260 non-null int64   
 30 SkillMoves 19260 non-null int64   
 31 AttackingWorkRate 19260 non-null object   
 32 DefensiveWorkRate 19260 non-null object   
 33 PaceTotal 19260 non-null int64   
 34 ShootingTotal 19260 non-null int64   
 35 PassingTotal 19260 non-null int64   
 36 DribblingTotal 19260 non-null int64   
 37 DefendingTotal 19260 non-null int64   
 38 PhysicalityTotal 19260 non-null int64   
 39 Crossing 19260 non-null int64   
 40 Finishing 19260 non-null int64   
 41 HeadingAccuracy 19260 non-null int64   
 42 ShortPassing 19260 non-null int64   
 43 Volleys 19260 non-null int64   
 44 Dribbling 19260 non-null int64   
 45 Curve 19260 non-null int64   
 46 FKAccuracy 19260 non-null int64   
 47 LongPassing 19260 non-null int64   
 48 BallControl 19260 non-null int64   
 49 Acceleration 19260 non-null int64   
 50 SprintSpeed 19260 non-null int64   
 51 Agility 19260 non-null int64   
 52 Reactions 19260 non-null int64   
 53 Balance 19260 non-null int64   
 54 ShotPower 19260 non-null int64   
 55 Jumping 19260 non-null int64   
 56 Stamina 19260 non-null int64   
 57 Strength 19260 non-null int64   
 58 LongShots 19260 non-null int64   
 59 Aggression 19260 non-null int64   
 60 Interceptions 19260 non-null int64   
 61 Positioning 19260 non-null int64   
 62 Vision 19260 non-null int64   
 63 Penalties 19260 non-null int64   
 64 Composure 19260 non-null int64   
 65 Marking 19260 non-null int64   
 66 StandingTackle 19260 non-null int64   
 67 SlidingTackle 19260 non-null int64   
 68 GKDiving 19260 non-null int64   
 69 GKHandling 19260 non-null int64   
 70 GKKicking 19260 non-null int64   
 71 GKPositioning 19260 non-null int64   
 72 GKReflexes 19260 non-null int64   
 73 STRating 19260 non-null int64   
 74 LWRating 19260 non-null int64   
 75 LFRating 19260 non-null int64   
 76 CFRating 19260 non-null int64   
 77 RFRating 19260 non-null int64   
 78 RWRating 19260 non-null int64   
 79 CAMRating 19260 non-null int64   
 80 LMRating 19260 non-null int64   
 81 CMRating 19260 non-null int64   
 82 RMRating 19260 non-null int64   
 83 LWBRating 19260 non-null int64   
 84 CDMRating 19260 non-null int64   
 85 RWBRating 19260 non-null int64   
 86 LBRating 19260 non-null int64   
 87 CBRating 19260 non-null int64   
 88 RBRating 19260 non-null int64   
 89 GKRating 19260 non-null int64   
dtypes: bool(1), float64(3), int64(73), object(13)  
memory usage: 13.1+ MB

# Deleting the ID columns  
del df['ID']  
df.columns

Index(['Name', 'FullName', 'Age', 'Height', 'Weight', 'PhotoUrl',  
 'Nationality', 'Overall', 'Potential', 'Growth', 'TotalStats',  
 'BaseStats', 'Positions', 'BestPosition', 'Club', 'ValueEUR', 'WageEUR',  
 'ReleaseClause', 'ClubPosition', 'ContractUntil', 'ClubNumber',  
 'ClubJoined', 'OnLoad', 'NationalTeam', 'NationalPosition',  
 'NationalNumber', 'PreferredFoot', 'IntReputation', 'WeakFoot',  
 'SkillMoves', 'AttackingWorkRate', 'DefensiveWorkRate', 'PaceTotal',  
 'ShootingTotal', 'PassingTotal', 'DribblingTotal', 'DefendingTotal',  
 'PhysicalityTotal', 'Crossing', 'Finishing', 'HeadingAccuracy',  
 'ShortPassing', 'Volleys', 'Dribbling', 'Curve', 'FKAccuracy',  
 'LongPassing', 'BallControl', 'Acceleration', 'SprintSpeed', 'Agility',  
 'Reactions', 'Balance', 'ShotPower', 'Jumping', 'Stamina', 'Strength',  
 'LongShots', 'Aggression', 'Interceptions', 'Positioning', 'Vision',  
 'Penalties', 'Composure', 'Marking', 'StandingTackle', 'SlidingTackle',  
 'GKDiving', 'GKHandling', 'GKKicking', 'GKPositioning', 'GKReflexes',  
 'STRating', 'LWRating', 'LFRating', 'CFRating', 'RFRating', 'RWRating',  
 'CAMRating', 'LMRating', 'CMRating', 'RMRating', 'LWBRating',  
 'CDMRating', 'RWBRating', 'LBRating', 'CBRating', 'RBRating',  
 'GKRating'],  
 dtype='object')

# Transforming categorical data to numerical data

# Converting AttackingWorkRate & DefensiveWorkRate to numerical columns  
df['AttackingWorkRate\_code'] = df['AttackingWorkRate'].apply(lambda x: 3 if x == 'High' \  
 else (2 if x == 'Medium' else 1))  
df['DefensiveWorkRate\_code'] = df['DefensiveWorkRate'].apply(lambda x: 3 if x == 'High' \  
 else (2 if x == 'Medium' else 1))

# Converting BestPosition to numerical categorical column  
df['BestPosition\_code'] = df['BestPosition'].astype('category').cat.codes  
# df[['BestPosition', 'BestPosition\_code']].value\_counts()

# Converting PreferredFoot to numerical categorical column  
df['PreferredFoot\_code'] = df['PreferredFoot'].astype('category').cat.codes  
# df[['PreferredFoot', 'PreferredFoot\_code']].value\_counts()

# Creating a dictionary of BestPoistion name and assigned code for future reference   
pos\_dict = df[['BestPosition', 'BestPosition\_code']].drop\_duplicates()\  
 .set\_index('BestPosition', drop=True).to\_dict(orient= 'dict')  
pos\_dict

{'BestPosition\_code': {'RW': 12,  
 'ST': 14,  
 'GK': 5,  
 'CM': 4,  
 'LW': 8,  
 'CDM': 2,  
 'LM': 7,  
 'CF': 3,  
 'CB': 1,  
 'CAM': 0,  
 'LB': 6,  
 'RB': 10,  
 'RM': 11,  
 'LWB': 9,  
 'RWB': 13}}

# Finally the transformed dataframe  
df.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 19260 entries, 0 to 19259  
Data columns (total 93 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 Name 19260 non-null object   
 1 FullName 19260 non-null object   
 2 Age 19260 non-null int64   
 3 Height 19260 non-null int64   
 4 Weight 19260 non-null int64   
 5 PhotoUrl 19260 non-null object   
 6 Nationality 19260 non-null object   
 7 Overall 19260 non-null int64   
 8 Potential 19260 non-null int64   
 9 Growth 19260 non-null int64   
 10 TotalStats 19260 non-null int64   
 11 BaseStats 19260 non-null int64   
 12 Positions 19260 non-null object   
 13 BestPosition 19260 non-null object   
 14 Club 19260 non-null object   
 15 ValueEUR 19260 non-null int64   
 16 WageEUR 19260 non-null int64   
 17 ReleaseClause 19260 non-null int64   
 18 ClubPosition 19184 non-null object   
 19 ContractUntil 19184 non-null float64  
 20 ClubNumber 19184 non-null float64  
 21 ClubJoined 19260 non-null int64   
 22 OnLoad 19260 non-null bool   
 23 NationalTeam 19260 non-null object   
 24 NationalPosition 757 non-null object   
 25 NationalNumber 757 non-null float64  
 26 PreferredFoot 19260 non-null object   
 27 IntReputation 19260 non-null int64   
 28 WeakFoot 19260 non-null int64   
 29 SkillMoves 19260 non-null int64   
 30 AttackingWorkRate 19260 non-null object   
 31 DefensiveWorkRate 19260 non-null object   
 32 PaceTotal 19260 non-null int64   
 33 ShootingTotal 19260 non-null int64   
 34 PassingTotal 19260 non-null int64   
 35 DribblingTotal 19260 non-null int64   
 36 DefendingTotal 19260 non-null int64   
 37 PhysicalityTotal 19260 non-null int64   
 38 Crossing 19260 non-null int64   
 39 Finishing 19260 non-null int64   
 40 HeadingAccuracy 19260 non-null int64   
 41 ShortPassing 19260 non-null int64   
 42 Volleys 19260 non-null int64   
 43 Dribbling 19260 non-null int64   
 44 Curve 19260 non-null int64   
 45 FKAccuracy 19260 non-null int64   
 46 LongPassing 19260 non-null int64   
 47 BallControl 19260 non-null int64   
 48 Acceleration 19260 non-null int64   
 49 SprintSpeed 19260 non-null int64   
 50 Agility 19260 non-null int64   
 51 Reactions 19260 non-null int64   
 52 Balance 19260 non-null int64   
 53 ShotPower 19260 non-null int64   
 54 Jumping 19260 non-null int64   
 55 Stamina 19260 non-null int64   
 56 Strength 19260 non-null int64   
 57 LongShots 19260 non-null int64   
 58 Aggression 19260 non-null int64   
 59 Interceptions 19260 non-null int64   
 60 Positioning 19260 non-null int64   
 61 Vision 19260 non-null int64   
 62 Penalties 19260 non-null int64   
 63 Composure 19260 non-null int64   
 64 Marking 19260 non-null int64   
 65 StandingTackle 19260 non-null int64   
 66 SlidingTackle 19260 non-null int64   
 67 GKDiving 19260 non-null int64   
 68 GKHandling 19260 non-null int64   
 69 GKKicking 19260 non-null int64   
 70 GKPositioning 19260 non-null int64   
 71 GKReflexes 19260 non-null int64   
 72 STRating 19260 non-null int64   
 73 LWRating 19260 non-null int64   
 74 LFRating 19260 non-null int64   
 75 CFRating 19260 non-null int64   
 76 RFRating 19260 non-null int64   
 77 RWRating 19260 non-null int64   
 78 CAMRating 19260 non-null int64   
 79 LMRating 19260 non-null int64   
 80 CMRating 19260 non-null int64   
 81 RMRating 19260 non-null int64   
 82 LWBRating 19260 non-null int64   
 83 CDMRating 19260 non-null int64   
 84 RWBRating 19260 non-null int64   
 85 LBRating 19260 non-null int64   
 86 CBRating 19260 non-null int64   
 87 RBRating 19260 non-null int64   
 88 GKRating 19260 non-null int64   
 89 AttackingWorkRate\_code 19260 non-null int64   
 90 DefensiveWorkRate\_code 19260 non-null int64   
 91 BestPosition\_code 19260 non-null int8   
 92 PreferredFoot\_code 19260 non-null int8   
dtypes: bool(1), float64(3), int64(74), int8(2), object(13)  
memory usage: 13.3+ MB

# Dropping unnecessary columns

# Dropping object data types as well as some unnecessary columns from the dataframe   
df\_select = df.drop(['Name', 'FullName','PhotoUrl', 'Nationality', 'Positions',\  
 'Club','ReleaseClause', 'ClubPosition', 'ContractUntil',\  
 'ClubNumber', 'ClubJoined', 'OnLoad', 'NationalTeam',\  
 'NationalPosition', 'NationalNumber', 'PreferredFoot',\  
 'AttackingWorkRate', 'DefensiveWorkRate'], axis = 1)  
df\_select.head()

Age Height Weight Overall Potential Growth TotalStats BaseStats \  
0 34 170 72 93 93 0 2219 462   
1 32 185 81 92 92 0 2212 460   
2 36 187 83 91 91 0 2208 457   
3 22 182 73 91 95 4 2175 470   
4 28 188 87 91 93 2 1413 489   
  
 BestPosition ValueEUR WageEUR IntReputation WeakFoot SkillMoves \  
0 RW 78000000 320000 5 4 4   
1 ST 119500000 270000 5 4 4   
2 ST 45000000 270000 5 4 5   
3 ST 194000000 230000 4 4 5   
4 GK 112000000 130000 5 3 1   
  
 PaceTotal ShootingTotal PassingTotal DribblingTotal DefendingTotal \  
0 85 92 91 95 34   
1 78 92 79 85 44   
2 87 94 80 87 34   
3 97 88 80 92 36   
4 87 92 78 90 52   
  
 PhysicalityTotal Crossing Finishing HeadingAccuracy ShortPassing \  
0 65 85 95 70 91   
1 82 71 95 90 85   
2 75 87 95 90 80   
3 77 78 93 72 85   
4 90 13 11 15 43   
  
 Volleys Dribbling Curve FKAccuracy LongPassing BallControl \  
0 88 96 93 94 91 96   
1 89 85 79 85 70 88   
2 86 88 81 84 77 88   
3 83 93 80 69 71 91   
4 13 12 13 14 40 30   
  
 Acceleration SprintSpeed Agility Reactions Balance ShotPower Jumping \  
0 91 80 91 94 95 86 68   
1 77 79 77 93 82 90 85   
2 85 88 86 94 74 94 95   
3 97 97 92 93 83 86 78   
4 43 60 67 88 49 59 78   
  
 Stamina Strength LongShots Aggression Interceptions Positioning \  
0 72 69 94 44 40 93   
1 76 86 87 81 49 95   
2 77 77 93 63 29 95   
3 88 77 82 62 38 92   
4 41 78 12 34 19 11   
  
 Vision Penalties Composure Marking StandingTackle SlidingTackle \  
0 95 75 96 20 35 24   
1 81 90 88 35 42 19   
2 76 88 95 24 32 24   
3 82 79 88 26 34 32   
4 65 11 68 27 12 18   
  
 GKDiving GKHandling GKKicking GKPositioning GKReflexes STRating \  
0 6 11 15 14 8 92   
1 15 6 12 8 10 92   
2 7 11 15 14 11 91   
3 13 5 7 11 6 91   
4 87 92 78 90 90 36   
  
 LWRating LFRating CFRating RFRating RWRating CAMRating LMRating \  
0 92 93 93 93 92 93 93   
1 85 88 88 88 85 89 87   
2 88 89 89 89 88 89 89   
3 90 90 90 90 90 92 92   
4 32 35 35 35 32 41 38   
  
 CMRating RMRating LWBRating CDMRating RWBRating LBRating CBRating \  
0 90 93 69 67 69 64 53   
1 83 87 67 69 67 64 63   
2 81 89 66 62 66 63 56   
3 84 92 70 66 70 66 57   
4 41 38 35 39 35 35 36   
  
 RBRating GKRating AttackingWorkRate\_code DefensiveWorkRate\_code \  
0 64 22 2 1   
1 64 22 3 2   
2 63 23 3 1   
3 66 21 3 1   
4 35 92 2 2   
  
 BestPosition\_code PreferredFoot\_code   
0 12 0   
1 14 1   
2 14 1   
3 14 1   
4 5 1

df\_select.info()

<class 'pandas.core.frame.DataFrame'>  
RangeIndex: 19260 entries, 0 to 19259  
Data columns (total 75 columns):  
 # Column Non-Null Count Dtype   
--- ------ -------------- -----   
 0 Age 19260 non-null int64   
 1 Height 19260 non-null int64   
 2 Weight 19260 non-null int64   
 3 Overall 19260 non-null int64   
 4 Potential 19260 non-null int64   
 5 Growth 19260 non-null int64   
 6 TotalStats 19260 non-null int64   
 7 BaseStats 19260 non-null int64   
 8 BestPosition 19260 non-null object  
 9 ValueEUR 19260 non-null int64   
 10 WageEUR 19260 non-null int64   
 11 IntReputation 19260 non-null int64   
 12 WeakFoot 19260 non-null int64   
 13 SkillMoves 19260 non-null int64   
 14 PaceTotal 19260 non-null int64   
 15 ShootingTotal 19260 non-null int64   
 16 PassingTotal 19260 non-null int64   
 17 DribblingTotal 19260 non-null int64   
 18 DefendingTotal 19260 non-null int64   
 19 PhysicalityTotal 19260 non-null int64   
 20 Crossing 19260 non-null int64   
 21 Finishing 19260 non-null int64   
 22 HeadingAccuracy 19260 non-null int64   
 23 ShortPassing 19260 non-null int64   
 24 Volleys 19260 non-null int64   
 25 Dribbling 19260 non-null int64   
 26 Curve 19260 non-null int64   
 27 FKAccuracy 19260 non-null int64   
 28 LongPassing 19260 non-null int64   
 29 BallControl 19260 non-null int64   
 30 Acceleration 19260 non-null int64   
 31 SprintSpeed 19260 non-null int64   
 32 Agility 19260 non-null int64   
 33 Reactions 19260 non-null int64   
 34 Balance 19260 non-null int64   
 35 ShotPower 19260 non-null int64   
 36 Jumping 19260 non-null int64   
 37 Stamina 19260 non-null int64   
 38 Strength 19260 non-null int64   
 39 LongShots 19260 non-null int64   
 40 Aggression 19260 non-null int64   
 41 Interceptions 19260 non-null int64   
 42 Positioning 19260 non-null int64   
 43 Vision 19260 non-null int64   
 44 Penalties 19260 non-null int64   
 45 Composure 19260 non-null int64   
 46 Marking 19260 non-null int64   
 47 StandingTackle 19260 non-null int64   
 48 SlidingTackle 19260 non-null int64   
 49 GKDiving 19260 non-null int64   
 50 GKHandling 19260 non-null int64   
 51 GKKicking 19260 non-null int64   
 52 GKPositioning 19260 non-null int64   
 53 GKReflexes 19260 non-null int64   
 54 STRating 19260 non-null int64   
 55 LWRating 19260 non-null int64   
 56 LFRating 19260 non-null int64   
 57 CFRating 19260 non-null int64   
 58 RFRating 19260 non-null int64   
 59 RWRating 19260 non-null int64   
 60 CAMRating 19260 non-null int64   
 61 LMRating 19260 non-null int64   
 62 CMRating 19260 non-null int64   
 63 RMRating 19260 non-null int64   
 64 LWBRating 19260 non-null int64   
 65 CDMRating 19260 non-null int64   
 66 RWBRating 19260 non-null int64   
 67 LBRating 19260 non-null int64   
 68 CBRating 19260 non-null int64   
 69 RBRating 19260 non-null int64   
 70 GKRating 19260 non-null int64   
 71 AttackingWorkRate\_code 19260 non-null int64   
 72 DefensiveWorkRate\_code 19260 non-null int64   
 73 BestPosition\_code 19260 non-null int8   
 74 PreferredFoot\_code 19260 non-null int8   
dtypes: int64(72), int8(2), object(1)  
memory usage: 10.8+ MB

# Phase 2

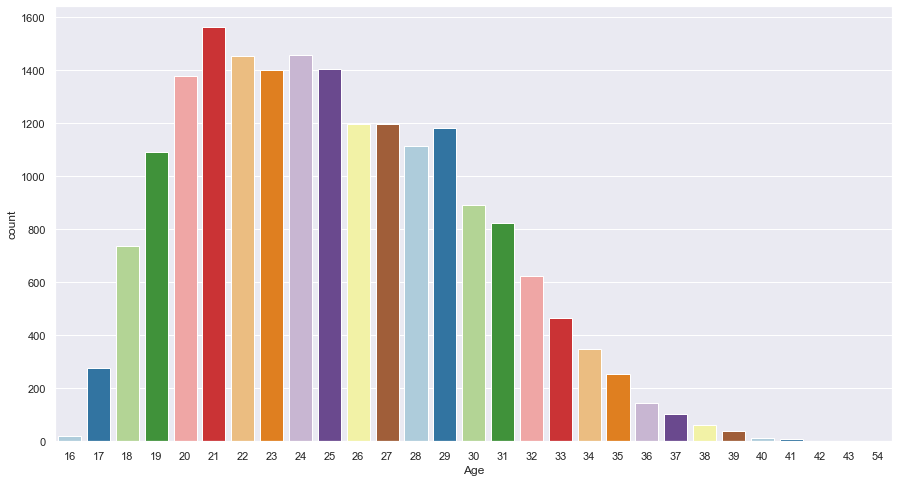
# Exploratory Data Analysis

# 5 number summary of the dataframe  
df\_select.describe()

Age Height Weight Overall Potential \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 25.184683 181.305036 74.950779 65.815628 71.100104   
std 4.737340 6.866151 7.066864 6.817297 6.092103   
min 16.000000 155.000000 49.000000 48.000000 49.000000   
25% 21.000000 176.000000 70.000000 62.000000 67.000000   
50% 25.000000 181.000000 75.000000 66.000000 71.000000   
75% 29.000000 186.000000 80.000000 70.000000 75.000000   
max 54.000000 206.000000 110.000000 93.000000 95.000000   
  
 Growth TotalStats BaseStats ValueEUR WageEUR \  
count 19260.000000 19260.000000 19260.000000 1.926000e+04 19260.000000   
mean 5.284476 1598.525909 357.062461 2.857652e+06 8973.528037   
std 5.472201 271.575855 39.910613 7.604532e+06 19415.497147   
min 0.000000 767.000000 227.000000 0.000000e+00 0.000000   
25% 0.000000 1462.000000 329.000000 4.750000e+05 1000.000000   
50% 4.000000 1633.000000 358.000000 9.750000e+05 3000.000000   
75% 9.000000 1782.000000 384.000000 2.000000e+06 8000.000000   
max 26.000000 2341.000000 501.000000 1.940000e+08 350000.000000   
  
 IntReputation WeakFoot SkillMoves PaceTotal ShootingTotal \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 1.092887 2.946677 2.354517 67.910228 53.535514   
std 0.369317 0.670731 0.767592 10.656453 13.813476   
min 1.000000 1.000000 1.000000 28.000000 18.000000   
25% 1.000000 3.000000 2.000000 62.000000 44.000000   
50% 1.000000 3.000000 2.000000 68.000000 56.000000   
75% 1.000000 3.000000 3.000000 75.000000 64.000000   
max 5.000000 5.000000 5.000000 97.000000 94.000000   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal \  
count 19260.000000 19260.000000 19260.0000 19260.000000   
mean 57.853323 63.028712 50.0581 64.676584   
std 9.835494 9.704853 16.3888 9.626269   
min 25.000000 26.000000 14.0000 29.000000   
25% 52.000000 58.000000 35.0000 58.000000   
50% 58.000000 64.000000 54.0000 66.000000   
75% 65.000000 69.000000 63.0000 72.000000   
max 93.000000 95.000000 91.0000 92.000000   
  
 Crossing Finishing HeadingAccuracy ShortPassing \  
count 19260.000000 19260.000000 19260.000000 19260.000000   
mean 49.642835 45.948390 51.844444 58.925493   
std 17.999983 19.702846 17.276796 14.452595   
min 6.000000 2.000000 5.000000 7.000000   
25% 38.000000 30.000000 44.000000 54.000000   
50% 54.000000 50.000000 55.000000 62.000000   
75% 63.000000 62.000000 64.000000 68.000000   
max 94.000000 95.000000 93.000000 94.000000   
  
 Volleys Dribbling Curve FKAccuracy LongPassing \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 42.497767 55.743873 47.338110 42.305763 53.140654   
std 17.627202 18.751755 18.150707 17.151637 14.990635   
min 3.000000 4.000000 6.000000 4.000000 9.000000   
25% 30.000000 50.000000 35.000000 31.000000 44.000000   
50% 43.000000 61.000000 49.000000 41.000000 56.000000   
75% 56.000000 68.000000 61.000000 55.000000 64.000000   
max 90.000000 96.000000 94.000000 94.000000 93.000000   
  
 BallControl Acceleration SprintSpeed Agility Reactions \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 58.541796 64.703115 64.783333 63.537643 61.489720   
std 16.613390 15.174514 14.969465 14.844798 8.992681   
min 8.000000 14.000000 15.000000 18.000000 25.000000   
25% 55.000000 58.000000 58.000000 55.000000 56.000000   
50% 63.000000 68.000000 68.000000 66.000000 62.000000   
75% 69.000000 75.000000 75.000000 74.000000 67.000000   
max 96.000000 97.000000 97.000000 96.000000 94.000000   
  
 Balance ShotPower Jumping Stamina Strength \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 64.073416 57.830270 64.787539 63.123572 65.034579   
std 14.320474 13.138295 12.120954 16.126570 12.636607   
min 15.000000 20.000000 22.000000 12.000000 19.000000   
25% 56.000000 48.000000 57.000000 56.000000 57.000000   
50% 66.000000 59.000000 65.000000 66.000000 66.000000   
75% 74.000000 68.000000 73.000000 74.000000 74.000000   
max 96.000000 95.000000 95.000000 97.000000 97.000000   
  
 LongShots Aggression Interceptions Positioning Vision \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 46.697664 55.595483 46.665057 50.388422 54.022690   
std 19.376949 16.949958 20.684778 19.595382 13.615786   
min 4.000000 10.000000 3.000000 2.000000 10.000000   
25% 32.000000 44.000000 26.000000 40.000000 45.000000   
50% 51.000000 58.000000 53.000000 56.000000 56.000000   
75% 62.000000 68.000000 64.000000 64.000000 64.000000   
max 94.000000 95.000000 91.000000 96.000000 95.000000   
  
 Penalties Composure Marking StandingTackle \  
count 19260.000000 19260.000000 19260.000000 19260.000000   
mean 47.917342 57.998183 46.648546 48.100779   
std 15.761717 12.107702 20.198663 21.234338   
min 7.000000 12.000000 4.000000 5.000000   
25% 38.000000 51.000000 29.000000 28.000000   
50% 49.000000 59.000000 52.000000 56.000000   
75% 60.000000 66.000000 63.000000 65.000000   
max 93.000000 96.000000 93.000000 93.000000   
  
 SlidingTackle GKDiving GKHandling GKKicking GKPositioning \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 45.948598 16.377882 16.173416 16.028245 16.201038   
std 20.760357 17.548996 16.823264 16.540705 17.035477   
min 5.000000 1.000000 1.000000 1.000000 1.000000   
25% 25.000000 8.000000 8.000000 8.000000 8.000000   
50% 53.000000 11.000000 11.000000 11.000000 11.000000   
75% 63.000000 14.000000 14.000000 14.000000 14.000000   
max 92.000000 91.000000 92.000000 93.000000 92.000000   
  
 GKReflexes STRating LWRating LFRating CFRating \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 16.468432 56.609813 55.729543 55.607425 55.607425   
std 17.870506 13.451570 14.621170 14.218337 14.218337   
min 1.000000 18.000000 14.000000 15.000000 15.000000   
25% 8.000000 51.000000 50.000000 49.000000 49.000000   
50% 11.000000 59.000000 59.000000 59.000000 59.000000   
75% 14.000000 66.000000 65.000000 65.000000 65.000000   
max 90.000000 92.000000 92.000000 93.000000 93.000000   
  
 RFRating RWRating CAMRating LMRating CMRating \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 55.607425 55.729543 57.823053 58.364434 57.227207   
std 14.218337 14.621170 13.934499 13.989045 13.212152   
min 15.000000 14.000000 17.000000 17.000000 17.000000   
25% 49.000000 50.000000 52.000000 54.000000 52.000000   
50% 59.000000 59.000000 61.000000 62.000000 60.000000   
75% 65.000000 65.000000 67.000000 67.000000 66.000000   
max 93.000000 92.000000 93.000000 93.000000 91.000000   
  
 RMRating LWBRating CDMRating RWBRating LBRating \  
count 19260.000000 19260.000000 19260.000000 19260.000000 19260.000000   
mean 58.364434 56.197040 55.771340 56.197040 55.539823   
std 13.989045 13.818416 13.856922 13.818416 14.063862   
min 17.000000 17.000000 18.000000 17.000000 16.000000   
25% 54.000000 51.000000 48.000000 51.000000 49.000000   
50% 62.000000 59.000000 59.000000 59.000000 59.000000   
75% 67.000000 65.000000 66.000000 65.000000 65.000000   
max 93.000000 88.000000 90.000000 88.000000 88.000000   
  
 CBRating RBRating GKRating AttackingWorkRate\_code \  
count 19260.000000 19260.000000 19260.000000 19260.000000   
mean 54.379595 55.539823 23.242939 2.240447   
std 14.678658 14.063862 15.053823 0.531743   
min 18.000000 16.000000 10.000000 1.000000   
25% 44.000000 49.000000 17.000000 2.000000   
50% 58.000000 59.000000 18.000000 2.000000   
75% 66.000000 65.000000 20.000000 3.000000   
max 89.000000 88.000000 92.000000 3.000000   
  
 DefensiveWorkRate\_code BestPosition\_code PreferredFoot\_code   
count 19260.000000 19260.000000 19260.000000   
mean 2.091693 5.839875 0.761682   
std 0.505542 4.939323 0.426066   
min 1.000000 0.000000 0.000000   
25% 2.000000 1.000000 1.000000   
50% 2.000000 5.000000 1.000000   
75% 2.000000 11.000000 1.000000   
max 3.000000 14.000000 1.000000

# Frequency distribution of age of players  
# df['Age'].hist(bins=10, legend= True)  
sns.set(rc = {'figure.figsize':(15,8)})  
sns.countplot(data=df\_select, x='Age', palette='Paired')

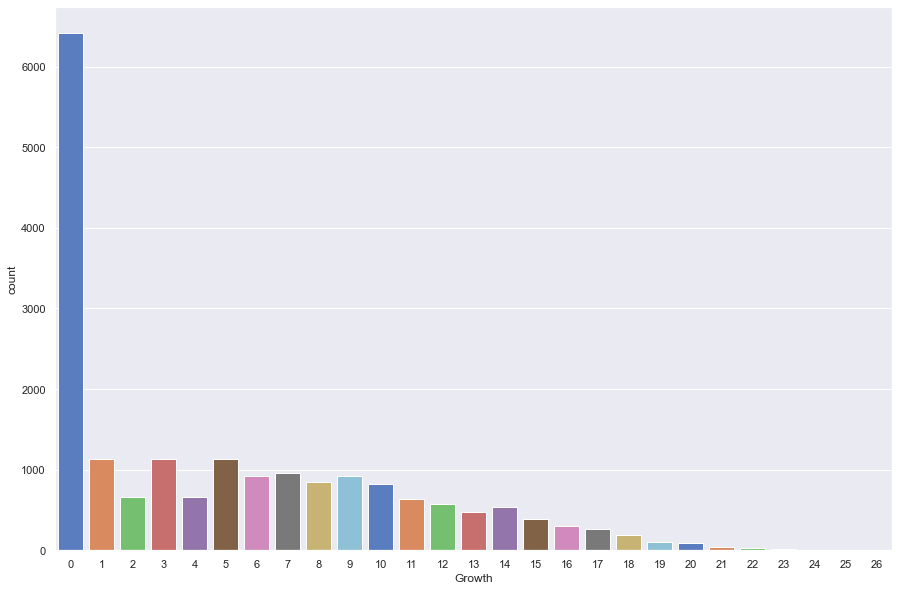
<AxesSubplot:xlabel='Age', ylabel='count'>



Most of the players are from 20-29 of age bucket.

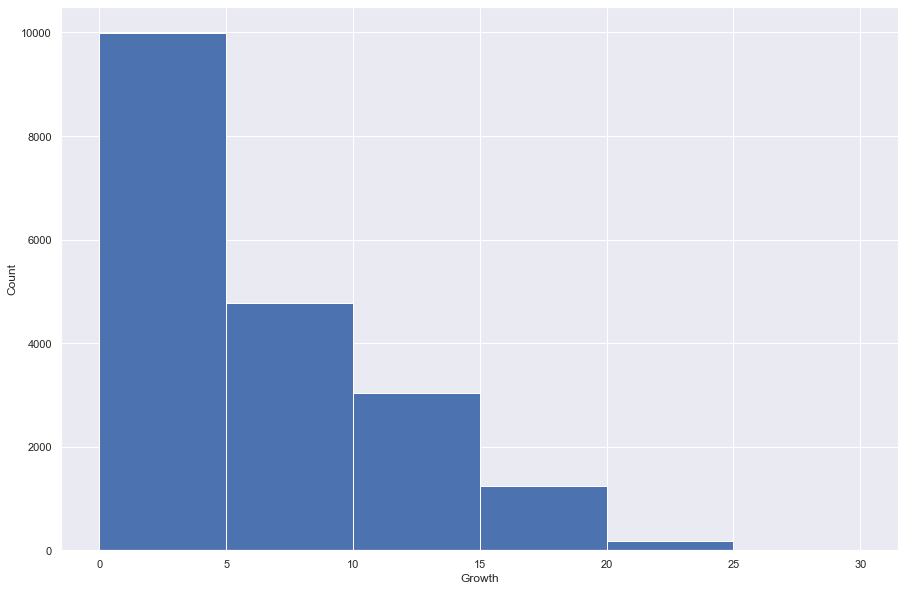
# Frequency distribution of growth of players  
# df['Growth'].hist(bins=4)  
sns.set(rc = {'figure.figsize':(15,10)})  
sns.countplot(data=df\_select, x='Growth', palette='muted')

<AxesSubplot:xlabel='Growth', ylabel='count'>



ax = df['Growth'].hist(bins=[0.,5.,10.,15.,20.,25.,30.])  
ax.set\_xlabel('Growth')  
ax.set\_ylabel('Count')

Text(0, 0.5, 'Count')



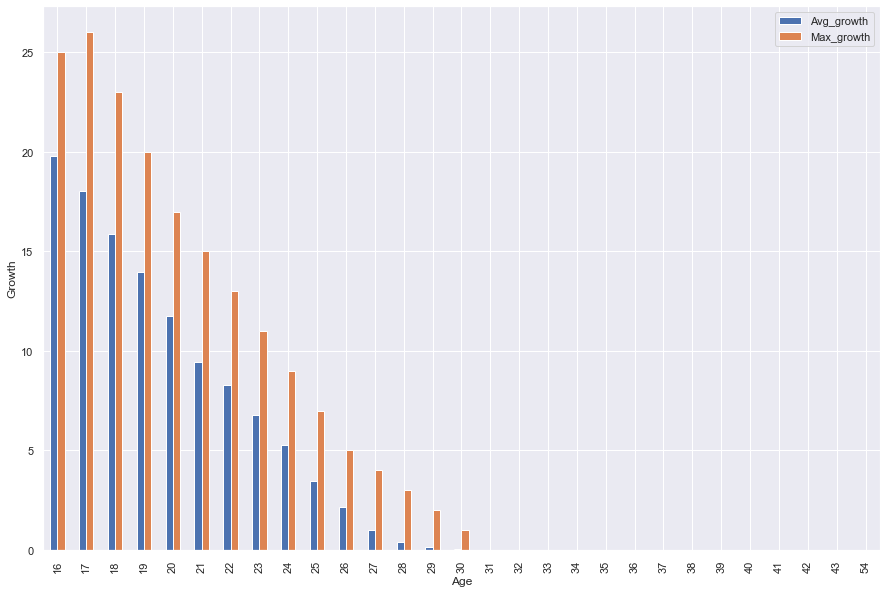
We see a good number of players with exceptional growths (>10).

# Frequency of Growth values   
df\_select.groupby('Growth', as\_index= False).agg(count= ('Age', 'count'))

Growth count  
0 0 6423  
1 1 1132  
2 2 656  
3 3 1127  
4 4 659  
5 5 1138  
6 6 923  
7 7 956  
8 8 843  
9 9 924  
10 10 817  
11 11 640  
12 12 571  
13 13 471  
14 14 540  
15 15 383  
16 16 302  
17 17 265  
18 18 194  
19 19 106  
20 20 96  
21 21 36  
22 22 27  
23 23 19  
24 24 4  
25 25 6  
26 26 2

# Age vs Average and max Growth  
ax = df\_select.groupby('Age', as\_index =True).agg(Avg\_growth = ('Growth', 'mean'),\  
 Max\_growth = ('Growth', 'max')).plot(kind= 'bar')  
ax.set\_ylabel('Growth',fontsize=12)

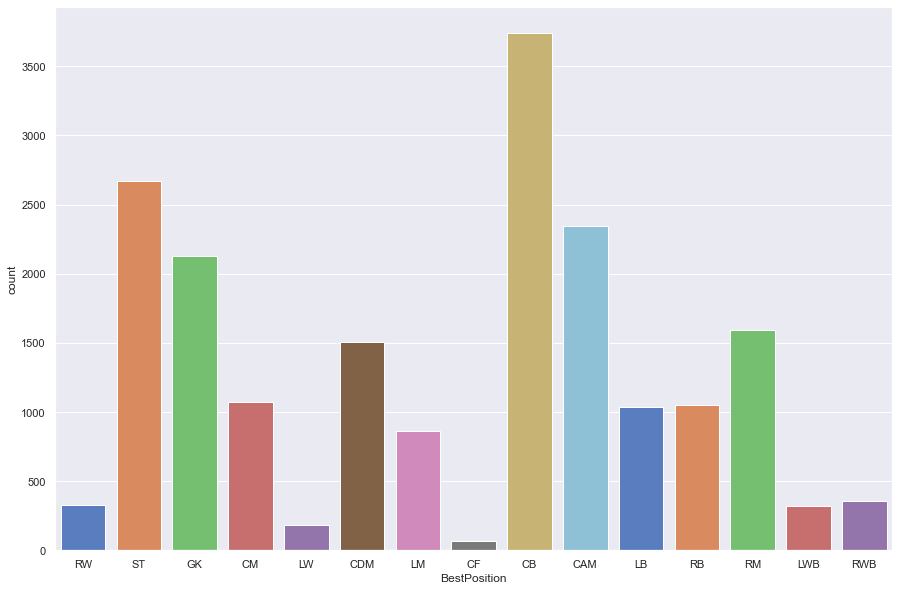
Text(0, 0.5, 'Growth')



As players grow older, their growth rate declines.

# Frequency distribution of player positions  
# df['BestPosition'].hist()  
# df['BestPosition'].value\_counts().plot(kind='bar')  
sns.set(rc = {'figure.figsize':(15,10)})  
sns.countplot(data=df\_select, x='BestPosition', palette='muted')

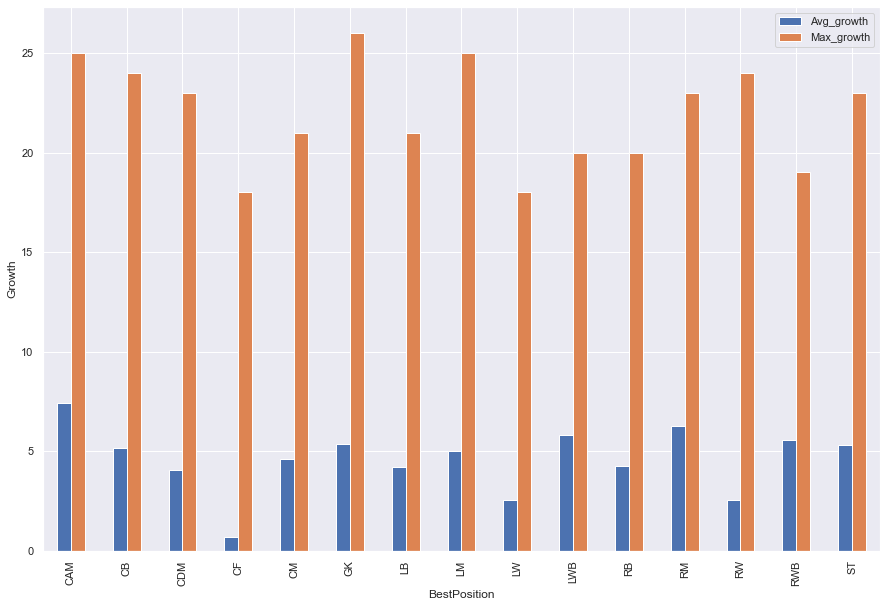
<AxesSubplot:xlabel='BestPosition', ylabel='count'>



We have a lot of CBs (Centre Back) in our dataset.

# Player position vs average and max growth  
ax = df\_select.groupby('BestPosition', as\_index =True).agg(Avg\_growth = ('Growth', 'mean'),\  
 Max\_growth = ('Growth', 'max')).plot(kind= 'bar')  
ax.set\_ylabel('Growth',fontsize=12)

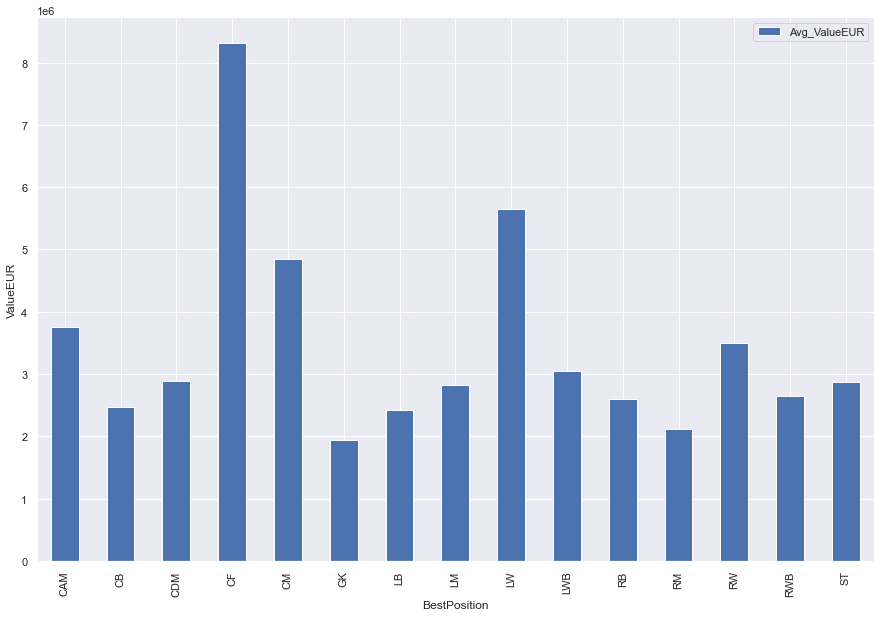
Text(0, 0.5, 'Growth')



GK has the highest growth but CAM has highest average growth.

# Player position vs ValueEUR  
ax = df\_select.groupby('BestPosition', as\_index =True).agg(Avg\_ValueEUR = ('ValueEUR', 'mean')).plot(kind= 'bar')  
ax.set\_ylabel('ValueEUR',fontsize=12)

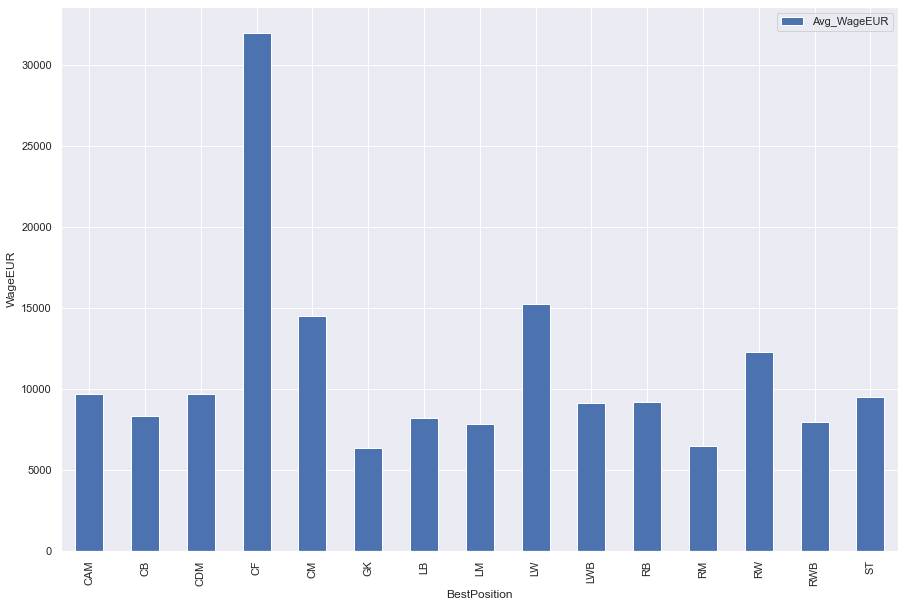
Text(0, 0.5, 'ValueEUR')



CF (Center Forward) & LW (Left Winger) tend to have higher values (Euro in millions) in the transfer market.

# Player position vs ValueEUR  
ax = df\_select.groupby('BestPosition', as\_index =True).agg(Avg\_WageEUR = ('WageEUR', 'mean')).plot(kind= 'bar')  
ax.set\_ylabel('WageEUR',fontsize=12)

Text(0, 0.5, 'WageEUR')



CF (Center Forward) & LW (Left Winger) tend to have higher wages as well.

# Problem statement

1. Does position have an impact on the wage and the valuation of a player?
2. Does growth have an impact on the wage and the valuation of a player?

# create correlation matrix with abs values  
corr\_matrix = df\_select.corr().abs()  
  
# change this value as needed, if 0.5 does not work for your scenario  
# threshold = 0.5  
  
filtered\_corr\_df = corr\_matrix[corr\_matrix != 1.000]   
  
unstacked = filtered\_corr\_df.unstack()  
unstacked\_ordered = unstacked.sort\_values(kind="quicksort", ascending=False)  
# unstacked\_ordered

# getting all combination of correlation between 2 keys with one being WageEUR  
corr\_df = pd.DataFrame(unstacked\_ordered).reset\_index().rename(columns = {'level\_0': 'Key1',\  
 'level\_1': 'Key2',\  
 0:'correlation'})  
corr\_df.dropna(how='any', inplace = True)  
corr\_df.reset\_index(drop = True, inplace=True)  
wage\_corr\_df = corr\_df[corr\_df['Key1'] == 'WageEUR']  
wage\_corr\_df.reset\_index(drop = True, inplace=True)  
wage\_corr\_df

Key1 Key2 correlation  
0 WageEUR ValueEUR 0.824979  
1 WageEUR IntReputation 0.670785  
2 WageEUR Overall 0.602329  
3 WageEUR Reactions 0.537883  
4 WageEUR Potential 0.499201  
5 WageEUR BaseStats 0.495028  
6 WageEUR Composure 0.434046  
7 WageEUR PassingTotal 0.426854  
8 WageEUR DribblingTotal 0.408923  
9 WageEUR TotalStats 0.374751  
10 WageEUR Vision 0.343533  
11 WageEUR CMRating 0.343061  
12 WageEUR ShotPower 0.340414  
13 WageEUR ShortPassing 0.322276  
14 WageEUR CAMRating 0.321911  
15 WageEUR STRating 0.314023  
16 WageEUR LFRating 0.310471  
17 WageEUR CFRating 0.310471  
18 WageEUR RFRating 0.310471  
19 WageEUR LMRating 0.309212  
20 WageEUR RMRating 0.309212  
21 WageEUR LongPassing 0.307845  
22 WageEUR ShootingTotal 0.297540  
23 WageEUR RWRating 0.293602  
24 WageEUR LWRating 0.293602  
25 WageEUR BallControl 0.291170  
26 WageEUR CDMRating 0.284743  
27 WageEUR Curve 0.278216  
28 WageEUR LWBRating 0.272329  
29 WageEUR RWBRating 0.272329  
30 WageEUR PhysicalityTotal 0.270739  
31 WageEUR SkillMoves 0.269309  
32 WageEUR Volleys 0.260614  
33 WageEUR RBRating 0.254528  
34 WageEUR LBRating 0.254528  
35 WageEUR LongShots 0.253849  
36 WageEUR Dribbling 0.253092  
37 WageEUR Crossing 0.250120  
38 WageEUR FKAccuracy 0.241446  
39 WageEUR Positioning 0.237452  
40 WageEUR CBRating 0.229234  
41 WageEUR Finishing 0.226275  
42 WageEUR Penalties 0.223623  
43 WageEUR Aggression 0.222687  
44 WageEUR HeadingAccuracy 0.205377  
45 WageEUR Stamina 0.199091  
46 WageEUR Growth 0.194634  
47 WageEUR DefendingTotal 0.190003  
48 WageEUR Agility 0.170241  
49 WageEUR Interceptions 0.169380  
50 WageEUR Marking 0.168146  
51 WageEUR PaceTotal 0.165141  
52 WageEUR Strength 0.160202  
53 WageEUR WeakFoot 0.159085  
54 WageEUR Age 0.151408  
55 WageEUR Jumping 0.147808  
56 WageEUR StandingTackle 0.143743  
57 WageEUR SprintSpeed 0.143169  
58 WageEUR Acceleration 0.133823  
59 WageEUR SlidingTackle 0.124969  
60 WageEUR Balance 0.105382  
61 WageEUR AttackingWorkRate\_code 0.101702  
62 WageEUR DefensiveWorkRate\_code 0.065593  
63 WageEUR Weight 0.063072  
64 WageEUR Height 0.027479  
65 WageEUR GKHandling 0.024182  
66 WageEUR GKKicking 0.024135  
67 WageEUR GKDiving 0.024103  
68 WageEUR GKPositioning 0.022690  
69 WageEUR GKReflexes 0.022481  
70 WageEUR GKRating 0.016229  
71 WageEUR PreferredFoot\_code 0.016147  
72 WageEUR BestPosition\_code 0.008704

# getting all combination of correlation between 2 keys with one being ValueEUR  
value\_corr\_df = corr\_df[corr\_df['Key1'] == 'ValueEUR']  
value\_corr\_df.reset\_index(drop = True, inplace=True)  
value\_corr\_df

Key1 Key2 correlation  
0 ValueEUR WageEUR 0.824979  
1 ValueEUR IntReputation 0.628230  
2 ValueEUR Overall 0.558037  
3 ValueEUR Potential 0.530576  
4 ValueEUR Reactions 0.493774  
5 ValueEUR BaseStats 0.463984  
6 ValueEUR PassingTotal 0.394191  
7 ValueEUR DribblingTotal 0.392375  
8 ValueEUR Composure 0.388428  
9 ValueEUR TotalStats 0.341679  
10 ValueEUR Vision 0.323786  
11 ValueEUR CMRating 0.317754  
12 ValueEUR CAMRating 0.304598  
13 ValueEUR ShotPower 0.299847  
14 ValueEUR ShortPassing 0.296960  
15 ValueEUR RMRating 0.294239  
16 ValueEUR LMRating 0.294239  
17 ValueEUR STRating 0.292632  
18 ValueEUR CFRating 0.292169  
19 ValueEUR RFRating 0.292169  
20 ValueEUR LFRating 0.292169  
21 ValueEUR LWRating 0.278812  
22 ValueEUR RWRating 0.278812  
23 ValueEUR LongPassing 0.277933  
24 ValueEUR ShootingTotal 0.277138  
25 ValueEUR BallControl 0.269438  
26 ValueEUR SkillMoves 0.256397  
27 ValueEUR CDMRating 0.254077  
28 ValueEUR Curve 0.251420  
29 ValueEUR LWBRating 0.249041  
30 ValueEUR RWBRating 0.249041  
31 ValueEUR Dribbling 0.240478  
32 ValueEUR Volleys 0.236613  
33 ValueEUR PhysicalityTotal 0.234197  
34 ValueEUR LBRating 0.230243  
35 ValueEUR RBRating 0.230243  
36 ValueEUR LongShots 0.229662  
37 ValueEUR Crossing 0.226071  
38 ValueEUR Positioning 0.222664  
39 ValueEUR Finishing 0.216133  
40 ValueEUR FKAccuracy 0.212114  
41 ValueEUR PaceTotal 0.205499  
42 ValueEUR Stamina 0.198764  
43 ValueEUR CBRating 0.196846  
44 ValueEUR Penalties 0.193610  
45 ValueEUR Aggression 0.181915  
46 ValueEUR Agility 0.177040  
47 ValueEUR SprintSpeed 0.165735  
48 ValueEUR HeadingAccuracy 0.165315  
49 ValueEUR Acceleration 0.158558  
50 ValueEUR DefendingTotal 0.156223  
51 ValueEUR WeakFoot 0.149022  
52 ValueEUR Marking 0.140441  
53 ValueEUR Interceptions 0.140331  
54 ValueEUR Strength 0.127401  
55 ValueEUR AttackingWorkRate\_code 0.123208  
56 ValueEUR StandingTackle 0.120377  
57 ValueEUR Jumping 0.118496  
58 ValueEUR Balance 0.118239  
59 ValueEUR Growth 0.104524  
60 ValueEUR SlidingTackle 0.103240  
61 ValueEUR DefensiveWorkRate\_code 0.057552  
62 ValueEUR Age 0.037133  
63 ValueEUR Weight 0.032599  
64 ValueEUR GKHandling 0.022524  
65 ValueEUR GKDiving 0.022224  
66 ValueEUR GKKicking 0.021602  
67 ValueEUR GKPositioning 0.021259  
68 ValueEUR GKReflexes 0.019900  
69 ValueEUR PreferredFoot\_code 0.019094  
70 ValueEUR BestPosition\_code 0.018958  
71 ValueEUR GKRating 0.015651  
72 ValueEUR Height 0.009591

# getting all combination of correlation between 2 keys with one being BestPosition\_code  
BestPosition\_corr\_df = corr\_df[corr\_df['Key1'] == 'BestPosition\_code']  
BestPosition\_corr\_df.reset\_index(drop = True, inplace=True)  
BestPosition\_corr\_df

Key1 Key2 correlation  
0 BestPosition\_code DefendingTotal 0.482821  
1 BestPosition\_code SlidingTackle 0.418813  
2 BestPosition\_code StandingTackle 0.416399  
3 BestPosition\_code Interceptions 0.408409  
4 BestPosition\_code Marking 0.392481  
5 BestPosition\_code Finishing 0.369563  
6 BestPosition\_code PaceTotal 0.365673  
7 BestPosition\_code ShootingTotal 0.339736  
8 BestPosition\_code Positioning 0.330035  
9 BestPosition\_code CBRating 0.303644  
10 BestPosition\_code Volleys 0.302053  
11 BestPosition\_code SprintSpeed 0.301283  
12 BestPosition\_code AttackingWorkRate\_code 0.295487  
13 BestPosition\_code Acceleration 0.286119  
14 BestPosition\_code Penalties 0.275848  
15 BestPosition\_code CDMRating 0.262487  
16 BestPosition\_code STRating 0.249707  
17 BestPosition\_code DefensiveWorkRate\_code 0.226640  
18 BestPosition\_code CFRating 0.226365  
19 BestPosition\_code RFRating 0.226365  
20 BestPosition\_code LFRating 0.226365  
21 BestPosition\_code LongShots 0.222136  
22 BestPosition\_code LWRating 0.220163  
23 BestPosition\_code RWRating 0.220163  
24 BestPosition\_code ShotPower 0.219100  
25 BestPosition\_code Agility 0.216811  
26 BestPosition\_code DribblingTotal 0.208491  
27 BestPosition\_code LongPassing 0.207885  
28 BestPosition\_code LBRating 0.205361  
29 BestPosition\_code RBRating 0.205361  
30 BestPosition\_code Dribbling 0.203432  
31 BestPosition\_code LMRating 0.173673  
32 BestPosition\_code RMRating 0.173673  
33 BestPosition\_code CAMRating 0.170418  
34 BestPosition\_code Aggression 0.158368  
35 BestPosition\_code LWBRating 0.157213  
36 BestPosition\_code RWBRating 0.157213  
37 BestPosition\_code SkillMoves 0.152405  
38 BestPosition\_code Curve 0.146029  
39 BestPosition\_code Crossing 0.141186  
40 BestPosition\_code Balance 0.127267  
41 BestPosition\_code BallControl 0.110914  
42 BestPosition\_code Height 0.107372  
43 BestPosition\_code WeakFoot 0.094752  
44 BestPosition\_code PhysicalityTotal 0.094734  
45 BestPosition\_code PreferredFoot\_code 0.076956  
46 BestPosition\_code Vision 0.070641  
47 BestPosition\_code Weight 0.064060  
48 BestPosition\_code Stamina 0.063230  
49 BestPosition\_code GKRating 0.060492  
50 BestPosition\_code GKPositioning 0.059145  
51 BestPosition\_code GKHandling 0.058693  
52 BestPosition\_code GKKicking 0.058391  
53 BestPosition\_code GKDiving 0.057362  
54 BestPosition\_code HeadingAccuracy 0.057160  
55 BestPosition\_code GKReflexes 0.056905  
56 BestPosition\_code PassingTotal 0.056332  
57 BestPosition\_code Potential 0.055129  
58 BestPosition\_code ShortPassing 0.052926  
59 BestPosition\_code Strength 0.047594  
60 BestPosition\_code TotalStats 0.047511  
61 BestPosition\_code Growth 0.037903  
62 BestPosition\_code FKAccuracy 0.036872  
63 BestPosition\_code BaseStats 0.030925  
64 BestPosition\_code IntReputation 0.023100  
65 BestPosition\_code Reactions 0.020704  
66 BestPosition\_code ValueEUR 0.018958  
67 BestPosition\_code Overall 0.018840  
68 BestPosition\_code Jumping 0.016927  
69 BestPosition\_code WageEUR 0.008704  
70 BestPosition\_code CMRating 0.003652  
71 BestPosition\_code Age 0.003598  
72 BestPosition\_code Composure 0.000046

So position has little to no impact on wage or valuation of a player.

# getting all combination of correlation between 2 keys with one being Growth  
Growth\_corr\_df = corr\_df[corr\_df['Key1'] == 'Growth']  
Growth\_corr\_df.reset\_index(drop = True, inplace=True)  
Growth\_corr\_df

Key1 Key2 correlation  
0 Growth Age 0.864210  
1 Growth Overall 0.526821  
2 Growth Reactions 0.501124  
3 Growth PhysicalityTotal 0.482302  
4 Growth BaseStats 0.468586  
5 Growth Composure 0.404850  
6 Growth PassingTotal 0.388168  
7 Growth Strength 0.369169  
8 Growth TotalStats 0.348054  
9 Growth ShotPower 0.327930  
10 Growth Potential 0.308712  
11 Growth Aggression 0.283428  
12 Growth ShootingTotal 0.281019  
13 Growth DribblingTotal 0.271337  
14 Growth CDMRating 0.269338  
15 Growth CMRating 0.265851  
16 Growth Vision 0.263840  
17 Growth LongPassing 0.256168  
18 Growth Stamina 0.255554  
19 Growth Jumping 0.245802  
20 Growth DefendingTotal 0.244727  
21 Growth CBRating 0.244409  
22 Growth FKAccuracy 0.241492  
23 Growth LongShots 0.239659  
24 Growth LWBRating 0.236504  
25 Growth RWBRating 0.236504  
26 Growth STRating 0.235027  
27 Growth ShortPassing 0.233228  
28 Growth RBRating 0.229215  
29 Growth LBRating 0.229215  
30 Growth Crossing 0.226592  
31 Growth Curve 0.221525  
32 Growth Interceptions 0.217350  
33 Growth CFRating 0.216324  
34 Growth LFRating 0.216324  
35 Growth RFRating 0.216324  
36 Growth HeadingAccuracy 0.214374  
37 Growth CAMRating 0.211864  
38 Growth Volleys 0.211339  
39 Growth Marking 0.207238  
40 Growth Weight 0.203145  
41 Growth RMRating 0.201865  
42 Growth LMRating 0.201865  
43 Growth Penalties 0.196805  
44 Growth LWRating 0.195027  
45 Growth RWRating 0.195027  
46 Growth WageEUR 0.194634  
47 Growth IntReputation 0.191124  
48 Growth Positioning 0.185731  
49 Growth BallControl 0.185175  
50 Growth Finishing 0.163790  
51 Growth SkillMoves 0.150940  
52 Growth StandingTackle 0.147415  
53 Growth SlidingTackle 0.132259  
54 Growth Dribbling 0.130111  
55 Growth ValueEUR 0.104524  
56 Growth WeakFoot 0.103269  
57 Growth Agility 0.086267  
58 Growth DefensiveWorkRate\_code 0.071274  
59 Growth GKRating 0.067714  
60 Growth Height 0.045978  
61 Growth AttackingWorkRate\_code 0.043887  
62 Growth GKPositioning 0.039918  
63 Growth BestPosition\_code 0.037903  
64 Growth GKReflexes 0.035209  
65 Growth GKHandling 0.034499  
66 Growth GKKicking 0.034228  
67 Growth GKDiving 0.032391  
68 Growth PaceTotal 0.026740  
69 Growth Acceleration 0.017616  
70 Growth PreferredFoot\_code 0.012867  
71 Growth SprintSpeed 0.005887  
72 Growth Balance 0.002023

df\_select[['Growth', 'Age']].corr()

Growth Age  
Growth 1.00000 -0.86421  
Age -0.86421 1.00000

So, Growth might have little impact on wage or valuation of a player. But it is strongly & negatively correlated with Age.

# Phase 3

We are looking for a model that can predict the best position for a player based on all the necessary features.

# Feature selection

# Selecting X and Y axis features  
X = df\_select.drop(['BestPosition','BestPosition\_code'], axis = 1)  
Y = df\_select['BestPosition\_code']  
# feature extraction  
test = SelectKBest(score\_func=f\_classif, k=30)  
fit = test.fit(X, Y)  
# summarize scores  
set\_printoptions(precision=5)  
# print(fit.scores\_)  
features = fit.transform(X)  
# summarize selected features  
# print(features[0:5,:])

# Shape of input array  
X.shape

(19260, 73)

# Shape of output array of feature selection  
features.shape

(19260, 30)

# Displaying feature importance/ scores as a dataframe  
dfscores = pd.DataFrame(fit.scores\_)  
dfcolumns = pd.DataFrame(X.columns)  
#concat two dataframes for better visualization   
featureScores = pd.concat([dfcolumns,dfscores],axis=1)  
featureScores.columns = ['Specs','Score']  
featureScores

Specs Score  
0 Age 67.190212  
1 Height 912.986303  
2 Weight 681.242937  
3 Overall 29.575567  
4 Potential 35.112654  
5 Growth 56.691974  
6 TotalStats 1989.576990  
7 BaseStats 215.095352  
8 ValueEUR 16.498515  
9 WageEUR 20.929972  
10 IntReputation 15.422596  
11 WeakFoot 169.077617  
12 SkillMoves 1686.724575  
13 PaceTotal 703.732257  
14 ShootingTotal 1795.707416  
15 PassingTotal 626.694386  
16 DribblingTotal 882.266955  
17 DefendingTotal 3099.881707  
18 PhysicalityTotal 495.471704  
19 Crossing 2881.065313  
20 Finishing 4586.887667  
21 HeadingAccuracy 4063.744509  
22 ShortPassing 2828.735534  
23 Volleys 2686.880900  
24 Dribbling 5306.344167  
25 Curve 2031.415516  
26 FKAccuracy 1492.046657  
27 LongPassing 1828.268246  
28 BallControl 4560.246415  
29 Acceleration 2253.688848  
30 SprintSpeed 1992.088156  
31 Agility 1896.555687  
32 Reactions 80.777536  
33 Balance 1425.678708  
34 ShotPower 700.035703  
35 Jumping 320.049905  
36 Stamina 2052.864481  
37 Strength 678.973971  
38 LongShots 2726.496657  
39 Aggression 1654.619595  
40 Interceptions 4168.014537  
41 Positioning 4847.117368  
42 Vision 1241.076822  
43 Penalties 2465.137659  
44 Composure 550.788135  
45 Marking 4548.897318  
46 StandingTackle 5314.139306  
47 SlidingTackle 5543.957641  
48 GKDiving 28268.224351  
49 GKHandling 27010.305107  
50 GKKicking 25348.547488  
51 GKPositioning 23830.911432  
52 GKReflexes 27603.770319  
53 STRating 4098.748788  
54 LWRating 4960.968868  
55 LFRating 4391.133437  
56 CFRating 4391.133437  
57 RFRating 4391.133437  
58 RWRating 4960.968868  
59 CAMRating 4105.480358  
60 LMRating 4580.326100  
61 CMRating 3268.056334  
62 RMRating 4580.326100  
63 LWBRating 4834.525933  
64 CDMRating 3871.580282  
65 RWBRating 4834.525933  
66 LBRating 5085.841395  
67 CBRating 4639.361814  
68 RBRating 5085.841395  
69 GKRating 33939.827637  
70 AttackingWorkRate\_code 356.053992  
71 DefensiveWorkRate\_code 265.263679  
72 PreferredFoot\_code 594.286433

# Taking the top 30 features according to their scores for modeling  
feature\_list = featureScores.sort\_values(by = 'Score',\  
 ascending = False).head(30)['Specs'].tolist()  
# feature\_list

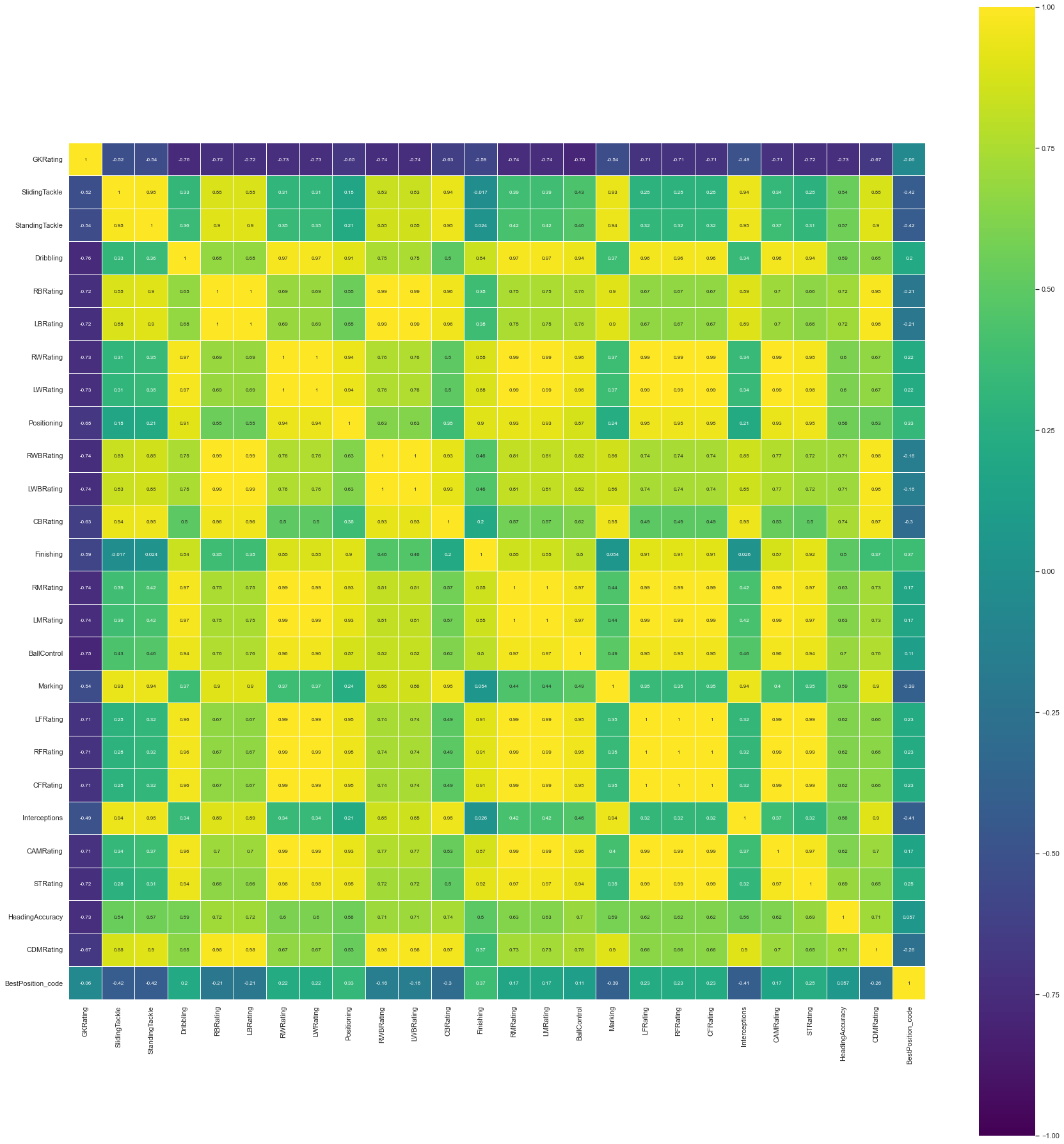
# Adding the calculated field needed which will be the predicted variable of our model to the list   
feature\_list.append('BestPosition\_code')  
# remove features which are too similar  
feature\_list.remove('GKDiving')  
feature\_list.remove('GKReflexes')  
feature\_list.remove('GKHandling')  
feature\_list.remove('GKKicking')  
feature\_list.remove('GKPositioning')  
feature\_list

['GKRating',  
 'SlidingTackle',  
 'StandingTackle',  
 'Dribbling',  
 'RBRating',  
 'LBRating',  
 'RWRating',  
 'LWRating',  
 'Positioning',  
 'RWBRating',  
 'LWBRating',  
 'CBRating',  
 'Finishing',  
 'RMRating',  
 'LMRating',  
 'BallControl',  
 'Marking',  
 'LFRating',  
 'RFRating',  
 'CFRating',  
 'Interceptions',  
 'CAMRating',  
 'STRating',  
 'HeadingAccuracy',  
 'CDMRating',  
 'BestPosition\_code']

final\_df = df\_select[feature\_list]  
final\_df.head()

GKRating SlidingTackle StandingTackle Dribbling RBRating LBRating \  
0 22 24 35 96 64 64   
1 22 19 42 85 64 64   
2 23 24 32 88 63 63   
3 21 32 34 93 66 66   
4 92 18 12 12 35 35   
  
 RWRating LWRating Positioning RWBRating LWBRating CBRating Finishing \  
0 92 92 93 69 69 53 95   
1 85 85 95 67 67 63 95   
2 88 88 95 66 66 56 95   
3 90 90 92 70 70 57 93   
4 32 32 11 35 35 36 11   
  
 RMRating LMRating BallControl Marking LFRating RFRating CFRating \  
0 93 93 96 20 93 93 93   
1 87 87 88 35 88 88 88   
2 89 89 88 24 89 89 89   
3 92 92 91 26 90 90 90   
4 38 38 30 27 35 35 35   
  
 Interceptions CAMRating STRating HeadingAccuracy CDMRating \  
0 40 93 92 70 67   
1 49 89 92 90 69   
2 29 89 91 90 62   
3 38 92 91 72 66   
4 19 41 36 15 39   
  
 BestPosition\_code   
0 12   
1 14   
2 14   
3 14   
4 5

# correlation heatmap  
corr = final\_df.corr()  
plt.figure(figsize=(25, 25))  
  
sns.heatmap(corr  
# [(corr >= 0.6) | (corr <= -0.6)]  
 ,   
 cmap='viridis', vmax=1.0, vmin=-1.0, linewidths=0.1,  
 annot=True, annot\_kws={"size": 8}, square=True);  
plt.tight\_layout()  
# plt.savefig('corr\_heatmap\_all.png', facecolor='w')

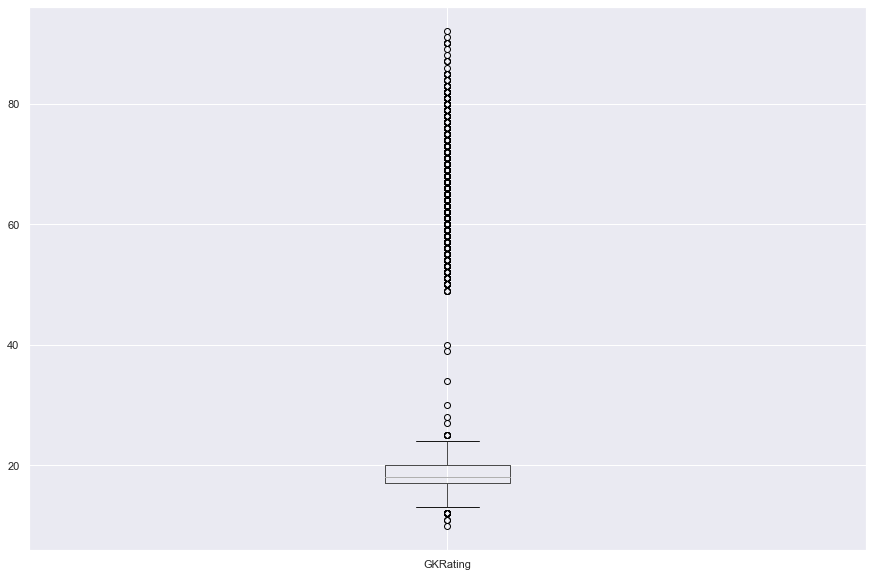


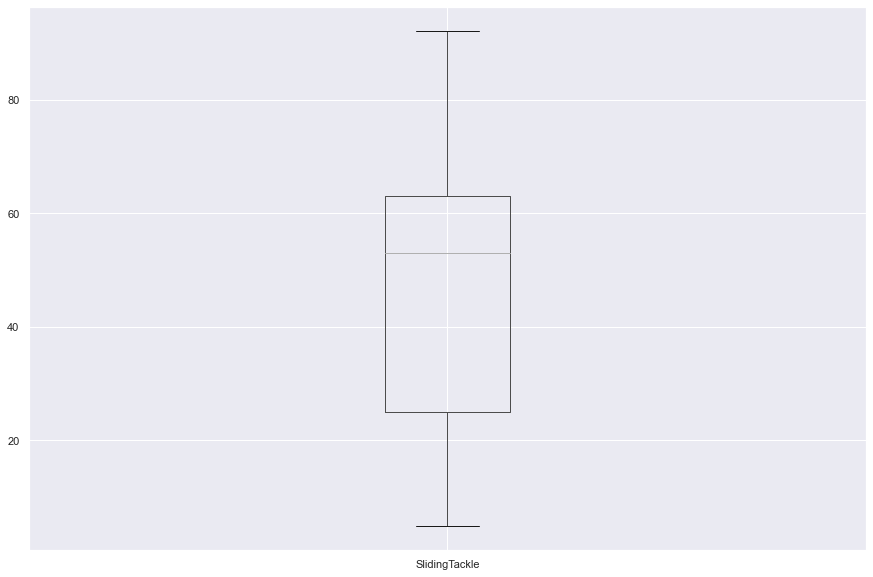
As 'BestPosition\_code' is a categorical variable, we do not see any strong correlations with other ratio variables.

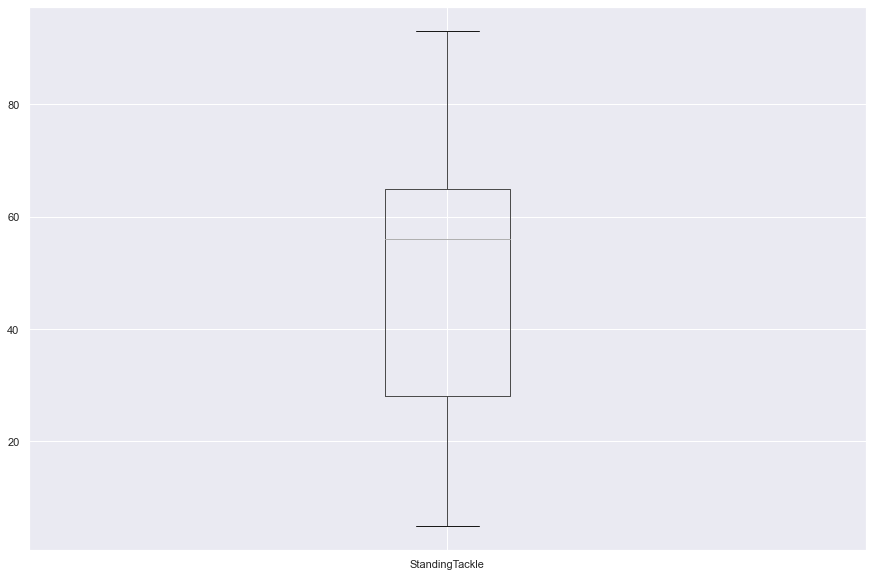
# Outlier detection

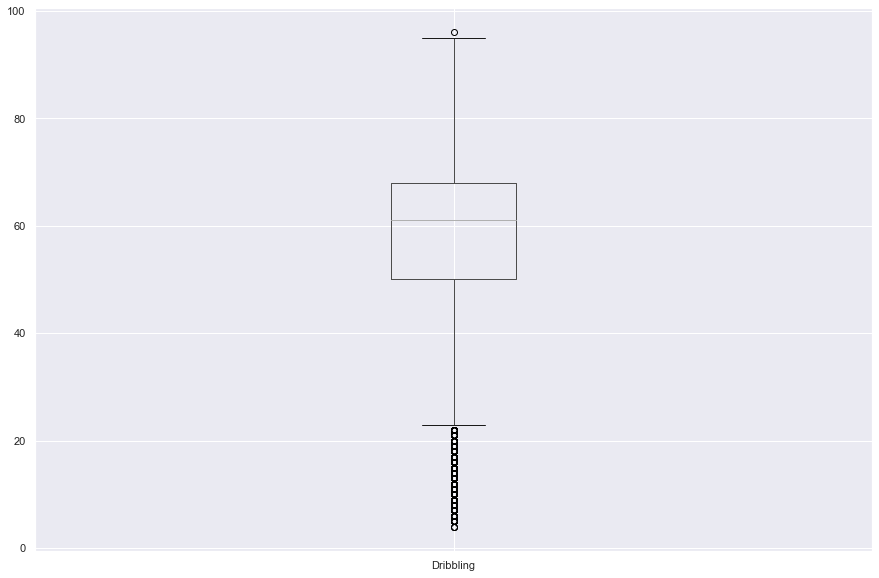
## Boxplots

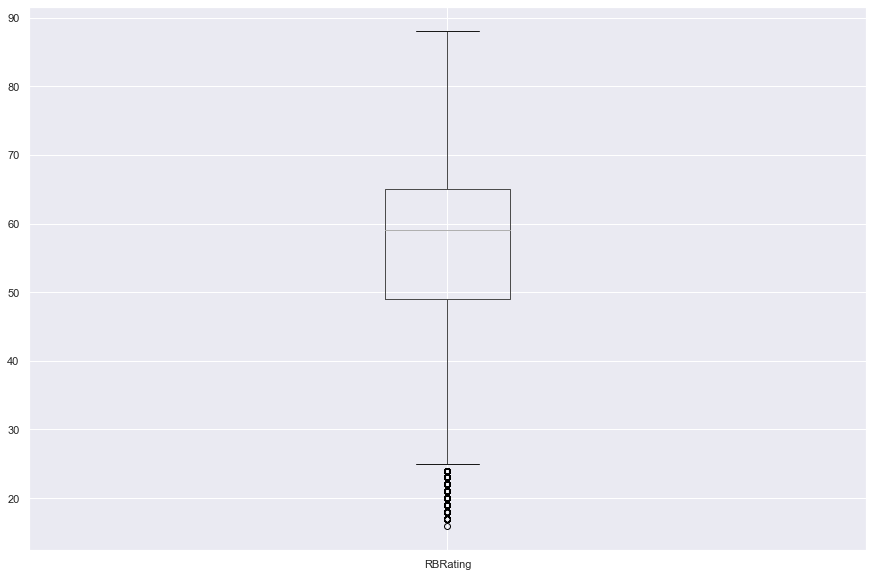
for column in final\_df:   
 if column == 'BestPosition\_code':  
 break  
 final\_df.boxplot([column])  
 plt.show()

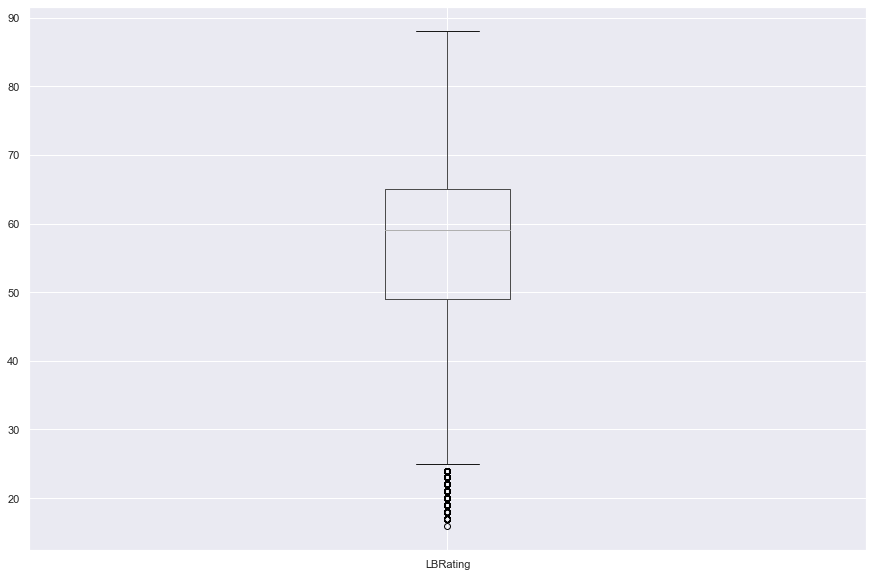


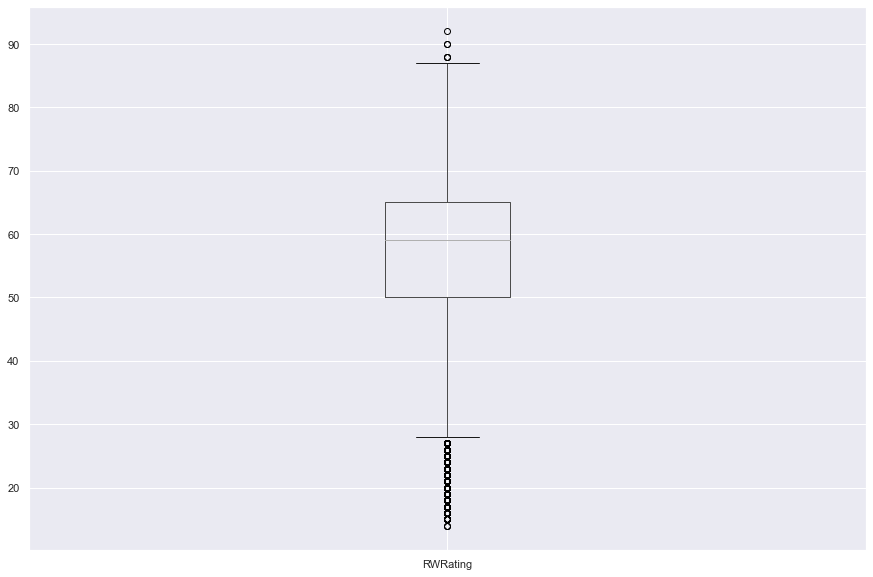


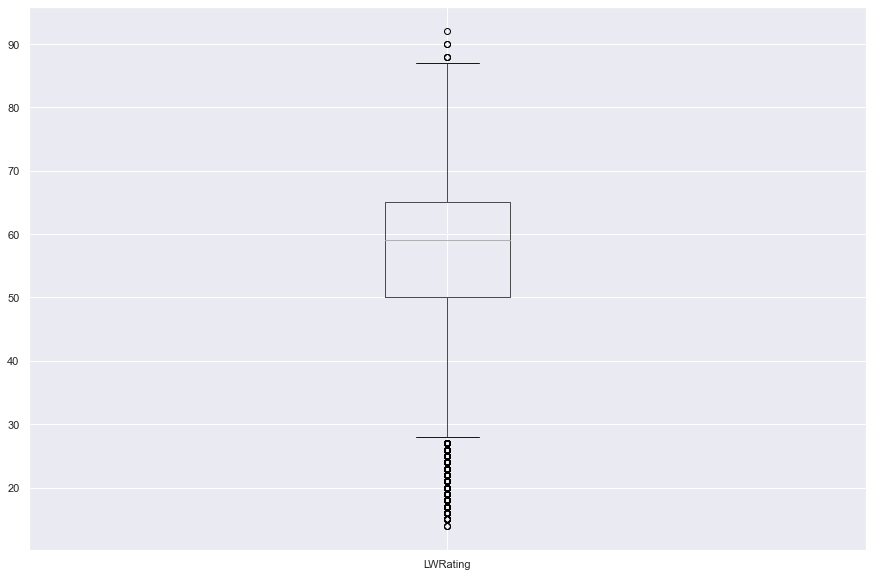


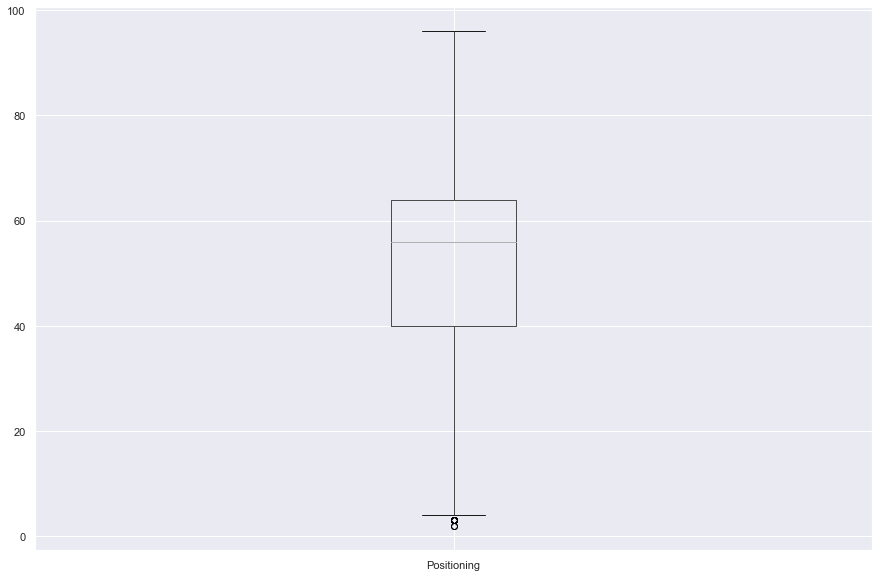


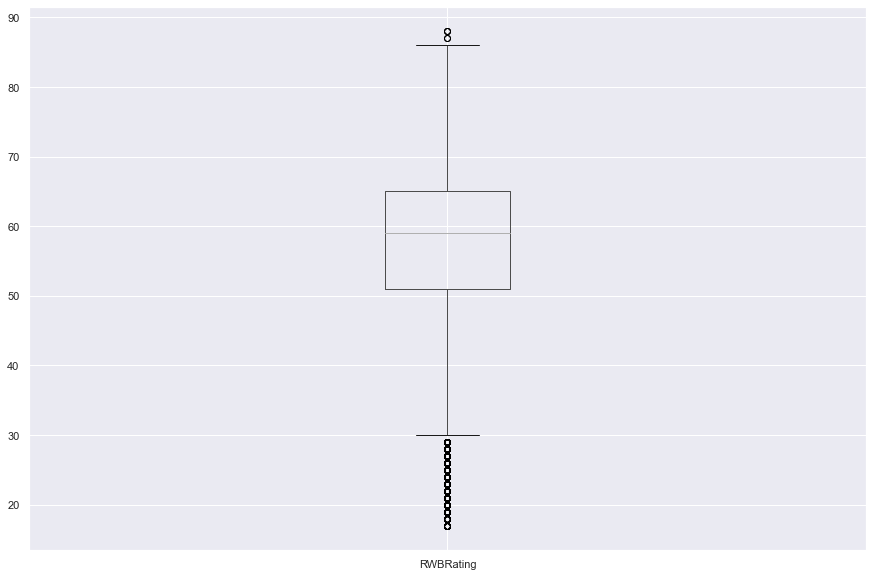


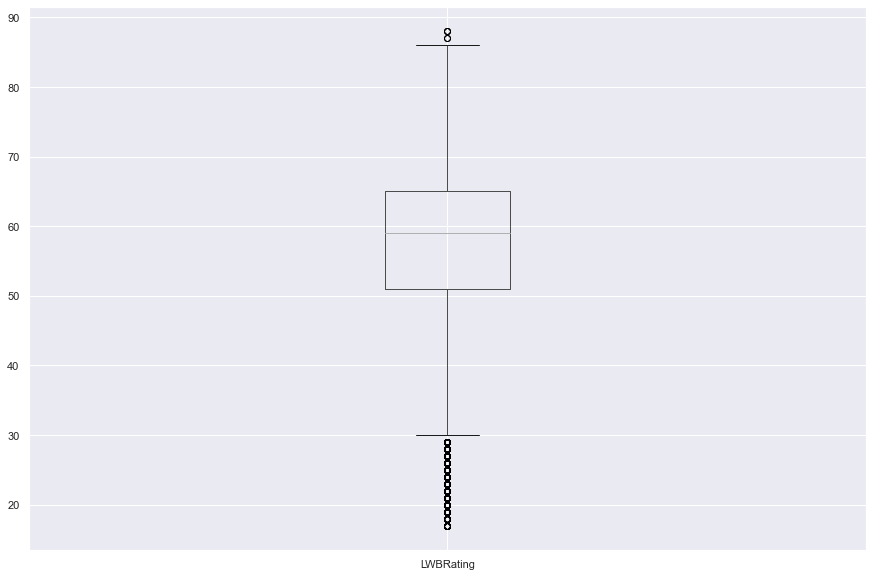


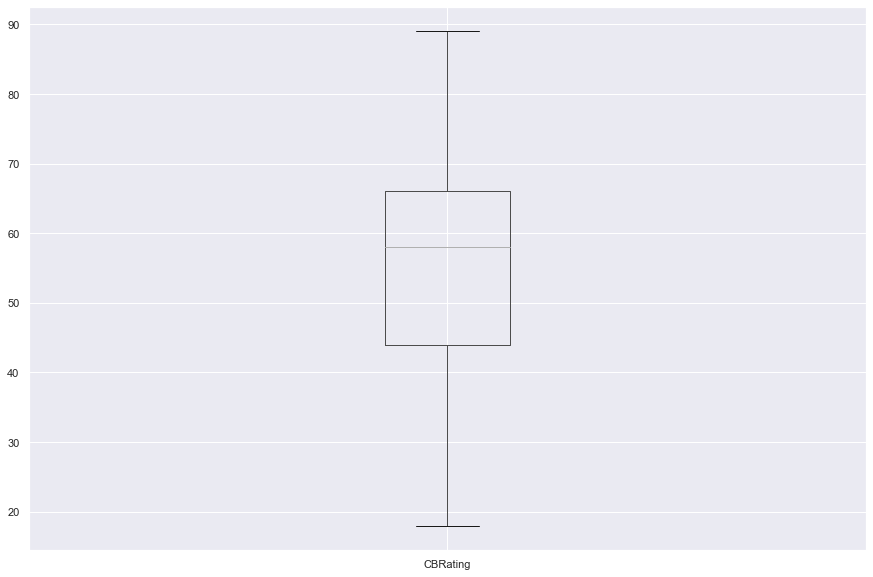


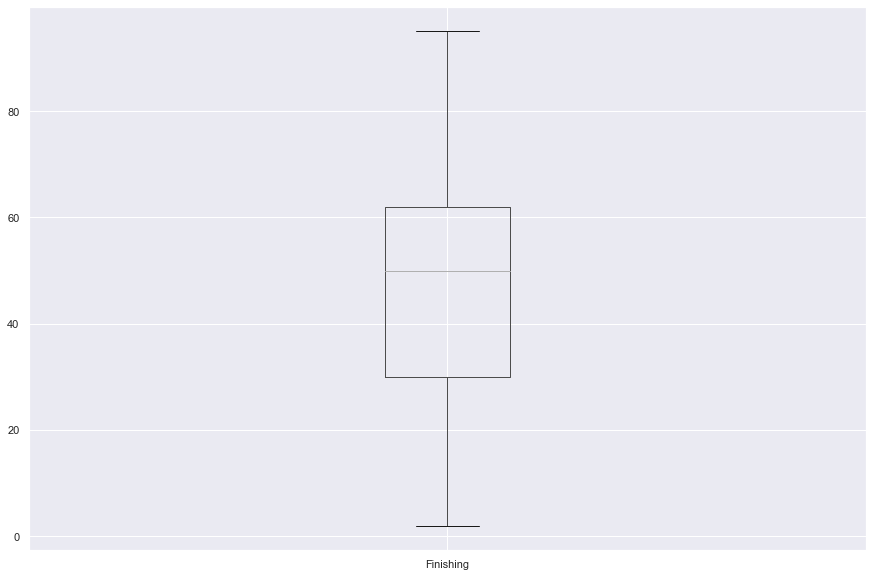




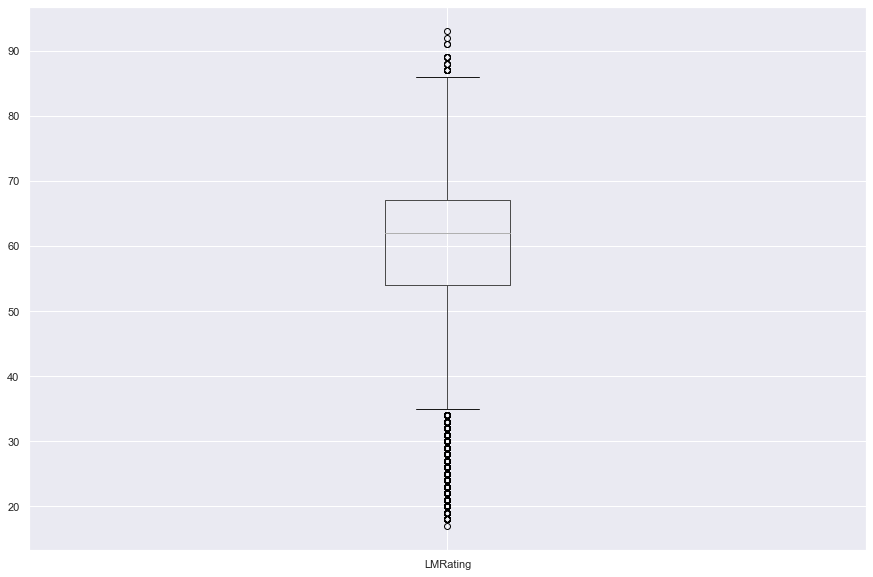


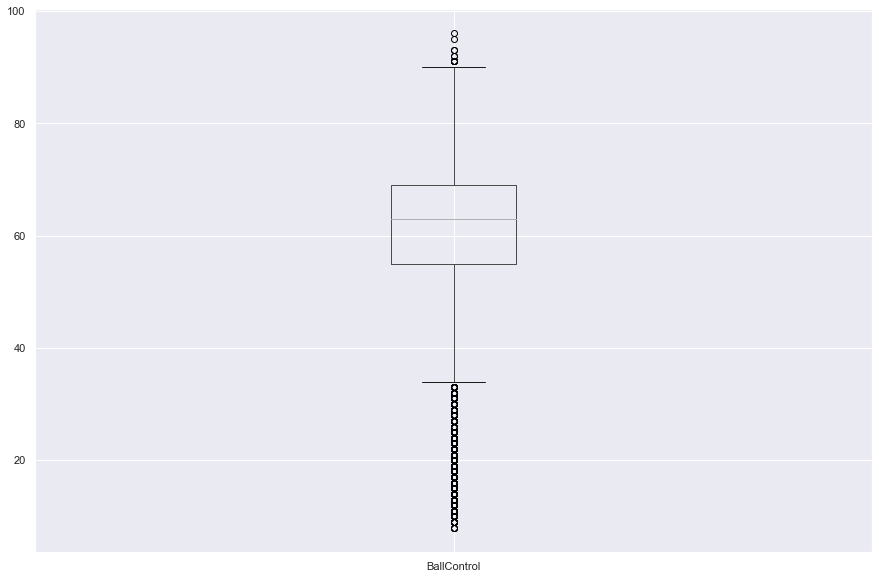


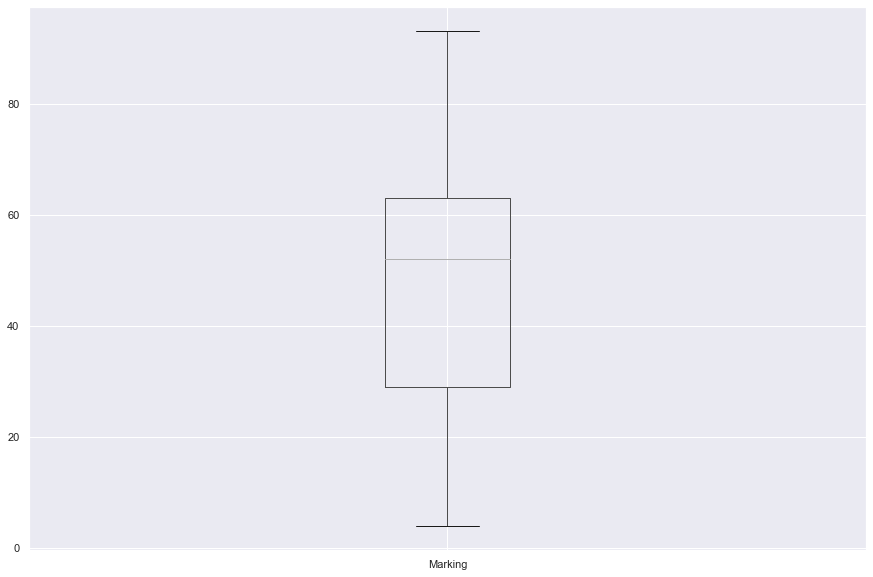


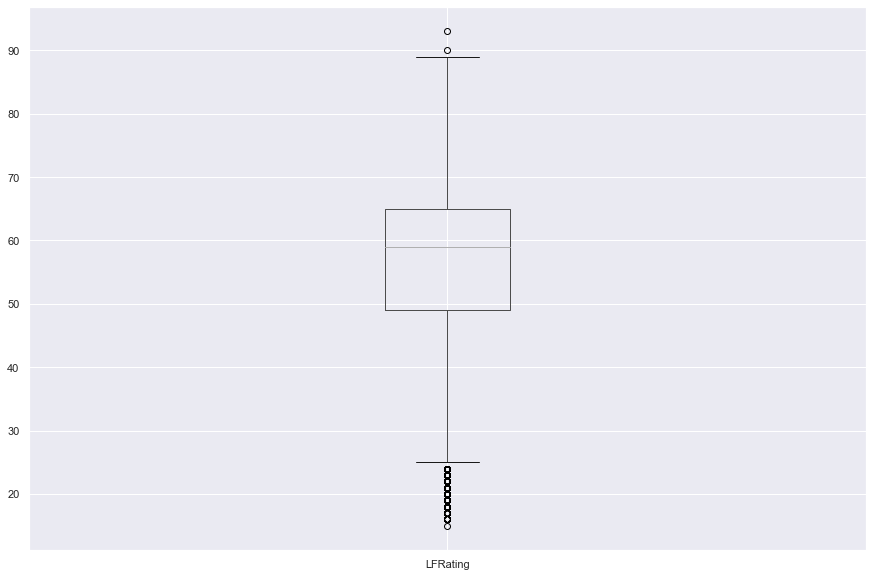


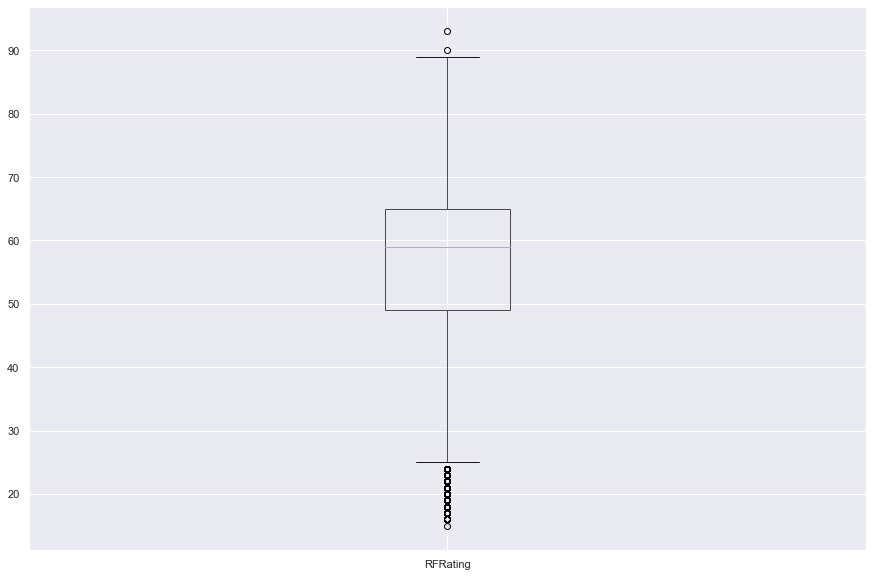


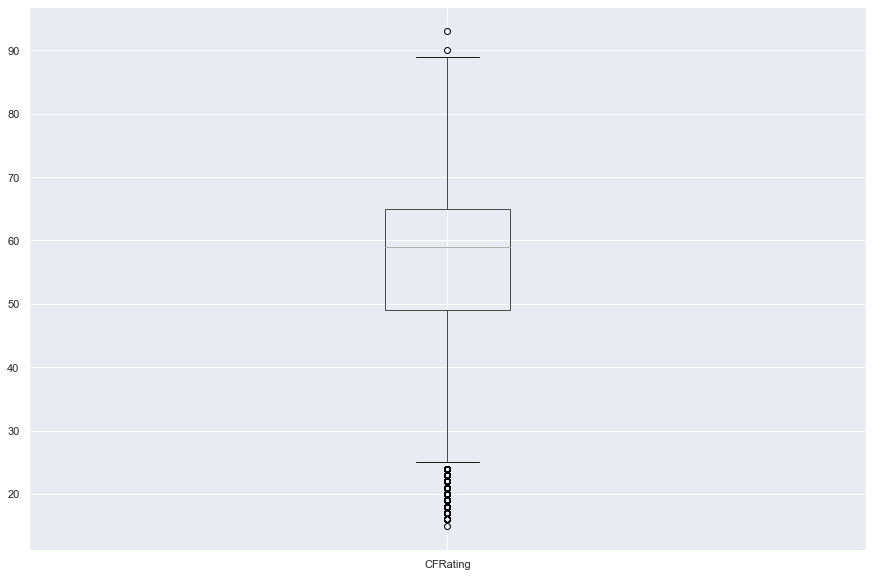


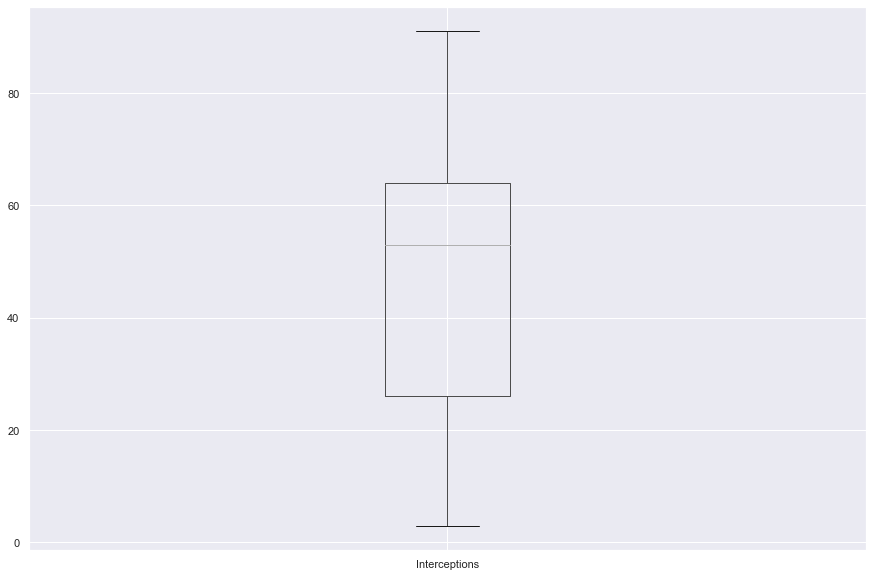


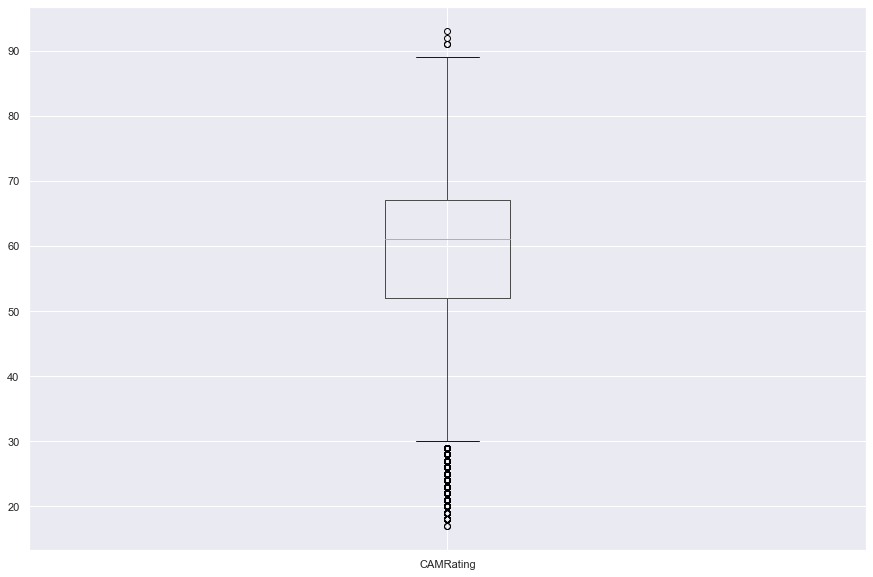


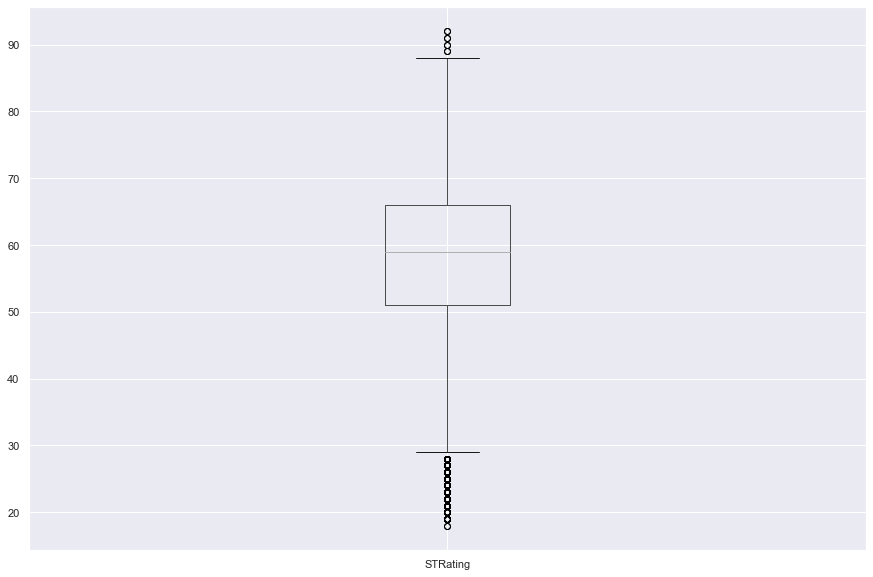


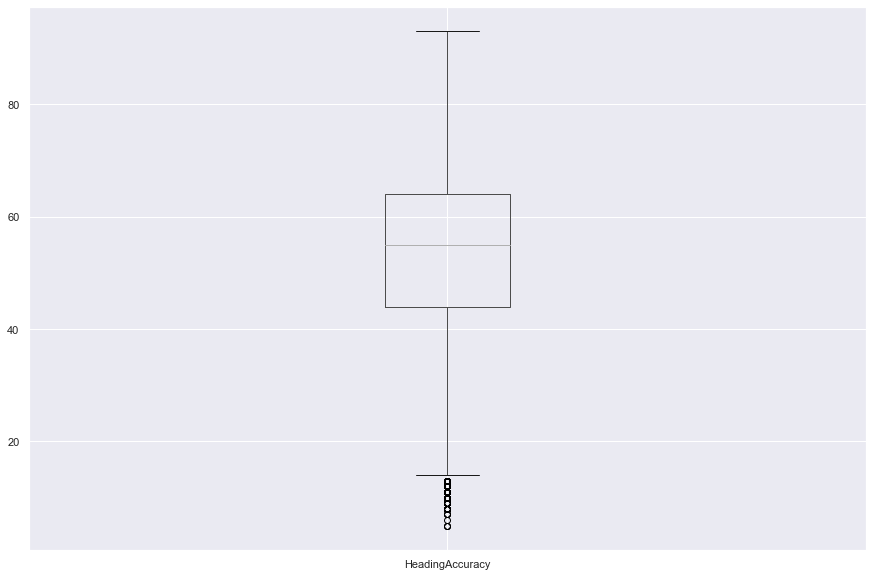


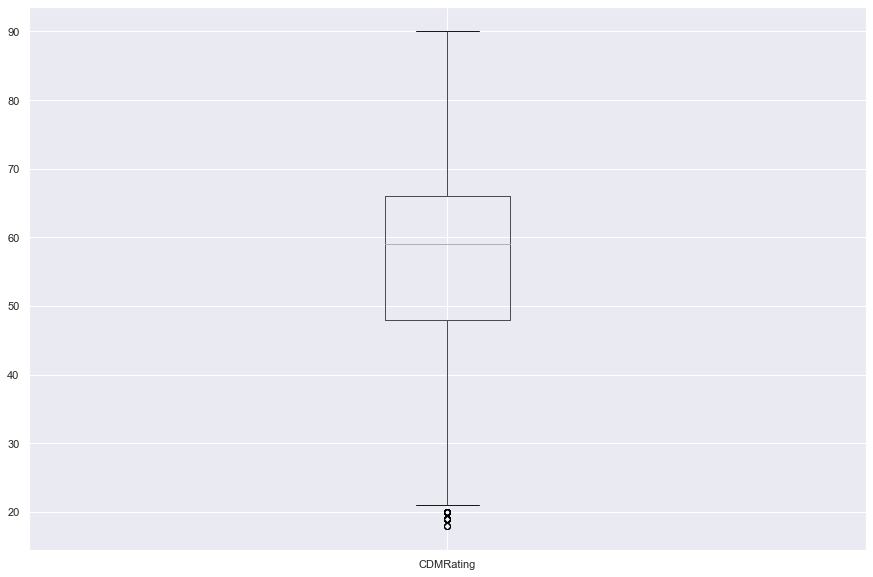










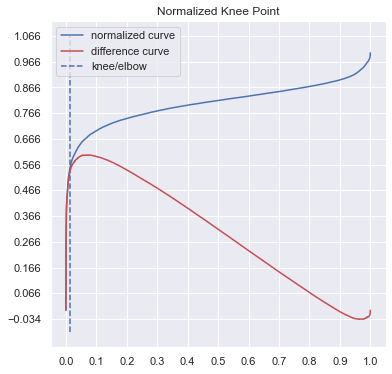


## DBSCAN

# Scaling the feature data   
X\_scaled = StandardScaler().fit\_transform(final\_df.drop(['BestPosition\_code'],\  
 axis=1).values)  
# As we have 26 columns, using >D+1 rule  
neighbors = 27  
  
nbrs = NearestNeighbors(n\_neighbors=neighbors).fit(X\_scaled)  
distances, indices = nbrs.kneighbors(X\_scaled)  
distance\_desc = sorted(distances[:,-1], reverse=True)  
#px.line(x=list(range(1,len(distance\_desc )+1)),y= distance\_desc)

from kneed import KneeLocator  
kneedle = KneeLocator(range(1,len(distance\_desc)+1), #x values  
 distance\_desc, # y values  
 S=1.0,   
 curve="convex",   
 direction="decreasing")

kneedle.plot\_knee\_normalized()



kneedle.knee\_y

1.4080565595824033

According to the graph above,value of eps should be 1.4080565595824033.

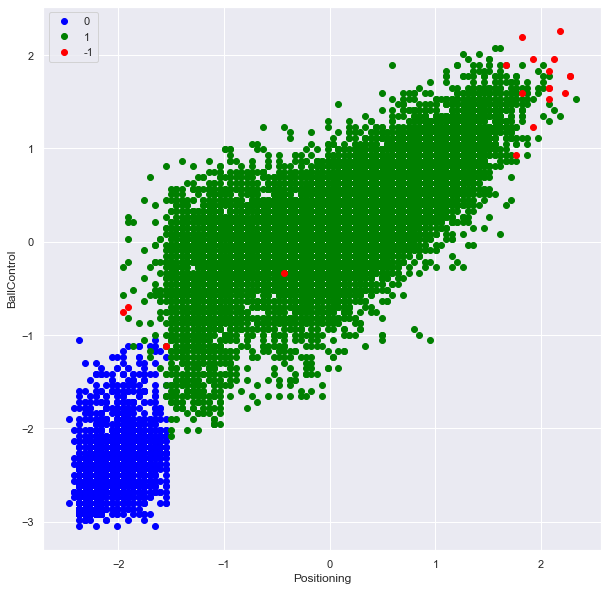
from sklearn.cluster import DBSCAN  
db = DBSCAN(eps=1.4080565595824033, min\_samples=27).fit(X\_scaled)  
labels = db.labels\_

pd.Series(labels).value\_counts()

1 17118  
 0 2123  
-1 19  
dtype: int64

plt.figure(figsize=(10,10))  
  
unique\_labels = set(labels)  
colors = ['blue','green', 'red']  
  
for color,label in zip(colors, unique\_labels):  
 sample\_mask = [True if l == label else False for l in labels]  
 plt.plot(X\_scaled[:,8][sample\_mask], X\_scaled[:, 15][sample\_mask], 'o', color=color);  
plt.xlabel('Positioning');  
plt.ylabel('BallControl');  
plt.gca().legend(('0','1','-1'))

<matplotlib.legend.Legend at 0x115948e1100>



According to DBSCAN method, we have 19 outliers in the dataset. We will leave them for now as they are not that drastic.

# Display the outliers  
df['Outlier'] = labels  
df[df['Outlier'] == -1].reset\_index(drop=True)

Name FullName Age Height Weight \  
0 L. Messi Lionel Messi 34 170 72   
1 R. Lewandowski Robert Lewandowski 32 185 81   
2 Cristiano Ronaldo C. Ronaldo dos Santos Aveiro 36 187 83   
3 K. Mbappé Kylian Mbappé 22 182 73   
4 K. De Bruyne Kevin De Bruyne 30 181 70   
5 Neymar Jr Neymar da Silva Santos Jr. 29 175 68   
6 H. Kane Harry Kane 27 188 89   
7 M. Neuer Manuel Neuer 35 193 93   
8 H. Son Heung Min Son 28 183 78   
9 M. Salah Mohamed Salah 29 175 71   
10 Ederson Ederson Santana de Moraes 27 188 86   
11 L. Suárez Luis Suárez 34 182 83   
12 R. Sterling Raheem Sterling 26 170 69   
13 J. Sancho Jadon Sancho 21 180 76   
14 L. Martínez Lautaro Martínez 23 174 72   
15 W. Weghorst Wout Weghorst 28 197 84   
16 B. Dost Bas Dost 32 196 83   
17 M. Riemann Manuel Riemann 32 186 83   
18 Jaimosa Cavaco Jaime Frederico Cavaco Dorta 21 168 60   
  
 PhotoUrl Nationality Overall \  
0 https://cdn.sofifa.com/players/158/023/22\_60.png Argentina 93   
1 https://cdn.sofifa.com/players/188/545/22\_60.png Poland 92   
2 https://cdn.sofifa.com/players/020/801/22\_60.png Portugal 91   
3 https://cdn.sofifa.com/players/231/747/22\_60.png France 91   
4 https://cdn.sofifa.com/players/192/985/22\_60.png Belgium 91   
5 https://cdn.sofifa.com/players/190/871/22\_60.png Brazil 91   
6 https://cdn.sofifa.com/players/202/126/22\_60.png England 90   
7 https://cdn.sofifa.com/players/167/495/22\_60.png Germany 90   
8 https://cdn.sofifa.com/players/200/104/22\_60.png Korea Republic 89   
9 https://cdn.sofifa.com/players/209/331/22\_60.png Egypt 89   
10 https://cdn.sofifa.com/players/210/257/22\_60.png Brazil 89   
11 https://cdn.sofifa.com/players/176/580/22\_60.png Uruguay 88   
12 https://cdn.sofifa.com/players/202/652/22\_60.png England 88   
13 https://cdn.sofifa.com/players/233/049/22\_60.png England 87   
14 https://cdn.sofifa.com/players/231/478/22\_60.png Argentina 85   
15 https://cdn.sofifa.com/players/223/689/22\_60.png Netherlands 83   
16 https://cdn.sofifa.com/players/189/068/22\_60.png Netherlands 77   
17 https://cdn.sofifa.com/players/199/339/22\_60.png Germany 74   
18 https://cdn.sofifa.com/players/230/194/22\_60.png Brazil 70   
  
 Potential Growth TotalStats BaseStats Positions BestPosition \  
0 93 0 2219 462 RW,ST,CF RW   
1 92 0 2212 460 ST ST   
2 91 0 2208 457 ST,LW ST   
3 95 4 2175 470 ST,LW ST   
4 91 0 2304 485 CM,CAM CM   
5 91 0 2183 454 LW,CAM LW   
6 90 0 2205 456 ST ST   
7 90 0 1534 501 GK GK   
8 89 0 2142 455 LM,CF,LW LM   
9 89 0 2211 468 RW RW   
10 91 2 1583 501 GK GK   
11 88 0 2307 457 ST ST   
12 89 1 2113 451 LW,RW LW   
13 91 4 2007 431 RM,CF,LM CAM   
14 89 4 2145 454 ST ST   
15 83 0 1919 412 ST ST   
16 77 0 1752 371 ST ST   
17 74 0 1434 414 GK GK   
18 70 0 1694 355 RB RB   
  
 Club ValueEUR WageEUR ReleaseClause ClubPosition \  
0 Paris Saint-Germain 78000000 320000 144300000 RW   
1 FC Bayern München 119500000 270000 197200000 ST   
2 Manchester United 45000000 270000 83300000 ST   
3 Paris Saint-Germain 194000000 230000 373500000 ST   
4 Manchester City 125500000 350000 232200000 CM   
5 Paris Saint-Germain 129000000 270000 238700000 LW   
6 Tottenham Hotspur 129500000 240000 246100000 ST   
7 FC Bayern München 13500000 86000 22300000 GK   
8 Tottenham Hotspur 104000000 220000 197600000 LW   
9 Liverpool 101000000 270000 186900000 RW   
10 Manchester City 94000000 200000 181000000 GK   
11 Atlético de Madrid 44500000 135000 91200000 ST   
12 Manchester City 107500000 290000 206900000 SUB   
13 Manchester United 116500000 150000 224300000 LM   
14 Inter 78000000 145000 138500000 ST   
15 VfL Wolfsburg 37000000 95000 62900000 ST   
16 Club Brugge KV 8500000 29000 11900000 SUB   
17 VfL Bochum 1848 2100000 16000 3800000 GK   
18 Juventude 1700000 9000 3200000 SUB   
  
 ContractUntil ClubNumber ClubJoined OnLoad NationalTeam \  
0 2023.0 30.0 2021 False Argentina   
1 2023.0 9.0 2014 False Poland   
2 2023.0 7.0 2021 False Portugal   
3 2022.0 7.0 2018 False France   
4 2025.0 17.0 2015 False Belgium   
5 2025.0 10.0 2017 False Not in team   
6 2024.0 10.0 2010 False England   
7 2023.0 1.0 2011 False Germany   
8 2025.0 7.0 2015 False Not in team   
9 2023.0 11.0 2017 False Not in team   
10 2026.0 31.0 2017 False Not in team   
11 2022.0 9.0 2020 False Not in team   
12 2023.0 7.0 2015 False England   
13 2026.0 25.0 2021 False England   
14 2023.0 10.0 2018 False Argentina   
15 2023.0 9.0 2018 False Netherlands   
16 2022.0 29.0 2021 False Not in team   
17 2023.0 1.0 2015 False Not in team   
18 2024.0 3.0 2021 False Not in team   
  
 NationalPosition NationalNumber PreferredFoot IntReputation WeakFoot \  
0 RW 10.0 Left 5 4   
1 ST 9.0 Right 5 4   
2 ST 7.0 Right 5 4   
3 LW 10.0 Right 4 4   
4 CM 7.0 Right 4 5   
5 NaN NaN Right 5 5   
6 ST 9.0 Right 4 5   
7 GK 1.0 Right 5 4   
8 NaN NaN Right 4 5   
9 NaN NaN Left 4 3   
10 NaN NaN Left 3 3   
11 NaN NaN Right 5 4   
12 LW 10.0 Right 4 3   
13 SUB 17.0 Right 3 3   
14 ST 22.0 Right 3 4   
15 SUB 19.0 Right 2 3   
16 NaN NaN Right 3 3   
17 NaN NaN Right 1 5   
18 NaN NaN Right 1 3   
  
 SkillMoves AttackingWorkRate DefensiveWorkRate PaceTotal ShootingTotal \  
0 4 Medium Low 85 92   
1 4 High Medium 78 92   
2 5 High Low 87 94   
3 5 High Low 97 88   
4 4 High High 76 86   
5 5 High Medium 91 83   
6 3 High High 70 91   
7 1 Medium Medium 88 88   
8 4 High High 88 87   
9 4 High Medium 90 87   
10 1 Medium Medium 87 82   
11 3 High Medium 72 90   
12 4 High Medium 91 82   
13 5 High Medium 81 76   
14 4 High Medium 84 81   
15 2 High High 63 83   
16 2 Medium Medium 45 79   
17 1 Medium Medium 73 71   
18 2 Low Medium 72 63   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal Crossing \  
0 91 95 34 65 85   
1 79 85 44 82 71   
2 80 87 34 75 87   
3 80 92 36 77 78   
4 93 88 64 78 94   
5 86 94 37 63 85   
6 83 82 47 83 80   
7 91 88 57 89 15   
8 82 86 43 69 83   
9 81 90 45 75 79   
10 93 88 63 88 20   
11 82 83 47 83 80   
12 79 88 45 66 79   
13 82 91 36 65 82   
14 72 85 48 84 56   
15 59 69 52 86 42   
16 57 64 50 76 38   
17 72 76 50 72 17   
18 52 56 66 46 72   
  
 Finishing HeadingAccuracy ShortPassing Volleys Dribbling Curve \  
0 95 70 91 88 96 93   
1 95 90 85 89 85 79   
2 95 90 80 86 88 81   
3 93 72 85 83 93 80   
4 82 55 94 82 88 85   
5 83 63 86 86 95 88   
6 94 86 85 88 83 83   
7 13 25 60 11 30 14   
8 88 68 84 78 87 85   
9 91 59 84 81 90 83   
10 14 14 61 18 23 15   
11 93 84 83 90 83 86   
12 85 46 83 67 87 79   
13 83 38 87 83 92 84   
14 85 87 78 88 83 78   
15 87 92 76 83 65 40   
16 81 91 73 82 66 61   
17 12 15 59 13 30 15   
18 75 58 56 70 50 46   
  
 FKAccuracy LongPassing BallControl Acceleration SprintSpeed Agility \  
0 94 91 96 91 80 91   
1 85 70 88 77 79 77   
2 84 77 88 85 88 86   
3 69 71 91 97 97 92   
4 83 93 91 76 76 79   
5 87 81 95 93 89 96   
6 65 86 85 65 74 71   
7 11 68 46 54 60 51   
8 74 74 84 85 90 86   
9 69 75 89 89 91 91   
10 20 66 40 64 63 60   
11 82 77 86 76 69 75   
12 72 69 86 95 88 94   
13 65 70 90 85 78 91   
14 43 73 85 86 83 86   
15 37 43 79 59 66 62   
16 30 37 74 43 47 33   
17 14 65 47 57 45 49   
18 31 47 53 71 72 70   
  
 Reactions Balance ShotPower Jumping Stamina Strength LongShots \  
0 94 95 86 68 72 69 94   
1 93 82 90 85 76 86 87   
2 94 74 94 95 77 77 93   
3 93 83 86 78 88 77 82   
4 91 78 91 63 89 74 91   
5 89 84 80 64 81 53 81   
6 92 70 91 79 83 85 86   
7 87 35 68 77 43 80 16   
8 91 78 88 60 88 64 89   
9 92 91 82 69 85 75 84   
10 88 48 70 66 41 78 18   
11 92 78 89 69 78 85 88   
12 90 93 78 57 79 64 83   
13 85 90 73 51 77 68 65   
14 89 89 80 88 80 84 74   
15 85 44 86 79 83 89 74   
16 81 33 80 44 63 87 69   
17 71 64 54 70 43 65 18   
18 69 88 43 35 84 24 63   
  
 Aggression Interceptions Positioning Vision Penalties Composure \  
0 44 40 93 95 75 96   
1 81 49 95 81 90 88   
2 63 29 95 76 88 95   
3 62 38 92 82 79 88   
4 76 66 88 94 83 89   
5 63 37 86 90 93 93   
6 80 44 94 87 91 91   
7 29 30 12 70 47 70   
8 62 39 91 83 73 89   
9 63 55 91 83 86 90   
10 38 27 20 70 51 70   
11 87 41 91 84 83 87   
12 59 30 91 82 67 78   
13 44 39 83 87 62 86   
14 86 45 86 81 74 83   
15 85 34 88 67 71 83   
16 75 39 85 69 90 81   
17 38 30 13 69 34 67   
18 55 70 42 35 50 73   
  
 Marking StandingTackle SlidingTackle GKDiving GKHandling GKKicking \  
0 20 35 24 6 11 15   
1 35 42 19 15 6 12   
2 24 32 24 7 11 15   
3 26 34 32 13 5 7   
4 68 65 53 15 13 5   
5 35 32 29 9 9 15   
6 50 36 38 8 10 11   
7 17 10 11 88 88 91   
8 50 34 33 11 13 13   
9 38 43 41 14 14 9   
10 29 15 8 87 82 93   
11 42 45 38 27 25 31   
12 47 53 47 15 12 12   
13 32 39 32 7 11 10   
14 55 31 44 11 8 8   
15 44 67 30 11 9 14   
16 38 65 26 6 12 15   
17 30 14 19 73 71 72   
18 68 62 69 12 10 18   
  
 GKPositioning GKReflexes STRating LWRating LFRating CFRating \  
0 14 8 92 92 93 93   
1 8 10 92 85 88 88   
2 14 11 91 88 89 89   
3 11 6 91 90 90 90   
4 10 13 86 88 87 87   
5 15 11 86 90 88 88   
6 14 11 90 84 86 86   
7 89 88 43 40 43 43   
8 6 10 88 87 87 87   
9 11 14 88 88 88 88   
10 88 88 43 41 43 43   
11 33 37 88 84 86 86   
12 15 9 84 87 86 86   
13 11 13 81 85 84 84   
14 8 13 87 81 84 84   
15 16 12 83 72 77 77   
16 11 8 77 66 73 73   
17 72 76 37 39 40 40   
18 12 14 59 59 56 56   
  
 RFRating RWRating CAMRating LMRating CMRating RMRating LWBRating \  
0 93 92 93 93 90 93 69   
1 88 85 89 87 83 87 67   
2 89 88 89 89 81 89 66   
3 90 90 92 92 84 92 70   
4 87 88 91 91 91 91 82   
5 88 90 91 91 85 91 70   
6 86 84 88 87 85 87 70   
7 43 40 50 47 53 47 39   
8 87 87 89 89 83 89 71   
9 88 88 89 89 85 89 74   
10 43 41 50 47 52 47 40   
11 86 84 88 86 83 86 69   
12 86 87 89 88 81 88 72   
13 84 85 88 88 82 88 69   
14 84 81 85 83 80 83 68   
15 77 72 75 72 72 72 61   
16 73 66 72 67 69 67 56   
17 40 39 48 44 50 44 39   
18 56 59 56 61 57 61 69   
  
 CDMRating RWBRating LBRating CBRating RBRating GKRating \  
0 67 69 64 53 64 22   
1 69 67 64 63 64 22   
2 62 66 63 56 63 23   
3 66 70 66 57 66 21   
4 83 82 78 72 78 24   
5 66 70 65 53 65 23   
6 71 70 67 64 67 23   
7 46 39 38 37 38 90   
8 67 71 67 57 67 22   
9 71 74 70 61 70 25   
10 47 40 39 38 39 90   
11 70 69 66 64 66 40   
12 67 72 68 58 68 24   
13 65 69 63 51 63 22   
14 69 68 66 65 66 22   
15 65 61 60 66 60 22   
16 62 56 56 63 56 20   
17 46 39 37 36 37 74   
18 62 69 70 61 70 21   
  
 AttackingWorkRate\_code DefensiveWorkRate\_code BestPosition\_code \  
0 2 1 12   
1 3 2 14   
2 3 1 14   
3 3 1 14   
4 3 3 4   
5 3 2 8   
6 3 3 14   
7 2 2 5   
8 3 3 7   
9 3 2 12   
10 2 2 5   
11 3 2 14   
12 3 2 8   
13 3 2 0   
14 3 2 14   
15 3 3 14   
16 2 2 14   
17 2 2 5   
18 1 2 10   
  
 PreferredFoot\_code Outlier   
0 0 -1   
1 1 -1   
2 1 -1   
3 1 -1   
4 1 -1   
5 1 -1   
6 1 -1   
7 1 -1   
8 1 -1   
9 0 -1   
10 0 -1   
11 1 -1   
12 1 -1   
13 1 -1   
14 1 -1   
15 1 -1   
16 1 -1   
17 1 -1   
18 1 -1

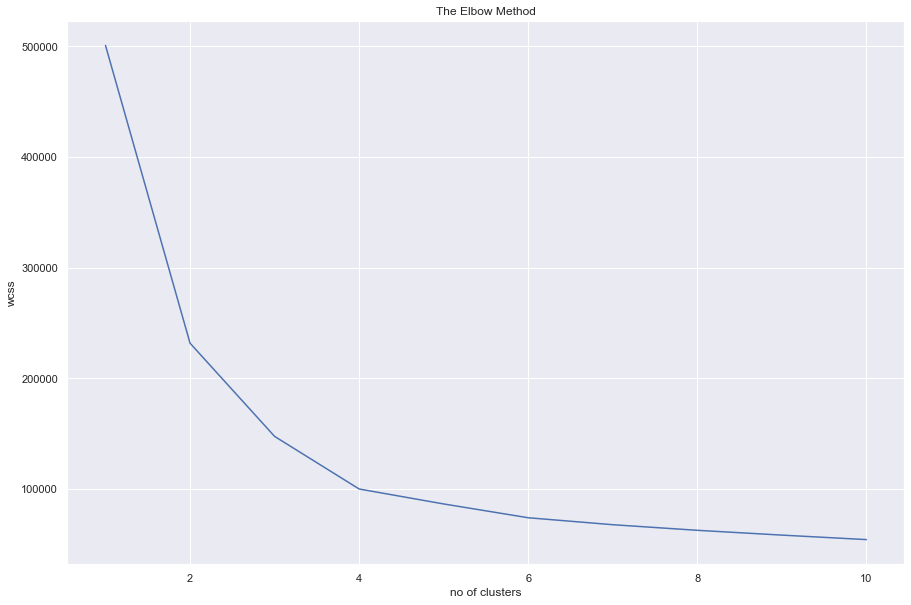
So, DBSCAN basically identified the 16 best players at present, 1 youth player and 2 not so good players for their age as outliers.

# K-means clustering

# Scaling the data  
X = final\_df.values  
X\_scaler = StandardScaler().fit(X)  
X\_scaled = X\_scaler.transform(X)

wcss = []  
  
# We always assume the max number of cluster would be 10  
# You can judge the number of clusters by doing averaging  
for i in range(1,11):  
 kmeans = KMeans(n\_clusters= i, init='k-means++', random\_state=0)  
 kmeans.fit(X\_scaled)  
 wcss.append(kmeans.inertia\_)  
  
 #inertia\_ is the formula used to segregate the data points into clusters

# Visualizing the ELBOW method to get the optimal value of K   
plt.plot(range(1,11), wcss)  
plt.title('The Elbow Method')  
plt.xlabel('no of clusters')  
plt.ylabel('wcss')  
plt.show()



We see the last elbow bend at n = 4

kmeansmodel = KMeans(n\_clusters= 4, random\_state=31)  
# unsupervised learning  
kmeansmodel.fit(X\_scaled)  
# y\_kmeans= kmeansmodel\_sup.labels\_  
# Predict the clusters  
y\_kmeans = kmeansmodel.predict(X\_scaled)

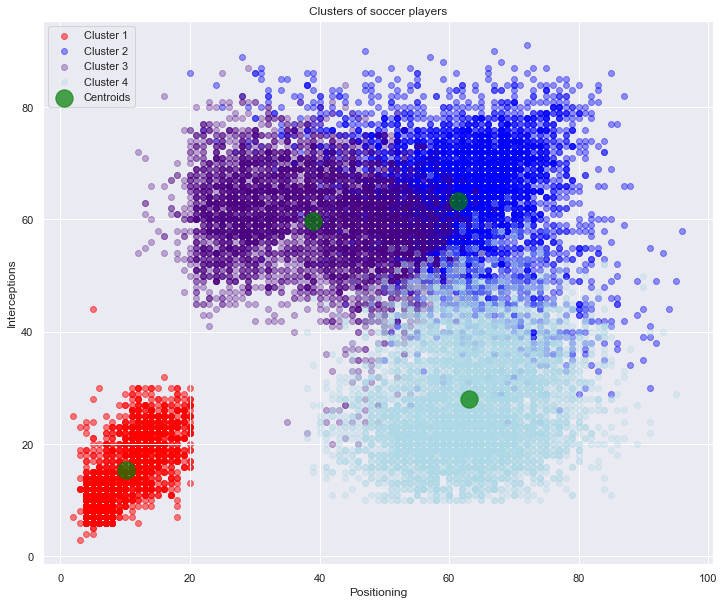
np.unique(y\_kmeans, return\_counts=True)

(array([0, 1, 2, 3]), array([2126, 6421, 5038, 5675], dtype=int64))

# Inverse scale transform the clusters' centers   
cluster\_center = X\_scaler.inverse\_transform(kmeansmodel.cluster\_centers\_)  
# cluster\_center

# for index, i in enumerate(feature\_list):  
# print(f'{index} - {i}')

#Visualizing all the clusters   
plt.figure(figsize=(12, 10))  
plt.scatter(X[y\_kmeans == 0, 8], X[y\_kmeans == 0, 20], c = 'red', \  
 label = 'Cluster 1', zorder = 0, alpha = .5)  
plt.scatter(X[y\_kmeans == 1, 8], X[y\_kmeans == 1, 20], c = 'blue', \  
 label = 'Cluster 2', zorder = 1, alpha = .4)  
plt.scatter(X[y\_kmeans == 2, 8], X[y\_kmeans == 2, 20], c = 'indigo',\  
 label = 'Cluster 3', zorder = 2, alpha = .3)  
plt.scatter(X[y\_kmeans == 3, 8], X[y\_kmeans == 3, 20], c = 'lightblue',\  
 label = 'Cluster 4', zorder = 3, alpha = .3)  
# plot centroids  
plt.scatter(cluster\_center[:, 8], cluster\_center[:, 20], s = 300, c = 'green',\  
 zorder = 6, label = 'Centroids', alpha = .7)  
plt.title('Clusters of soccer players')  
plt.xlabel('Positioning')  
plt.ylabel('Interceptions')  
plt.legend()  
plt.show()



df["Kmean\_cluster"] = y\_kmeans  
df['Kmean\_cluster'].value\_counts()

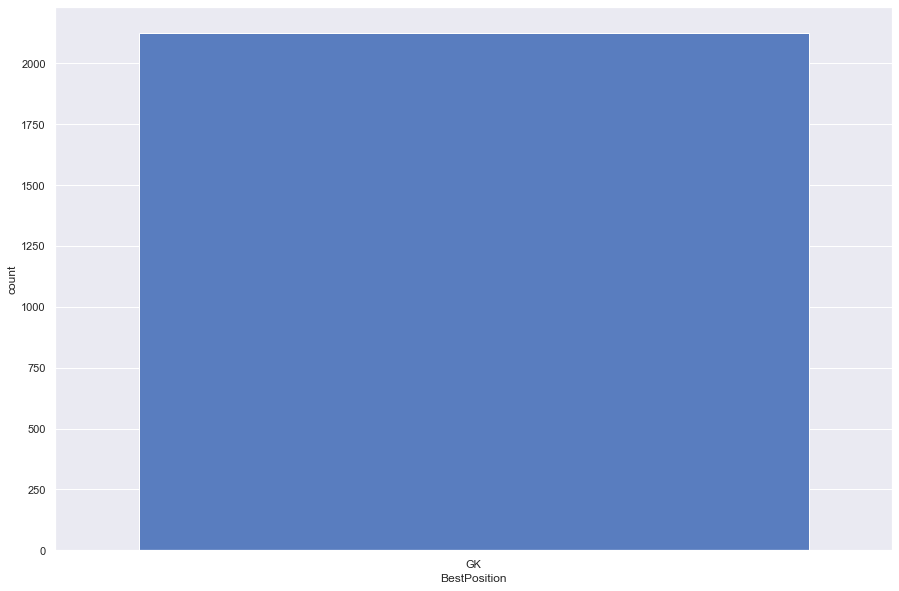
1 6421  
3 5675  
2 5038  
0 2126  
Name: Kmean\_cluster, dtype: int64

# Clusters visualizations

## Cluster 0

sns.set(rc = {'figure.figsize':(15,10)})  
sns.countplot(data=df[df['Kmean\_cluster'] == 0], x='BestPosition', palette='muted')

<AxesSubplot:xlabel='BestPosition', ylabel='count'>



df[df['Kmean\_cluster'] == 0].head()

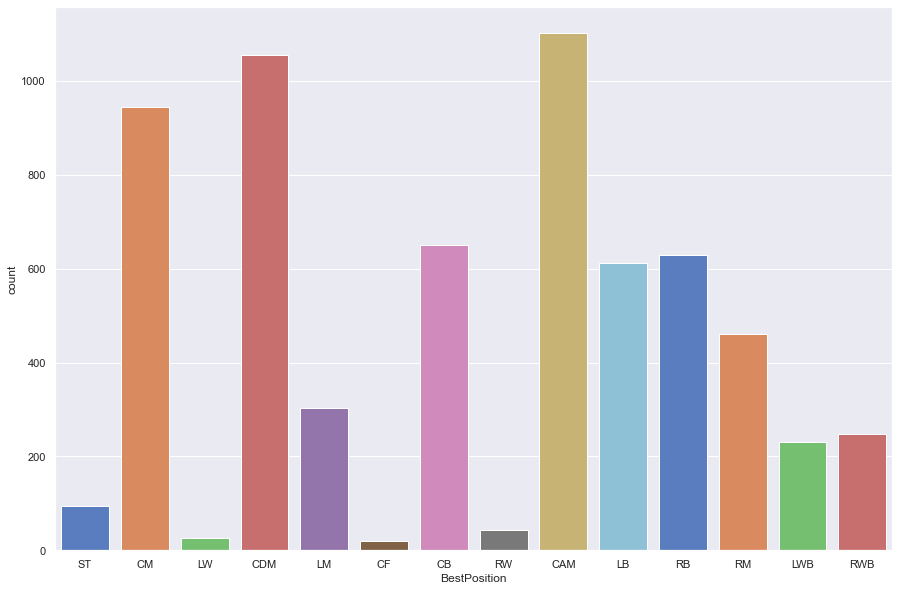
Name FullName Age Height Weight \  
4 J. Oblak Jan Oblak 28 188 87   
9 M. ter Stegen Marc-André ter Stegen 29 187 85   
10 M. Neuer Manuel Neuer 35 193 93   
14 T. Courtois Thibaut Courtois 29 199 96   
17 G. Donnarumma Gianluigi Donnarumma 22 196 90   
  
 PhotoUrl Nationality Overall \  
4 https://cdn.sofifa.com/players/200/389/22\_60.png Slovenia 91   
9 https://cdn.sofifa.com/players/192/448/22\_60.png Germany 90   
10 https://cdn.sofifa.com/players/167/495/22\_60.png Germany 90   
14 https://cdn.sofifa.com/players/192/119/22\_60.png Belgium 89   
17 https://cdn.sofifa.com/players/230/621/22\_60.png Italy 89   
  
 Potential Growth TotalStats BaseStats Positions BestPosition \  
4 93 2 1413 489 GK GK   
9 92 2 1444 484 GK GK   
10 90 0 1534 501 GK GK   
14 91 2 1327 469 GK GK   
17 93 4 1377 481 GK GK   
  
 Club ValueEUR WageEUR ReleaseClause ClubPosition \  
4 Atlético de Madrid 112000000 130000 238000000 GK   
9 FC Barcelona 99000000 250000 210400000 GK   
10 FC Bayern München 13500000 86000 22300000 GK   
14 Real Madrid CF 85500000 250000 181700000 GK   
17 Paris Saint-Germain 119500000 110000 230000000 GK   
  
 ContractUntil ClubNumber ClubJoined OnLoad NationalTeam \  
4 2023.0 13.0 2014 False Not in team   
9 2025.0 1.0 2014 False Not in team   
10 2023.0 1.0 2011 False Germany   
14 2026.0 1.0 2018 False Belgium   
17 2026.0 50.0 2021 False Italy   
  
 NationalPosition NationalNumber PreferredFoot IntReputation WeakFoot \  
4 NaN NaN Right 5 3   
9 NaN NaN Right 4 4   
10 GK 1.0 Right 5 4   
14 GK 1.0 Left 4 3   
17 GK 21.0 Right 3 3   
  
 SkillMoves AttackingWorkRate DefensiveWorkRate PaceTotal ShootingTotal \  
4 1 Medium Medium 87 92   
9 1 Medium Medium 88 85   
10 1 Medium Medium 88 88   
14 1 Medium Medium 84 89   
17 1 Medium Medium 91 83   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal Crossing \  
4 78 90 52 90 13   
9 88 90 45 88 18   
10 91 88 57 89 15   
14 74 88 48 86 14   
17 79 90 53 85 12   
  
 Finishing HeadingAccuracy ShortPassing Volleys Dribbling Curve \  
4 11 15 43 13 12 13   
9 14 11 61 14 21 18   
10 13 25 60 11 30 14   
14 14 13 33 12 13 19   
17 12 12 36 8 28 12   
  
 FKAccuracy LongPassing BallControl Acceleration SprintSpeed Agility \  
4 14 40 30 43 60 67   
9 12 63 30 38 50 39   
10 11 68 46 54 60 51   
14 20 35 23 42 52 62   
17 14 34 30 50 55 64   
  
 Reactions Balance ShotPower Jumping Stamina Strength LongShots \  
4 88 49 59 78 41 78 12   
9 86 43 66 79 35 78 10   
10 87 35 68 77 43 80 16   
14 84 45 56 68 38 70 17   
17 85 38 59 72 34 72 18   
  
 Aggression Interceptions Positioning Vision Penalties Composure \  
4 34 19 11 65 11 68   
9 43 22 11 70 25 70   
10 29 30 12 70 47 70   
14 23 15 13 44 27 66   
17 30 26 14 60 24 68   
  
 Marking StandingTackle SlidingTackle GKDiving GKHandling GKKicking \  
4 27 12 18 87 92 78   
9 25 13 10 88 85 88   
10 17 10 11 88 88 91   
14 20 18 16 84 89 74   
17 20 14 16 91 83 79   
  
 GKPositioning GKReflexes STRating LWRating LFRating CFRating \  
4 90 90 36 32 35 35   
9 88 90 38 35 38 38   
10 89 88 43 40 43 43   
14 86 88 34 29 31 31   
17 85 90 37 34 36 36   
  
 RFRating RWRating CAMRating LMRating CMRating RMRating LWBRating \  
4 35 32 41 38 41 38 35   
9 38 35 45 42 48 42 36   
10 43 40 50 47 53 47 39   
14 31 29 35 34 35 34 32   
17 36 34 41 38 40 38 34   
  
 CDMRating RWBRating LBRating CBRating RBRating GKRating \  
4 39 35 35 36 35 92   
9 44 36 34 36 34 91   
10 46 39 38 37 38 90   
14 34 32 32 32 32 89   
17 37 34 34 34 34 90   
  
 AttackingWorkRate\_code DefensiveWorkRate\_code BestPosition\_code \  
4 2 2 5   
9 2 2 5   
10 2 2 5   
14 2 2 5   
17 2 2 5   
  
 PreferredFoot\_code Outlier Kmean\_cluster   
4 1 0 0   
9 1 0 0   
10 1 -1 0   
14 0 0 0   
17 1 0 0

Only Goalkeepers were selected in this cluster.

## Cluster 1

sns.set(rc = {'figure.figsize':(15,10)})  
sns.countplot(data=df[df['Kmean\_cluster'] == 1], x='BestPosition', palette='muted')

<AxesSubplot:xlabel='BestPosition', ylabel='count'>



df[df['Kmean\_cluster'] == 1].head()

Name FullName Age Height Weight \  
1 R. Lewandowski Robert Lewandowski 32 185 81   
3 K. Mbappé Kylian Mbappé 22 182 73   
5 K. De Bruyne Kevin De Bruyne 30 181 70   
6 Neymar Jr Neymar da Silva Santos Jr. 29 175 68   
7 N. Kanté N'Golo Kanté 30 168 70   
  
 PhotoUrl Nationality Overall \  
1 https://cdn.sofifa.com/players/188/545/22\_60.png Poland 92   
3 https://cdn.sofifa.com/players/231/747/22\_60.png France 91   
5 https://cdn.sofifa.com/players/192/985/22\_60.png Belgium 91   
6 https://cdn.sofifa.com/players/190/871/22\_60.png Brazil 91   
7 https://cdn.sofifa.com/players/215/914/22\_60.png France 90   
  
 Potential Growth TotalStats BaseStats Positions BestPosition \  
1 92 0 2212 460 ST ST   
3 95 4 2175 470 ST,LW ST   
5 91 0 2304 485 CM,CAM CM   
6 91 0 2183 454 LW,CAM LW   
7 90 0 2179 470 CDM,CM CDM   
  
 Club ValueEUR WageEUR ReleaseClause ClubPosition \  
1 FC Bayern München 119500000 270000 197200000 ST   
3 Paris Saint-Germain 194000000 230000 373500000 ST   
5 Manchester City 125500000 350000 232200000 CM   
6 Paris Saint-Germain 129000000 270000 238700000 LW   
7 Chelsea 100000000 230000 185000000 CM   
  
 ContractUntil ClubNumber ClubJoined OnLoad NationalTeam \  
1 2023.0 9.0 2014 False Poland   
3 2022.0 7.0 2018 False France   
5 2025.0 17.0 2015 False Belgium   
6 2025.0 10.0 2017 False Not in team   
7 2023.0 7.0 2016 False France   
  
 NationalPosition NationalNumber PreferredFoot IntReputation WeakFoot \  
1 ST 9.0 Right 5 4   
3 LW 10.0 Right 4 4   
5 CM 7.0 Right 4 5   
6 NaN NaN Right 5 5   
7 CDM 13.0 Right 4 3   
  
 SkillMoves AttackingWorkRate DefensiveWorkRate PaceTotal ShootingTotal \  
1 4 High Medium 78 92   
3 5 High Low 97 88   
5 4 High High 76 86   
6 5 High Medium 91 83   
7 2 Medium High 78 66   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal Crossing \  
1 79 85 44 82 71   
3 80 92 36 77 78   
5 93 88 64 78 94   
6 86 94 37 63 85   
7 75 81 87 83 68   
  
 Finishing HeadingAccuracy ShortPassing Volleys Dribbling Curve \  
1 95 90 85 89 85 79   
3 93 72 85 83 93 80   
5 82 55 94 82 88 85   
6 83 63 86 86 95 88   
7 65 54 82 56 79 49   
  
 FKAccuracy LongPassing BallControl Acceleration SprintSpeed Agility \  
1 85 70 88 77 79 77   
3 69 71 91 97 97 92   
5 83 93 91 76 76 79   
6 87 81 95 93 89 96   
7 49 79 81 82 75 82   
  
 Reactions Balance ShotPower Jumping Stamina Strength LongShots \  
1 93 82 90 85 76 86 87   
3 93 83 86 78 88 77 82   
5 91 78 91 63 89 74 91   
6 89 84 80 64 81 53 81   
7 93 92 71 77 97 72 65   
  
 Aggression Interceptions Positioning Vision Penalties Composure \  
1 81 49 95 81 90 88   
3 62 38 92 82 79 88   
5 76 66 88 94 83 89   
6 63 37 86 90 93 93   
7 93 91 72 78 54 84   
  
 Marking StandingTackle SlidingTackle GKDiving GKHandling GKKicking \  
1 35 42 19 15 6 12   
3 26 34 32 13 5 7   
5 68 65 53 15 13 5   
6 35 32 29 9 9 15   
7 90 93 86 15 12 10   
  
 GKPositioning GKReflexes STRating LWRating LFRating CFRating \  
1 8 10 92 85 88 88   
3 11 6 91 90 90 90   
5 10 13 86 88 87 87   
6 15 11 86 90 88 88   
7 7 10 75 77 77 77   
  
 RFRating RWRating CAMRating LMRating CMRating RMRating LWBRating \  
1 88 85 89 87 83 87 67   
3 90 90 92 92 84 92 70   
5 87 88 91 91 91 91 82   
6 88 90 91 91 85 91 70   
7 77 77 81 82 85 82 88   
  
 CDMRating RWBRating LBRating CBRating RBRating GKRating \  
1 69 67 64 63 64 22   
3 66 70 66 57 66 21   
5 83 82 78 72 78 24   
6 66 70 65 53 65 23   
7 90 88 88 87 88 23   
  
 AttackingWorkRate\_code DefensiveWorkRate\_code BestPosition\_code \  
1 3 2 14   
3 3 1 14   
5 3 3 4   
6 3 2 8   
7 2 3 2   
  
 PreferredFoot\_code Outlier Kmean\_cluster   
1 1 -1 1   
3 1 -1 1   
5 1 -1 1   
6 1 -1 1   
7 1 1 1

df[df['Kmean\_cluster'] == 1].describe()

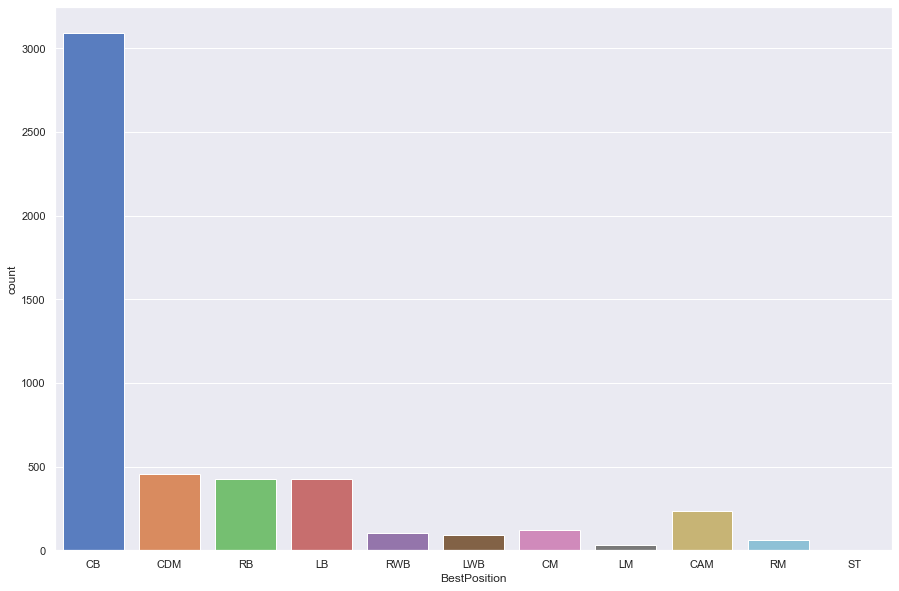
Age Height Weight Overall Potential \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 26.351347 179.232363 73.363962 69.714530 73.196698   
std 4.239875 5.810578 5.898683 5.574778 5.924953   
min 16.000000 158.000000 51.000000 57.000000 59.000000   
25% 23.000000 175.000000 70.000000 66.000000 69.000000   
50% 26.000000 179.000000 73.000000 69.000000 73.000000   
75% 29.000000 183.000000 77.000000 73.000000 77.000000   
max 40.000000 199.000000 100.000000 92.000000 95.000000   
  
 Growth TotalStats BaseStats ValueEUR WageEUR \  
count 6421.000000 6421.000000 6421.000000 6.421000e+03 6421.000000   
mean 3.482168 1832.644604 390.870269 5.084487e+06 15145.211026   
std 4.213448 131.619459 26.212863 1.109439e+07 27015.183441   
min 0.000000 1482.000000 320.000000 0.000000e+00 0.000000   
25% 0.000000 1737.000000 372.000000 9.000000e+05 2000.000000   
50% 1.000000 1815.000000 387.000000 1.600000e+06 6000.000000   
75% 6.000000 1919.000000 407.000000 3.700000e+06 16000.000000   
max 23.000000 2341.000000 496.000000 1.940000e+08 350000.000000   
  
 ReleaseClause ContractUntil ClubNumber ClubJoined NationalNumber \  
count 6.421000e+03 6391.000000 6391.000000 6421.000000 440.000000   
mean 9.159434e+06 2022.902206 18.619621 2018.999689 12.700000   
std 2.122534e+07 1.243567 16.148685 2.339156 6.905216   
min 0.000000e+00 2021.000000 1.000000 2003.000000 2.000000   
25% 1.300000e+06 2022.000000 8.000000 2018.000000 7.000000   
50% 2.600000e+06 2023.000000 16.000000 2020.000000 13.000000   
75% 6.600000e+06 2024.000000 24.000000 2021.000000 18.000000   
max 3.735000e+08 2031.000000 99.000000 2021.000000 28.000000   
  
 IntReputation WeakFoot SkillMoves PaceTotal ShootingTotal \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 1.177387 3.074910 2.692727 69.706432 57.013394   
std 0.500616 0.645493 0.637942 9.758450 9.925398   
min 1.000000 1.000000 2.000000 28.000000 26.000000   
25% 1.000000 3.000000 2.000000 65.000000 50.000000   
50% 1.000000 3.000000 3.000000 70.000000 57.000000   
75% 1.000000 3.000000 3.000000 76.000000 64.000000   
max 5.000000 5.000000 5.000000 97.000000 92.000000   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal \  
count 6421.000000 6421.000000 6421.000000 6421.000000   
mean 64.965893 68.204485 62.544152 68.435913   
std 6.450791 6.016780 8.572167 7.745573   
min 45.000000 48.000000 36.000000 39.000000   
25% 60.000000 64.000000 57.000000 64.000000   
50% 64.000000 67.000000 63.000000 69.000000   
75% 69.000000 72.000000 68.000000 74.000000   
max 93.000000 94.000000 91.000000 90.000000   
  
 Crossing Finishing HeadingAccuracy ShortPassing Volleys \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 62.564398 53.809375 57.485750 69.304625 49.683071   
std 9.553071 11.784717 10.407207 6.286342 12.621954   
min 17.000000 17.000000 19.000000 40.000000 12.000000   
25% 57.000000 46.000000 50.000000 65.000000 40.000000   
50% 63.000000 55.000000 57.000000 69.000000 49.000000   
75% 69.000000 62.000000 64.000000 73.000000 58.000000   
max 94.000000 95.000000 92.000000 94.000000 90.000000   
  
 Dribbling Curve FKAccuracy LongPassing BallControl \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 67.163526 59.313035 53.061050 65.320511 68.964024   
std 6.964868 12.231177 13.654744 7.655047 6.275710   
min 34.000000 19.000000 16.000000 28.000000 50.000000   
25% 62.000000 51.000000 42.000000 61.000000 64.000000   
50% 66.000000 60.000000 53.000000 65.000000 68.000000   
75% 72.000000 68.000000 64.000000 70.000000 73.000000   
max 95.000000 92.000000 92.000000 93.000000 95.000000   
  
 Acceleration SprintSpeed Agility Reactions Balance \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 69.958885 69.477496 70.607382 66.740383 70.237346   
std 9.935878 10.355624 9.182972 7.061195 10.001871   
min 29.000000 28.000000 30.000000 40.000000 30.000000   
25% 65.000000 64.000000 65.000000 62.000000 65.000000   
50% 70.000000 70.000000 71.000000 66.000000 71.000000   
75% 76.000000 76.000000 77.000000 71.000000 77.000000   
max 97.000000 97.000000 96.000000 94.000000 96.000000   
  
 ShotPower Jumping Stamina Strength LongShots \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 64.531849 67.154182 73.906868 66.618907 58.182370   
std 10.350589 11.248755 9.359157 10.320385 12.093673   
min 21.000000 29.000000 31.000000 24.000000 13.000000   
25% 58.000000 60.000000 69.000000 61.000000 50.000000   
50% 65.000000 68.000000 74.000000 67.000000 59.000000   
75% 72.000000 75.000000 80.000000 74.000000 67.000000   
max 95.000000 95.000000 97.000000 95.000000 91.000000   
  
 Aggression Interceptions Positioning Vision Penalties \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 66.324560 63.249027 61.441676 63.753621 53.510980   
std 10.742204 9.455526 8.960900 8.981655 11.071479   
min 24.000000 24.000000 20.000000 26.000000 18.000000   
25% 59.000000 58.000000 56.000000 58.000000 45.000000   
50% 67.000000 64.000000 61.000000 64.000000 53.000000   
75% 74.000000 70.000000 67.000000 70.000000 61.000000   
max 95.000000 91.000000 96.000000 94.000000 93.000000   
  
 Composure Marking StandingTackle SlidingTackle GKDiving \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 65.694907 61.871204 64.557546 61.606915 10.418315   
std 8.372791 10.056418 8.923105 9.952409 3.039829   
min 38.000000 20.000000 24.000000 18.000000 2.000000   
25% 60.000000 56.000000 59.000000 56.000000 8.000000   
50% 66.000000 63.000000 65.000000 62.000000 10.000000   
75% 71.000000 68.000000 70.000000 68.000000 13.000000   
max 94.000000 93.000000 93.000000 92.000000 27.000000   
  
 GKHandling GKKicking GKPositioning GKReflexes STRating \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 10.498520 10.458963 10.429995 10.438249 64.587447   
std 3.019904 3.069862 3.031038 3.040095 5.902137   
min 2.000000 2.000000 2.000000 2.000000 50.000000   
25% 8.000000 8.000000 8.000000 8.000000 60.000000   
50% 10.000000 10.000000 10.000000 10.000000 64.000000   
75% 13.000000 13.000000 13.000000 13.000000 68.000000   
max 25.000000 31.000000 33.000000 37.000000 92.000000   
  
 LWRating LFRating CFRating RFRating RWRating \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 65.172092 64.668743 64.668743 64.668743 65.172092   
std 5.823158 5.975865 5.975865 5.975865 5.823158   
min 52.000000 53.000000 53.000000 53.000000 52.000000   
25% 61.000000 60.000000 60.000000 60.000000 61.000000   
50% 64.000000 64.000000 64.000000 64.000000 64.000000   
75% 69.000000 68.000000 68.000000 68.000000 69.000000   
max 90.000000 90.000000 90.000000 90.000000 90.000000   
  
 CAMRating LMRating CMRating RMRating LWBRating \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 67.338577 67.993147 67.948606 67.993147 68.086279   
std 5.810748 5.386693 5.509847 5.386693 5.098961   
min 55.000000 57.000000 53.000000 57.000000 55.000000   
25% 63.000000 64.000000 64.000000 64.000000 64.000000   
50% 66.000000 67.000000 67.000000 67.000000 68.000000   
75% 71.000000 71.000000 71.000000 71.000000 71.000000   
max 92.000000 92.000000 91.000000 92.000000 88.000000   
  
 CDMRating RWBRating LBRating CBRating RBRating \  
count 6421.000000 6421.000000 6421.000000 6421.000000 6421.000000   
mean 67.805793 68.086279 67.448373 65.866064 67.448373   
std 5.649619 5.098961 5.408142 6.737569 5.408142   
min 55.000000 55.000000 54.000000 49.000000 54.000000   
25% 64.000000 64.000000 64.000000 61.000000 64.000000   
50% 67.000000 68.000000 67.000000 66.000000 67.000000   
75% 71.000000 71.000000 71.000000 70.000000 71.000000   
max 90.000000 88.000000 88.000000 89.000000 88.000000   
  
 GKRating AttackingWorkRate\_code DefensiveWorkRate\_code \  
count 6421.000000 6421.000000 6421.000000   
mean 18.677465 2.369257 2.244043   
std 1.712199 0.545084 0.507971   
min 11.000000 1.000000 1.000000   
25% 18.000000 2.000000 2.000000   
50% 19.000000 2.000000 2.000000   
75% 20.000000 3.000000 3.000000   
max 40.000000 3.000000 3.000000   
  
 BestPosition\_code PreferredFoot\_code Outlier Kmean\_cluster   
count 6421.000000 6421.000000 6421.000000 6421.0   
mean 4.843015 0.704875 0.995951 1.0   
std 4.177185 0.456135 0.089907 0.0   
min 0.000000 0.000000 -1.000000 1.0   
25% 1.000000 0.000000 1.000000 1.0   
50% 4.000000 1.000000 1.000000 1.0   
75% 9.000000 1.000000 1.000000 1.0   
max 14.000000 1.000000 1.000000 1.0

Players with good playmaking abilities were selected in this cluster.

## Cluster 2

sns.set(rc = {'figure.figsize':(15,10)})  
sns.countplot(data=df[df['Kmean\_cluster'] == 2], x='BestPosition', palette='muted')

<AxesSubplot:xlabel='BestPosition', ylabel='count'>



df[df['Kmean\_cluster'] == 2].head()

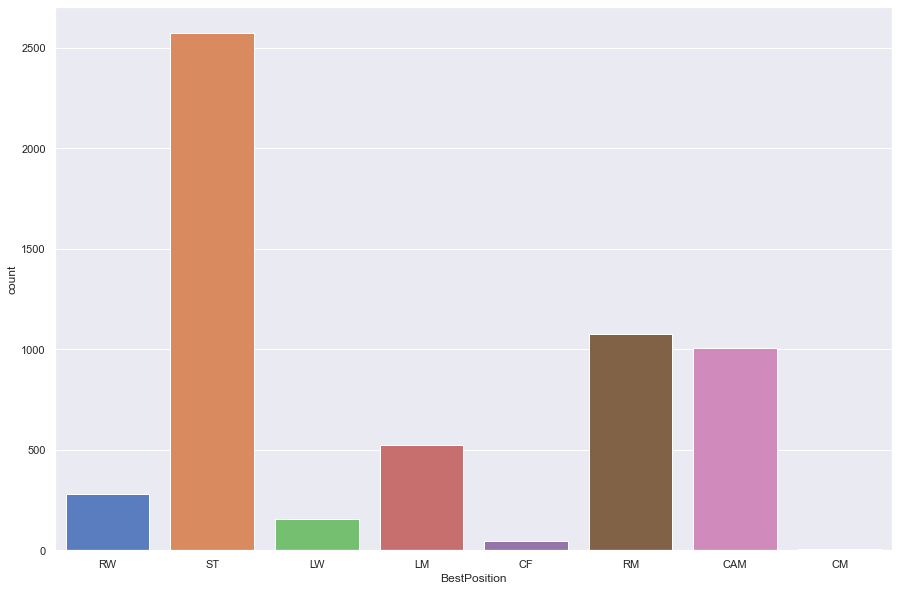
Name FullName Age Height Weight \  
118 S. Savić Stefan Savić 30 187 80   
150 K. Manolas Kostas Manolas 30 189 83   
219 N. Süle Niklas Süle 25 195 99   
306 W. Orban Willi Orban 28 186 87   
417 J. Evans Jonny Evans 33 188 77   
  
 PhotoUrl Nationality \  
118 https://cdn.sofifa.com/players/204/639/22\_60.png Montenegro   
150 https://cdn.sofifa.com/players/192/774/22\_60.png Greece   
219 https://cdn.sofifa.com/players/212/190/22\_60.png Germany   
306 https://cdn.sofifa.com/players/204/638/22\_60.png Hungary   
417 https://cdn.sofifa.com/players/169/588/22\_60.png Northern Ireland   
  
 Overall Potential Growth TotalStats BaseStats Positions BestPosition \  
118 84 84 0 1730 375 CB CB   
150 83 83 0 1652 377 CB CB   
219 82 87 5 1675 379 CB,RB CB   
306 81 82 1 1673 355 CB CB   
417 80 80 0 1729 363 CB CB   
  
 Club ValueEUR WageEUR ReleaseClause ClubPosition \  
118 Atlético de Madrid 34500000 74000 70700000 CB   
150 Napoli 30000000 80000 51000000 CB   
219 FC Bayern München 40500000 78000 69900000 CB   
306 RB Leipzig 24500000 79000 43500000 CB   
417 Leicester City 10000000 95000 19000000 SUB   
  
 ContractUntil ClubNumber ClubJoined OnLoad NationalTeam \  
118 2024.0 15.0 2015 False Not in team   
150 2024.0 44.0 2019 False Not in team   
219 2022.0 4.0 2017 False Germany   
306 2025.0 4.0 2015 False Hungary   
417 2023.0 6.0 2018 False Northern Ireland   
  
 NationalPosition NationalNumber PreferredFoot IntReputation WeakFoot \  
118 NaN NaN Right 2 3   
150 NaN NaN Right 3 2   
219 CB 15.0 Right 3 3   
306 CB 6.0 Right 2 3   
417 CB 5.0 Right 2 4   
  
 SkillMoves AttackingWorkRate DefensiveWorkRate PaceTotal ShootingTotal \  
118 2 Medium High 64 35   
150 2 Low Medium 81 25   
219 2 Medium Medium 65 42   
306 2 Medium Medium 49 36   
417 2 Medium High 52 32   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal Crossing \  
118 57 56 87 76 35   
150 44 61 85 81 27   
219 59 49 83 81 34   
306 52 54 84 80 39   
417 61 58 83 77 52   
  
 Finishing HeadingAccuracy ShortPassing Volleys Dribbling Curve \  
118 34 84 73 27 47 34   
150 25 80 65 16 54 30   
219 22 84 78 33 43 28   
306 38 85 69 31 52 28   
417 28 80 74 46 51 34   
  
 FKAccuracy LongPassing BallControl Acceleration SprintSpeed Agility \  
118 28 72 67 60 67 59   
150 10 61 66 77 84 70   
219 30 69 62 48 79 33   
306 32 58 57 50 49 48   
417 37 65 71 51 53 45   
  
 Reactions Balance ShotPower Jumping Stamina Strength LongShots \  
118 83 45 43 86 53 83 31   
150 81 60 40 91 69 83 12   
219 79 32 77 39 63 93 62   
306 75 56 37 91 68 85 29   
417 82 57 42 80 67 80 24   
  
 Aggression Interceptions Positioning Vision Penalties Composure \  
118 86 87 29 55 37 80   
150 86 86 25 22 26 83   
219 85 80 26 57 32 75   
306 80 81 42 44 41 72   
417 82 81 30 57 44 81   
  
 Marking StandingTackle SlidingTackle GKDiving GKHandling GKKicking \  
118 90 86 83 14 13 14   
150 85 85 87 8 6 8   
219 81 86 82 15 7 14   
306 84 86 80 15 12 15   
417 85 83 81 12 16 16   
  
 GKPositioning GKReflexes STRating LWRating LFRating CFRating \  
118 13 12 56 53 54 54   
150 15 12 55 51 50 50   
219 7 15 59 51 54 54   
306 10 6 55 51 52 52   
417 15 8 54 53 53 53   
  
 RFRating RWRating CAMRating LMRating CMRating RMRating LWBRating \  
118 54 53 59 58 66 58 72   
150 50 51 54 57 60 57 74   
219 54 51 59 57 67 57 72   
306 52 51 55 55 61 55 69   
417 53 53 58 59 66 59 73   
  
 CDMRating RWBRating LBRating CBRating RBRating GKRating \  
118 80 72 76 84 76 23   
150 77 74 78 83 78 21   
219 79 72 76 84 76 22   
306 75 69 72 82 72 20   
417 78 73 75 80 75 23   
  
 AttackingWorkRate\_code DefensiveWorkRate\_code BestPosition\_code \  
118 2 3 1   
150 1 2 1   
219 2 2 1   
306 2 2 1   
417 2 3 1   
  
 PreferredFoot\_code Outlier Kmean\_cluster   
118 1 1 2   
150 1 1 2   
219 1 1 2   
306 1 1 2   
417 1 1 2

Players with good defending abilities were selected in this cluster.

## Cluster 3

sns.set(rc = {'figure.figsize':(15,10)})  
sns.countplot(data=df[df['Kmean\_cluster'] == 3], x='BestPosition', palette='muted')

<AxesSubplot:xlabel='BestPosition', ylabel='count'>



df[df['Kmean\_cluster'] == 3].head()

Name FullName Age Height Weight \  
0 L. Messi Lionel Messi 34 170 72   
2 Cristiano Ronaldo C. Ronaldo dos Santos Aveiro 36 187 83   
22 R. Lukaku Romelu Lukaku 28 191 94   
35 C. Immobile Ciro Immobile 31 185 85   
44 S. Agüero Sergio Agüero 33 173 70   
  
 PhotoUrl Nationality Overall \  
0 https://cdn.sofifa.com/players/158/023/22\_60.png Argentina 93   
2 https://cdn.sofifa.com/players/020/801/22\_60.png Portugal 91   
22 https://cdn.sofifa.com/players/192/505/22\_60.png Belgium 88   
35 https://cdn.sofifa.com/players/192/387/22\_60.png Italy 87   
44 https://cdn.sofifa.com/players/153/079/22\_60.png Argentina 87   
  
 Potential Growth TotalStats BaseStats Positions BestPosition \  
0 93 0 2219 462 RW,ST,CF RW   
2 91 0 2208 457 ST,LW ST   
22 88 0 2064 445 ST ST   
35 87 0 2065 437 ST ST   
44 87 0 2068 424 ST ST   
  
 Club ValueEUR WageEUR ReleaseClause ClubPosition \  
0 Paris Saint-Germain 78000000 320000 144300000 RW   
2 Manchester United 45000000 270000 83300000 ST   
22 Chelsea 93500000 260000 173000000 ST   
35 Lazio 67500000 125000 114800000 ST   
44 FC Barcelona 51000000 260000 104600000 ST   
  
 ContractUntil ClubNumber ClubJoined OnLoad NationalTeam \  
0 2023.0 30.0 2021 False Argentina   
2 2023.0 7.0 2021 False Portugal   
22 2026.0 9.0 2021 False Belgium   
35 2025.0 17.0 2016 False Italy   
44 2023.0 19.0 2021 False Argentina   
  
 NationalPosition NationalNumber PreferredFoot IntReputation WeakFoot \  
0 RW 10.0 Left 5 4   
2 ST 7.0 Right 5 4   
22 ST 9.0 Left 4 4   
35 ST 17.0 Right 4 4   
44 SUB 9.0 Right 4 4   
  
 SkillMoves AttackingWorkRate DefensiveWorkRate PaceTotal ShootingTotal \  
0 4 Medium Low 85 92   
2 5 High Low 87 94   
22 3 Medium Medium 84 87   
35 3 High Medium 86 87   
44 4 High Medium 71 89   
  
 PassingTotal DribblingTotal DefendingTotal PhysicalityTotal Crossing \  
0 91 95 34 65 85   
2 80 87 34 75 87   
22 74 78 39 83 73   
35 67 81 39 77 55   
44 75 87 33 69 70   
  
 Finishing HeadingAccuracy ShortPassing Volleys Dribbling Curve \  
0 95 70 91 88 96 93   
2 95 90 80 86 88 81   
22 92 89 77 78 83 78   
35 91 81 80 85 80 70   
44 93 78 80 85 86 83   
  
 FKAccuracy LongPassing BallControl Acceleration SprintSpeed Agility \  
0 94 91 96 91 80 91   
2 84 77 88 85 88 86   
22 66 70 77 81 87 60   
35 51 53 83 84 87 75   
44 73 64 88 75 68 82   
  
 Reactions Balance ShotPower Jumping Stamina Strength LongShots \  
0 94 95 86 68 72 69 94   
2 94 74 94 95 77 77 93   
22 91 52 89 76 71 95 74   
35 90 85 86 82 78 76 80   
44 89 90 90 78 62 74 83   
  
 Aggression Interceptions Positioning Vision Penalties Composure \  
0 44 40 93 95 75 96   
2 63 29 95 76 88 95   
22 71 27 89 75 88 86   
35 77 40 91 71 88 79   
44 65 24 91 80 75 91   
  
 Marking StandingTackle SlidingTackle GKDiving GKHandling GKKicking \  
0 20 35 24 6 11 15   
2 24 32 24 7 11 15   
22 30 41 30 8 15 14   
35 34 33 32 6 8 15   
44 30 29 24 13 15 6   
  
 GKPositioning GKReflexes STRating LWRating LFRating CFRating \  
0 14 8 92 92 93 93   
2 14 11 91 88 89 89   
22 7 10 88 81 84 84   
35 12 6 87 81 84 84   
44 11 14 87 83 85 85   
  
 RFRating RWRating CAMRating LMRating CMRating RMRating LWBRating \  
0 93 92 93 93 90 93 69   
2 89 88 89 89 81 89 66   
22 84 81 83 83 77 83 64   
35 84 81 84 82 76 82 64   
44 85 83 87 83 78 83 60   
  
 CDMRating RWBRating LBRating CBRating RBRating GKRating \  
0 67 69 64 53 64 22   
2 62 66 63 56 63 23   
22 62 64 61 60 61 22   
35 63 64 62 59 62 20   
44 60 60 57 53 57 24   
  
 AttackingWorkRate\_code DefensiveWorkRate\_code BestPosition\_code \  
0 2 1 12   
2 3 1 14   
22 2 2 14   
35 3 2 14   
44 3 2 14   
  
 PreferredFoot\_code Outlier Kmean\_cluster   
0 0 -1 3   
2 1 -1 3   
22 0 1 3   
35 1 1 3   
44 1 1 3

Players with finishing (better goal scoring abilities) were selected in this cluster.

# Decision tree

final\_df.columns

Index(['GKRating', 'SlidingTackle', 'StandingTackle', 'Dribbling', 'RBRating',  
 'LBRating', 'RWRating', 'LWRating', 'Positioning', 'RWBRating',  
 'LWBRating', 'CBRating', 'Finishing', 'RMRating', 'LMRating',  
 'BallControl', 'Marking', 'LFRating', 'RFRating', 'CFRating',  
 'Interceptions', 'CAMRating', 'STRating', 'HeadingAccuracy',  
 'CDMRating', 'BestPosition\_code'],  
 dtype='object')

x= final\_df.drop('BestPosition\_code', axis=1).values   
y = final\_df['BestPosition\_code'].values

X\_train, X\_test, y\_train, y\_test = train\_test\_split(x, y, random\_state=42)

cv = KFold(n\_splits=10) # Desired number of Cross Validation folds  
accuracies = list()  
max\_attributes = len(list(final\_df.columns))  
depth\_range = range(1, max\_attributes + 1)  
  
# Testing max\_depths from 1 to max attributes  
# Uncomment prints for details about each Cross Validation pass  
for depth in depth\_range:  
 fold\_accuracy = []  
 tree\_model = tree.DecisionTreeClassifier(max\_depth = depth)  
 # print("Current max depth: ", depth, "\n")  
 for train\_fold, valid\_fold in cv.split(final\_df):  
 f\_train = final\_df.loc[train\_fold] # Extract train data with cv indices  
 f\_valid = final\_df.loc[valid\_fold] # Extract valid data with cv indices  
 # We fit the model with the fold train data  
 model = tree\_model.fit(X = f\_train.drop('BestPosition\_code', axis=1),   
 y = f\_train["BestPosition\_code"])   
 # We calculate accuracy with the fold validation data  
 valid\_acc = model.score(X = f\_valid.drop('BestPosition\_code', axis=1),   
 y = f\_valid["BestPosition\_code"])  
   
 fold\_accuracy.append(valid\_acc)  
  
 avg = sum(fold\_accuracy)/len(fold\_accuracy)  
 accuracies.append(avg)  
 # print("Accuracy per fold: ", fold\_accuracy, "\n")  
 # print("Average accuracy: ", avg)  
 # print("\n")  
   
# Just to show results conveniently  
df\_ = pd.DataFrame({"Max Depth": depth\_range, "Average Accuracy": accuracies})  
df\_ = df\_[["Max Depth", "Average Accuracy"]]  
print(df\_.to\_string(index=False))

Max Depth Average Accuracy  
 1 0.304777  
 2 0.384216  
 3 0.437902  
 4 0.495327  
 5 0.522897  
 6 0.533022  
 7 0.549740  
 8 0.566822  
 9 0.568795  
 10 0.573053  
 11 0.575182  
 12 0.580426  
 13 0.574299  
 14 0.570613  
 15 0.568172  
 16 0.565057  
 17 0.558463  
 18 0.555192  
 19 0.554154  
 20 0.560021  
 21 0.554517  
 22 0.553790  
 23 0.553323  
 24 0.555815  
 25 0.556023  
 26 0.556490

So, at max\_depth = 12 we get the highest average accuracy.

# Create Decision Tree with max\_depth = 12  
decision\_tree = tree.DecisionTreeClassifier(max\_depth = 12)  
decision\_tree.fit(X\_train, y\_train)  
  
# Predicting results for test dataset  
y\_pred = decision\_tree.predict(X\_test)

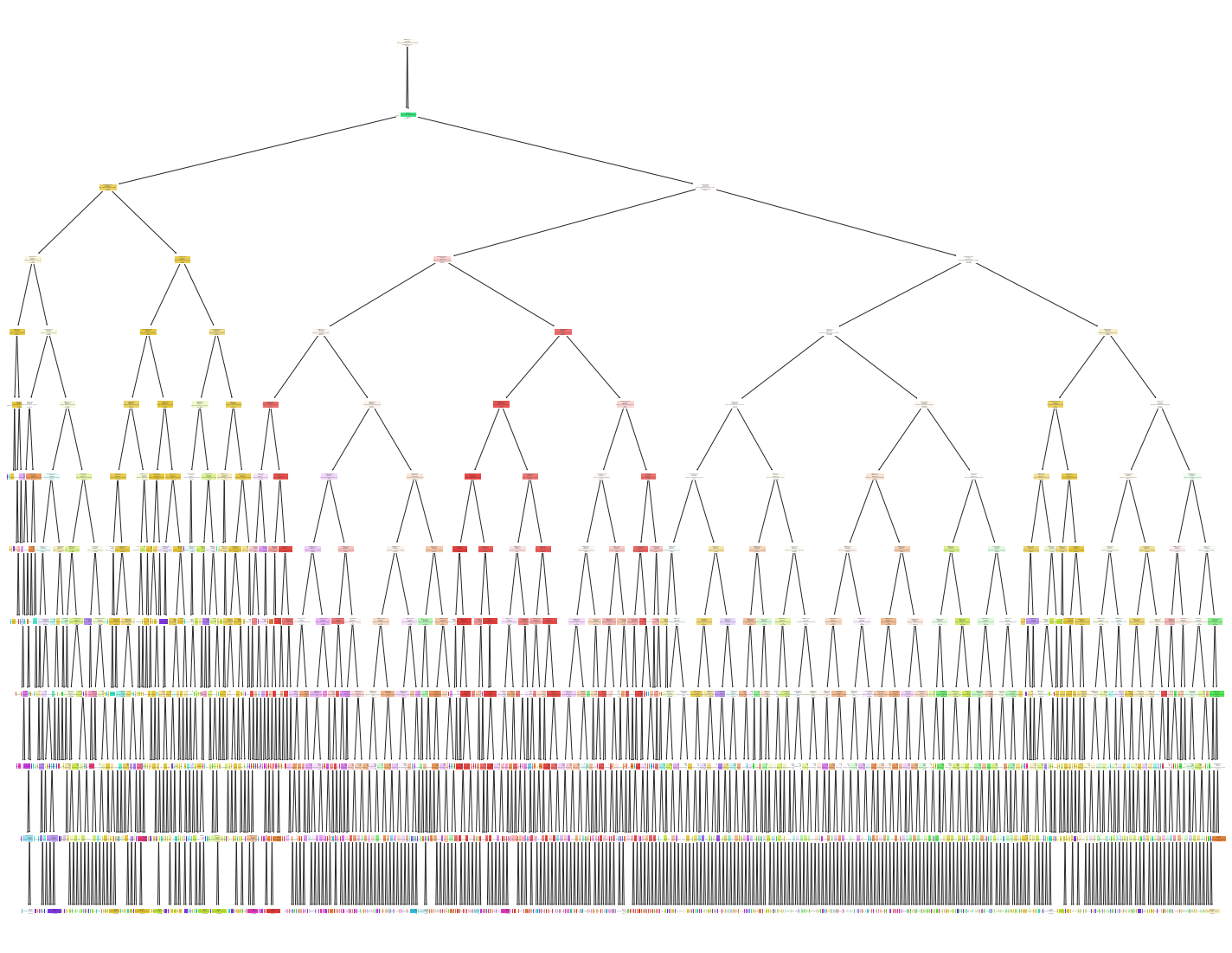
acc\_decision\_tree = round(decision\_tree.score(X\_train, y\_train) \* 100, 2)  
acc\_decision\_tree

74.9

round(decision\_tree.score(X\_test, y\_test) \* 100, 2)

61.77

# Decision tree diagram  
fig = plt.figure(figsize=(25,20))  
\_ = tree.plot\_tree(decision\_tree,   
 feature\_names=final\_df.columns[:-1],   
 class\_names=sorted(df['BestPosition'].unique()),  
 filled=True)



print(confusion\_matrix(y\_test, y\_pred))

[[371 1 9 3 57 0 5 26 3 1 4 84 2 0 26]  
 [ 3 797 57 0 6 0 27 0 0 1 24 0 0 1 0]  
 [ 4 35 221 0 47 0 31 2 0 4 17 6 0 0 0]  
 [ 5 0 0 0 0 0 0 1 1 0 0 0 1 0 6]  
 [ 46 1 53 0 136 0 11 4 0 0 13 10 0 0 0]  
 [ 0 0 0 0 0 491 0 0 0 0 0 0 0 0 0]  
 [ 2 30 39 0 21 0 95 2 0 14 52 13 0 5 1]  
 [ 69 0 2 0 11 0 5 25 3 1 4 95 4 0 9]  
 [ 14 0 0 1 0 0 0 2 2 0 0 20 2 0 7]  
 [ 3 2 20 0 7 0 29 1 0 8 8 3 0 2 0]  
 [ 2 34 50 0 21 0 101 2 0 3 45 4 0 2 0]  
 [124 2 5 0 26 0 8 26 2 1 2 167 6 1 18]  
 [ 23 0 0 1 0 0 0 3 3 0 0 37 5 0 26]  
 [ 4 3 13 0 4 0 39 4 0 10 9 0 0 2 0]  
 [ 24 1 1 1 2 0 1 11 1 0 1 33 5 0 609]]

print(classification\_report(y\_test, y\_pred))

precision recall f1-score support  
  
 0 0.53 0.63 0.58 592  
 1 0.88 0.87 0.87 916  
 2 0.47 0.60 0.53 367  
 3 0.00 0.00 0.00 14  
 4 0.40 0.50 0.44 274  
 5 1.00 1.00 1.00 491  
 6 0.27 0.35 0.30 274  
 7 0.23 0.11 0.15 228  
 8 0.13 0.04 0.06 48  
 9 0.19 0.10 0.13 83  
 10 0.25 0.17 0.20 264  
 11 0.35 0.43 0.39 388  
 12 0.20 0.05 0.08 98  
 13 0.15 0.02 0.04 88  
 14 0.87 0.88 0.88 690  
  
 accuracy 0.62 4815  
 macro avg 0.40 0.38 0.38 4815  
weighted avg 0.60 0.62 0.60 4815

So, our decision tree model has ~62% accuracy.

Also, the tree diagram is huge and hard to decipher. So I looked for alternative classification approaches.

# Random Forest Classifier

# Set features. This will also be used as x values.  
X = df.select\_dtypes(include=['int64']).drop(['Outlier', 'ClubJoined',\  
 'IntReputation', 'ReleaseClause'], axis = 1)  
y = df["BestPosition"]  
print(X.shape, y.shape)

(19260, 71) (19260,)

# Split data into training and testing groups and scale data   
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, random\_state=1)

# Fit the data into model  
rfm = RandomForestClassifier(n\_estimators=200, random\_state=1)  
rfm.fit(X\_train, y\_train)

RandomForestClassifier(n\_estimators=200, random\_state=1)

sorted(zip(rfm.feature\_importances\_, X.columns), reverse=True)

[(0.04611431524546392, 'HeadingAccuracy'),  
 (0.03237634073478377, 'DefendingTotal'),  
 (0.02911288185459061, 'Crossing'),  
 (0.029090570739770834, 'SlidingTackle'),  
 (0.028202222731977394, 'Positioning'),  
 (0.02451899201169612, 'Marking'),  
 (0.02417367159087792, 'Strength'),  
 (0.022791706847263882, 'GKReflexes'),  
 (0.02217030060454052, 'StandingTackle'),  
 (0.02170167960242648, 'Finishing'),  
 (0.020040494243249517, 'CBRating'),  
 (0.019594903600219024, 'LWRating'),  
 (0.019505150102041888, 'LongPassing'),  
 (0.018956729316211014, 'Vision'),  
 (0.018651253087254217, 'RWRating'),  
 (0.018347509927537097, 'Interceptions'),  
 (0.017587709127312084, 'GKKicking'),  
 (0.01711616090171432, 'PaceTotal'),  
 (0.016630374002905218, 'ShootingTotal'),  
 (0.01639659147736857, 'SprintSpeed'),  
 (0.01596816638928697, 'LBRating'),  
 (0.01579593735189817, 'PhysicalityTotal'),  
 (0.01571882153143265, 'GKPositioning'),  
 (0.01564023364707097, 'Acceleration'),  
 (0.015394936494765967, 'RWBRating'),  
 (0.01534058728605215, 'RMRating'),  
 (0.015301620031790869, 'GKDiving'),  
 (0.01472504083229949, 'LWBRating'),  
 (0.014510483630056283, 'GKHandling'),  
 (0.013160597857914257, 'GKRating'),  
 (0.013134253745305993, 'Height'),  
 (0.012731901449299947, 'RBRating'),  
 (0.012660075698334128, 'PassingTotal'),  
 (0.012527812021824403, 'SkillMoves'),  
 (0.012449009853477069, 'CFRating'),  
 (0.012215850446912544, 'LongShots'),  
 (0.01196207306732134, 'Stamina'),  
 (0.011697466313160614, 'Jumping'),  
 (0.011595297263283151, 'Dribbling'),  
 (0.011579988380582457, 'ShortPassing'),  
 (0.01151117493337366, 'RFRating'),  
 (0.011457260210883898, 'CAMRating'),  
 (0.011143932412053055, 'STRating'),  
 (0.011130214459921146, 'LFRating'),  
 (0.01097929285930847, 'Weight'),  
 (0.01096004381361957, 'Agility'),  
 (0.010910882358835567, 'Volleys'),  
 (0.010695771948649482, 'LMRating'),  
 (0.010435611924949136, 'Aggression'),  
 (0.010380730957590057, 'Balance'),  
 (0.010346633887862859, 'FKAccuracy'),  
 (0.010287180157777394, 'DribblingTotal'),  
 (0.009900282138013848, 'Curve'),  
 (0.009705448812686042, 'Penalties'),  
 (0.009331688413342264, 'CDMRating'),  
 (0.008945639575938264, 'CMRating'),  
 (0.00861722310482904, 'ShotPower'),  
 (0.008406554051652462, 'Age'),  
 (0.007775050744812666, 'BallControl'),  
 (0.007761265077966769, 'Growth'),  
 (0.007688589378498641, 'Composure'),  
 (0.007567366360022049, 'TotalStats'),  
 (0.007375966754154382, 'ValueEUR'),  
 (0.007249350124882528, 'BaseStats'),  
 (0.007129384525708989, 'Reactions'),  
 (0.006668672813919499, 'Potential'),  
 (0.005932654653827974, 'WageEUR'),  
 (0.005585621548763353, 'Overall'),  
 (0.002628973226462637, 'WeakFoot'),  
 (0.002194130232823609, 'AttackingWorkRate\_code'),  
 (0.0021076974955969276, 'DefensiveWorkRate\_code')]

predictions = rfm.predict(X\_test)  
base\_train\_accuracy = round(rfm.score(X\_train, y\_train)\*100,3)  
base\_test\_accuracy = round(rfm.score(X\_test, y\_test)\*100,3)  
print(f"Training Data Score: {rfm.score(X\_train, y\_train)}")  
print(f"Testing Data Score: {rfm.score(X\_test, y\_test)}")

Training Data Score: 1.0  
Testing Data Score: 0.7246105919003115

print(classification\_report(y\_test, predictions))

precision recall f1-score support  
  
 CAM 0.65 0.82 0.73 599  
 CB 0.90 0.94 0.92 958  
 CDM 0.68 0.81 0.74 389  
 CF 0.00 0.00 0.00 16  
 CM 0.60 0.61 0.60 249  
 GK 1.00 1.00 1.00 515  
 LB 0.43 0.43 0.43 269  
 LM 0.29 0.05 0.08 217  
 LW 1.00 0.02 0.04 45  
 LWB 0.14 0.01 0.03 70  
 RB 0.38 0.43 0.40 260  
 RM 0.47 0.57 0.52 376  
 RW 0.40 0.08 0.13 79  
 RWB 0.35 0.08 0.13 88  
 ST 0.89 0.94 0.91 685  
  
 accuracy 0.72 4815  
 macro avg 0.55 0.45 0.44 4815  
weighted avg 0.70 0.72 0.70 4815

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1245: UndefinedMetricWarning:  
  
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
  
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1245: UndefinedMetricWarning:  
  
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
  
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1245: UndefinedMetricWarning:  
  
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.

# Hyperparameter Tuning

### No need to run the next 4 blocks as it takes nearly 30 mins to execute!!!

# Get randomforest params  
# rfm.get\_params()

# Create the GridSearchCV model  
# param\_grid = {  
# 'n\_estimators': [200, 600, 1200, 100],  
# 'max\_features': ['auto', 'sqrt', 'log2'],  
# 'criterion': ['gini', 'entropy']  
# }  
# grid = GridSearchCV(rfm, param\_grid, cv=3, verbose=0)

# Train the model with GridSearch  
# grid.fit(X\_train, y\_train)  
  
# Output  
# GridSearchCV(cv=3, estimator=RandomForestClassifier(n\_estimators=200),  
# param\_grid={'criterion': ['gini', 'entropy'],  
# 'max\_features': ['auto', 'sqrt', 'log2'],  
# 'n\_estimators': [200, 600, 1200, 100]})

# print(grid.best\_params\_)  
# print(grid.best\_score\_)  
  
# Output  
# {'criterion': 'gini', 'max\_features': 'sqrt', 'n\_estimators': 600}  
# 0.7219799238490827

### Run from here

# Creating model using the tuned parameters  
rfm = RandomForestClassifier(n\_estimators=600,criterion= 'gini',\  
 max\_features= 'sqrt', random\_state=1)  
rfm.fit(X\_train, y\_train)

RandomForestClassifier(max\_features='sqrt', n\_estimators=600, random\_state=1)

predictions = rfm.predict(X\_test)  
tuned\_train\_accuracy = round(rfm.score(X\_train, y\_train)\*100,3)  
tuned\_test\_accuracy = round(rfm.score(X\_test, y\_test)\*100,3)  
print(f"Training Data Score: {rfm.score(X\_train, y\_train)}")  
print(f"Testing Data Score: {rfm.score(X\_test, y\_test)}")

Training Data Score: 1.0  
Testing Data Score: 0.7273104880581516

print(classification\_report(y\_test, predictions))

precision recall f1-score support  
  
 CAM 0.65 0.82 0.73 599  
 CB 0.90 0.94 0.92 958  
 CDM 0.69 0.82 0.75 389  
 CF 0.00 0.00 0.00 16  
 CM 0.61 0.63 0.62 249  
 GK 1.00 1.00 1.00 515  
 LB 0.42 0.43 0.43 269  
 LM 0.28 0.04 0.07 217  
 LW 1.00 0.02 0.04 45  
 LWB 0.29 0.03 0.05 70  
 RB 0.39 0.44 0.41 260  
 RM 0.48 0.58 0.52 376  
 RW 0.33 0.06 0.11 79  
 RWB 0.35 0.07 0.11 88  
 ST 0.89 0.94 0.91 685  
  
 accuracy 0.73 4815  
 macro avg 0.55 0.46 0.45 4815  
weighted avg 0.70 0.73 0.70 4815

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1245: UndefinedMetricWarning:  
  
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
  
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1245: UndefinedMetricWarning:  
  
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.  
  
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1245: UndefinedMetricWarning:  
  
Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.

predictions = rfm.predict(X\_test)  
print(f"Predicted Labels: {predictions[:5]}")  
print(f"Actual Labels: {list(y\_test[:5])}")

Predicted Labels: ['ST' 'CDM' 'RM' 'ST' 'CM']  
Actual Labels: ['ST', 'CDM', 'RM', 'LM', 'CM']

# Model evaluation

evaluations = {'': ['Base Train Model', 'Base Test Model', 'Tuned Train Model', 'Tuned Test Model'],  
 'RF Accuracy': [f"{base\_train\_accuracy}%", f"{base\_test\_accuracy}%",\  
 f"{tuned\_train\_accuracy}%", f"{tuned\_test\_accuracy}%"]}  
  
evaluations\_df = pd.DataFrame(evaluations)  
evaluations\_df = evaluations\_df.set\_index('')  
evaluations\_df

RF Accuracy  
   
Base Train Model 100.0%  
Base Test Model 72.461%  
Tuned Train Model 100.0%  
Tuned Test Model 72.731%

print(f'So our final tuned random forest classifier model has {tuned\_test\_accuracy}% accuracy.')

So our final tuned random forest classifier model has 72.731% accuracy.

This model has way better accuracy score than the decision tree classifier model.