MSA 8010 Final Project Report
Authors : Naga Prudhvi Panguluri, Kevin Vettickatt, Rahul Reddy Bommu.

Introduction

The main idea of our project is to build a player recommender system based on the data from the FIFA 23 game. To keep it simple, when we enter a name of a player, the computer should spit out 3 other players with the most similar traits(not specifically based on their position, but mainly on their traits,skills and style).

Can this be useful for anyone?

The EA sports FIFA game is the closest thing to an accurate measurment of the player's real life attributes and skills. They motion-capture players, record their skills, include their unique goal celebrations and emotions. This recommender system could be a very useful tool specifically for the FIFA gamers to replace an outgoing player from their teams. On top of that, it could also be a handy tool for scouts and teams to replace an outgoing player.

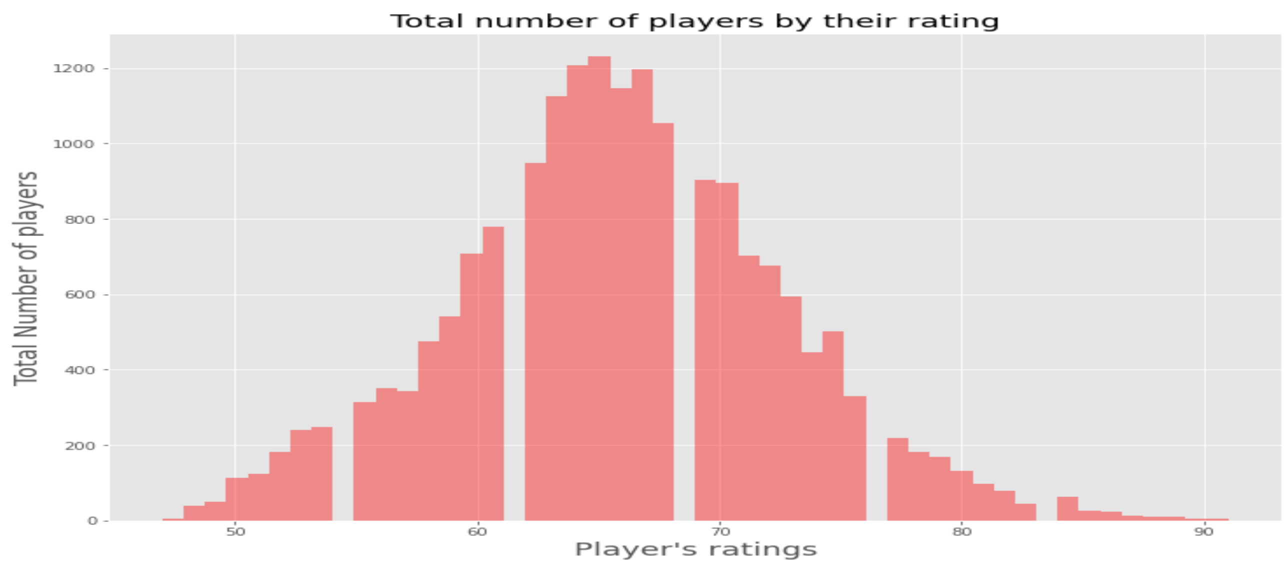
Along with our recommender system, we have also made a list of the best young talent from around the globe. If the teams/gamers would like to build a side for the future, this list could be a very good reference to them.

Data Exploration

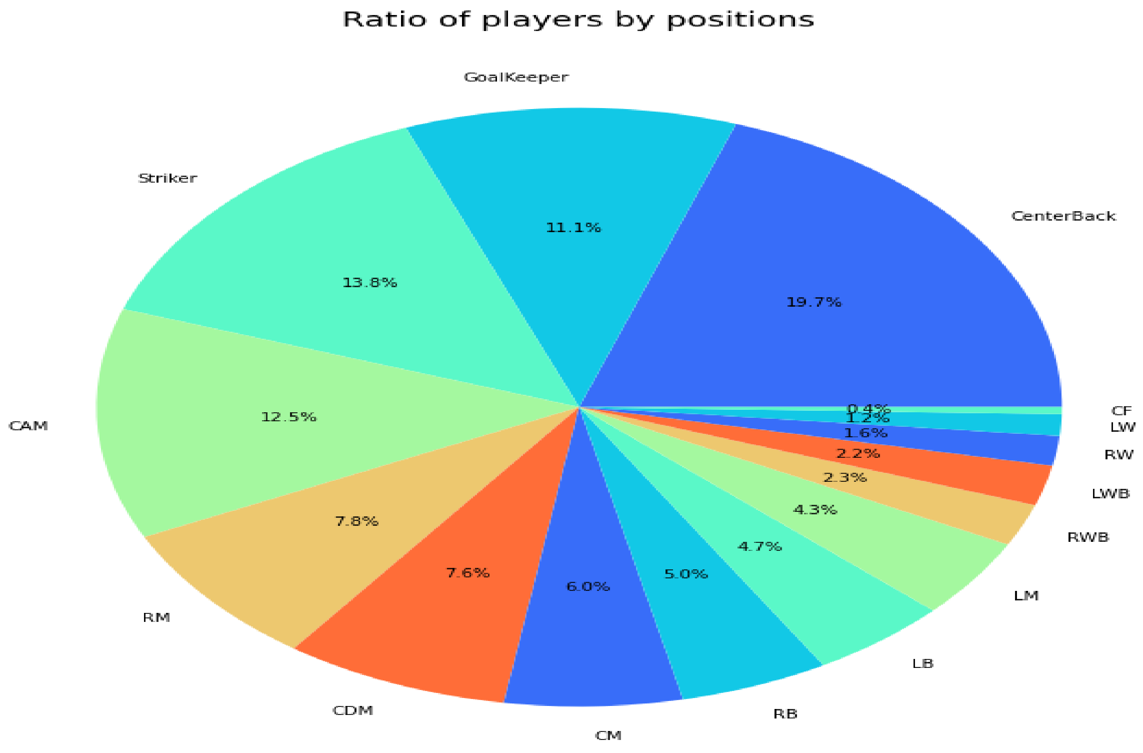
This is a massive data set with info about 18,538 professional football players from around the globe. It has 89 columns with information on the likes of the player's biography, personal details, technical abilities and ratings on various related traits.

Fortunately the data set has no null values, so we don't have to do any significant data cleaning. Let us go ahead and do some data exploration to understand the data set in a more detailed fashion.

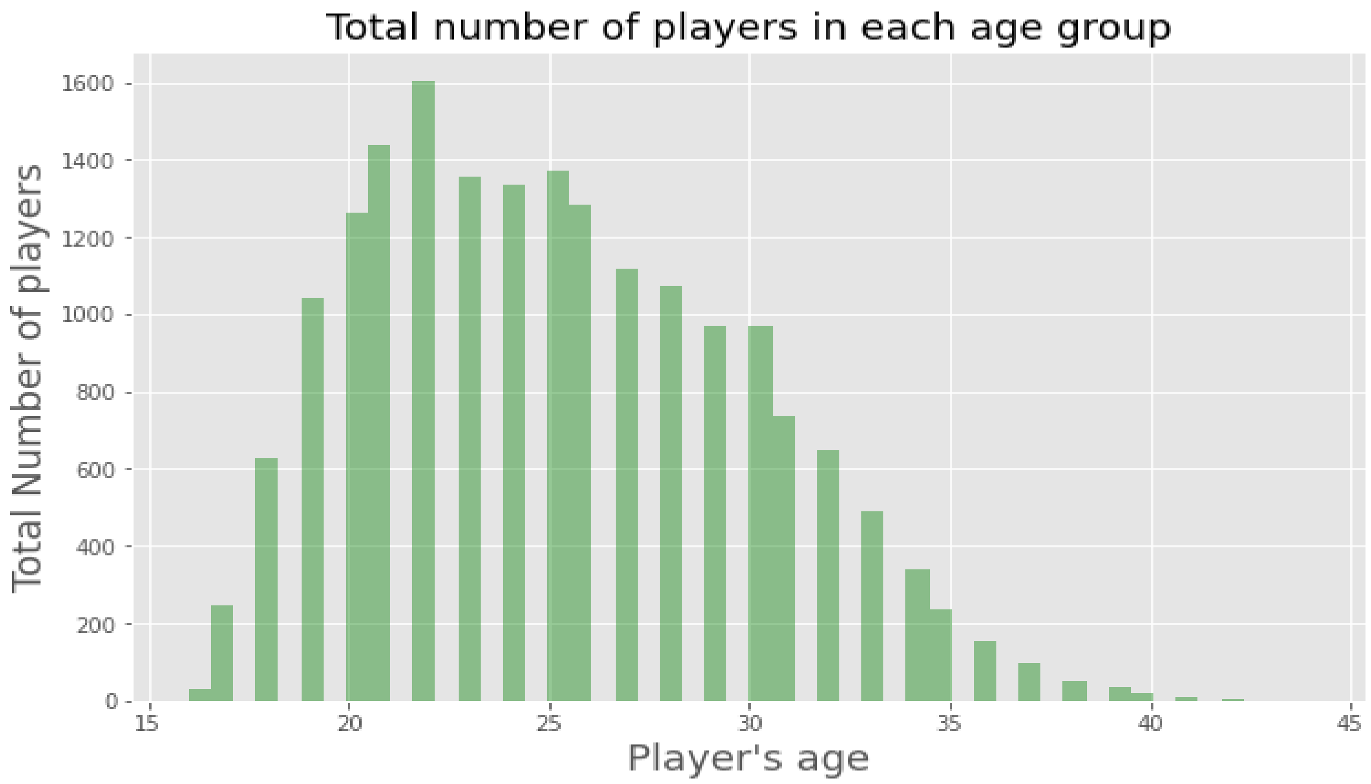
Here are few graphs to detail the information in our data set:



The above graph illustrates the number of players by their rating. The graph is perfectly balanced with normal distribution. Majority of the players have a rating between 60 and 70. So for our purpose, it would rather be easier to replace an outgoing player with that rating. But the total number of players goes down as the rating increases. So the number of options will go down as the quality of the player increases.



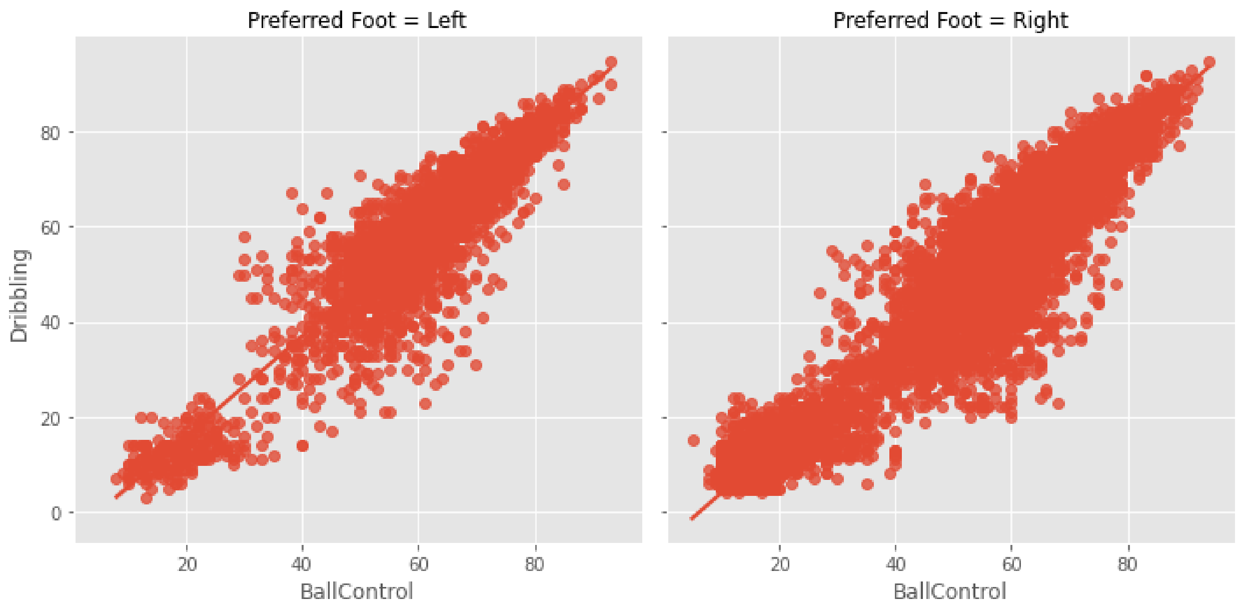
This Pie graph illustrates the percentage of players by their preferred position. Almost 1 out of every 5 players from our dataset are center backs. Strikers, Central Attacking Midfielders, and Goalkeepers are the next highest on the list. Usually football teams would like to have at least 3 players for each position on their roster and more than 3 for positions like center Back. But as we see from the above pie chart, there are very few players in the LW and RW positions. So it is very hard for all the teams to have 3 players in these positions and at the same time it will also be very hard to replace an outgoing player in these positions. On a few occasions, teams might even be required to pay well over the odds for players in these positions because of the imbalance in the supply and demand.



The above plot illustrates the number of players in each age group. Majority of the players are between 19 and 27. So for our cause, we will have more options to replace a player within these age groups.

Checking the accuracy of the Data set as fans of the game

Before working with this data set, we would like to check the accuracy of the data set with few tests of our own. Usually players with better ball control have good dribbling skills. In a nutshell these traits are correlated. So we have decided to do regression plots on these traits to see what our data would reflect. Also let's use this as an opportunity to compare right and left footed players. If the data is accurate these traits should be positively correlated.



The above graph is a massive testament to the accuracy of the data set. From the plot, Players with better ball control also happen to have high dribbling skills.

Also, the correlation map we plotted yielded results that support the accuracy of the dataset. All the goalkeeper based ratings have negative correlation with shooting related ratings (which are usually good for forwards).

Youth Guide

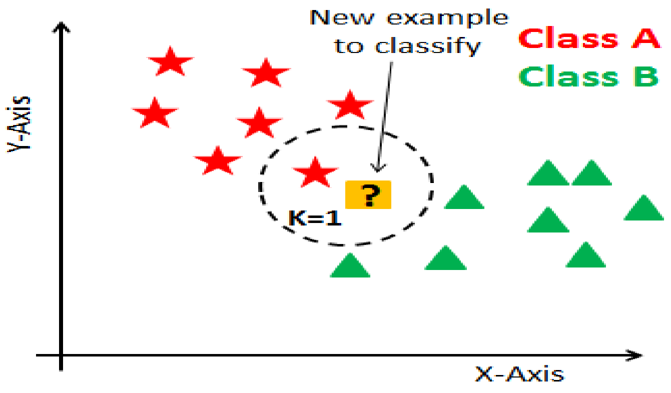
As I have stated earlier, we created a youth guide to present our user with the list of the best young talent from around the globe. Sometimes, teams/gamers would like to build their teams with an eye on the future. For such causes, this list could come in very handy.

Our approach

Since we will primarily be using the stats to compare and find players, we dropped all the categorical information and other useless information for this type of analysis like Player’s wage and players’ status with his national team. We used the k-Nearest Neighbours algorithm to build this tool , as it would help us figure out the K closest neighbors (K similar players in our case) to the input provided.

K-Nearest Neighbors

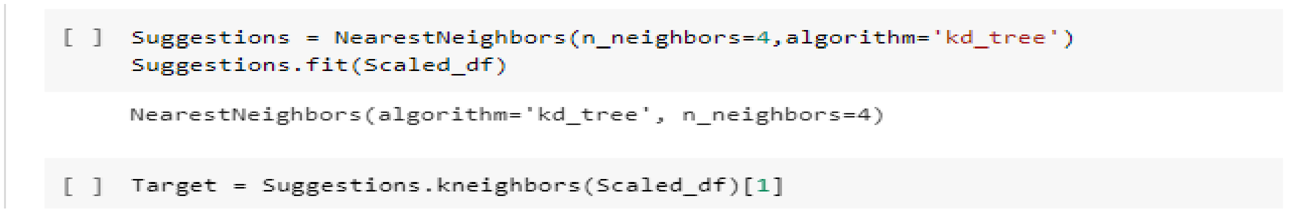
K-Nearest Neighbors or (KNN) is a machine learning algorithm which we used for this particular project. K-nearest neighbors is an algorithm usually run to classify new inputs as part of a specific group. The particular branch of KNN we used involves using the k-d tree. K-d tree constructs a Euclidean plane of k-dimensions, and uses quantitative descriptors to place each element on a plane. KNN then inputs new points into the plane and uses the preselected closest amount of points nearest to that particular point to identify that point into a group. An example of this is shown in the image below.



The image above shows an example where a new point is placed and then the closest point is used to identify it. In the case shown above, the new point would be classified as a red star in Class A. KNN also only uses quantitative data, therefore, when we used it, we had to drop all categorical data related to each player.

However, for the purposes of our project we use KNN in a different way, rather than use the surrounding points to classify our input (player we want a recommendation for), we actually retrieve the points nearest. Since these points should ideally be very similar, they should provide very good recommendations for the player we input. So we use KNN in this creative way to find recommendations rather than actually classify the player we input.

A snapshot of the code we used for KNN is below:

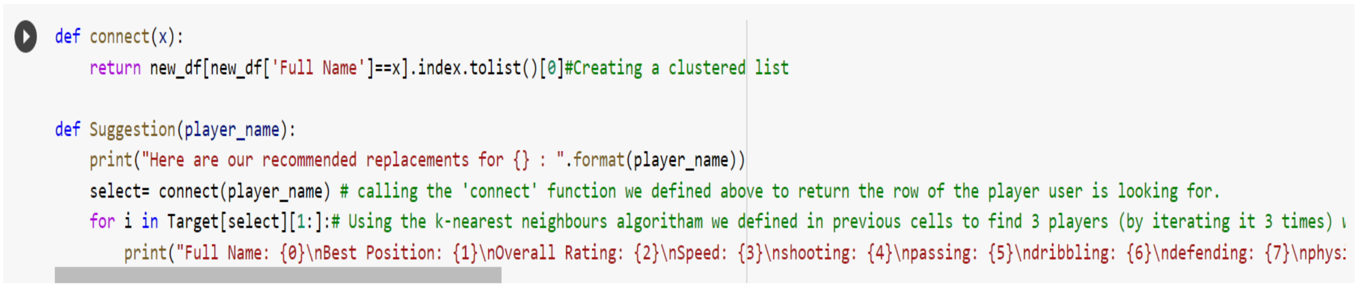


```
[ ]  Suggestions = NearestNeighbors(n_neighbors=4,algorithm='kd_tree')
     Suggestions.fit(Scaled_df)

     NearestNeighbors(algorithm='kd_tree', n_neighbors=4)

[ ]  Target = Suggestions.kneighbors(Scaled_df)[1]
```

In the code above we can see where we identified the k-d tree using the algorithm input. Additionally, we can see how many results we returned. N_neighbors is defined as 4. This means that it will return 4 results, however, the closest result will actually be the point itself. So we will remove this from the results when we print and the function should print 3 results in the end.

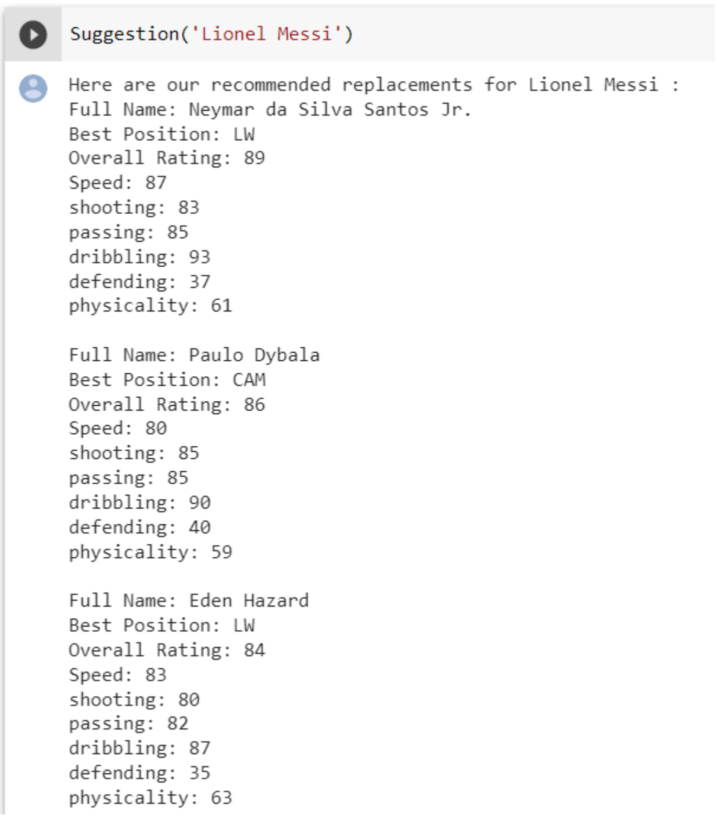


```
def connect(x):
    return new_df[new_df['Full Name']==x].index.tolist()[0]#Creating a clustered list

def Suggestion(player_name):
    print("Here are our recommended replacements for {} : ".format(player_name))
    select= connect(player_name) # calling the 'connect' function we defined above to return the row of the player user is looking for.
    for i in Target[select][1:]:# Using the k-nearest neighbours algoritham we defined in previous cells to find 3 players (by iterating it 3 times)
        print("Full Name: {0}\nBest Position: {1}\nOverall Rating: {2}\nSpeed: {3}\nshooting: {4}\npassing: {5}\ndribbling: {6}\ndefending: {7}\nphys
```

In this snapshot, we created two separate functions and linked them with the KNN algorithm from our previous snapshot.

Results



```
Suggestion('Lionel Messi')
```

```
Here are our recommended replacements for Lionel Messi :
Full Name: Neymar da Silva Santos Jr.
Best Position: LW
Overall Rating: 89
Speed: 87
shooting: 83
passing: 85
dribbling: 93
defending: 37
physicality: 61

Full Name: Paulo Dybala
Best Position: CAM
Overall Rating: 86
Speed: 80
shooting: 85
passing: 85
dribbling: 90
defending: 40
physicality: 59

Full Name: Eden Hazard
Best Position: LW
Overall Rating: 84
Speed: 83
shooting: 80
passing: 82
dribbling: 87
defending: 35
physicality: 63
```

The User will be required to enter the full name of the player to get the desired results. In this case, we entered Lionel Messi's name and got three players with the highest similarity to him in terms of style and rating. If you go out and ask soccer fans randomly to list three players that are similar to Messi (based on a combination purely based on style and rating), We are pretty sure that 80-90% of them are gonna go with our output.

Conclusion

If the teams/gamers are only looking to replace aged players with young and upcoming talent, unfortunately our tool doesn't support them. Our tool just lists out 3 players with similar style and rating. So to upgrade our code, we can also build a young player recommender system to only print out the names of young players with similar style and similar future potential.

Also to upgrade our tool, we can provide the user with more options to choose from, to find the exact kind of player they are looking for.

Citations

Navlani, Avinash. "KNN Classification Tutorial Using Sklearn Python." *DataCamp*, DataCamp, 2 Aug. 2018, https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn.