Soccer Player Worth Prediction

Justin Quinlan

May 7th, 2020

Abstract:

On August 3rd, 2017 a new transfer record fee had been set in the soccer world. This was the transfer of Neymar from Barcelona to PSG. The transfer fee was a staggering 222 million Euros ($241,402,800), which is more than double of the previous record of 105 million Euros set back in 2016. This type of money can be used to finance and improve many different areas of a team. This made me start to wonder if it would be possible to predict soccer player market value based on their season stats. This could be used to help teams with lower budgets grow and have a better chance to compete against those with seemingly unlimited funds.

Introduction:

The sport of soccer is the most popular sport in the world. There are hundreds of leagues around the world. The players a team has, makes all the difference. In order to win teams will spend a lot of money in order to get specific players to their teams. The players in the top five leagues are more expensive than players outside of the top five.

My goal is to build a model that can predict the value of a player based on their statistical performance in a season. For example, say Manchester United want to buy the contract rights to a player from another team, so that they sign the player. If they ran his statistics through my algorithm it would give them an accurate idea of his market value. It would then be their decision if they wanted to attempt to low ball their offer or offer more than the player is worth. This could also be used in reverse. A team could use this algorithm to ensure that they are getting a fair amount of money for a player they are selling. I would like to include a clustering analysis that shows which players are statistically similar as well. My hope for this system is that it will help teams with lower budgets strengthen their squads or invest for the future.

I have reached out to the New England Revolution to see if they would be interested in my Idea. The New England Revolution is a soccer team in the MLS (United States soccer league) that represent New England and are based out of Massachusetts. For the last couple of years, they haven't played bad, but they haven't improved. They practically finished in the same placement for the last 4 -5 years. This year they have a new coach, who has proved to be very good in his past. I want to see if my system can help them improve their squad. I am still waiting for their response.

Team	Season	Placement
Revolution	2015	11th out of 20
Revolution	2016	14th out of 20
Revolution	2017	15th out of 22
Revolution	2018	16th out of 24
Revolution	2019	14th out of 24

Table 1: Placements of the New England Revolution in the MLS during each season.

Data Collection

Data scraping was a significant part of this project. In order to create my algorithm, I first needed a dataset to analyze. My ideal dataset did not exist, so I had to create my own. The dataset was created from data scraping three different websites. Each had their own purpose.
TransferMarkt.co.uk: This website was scraped with the python package BeautifulSoup4 in order to obtain the player values for each season. BeautifulSoup4 allowed me to request and download a specific websites html.
Fbref.com: This website was scraped with the R package known as R-vest in order to obtain basic player information. R-vest allowed me to go into the source code of Fbref.com and target specific symbols. The table I wanted to get information from was located inside of source HTML code but it was inside a comment. In order to get that information, I had to use R-vest to target the HTML comment symbols and remove them.
PremierLeague.com: This website was scraped with R-Selenium in order to obtain the statistics of each player. R-selenium automates a web browser. I needed R-selenium because the website was dynamic.

I gathered all the information on these sites that occurred between the period of August 2015 and May 2019. This period of time would give me four separate seasons of data. After merging all the data scraped, I was left with one large dataframe. This dataframe contained 41 attributes and 1987 players for a total of 81,467 observations.

Data Cleaning:

There were a number of tasks to perform in order to finalize my master dataset. There were a couple of issues I had to fix when I had all four of my datasets built. Certain Ascii symbols popped up in players' names when they were initially scraped. I had to go through each dataset to fix the names of certain players that had ascii symbols in them.

First, I replaced all the "NA" with zeros. The reason I did this is because this dataset should be a complete dataset. If a player has an "NA" under say "Goals", that would make sense. Not all players score goals in a season. Therefore, it makes perfect sense to replace the "NA" with a zero.

Second, in order to ensure that the players in the dataset had enough information, I removed all players that had less than 10 appearances in a season. There are 38 games in a season, and I didn't want to remove too many players.

The sport of soccer has 4 main positions: Forwards, Midfielders, Defenders and Goalkeepers. All of the positions other than the Goalkeeper have essentially subcategories of more specific positions. For example, the subcategories that fall under the Midfielder position are “Attacking Midfielder”, “Central Midfielder”, “Outside Midfielder”, and “Defensive Midfielder”. Not all positions are played as many teams use different formations. It is all up to the Manager or the Coach.

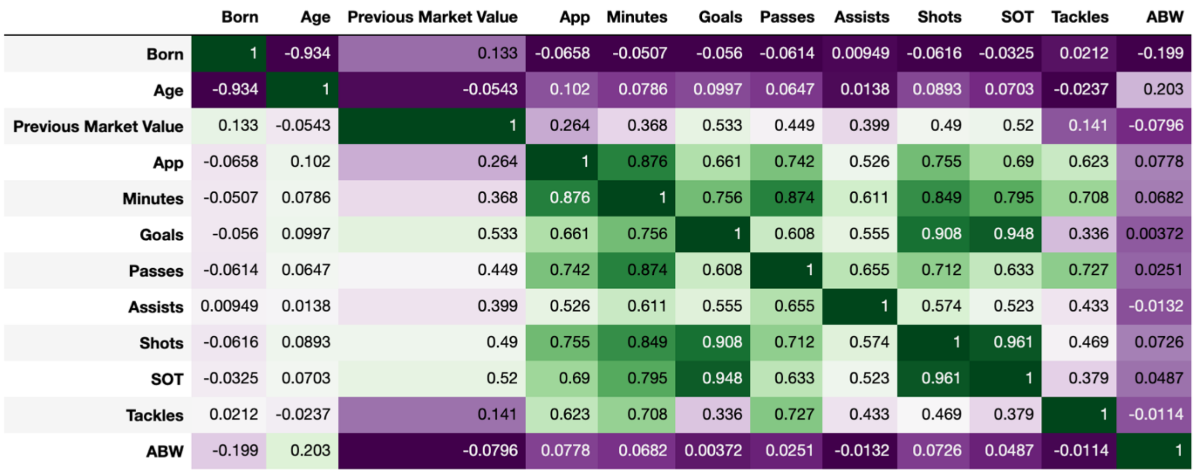
After going through the data, I had, I realized that there was another thing I to do. I had to remove the Goalkeeper position from my dataset. Goalkeepers have completely different statistics then those of the other three above. They would only create noise and cause issues in any of the methods I use for the analysis.

The final dataset would be a total of 41 attributes such as “goals scored” , “passes made” and “fouls committed”. There are 1555 players and a total of 63,755 observations.

Lastly, I separated the master dataset into three different sub datasets. Each sub dataset covered one of the three positions left. All three datasets have the same common stats. For example, all three datasets have both the “goals scored” and the “passes completed” statistics. Then depending on which position they are for, there are separate stats in the subset. For example, the midfielder subset is the only one with the “crosses performed” statistic.
The “Forwards” subset ended up with 18 attributes and 365 players for a total of 6,570 observations. The “Midfielders” subset ended up with 25 attributes and 574 players for a total of 14,350 observations. Finally, the “Defenders” subset ended up with 23 attributes and 526 players for a total of 12.098 observations. All three subsets would make a total of 33,018 observations.

Preliminary EDA

I ran a correlation matrix across each csv file I created. The correlation matrix gave me a quick understanding of the dataset and how well each attribute correlated with each other. There were times when certain attributes had high a correlation with each other that did surprise me. Other times there were disappointments as well as correlations that were to be expected. A correlation that surprised me was the positive correlation between “tackles” and “passes”. Meanwhile the negative correlation between “assists” and “errors leading to goal” was completely expected.



	Born	Age	Previous Market Value	App	Minutes	Goals	Passes	Assists	Shots	SOT	Tackles	ABW
Born	1	-0.934	0.133	-0.0658	-0.0507	-0.056	-0.0614	0.00949	-0.0616	-0.0325	0.0212	-0.199
Age	-0.934	1	-0.0543	0.102	0.0786	0.0997	0.0647	0.0138	0.0893	0.0703	-0.0237	0.203
Previous Market Value	0.133	-0.0543	1	0.264	0.368	0.533	0.449	0.399	0.49	0.52	0.141	-0.0796
App	-0.0658	0.102	0.264	1	0.876	0.661	0.742	0.526	0.755	0.69	0.623	0.0778
Minutes	-0.0507	0.0786	0.368	0.876	1	0.756	0.874	0.611	0.849	0.795	0.708	0.0682
Goals	-0.056	0.0997	0.533	0.661	0.756	1	0.608	0.555	0.908	0.948	0.336	0.00372
Passes	-0.0614	0.0647	0.449	0.742	0.874	0.608	1	0.655	0.712	0.633	0.727	0.0251
Assists	0.00949	0.0138	0.399	0.526	0.611	0.555	0.655	1	0.574	0.523	0.433	-0.0132
Shots	-0.0616	0.0893	0.49	0.755	0.849	0.908	0.712	0.574	1	0.961	0.469	0.0726
SOT	-0.0325	0.0703	0.52	0.69	0.795	0.948	0.633	0.523	0.961	1	0.379	0.0487
Tackles	0.0212	-0.0237	0.141	0.623	0.708	0.336	0.727	0.433	0.469	0.379	1	-0.0114
ABW	-0.199	0.203	-0.0796	0.0778	0.0682	0.00372	0.0251	-0.0132	0.0726	0.0487	-0.0114	1

Figure 1: Correlation Matrix of the Forwards subset. The correlation matrix above is showing how each attribute correlates with the other. Green is positive correlation, while purple is negative correlation.

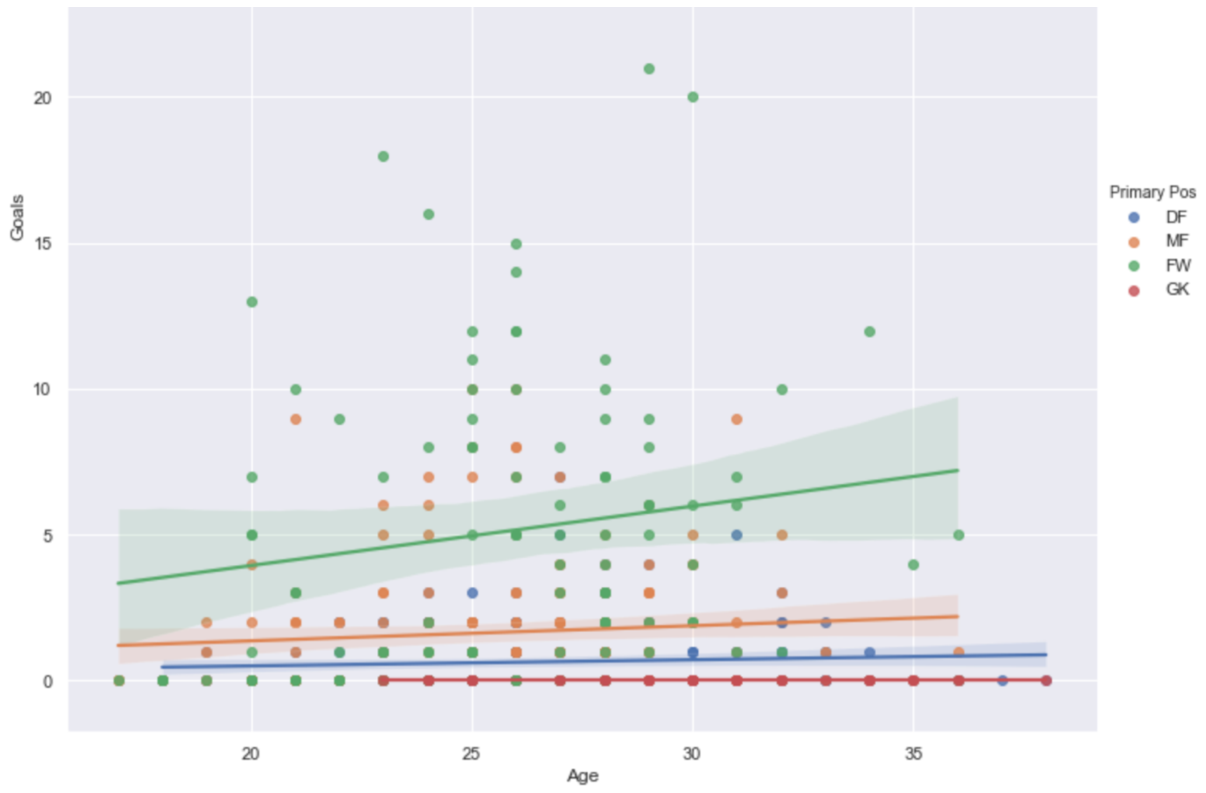


Figure 2: Scatterplot showing the average number of goals scored by age and position. This graph shows the trend of the average amount of goals for each position by age. The dark shaded line is the linear regression model fit. While the translucent lines around the dark shaded line are a confidence interval generated for the estimate.

Methods:

Random forest is a highly accurate ensemble technique that includes using multiple decision trees to create the best possible outcome. It can be used for both regression and classification. In my case I used it for regression in order to predict the market value of players based on their season statistics. I had three different subsets and I ran the same random forest regression model across all three. The specific package I used was "sklearn.ensemble.RandomForestRegressor". The test size was 0.3 or 30%. I made sure to set the "max depth" factor in the model to "None". This ensures that all the nodes are expanded until all leaves contain less than the "min_samples_split" samples. I set the "min_samples_split" factor to five. All this does is set a minimum number of samples required before splitting an internal node.

Cross Validation is a technique that involves putting aside a part of your dataset that has not been trained on the model. Later, you test your model on this piece to gauge the model performance. I used this to test model accuracy. "CV" below in table 2 is just short for "Cross Validation". The cross-validation minimum is describing the lowest accuracy rating the model produced. Meanwhile the cross-validation maximum is describing the highest accuracy the model produced. The cross-validation mean shows the averages of the accuracy ratings produced across an entire subset. For reference, take a look at the "CV_Mean" in "Forwards" located in table 2. The output here of 0.814 indicates that the average accuracy of the run across "Forwards" is about 81.4%. Cross-validation standard deviation was very important to the understanding of this model. In order for me to be able to make my assumptions that the "CV_Mean" equals model accuracy, the standard deviation needed to check out. As you can see the standard deviation for all three subsets are low. This is important because this tells me that most of the model performance numbers are close to the average. Therefore, I can make the assumption that the "CV_Mean" is the model accuracy.

Subsets	CV_Mean	CV_Std	CV_Min	CV_Max
Forwards	0.814	0.055	0.716	0.877
Midfielders	0.834	0.064	0.747	0.919
Defenders	0.749	0.056	0.649	0.807

Table 2: Cross validation model performance outputs for each subset.

Visualizations:

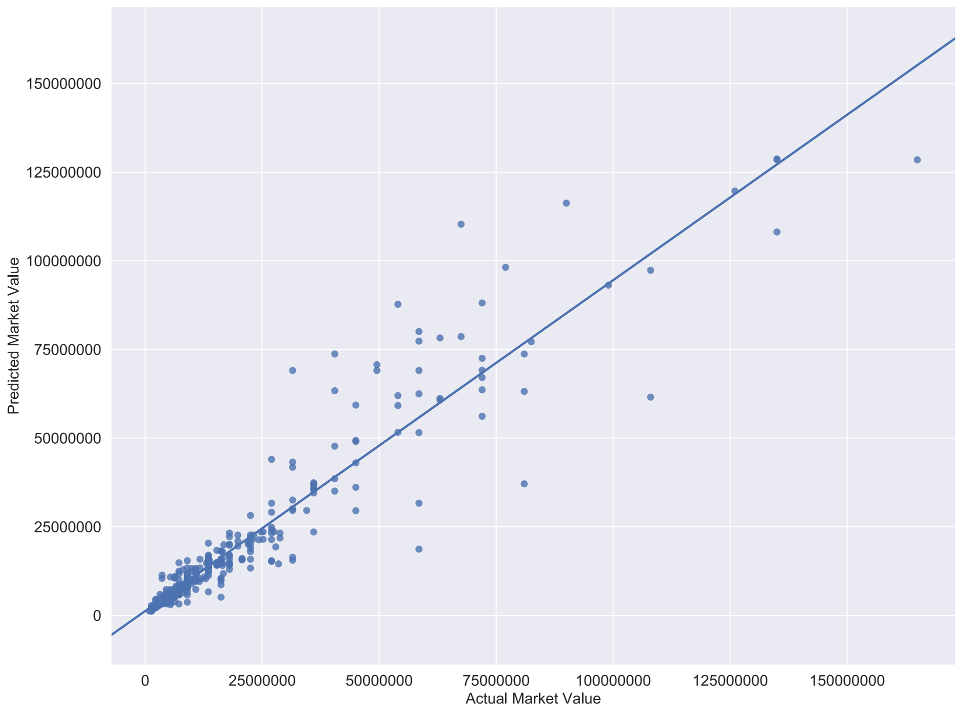


Figure 3: Regression plot of the actual market value of players vs the predicted market value of Forwards. In the graph you can clearly see the dark shaded line. That line is the linear regression model fit.

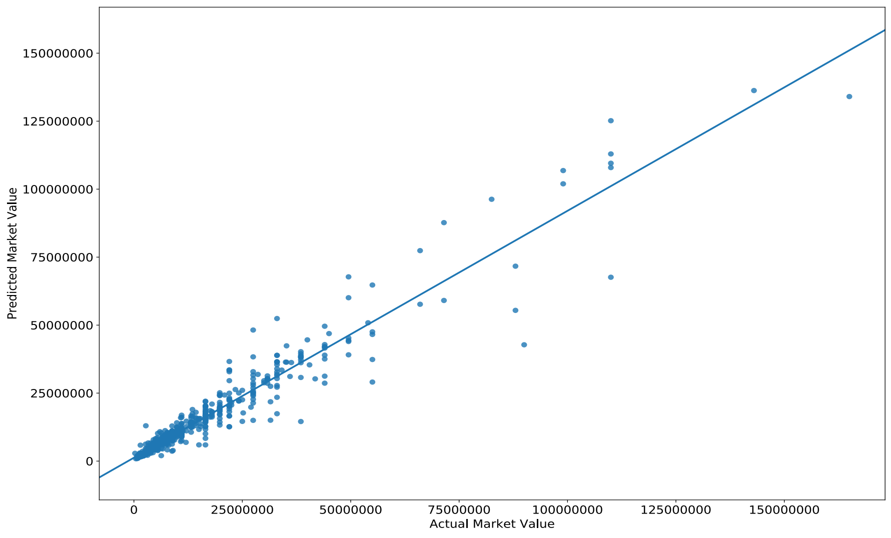


Figure 4: Regression plot of the actual market value of players vs the predicted market value of Midfielders. As you can see the values of midfielders are generally just as high as those of forwards.

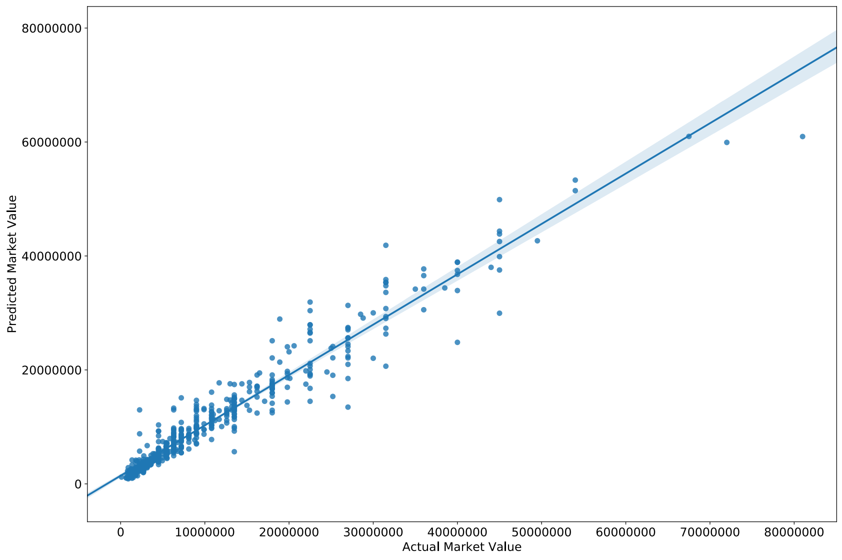


Figure 5: Regression plot of the actual market value of players vs the predicted market value of Defenders. If you take a glance difference in values, you can see that the defender position is

valued less than the other two positions above. The translucent lines in the graph above are a confidence interval generated for the estimate linear model regression fit line.

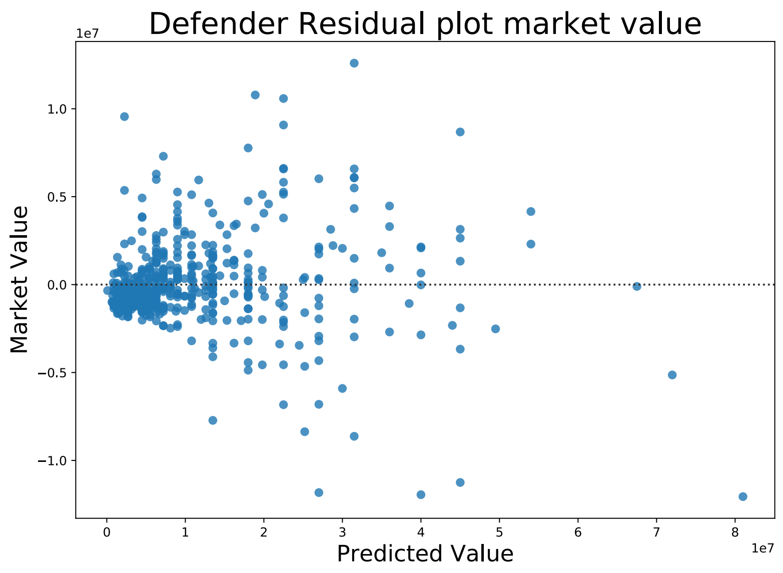


Figure 6: Residual plot of predicted vs actual market values of Defenders. This residual plot and the ones below it show that most of the points are close to zero, this reinforces how accurate my model is.

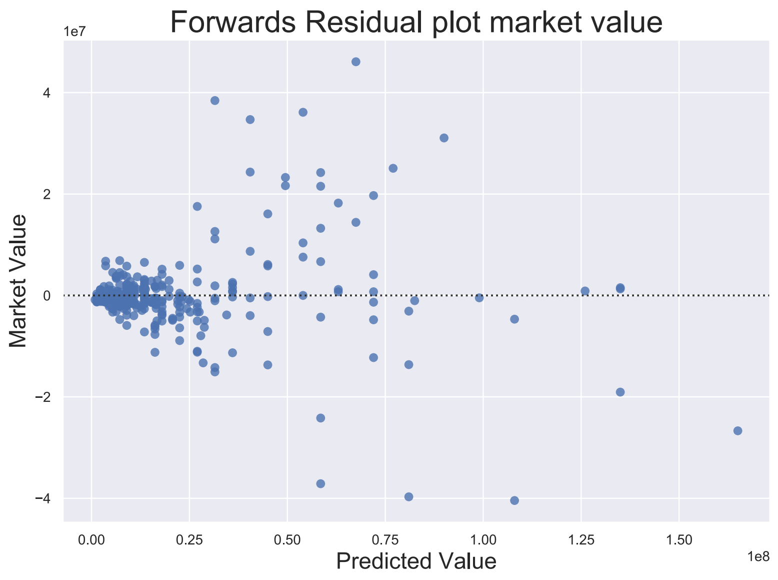


Figure 7: Residual plot of predicted vs actual market values of Forwards.

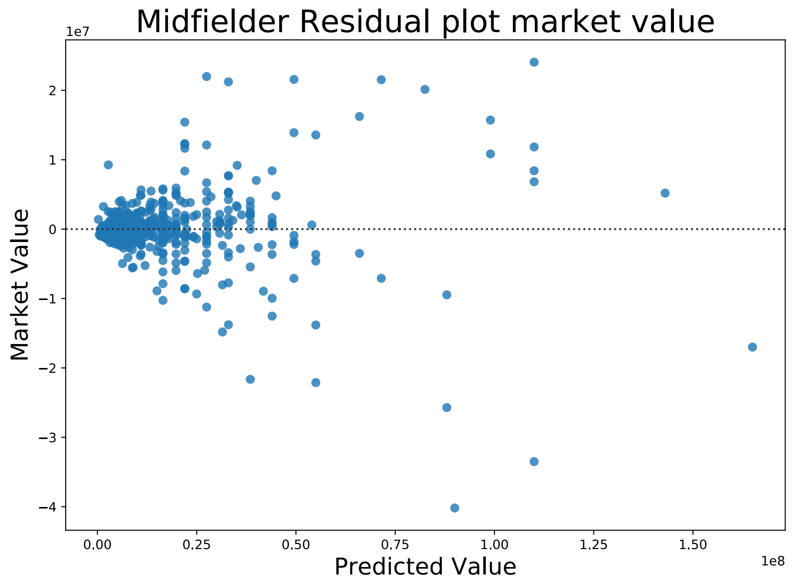
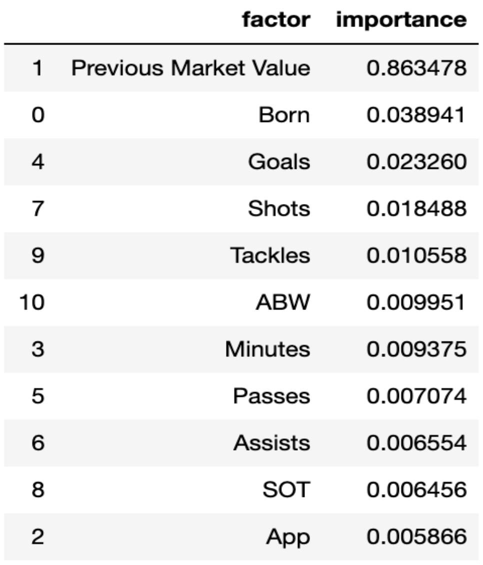


Figure 8: Residual plot of predicted vs actual market values of Midfielders.

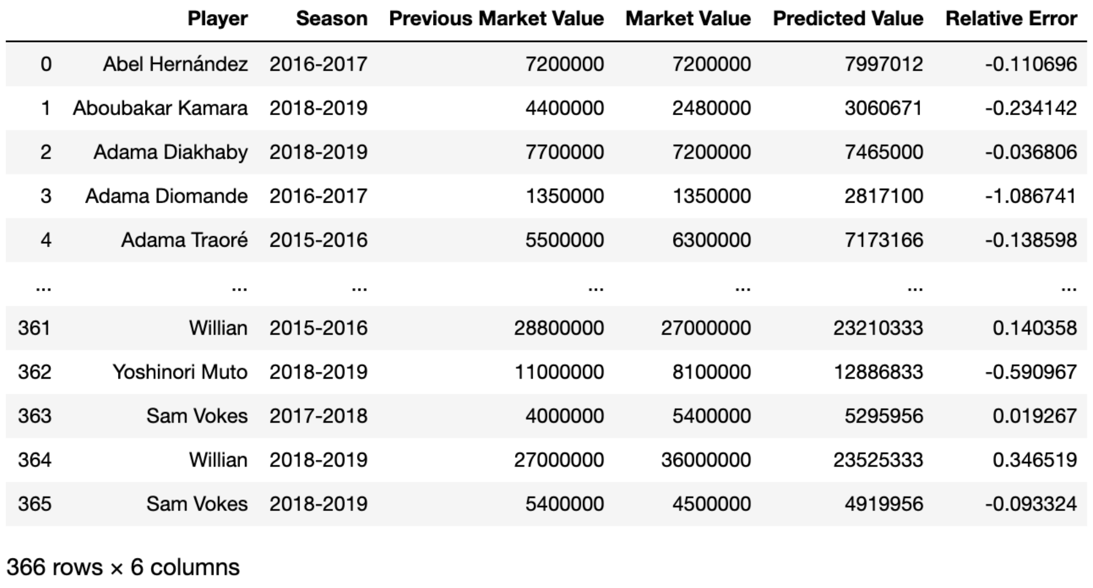
Relative feature importance is another feature of the Random forest. Using it allowed me to see the most contributing features for the classifier. The picture below is the Relative feature importance table output for the analysis of the forwards.



	factor	importance
1	Previous Market Value	0.863478
0	Born	0.038941
4	Goals	0.023260
7	Shots	0.018488
9	Tackles	0.010558
10	ABW	0.009951
3	Minutes	0.009375
5	Passes	0.007074
6	Assists	0.006554
8	SOT	0.006456
2	App	0.005866

Table 3: Relative feature importance table output for the “Forwards” subset. This table shows the importance of each factor that is put into the random forest model. As you can see the “Previous Market Value” is the most important factor by a lot. This reinforces the fact that historical data can and always will be very powerful.

I found the relative error of the random forest model by subtracting the predicted value from the actual market value and then dividing that by the actual market value. Finding this allowed me to get a better idea of the error in my model and figure out what was going on. For the most part it allowed me to see where all of the large error was. All of the large relative error was negative which indicated that it was over-predicting the values. After looking further into the players whose values were over-predicted, I was able to determine that they were age based. My model has difficultly predicting the values of players who are under the age of 21 and over the age of 32. When players are under 21 ,they normally haven’t had enough minutes to be measured the same. Granted they are young, so they normally have a higher market value. They haven’t yet had the chance to show their potential and because of the unknown mystery of the player, their values go up. Meanwhile players over the age of 32 will have their values decrease. Even if they are putting up good statistical numbers their values will drop. They essentially have less time to play before retirement and they have no potential remaining so the value drops.



	Player	Season	Previous Market Value	Market Value	Predicted Value	Relative Error
0	Abel Hernández	2016-2017	7200000	7200000	7997012	-0.110696
1	Aboubakar Kamara	2018-2019	4400000	2480000	3060671	-0.234142
2	Adama Diakhaby	2018-2019	7700000	7200000	7465000	-0.036806
3	Adama Diomande	2016-2017	1350000	1350000	2817100	-1.086741
4	Adama Traoré	2015-2016	5500000	6300000	7173166	-0.138598
...	...	...	...	...	...	...
361	Willian	2015-2016	28800000	27000000	23210333	0.140358
362	Yoshinori Muto	2018-2019	11000000	8100000	12886833	-0.590967
363	Sam Vokes	2017-2018	4000000	5400000	5295956	0.019267
364	Willian	2018-2019	27000000	36000000	23525333	0.346519
365	Sam Vokes	2018-2019	5400000	4500000	4919956	-0.093324

366 rows × 6 columns

Table 4: Shows the top and bottom of the new table I created to get a better understanding of how accurate my model outputs were.

Future Work:

I am happy with my project and how far I have gotten. There are a couple of things that I want to add in the future. I need to figure out a way to account for inflation in my model. Player values are always expected to rise just like everything else because of inflation. I also want to add more attributes to help identify if the player had a “breakout season” or was “injured”. These attributes would just be binary with 0 = No and 1 = Yes. This would allow me to hopefully remove the most influential factor of “Previous Market Value” to see if my model would perform the same, worse or better. Lastly, I would like to create a clustering algorithm that identifies players with statistical similarities.

Conclusion:

When I first chose this project back in September 2019, I had no idea the amount of work I would put into it. I have learned how to do so many things in this project alone. Just to name a few. I can now data scrape in both R and Python, efficiently clean data, read html, create great visualizations and sometimes even create interactive ones. There are a couple dataset uploads to Kaggle in the near future. I will first be uploading my final dataset. My final dataset is the one I used for this project. It would be very interesting to me to see how others use it. The other uploads would include a dataset with the players I dropped from the final and a separate goal-keeper dataset. The link to a similar report can be found in the references under “Predicting Market Values”. That report helped me accomplish this project and gain a better understanding about random forest models. The creator of this report got a result of 75% accuracy. There are differences in our reports, I modified the random forest model he made, to better fit my project. I used four different seasons of one league, while he used one season across many leagues. He did not separate the positions before analysis, while I did. He also didn't use relative error or create interactive graphs. I've accomplished my goal. I wanted to see if I could get a better accuracy rating then the report I just mentioned, and I did just that. I am glad I chose this project and I plan to work on similar projects after graduation.

References:

My Github:
https://github.com/JBQ3/Data-Science-Caspstone

Column meanings:

Player: name of the player

Squad: Team the player is on for that season

Nation: Nationality of the player

Born: Year player was born in

Age: How old the player is

Position: Can be one of the three below:

1. **DF** = Defensive player
2. **MF** = Midfield player
3. **FW** = Forward or Attacking Player

ABL: Number of Aerial Battles Lost

ABW: Number of Aerial Battles Won

APP: Number of Player Appearances in the season

Minutes: Total number of minutes played

Assists: Number of Assists a player created within the season

Blocks: Blocked shots

Clears: ball clearances

Corners: number of corners taken

Cross: Number of times the player crossed the ball

ELG: Errors lead to a goal

FKGoal: Number of goals scored from a Freekick

Fouls: number of fouls committed

Goals: Number of Goals scored

HeadGoal: Number of goals scored from a header

HeadClear: Number of ball clearances from headers

HitPost: Number of times a shot hit the post (woodwork or crossbar)

Interceptions: Number of times a pass is intercepted by the opposing team

Last Man: Last defender successfully tackling opposition on the attack

Misses: Number of shots that have missed the goal

Offsides: Number of times a player is called offsides

Passes: Total number of passes

PenGoal: Goals scored from penalties

Red: Number of red cards a player has received

Shots: Total number of shots a player took

SOT: Total number of shots on target a player took

SubOff: Number of times a player is substituted off the field

SubOn: Number of times a player is substituted onto the field

Tackles: Number of times a defending player challenges a player with the ball

ThrBall: Number of through balls a player makes to his teammates

Touches: Number of touches a player gets

Yellow: Number of yellow cards a player gets

Research:

Predicting market values:
https://github.com/pbarcellona/footballer-market-value-model/blob/master/Model%20workbook.ipynb

Measuring Player Similarity:
https://towardsdatascience.com/which-nba-players-are-most-similar-machine-learning-provides-the-answers-r-project-b903f9b2fe1f

Prediction with neural networks:
https://medium.com/@robertjohn_15390/simple-housing-price-prediction-using-neural-networks-with-tensorflow-8b486d3db3ca

Clustering in Python:
https://towardsdatascience.com/an-introduction-to-clustering-algorithms-in-python-123438574097

https://thevi5ion.wordpress.com/2017/07/13/classifying-nba-players-using-machine-learning-in-python/

Web scraping Research:
https://www.fernandomc.com/posts/using-requests-to-get-and-post/
https://codeburst.io/web-scraping-101-with-python-beautiful-soup-bb617be1f486

NBA salary Prediction:
https://towardsdatascience.com/nba-salary-predictions-4cd09931eb55

https://www.goal.com/en-us/news/the-100-most-expensive-football-transfers-of-all-time/ikr3oojohla51fh9adq3qkwpu

Data Scraped Sources:

Basic Player statistics:
https://fbref.com/

More Descriptive player statistics:
https://www.foxsports.com/soccer/premier-league

Player market values:
https://www.transfermarkt.co.uk/premier-league/marktwerte/wettbewerb/GB1