Machine Learning Final Project

SOCCER

PREDICTING THE LIKELIHOOD OF A GOAL

XAVIER TORRES , BRANDON SUNG , GARRETT RAMOS ,
AYUSSH AHUJA AND YUVRAJ SANJAY LULLA .

Project Scope

Ayussh Ahuja and Yuvraj Sanjay Lulla

The goal of our project was to predict the likelihood of a goal being scored in soccer. As more and more data becomes available, our team found that it would be possible to use certain aspects of the shot itself to predict the likelihood of similar shots being converted into goals, by leveraging Machine Learning Algorithms.

Historically, data science has been used in soccer to predict the winning team in a match, number of goals scored, and also the winner of a particular trophy (i.e. the world cup). However, soccer is a highly unpredictable sport and “weaker” teams often triumph over “stronger” ones, “In the premier league 2015/2016 season, we had a very unexpected champion, and their probability to win the title at the beginning of the season was one to five thousand.” (Toward Data Science). However, this hasn’t swayed soccer fanatics from trying to predict every aspect of a game. With sports betting and fantasy football gaining immense popularity in recent years, there have been various approaches to implement data analysis in soccer. More importantly, various sports clubs now hire data scientists to provide in-depth analysis of the game. Data is used by coaches to understand how an opposing team plays, and also how their own team performs. While a majority of models focus on predicting the outcome of a game, our team aims to go even deeper, by predicting the result of individual shots. By analyzing various factors like distance to goal, shot angle, body part used, etc, we hope to evaluate whether a particular shot has a higher chance of resulting in a goal. This could be extremely useful to coaches for tactical analysis, but could also be pretty interesting for soccer fans who love to analyze the finer details of the game.

Stakeholders Yuvraj Sanjay Lulla

Concerning who this project benefits, there are a number of stakeholders that could benefit from the results of these algorithms.

- **Soccer Betting Groups and Fantasy Football Players** - With millions, if not billions of fans around the world, there is a massive target market of individuals who are obsessed with having the ability to forecast how the game unfolds. This belief is cemented by the expansive presence of sports betting agencies globally. Leveraging such predictive algorithms to predict a players/team performance would allow for data backed bets. Additionally, there are also a number of people who play online games like "fantasy football". This allows for soccer fans to create their own teams composed of players in a specific league, with real-life performance influencing the points they earn in-game. "Research shows that fans are interested in the integration of match data and analysis, and these insights will help tell more stories about Premier League matches, providing fans a data-rich experience" (Oracle). This data would allow people to interpret and forecast results and subsequently place specific players in their teams to score the most points amongst their friends.
- **Soccer Clubs and International Soccer Teams** - With over 3903 professional clubs in over 201 countries , there is an expansive target market in terms of clubs and organizations that could leverage the results of this project. This could be on both the level of an individual player and the tactics of the team as a whole. Soccer clubs have a massive integrated network of data scientists that are constantly collecting data from each game to educate their players on their performance. The club could use these predictive algorithms to devise specific tactics that would ultimately create the highest probability

for a goal. For instance, if the algorithm found that the highest likelihood of a goal was from 10 meters away at a 10 degree angle, the team could ensure that the strategy in play would allow the ball to eventually be near this position.

- **Football Associations and Regulatory Authorities -** With the advent of video assisted referees and constant playbacks, technology continues to be further integrated into Soccer, ensuring that the game is as fair as possible. For this reason, authorities could leverage these predictive algorithms and focus their efforts on areas with the highest frequency of observations.
- **Football Pundits -** An integral part of the soccer games is the half-time and post game analysis. Usually, reputable retired players and football "pundits" discuss the performance of each team in a game, leveraging multiple data visualization tools like xG. "Expected goals (or xG) measures the quality of a chance by calculating the likelihood that it will be scored from a particular position on the pitch during a particular phase of play (Whitmore). Similarly, these individuals could use the results of our algorithm to convey further information to their audience.
- **Training Academies -** Numerous emerging talents arrive into the "First Team's" (The version of the team that plays in its most competitive and prestigious league) of clubs around the world from the clubs very own training academy. These facilities breed talent that is eventually picked up by the coaches to play competitively. The importance of this data to such organizations is self explanatory. The results can be used to analyze the "Style of Play" ("The general behaviour of the whole team to achieve its attacking and defending objectives."(ResearchOnline)). of the team in question, and can thus influence

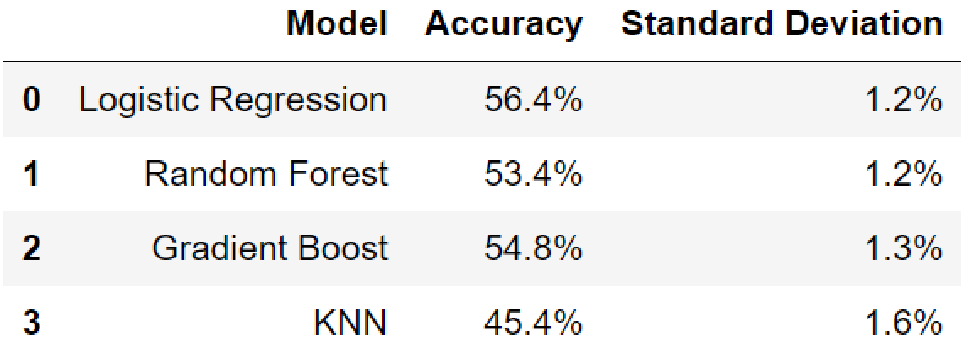
the operations of the training academy so that when the time comes, the younger player will fit well into the style of play of the “First Team".

History of Soccer and Machine Learning Yuvraj Sanjay Lulla

As mentioned earlier, the notion of incorporating data analytics and predictive algorithms in soccer has been around for years. While our project focuses on the conversion of individual shots, there are a number of projects completed by researchers around the world that focus on different aspects of the game.

- **Match Insights powered by Oracle Cloud -** “Oracle’s data and analytics and machine learning technologies will deliver these groundbreaking statistics in real-time to a global audience of billions each season.” The goal with this was to increase the excitement around each game by providing active data visualizations to make player performance more intuitive for the audience. This algorithm would combine historical performance and data from live-streams to create results. These will include Formation Prediction (to understand how teams will structure themselves), Win Probability and Momentum Tracker (Forecasting the likelihood of a goal from the team in possession) (Oracle).
- **Beating the Odds using Machine Learning - Arthur Caldes** - Using data from the Premier League (The most competitive league in England) over a 16 year period, Arthur used a number of Machine Learning Algorithms itself. These included K Nearest Neighbors, Random Forest, Boosted Trees and Logistic Regression. This data was collected on a website made for Soccer Match Betting called Odds Portal. This way, Arthur was able to factor in the odds of a specific team winning prior to a game, based on the betting odds provided (medium.com). Our team identified certain variables that added

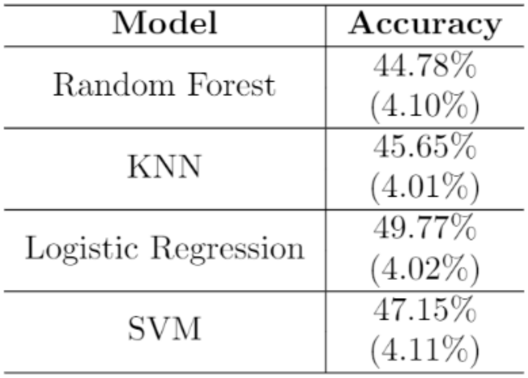
an interesting perspective to the algorithm. Firstly, the impact of the location on the result of the game (i.e. if the game was played at the teams home ground or at the opposing team ground). It was found that there was a high correlation between the likelihood of a win and the game being played at home. This is already common knowledge in the soccer world, as a greater number of fans in the stadium tends to boost player morale. Another interesting variable was the "win streak" variable, which factors in the momentum a team carries upon winning a number of games in succession. Below is a distribution of the accuracies the model was able to attain.



	Model	Accuracy	Standard Deviation
0	Logistic Regression	56.4%	1.2%
1	Random Forest	53.4%	1.2%
2	Gradient Boost	54.8%	1.3%
3	KNN	45.4%	1.6%

- **Predictive Soccer Model - George Pipis** - Similar to the previous mode, this model was aimed at predicting the results of a specific game between two specific teams. For instance, if Southampton were to play Chelsea, it would only refer to the data of these two teams. This was done using the H2O Machine Learning Algorithm as a feedforward Artificial Neural Network. It measured a number of in game performance related variables including corners, yellow cards and shots on target, to name a few. Similar to the previous example, these variables were again split between home and away games. The author mentioned that one of the ways to increase the accuracy of the model, would be to provide a higher weightage to more recent games to incorporate “form” (recent team performance) (Pipis).

- **Football Prediction - Matheus Kempa -** This model used KNN, Logistic Regression, Random Forest and Support Vector Machines**.** One of the interesting methods used in the data preprocessing was moving averages. This was done to factor in the recent performance of a specific team across their last 5, last 3 and last games. Matheus also used a variable importance plot to identify which variables had a higher impact on the model. Interestingly, the top 3 in this case were the crowd over the last three matches, the number of yellow cards and lastly, the crowd over the entire season (Kempa).



Model	Accuracy
Random Forest	44.78% (4.10%)
KNN	45.65% (4.01%)
Logistic Regression	49.77% (4.02%)
SVM	47.15% (4.11%)

Table 1: Mean and standard deviation of performance

Methods Employed and Overall Results

Yuvraj Sanjay Lulla

Method	Accuracy	Sensitivity	Specificity
Support Vector Machines	71.07%	69.79%	71.23%
Artificial Neural Networks	73.59%	67.26%	74.26%
Random Forests	78.35%	51.40%	81.50%

As mentioned, our team preprocessed the data and used 3 Machine Learning Methods to predict the likelihood of a goal. The results of these are attached above. As we can see, we had the highest overall accuracy from Random Forests (RF) at 78.35%, followed by Artificial Neural Networks (ANN) and then Support Vector Machines (SVM). Sensitivity, which predicts goals (as opposed to misses) in our program, had exactly the opposite, with SVM taking the top spot (69.79%), followed by ANN and then RF. It should be mentioned that our team prioritized overall accuracy over everything else. These high sensitivity figures were only possible through ROSE undersampling. Lastly, Specificity had relatively higher figures across all three methods, owing to the larger proportion of predicted misses in the data set. Random Forests has the highest at 81.5%, with ANN following in second and SVM third.

Data Preprocessing Garrett Ramos

Preliminary Cleaning

The original dataset consisted of 'event' data, consisting of information related to the unique events, or actions taken during the course of thousands of matches across Europe's major soccer leagues, and competitions. Each country had its own separate dataset, while data from the 2018 World Cup and 2016 European Championships had data compiled in separate datasets as well. While our specific focus was on shots taken during the course of these matches, the original data had a much larger scope, with information related to passes, defensive actions, duels, and more, specified by an event identification number indicating which type of event the observation was. Because our focus was on shots, and creating a model able to predict the likelihood of a goal, we removed the observations in our data related to other events that were not shots. Even still, there was information within the remaining observations which we did not require for our purposes. Information related to the post-shot location of the ball was also removed. While certainly valuable for alternative analysis, our goal with the data in hand, was to evaluate the likelihood of a goal *prior* to the actual shot. Post-shot information, such as where the shot ended up locationally to the goal, was considered irrelevant. After removing observations related to post-shot location, there were still multiple observations with a common event identification number. The event identification number signified a unique event during the course of the match. Even though there were multiple observations with a common event identification number, our understanding of the data was that there was valuable information in each of these observations that we would eventually want to concatenate into one complete row of data. As an example, if an event took place where a counter attack by the offensive team resulted in a goal, there would be three rows of data all with the same event identification number. The only difference between the three rows would be under the sub-event identification column, where one row would be labeled as a counter attack, the next row labeled as a shot with either the left foot, right foot, or head, and then the last of the three rows would be labeled as a goal as its sub-event identification. All three of these observations, while distinguished by three separate rows, were in fact, the same observation, the same event during the course of the soccer match. In addition to the various sub-event identifications associated with one unique event, there was also evidence of data imputation errors. The common errors included observations with x and y values of either 100 or 0, which we considered out of the realm of possible event locations. These observations were removed.

Having done all of the preprocessing up to this point in Excel, the data from all 7 separate datasets, including data from England, Germany, Italy, France, Spain, the 2018 World Cup, and the 2016 European Championships, were loaded into R Studio and combined into one master dataset. As mentioned, our primary objective during the cleaning process was to concatenate the various rows with various sub-event information into single observations based on the unique event identification numbers. Subsetting using the Base R function ‘duplicated’, we were able to identify which observations appeared multiple times with the same event identification number. After identifying these ‘duplicate’ rows, we were eventually able to merge the rows back to the master dataset by using the ‘merge’ function in R, and by utilizing the common key of the event identification number. By creating two new columns, labeled ‘goal’ and ‘counter’ we were able to retain the information of the ‘duplicate’ rows, but while reducing the information contained by those ‘duplicate’ rows to merely a 0 or 1 as the outcome of a binary variable depending on whether the observation was a goal or not, as well as whether the observation was a counter attack or not. Following this process, our dataset was reduced only to unique events, so that each row of the data corresponded to a single unique event identification number.

New Variable Creation

After finding the data cleaned to our satisfaction, we set out to create a couple of additional numeric predictor variables using some of the data we had. While likely correlated to the x and y variables of our data, we wanted to create numeric variables associated with both the angle of the shot, and the distance to goal of the shot, as a means of both better exploratory analysis, and potentially better model performance. Since both the x and y variables were percentages relative to the pitch length and width, we felt that a distance to goal metric, in a more defined measurement, such as meters, would be better conceptually for our data to include. Converting both the x and y variables from percentages into meters, we then used the euclidean distance formula to formulate our distance to goal metric. Creating our shot angle variable followed a similar process. When picturing shot angle conceptually, we pictured the two goal posts, and the shot location as the three points of a triangle. Since we knew the exact locations of all three of those points, and could use euclidean distance to find the distance between all three of those points (the length of the triangle's sides), simple trigonometric functions within R allowed us to find the shooter's angle to goal.

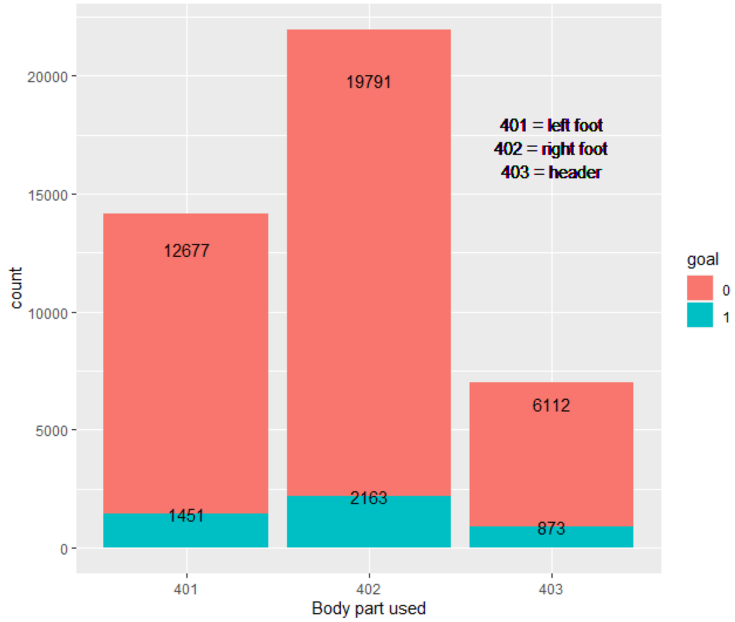
Preparing Data for Oversampling With ROSE

When all was said and done, the conclusion of our cleaning process left us with 43,067 observations and 8 predictor variables, including the body part used to shoot, the x and y percentage coordinates, the half which the shot occurred, the second of the half, whether the shot was on the counter attack, the shot angle, and the distance to goal. Of the 43,067 shots we observed in our data, 4,487 of those shots were goals scored. Presented with this class imbalance, given only 10.42% of our observations were classified as goals, we decided to explore oversampling techniques with the hope of aiding our model building process. Using the R package 'ROSE', we conducted model building trials using oversampling, undersampling, as well as a combination of both, in order to artificially balance our data. ROSE uses a bootstrap method of oversampling, where observations of the positive class are randomly drawn with replacement and added to the newly oversampled dataset as synthetic observations. When oversampling, our training dataset ballooned to 61,664 observations. Importantly, of those 61,664 observations, we now had 30,800 observations of the positive class to help teach our model to better identify a goal being scored from a given observation in our training dataset. While conducting undersampling in ROSE, as well as a combination of both over and undersampling, we observed a similar evolution of the training set to a significantly more balanced dataset.

Exploratory Analysis Garrett Ramos

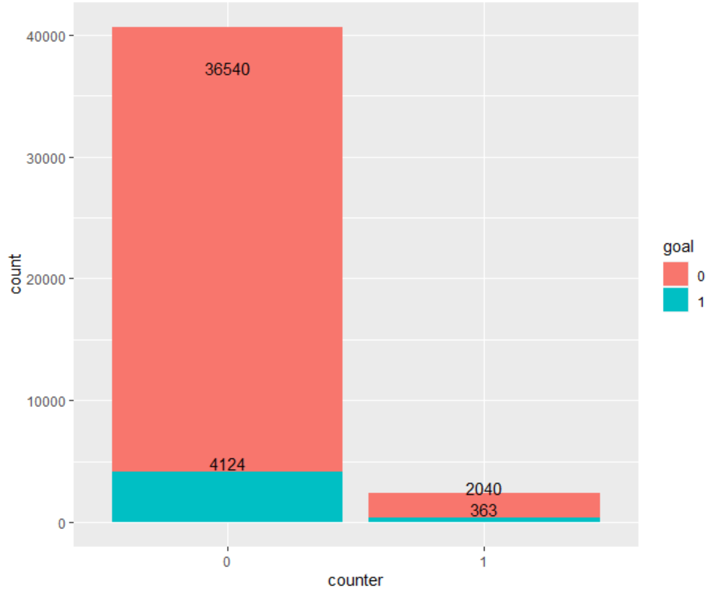
Categorical Variables

Body Part Used



Splitting the data into three categories based on which body part was used to shoot with, we observed shots that were taken with the left foot (as signified by 401 on the x-axis), shots taken by the right foot (as signified by 402 on the x-axis), and shots that were headers (as signified by 403 on the x-axis). We also observed the data based on the outcome of the shot. Shots that resulted in goals are colored blue, while shots that did not result in goals are colored in blue. Approximately, half of the shots in our dataset were taken with the right foot, while about 33% of the observations were considered left foot shots. Headed shots accounted for about 16% of the observations in our dataset. While somewhat surprising, initially, to see such a large majority of shots taken by the right foot, our inferred takeaway from this was that the majority of the players captured in our data are right footed. This logic was confirmed by a paper accessed through the University of Sheffield, which found that 60% of professional soccer players were right footed, while only 22% of players were identified as left footed. The remaining players were considered equally strong using both feet. In terms of shots taken using the foot, either the left or right foot, we found that there was approximately a 10% conversion rate, meaning only about 10% of foot struck shots resulted in goals. In comparison, we found that, while representing a minority of the data, roughly 12.5% of headed shots resulted in goals, which provided interesting insight into the effectiveness of headers, and potentially the value of corner kicks, and set pieces to be analyzed in future research.

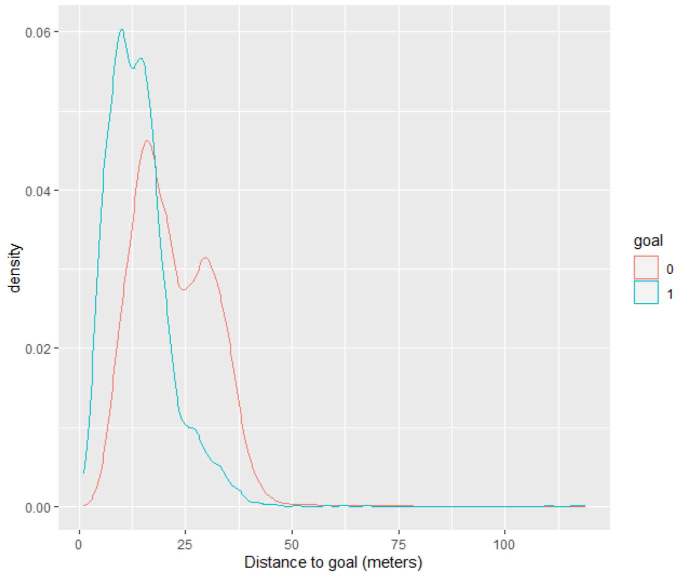
Counter Attack



By splitting the data based on the binary variable ‘counter’, we observed our data based on shots that were taken while on the counter attack, as opposed to shots taken while not on the counter attack. Shots taken while not on the counter attack accounted for a vast majority of the observations, and were identified on the x-axis by 0. Shots taken while on the counter attack represented only 5.6% of the observations, and were identified on the x-axis by 1. While few in observations, shots while on the counter attack led to 363 goals, identified by the blue portion of the column. A relatively small number compared to the 4124 goals scored while not on the counter attack, we found that the conversion rate was actually notably higher for shots taken while on the counter attack. Roughly 15% of shots taken on the counter attack resulted in a goal, in comparison to shots taken while not on the counter attack, where only 10% resulted in a goal. This insight leads us to believe that counter attacks could be an important factor related to the likelihood of scoring a goal in a match. Teams may prioritize a faster, more direct style of play in order to play on the counter, and create, perhaps fewer, but more dangerous scoring opportunities with a greater likelihood of scoring.

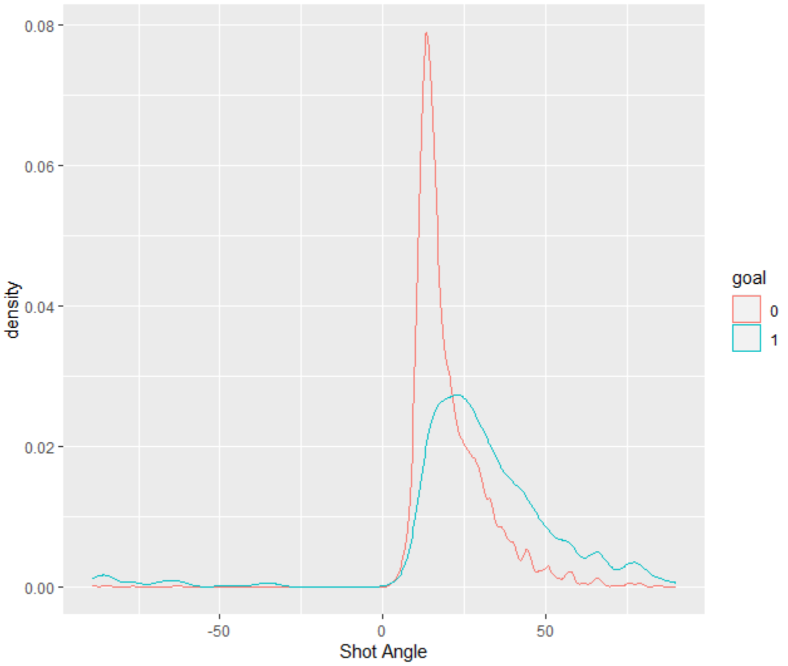
Numerical Variables

Distance to Goal



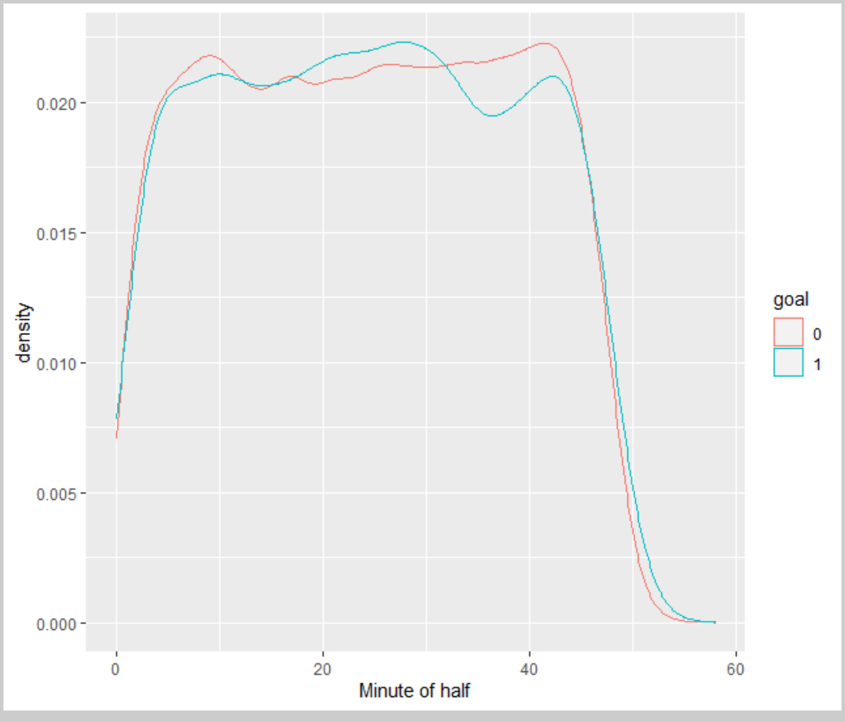
We created a density plot in order to visualize our data in accordance to each observation's distance to goal when the shot was taken. The y-axis helps denote the 'density' or frequency of each point on the line graph occurring within our data. We also split our data into two separate lines. The red line represents all shots that did not result in goals, while the blue line represents all shots that resulted in goals. Our expectation prior to this analysis was that the representing goals scored would peak closer to the goal (distance wise), than missed shots. Logic would infer that the closer to goal one is, the more often they are likely to score. The density plot above follows this logic, as we see the blue line peak at the 10 meter mark along the y-axis, compared to the red line which peaks at about 15 meters. This tells us that most goals in our dataset are scored approximately 10 meters from goal, while most misses occur approximately 16 meters away from goal.

Shot Angle



Similar to our analysis of shots based on our distance to goal variable, we used common logic prior to analysis to infer what we expected to see in terms of the angle of a shot. Once again, the data was split based on whether the shot resulted in a goal or not. We expected goals to be scored from a wider angle as opposed to misses because common logic would lead us to believe that it is much harder to score a goal from a tight angle as opposed to a more neutral angle where the shooter is directly in front of the middle of the goal. Having conducted our visual analysis using the density plot above, we found that the goals from our data were scored from an angle of about 21 degrees, whereas shots that did not result in a goal peaked at a shot angle of about 12 degrees. This followed our initial logic, where we figured that shots from tighter, more acute angles would often lead to misses rather than goals.

Minute of Half



Converting the ‘eventsec’ variable, originally the seconds into a half where the event took place, into minutes, we observed the shots from our data relative to the minute of the half when the shot took place. There is typically 45 minutes in a half of soccer, with no more than 3 to 4 added minutes in the form of extra time, depending on injuries, substitutions or other match interruptions. An author, and Professor of Applied Mathematics at the University of Uppsala in Sweden, David Sumpter has spoken about a common misconception that many soccer fans have when watching a match, where they believe that a team has ‘one more big chance left’. In essence, fans often believe that during the latter moments of a match, teams are more likely to score or have a ‘big chance’, a big shooting opportunity. In the analysis conducted using the density plot above, we looked at both shots and goals scored and found, similar to Sumpter, that there is no correlation between any given minute in a match and a goal being scored, or even a shot taking place. This was very interesting, and insightful to us because it essentially debunked the ‘one big chance’ novelty that many fans have when watching a match. One may think that in the beginning of a match, the opposition is still asleep, and that that is the opportunity to often score, or earn shooting opportunities. Similarly, one may think that at the end of a match, when an opposition is tired, or starting to lose focus, that that is the chance to score, or get shots on goal. In reality, there is the same likelihood of scoring, and shooting in the beginning, middle and end of the match. Any given minute of the match has the same likelihood of providing a goal, or shot.

Artificial Neural Networks Ayussh Ahuja

Introduction & Background

As mentioned earlier, George Pipis used an Artificial Neural Networks (ANN) based model to predict the outcome of soccer matches between two teams, Southampton, and Chelsea. While our project also includes an ANN model, it has a completely different goal. We aim to predict the likelihood of a shot resulting in a goal. We also relied on the H20 algorithm to create a feedforward Artificial Neural Network. A table outlining the predictors used can be seen below.

Predictor	Description	Format
Body part used	3 level categorical variable depicting which body part was used to shoot with during the event. 401 signifies a right foot shot. 402 signifies a left foot shot. 403 signifies a headed shot.	Categorical

y	The nearness to the right side of the pitch in percentage terms.	Numerical
x	The nearness to the goal (attacking team perspective) in percentage terms.	Numerical
matchPeriod	2 level categorical variable indicating which half of the match the event took place in. 0 signifies the first half. 1 signifies the second half of the match.	Categorical
eventSec	indicates the time in seconds when the event took place following the start of the half. So if the row indicates that the event took place in matchPeriod 1 (2nd half of match), and the eventSec says 250. The event took place 250 seconds into the second half, or approximately 4 minutes and 10 seconds into the second half (which would read 49:10 on a scoreboard).	Numerical
counter	2 level categorical variable indicating whether the event took place on the counter attack (fast break). 0 signifies the shot was not the result of a counter, while 1 signifies the shot was a result of a counter attack.	Categorical
shotangle	the angle to goal from the shooter's perspective. The shooter, and each post of the goal represent the three points of the triangle, with the angle in question being at the shooter’s point.	Numerical
distance_to_goal	the distance in meters measured from the shooter to the middle of the goal.	Numerical

While our data included additional variables, matchID, eventID, teamID, and playerID, these were obviously removed from any exploratory analysis since they are simply identifiers that play no role in influencing the likelihood of a shot resulting in a goal.

H20 Cluster: We used trial & error to run the model with multiple variations of our h20 cluster that utilized a combination of nodes and hidden layers. The final line of code we used is listed below:

```
hid.list=list(c(2), c(2,2), c(3,3), c(4,4), c(5,5), c(2,2,2), c(3,3,3), c(4,4,4), c(5,5,5), c(4,3,2),
c(5,4,3,2))
```

Regularization Parameter: We used the LASSO technique to limit overfitting of our model. Our lasso parameters included values of 0, 0.001, 0.01 and 0.05.

H20 grid: Finally, we set up the H20 grid using the Rectifier activation function on the hidden layers, while the softmax function operated on the output layer in order to convert the output into classification probabilities.

Solutions & Limitations

Similar to other models, we also used the ROSE package in ANN to oversample our data in order to better predict the 1's (goals). This reduced the inherent imbalance in our data since only 10.4% of observations were goals. We also experimented in running the model using a hyperbolic tangent as activation function. A summary of the results from different ANN models can be seen in the table below. The table only includes the best models based on oversampling and which activation function to use (after first finding the right number of nodes, lasso and variables to use).

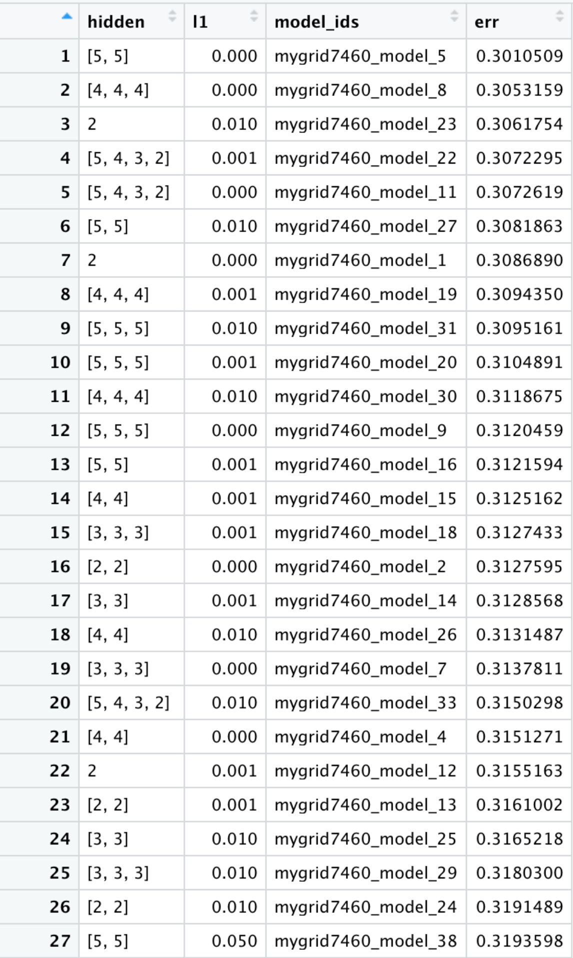
Model Evaluation

Activation Function	Oversampling	Accuracy	Sensitivity	Specificity
Rectifier	Yes	73.59%	67.26%	74.26%

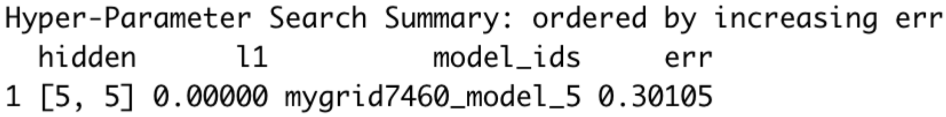
Hyperbolic Tangent	Yes	70.811%	70.46%	70.85%
Hyperbolic Tangent	No	89.75%	3.24%	99.81%

Analysis

As you can see, the best approach was to use the ROSE package to oversample the training set and then utilize a rectifier activation function. The grid summary table from this approach is shown below. The best model is a model with two hidden layers with 5 nodes each and runs with a lasso parameter of 0 (maximum constraints).



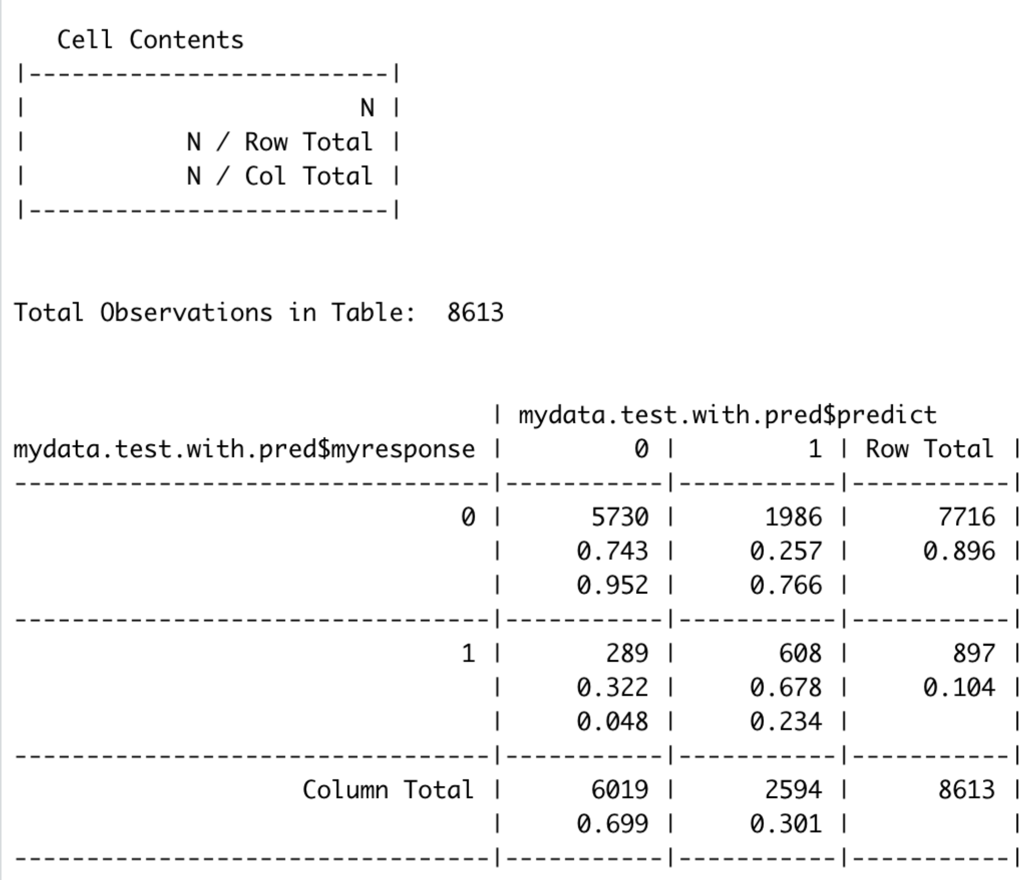
	hidden	l1	model_ids	err
1	[5, 5]	0.000	mygrid7460_model_5	0.3010509
2	[4, 4, 4]	0.000	mygrid7460_model_8	0.3053159
3	2	0.010	mygrid7460_model_23	0.3061754
4	[5, 4, 3, 2]	0.001	mygrid7460_model_22	0.3072295
5	[5, 4, 3, 2]	0.000	mygrid7460_model_11	0.3072619
6	[5, 5]	0.010	mygrid7460_model_27	0.3081863
7	2	0.000	mygrid7460_model_1	0.3086890
8	[4, 4, 4]	0.001	mygrid7460_model_19	0.3094350
9	[5, 5, 5]	0.010	mygrid7460_model_31	0.3095161
10	[5, 5, 5]	0.001	mygrid7460_model_20	0.3104891
11	[4, 4, 4]	0.010	mygrid7460_model_30	0.3118675
12	[5, 5, 5]	0.000	mygrid7460_model_9	0.3120459
13	[5, 5]	0.001	mygrid7460_model_16	0.3121594
14	[4, 4]	0.001	mygrid7460_model_15	0.3125162
15	[3, 3, 3]	0.001	mygrid7460_model_18	0.3127433
16	[2, 2]	0.000	mygrid7460_model_2	0.3127595
17	[3, 3]	0.001	mygrid7460_model_14	0.3128568
18	[4, 4]	0.010	mygrid7460_model_26	0.3131487
19	[3, 3, 3]	0.000	mygrid7460_model_7	0.3137811
20	[5, 4, 3, 2]	0.010	mygrid7460_model_33	0.3150298
21	[4, 4]	0.000	mygrid7460_model_4	0.3151271
22	2	0.001	mygrid7460_model_12	0.3155163
23	[2, 2]	0.001	mygrid7460_model_13	0.3161002
24	[3, 3]	0.010	mygrid7460_model_25	0.3165218
25	[3, 3, 3]	0.010	mygrid7460_model_29	0.3180300
26	[2, 2]	0.010	mygrid7460_model_24	0.3191489
27	[5, 5]	0.050	mygrid7460_model_38	0.3193598



```
Hyper-Parameter Search Summary: ordered by increasing err
  hidden      l1           model_ids     err
1 [5, 5] 0.00000 mygrid7460_model_5 0.30105
```

```
Model Details:
==============

H2OBinomialModel: deeplearning
Model ID:  mygrid7460_model_5
Status of Neuron Layers: predicting myresponse, 2-class classification, bernoull
i distribution, CrossEntropy loss, 107 weights/biases, 5.7 KB, 246,656 training
 samples, mini-batch size 1
  layer units      type dropout       l1       l2 mean_rate rate_rms
1     1    12     Input  0.00 %       NA       NA        NA       NA
2     2     5 Rectifier  0.00 % 0.000000 0.000000  0.173505 0.380851
3     3     5 Rectifier  0.00 % 0.000000 0.000000  0.003179 0.006038
4     4     2   Softmax      NA 0.000000 0.000000  0.003051 0.001647
  momentum mean_weight weight_rms mean_bias bias_rms
1       NA          NA         NA        NA       NA
2 0.000000    0.068498   0.334940  0.574881 0.425304
3 0.000000   -0.270704   0.789125  0.611265 0.825648
4 0.000000    0.140691   2.777061  0.000000 0.773721
```

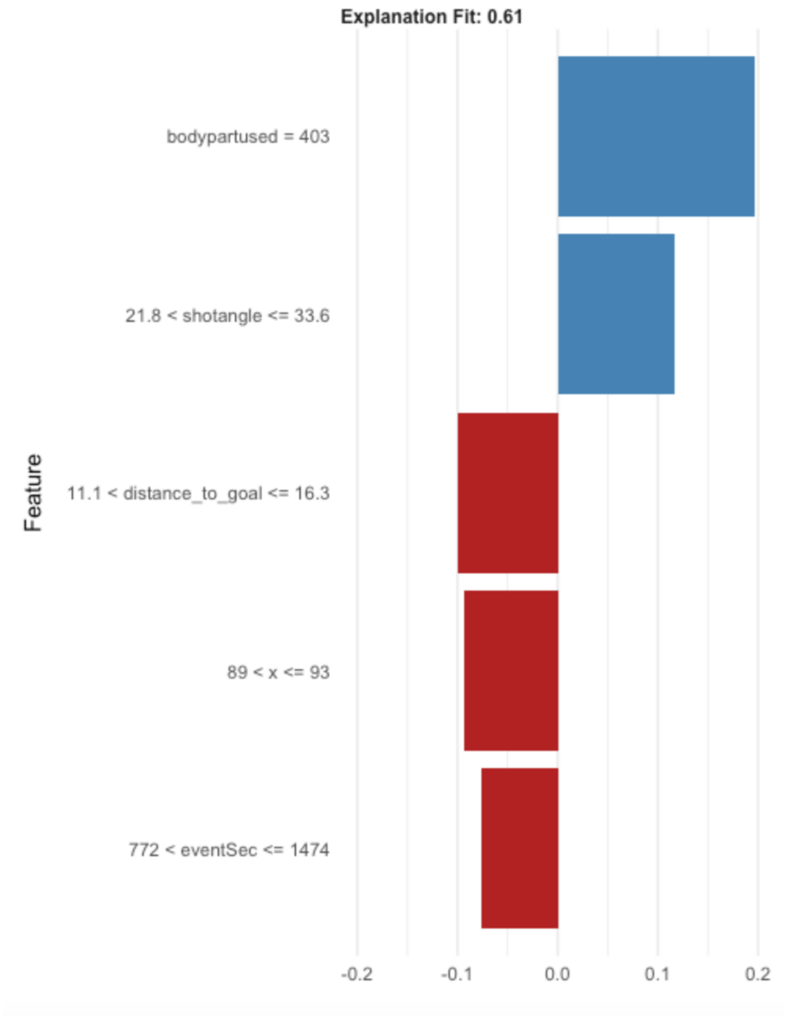


```
   Cell Contents
|-------------------------|
|                       N |
|           N / Row Total |
|           N / Col Total |
|-------------------------|


Total Observations in Table:  8613


                                 | mydata.test.with.pred$predict
mydata.test.with.pred$myresponse |         0 |         1 | Row Total |
---------------------------------|-----------|-----------|-----------|
                               0 |      5730 |      1986 |      7716 |
                                 |     0.743 |     0.257 |     0.896 |
                                 |     0.952 |     0.766 |           |
---------------------------------|-----------|-----------|-----------|
                               1 |       289 |       608 |       897 |
                                 |     0.322 |     0.678 |     0.104 |
                                 |     0.048 |     0.234 |           |
---------------------------------|-----------|-----------|-----------|
                    Column Total |      6019 |      2594 |      8613 |
                                 |     0.699 |     0.301 |           |
---------------------------------|-----------|-----------|-----------|
```

Finally, the confusion matrix for this model can be seen above. Overall, the model had an accuracy of 73.59% (higher than SVM but slightly lower than Random Forests). This means that the model performed relatively well in summary, and predicted 74% of observations right, on average. The specificity of the model was 74.26%. Therefore, the model predicted 74% of 0's correctly. Ultimately, since our mission is to predict the likelihood of a goal, we need to pay attention to sensitivity. This model had a sensitivity of 67.26% (lower than SVM but higher than Random Forests), implying that the model predicted 67% of 1's correctly. Therefore, the model performed better when predicting the 0's rather than the 1's. This is not ideal since we hope to predict the 1's (goals). However, it is understandable considering the fact that the data was originally skewed towards 0's and even after oversampling, there remains a higher uncertainty in predicting goals. Thinking rationally, it is obviously much harder to predict when a shot results in a goal, rather than when it does not. Most shots often miss.



We also created a lime plot to further analyse the fifth row in our dataset. With an explanation fit of 0.61, we can reasonably trust this explanation because it is above the 0.50 cutoff. According to the lime plot, having a body part used of 403, a header, was the most important predictor for this shot. This follows on from our analysis that headed shots have a higher chance of resulting in a goal. Next, a shot angle between 21.8 and 33.5 (32.439 for this shot) positively contributed to the shot being predicted a goal. On the other hand, distance to goal, x and eventSec negatively contributed to this prediction. The eventsec finding is most surprising since our explanatory analysis showed there was almost no correlation here. However, it was also the least most important predictor on the LIME plot. Another surprising factor was that distance to goal ranked so low in the lime plot, while showing up as the most important predictor in random forests. This might signal a problem, since common sense does tell us that the closer you are to a goal, the higher the chances of you scoring. Therefore, further research for ANN models could assign weights to certain important predictors such as these.

Conclusion

Overall, the best model from ANN performs reasonably well in summary. It has a higher accuracy and specificity than the other models while only slightly missing on specificity. Over the course of our project, we reiterated our model multiple times to arrive at a reasonable final model. For example, our first iteration had less than 50% accuracy and a 4% sensitivity. While we made significant adjustments to that initial model, we can further improve our model by experimenting with different parameters. For instance using different activation functions like Maxout, or a rectifier with dropout could yield better results. While we did experiment extensively with the H20 cluster, most models seemed to perform better with 2-3 hidden layers and usually 5-6 nodes per hidden layer. However, since each iteration took a significant amount of time, shifting to the cloud could allow more experimentation in the future. We could also make use of the “AML” function in H20, which would use code to configure the right number of hidden layers and nodes. Ultimately, there will always be a slight error in any predictive model because of the unpredictable nature of a sport played by humans.

Random Forest Xavier Torres

Introduction and Background

As stated before, Random Forest has been used before to predict the outcome of matches using data scraped before. Random Forest is a particularly useful method as it has special properties that allow for the prevention of ‘overfitting’ as well as the gathering of the ‘best’ or the variable that has the most predictive power. Allowing our team to find the most explanatory variable is unique to Random Forest as it is calculated using OOB 'out of bag’ error. A major point of using RandomForest was the natural ability to negate overfitting since it's an ensemble method. Especially with prioritizing the accuracy on goals rather than misses, it can be quite easy for our models to become overfit to the training data. As a result we decided as a team to include Random Forest as one of our models as a countermeasure against overfitting. We ran 3 iterations of Random Forest: Random Forest basic, Random Forest with ROSE oversampling, and finally Random Forest with ROSE oversampling and Bagging. The input variables for RandomForest models are within the ‘introduction and background’ section of our ANN model.

Solutions and Limitations

In accordance with all the other models,we see some significant problems with the distribution of goals and non goals in the data set. As such in order to be able to predict goals with better accuracy than misses, we decided to oversample using the ROSE package. ROSE(‘Random Over-Sampling Examples’) is a package that allows for oversampling of certain aspects of the training set. We oversampled the number of goals in the training set so the model could predict goals well. The value of predicting missed goals is much less compared to the ability to accurately predict goals. Therefore sacrificing overall model performance for better sensitivity seemed like a good trade. The negation of overfitting was one of our main criteria when selecting random forest. We realized that using the ROSE oversampling technique would essentially be allowing our models to simulate more data. This can easily stray the model into overfitting. Random forest models allow us to prevent some of that overfitting as it has built in elements of randomizing variables.

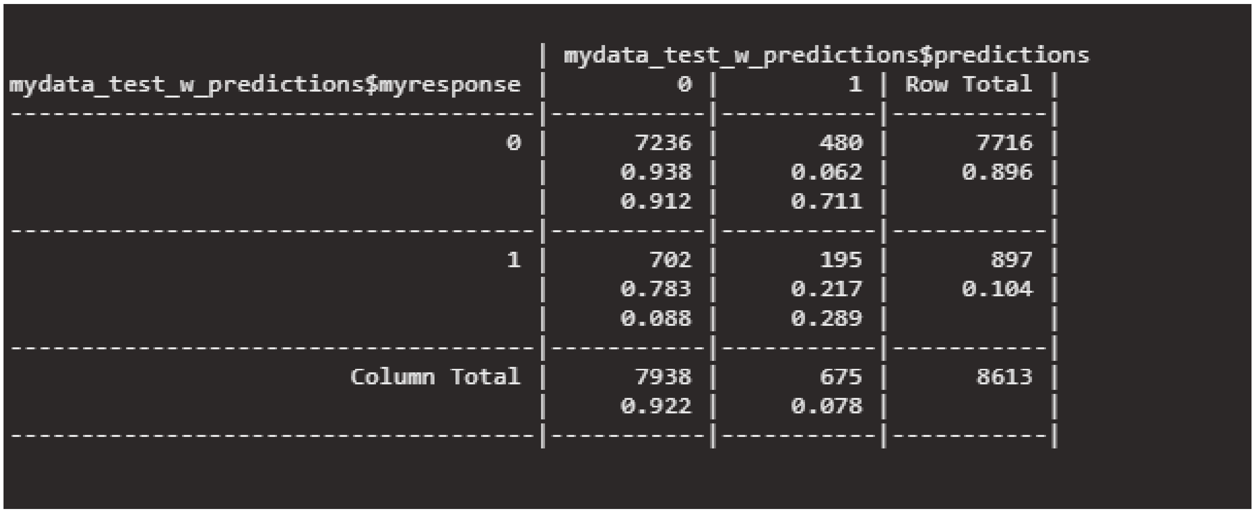
Model Evaluation

We conducted three different random Forest models with different parameters. Here are the results:

Name	Accuracy	Sensitivity	Specificity
RF.rose.bagging	88.51%	14.4%	97.1%
RF.rose	78.35%	51.4%	81.5%
RF	89.66%	10.6%	98.9%

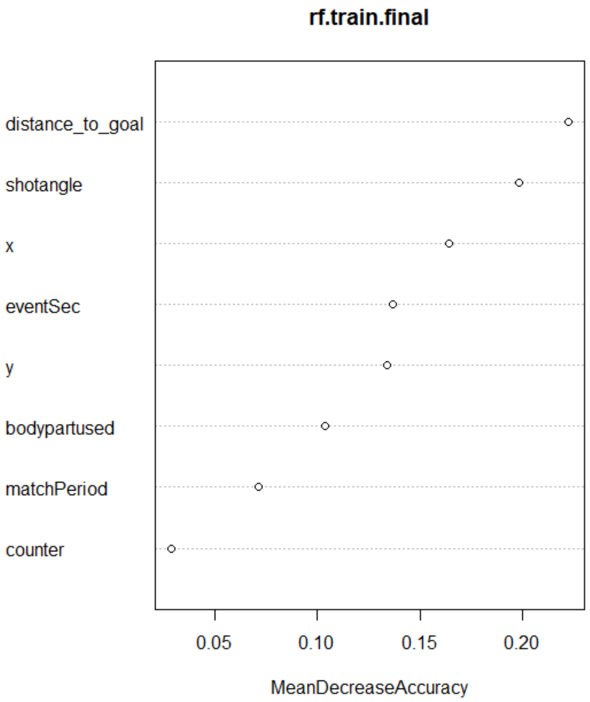
Analysis

The best model is the RF.rose model. This includes ROSE oversampling without bagging. This way is best to maximize the goal variable of sensitivity. The model achieves an accuracy of 78.35% , a sensitivity of 51.4% and a specificity of 81.5%



```
                                     | mydata_test_w_predictions$predictions
mydata_test_w_predictions$myresponse |         0 |         1 | Row Total |
-------------------------------------|-----------|-----------|-----------|
                                   0 |      7236 |       480 |      7716 |
                                     |     0.938 |     0.062 |     0.896 |
                                     |     0.912 |     0.711 |           |
-------------------------------------|-----------|-----------|-----------|
                                   1 |       702 |       195 |       897 |
                                     |     0.783 |     0.217 |     0.104 |
                                     |     0.088 |     0.289 |           |
-------------------------------------|-----------|-----------|-----------|
                        Column Total |      7938 |       675 |      8613 |
                                     |     0.922 |     0.078 |           |
-------------------------------------|-----------|-----------|-----------|
```

Another thing that the random forest gets to show us is the most important variables that explain the most in the model.



Here we see that distance_to_goal is the most important variable at about twenty percent reduction of accuracy from the model. We also see the importance of other variables as well. ‘Shotangle’ also has an accuracy decrease of 19%. These insights are unique to random forest as it collects these from OOB error which is unique compared to other models.

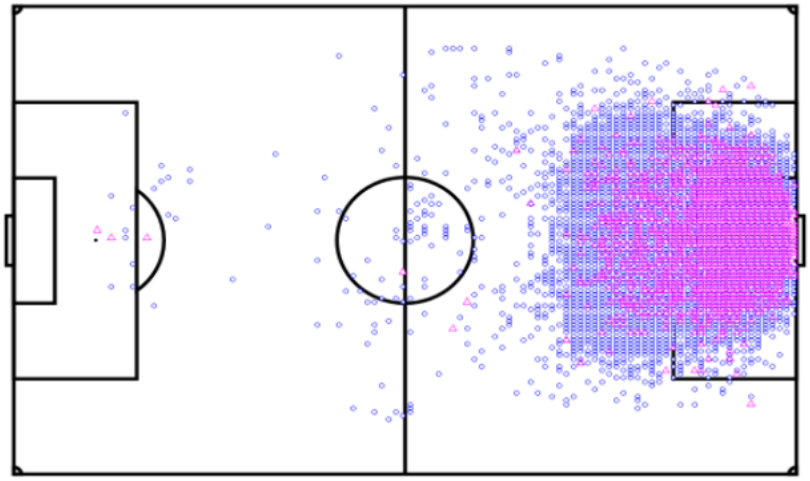
Conclusion

In conclusion, while random forest is not the best predictor in terms of accuracy,sensitivity or specificity there is a definite utility to it. It allows us to cross refrence the importance of the variables that we outlined in the data preprocessing and exploratory phase. The model generally agrees that shot angle and distance to goal were the most important and had the most explanatory power. The point of random forest was mainly to try to deter overfitting within our model. Our best model for random forest gave us an accuracy of 78%, a sensitivity of 51% and a specificity of 82%. While some of these numbers are not the best compared to other models, the overall accuracy of the model seems to be the best of the bunch.

Support Vector Machines Brandon Sung

Introduction & Background

The implementation of Support Vector Machines (SVM) has been seen in previous cases of Machine Learning applications in soccer predictions. As mentioned earlier, the research "Machine Learning Algorithms for Football Predictions" by Matheus Kempa used Support Vector Machines as a method achieving a 47.15% accuracy of soccer matches (Kempa). While relatively high compared to his other Machine Learning Model, it never reached the 50% accuracy goal set by Kempa. Kempa argues that "SVM has a lot of kernels, trying different kernels might be a solution to reach a better performance." (Kempa) While kernels will profoundly impact the accuracy of the model, it's important to note that the relationship between columns is very important to SVM, especially with the scaling needed to complete the model. Additionally, Kempa used web scraped data that "did not have patterns" and were "complex" (Kempa). As a result, SVM struggled with his data. This project uses cleaner data, as it has become more available than during Kempa's research. The data also contains locational features such as the x and y axis that can clearly be mapped in a 2D plot.



SVM Table. X and Y coordinates plotted. Blue-0, Pink-1.

As seen on the table, a visible relationship between the goals and misses of the new dataset looks promising. Positional features include:

- X coordinates
- Y coordinates
- Shot angle
- Distance to goal

Other features:

- Body part used
- Match period
- eventSec
- Counter

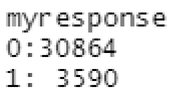
The modeling done in this paper used three kernels and multiple parameters in order to try and capture the trends of the data: Soft Margin, Polynomial Kernel and Radial Basis Kernel.

Solutions & Limitations

The SVM in this project provides a unique challenge in terms of time and hardware limitations. Given the nature of this project, all of the code was run locally on the Babson laptop. There was limited to no access to cloud processing or more powerful computational heavy equipment. While this did not seem to be a problem with other models, SVM proved to require much more processing power and time. Since previous attempts at soccer data, such as Kempa's research, used fewer parameters and kernels with mild results, it seems that improvements on previous works would require running more models with different parameters. Initially, the following parameters were decided on:

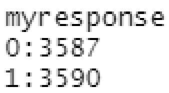
Soft Margin	Polynomial Kernel		Radial Basis Kernel	
Cost	Cost	Degree	Cost	Gamma
0.001	0.001	2	0.001	0.5
0.01	0.01	3	0.01	1
0.1	0.1	4	0.1	2
1	1	5	1	
5	5		5	
10	10		10	
100	100		100	

An initial attempt to run these models resulted in excessive processing times and resulted in extreme biases to misses. The training data contained a prevalence of 10.4% with the data split of:



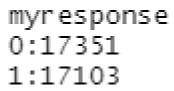
```
myresponse
0:30864
1: 3590
```

The ROSE package was used to undersample the data with the new split to be:



```
myresponse
0:3587
1:3590
```

The 34454 training set was reduced to 7177 observations. Using the ROSE package, the data was both under and oversampled creating another dataset:



```
myresponse
0:17351
1:17103
```

The undersampling dataset used the same parameters as originally planned. The "both" sampled data used the following parameters to lower the processing time:

Soft Margin	Polynomial Kernel		Radial Basis Kernel	
Cost	Cost	Degree	Cost	Gamma
0.01	1	4	Same as original	
5	10	6		
100	100			

Model Evaluation

Below are the lowest 10-fold cross validation errors:

	Soft Margin	Polynomial Kernel	Radial Basis Kernel
Undersampled	30.0% (cost=1)	29.2% (cost=10, degree=4)	29.3% (cost=.1, gamma=0.5)
"Both" sampled	29.5% (cost=5)	28.3% (cost=100, degree=4)	16.8% (cost=100, gamma=2)

Below are the testing set confusion matrix results:

		Soft Margin	Polynomial Kernel	Radial Basis Kernel
Undersampling	Accuracy	0.647393475	0.6886102403	0.6985951469
	Sensitivity	0.7703455964	0.7257525084	0.7112597547
	Specificity	0.6331000518	0.6842923795	0.6971228616
"Both" sampling	Accuracy	0.6482061999	0.7107860211	0.7042842215
	Sensitivity	0.7692307692	0.6978818283	0.5016722408

	Specificity	0.6341368585	0.7122861586	0.7278382582

Analysis

The best model for accuracy is the “both” sampled polynomial kernel with a 71.1% accuracy. The best model for predicting goals is the undersampled soft margin with a 77.0% sensitivity. The best model for predicting goals is the “both” sampled radial basis with a 72.8% specificity.

At a sensitivity of 77.0% and 76.9% for under and “both” sampling, respectively, soft margins seemed to provide the best predictors for goals. Radial basis kernels sacrifice sensitivity to be the best predictor for misses. Under and “both” sampling yielded the best results for specificity. It seems that the most well rounded model is polynomial kernels as their cross validation errors were relatively small and the sensitivity and specificity were around similar values.

As detailed above the Radial Basis Kernel outperformed in both sampling methods of the cross validation errors. The surprisingly low cross validation error of “both” sampling Radial Basis Kernel showed promising results for its prediction potential. At a shocking 16.8% cross validation error, this model seemed to outperform the next best SVM by over 40%. The resulting confusion matrix yielded an accuracy of 70.4%. As compared to the other models, the results were only .6% off the best SVM (“Both” sampled polynomial kernel). However, the high performing cross validation suggests this model was overfitted to the training set. A lower cost might be able to reduce the overfitting given enough time to run the other models.

SVM Conclusion

Overall, the champion SVM model is the “both” sampled polynomial kernel (cost=100, degree=4) with a best overall predictive accuracy of 71.1%. While the purpose of the project was to predict goals, the best sensitivity model (undersampled soft margin with cost=1) had an overall accuracy of 64.7%. The model over emphasizes the goals and sacrifices specificity (63.3%) at a much greater rate than its gain in sensitivity. Similarly, the best specificity model ("both" sampled radial basis cost=100 gamma=2) had a sensitivity of 50.2%. Maintaining the original objectives of this project, this model would not be appropriate.

Given the overall SVM accuracy of 71.1%, a number of improvements can be made going forward. Similar to Matheus Kempa's original self-reflection, certain parameters could be fine tuned to create a better performing model. Trying other kernels or costs can help increase accuracy. Given the hardware and time limitations, only so much can be done to correct for the lack of experimentation. Additionally, overfitting became a big problem for certain models. A possible solution would be to reduce cost parameters and prevent the model from forming too specifically to the testing data. Another solution would be to use data that has a higher prevalence. The "both" sampling creates goal observations that puts models into the danger of overfitting. Finally, more data is shown to slightly impact the accuracy. Once future data becomes readily available, there could be a potential to build on the SVM models presented here.

Overall Conclusion Entire Team

To summarize our key findings from this project:

- Around 15% of shots were scored when they were taken on the counter attack, as opposed to the approx 10% of shots that were scored when not on the counter attack. Additionally, the majority of goals were scored from 10 meters away at an approx 20 degree angle. As mentioned, this would assist stakeholders in knowing the performance of specific players when making decisions about which players to opt for.

- SVM was a powerful model for predicting goals. The top SVM models had a sensitivity of around 69.8% all the way up to 77.0%. The high sensitivity does come at a cost of specificity. However with total accuracy hovering around 70%, the predictive accuracy of SVM is acceptable to the objectives of this project.
- ANN was the most balanced model when looking at predicting performance. The best ANN model had 73.59% accuracy, 67.26% sensitivity and 74.26% specificity. When changing the activation function, we also found another model with ~70% accuracy, sensitivity and specificity across the board. We believe this provides a solid foundation that can be improved on with more data collection and configuration as discussed in a future section.
- Random Forest provided the highest total accuracy with models ranging between 80-90% accuracy. However, this came at the cost of a much lower sensitivity, with the best model only predicting 51% of goals. On the other hand, specificity was also between 80-90%. Therefore while not directly aligning with our aim to predict goals, Random Forests and Bagging provided interesting findings about the most important predictors.

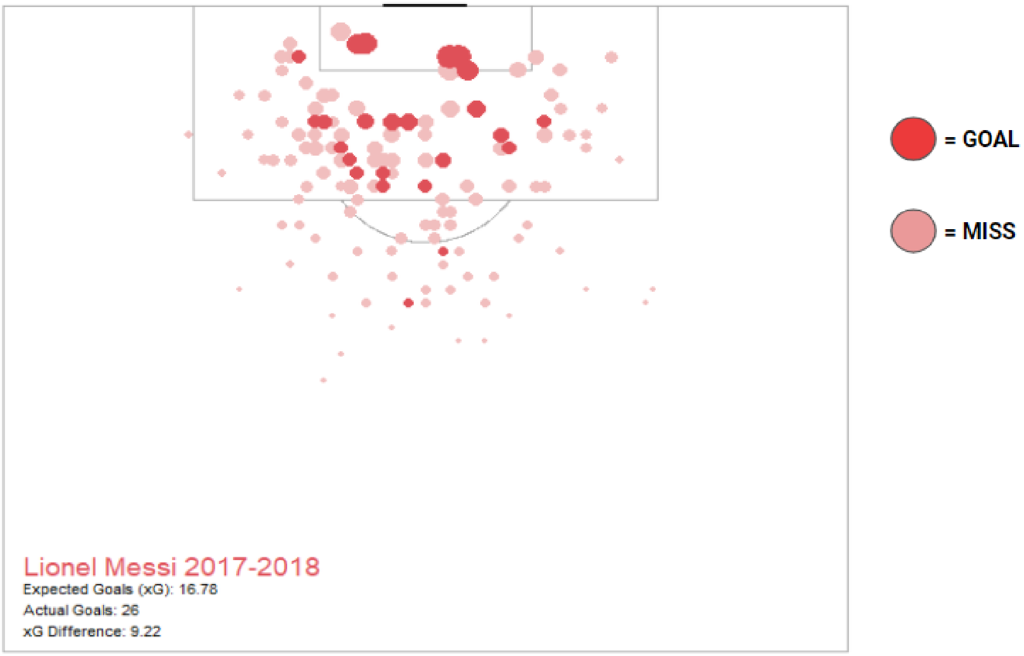
Suggestions for future research

Expected Goals Ayussh Ahuja

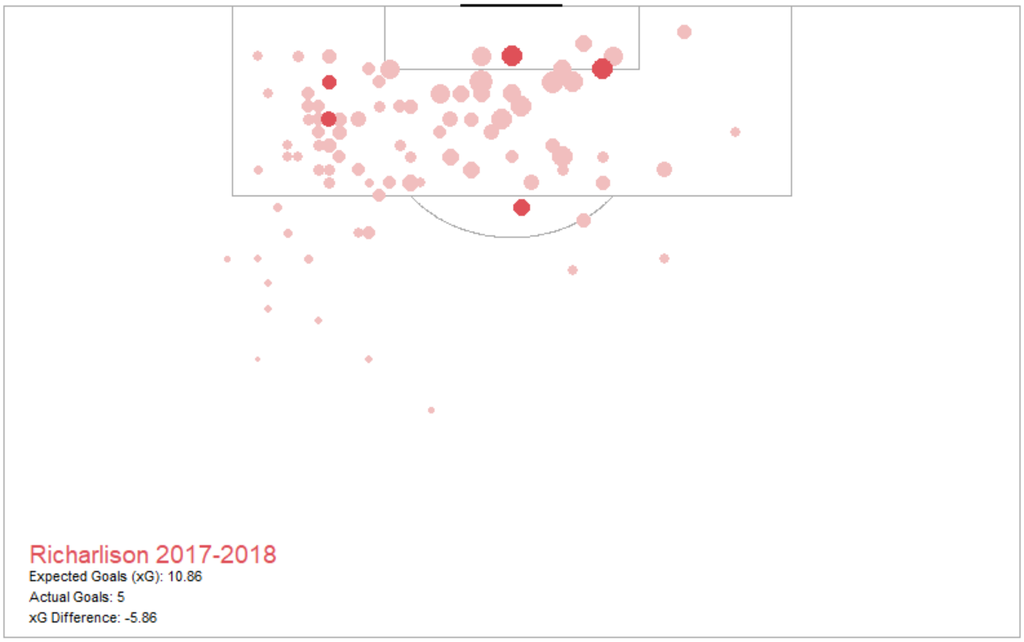
As we look to monetize our findings, we could apply them to real-world applications. One such approach centers around using a statistic, Expected goals (xG), to predict the probability of a shot resulting in a goal. Goal.com defines xG as “a statistical measurement of the quality of goalscoring chances and the likelihood of them being scored.” In layman’s terms, xG ranks how easy a chance is to score by considering factors like distance from goal, body part used, shot angle, and more. The statistic has gained wide popularity with sports pundits, fans and even coaches relying on it as a talking point when analyzing games. When betting on a team to win or choosing a player in fantasy football, supporters often look for a higher xG rating.

XG Visualization

Garrett Ramos



With Expected Goals (xG) in mind, we were able to utilize the models we built to help visualize this metric in the form of the shot map above. This real world application uses the probabilities predicted by our models to provide the likely number of goals scored based on the quality of the shots taken. Using Lionel Messi as an example, we plotted all 142 shots from his 2017-2018 season from our dataset, in accordance to the x and y coordinates on the pitch. Each shot from our dataset has a predicted likelihood of resulting in a goal. This probability is equal to the expected goals (xG) measure of the shot in question. The bigger the circle in the visualization above, the larger the probability, and the greater the likelihood of that shot resulting in a goal according to our model's predictions. When all of the shot's probabilities are summed, we get an expected goals measure for Lionel Messi's entire 2017-2018 season. What we found is that based on the quality of the shots taken, our model expected Lionel Messi to score 16.78 goals from the 142 shots he took during the course of the season. In reality, Messi scored a staggering 26 goals from open play according to our data, outperforming our model's expectation by 9.22 goals. What is important to note is that our model's predictions are based on the average player portrayed by the data. So, based on the quality of shots taken by Lionel Messi, our model expected the average player to score only about 16 goals. What we know is that Messi is not 'the average player', he is one of the greatest soccer players to ever grace the sport. The difference between our model's expectation, and Messi's actual performance simply goes to show how truly talented he is, especially in terms of his goal scoring ability. What we find during the course of further analysis is that the most talented goal scorers often score more goals than our model's expectations. Players like Messi, Cristiano Ronaldo, and Mo Salah will consistently outperform any model's expectations because that is what makes them so uniquely great, in comparison to the average players of the sport.



The reverse can also be said about players who are considered ‘below average’. Take the example from above, Brazilian forward Richarlison. In this particular season, Richarlison had a number of really good scoring chances that he failed to convert to goals. Our model predicted that Richarlison should have scored 10.86 goals, and yet, he only scored 5. These kinds of insights provide football clubs, and other stakeholders with vital information related to the true quality of a particular player or team. Scoring 5 goals is certainly a decent tally for a player competing in the most difficult professional league in the world. But if you were told that the player in question should have scored 10 or 11 goals based on the chances he or she had, it would change your perspective on how impressive scoring 5 goals really is. This ties in to our hope to sort the data by players, thus reducing the “abnormality” from some exceptional players, and allowing for a more authentic predictive model. The next section describes this in more detail.

Data Configuration Ayussh Ahuja

A simple approach to improve our models' predictive accuracy would be to increase the amount of data we feed it. Since our current dataset covers only one particular season, we could easily expand the dataset to cover multiple seasons with more statistics from professional leagues. This would limit the problems of overfitting, and would also reduce outliers from a particular calendar year. Additionally, we could group our dataset by teams. For this, we could use a clustering analysis to group different teams together. This would allow us to more intricately analyse the probability of a shot being converted to a goal depending on playing style. For example, certain teams tend to play more attacking football, while others play on the "counter." Additionally, teams with the best players might have a higher conversion rate. This was evidenced by Lionel Messi's xG graph versus Richarlison's. Better players may find certain chances easier to score, therefore, we could also sort by players. Finally, as some of our classmates mentioned, we could add variables like goalkeeper rating, player form and team formation to enhance our analysis. Ultimately, our models have given us a strong foundation to build open. With further improvements, we think this could find a lot of use in the football world.

Bibliography

Kempa, Matheus. “Machine Learning Algorithms for Football Predictions.” *Medium*, Towards Data Science, 24 Sept. 2020, https://towardsdatascience.com/machine-learning-algorithms-for-football-prediction-using-statistics-from-brazilian-championship-51b7d4ea0bc8.

Pipis, George. “How to Build a Predictive Soccer Model.” *Predictive Hacks*, 14 Nov. 2020, https://predictivehacks.com/how-to-build-a-predictive-soccer-model/.

“Premier League Selects Oracle Cloud Infrastructure to Power New Advanced Football Analytics.” *Oracle*, www.oracle.com/news/announcement/premier-league-selects-oracle-cloud-infrastructure-2021-05-06/.

Bryson, Alex. Frick, Bernd. Simmons, Rob. “*The Returns to Scarce Talent: Footedness and Player Remuneration in European Soccer*.” University of Sheffield. https://www.sheffield.ac.uk/polopoly_fs/1.105580!/file/left-foot-papermay2009.pdf

Sumpter, David. “*Friends of Tracking: Randomness and Prediction of Matches*.” Youtube. https://www.youtube.com/watch?v=3pnkARyrtMo

Sumpter, David. *Soccermatics: Mathematical Adventures in the Beautiful Game*. Bloomsbury Sigma, 2017.