Soccer Goal Prediction
CSE 158 Fall 2017
Assignment 2

Kenan Mesic
PID: A12064508

Emil Kirov
PID: A10906136

Christian Gunther
PID: A11907943

Sahil Bansal
PID: A12008027

I. Abstract

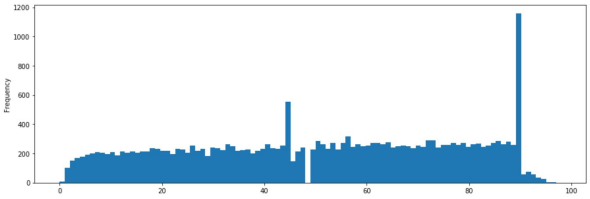
This project considers the relationship between the features during which a soccer player takes a shot and the outcome of the shot, whether it was a goal or not. Essentially, we are trying to predict if a given shot will be a goal. We used the data from ~9000 soccer games across Europe, which consisted of about ~900,000 events. We chose to use shot percentage, team_total_goals, opponent_conceded_goals, time, shot_place, location, bodypart, assist_method, and situation, as our quantifiable predictors. We explored using different models, including support vector machines, logistic regression, Naive Bayes, and random forest classifier, and it turned out that the random forest classifier modeled our data the best. To get an idea of the accuracy of our model, when testing on our test set, we arrived at a training accuracy of 0.9590, a validation accuracy of 0.9342, and a testing accuracy of 0.9346.

II. Dataset Analysis

The data set that we ended up using to study is a combination of all the in-game events that occurred in 9,074 soccer games that took place across Europe's top 5 soccer leagues, England, Spain, Germany, Italy, France. Each sample in that data set contained the following attributes: 'id_odsp', 'id_event', 'sort_order', 'time', 'text', 'event_type', 'event_type2', 'side', 'event_team', 'opponent', 'player', 'player2', 'player_in', 'player_out', 'shot_place', 'shot_outcome', 'is_goal', 'location', 'bodypart', 'assist_method', 'situation', and 'fast_break.' Furthermore, in total there were 941,007 rows of data, which was plenty. When initially looking at this dataset, we already had a general idea as to what predictors would make sense in predicting whether a shot was a goal or not. For example, 'id_odsp', 'id_event', 'sort_order', 'text', 'event_type', 'event_type2', 'player_in', and 'player_out' were all attributes that didn't have a strong correlation with the act of shooting, meaning that they didn't provide any useful information as to if a shot could be a goal or not. Certain attributes such as 'side', 'event_team', 'opponent', 'player', 'player2', and 'fast_break' were attributes that we thought could have a significant effect on determining whether a shot was a goal, but we could only determine so once we tried them out in our models. However, the remaining attributes we found were quite useful as predictors. We thought that for 'time,' players tended to score around a certain time within the

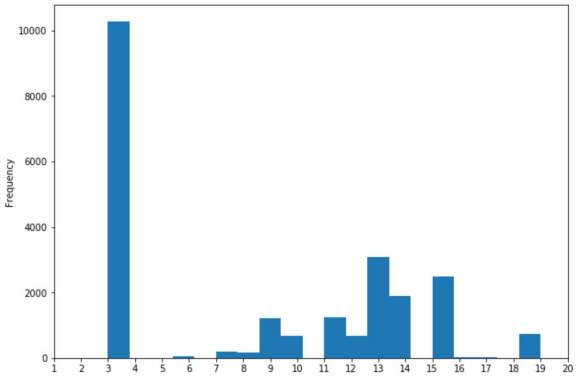
game. By looking at the graph where we plotted goals vs. time, we can notice that there are a significantly higher number of goals during the period of time right before the end of a half.

All Goals - Time vs Frequency



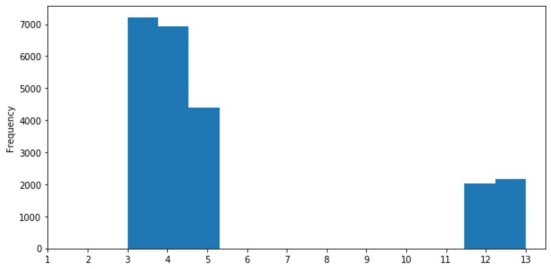
'Location' was useful for us since we noticed that if a player took a shot in a certain location, they had a much higher chance of converting the shot into a goal. As we can see in the graph below, the highest frequency of goals, occurs at location 3, which is considered the 'Centre of the box.'

All Goals - Location vs Frequency



We also took 'shot_place' into significant consideration because certain types of shots have better chances of becoming goals than other types of shots. Once again, we plotted this data to get a better understanding of this intuition and as we can see in the graph below, the shot placements of 3,4,5,12,13, which correspond to bottom left corner, bottom right corner, center of goal, top left corner, and top right corner respectively, tend to have a better goal scoring rate.

All Goals - Shot placement vs Frequency



'bodypart', 'assist_method', and 'situation' were the last predictors we knew we wanted to add to our model because we realized that a certain player's body part may be a more effective way of scoring than other body parts. Similarly, certain assist methods, such as pass, cross, through ball, and certain situations, such as open play or set piece, can also have one method that's significantly better than the other.

III. Predictive Task

In the world of soccer, the most amount of goals will win the game. However, each goal is extremely valuable and many shots are tried on goal. In the top 5 leagues, there are about 3 goals a game, yet there are around 15 shots a game. This means that only a couple of shots a game will end up going into goal.

Evaluation

We decided to predict whether a shot would become a goal. To evaluate this model, we would just use basic accuracy measures. This would be: $\frac{number\ of\ correct\ predictions}{total\ predictions}$.

Baselines

The first baseline would be a trivial predictor that would treat every shot as having a 50%-50% of going into the goal. The baseline predictor would then randomly pick if a shot would become a goal. We would expect this predictor to have an accuracy rate of 50%.

However, looking closer at our data, we will find that only about 10% of the shots are goals. Thus, another trivial predictor would be to predict that every shot would not become a goal and this would yield around a 90% accuracy rate.Thus building a predictor that is above 90% accuracy would be a fairly good classifier for shots that turn into goals.

Preprocessing

To preprocess this dataset, we had a number of things to clean. First we got rid of all events that were not shots. Then we had to get rid of own goals because we didn't want this to affect the predictor as own goals were so small that they were extreme outliers. Finally, we got rid of all events that had null values in the features that we decided to use.

Features

Choosing the features to use was very important in creating a good model. We decided to use 9 features, some of which came directly from the dataset, others that we had to build from the dataset. Our first feature was the shot to goal ratio of the player shooting the ball. Shot to goal ratio of player X: $\frac{number\ of\ shots\ by\ Player\ X}{number\ of\ goals\ by\ Player\ X}$. We had to create a dictionary of all the players in the dataset and then store the total number of shots by each player and their total number of goals to calculate this feature.

The second feature we chose was the number of goals the team that shot the ball made in matches. We calculated this by storing a dictionary of all the teams and counting the number of goals they scored. Our third feature was the number of goals that the opposing team conceded. We used a similar calculation as the 2nd feature by storing all the teams in a dictionary and counting how many goals they conceded.

The next six features we received from the event data that was in the dataset. We got the time of the shot in the game. A game is 90 minutes long. However, with extra time, a game can range from 0-100 min, which is the value we got from shot event. The next feature was the shot placement in perspective of the goal. This feature ranged from 1 to 13 and the values represented where the shot was in goal like bottom left corner, bottom right corner, center of goal, and etc. Then, we grabbed the location of the shot, which is where on the field the shot was taken from. This feature ranged from 1 to 19, where it could represent values like left side of the box, right side of the box, more than 35 yards, and etc. The next feature we included was with what body part the shot was taken with. The value ranged from 1 to 3 and represented body parts like right leg, left leg, and head. For another feature, we added the assist method the shot. This value ranged from 0 to 4 and represented how the ball got to the person taking the shot like pass, cross, headed pass, and etc. The final feature we included in our feature vector was the situation of the play of the shot. This value ranged from 1 to 4 and represented values like open play, set piece, corner kick, and free kick.

IV. Model

Splitting Data

After we decided on our feature vectors, we needed to pick a model that would be better than the baselines to predict if a shot was a goal. We investigated several different models, which included support vector machines, naive bayes, random forests, and logistic regression. To choose the best model, we randomly

sampled our data and split it split into three equal sets, each of around 75800 shots.

Best Model

After training each of these models on the training set and then running them against the validation sets we saw that the random forest performed the best. To optimize the random forest tree we needed to play around with the maximum depth of the tree. Originally we set it to a maximum depth of 4, but as we increased the maximum depth by powers of 2, the training accuracy kept increasing. However, once we got to 16, it reach a peaking point where when we would increase it past 16, it would overfit the data. The training accuracy would increase, but the validation accuracy decreased. Thus we found that the optimal maximum depth of the tree was 16.

Comparison To Other Models

We first tried to train our dataset with LinearSVC with a C value of 1. This model performed support vector classification, but with a linear kernel. While the performance in terms of speed was pretty good with this model, it seemed that this model was underfitting a little too much, where it would almost never predict a goal. We thought that maybe because the dataset had a small amount of goals in it that it didn't help the svc distinguish between it. So instead we tried, splitting the data where 50% of the training was goals. While this did help get rid of our almost trivial predictor, it did significantly bring down our training and validation accuracy to around 70%. We thus decided to move to another model to see if we could get better results.

We tried running logistic regression, but almost got identical results as the LinearSVC. We even applied the 50% goals to 50% shots in the training set, but the results, while better than LinearSVC, weren't much of an improvement.

We then decided to try Naive Bayes. While Naive Bayes accuracy was lower than LinearSVC and logistic regression, it was actually predicting goals for some of the shots. This meant that it wasn't just a trivial predictor. However the accuracy at the end of day was still lower than baseline of 90%.

So we thought that maybe our data is not linearly separable with a hyperplane. We decided to run SVC with rbf (Radial basis function) kernel and a C value of 1. We got much better results and got an accuracy above our 90% baseline measure. However, the biggest issue with this was the performance in terms of speed was extremely bad. It would take much longer than any of the other models. When we saw how this model performed, we decided to look for other models. We then decided to try random forests and as written above, got better results. Not only this, but the performance of the model was much faster than SVC. We thus decided to pick random forest as model to predict goals from shots.

V. Literature

This dataset came from Kaggle and incorporates data from 9,074 soccer games played in the 5 largest European leagues (England, Spain, Germany, Italy, France). The creator of the dataset took text commentary for each game and used regex to extract categories from the text, identifying 941,009 distinct events. The creator has utilized the data to create predictive models for soccer games to determine the winner, make visualizations about upcoming games, build expected goals models, and compare player statistics.

Soccer is a major subject around the globe, so predictions regarding it have been created and

studied a lot. There are many websites such as football-data.co.uk and enetscores.com that compile soccer data. FIFA, the Federation Internationale de Football Association, compiles many datasets about match outcomes as well. Most predictive tasks are used to determine the outcome of a match between two teams. One study conducted at Stanford predicted the outcome of soccer matches utilizing several state of the art algorithms[1]. Their predictor utilized feature data to create five different classifiers: Linear from stochastic gradient descent, Naive Bayes, Hidden Markov Model, Support Vector Machine (SVM), and Random Forest.

Another study tries to predict whether a certain game state will lead to an attempt on goal, and then predicts whether that attempt will be a goal, which is what we are trying to predict [2]. It uses a probabilistic classification to use spatio-temporal data to evaluate what play states will lead to a goal by determining which play states lead to an attempt and which attempt leads to a goal. The part of this research related to our question achieved an 73-79% accuracy of determining whether an attempt led to a goal.

VI. Results & Conclusion

Model	Training Accuracy	Validation Accuracy	Test Accuracy
Random Forest Classifier	0.9590232061108985	0.9342339608701962	0.934578699489452 6
SVC (C=1)	0.9434689111993562	0.9252100951199884	0.899816625110486 7
Naive Bayes	0.8584018258816 08	0.8615285162073378	0.862904183322119 7
Logistic Regression	0.8993522341983403	0.9010541036161427	0.899829817548581 2
Linear SVC (C=1)	0.8993786197707094	0.9010672964023272	0.899843009986675 7

As we can see from the table above, our model performs the best when compared to the alternatives. While the SVC has a validation accuracy of 0.9252, the LinearSVC an accuracy of 0.9011, the Naive Bayes an accuracy of 0.8615, and the Logistic Regression an accuracy of 0.9011, our model has a validation accuracy of 0.9342. This is really interesting to see that we can accurately predict whether a soccer player's (in the top 5 European Leagues) shot will go into the goal or not at an accuracy of up to 93.4%. The feature representation that provided us the most success was a vector of the following nature: ['shot_perc', 'team_total_goals', 'opponent_conceded_goals', 'time', 'shot_place', 'location', 'bodypart', 'assist_method', 'situation'].

We tried other feature vectors but usually they provided us with lower validation accuracies. For example, at first we used a feature vector that included nearly all the attributes in our dataset. However, training on this feature vector led to a trivial predictor, most likely because a lot of the valuable data was confused with a lot of the unnecessary data. After removing the following columns, 'id_odsp', 'id_event', 'sort_order', 'text', 'event_type', 'event_type2', 'player_in', and 'player_out,' on the basis that they held unimportant data, we started

approaching better validation accuracies. Other columns including ‘side’, ‘event_team’, ‘opponent’, ‘player’, ‘player2’, and ‘fast_break’ were eventually removed after playing around with different feature vectors and understanding that these attributes were actually also pushing the accuracies down.

Other attributes that we definitely did want to capture were ‘shot_perc,’ ‘team_total_goals’, and ‘opponent_conceded_goals.’ As we mentioned before, these weren’t included in the initial dataset. We knew that with ‘shot_perc,’ it would be valuable to include a player’s past goal scoring ratio because a higher ratio generally meant that the player had a higher chance of scoring a goal. Similarly, ‘team_total_goals’ and ‘opponent_conceded_goals’ were also valuable attributes because they gave us insight as to whether a team has a better chance of scoring a goal or not. Therefore, after calculating these values, we were able to arrive at our most successful feature vector listed above.

The main parameters for our random forest classifier can be seen using the feature_importances_ function that is provided by the sklearn random forest classifier package. This function tells us how important each feature is in relation to our classifier. The top three features, ‘shot_placement,’ ‘location,’ and ‘shot_perc’ had the highest values as we can see in the feature_importances vector below.

array([0.09668451, 0.05744299, 0.06018606,
0.06163703, 0.39863423, 0.19585056,
0.02379828, 0.02789066, 0.07787568])

[‘shot_perc’, ‘team_total_goals’, ‘opponent conceded goals’, ‘time’, ‘shot_place’, ‘location’, ‘bodypart’, ‘assist_method’, ‘situation’].

Intuitively, this makes sense as the placement of the shot, the location from where the shot was taken from, and the player’s past goal scoring ratio definitely do have a direct effect in determining whether the ball goes in the net.

Overall, our random forest classifier succeeded because it generalized well with the data and surpassed our baseline functions. The linear SVC wasn’t successful because it seemed as though our data wasn’t linearly separable and as for the other SVC, Naive Bayes, and Logistic Regression models, these didn’t conform well with the data as evidenced by our resulting validation and test accuracies. Just looking at the accuracy table above, the failed models could barely beat our baselines.

VII. References

[1] Dataset. https://www.kaggle.com/secareanualin/football-events

[2] Fernandez, Matthew. Ulmer, Ben. “Predicting Soccer Match Results in the English Premier League”. Web. December 2. http://cs229.stanford.edu/proj2014/Ben%20Ulmer,%20Matt%20Fernandez,%20Predicting%20Soccer%20Results%20in%20the%20English%20Premier%20League.pdf

[3] Decroos, Tom. Dzyuba, Vladimir. Van Haaren, Jan. Davis, Jesse. “Predicting Soccer Highlights from Spatio-temporal Match Event Streams”. Web. December 2. https://people.cs.kuleuven.be/~vladimir.dzyuba/papers/predicting_soccer_highlights_from_spatiotemporal_match_event_streams-aaai2017.pdf