Predicting Goals in Football: Estimating the Probability of Scoring from Shots by FC Barcelona Players

Oriol Garrobé and Dávid Hrabovszki

07 September 2020

Abstract

The world is changing with the advance of technology and the vast amount of data available, and so does sport. Team sports such as football, are starting to introduce mathematical methods to analyse performances, recognize trends and patterns and predict results. The purpose of this study is to develop a new metric called Scoring Probability using machine learning models that is capable of predicting when a shot is going to end up being a goal. Scoring Probability is inspired by the Expected Goals metric.

To do so, many different mathematical approaches are trained on a dataset that tries to represent the exact moment of the shot in as much detail as possible. Also, the dataset includes information about the players, their skills and the moment of the game. By doing so, a Logistic Regression algorithm is the one that yields the best performance when predicting goals, with an F1 score of 35.24%, improving the previous works on the field.

This study intends to provide new information and hidden insights to professionals - players, coaches or managers - in order to improve the game, looking for a better decision process when it comes to scoring a goal.

Introduction

Trying to play the best football possible in order to win all the titles is a quest that all the best teams in the world follow. Practicing methods have changed, and therefore football istelf. Because of the evolution of technology and the rapidly increasing amount of data - as done in other sports such as baseball - football is introducing data science techniques to its toolbox in order to improve the game.

Teams use data science and machine learning approaches to analyse which aspects of the games can be improved to become the best team. Artificial intelligence (AI) in sports is very relevant nowadays, and it changed the way coaches approach their practice sessions.

From this point of view, a large number of researchers have worked on sports analytics and in particular in football trying to build the state of the art models to find the hidden insights from the sport that the human eye cannot see by analysing data.

As mentioned, there is a vast number of papers regarding football, but there is still room for more. There are so many aspects of the game that can be analysed, and the game can be heavily improved using AI. This project is based on a metric from StatsBomb called Expected Goal (Larrousse 2019). The aim is to improve this metric by adding some other variables and trying different machine learning models that could give a better result.

The data used is the datased provided by StatsBomb itself. ("Open Dataset from Statsbomb" 2020)

Apart from the StatsBomb dataset, aiming to go one step further, ratings from the players in the game are used, since it is believed that are players that in the same circumstances achieve different results. This means that not only the scenario is considered in this project but also the actors. For this purpose player ratings are included to the dataset coming from the EA Sports FIFA video games.

Related work

Many researchers have worked on the topic of predicting goals and match outcomes in football from many different approaches. The most mainstream metric for this kind of analysis is the Expected Goals (xG), the origin of which is debated among experts, but the first paper published about it that has a very strong connection to what we call xG today was written by Pollard et al. in 2004. They used a logistic regression model to determine the success of a shot and identified important factors, most of which we also work with. (Pollard 2004)

Mackay used ridged logistic regression to predict goals, but his analysis was based on possession sequences, not just shots. He used event based data with player locations, similar to this paper. The analysis concluded that although involving events before the shot improves prediction, the last event of the sequence is by far the most important factor. (Mackay 2017)

Herbinet conducted an analysis that predicts match outcomes, but an important part of the paper is a shot xG generation, similar to ours. He developed a metric that uses some of the features included in the original xG and that we also use in this paper. He added coefficients that take into account if the goal is scored towards the end of the match, or if the team has more players on the pitch due to an opponent's red card. We managed to achieve higher accuracy and F1 score with our model described later in this paper. (Herbinet 2018)

Our research makes use of the database's spatial information containing coordinates of all the players to calculate features that are similar to those used by other analysts. We also included player ratings in the analysis, which is not part of a traditional xG model, therefore we call our metric Scoring Probability. With this addition, we aim to create a metric that predicts shot outcomes more accurately.

Methods

Dataset

The project is mainly based on a dataset from StatsBomb ("Open Dataset from Statsbomb" 2020). This is a free dataset provided in order to new research projects in football analytics. The data from StatsBomb includes very detailed and interesting features relevant for the project such as: location of the players on the pitch in any shot - including he position and actions of the Goalkeeper-, detailed information on defensive players applying pressure on the player in possession, or which foot the player on possession uses, among others.

The data is provided as JSON files downloaded from the StatsBomb Data GitHub page, in the following structure:

- Competition and seasons stored in competitions.json.
- Matches for each competition and season, stored in matches. Each folder within is named for a competition ID, each file is named for a season ID within that competition.
- Events for each match, stored in events. The file is named for a match ID.
- Lineups for each match, stored in lineups. The file is named for a match ID.

In particular, the dataset only contains information about FC Barcelona games in the spanish national championship La Liga from 2005 up until 2019. There are 6428 shots in the dataset, 1044 of which became goals. For our analysis, we split the data to training and test sets (70% and 30% respectively).

As stated earlier, player skills are also considered in this project. We gathered ratings of the shooters and the opponent goalkeepers from the FIFA video games by EA Sports ("FIFA," n.d.). Also, it is considered which is the preferred foot of the player, information sourced from the same database. More in particular, this player information is obtained from *FIFAindex* (FIFAindex, n.d.). We downloaded player ratings for all the seasons that are relevant for our dataset to account for changing ability levels.

Data Preprocessing

Data from StatsBombs comes in JSON format. One easy way to work with JSON data in R is through the **jsonlite** package that transforms JSON data - which is a combination of nested lists - to nested dataframes.

The StatsBomb dataset containing all the relevant information for the project is the **Events** dataset. Each data point in this data set is an event that occurred in a specific game, such as: pass, block, dribble or shot, among others.

The first step working with the dataset is to erase all the incomplete datapoints or those with wrong information that would afterwards make the models fail. From this point, and because this project is only focused on shots, the dataset is filtered by the variable **type** which contains the type of action of the event. This variable is filtered so only the events regarding shots remain. A **shots** dataset, which is much smaller in size, than the one used before is created. We also filter out penalties and free kicks, because our analysis focuses on open play shots only.

It is important to emphasize the fact that this dataset contains extremely useful information which is enclosed in the variable called **shot.freeze_frame**. This **shot.freeze_frame** states where all the players are positioned on the pitch at the moment of the event. It is considered so relevant since it allows to make a perfect picture of the scenario at the moment of the shot. From this regards, and using the information in **shot.freeze_frame**, a number of new features can be computed and added to the dataset. More in particular, geometric or spatial features regarding the position of the striker, the defenders and the goalkeeper are computed. Such features will allow to know, for instance, what the shooting distance is, whether the goalkeeper is properly positioned or if there are defenders close enough to disturb the striker. These calculated features will be introduced in more detail in the Feature engineering part of this paper.

Finally, every data point from the *shots* dataset, containing also the geometric features computed, is complemented with information on the players. For instance, the ratings of the striker and the goalkeeper in action are added or whether the shooter is using his preferred foot or not.

Finally, some features from the **shots** dataset which are not relevant to the study are removed, creating a dataset for analysis called **data** that will be the one used to train and test the models. The final features in the **data** dataset can be seen in Table 1.

Table 1: Features used for modelling.

Variable	Definition
id	Numeric. Unique number representing a shot.
strong_foot	Boolean. States whether the player used the strong foot or not.
overall_rating	Numeric. Overall rating from FIFA ratings.
shot.power	Numeric. Shot power from FIFA ratings.
shot.finishing	Numeric. Shot finishing from FIFA ratings.
gk_rating	Numeric. Overall rating for goalkeeper from FIFA ratings.
gk.reflexes	Numeric. Goalkeeper reflexes from FIFA ratings.
gk.rushing	Numeric. Goalkeeper rushing from FIFA ratings.
gk.handling	Numeric. Goalkeeper handling from FIFA ratings.
gk.positioning	Numeric. Goalkeeper positioning from FIFA ratings.
shot.first_time	Boolean. States if the shot was taken first time.
under_pressure	Boolean. States if the shooter was under pressure or not.
home	Boolean. States whether the shooter is playing home or away.
dist	Numeric. Distance to center of the goal from shooter.
angle	Numeric. Angle of the shot from the shooter (in degrees).
obstacles	Numeric. Number of players (teammates & opponents NOT including GK) between goal and shooter (inside the triangle of the goalposts and the shooter).
pressure_prox	Numeric. Distance from closest opponent to shooter.
pressure_block	Boolean. States whether the closest opponent can save the shot by being inside the triangle.
gk_obstacle	Boolean. States whether the goalkeeper can save the shot by being inside the triangle.
gk_pos	Numeric. Goalkeeper's positioning, best if gk is standing on the line that halves the angle of the shot (value between 0 (angle is the same), to 1 (angle is halved).
gk_pos_adjusted	Numeric. Same as gk_pos, but it is less strict with shots with a tight angle.
gk_dist_from_player	Numeric. Distance between goalkeeper and shooter.
gk_dist_from_goal	Numeric. Distance between goalkeeper and the center of the goal.
goal	Binary. 1 if the shot is goal, 0 if the shot is not goal.

Feature engineering

We created new features based on player locations for every shot in the dataset, similar to traditional xG metrics. The difference is that we calculated many more variables that the traditional approaches do not take into account. The StatsBomb dataset contains the x and y coordinates of the shooter and other players that are relevant to the shot (both teammates and opponents including the goalkeeper). In the next part we explain what features we created and how.

Distance Euclidean distance between the shooter and the center of the goal.

Angle Angle of the shot from the shooter's point of view in degrees. Picturing a triangle (later referring to this as the triangle), where one side is the goal line and the other two are the imaginary lines connecting each goalpost to the shooter, we need the angle opposite of the goal line. We used the following formula:

$$\alpha = \arccos\left(\frac{b^2 + c^2 - a^2}{2 \cdot b \cdot c}\right) \cdot \frac{180}{\pi}$$

where a is the goal line, and b and c are the imaginary lines between the shooter and the posts (Calculator, n.d.).

Obstacles Number of players (teammates and opponents not including the opponent goalkeeper) between the goal and shooter – in other words, inside the triangle defined by the shooter and the goalposts. To calculate this, we first need to evaluate if a player is inside said triangle or not. If the area of this triangle is equal to the sum of the partial triangles defined by:

1. The shooter, one goalpost and the player being evaluated
2. The shooter, the other goalpost and the player being evaluated
3. The two goalposts and the player being evaluated

Then we conclude that the player being evaluated is inside the main triangle, therefore he is an obstacle. To calculate the area of a triangle based on the coordinates of its points, we used the following formula:

$$area = \left| \frac{a_1 \cdot (b_2 - c_2) + b_1 \cdot (c_2 - a_2) + c_1 \cdot (a_2 - b_2)}{2} \right|$$

where a, b and c are numeric vectors of length 2 (x and y coordinates) representing the points of the triangle ("Check Whether a Given Point Lies Inside a Triangle or Not," n.d.).

Pressure proximity The Euclidean distance between the shooter and the opponent closest to him.

Pressure block The closest opponent's physical ability to block the shot by being inside the triangle defined by the shooter and the goalposts. Boolean value.

Goalkeeper obstacle The opponent goalkeeper's physical ability to save the shot by being inside the triangle defined by the shooter and the goalposts. Boolean value.

Goalkeeper positioning Positioning of the opponent goalkeeper. The value is between 0 and 1, where a value of 1 means that the goalkeeper halves the angle of the shot, while a value of 0 means that the angle remains the same. This does not take into account if the goalkeeper is standing on the line or right in front of the shooter, only that he is positioned on the line that halves the angle of the shot.

We used a Gaussian kernel that can take a value between 0 and 1 depending on the input. The input, in this case, is the ratio of the angle split by the goalkeeper and the full shot angle. A larger kernel width means that the kernel distinguishes less between bad and good goalkeeper positioning, while a small value means that only angle ratios close to 0.5 can get a good mark for goalkeeper positioning, as it can be observed on Figure 1. We used a kernel width of 0.2 for this feature.

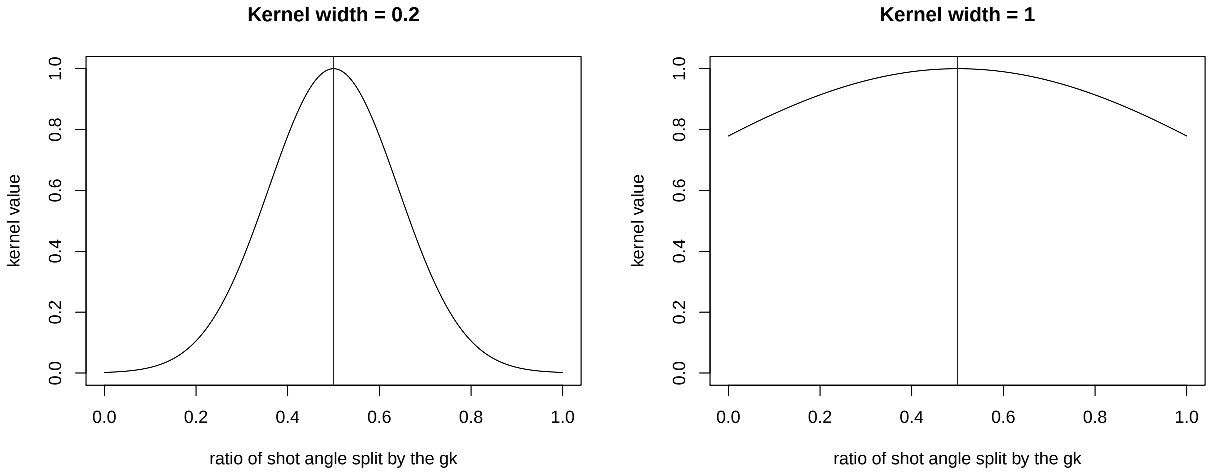

Figure 1: Gaussian kernels that give a higher rating to values around 0.5 (perfect split angle by the goalkeeper). Kernel width controls how strict the kernel is.

This feature was not included in the analysis, because it proved to be too strict when dealing with tighter shots.

Goalkeeper positioning adjusted Very similar to Goalkeeper positioning, but this feature is adjusted with the shot angle, meaning that it takes into account the full shot angle when evaluating goalkeeper positioning. This is necessary, because the previous variable was too strict when it came to tight shots, and it gave a bad value for the goalkeeper, even though he was still positioned pretty well. For example, if the angle of the shot was 5 degrees, the goalkeeper should not be given a bad mark for his positioning if he splits the angle to 1 and 4 degrees. With such tight angles, it does not really matter where he stands, as long as he is obstructing the goal, of course.

We solved this problem by introducing another Gaussian kernel that gives a high output value for small input values (full shot angles) and outputs a value close to 0.2 for larger input values (full shot angles). The kernel width in this case was chosen to be 20 (see Figure 2). Then, this adjusting kernel value was used as the kernel width for the final Gaussian kernel described above in the previous feature. This way we achieved that the Goalkeeper positioning adjusted feature is more lenient with tight shots than Goalkeeper positioning, thus more accurate in evaluating real life goalkeeper positioning. For example, for a shot that has an angle of 10 degrees, the adjusting kernel value will be around 1, which will result in a final kernel that takes the shape of the right figure in Figure 1. A shot angle of 50 degrees however, will get an adjusting kernel value of 0.2, therefore the final kernel will have the shape of the left figure in Figure 1.

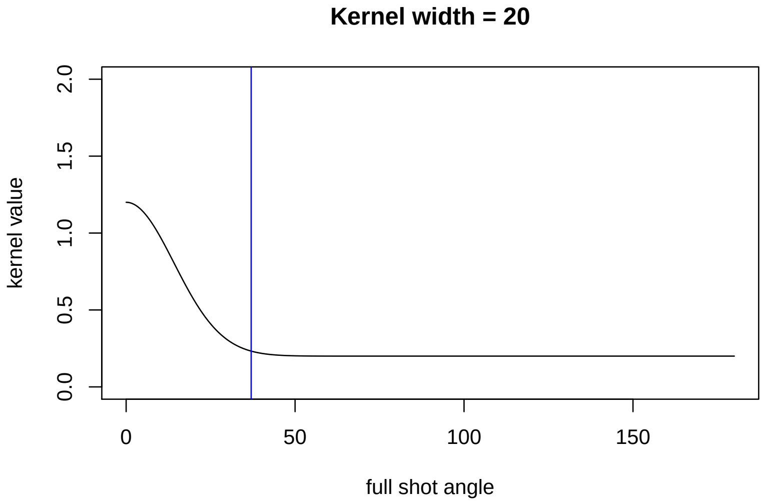

Figure 2: Adjusting Gaussian kernel that gives more weight to lower values (shot angles)

Goalkeeper distance from player Euclidean distance between shooter and goalkeeper.

Goalkeeper distance from goal Euclidean distance between center of the goal and goalkeeper.

Modelling

Logistic Regression

From the moment that one tries to develop a classification algorithm, the first choice usually is the Logistic Regression. This study is no different.

The Logistic Regression model is based on the logistic function (sigmoid) which is an S-shaped curve that takes any real number and maps it to a value between 0 and 1, but never reaches these numbers. From this point, by setting a threshold, also between 0 and 1, it is possible to divide data points in two classes. For instance, if the threshold is set at 0.7, those data points with a probability higher than 0.7 will be classified as one class and those with a probability lower than 0.7 will be classified as the other. The regression coefficients (betas) of the Logistic Regression are estimated by training the data.

In particular, the threshold set for this study a threshold of 0.5. In order to run the Logistic Regression algorithm, the package **glm** (Schlegel 2019) from CRAN is used. We chose this value of threshold, because it made the most sense to classify shots as goals that have a Scoring Probability of more than 50%. This also gave us the best overall accuracy, and a good balance between precision, recall and F1 score. These metrics could have been improved by changing the threshold, but only at the expense of the others.

We used 70% of the dataset for training and 30% for testing. We also made sure that the ratio of goals to not goals is the same in both subsets using stratified sampling.

The trained model estimated the following coefficients for the features that were considered statistically significant (see Table 2). Shooting with the strong foot has the largest positive effect on the probability of scoring, but a goalkeeper standing far off the goal line and a larger angle of the shot also help the shooter. It is also better for scoring if the closest opponent is far from the shooter. Good goalkeeper positioning, many players obscuring the goal and the possibility of getting the shot blocked all have a negative effect, understandably. It is interesting that the player's finishing ability only has a small positive influence on the probability of scoring.

Table 2: Estimated coefficients of significant features in logistic regression

Feature	Coefficient
strong_footTRUE	0.368
pressure_prox	0.182
gk_dist_from_goal	0.103
angle	0.030
shot.finishing	0.016
gk.positioning	-0.022
obstacles	-0.184
pressure_blockTRUE	-0.464

Random Forest and AdaBoost

Both Random Forest and AdaBoost models are good choices when training a classification model. These models are built on the decision tree model.

The Random Forest model in particular generates a number of trees based on a bootstrapped subset of the sample and only considers a subset of variables at each step. The result is a wide variety of trees. From this point the data is run over all trees and that yield a decision, the decision given by more trees is the one that prevails. This is called Bagging. To run the Random Forest model, the package **randomForest** (Breiman and Wiener 2018) from CRAN is used.

AdaBoost on the other hand also generates a number of trees, but only trees that have a root node and two leaves with no children, these trees are called stumps and are not great at making accurate classification, they are weak learners. Once one stump is made, the data is run through it, and a decision is made. The errors made by this stump influence when creating the next one and so on. Therefore, some stumps have a greater impact to the model. To run the AdaBoost model, the package **mboost** (Hofner 2020) from CRAN is used.

In order to get the best model possible and to not make it too computationally expensive, in both models it is important to choose the optimal number of trees. To do so, both algorithms are run a number of times with different amount of trees. Based on the error rates, the best model is chosen, and afterwards trained with the data to get the best results possible. The error rates against the number of trees can be seen in Figure 3. The smallest error rate for training data in the Random Forest model appears when 30 trees are created, and for the AdaBoost model it appears when 40 trees are used. Therefore, the final models are set.

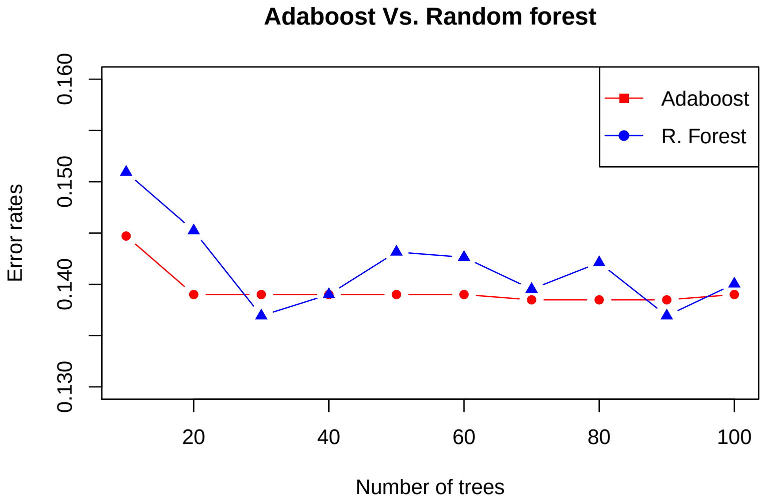

Figure 3: Random Forest and AdaBoost models error rates against the number of trees used.

K-NN model

The K-NN (K-Nearest Neighbours) model is a classification model that identifies the k number of closest labelled points to the one studied and estimates its class. The class chosen is the one with the bigger amount of neighbors with said class. This is a powerful algorithm but it is very important to choose the optimal number of neighbours, in other words, choose the value of k. This value is chosen with Cross Validation (CV), which is a model validation, which partitions the data in complementary subsets, performs analysis on one subsets and validates on the others in order to give a more accurate model. The final number of neighbours used in the KNN algorithm is 9, which gives the best result for prediction. To run the KNN algorithm with CV, the package **kknn** from CRAN is used.

Results

The results are evaluated using the following metrics:

- **Accuracy**. Stands for the ratio of correctly predicted observations over the total observations. If it is high, it means that there are a lot of good predictions, but it must be taken into account that for imbalanced datasets - where one class is more frequent than the other - it can be biased. In particular, in the dataset used there are much fewer shots that ended up being a goal than those that did not. So, with a naive predictor all the datapoints could have been set to not goal/miss and the accuracy would have been still high. That is, other metrics must be used in order to take this into account.
- **Recall**. Otherwise called sensitivity, stands for the correctly predicted positive observations over all observations in the actual positive class. This metric checks how many properly predicted points are predicted within a class, in this case **goal**. This means that it tries to show how many predicted goals are actually goals. Since in football there are not many goals, it is a very innacurate metric, as many times a situation that should clearly end up in goal in the end does not. This is why this one is not a very indicative metric for the purposes or the project. However, it is good to compute it in order to get some conclusions.
- **F1 score**. This metric is a weighted average of precision and recall, where precision is the number of properly predicted observations in the positive class over all the observations from that class. From this point, it takes both false positives and false negatives into account too, it is very useful when the goal

is to predict uneven classes. In this case, as there are much more observations that are not goal than those that are goal, this metric is the one that mirrors the quality of the model best.

The results that the different models yielded are in Table 3, along with the xG results for comparison (these Expected Goals probabilities were part of the downloaded dataset from StatsBomb).

Table 3: Comparison of models with StatsBomb's xG.

Model	Accuracy (%)	Recall (%)	F1 Score (%)
xG StatsBomb	86,36	21,79	34,09
Logistic Regression	85,89	23,72	35,24
Random Forest	60,00	22,05	32,24
AdaBoost	83,33	11,18	19,71
K-NN	68,50	27,79	39,54

Discussion

In this study, we developed a metric to quantify the probability of a football player scoring from a shot using machine learning models, based on the positioning of the players on the pitch, their skills and a few other factors. Four classification approaches using 21 variables, which derived from the particular scenario at the moment of the shot, were examined. It has been proven that it is possible to classify whether a shot is going to be a goal or not with good accuracy, the logistic regression being the best performing model with an F1 score of 35.24%, a recall of 23.72% and an accuracy of 85.89%. It also improved the xG metric from StatsBomb by adding player skills to the dataset and testing other machine learning approaches. This will provide professionals of the sport such as players, coaches or managers some insights to the game that can be very useful to improve their performance.

Models based on decision trees such as Random Forest or AdaBoost provided mixed results that did not improve the previous works on the field. In particular, the optimal Random Forest algorithm with 30 trees yielded an F1 score of 32.24%, a recall of 22.05% and an accuracy of 60%. Looking at these results, it can be seen that even though the F1 score is close to the one from xG, the overall accuracy is poorer, which means that many shots that were classified as miss were actually a goal. On the other hand, the optimal AdaBoost algorithm with 40 trees yielded an F1 score of 19.71%, a recall of 11.18% and an accuracy of 83.33%. All the results are worse than the xG metric by StatsBomb, bringing no improvement to the field. This model failed as almost no goals were predicted, only 11.18% (recall) of the goals were predicted properly. From this point of view, these two algorithms do not bring improvements to the field or relevant information that can be used to improve the game.

Another model used to classify goals is the K-NN algorithm. This algorithm yielded better results than models based on decision trees with an F1 score of 39.54%, a recall of 27.79% and an accuracy of 68.50%. It had the best F1 score of all models used and also improved the xG metric. As stated before, the F1 score is the metric that mirrors the quality of this algorithm best, and this method could be the one chosen for the project. It also had the best recall among all the models. However, it had a significantly lower overall accuracy compared to xG or Logistic Regression. It yielded a better F1 score and a better recall due the fact that it predicted much more goals than Random Forest, Adaboost or xG. However, many situations that were not goals were classified as goals, this is why this model has a lot of room to improve.

After building several machine learning models, we conclude that logistic regression is the best suited for this problem, because it produced a relatively high F1 score, while keeping the overall accuracy high as well. Therefore, going forward we apply logistic regression in this paper.

Player performances

In the previous sections we engineered relevant features, applied models to predict goals, chose the best one, and now we are going to interpret the results on the player's level. We aim to find out how well each player performs from the aspect of converting shots to goals. We suspect that there are players that score more goals than they are expected to, and there are some that score fewer. It is important to note, that the database we used for analysis had 1044 goals, while we only predicted 320 goals with our best model for the whole dataset. Therefore, to make the amount of actual and predicted goals by each player comparable, we scaled up the amount of predicted goals for the sake of this performance analysis, so they also sum up to 1044.

The results can be observed in Table 4, which is ordered by the amount of goals scored. It is no surprise that Lionel Messi scored 33.85 more goals (11% more) than he should have, based on his chances. This is in line with our belief that he is a very efficient striker. Even more efficient than Messi are Ivan Rakitic and Daniel Alves. The latter player is primarily a defender, but still managed to score 11 goals, even though none were predicted for him.

The most underperforming players, when it comes to converting chances, were Samuel Eto'o and Zlatan Ibrahimovic. Luis Suárez also scored much fewer goals (27.76), than expected.

Table 4: Player performances (who scored more than 10 goals)

Name	Goals predicted (scaled up)	Goals scored	Diff. (amount)	Diff. (ratio)
Lionel Andrés Messi Cuccittini	300.15	334	33.85	0.11
Luis Alberto Suárez Díaz	133.76	106	-27.76	-0.21
Samuel Eto"o Fils	104.40	62	-42.40	-0.41
Pedro Eliezer Rodríguez Ledesma	55.46	48	-7.46	-0.13
Neymar da Silva Santos Junior	58.73	46	-12.73	-0.22
Thierry Henry	42.41	32	-10.41	-0.25
David Villa Sánchez	35.89	31	-4.89	-0.14
Xavier Hernández Creus	26.10	31	4.90	0.19
Alexis Alejandro Sánchez Sánchez	19.58	29	9.42	0.48
Andrés Iniesta Luján	22.84	25	2.16	0.09
Gerard Piqué Bernabéu	13.05	25	11.95	0.92
Ivan Rakitic	6.53	22	15.47	2.37
Francesc Fàbregas i Soler	13.05	20	6.95	0.53
Bojan Krkíc Pérez	9.79	19	9.21	0.94
Seydou Kéita	9.79	12	2.21	0.23
Zlatan Ibrahimovic	19.58	12	-7.58	-0.39
Daniel Alves da Silva	0.00	11	11.00	Inf
Ronaldo de Assis Moreira	6.53	11	4.47	0.69

The next section will present some shots visually that might help explain these differences.

Visualising shots

Now we plot some of the shots to illustrate how such differences can occur between reality and prediction, and also to show some chances that had a high Scoring Probability, but were missed, or had a low probability, but still went in. We used the package **soccermatics** (Gallagher 2018) to plot the empty pitch, and then we wrote our own function that places the players and the shot itself on the pitch.

Figures 4 and 5 show two goals from Daniel Alves that he scored against the odds. These efforts contribute to him performing much better, than expected (see Table 4 in the previous section).

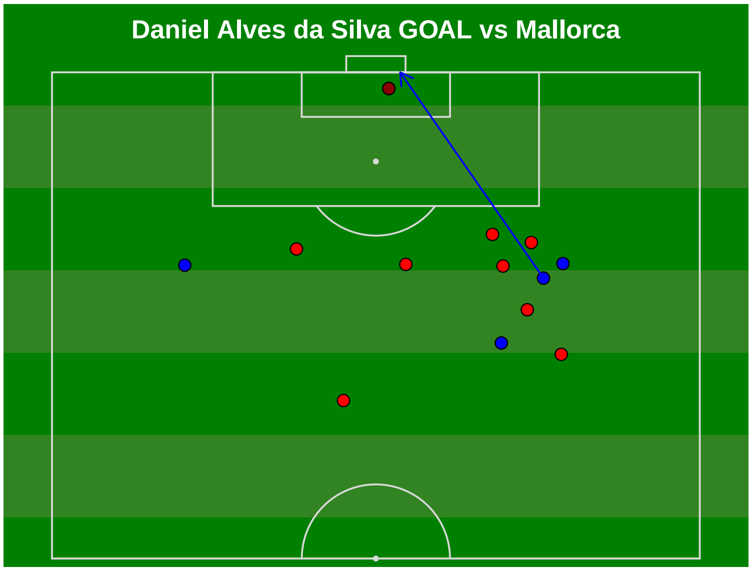

Figure 4: Daniel Alves goal (Scoring Probability = 0.029)

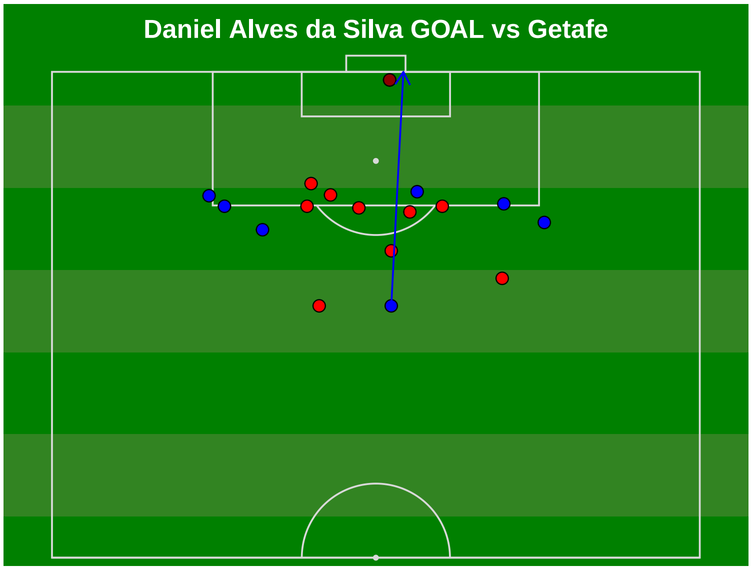

Figure 5: Daniel Alves goal (Scoring Probability = 0.038)

Other shots however, were not converted despite having a large Scoring Probability. Ludovic Giuly's header for example flew over the crossbar (Figure 6).

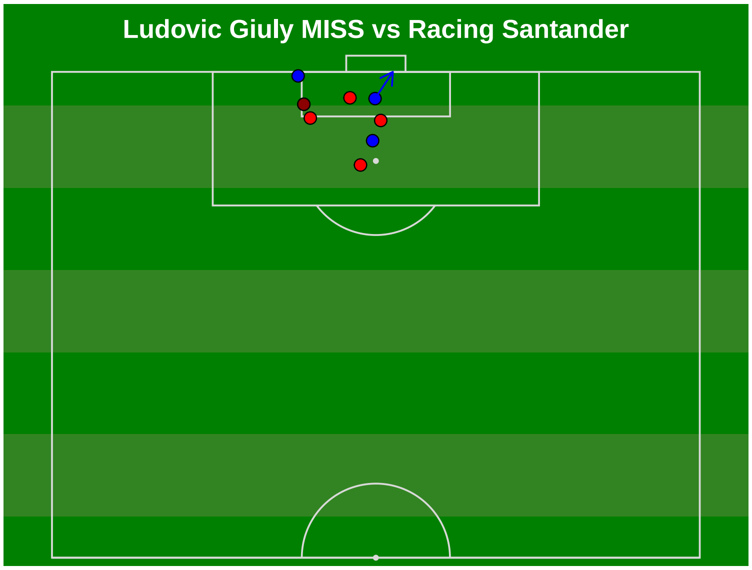

Figure 6: Ludovic Giuly miss (Scoring Probability = 0.886)

Samuel Eto'o failed to score from a relatively easy position (Figure 7) against Real Madrid.

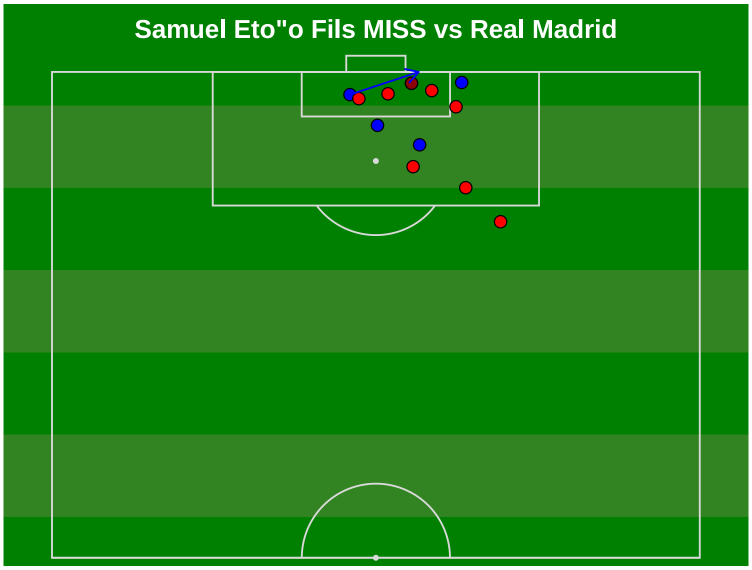

Figure 7: Samuel Eto'o miss (Scoring Probability = 0.823)

Finally a Lionel Messi goal against Levante, where he had less than 50% probability of scoring, but still managed to put the ball into the net.

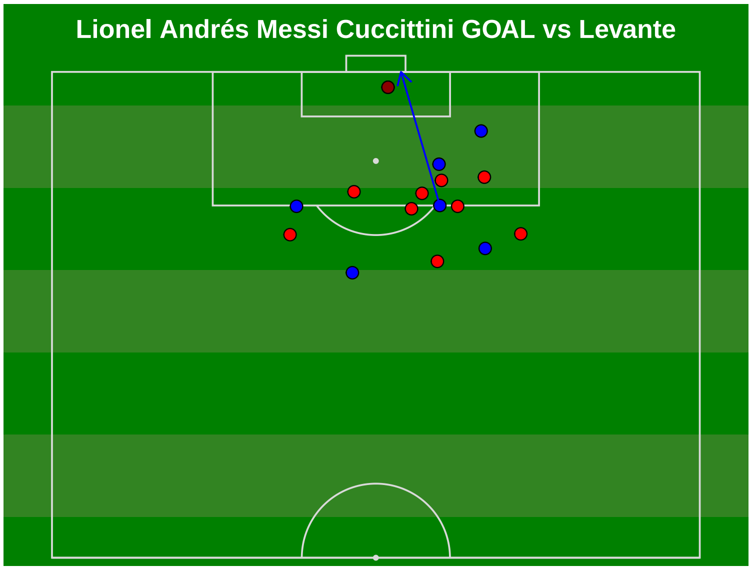

Figure 8: Lionel Messi goal (Scoring Probability = 0.120)

Future improvements

The limitations of this study must be acknowledged. The data sample consist only of **FC Barcelona** games in the national regular season **La Liga**. From this regards, Barcelona players - who are known to be very effective - playing against worse teams can lead to some biased results. It is also worth to be mentioned than as the dataset starts in 2005, many of the shots are from **Lionel Messi**, who is considered the best player in history and definitely one of the best strikers of all times. This can also lead to not very relevant results for the football community. This is why this project used player ratings, as an attempt to introduce this information to the models, that can be used therefore at all levels. It would be good therefore, to add more shot situations from many different competitions from different countries and different teams. This would give a more realistic dataset and also it would provide more data points that would definitely help train the algorithm, probably achieving better results.

Another possible improvement would be to create more variables. With the dataset provided the authors created a dataset that comprised all those variables that could have an effect on the result of the shot. New variables were computed giving a proper dataset to work on. However, more variables could have been added or created. With more variables, a better picture of the situation of the shot could be drawn. With this new dataset other machine learning approaches, more modern and complex, could have been applied. For instance, it might be good to train a Neural Network in order to predict goals. In order to do so, more computational resources are needed, that is why Neural Networks are not used in this project, but could probably improve the results.

The aim of this analysis was to predict goals from shots, but we deviated from the traditional xG metric by including player ratings in the models. The ratings however are just - mostly - subjective assessments of players by experts, who created the FIFA video game. This introduces bias into the model, so a better way of including player abilites might be to build separate models for each player without their FIFA rating. This is of course much more computationally expensive and we did not have enough data for every player to carry this out anyway.

An extension to this study would be to apply the same methods to other types of events other than **shots**, such as success in **passes** or **tackles**. This would provide information not only to the strikers but to every player on the pitch, improving - as this is the aim of football analysis - the game.

A probable future use for this analysis could be a system that coaches could use to show a player positions where he should have passed the ball to a teammate instead of shooting, because he would have had a higher Scoring Probability. This would mean that the system would look at all the teammates' positions and calculate the Scoring Probability for them as well.

Conclusion

Football is a very demanding sport, and the high impact that it has in society makes teams strive for perfection. From this point, game demands are extensively analysed and new concepts coming from data science are introduced to coaches' plans. This project created a new metric called Scoring Probability, that uses machine learning models to predict goals from shots. Our model successfully predicts whether or not a shot will be converted to a goal by the shooter, by taking into account several spatial features along with more traditional variables and player ratings. We proposed several possible improvements and extensions to this project that can make it into a useful tool for football coaches.

The algorithm, built on a simple logistic regression and 21 relevant features created the Scoring Probability metric that can be used for goal prediction with a combined precision and recall of 35.24% (F1 score), a recall of 23.72% and an accuracy of 85.89%, improving the results of the xG metric from StatsBomb. This study aimed to improve the football analysis community and football itself providing a tool to assess shooting scenarios in professional football.

References

Figure 9: Statsbomb logo

Breiman, Cutler, Leo, and Matthew Wiener. 2018. "Breiman and Cutler's Random Forests for Classification and Regression." *CRAN*.

Calculator, Omni. n.d. "How to Find the Angle of a Triangle." https://www.omnicalculator.com/math/triangle-angle#how-to-find-the-angle-of-a-triangle.

"Check Whether a Given Point Lies Inside a Triangle or Not." n.d. Geeks for Geeks. https://www.geeksforgeeks.org/check-whether-a-given-point-lies-inside-a-triangle-or-not/.

"FIFA." n.d. Electronic Arts. https://www.ea.com/en-gb/games/fifa/fifa-21.

FIFAindex. n.d. "Player Stats Database." https://www.fifaindex.com/players/top/.

Gallagher, Joe. 2018. "Visualise Spatial Data from Soccer Matches." *CRAN*.

Herbinet. 2018. "Predicting Football Results Using Machine Learning Techniques." *Imperial College London*. https://www.imperial.ac.uk/media/imperial-college/faculty-of-engineering/computing/public/1718-ug-projects/Corentin-Herbinet-Using-Machine-Learning-techniques-to-predict-the-outcome-of-profressional-football-matches.pdf.

Hofner, Benjamin. 2020. "Model-Based Boosting." *CRAN*.

Larrousse, Benjamin. 2019. "Improving Decision Making for Shots." *StatsBomb*.

Mackay. 2017. "Predicting Goal Probabilities for Possessions in Football." *Vrije Universiteit Amsterdam*. https://science.vu.nl/en/Images/werkstuk-mackay_tcm296-849981.pdf.

"Open Dataset from Statsbomb." 2020. https://github.com/statsbomb/open-data.

Pollard, Taylor, Ensum. 2004. "ESTIMATING the Probability of a Shot Resulting in a Goal: THE Effects of Distance, Angle and Space." *International Journal of Soccer and Science*. https://www.researchgate.net/publication/240641737_Estimating_the_probability_of_a_shot_resulting_in_a_goal_The_effects_of_distance_angle_and_space.

Schlegel, Benjamin. 2019. "Predicted Values and Discrete Changes for Glm." *CRAN*.