STOR 565 Project - Building an xG Model for Soccer

Group 3 - Jarod Godwin, Dennis Perumov, Neo Zhu, Katie Dixon, and Rawad Nasrallah

2023-12-06

Project Summary

In this project, we attempt to build a model that predicts the xG, or expected goals, of any given shot taken within a soccer match. Because relatively few goals are scored in a given match, the randomness of whether good chances are converted can sometimes influence a match's outcome more than the game's dominance or the quality of chances created. To fill the gap, xG is a statistic that is able to explain how many goals a team could have expected to score given the chances it created, thus being more descriptive of how a team performed in a match. If a team won 3 to 0 against an opponent but trailed in xG 0.5 to 2.5, we can attribute their win to either luck or world-class shooting.

We obtain data from StatsBomb, a commercial soccer data provider, serving as a launchpad for our investigation. From a dataset containing information regarding each recordable action in soccer matches over a given season, we prepare a dataset of roughly twenty-thousand shots to train and test our models. We model xG by using a variety of given and engineered features: distance to goal, angle to goal, the number of defenders and attackers on the ball, and many more. To provide xG estimates, we select machine learning models that are able to provide binary class probability predictions. We evaluate the efficacy of various logistic regression, random forest, and XGBoost models in calculating xG, and are able to attain xG estimates that are comparable overall to xG metrics calculated and provided by StatsBomb.

After building and evaluating our xG models, we produced a few examples of how xG can be used in practice. We produce visualizations regarding which types of shots are worthwhile and have a high probability of resulting in a goal, as well as which types of shots may not be worthwhile and have low probability of resulting in a goal. This can lead to teams focusing their attention on creating opportunities that lead to a higher xG, thus boosting their chances of winning. We also produce analyses of players using xG, similar to how professional clubs might in their scouting and player-evaluation efforts.

AI Usage Disclaimer

We attest that this project made no use of AI. The work within this project is solely the work of its authors unless cited as otherwise.

Member Contribution Statement

Each member of the group collaborated to produce each section of the work displayed. Katie was especially helpful in creating our slideshow. Dennis was especially helpful in interpreting each model. Rawad was especially helpful in generating commentary for the final report. Neo was especially helpful in gaining understanding of our data set. Jarod was especially helpful in developing the project idea.

Exploratory Data Analysis

To create a model for xG, we used data from StatsBomb, a soccer data provider. StatsBomb has a package within R (https://statsbomb.com/wp-content/uploads/2021/11/Working-with-R.pdf) that allows access to a portion of their data for free. There is match data for 67 entire seasons and tournaments. Match data includes variables such as shot outcome, player locations, angle to goal, and 180 additional variables that essentially capture every quantifiable event that occurs within a match. For this project, we pulled match data from the 2015/16 La Liga and Premier League seasons. Furthermore, we created a new dataframe that contains the shot event data for the seasons, as xG (expected goals) is a measure of shot/opportunity quality, while excluding irrelevant columns from our dataset (i.e. columns pertaining to non-shot related events like substitutions or passing). We also choose to exclude rows that contain shots taken from penalty and free kicks, as these would need to be modeled as part of a separate exercise.

As previously discussed, all the data comes from the 2015-2016 La Liga and Premier League seasons, which means the models apply to these cases but may not have significance for other leagues and seasons. Moreover, the data comes from leagues with some of the best players in the world, which could make the models inaccurate for many lower level soccer matches. Lastly, the xG models do not take into account the skill quality of the shooter or goalkeeper. The xG statistic is meant to be a player-agnostic measure of goal-scoring chance, and thus does not incorporate the differences in quality between players who may be shooting.

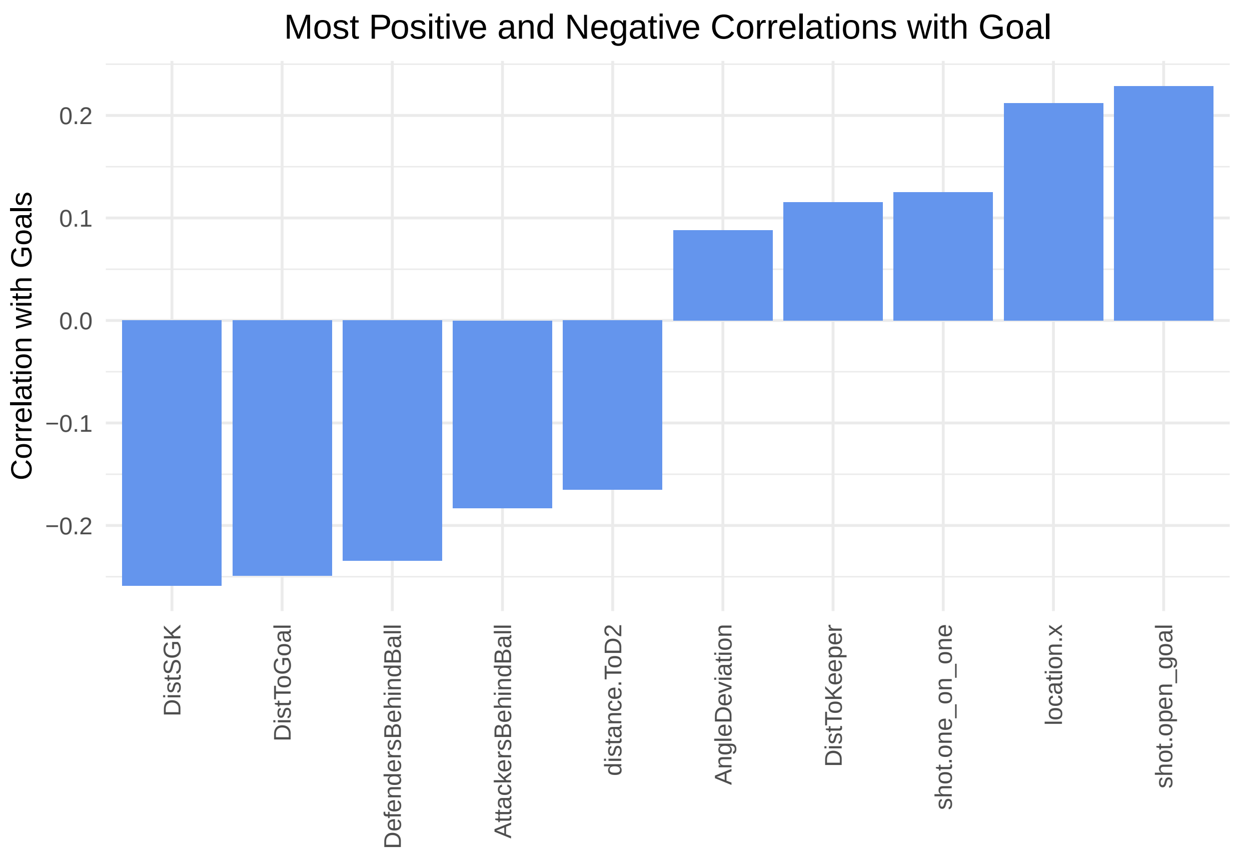
Table 1: Shot Data Field Categories

Shot Location	Shot Type	Shot Surroundings
location.x	shot.aerial_won	DistToKeeper
location.y	shot.aerial_won	DefendersBehindBall
DistToGoal	shot.technique.overheadkick	under_pressure
AngleToGoal	shot.follows_dribble	shot.open_goal
...	...	...

To familiarize ourselves with the data, we start by understanding the fields provided within the data. Ultimately, the data contains a few dozen fields that pertain to shot-related data. Because there are too many fields to enumerate specifically and in detail, we provide a brief overview of the categories of fields from which we select predictors in our models, as seen in the table above. We can see that there are a variety of fields that pertain to shot location, shot type, and shot surroundings. Intuitively, the xG of a shot, or its probability of resulting in a goal, seems up to the confluence of these three factors.

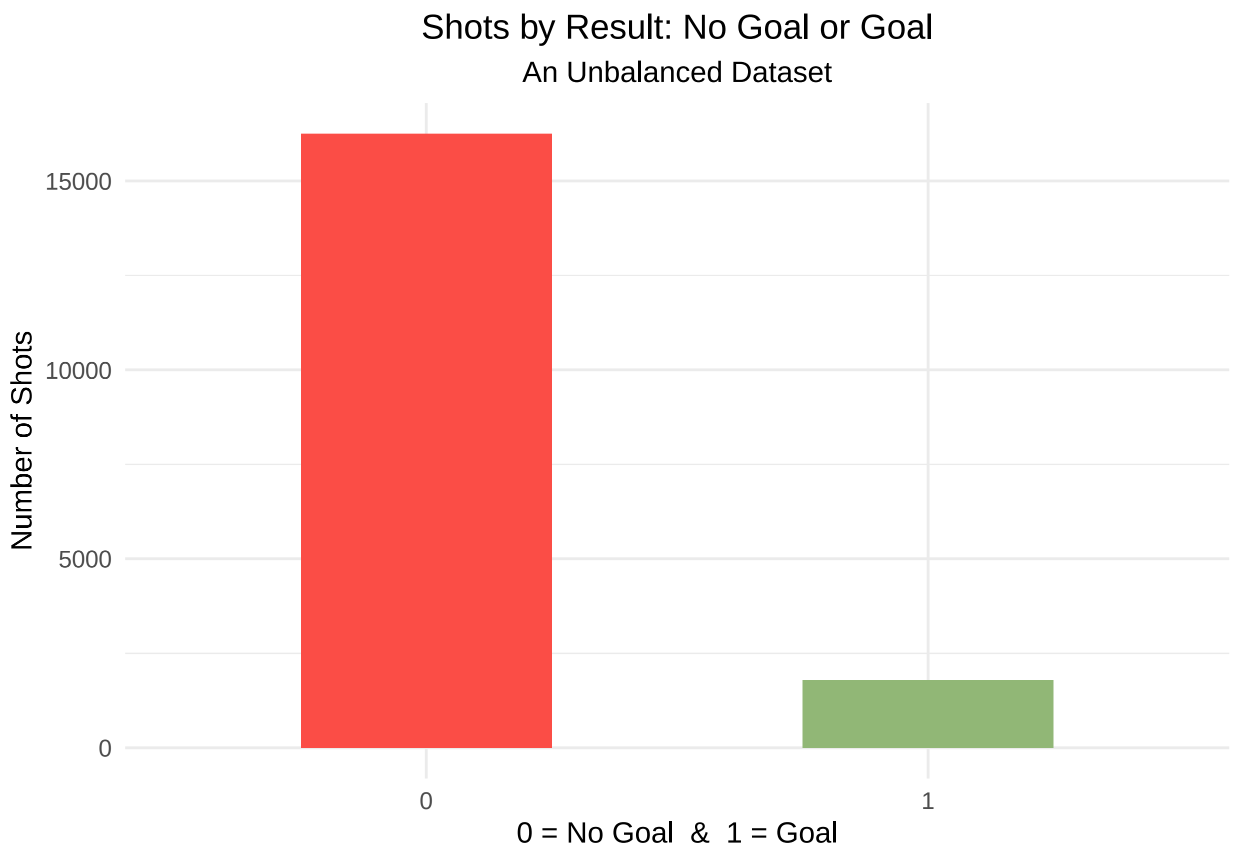
To prepare the relevant shot data fields for modeling, we performed feature engineering. This process included calculating a few predictors using the predictors in our set, attempting to address gaps within the set of data fields. For instance, we used the AngleToGoal field, which ranges from 0 to 180 degrees, to calculate an AngleToGoalNorm field, which ranges from zero to ninety and measures the angle deviation from directly in front of the goal. Additionally, we transformed categorical variables into modeling friendly variables via one-hot encoding.

To begin exploring relationships between variables in our data, we calculate the correlation of each variable with goals scored. This should provide insight as to which variables we can expect to increase xG, or the probability of a goal given a shot, within our model. From the correlation matrix, we produced a plot that displays the five most positive and five most negative correlations with goals scored.



We can see that the variables included in this plot are intuitive. The positive correlations include whether the shot was on an open goal (shot.open_goal), the proximity to goal (location.x), and whether the shooter was one-on-one with the goalkeeper (shot.one_on_one). Each of these intuitively portend to high probability of scoring a goal. Meanwhile, the distance from the shooter to the goalkeeper (DistSGK), the distance from the shooter to the goal (DistToGoal), and the number of defenders between the shooter and the goal (DefendersBehindBall), negatively correlate with goals scored. Again, high values in each of these qualities intuitively portend a shot with a low probability of scoring a goal.

Upon exploring our dataset via the plot shown below, we observe that roughly ten percent of shots result in goals, presenting us with an unbalanced dataset. To ensure the integrity of our models, we need to address the issues that may accompany modeling on unbalanced data, and make design decisions accordingly. In considering how to address the potential issues, we first consider the desired outcome of our investigation. We are not primarily interested in classifying shots as goals or not goals, in which case we might choose to assign different weights to our observations or resample our dataset. Instead, we are interested in xG, or the probability that any given shot results in a goal. We are interested in modeling both instances of shots with high probability of resulting in a goal and instances of shots with low probability of resulting in a goal. For this reason, AUC is among our preferred measures of model performance, as it assesses the performance of the model across different probability thresholds, or in our case, different levels of calculated xG. We also use balanced accuracy to assess model performance, as well as the sum of xG in our dataset relative to the sum of goals scored in our dataset, which ensures that xG estimates are not biased downwards by the class imbalance.



Model Fitting

Full Logistic Regression

We first split the data into training and test sets with an 80/20 ratio to fit the models. For the first model, we used full logistic regression with no variable selection, meaning that all shot-related features were kept in the model. We chose to use Full Logistic Regression since we are predicting the probability of an event's (a shot on goal) success or failure. We anticipate that using all possible features within our model may detract from performance by introducing too much model variance for the sake of low model bias. We aim to solve this issue in our next set of models by penalizing the inclusion of excess features in the model.

Penalized Logistic Regression

We used the following models to reduce overfitting by introducing a penalty term: LASSO, Ridge, and Elastic Net Logistic Regression.

For LASSO, we used 5-fold cross-validation to find the value of λ that minimized the cross-validation error. Our result was a λ value of ~0.0006. Since LASSO uses L1 regularization (adds the absolute value of the coefficients as a penalty term), we can see that the majority of the shot features were tuned down to zero, only leaving the most important variables such as AngleToGoal and DefendersBehindBall with coefficients.

```
## 33 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)                0.386434807
## under_pressure             .
## shot.first_time            .
## shot.one_on_one            0.009170621
## shot.aerial_won              -0.258445663
## shot.open_goal                1.527777625
## shot.follows_dribble          .
## DistToGoal                   -0.058126688
## DistToKeeper                  0.030409907
## AngleToGoal                   .
## AngleToKeeper                 .
## AngleDeviation                .
## DistSGK                      -0.060059527
## density                      -0.121303181
## density.incone                .
## distance.ToD1                 .
## distance.ToD2                 .
## AttackersBehindBall           .
## DefendersBehindBall          -0.056977915
## DefendersInCone              -0.076095736
## InCone.GK                    -0.237592452
## DefArea                       .
## distance.ToD1.360             0.016626862
## distance.ToD2.360             .
## shot.technique.halfvolley     .
## shot.technique.backheel       .
## shot.technique.volley         .
## shot.technique.overheadkick   .
## shot.technique.divingheader   .
## shot.technique.lob            .
## shot.body_part.head          -0.218446342
## shot.body_part.other          .
## AngleToGoalNorm              -0.012515637
```

In Ridge Regression, we used 5-fold cross-validation to find the value of λ that minimized the cross-validation error. Our result was a λ value of ~0.0078. Ridge uses L2 regularization as opposed to L1 regularization. L2 regularization adds the squared magnitude of coefficients as a penalty term, leaving every variable with a coefficient, unlike LASSO. Thus, every variable is included to a varying extent.

Elastic Net Regression combines elements of both L1 and L2 regularization in a regression model. In this way, Elastic Net Regression aims to combat some of the limitations that can arise when using LASSO and Ridge. Some of these limitations are inconsistent variable selection in LASSO in cases of multicollinearity and a lack of sparse solutions in Ridge. Another benefit of Elastic Net Regression is that it balances variable selection and coefficient shrinkage in cases of high dimensional data such as ours. Using 5-fold cross-validation, we calculated the λ that minimized the cross-validation error and ended up with λ = ~0.00086.

Random Forest

The next set of models we used were Random Forest models. We wanted to see if a sacrifice in interpretability would help us achieve an increased predictive accuracy. Our first model was a bagging model, with the following two models being random forest model.

With our Bagging Random Forest model, we decided to use bagging to reduce variance to combat overfitting. We set the number of predictors randomly selected at each split to 32 (mtry = p). Normally, in Random Forest models, we force the model only to consider a subset of the predictors; however, when we consider every predictor, some of the advantages are that no valuable information is lost and complex interactions between predictors are captured.

In the next Random Forest model, we set mtry = (pˆ0.5). Thus, in this model we are taking a subset of the predictors as opposed to using all of them in the previous model. Some of the advantages of using a subset of predictors are that model complexity decreases and introduces diversity among the trees. Increased diversity benefits the model with a more balanced complexity and reduction in overfitting.

In our third model, we decided to decrease the m value again by setting mtry = p/3. This means that we only consider a third of the total number of predictors. A further decrease in m can lead to increased diversity and the risk of not including significant predictors, which can also lead to an increase in bias. Given these drawbacks, we expect that this model may perform worse than the other two Random Forest models;.

XGBoost

For our last model, we decided to use XGBoost. We wanted to fit an XGBoost model to the data as it is an acclaimed model that boasts high predictive capability. Advantages of XGBoost include tree pruning (reduces model complexity and prevents overfitting) and the ability to handle high-dimensional datasets. We thought this model would perform well with our data since XGBoost incorporates both L1 and L2 regularization, similar to the previously mentioned regression models. By using 5-fold cross-validation, we determined the best number of boosting rounds to be 16 and used this to create our model.

Model Evaluation

Table 2: Model Evaluation Statistics

	AUC	Specificity	Sensitivity	Balanced Accuracy	Sum Total of xG	Total xG minus Goals
StatsBomb.xG	0.826	0.994	0.162	0.578	352.4	-5.6
Full.Logistic.Regression.xG	0.811	0.994	0.131	0.563	356.0	-2.0
LASSO.xG	0.811	0.996	0.120	0.558	356.4	-1.6
Ridge.xG	0.810	0.996	0.101	0.548	356.5	-1.5
Elastic.Net.xG	0.811	0.996	0.120	0.558	356.4	-1.6
Bagging.xG	0.781	0.990	0.137	0.563	398.1	40.1
Random.Forest.1.xG	0.793	0.991	0.131	0.561	386.1	28.1
Random.Forest.2.xG	0.793	0.990	0.145	0.568	393.7	35.7
XGBoost.xG	0.804	0.995	0.109	0.552	365.7	7.7

In the table above are various metrics with which we can measure the effectiveness of our xG models. In the top row are metrics for StatsBomb's proprietary xG model. The following eight rows display metrics for our models. First, in evaluating AUC, we can see a clustering of values near 0.81, with the logistic regression models and the XGBoost models reaching this level. These are just shy of the StatsBomb model AUC, which we view as a sign that our model provides good estimates of xG across various levels of xG between 0 and 1. We can also see that our random forest models have poorer AUC values than the rest of the models. Specificity and Sensitivity are included to display the proportions of no-goal and goal results that are correctly labelled by the model. We can see that each model is quite good at classifying no-goal results, and is quite poor at classifying goal results. Considering the majority of our dataset is composed of no-goal results, this results in only a relatively high accuracy. Additionally, we need consider why the models are so poor at predicting goal results. A person who regularly watches soccer might tell you that the majority of goals scored are not actually high xG chances, with most goals coming from shots where one does not expect to score the majority of the time. Thus, we would not expect goals within our dataset to necessarily have high xG values.

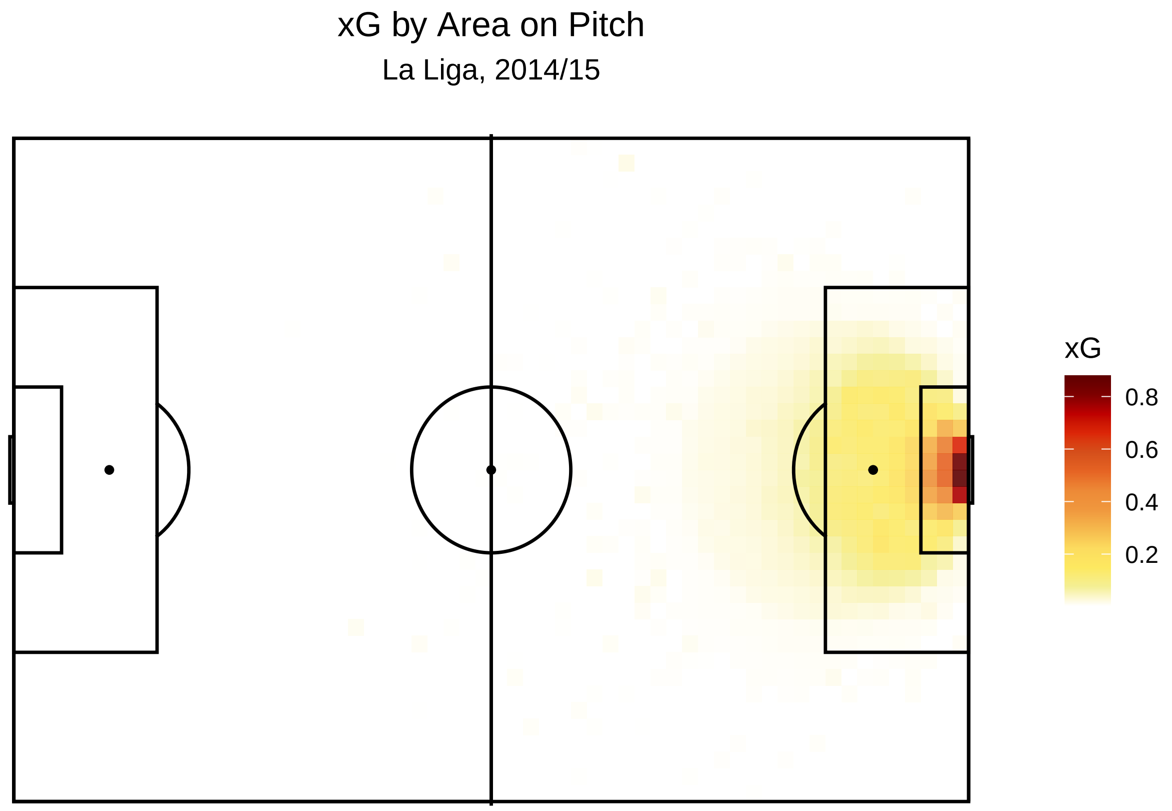
To evaluate each model's xG calculations, we can sum the total xG within our test dataset and compare that value to the total number of goals within the dataset. The law of large numbers necessitates that with a good model, these numbers should converge with a sufficiently large set of shots. We see that for our logistic regression models, these numbers differ by two goals or less, which is less than one percent of the goals within the test set. This is better performance than even the StatsBomb model, which sports a difference of 5.6 goals. Here, we can see the poor performance of our Random Forest models, which overstate total xG within the dataset by up to 40 goals. Our XGBoost model provides a good performance in this metric, but is an inferior option to logistic regression or StatsBomb models, with a difference of 7.7 goals.

Summarily, we are happy with the fit and output of our models, and present our penalized logistic regression models as the best of the group we tested. Within this group, we favor the LASSO Logistic Regression model for its sparsity, producing a model that requires less data and produces the same quality of xG calculation. This model boasts AUC and Total xG Minus Goals metrics that are in line with that of the industry standard StatsBomb xG model.

xG Model Uses

Thus far, we have demonstrated the process of building models for xG and testing for their performance and effectiveness. Now, we would like to pivot to demonstrating a few of the applications of xG, as to elucidate the usefulness of our modeling.

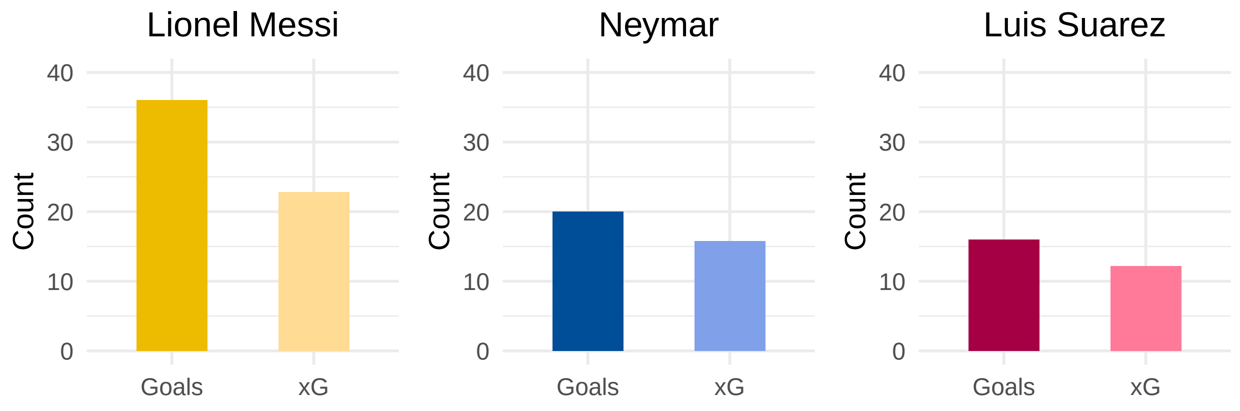
Shot xG Heatmap

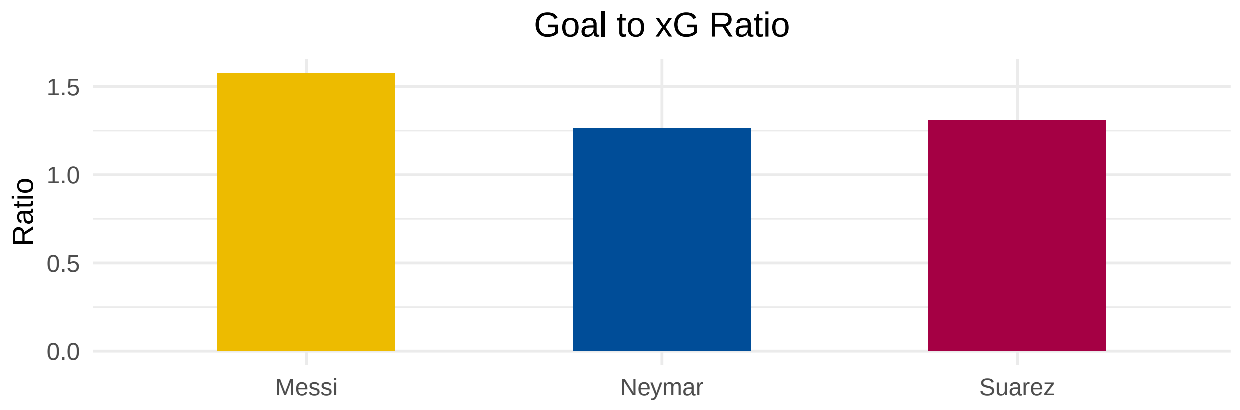


We can plot average xG for different regions on a soccer field—often called a pitch—in order to visualize the effect of location on the probability of a shot resulting in a goal. We can see in the figure that xG is highest in the region directly in front of the goal. Intuitively, this makes sense, as shots taken from directly in front of goal generally require the least amount of accuracy, and are likely to feature few obstructions in the form of other players. Also, we can see the decline in average xG as we move to regions that are either further from the goal, or at a larger angle relative to the center of the goal. Regions outside of the 18-yard box feature low average xG values. This information could be useful to players and coaches as they consider tactics. For a player, this type of visualization could inform them of areas in which it would or would not be advantageous to shoot. For a coach, this figure could guide the areas of the pitch they target in possession of the ball.

Player Analysis - Leo Messi xG vs Goals Scored

When evaluating player performance in soccer, the most popular metrics include goals and assists. But for measuring attacking output, there are few other observable quantities that can be used to evaluate the performance or quality of a player. The xG statistic provides another point of reference for judging the performance of a player. Our plot displays the output over a season of three attacking players from the 2014/15 Barcelona team—Lionel Messi, Neymar, and Luis Suarez. Traditionally, goals are used to measure the contribution of an attacking player over a season. However, the number of goals scored only conveys so much information about a player's output. We can see that with 36 goals, Lionel Messi was the most productive of the group, seeming to indicate that Messi performed quite well over the season. However, this doesn't tell us anything regarding how many goals you might have expected Messi to score given the positions he was in throughout the season. One wouldn't know from goals scored whether Messi had been wasteful or had been brilliant with his chances. By using our model to calculate the xG of each shot taken by Messi over the course of the season, and adding the xG of each individual shot, we find the xG of the entire season for Messi. For each attacker, we compare the ratio of Goals to xG to determine who was the most efficient with their chances.





We conclude a number of things from this type of evaluation. Firstly, we can see that for each of Messi, Neymar, and Suarez, their Goal to xG ratios are well above 1, indicating that each player is more efficient with their chances than the average player in our model. To those familiar with the 2014/15 Barcelona team, this should be no surprise, as Messi, Neymar, and Suarez were heralded as some of the best attacking players in the world. Secondly, we can see that even compared to Neymar and Suarez, Messi has a high Goals to xG ratio. This means that not only was Messi scoring goals at a superior rate, but he was also more efficient with his chances than his world-class teammates. While this example provides one instance of how xG can be used to evaluate a player, in reality there are other types of analysis and realms of application than this one. For instance, soccer clubs are increasingly relying on xG as they scout players and consider who to recruit to their teams, making models like ours a part of the operations of some of the most recognizable clubs and brands in the world.

Recap

Overview of Results

We began by questioning how we could quantify the probability of a goal in soccer given a shot. From there, we proceeded to gather soccer data from StatsBomb, ultimately training and testing our model on a set of nearly twenty-thousand shots, including nearly 60 given and engineered features. Models were selected that could provide us with the probability of a given shot resulting in a goal, and included a logistic regression model, penalized logistic regression models, random forest models, and an XGBoost model. For each model, we calculated xG for each shot in our test dataset. Using the calculated xG value in conjunction with the result of each shot, we evaluated the performance of each model on the basis of metrics like AUC, balanced accuracy, and the sum of xG. A few of our models performed quite well, notably the logistic regression models and the XGBoost model, which produced overall results comparable to that of xG calculated by StatsBomb. Beyond fitting and evaluating the models, we produced a few examples demonstrating the usefulness of the xG statistic in tactical and player-specific evaluation.

Limits of study and possible areas of future exploration

While our models prove useful in calculating the xG for a given shot, there are certainly limitations and weaknesses within our models. Firstly, while the calculated xG may appear to be a quality metric, the same models produce classification results that are imperfect, with each struggling to successfully label goals. While this is a limitation of which to be aware, it is thankfully not negative in effect, as ex-post classification of shots is largely not helpful—we know whether recorded shots resulted in goals or not. Secondly, our model does not necessarily provide intuitive xG for each individual shot in our dataset, as our data does not fully encapsulate some complexities that can be recognized by an observer. Remedying this problem likely looks like incorporating additional predictors into our model, such as the nature of the pass or action preceding the shot, or even the weather at the moment of the shot.

References

https://statsbomb.com/wp-content/uploads/2021/11/Working-with-R.pdf

https://soccermatics.readthedocs.io/en/latest/lesson2/introducingExpectedGoals.html

https://theanalyst.com/na/2023/08/what-is-expected-goals-xg/

https://github.com/statsbomb/StatsBombR/blob/master/R/freezeframeinfo.R

https://xgboost.readthedocs.io/en/stable/parameter.html