Predicting goal probabilities for possessions in football

Research Paper Business Analytics

Nils Mackay

Vrije Universiteit Amsterdam

Abstract

Football analytics has been on the rise, yet no previous efforts have been made to compute the probability a possession becomes a goal, like in basketball. In this paper goal probabilities will be modeled for the English Premier League for the season 2016/2017. Ridged logistic regression with a sliding window approach will be used to model the sequential information in possession chains. Several sizes of sliding windows will be used, as well as recurrent sliding windows. The performance of the models will then be tested using the AUC from the ROC Curve, the Log Loss, and the RMSE. These metrics will then be compared with benchmark models to see if a significant improvement has been made. Finally, a top 15 of players will be deducted from the results, showing the applicability of possession probability models to football clubs.

Contents

1 Introduction

Football is the biggest sport in the world with almost 4 billion people following the sport. It is therefore no surprise that the amount of money being spent in football is still on the rise. For instance, in the Premier League, one of the biggest football leagues in the world, each club receives 84.4 million pounds from the sale of TV rights only. The importance of success has thus become not only emotional, but also financial. Football clubs embrace every possibility that could improve the performance of their first team.

Following the surge for analytics in baseball, football has now begun to follow suit. The value of mathematical modeling has become clear, and the search for models that could improve the performance of the club has started.

This paper will cover the mathematical modeling of goal probabilities for possessions. The main goal is to test whether it is possible to model this more accurately than a base line benchmark. Furthermore, the difference between models with and without sequential variables will be compared, to see if adding sequential information will improve model performance.

2 Background & literature research

Analysts have been using mathematical models in sports for a long time. Charles Reep has been credited to be the first person to apply mathematics in football[1]. Around 1950, he used mathematics in order to try to improve the probability of his team winning a game. As a pioneer, he is believed to have a great impact on football in British and Norwegian football. Back then, the mathematics used were simple and done by hand. Advances in mathematics and computer science now makes it possible for sports teams to use data science in order to analyze the sport. This can be used to gain edges in a competitive world where every improvement can make a world of difference.

Where football has been slow to adapt to mathematical advances, other sports have been much faster to adapt new methodologies in their analyses. Baseball has always been at the forefront of mathematical analysis, mostly due to the static nature of the sport making it easy to analyze. The success story of the Oakland A's surrounding data analysis has even been captured in the movie Moneyball, after the book by Michael Lewis published in 2003.

Where a static sport like baseball can easily be analyzed with data analysis, more fluent and dynamic sports like basketball, ice hockey, and football, face more difficulties. Not many papers are focused around sports analytics, and therefore a historical track of progress is mostly lacking. Public analysts fill this gap somewhat by writing about progress in an open manner, even though these 'blogs' tend to be of a less official nature.

Public research into goal probabilities first arose in ice hockey in 2004, when several public analysts wrote about differences in shot quality. Alan Ryder noticed that the probability of a shot being converted was highly dependent on the distance from where the shot was taken.[2] Individual statements of factors influencing goal probabilities later changed into models, predicting the goal probability of a shot giving its pre-shot characteristics.

Similar models were soon adapted to football, and models determining the probability of a shot being converted were called 'Expected Goal' (or xG short) models. The first to introduce the concept of xG in football was Sam Green in 2012.[3] Since this introduction, the metric has been widely adopted in football analysis, and is one of the most used metrics in public analysis.

While xG is only based on shots, the search for models calculating goal probabilities for other actions was less active. In basketball, however, these models became easier to build with the introduction of tracking data. In 2014, Cervone et al. released a paper on EPV, or Expected Possession Value, at the Sloan Sports Analytics

Conference.[4] This was the first attempt at modeling possession goal probabilities that became widely known. This method has since been adapted by many analysts to evaluate decision-making in basketball.

However, the lack of public availability to tracking data in football has made the progress into a similar metric in football analytics difficult and slow. Alternatives have been opted, like the model by Neil Charles which he introduced at the 2017 OptaPro Forum. He constructed a model to evaluate decisions in a similar way to EPV in football.[5] This, however, still requires either tracking data or manual data entry for every individual situation.

In this paper, I propose a way of modeling goal probabilities in football without tracking data, but with the more publicly available event data from Opta.

3 Data

The data used in this research is F24 Opta event data. Opta is a company that records football data from match footage by manual labor. Every event (e.g., pass, shot, etc.) on the pitch is recorded with additional variables describing each event. The data is stored in XML files. One XML file contains data for one match. For this research, I will be using data from the last 5 seasons of the English Premier League. A season consists of 380 matches, which makes the total 1,900 XML files.

An example of a part of such an XML file is the following:

```
...
<filters>
    <goal_keeping>
    <goals_attempts>
    <headed_duels>
    <interceptions>
    ...
    <all_passes>
        <event type = "completed" player_id = "81" team_id = "34" mins = "8" secs = "13">
            <start> 56.5, 47.1 </start>
            <end> 75.2, 60.4 </end>
            <through_ball> true </through_ball>
        </event>
        <event>
            ...
        </event>
    </all_passes>
    ...
</filters>
...
```

Fig. 1. Example of part of XML file data structure.

As can be seen, every event in the match has a separate element that starts with <event> and ends with </event>. Which attributes are given within an element depends on the type of event.

3.1 Data pre-processing

Data transformation
To make the data usable for this research, it needs to be converted to a data frame. Every event becomes a single row, with columns describing the attributes of the event. Since some events have attributes that others do not, some attributes are not included in the data frame, while other attributes may be empty for certain events. The following variables can be directly extracted from the data:

Variable	Description
mins	The minute of the match in which the event took place.
secs	The second of the minute in which the event took place.
player_id	The id of the player who executed the event.
team_id	The id of the team in which that player plays.
opponent_id	The id of the opposing team.
home	The id of the team that plays at home.
away	The id of the team that plays away.
type	The type of the event (e.g., pass/shot/long ball).
headed	1 if the event is executed with the head, 0 otherwise.
x_start	The x-coordinate of where the event started.
y_start	The y-coordinate of where the event started.
x_end	The x-coordinate of where the event ended.
y_end	The y-coordinate of where the event ended.
completed	1 if the event is completed, 0 if it failed.

Table 1. Variable description.

Data creation

Opta event data only includes events that took place. This means that if a pass was completed from A to B, the next event could be a pass from C to D from the same team. In such a case, a player most probably dribbled from B to C, however, since this is not an 'action', no event is recorded. This situation is shown in Figure 2.

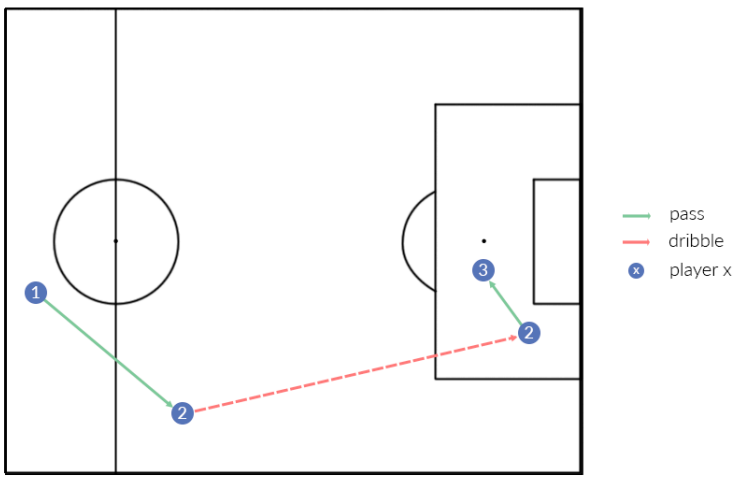

Fig. 2. Example of a possession containing a gap due to a dribble.

It is, however, possible to create these 'events' by extrapolating the data. This way we can ensure that there are no gaps in the possession chains. A dribble is added in the data if the following criteria are met:

- The end location of the first event != the start location of the following event.
- The team that executes the first event, also executes the second event.
- The first event is a completed event.
- The first event is of a passing type, i.e., the event type is from {pass, long_ball, through_ball, headed_pass, throw_in}, or a takeon event.
- The second event is an event that could take place after a dribble, i.e., the event type is from {pass, long_ball, through_ball, shot, takeon, cross}.
- The speed of the dribble is at most 10 m/s.

In total, this means that roughly 150,000 dribbles per season are added, bringing the total number of rows per season to roughly 600,000.

Dribbles that follow a takeon type event are defined as a 'takeondribble' type, since it is expected that these types of dribbles are significantly different from normal dribbles.

For both dribble and takeondribble type events, the remaining columns are filled as follows. For mins and secs the average between the preceding and the following event is taken. The player_id is the id of the player who made the dribble, in this case assumed to be the player who executed the following event. The team_id, opponent_id, home and away variables are the same as those from the preceding/following event. The headed variable is set to 0, as you usually do not dribble with your head. The start location is the end location of the preceding pass. The end location is the start location of the following pass. Finally, the completed variable is set to 1. Incomplete dribbles are not recorded.

3.2 Feature engineering

In this section, some extra variables are created. Some are created to simplify future programming, whereas some variables are added that are expected to influence the probability of a possession ending with a goal. Also, the dependent variable is created and explained.

Variable	Description
game_id	A unique id for every game. Used for easier data manipulation.
possession_id	A unique id for every possession. A new id is chosen when possession changes team. Used for easier data manipulation.
event_id	A unique id for every event. Used for easier data manipulation.
goal	The dependent variable. 1 if the current possession ends with a shot

	that was a goal, 0 otherwise.
time	The time between the execution of this event and the next event.
distance	The distance between the start location of this event and the end location.
speed	Distance (in meters) divided by time (in seconds).

Table 2. Feature engineering variable description.

Apart from these variables, one more variable will be modeled. Previous research into goal probabilities from shots have shown that the location of a shot has a great influence on its goal probability.[6] Given that a player can decide to shoot at any time during a possession, it follows that the location of a possession is most likely also of great importance to its goal probability. To accurately include this factor, one can choose to estimate the probability of a goal when the ball was at a certain location on the field.

One way to do this is to split the field into many small two-dimensional bins, and calculate the observed probability per bin. This gives the following image:

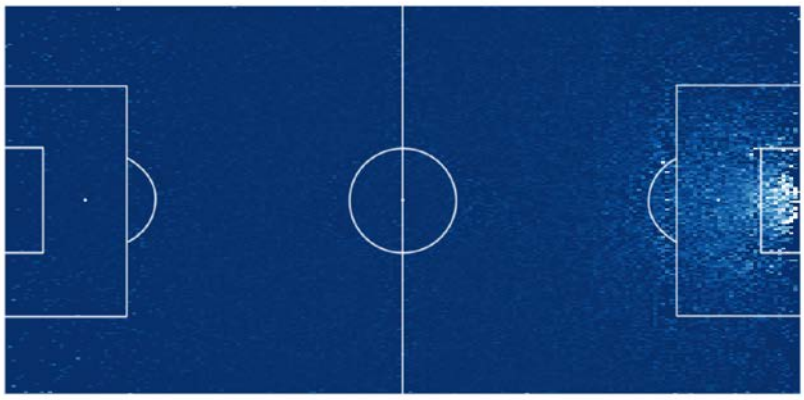

Fig. 3. Observed goal probabilities for possessions. The whiter, the bigger chance of a goal. The team in possession attacks to the right.

Due to sample size issues in a lot of the bins, the probabilities vary greatly from one bin to the next. Some smoothing is needed.

To find the relation between location on the field and goal probability, a GAM (General Additive Model) is fitted using a tensor product smooth. A tensor product smooth is used since the units of the x-coordinate and the y-coordinate are not the same. Both run from 0 to 100, but since a football field is not square the distance covered by a change of 1 in the x-coordinate is not the same as the distance covered by a change of 1 in the y-coordinate.

In a tensor product smooth, the probability will be estimated by the following formula:

$$= {}_1 + {}_2 + {}_3 {}_1( ) + + {}_{(-2)}( ) + \\ {}_1 + {}_2 + {}_3 {}_1( ) + + {}_{(-2)}( ) + ,$$

where the functions f are simple functions on x and y respectively. Since this can be seen as a tensor product, it is possible to rewrite this in order to find the least squares estimators of the coefficients .

After fitting the model for and equal to 15 (due to computational constraints), we get the following smoothed space:

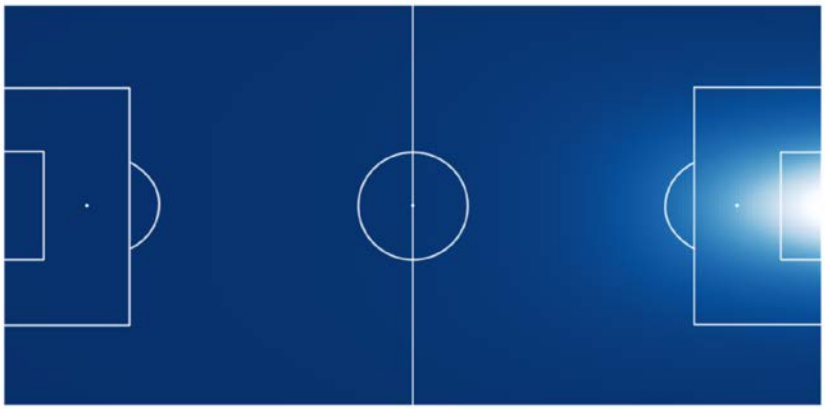

Fig. 4. Smoothed goal probabilities for possessions. The whiter, the bigger chance of a goal. The team in possession attacks to the right.

This smoothed space can be used to predict goal probabilities by location. This will be used later on in the model, and such predictions of location based goal probabilities will be referred to as the variable locationxG.

How all these extra variables will be used in the models will be explained in the next chapter.

4 Modeling

4.1 Model choice

The model that will be used is a ridged logistic regression. A logistic regression is a perfect fit for this problem, due to the fact that the data is very unbalanced. About 1.2% of all possessions has a positive response variable (a goal). This means that a lot of machine learning methods can be eliminated due to poor performance for unbalanced data sets, such as random forests.

Another characteristic of a logistic regression model, is that predictions from such a model can easily be explained and derived from the models results. This means that it will be clear 'why' the model will have predicted a certain score.

Ridged logistic regression

A logistic regression is a generalized linear model. A generalized linear model allows the distribution of the response variable to be different than usually. The basic structure of a generalized linear model is:

$$(\ ) = \quad ,$$

where g denotes the link function. For a logistic regression, we have a binomial link function, given by:

$$(\ ) = \log \frac{}{1 -} .$$

The log-likelihood of for a generalized linear model is given by:

$$(\ ) = \sum_{=1} \frac{[\quad - \quad (\ )]}{(\ )} + \quad (\ ,\ ).$$

The optimal is the one that maximizes this function. The outcome of such a model is a value between 0 and 1, denoting the probability of a positive response variable given the explanatory variables. In a 'ridged' logistic regression, we adjust the maximization function as follows:

$$(\ ) = \ (\ ) - \ \| \ \|^2,$$

such that if we choose λ equal to zero, we will get a regular logistic regression, and such that if we choose λ very big the coefficients will go towards zero. This

addition usually increases the mean-squared error of the fit on the training data, but improves the predictive value and thus decreases the mean-squared error in out-of-sample predictions. The actual value of λ will be chosen by the algorithm used to fit the ridged logistic regression.

The values of the coefficients are dependent on the variance of the corresponding variables. Therefore, if we want to penalize all coefficient values fairly, we have to standardize the variables to their variance. An additional advantage this gives is that the absolute values of the coefficients can be used to measure feature importance on model prediction.

This method is a good fit for the problem, since for different models different number of variables will be used. It is unknown if all these variables will have a positive effect on predictability. Using a ridged regression will ensure that insignificant variables will get a low coefficient, negating their influence on the prediction of the model. This should make sure the model does not overfit.

Sliding window

To test whether sequential information will improve the quality of the model, a sliding window approach will be used. In a regular sliding window method, the value of is predicted by using the variables , where is the size of the sliding window.[7]

Since the problem at hand does not allow for use of variables that occurred after the event ( ), the sliding window will go backwards only. This method will be applied to this problem for $= \{0,1,2,3\}$.

A way in which a sliding window approach can be improved is by making them recurrent. This means that in addition to the variables used in a regular sliding window approach, also the values of will be used to predict . Since the true values of these variables are not known, the prediction from the model will be used.

This second method does bring with it some problems regarding training. What values of should be used? One approach opted by T.G. Dietterich[7] involved first predicting using a non-recurrent classifier, and use that as input. Using iteration, we can then use the output of the fitted model as input again, and repeat this process until the changes in predictions are sufficiently small. This is the method used in this paper. The iterations are stopped when the maximum change of all predictions is smaller than 0.01 in comparison to the previous iteration. For out-of-sample prediction, iteration is also used to get a final prediction. This method will be applied for $= \{1,2,3\}$.

4.2 Variables

The response variable ‘goal’ is given a 1 when the same possession ends in a goal, and a 0 otherwise. Four different explanatory variables will be used:

Variable	Description
locationxG	As described in Section 3.2.
type	As described in Section 3.1.
headed	1 if the current action ends on the head of a teammate, 0 otherwise.
speed	As described in Section 3.2.
direct_speed	The speed of the direct distance covered. Example below.

Table 3. Variables used in the models.

When using sliding windows, these variables will also be used from previous actions. To avoid confusion, the value of the corresponding in , – will be appended to the variable name. When recurrent sliding windows are used, the value of , – will be denoted by the variable name xG, appended by the value of . To illustrate this, an example is given below:

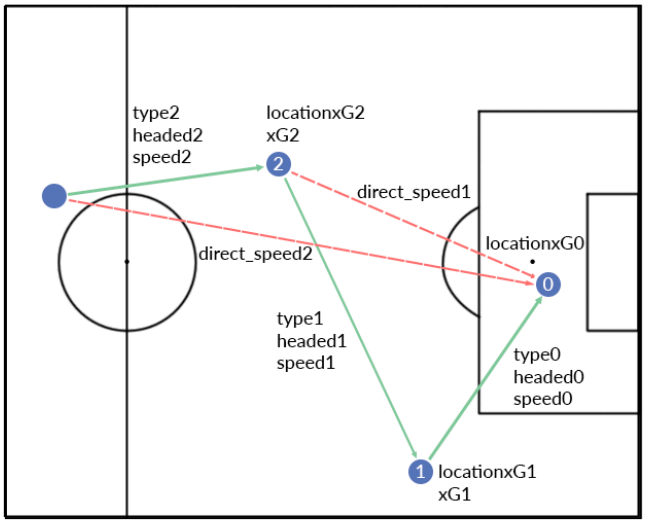

Fig. 5. An illustration of the variables corresponding to different time steps.

In the above example, the variables corresponding to different actions are given. For instance, the variables type0, headed0, and speed0 correspond to the final action. As you can see, the variables direct_speed1 and direct_speed2 correspond to the distance

between the final location and the second-to-last and third-to-last location, respectively.

4.3 Evaluation method

To evaluate the predictions of the different models on the test set, three evaluation metrics will be used.

RMSE

RMSE or Root Mean Squared Error is an evaluation metric used to assess the quality of probability estimation. It is given by:

$$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2},$$

where y_i is the outcome, which is 1 or 0, and $\hat{y}_i$ is the predicted probability. The RMSE is a good fit for this problem, since we want to measure the actual accuracy of the probability estimations.

Log Loss

Log Loss is a widely used evaluation metric to measure the performance of a classification model. In our specific case, if we have predicted probability $\hat{y}_i$ and outcome y_i of a certain row i, the Log Loss of that row is given by:

$$L_i = -y_i \log \hat{y}_i - (1 - y_i) \log(1 - \hat{y}_i).$$

Generalizing this for N rows, we get a total Log loss of:

$$L = \frac{1}{N}\sum_{i=1}^{N} L_i.$$

Log Loss thus heavily penalizes high probability estimations for which the outcome was negative, or low probability estimations for which the outcome is positive. In other words, the metric measures the amount of surprise between a prediction and the actual outcome. The bigger surprise, the worse. A good model would minimize the Log Loss, as that would mean surprises were kept to a minimum.

AUC

The AUC or Area Under Curve is an evaluation metric coming from the ROC, or Receiver Operating Characteristic. The ROC is a curvature described by the ‘false positive rate’ on the x-axis, and the ‘true positive rate’ on the y-axis. It calculates

points on this plane by using different thresholds, above which all instances with a probability estimation of at least that threshold is taken as a positive prediction. Similarly, all instances with a lower probability estimation than the threshold is taken as a negative prediction. By going through different threshold values, a curvature between the coordinates [1,0] and [0,1] is created.

The AUC is the area under this ROC curve. For a random prediction, this AUC has a value of 0.5. The optimal predictor will have a value of 1.0.

The AUC can be used as a metric to evaluate the ability of the predictions to distinguish between positive and negative instances. There are some issues with using the AUC regarding this problem though. First of all, the AUC does not measure the accuracy of the probability estimations. It merely looks at the relative difference between different instances, regardless of the actual values. Therefore, a model that estimates probabilities much better, might only see a slight increase in AUC. Furthermore, as Davis & Goadrich[8] noted, in unbalanced data the AUC might not be the best metric to use.

Significance

In this paper, these three metrics will be used to evaluate the quality of different models. Two benchmark models will be introduced. The first benchmark represents the minimum performance that has to be achieved. This benchmark will be the values of locationxG0, taken from the smoothed surface introduced in Figure 4. To check if goal probabilities can be accurately measured, a bootstrap between the scores of the best model and the benchmark will be used. If the difference is significant, the conclusion will be that it is possible to model goal probabilities.

To get an idea of how much better the model can become, another benchmark will be introduced. This benchmark will consist of the predicted probabilities, but different outcomes than the observed outcomes. As outcomes, Bernoulli trials on the predicted probabilities will be used. This way, we can say that the probabilities are the true probabilities. The average of 1000 simulations will be used as the model's evaluation score. This will give an indication as to how good the model's performance scores are compared to a 'perfect' model.

Finally, the best model will be compared to the model without sequential information. The difference in scores will be bootstrapped to check for significance. When the difference is significant, it can be concluded that including sequential information significantly increases model performance.

4.4 Measuring player performance

To try to capture player performance, the following method will be used. For every action, the difference between the , at the moment the player receives the ball and the , after the player has had the ball. The idea behind this is, that a good player

will be able to progress the ball to a more dangerous situation regularly. Or, put the other way around, when a player is able to systematically progress the ball into a more dangerous situation, he presumably is a good player. To check whether this method yields any useful results, the top 15 players will be evaluated by looking at their market value according to Transfermarkt.com. These values will be compared to the league average.

The expectation is that the 15 will consist of more expensive players than the average. This is not a perfect way to measure this, as strikers are usually the most expensive, but not known for progressing the ball. These strikers will likely fall outside the top 15, and thus negatively influence the results somewhat. The expectation is that the best midfielders will come out on top of the list, as their job is to progress the ball to a more dangerous situation.

5 Results

5.1 Model results

The models are denoted as Lr_ , where is the value of the size of the sliding window . If the recurrent sliding window is used, an is appended to the model name.

First, the Lr_1 is fitted. This gives an optimal λ of 0.01. Due to computational constraints, this same value for λ was chosen for the other models. This might decrease their performance somewhat, but finding an optimal λ for all models was infeasible due to computational and time constraints.

For the recurrent models, iteration was used until predictions were stable. In practice, this convergence occurred very quickly. The convergence of the maximum change of a single prediction between iterations is shown for all recurrent models:

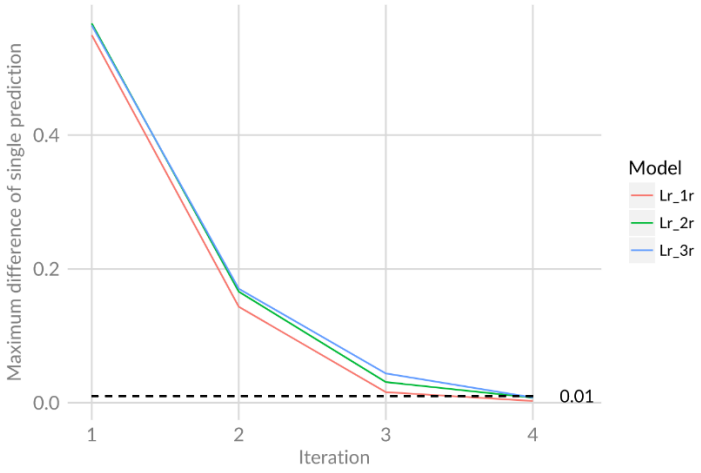

Fig. 6. Convergence of the recurrent models.

To get an idea of variable coefficients, the coefficients of the Lr_3r model are shown below. These coefficients are de-standardized, and are thus applicable to the variables in their original state:

	Coefficients			
Variable	= 0	= 1	= 2	= 3
Intercept	-4.722	-	-	-
locationxG	14.524	9.678	3.416	1.481
headed	-0.229	-0.111	-0.087	-0.027

speed	0.004	-0.003	-0.004	-0.005
direct_speed	-	0.011	0.003	0.003
xG (recurrent)	-	-1.369	0.785	0.649
type: clearance	-	-0.017	-0.027	-0.029
type: corner	-	0.399	0.279	0.215
type: cross	0.059	-0.184	-0.257	-0.086
type: dribble	0.042	0.049	0.059	0.055
type: foul	-	-0.025	-0.107	-0.128
type: headed_duel	-	-0.081	-0.081	-0.041
type: headed_pass	-0.112	-0.100	-0.053	-0.068
type: interception	-	0.085	0.061	0.049
type: long_ball	-0.075	-0.024	-0.025	-0.021
type: pass	-0.031	-0.022	-0.010	0.012
type: shot	-	-0.172	-0.071	-0.025
type: tackle	-	-0.033	-0.087	-0.072
type: takeon	-0.227	0.017	-0.003	-0.036
type: takeondribble	0.428	0.209	0.184	0.198
type: through_ball	1.745	1.383	0.717	0.408
type: throw_in	-0.158	-0.154	-0.129	-0.110

Table 4. Coefficients for all variables for the Lr_3r model.

These coefficients vary for each model, but are in general very close to one another. It is also interesting to see that the coefficients for each variable are relatively stable across different values for . For instance, only 4 out of 22 variables have a change of sign across all values for . Furthermore, it looks like variables with a relatively high coefficient for $= 0$ or $= 1$, have their coefficients weakened as increases. This makes sense, as the further back we go the less influence from these variables we expect. Examples of this in the above table include but are not limited to: locationxG, headed, and the type factors: corner, shot, takeondribble, and through_ball.

To get an idea of feature importance, we can look at the absolute values of the standardized coefficients. Since all variables are standardized to their respective variances, the coefficients describe the influence of each variable on the predictions. The top ten most influential variables are shown below:

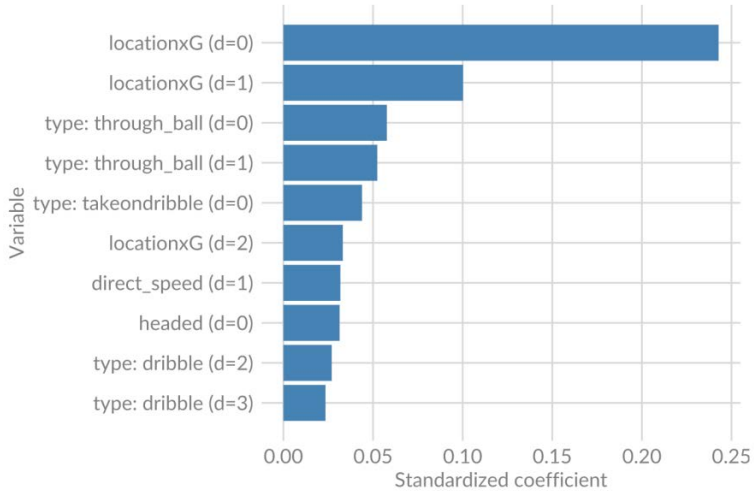

Fig. 7. Feature importance in the Lr_3r model.

The end location of the final pass (locationxG (= 0)), is by far the most importance variable. In general, the locations of the passes are amongst the most influential variables. Other important variables include the type, when the factor is either a through_ball, takeondribble, or a dribble. Finally, variables from more recent events (with a low), seem to have a higher influence than events from longer ago. The complete table with all standardized coefficients can be found in the Appendix.

5.2 Performance evaluation

All models are used to predict the response variable for the test set. The quality of these predictions is tested using the AUC from the ROC curve, as well as the RMSE and the Log Loss between the predictions and the outcomes. This gave the following results:

Model	AUC	RMSE	Log loss
Benchmark 1: location	0.6735	0.1226311	0.07546543
Benchmark 2: perfect	0.6163	0.1202387	0.07437248
Lr_0: no sequential info	0.6651	0.1222823	0.07541043
Lr_1	0.6750	0.1221798	0.07531453
Lr_2	0.6730	0.1221382	0.07525101
Lr_3	0.6725	0.1221305	0.07523350
Lr_1r	0.6744	0.1221737	0.07530323
Lr_2r	0.6729	0.1221141	0.07522148
Lr_3r	0.6725	0.1221073	0.07520444

Table 5. Performance measures for all models.

The results are very surprising. Where the scores for the RMSE and Log Loss follow intuition, and agree on model quality, the AUC results are very different. Not only does Benchmark 1 have one of the best scores, the simpler models seem to give higher scores than the more complex models. While these are somewhat unexpected, especially the score for the perfect model, which is by far the lowest, casts some doubt on the usability of the AUC for model selection in this paper. Similar contradicting results seem unprecedented in literature. The reasons for these contradicting results remain unclear, and could be a starting point for further research. For now, however, the AUC scores will be ignored.

For the RMSE and Log Loss, model performance seems to improve with model complexity. Bigger sliding windows yield better results, and recurrent sliding windows outperform non-recurrent sliding windows. The best model in both measures is the Lr_3r model, the most complex model. This might indicate that including sequential information from a bigger sliding window will improve performance even more. The Lr_3r model will be compared to Benchmark 1 to check if it is possible to improve goal probability estimation from standard methods. Furthermore, the Lr_3r model will be compared to the Lr_0 model to check if adding sequential information will significantly improve model performance.

To check whether the obtained differences in scores are significant, bootstrapping methods ($= 1{,}000$) are applied. First, we see the comparison of the Lr_3r model to the Benchmark 1 model. Positive values mean the model outperformed the benchmark:

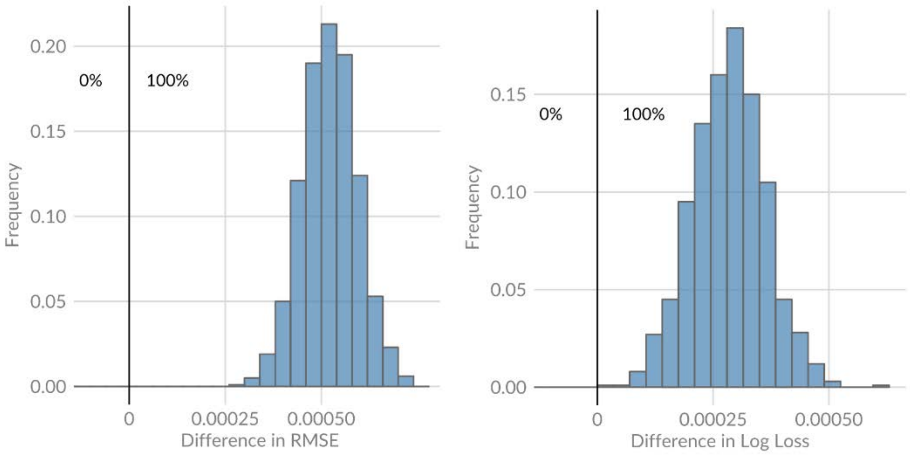

Fig. 8. Bootstrapped difference in RMSE and Log Loss between Lr_3r and Benchmark 1.

As can be seen, in both performance measures, the Lr_3r model significantly outperforms the Benchmark 1 model. Indeed, in every single instance the Lr_3r

model outperformed the benchmark, which is indicated by the percentages in the graph. This indicates that it is indeed possible to model goal probabilities better than a simple benchmark model.

Next, we see the comparison between the Lr_3r model and the Lr_0 model. Positive values mean the Lr_3r outperformed the Lr_0 model:

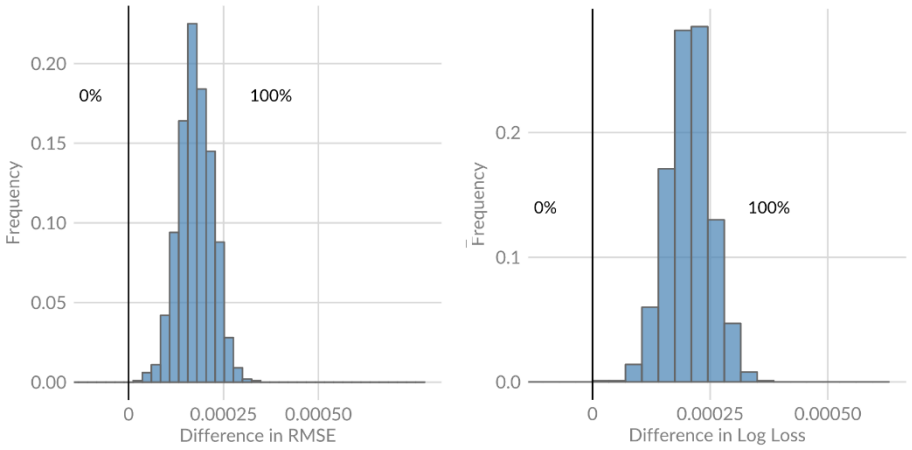

Fig. 9. Bootstrapped difference in RMSE and Log Loss between Lr_3r and Lr_0.

As can be seen, in both performance measures the Lr_3r model significantly outperforms the Lr_0 model. Just like in the previous test, in every single instance this is the case. Even though the differences are smaller than in the previous test, so is the variation in the differences. We can conclude that sequential information does indeed improve the model quality.

5.3 Results from player performance

Finally, we take the Lr_3r model, since it performs best, to measure player performance. For every player, we look at the difference between goal probabilities for every action he made. The sum of all these differences will then give an indication of the total amount of danger that player created. Since some players had more playing time than other players, the total is given as an average per 90 minutes, the length of a single football match. Players that had fewer than 1,350 minutes (15 full games) of playing time are not taken into account, to avoid players with a limited sample size. This leaves a total of roughly 260 players, or about 13 per team. The top 15 is given below:

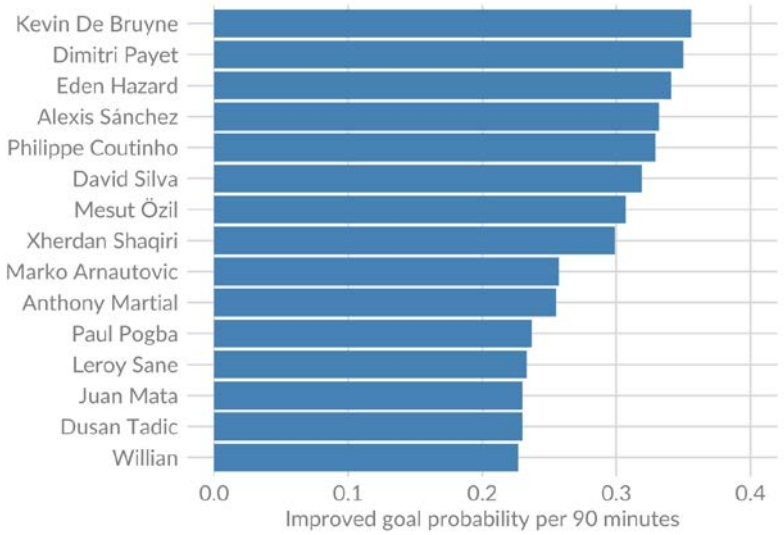

Fig. 10. The top 15 players of the English Premier League 16/17, in improving the probability of his team's possessions.

All of these players are players whom are generally praised by the public. To get a somewhat objective evaluation of these players, their market value is compared to the average market value of the other players that were considered. Market values are taken from Transfermarkt.com.

The average market value of the top 15 is roughly 40 million euros, whereas the other players have an average market value of roughly 15 million. It must be noted that this is not a very good way to assess player quality. Other variables, like player age, position, and more, also have great influence on market value. For this purpose, however, it can be used to give a general idea of the possible value of the model in this paper.

6 Conclusion & Discussion

6.1 Conclusion

Using a ridged logistic regression, it is possible to model goal probabilities in football above a standard base line. Furthermore, including sequential information in the model by using a sliding window approach, will improve the model quality significantly. While the optimal size of the sliding window in this research turned out to be $= 3$, it is possible that a bigger sliding window will improve performance even more. Adding a recurrent variable improved model performance in all cases, but these differences are possibly not significant. The added modeling complexity might not be worth the additional performance, although this is dependent on the purpose of the model and the time available.

Even though it is indeed possible to improve model quality by using sequential information, the location of the ball at $= 0$ is by far the most important factor in determining the goal probability. Other important variables include but are not limited to the previous locations of the ball, and whether a through ball has been played recently.

Using the best model to assess player performance in the Premier League yields promising results. The players that are rated highly turn out to be generally more expensive players. The same technique can be used on other leagues, where players might be lesser known. This method then offers a way to objectively rate players without having to follow that specific league closely.

6.2 Discussion

Theoretically possible improvement
Even though we have found it is possible to outperform standard base lines, there is still much improvement to be made. This can be seen when looking at the difference between the theoretically perfect model scores, and the Lr_3r model scores. Part of this gap might be able to be closed by improving modeling techniques. However, it is very likely that without using tracking data a significant gap is going to persist. This is because without tracking data, a lot of important information, like defender positioning, will be missing. Without this extra information, improvement will probably be limited.

Another reason why there is a cap on performance, is that the game of football is inherently random. We will never be able to predict at the start of a possession, whether that possession will be converted to a goal. This is because whether the possession ends successfully, is dependent on events that are unknown at the start of a possession. The occurrence of these events will be paired with an increase or decrease in goal probability, but beforehand this information will not be available/predictable.

While this caps model performance, this is not a problem for the practical use of the model. In the proposed use of the model, for instance, the fact that certain events change the goal probability is needed to assign scores to players. If we already knew if a possession would become a goal at the start of a possession, this would not be possible.

Practical improvement possibilities

There were also some issues in modeling. The most notable is the non-intuitive results of the AUC metric. Whether this has to do with the models, or with the metric itself, is unknown. Since the AUC is one of the most generally used evaluation metrics for classification problems, it is interesting to see these unexpected results.

Improvement in the modeling process is possible in a few ways. First of all, a bigger sliding window can be used to possibly increase model performance. Secondly, the value of λ can be estimated for all models individually. Even though this is not done in this research due to computational constraints, improving the value of λ may increase performance as well. Finally, other machine learning techniques may be used to try to increase performance. It must be noted that whatever technique is used, it should be able to handle imbalanced data properly.

The use of the model for the purpose of evaluating players can be significantly improved upon. For instance, negative scores can be counted for every time a player loses the ball. Also, the value of a 'negative' pass can be capped, to avoid penalizing players who are forced to pass backwards due to a lack of support. Many more improvements are likely possible, but since that is not the aim of the paper this will not be visited in depth.

Applications

Even though this paper is focused around football, the methods used can also be applied to other areas of research. The most obvious application is the use of similar methodology in other sports, like ice hockey and rugby. Any sport which consists of possessions of a ball could be applicable. Outside of sports, these methods can also be interesting. Think of a manufacturing line where it can be used to predict faulty products. Basically, any situation where you want to predict success in a system with consecutive actions can make use of this methodology. It is, however, not necessarily true that results from this paper will also translate to different scenarios.

7 References

1. Larson, O. (2001). Charles Reep: A major influence on British and Norwegian football. *Soccer & Society*, 2(3), 58-78.
2. Ryder, A. (2004, January). Shot Quality. Retrieved June 25, 2017, from http://hockeyanalytics.com/Research_files/Shot_Quality.pdf
3. Green, S. (2012, April 12). Assessing the Performance of Premier Leauge Goalscorers. *OptaPro Blog*, Retrieved June 25, 2017, from http://www.optasportspro.com/about/optapro-blog/posts/2012/blog-assessing-the-performance-of-premier-league-goalscorers/
4. Cervone, D., D'Amour, A., Bornn, L., & Goldsberry, K. (2014, February). POINTWISE: Predicting points and valuing decisions in real time with NBA optical tracking data. In *8th Annual MIT sloan sports analytics conference, February* (Vol. 28).
5. Charles, N. (2017, Febraury). Shoot! Optimizing the Location of Attempts on Goal. In *2017 OptaPro Analytics Forum*. Retrieved June 25, 2017, from http://www.optasportspro.com/media/936777/Neil-Charles-Website.pdf
6. Caley, M. (2014, September 11). Premier League Projections. Retrieved June 25, 2017, from https://cartilagefreecaptain.sbnation.com/2014/9/11/6131661/premier-league-projections-2014#methoderology
7. Dietterich, T. (2002). Machine learning for sequential data: A review. *Structural, syntactic, and statistical pattern recognition*, 227-246.
8. Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd international conference on Machine learning* (pp. 233-240). ACM.

8 Appendix

8.1 Feature importance

	Absolute standardized coefficients			
Variable	= 0	= 1	= 2	= 3
locationxG	0.2428	0.1002	0.0331	0.0139
headed	0.0313	0.0181	0.0160	0.0046
speed	0.0155	0.0108	0.0160	0.0192
direct_speed	-	0.0318	0.0104	0.0116
xG (recurrent)	-	0.0199	0.0086	0.0065
type: clearance	-	0.0026	0.0043	0.0041
type: corner	-	0.0112	0.0071	0.0042
type: cross	0.0050	0.0133	0.0224	0.0071
type: dribble	0.0203	0.0229	0.0270	0.0235
type: foul	-	0.0022	0.0095	0.0107
type: headed_duel	-	0.0068	0.0070	0.0032
type: headed_pass	0.0182	0.0186	0.0093	0.0109
type: interception	-	0.0102	0.0072	0.0052
type: long_ball	0.0150	0.0053	0.0052	0.0042
type: pass	0.0156	0.0111	0.0050	0.0058
type: shot	-	0.0162	0.0067	0.0023
type: tackle	-	0.0031	0.0090	0.0069
type: takeon	0.0130	0.0016	0.0002	0.0030
type: takeondribble	0.0439	0.0171	0.0137	0.0127
type: through_ball	0.0577	0.0524	0.0227	0.0125
type: throw_in	0.0188	0.0182	0.0143	0.0114

Table 6. Absolute standardized coefficients of the Lr_3r model, indicating the feature importance.