Advanced Football Metrics

Dávid Varga, Márton Németh

Institute of Informatics, Information Systems and Software Engineering
Faculty of Informatics and Information Technologies
Slovak University of Technology in Bratislava
Knowledge Discovery

May 24, 2021

1 Introduction

In this project we examine two aspects of football games. First, we study how footedness affects the player's performance. In the second part, we create a custom model, similar to the popular xG model, that can be used to predict how dangerous a shot is.

2 Dataset

Our data is from understat.com which is one of the biggest freely accessible websites that provides advanced statistical data for football matches in all five major football leagues in Europe (EPL - England, LaLiga - Spain, Bundesliga - Germany, Serie A - Italy, Ligue 1 - France). We chose this site, because, in addition to basic metrics, such as shots on target, corners, faults, etc., it provides more detailed data, like the xG metric.

2.1 Scraper

Using web crawling we managed to extract 3 types of data from the site:

- matches ($/data/games$)
- players ($/data/players$)
- shots ($/data/shots$)

For every data type, we created a simple web crawler in R (available in /crawler) that loops through every match in a given season and saves the extracted data as .csv files, hence our data is divided into multiple files based on the league, the data type, and the season. We use the following naming convention: <`league`>_<`type`>_<`season`>, eg. *Bundesliga_shots_*2019.

2.2 Shots

The number of observations is around 8000-9000/league/season. One row represents one shot that is characterised by the following variables:

- 'xG': 'expected goals' - metric computed by Understat, measures the quality of a shot (numerical) <0; 1>
- 'h_goals'/'a_goals': number of goals scored by the home/away team (numerical)
- 'result': result of the shot (categorical, nominal) 'SavedShot', 'MissedShot', 'OwnGoal', 'BlockedShot', 'ShotOnPost', 'Goal'
- 'X'/'Y': X and Y coordinates of the shot, normalized by Understat (numerical) <0; 1>
- 'situation': the situation that preceded the shot (categorical, nominal) 'DirectFreekick', 'FromCorner', 'OpenPlay', 'Penalty', 'SetPiece'
- 'shotType': the type of the shot (categorical, nominal) 'LeftFoot', 'RightFoot', 'Head', 'OtherBodyPart'
- 'h_a': whether the shot came from the home or the away team (categorical, nominal) 'h', 'a'
- 'lastAction': the last action before the shot (categorical, nominal) 'Pass', 'Cross', 'Aerial', 'TakeOn', 'Chipped', 'None'

Further variables:

- 'id', 'player_id', 'match_id', 'date', 'minute', 'season' are self-explanatory
- 'h_team'/'a_team': name of the home/away team
- 'player': name of the player
- 'player_assisted': name of the player that assisted (NA - in case of set pieces)

2.3 Matches

Depending on the number of teams in a league, the dataset for one season in a given league consists of 300-400 observations. Every observation represents one game that has the following variables:

- 'h_goals'/'a_goals': number of goals scored by the home/away team (numerical)
- 'h_shot'/'a_shot': number of shots (numerical)
- 'h_shotOnTarget'/'a_shotOnTarget': number of shots on target (numerical)
- 'h_xg'/'a_xg': sum of every xG in the game (numerical)
- 'h_deep'/'a_deep': metric computed by Understat, counts passes completed within an estimated 20 yards of goal (numerical)
- 'h_ppda'/'a_ppda': 'power of pressure' - metric computed by Understat, counts passes allowed per defensive action in the opposition half (numerical)
- 'h_w'/'h_l': the probability that the home team will win/lose, computed by Understat (numerical)
- 'h_d': the probability of draw, computed by Understat (numerical)

Further variables:

- 'id', 'date', 'league_id', 'season' are self-explanatory
- 'fid': fixture id
- 'h'/'a': id of the home/away team
- 'team_h'/'team_a': name of the home/away team

2.4 Players

The number of observations is around 8000-10000/league/season, one row contains information about one player in one match:

- 'goals': number of goals scored by the player in the match (numerical)
- 'own_goals': number of goals scored by the player in the match (numerical)

- 'shots': number of shots of the player in the match (numerical)
- 'xG': sum of every xG of the player in the match (numerical)
- 'time': minutes played (numerical)
- 'position': the position of the player (categorical, nominal) 'GK' - goalkeeper, 'DR' - right defender, ..., 'Sub' - substitute
- 'h_a': whether the player played for the home or away team (categorical, nominal) 'h', 'a'
- 'yellow_card': number of yellow cards of the player in the match (numerical)
- 'red_card': number of red cards of the player in the match (numerical)
- 'key_passes': number of key passes (final passes leading to shots on target) (numerical)
- 'assists': number of assists (numerical)
- 'xA': 'expected assists' - metric computed by Understat, measures the likelihood that a given pass will become a goal assist (numerical)
- 'xGChain': metric computed by Understat, calculates the expected value of all the team's possessions in which the player participated (numerical)
- 'xGBuildup': metric computed by Understat, calculates all the possession chains in which the playe

r made a successful pass or dribble but didn not take a shot (numerical)

Further variables:

- 'id', 'player_id', 'match_id', 'team_id' are self-explanatory
- 'player': name of the player
- 'positionOrder': the identification number of the position of the player
- 'roster_in': id of the player who is brought on to the pitch (0 if the player was not a substitute)
- 'roster_out': id of the player who has been substituted off (0 if the player started)

3 Exploratory Data Analysis

In this section we are going to analyse the basic attributes of the shots data-set. As we can see, there are several discrete and continuous attributes in this data-set. With command "summary" we can calculate the minimum and maximum, mean and median, 1st and 3rd quartile values of the continuous attributes.

minute	h_goals	a_goals	situation	lastAction	X	Y	shotType	xG	result
Min. : 0.00	Min. :0.000	Min. :0.000	Length:8148	Length:8148	Min. :0.0040	Min. :0.0500	Length:8148	Min. :0.00000	Length:8148
1st Qu.:27.00	1st Qu.:1.000	1st Qu.:1.000	Class :character	Class :character	1st Qu.:0.7870	1st Qu.:0.4180	Class :character	1st Qu.:0.02584	Class :character
Median :49.00	Median :1.000	Median :1.000	Mode :character	Mode :character	Median :0.8680	Median :0.5000	Mode :character	Median :0.05354	Mode :character
Mean :48.92	Mean :1.696	Mean :1.582			Mean :0.8488	Mean :0.5028		Mean :0.11916	
3rd Qu.:72.00	3rd Qu.:2.000	3rd Qu.:2.000			3rd Qu.:0.9130	3rd Qu.:0.5880		3rd Qu.:0.10300	
Max. :97.00	Max. :8.000	Max. :6.000			Max. :0.9950	Max. :0.9750		Max. :0.97515	

Figure 1: Summary of atributes of the shots dataset

DirectFreekick	FromCorner	OpenPlay	Penalty	SetPiece
299	1316	5893	73	567

Figure 2: Situation of the shot

As we can see the most common situation of a shot is open play. The second most common situation of a shot is corner. The frequency of set pieces and direct free kicks is significant, too. However the frequency of penalties as a situation of a shot is insignificant.

Aerial	BallRecovery	BallTouch	BlockedPass	Card
735	175	202	7	4
Challenge	Chipped	Clearance	CornerAwarded	Cross
5	381	4	8	1192
Dispossessed	End	Foul	Goal	HeadPass
38	4	16	7	198
Interception	LayOff	None	OffsidePass	Pass
6	44	973	2	2805
Punch	Rebound	Save	Standard	Start
1	367	2	372	1
SubstitutionOn	Tackle	TakeOn	Throughball	
2	11	450	136	

Figure 3: Last action before the shot

Obviously the most common action before a shot is a pass. However the frequency of crosses and aerials is significant, too.

The second most common action before a shot is none - resp. the shot is immediately after a ball recovery.

The frequency of standard situations (set pieces), chipped balls, take-ons, rebounds, ball recoveries, head passes, through balls and ball touches is also notable.

The frequency of the remaining values is insignificant.

Head	LeftFoot	OtherBodyPart	RightFoot
1499	2437	34	4178

Figure 4: Type of the shot

As expected the majority of shots is shot by right or left foot. However the frequency of headers is significant, too.

The remaining shots are shot with other body part.

BlockedShot	Goal	MissedShots	OwnGoal	SavedShot	ShotOnPost
2016	954	3132	28	1897	121

Figure 5: Result of the shot

The result of the shot most frequently is a miss or a blocked shot. However the the amount of saved shots is almost equal with the amount of saved shots.

If a shot is not missed, blocked or saved by the goalkeeper it can be a goal or it can hit the post.

The amount of own goals is insignificant.

	Head	LeftFoot	OtherBodyPart	RightFoot
BlockedShot	187	687	2	1140
Goal	168	284	2	500
MissedShots	822	865	17	1428
OwnGoal	4	12	5	7
SavedShot	299	547	7	1044
ShotOnPost	19	42	1	59

Figure 6: Result vs. shot type

As we can see from the table, the number of left footed shots is about the half of the right footed shots in the Bundesliga. This means that in Bundesliga the number of left footed players is significantly smaller then the number of right footed players.

The second interesting thing to notice is the number of blocked and missed shots in case of headers. The distribution in this case is different because it is really hard to block a header, however the the percentage of goals is unchanged. The reason of this phenomenon is the higher number of missed shots.

	Head	LeftFoot	OtherBodyPart	RightFoot
Aerial	561	64	13	97
BallRecovery	0	65	0	110
BallTouch	5	64	1	132
BlockedPass	0	3	0	4
Card	0	1	2	1
Challenge	1	1	0	3
Chipped	75	108	0	198
Clearance	1	1	1	1
CornerAwarded	3	1	0	4
Cross	726	160	14	292
Dispossessed	1	12	1	24
End	0	2	0	2
Foul	7	2	0	7
Goal	3	3	0	1
HeadPass	31	55	0	112
Interception	0	2	0	4
LayOff	0	13	0	31
None	50	362	2	559
OffsidePass	0	0	0	2
Pass	11	1002	0	1792
Punch	1	0	0	0
Rebound	21	130	0	216
Save	0	2	0	0
Standard	0	147	0	225
Start	0	0	0	1
SubstitutionOn	1	0	0	1
Tackle	0	7	0	4
TakeOn	0	183	0	267
Throughball	1	47	0	88

Figure 7: Last action vs. shot type

As in the case of the previous table, we can see from the table that the number of left footed shots is about the half of the right footed shots.

One interesting thing to notice is that the headers usually come from a cross or an aerial duel. The number of shots from aerial duels and crosses is significantly smaller in case of shots with foot. However the number of shots from passes, standard situations, rebounds is higher in case of shots taken with foot.

	Head	LeftFoot	OtherBodyPart	RightFoot
DirectFreekick	0	131	0	168
FromCorner	646	264	11	395
OpenPlay	583	1925	16	3369
Penalty	0	16	0	57
SetPiece	270	101	7	189

Figure 8: Situation vs. shot type

If we compare the situations with the shot types one thing to notice is that headers mainly come from corners and set pieces, the percentage of headers from open play is significantly smaller than in the case of shots taken with foot.

	BlockedShot	Goal	MissedShots	OwnGoal	SavedShot	ShotOnPost
Aerial	140	50	418	0	121	6
BallRecovery	40	12	64	0	57	2
BallTouch	60	13	84	1	39	5
BlockedPass	3	0	0	0	4	0
Card	0	0	2	2	0	0
Challenge	1	0	0	3	1	0
Chipped	65	38	162	0	113	3
Clearance	0	0	0	4	0	0
CornerAwarded	4	0	2	0	2	0
Cross	176	166	610	0	222	18
Dispossessed	16	4	5	2	10	1
End	1	0	1	0	2	0
Foul	7	2	3	3	1	0
Goal	4	1	0	0	2	0
HeadPass	38	27	90	0	38	5
Interception	3	0	2	0	1	0
LayOff	20	2	11	0	11	0
None	306	74	402	9	170	12
OffsidePass	1	0	0	0	1	0
Pass	804	289	924	0	742	46
Punch	0	0	0	1	0	0
Rebound	92	79	121	0	69	6
Save	0	0	0	2	0	0
Standard	89	77	93	0	106	7
Start	0	0	0	0	1	0
SubstitutionOn	1	1	0	0	0	0
Tackle	3	0	4	0	4	0
TakeOn	132	66	114	1	131	6
Throughball	10	53	20	0	49	4

Figure 9: Last action vs. result

As we can see from the table, the most common action before a shot is a pass. However there is a difference between the second most common action in case of blocked shot and goal. The second most common action before a goal is cross, however the the second most common action before a blocked shot is none - which means that ball recovery is not as common before a goal than a cross.

	BlockedShot	Goal	MissedShots	OwnGoal	SavedShot	ShotOnPost
DirectFreekick	89	18	93	0	93	6
FromCorner	353	116	652	3	176	16
OpenPlay	1467	701	2119	22	1494	90
Penalty	0	59	0	0	13	1
SetPiece	107	60	268	3	121	8

Figure 10: Situation vs. result

One thing to notice from the table is the really good defence against corners. The ratio of goals from corners in comparison with blocked shots and missed shots is significantly higher than in case of open play situations. The number of saved shots from corner is very small, because of the good defence.

	DirectFreekick	FromCorner	OpenPlay	Penalty	SetPiece
Aerial	0	340	267	0	128
BallRecovery	0	0	175	0	0
BallTouch	0	33	157	0	12
BlockedPass	0	1	6	0	0
Card	0	0	3	0	1
Challenge	0	1	4	0	0
Chipped	0	19	324	0	38
Clearance	0	0	4	0	0
CornerAwarded	0	3	4	0	1
Cross	0	402	645	0	145
Dispossessed	0	2	35	0	1
End	0	1	3	0	0
Foul	0	5	8	0	3
Goal	0	1	4	0	2
HeadPass	0	44	132	0	22
Interception	0	1	5	0	0
LayOff	0	0	44	0	0
None	0	254	614	0	105
OffsidePass	0	0	2	0	0
Pass	0	102	2651	0	52
Punch	0	1	0	0	0
Rebound	0	90	230	0	47
Save	0	0	2	0	0
Standard	299	0	0	73	0
Start	0	0	1	0	0
SubstitutionOn	0	0	2	0	0
Tackle	0	2	9	0	0
TakeOn	0	14	428	0	8
Throughball	0	0	134	0	2

Figure 11: Last action vs. situation

First thing to notice is that standard situation can be only a direct free kick or a penalty. The second thing to notice is that from corners and set pieces teams tend to create chances with

aerial duels and crosses. The probability of a pass in this situations is very low in comparison with open play.

	shotType	xG.Min.	xG.1st Qu.	xG.Median	xG.Mean	xG.3rd Qu.	xG.Max.
1	Head	0.00000000	0.03194033	0.05912367	0.12088503	0.12052879	0.93462932
2	LeftFoot	0.00000000	0.02453937	0.05030815	0.11242970	0.09488232	0.97515047
3	OtherBodyPart	0.00000000	0.03723685	0.05526645	0.13402930	0.16638546	0.80551404
4	RightFoot	0.00000000	0.02497848	0.05274451	0.12235386	0.10147913	0.97444212

Figure 12: xG vs. shot type

If we compare the continuous variable of xG with shot type we can see, that the the left footed shots are less dangerous than the right footed shots. The reason behind this phenomenon is the number of right footed players in Bundesliga. The right footed players tend to be less dangerous with their left (weeker) foot then with their right foot. However the headers are as efficient as the right footed shots.

	situation	xG.Min.	xG.1st Qu.	xG.Median	xG.Mean	xG.3rd Qu.	xG.Max.
1	DirectFreekick	0.01576263	0.04009316	0.04944210	0.05730367	0.06439684	0.45049843
2	FromCorner	0.00000000	0.02258906	0.04572779	0.09020780	0.08309084	0.94018155
3	OpenPlay	0.00000000	0.02597146	0.05580397	0.12088174	0.10839295	0.97515047
4	Penalty	0.75736868	0.75777668	0.75777668	0.75774345	0.75777668	0.75777668
5	SetPiece	0.00000000	0.02050503	0.05005096	0.11892526	0.11028392	0.95154202

Figure 13: xG vs. situation

If we compare the continuous variable xG with situations we can see, that penalties has very high mean and median in comparison with other situations. The reason behind this phenomenon is that penalties has a very high success rate, too - resp. around 75% of the penalties is scored.

Another thing to note is the danger of direct free-kicks. In case of direct free-kicks the goalkeeper usually has a wall of players to protect the striker from scoring. This means that the average xG of these situations is relatively small.

The same thing applies to corners, where the defending team has time to prepare for the situation.

	result	xG.Min.	xG.1st Qu.	xG.Median	xG.Mean	xG.3rd Qu.	xG.Max.
1	BlockedShot	0.006098521	0.023662352	0.046925854	0.063925391	0.076051876	0.906968832
2	Goal	0.008901252	0.081201090	0.348315194	0.342520259	0.547447070	0.975150466
3	MissedShots	0.005251109	0.021591197	0.044928486	0.084799500	0.082207451	0.854806662
4	OwnGoal	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000	0.000000000
5	SavedShot	0.005597099	0.028085340	0.054072939	0.119456712	0.111933261	0.846527636
6	ShotOnPost	0.013127367	0.046343926	0.082838558	0.190991162	0.319782346	0.786608458

Figure 14: xG vs. result

If we compare the continuous variable xG with result we can notice that the average xG needed for a goal is 0.3.

Shots which hit the post usually have xG close to 0.2.

The average xG of blocked shots and missed shots is very low.

situation	X.Min.	X.1st Qu.	X.Median	X.Mean	X.3rd Qu.	X.Max.
DirectFreekick	0.3910000	0.7220000	0.7500000	0.7539331	0.7815000	0.9340000
FromCorner	0.0180000	0.8517500	0.9090000	0.8823108	0.9310000	0.9900000
OpenPlay	0.0040000	0.7860000	0.8600000	0.8433112	0.9050000	0.9950000
Penalty	0.8850000	0.8850000	0.8850000	0.8850000	0.8850000	0.8850000
SetPiece	0.0230000	0.8455000	0.8960000	0.8727866	0.9220000	0.9900000

Figure 15: X coordinate vs. situation

If we compare the continuous variable X with the situation, we can see that the shots are taken very close to the goal. However the direct free-kicks tend to be farther from the goal - which may explain the low efficiency of these types of shots.

shotType	X.Min.	X.1st Qu.	X.Median	X.Mean	X.3rd Qu.	X.Max.
Head	0.0640000	0.8980000	0.9160000	0.9131961	0.9320000	0.9900000
LeftFoot	0.0180000	0.7720000	0.8490000	0.8327948	0.8960000	0.9920000
OtherBodyPart	0.0100000	0.8867500	0.9170000	0.7894412	0.9295000	0.9740000
RightFoot	0.0040000	0.7740000	0.8500000	0.8354265	0.8970000	0.9950000

Figure 16: X coordinate vs. shot type

If we compere the continuous variable X with shot type we an notice that shots with left and right foot are equally close to the goal. However shots taken with head are closer to the goal.

This phenomenon can be explained with the fact, that corners are curved very close to the goal.

result	X.Min.	X.1st Qu.	X.Median	X.Mean	X.3rd Qu.	X.Max.
BlockedShot	0.63599998	0.77800003	0.84800003	0.83567560	0.89000000	0.97900002
Goal	0.64199997	0.87199997	0.90900002	0.89792977	0.93599998	0.99199997
MissedShots	0.33700001	0.77699997	0.86699997	0.84662069	0.91300003	0.99500000
OwnGoal	0.00400000	0.02675000	0.04150000	0.04742857	0.06725000	0.12600000
SavedShot	0.28299999	0.79400002	0.87000000	0.85199473	0.91099998	0.98400002
ShotOnPost	0.69599998	0.81500000	0.88500000	0.86884297	0.92300003	0.99000000

Figure 17: X coordinate vs. result

If we compare X with result we can notice that goals usually are from very short distance. If the distance from the goal is bigger, then the chance of the goal is smaller.

result	minute.Min.	minute.1st Qu.	minute.Median	minute.Mean	minute.3rd Qu.	minute.Max.
BlockedShot	0.00000	27.00000	50.00000	48.84077	71.00000	96.00000
Goal	0.00000	28.25000	51.00000	49.78616	73.00000	97.00000
MissedShots	0.00000	27.00000	49.00000	48.58525	71.00000	97.00000
OwnGoal	3.00000	32.00000	54.00000	50.35714	70.00000	88.00000
SavedShot	0.00000	27.00000	49.00000	49.16342	72.00000	97.00000
ShotOnPost	1.00000	24.00000	44.00000	47.59504	77.00000	92.00000

Figure 18: Time vs. result

If we compare time with result we can notice a small asymmetry between the halftimes. The players are more active in the second half - after the 45. minute.

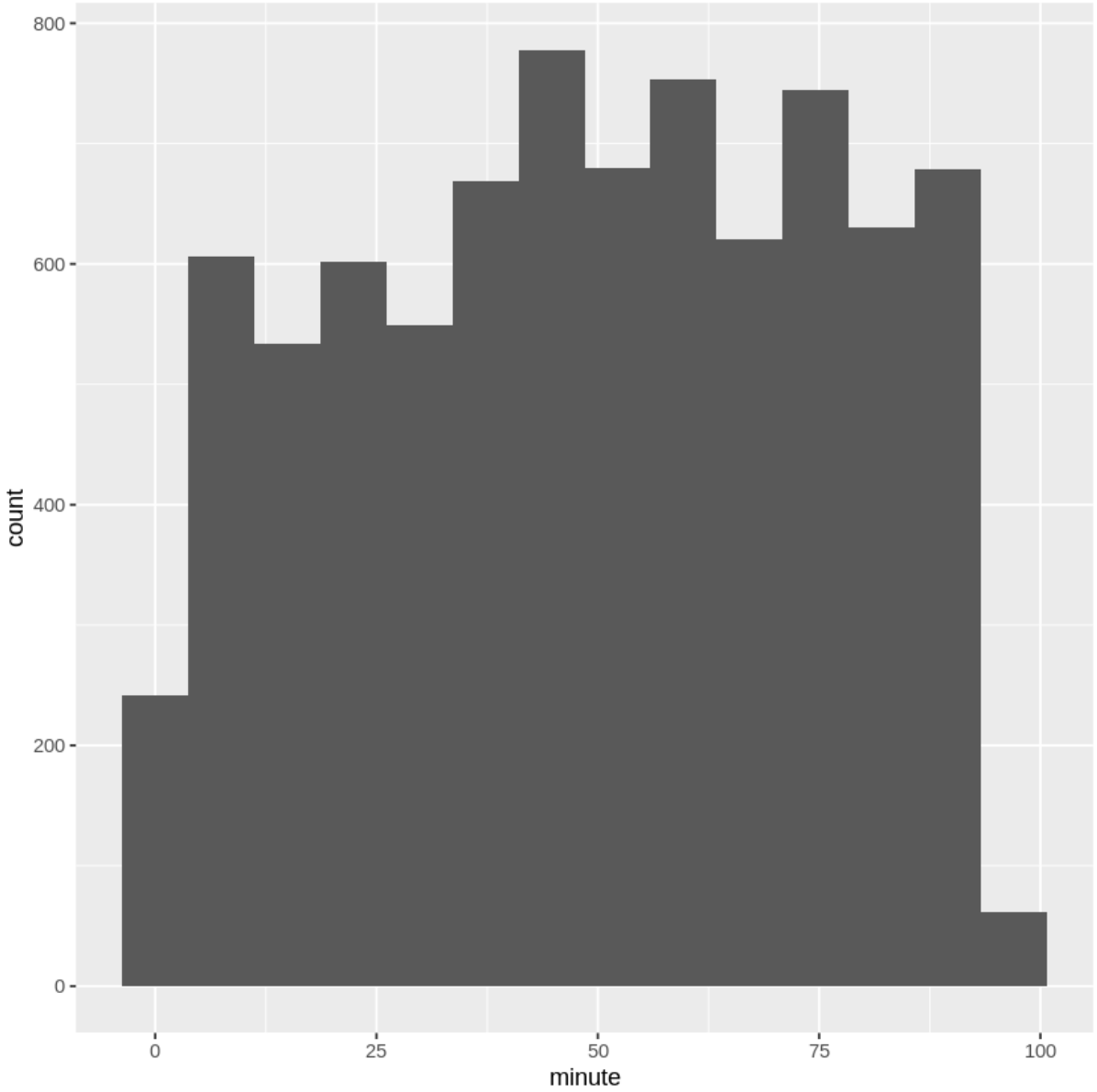


Figure 19: Time vs. number of shots

We can see on the graph that the players are more active in second half. The number of

shots taken in the second half is significantly higher than in the first half.

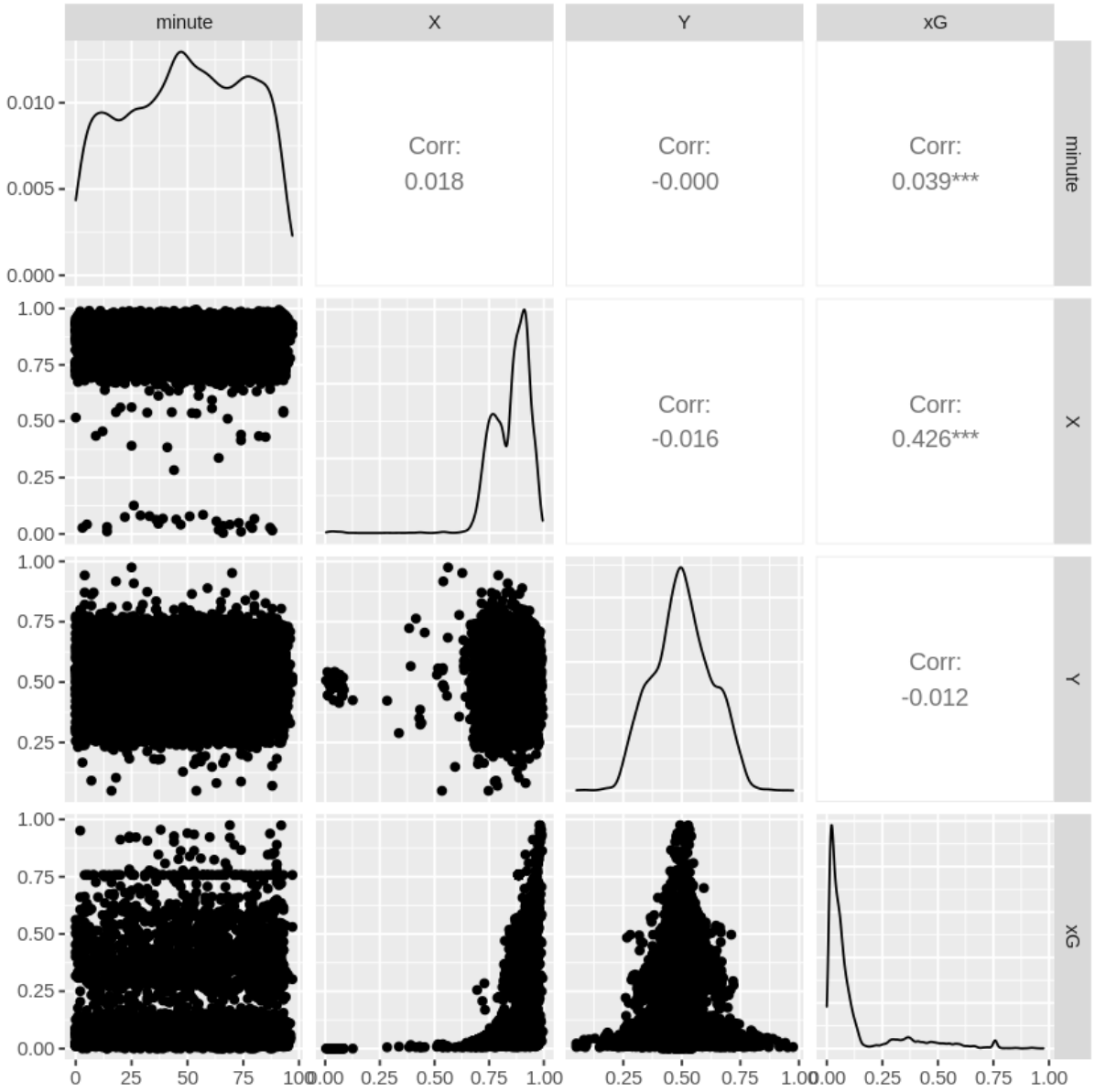


Figure 20: Correlation map of continuous attributes

If we talk about the correlation of continuous variables, then we can notice, that the only significant correlation is between the xG and the X variable. Which means, that the distance from the goal is the main factor, which determines the size of xG.

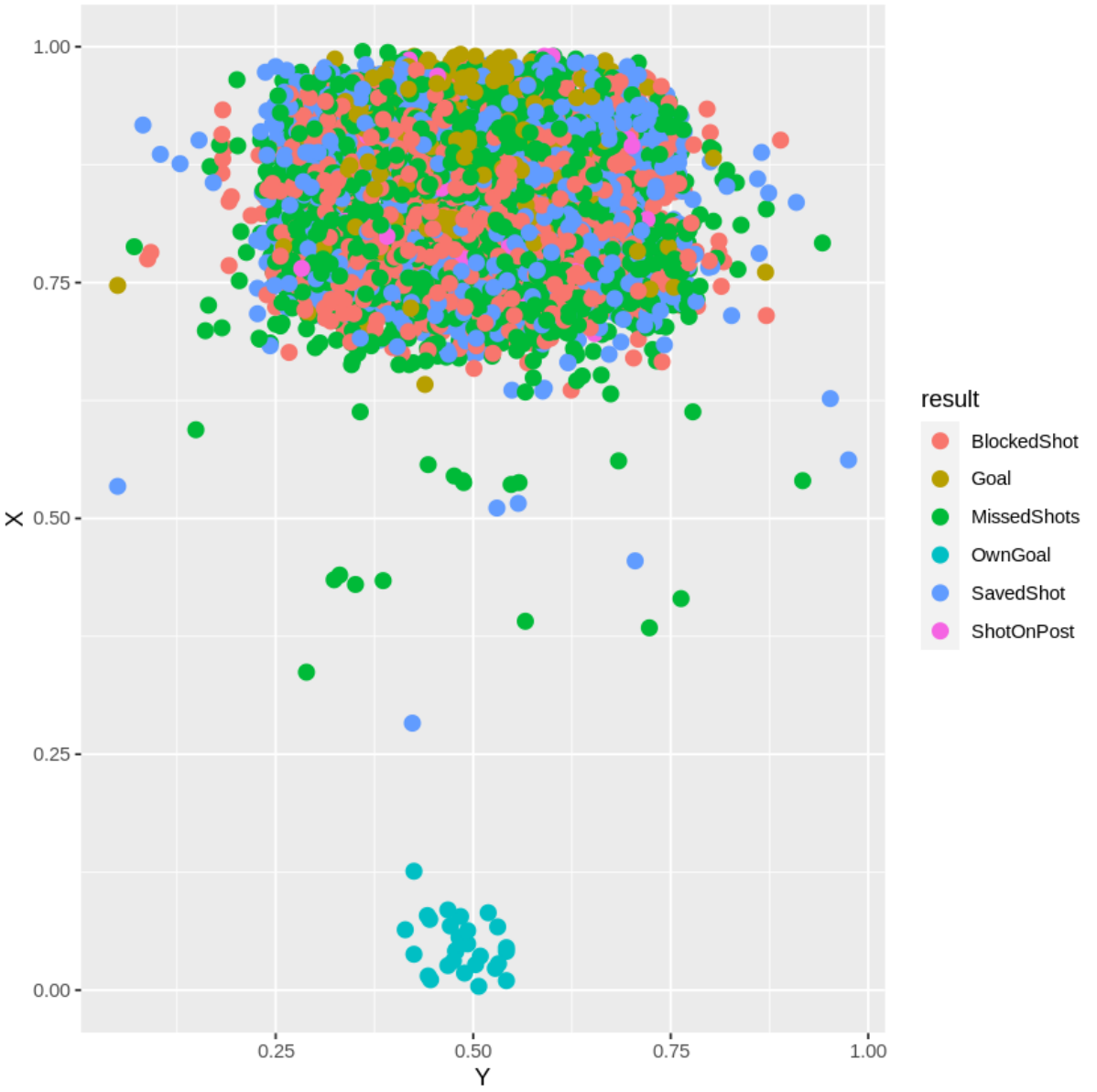


Figure 21: Distance from goal vs horizontal position with shot results

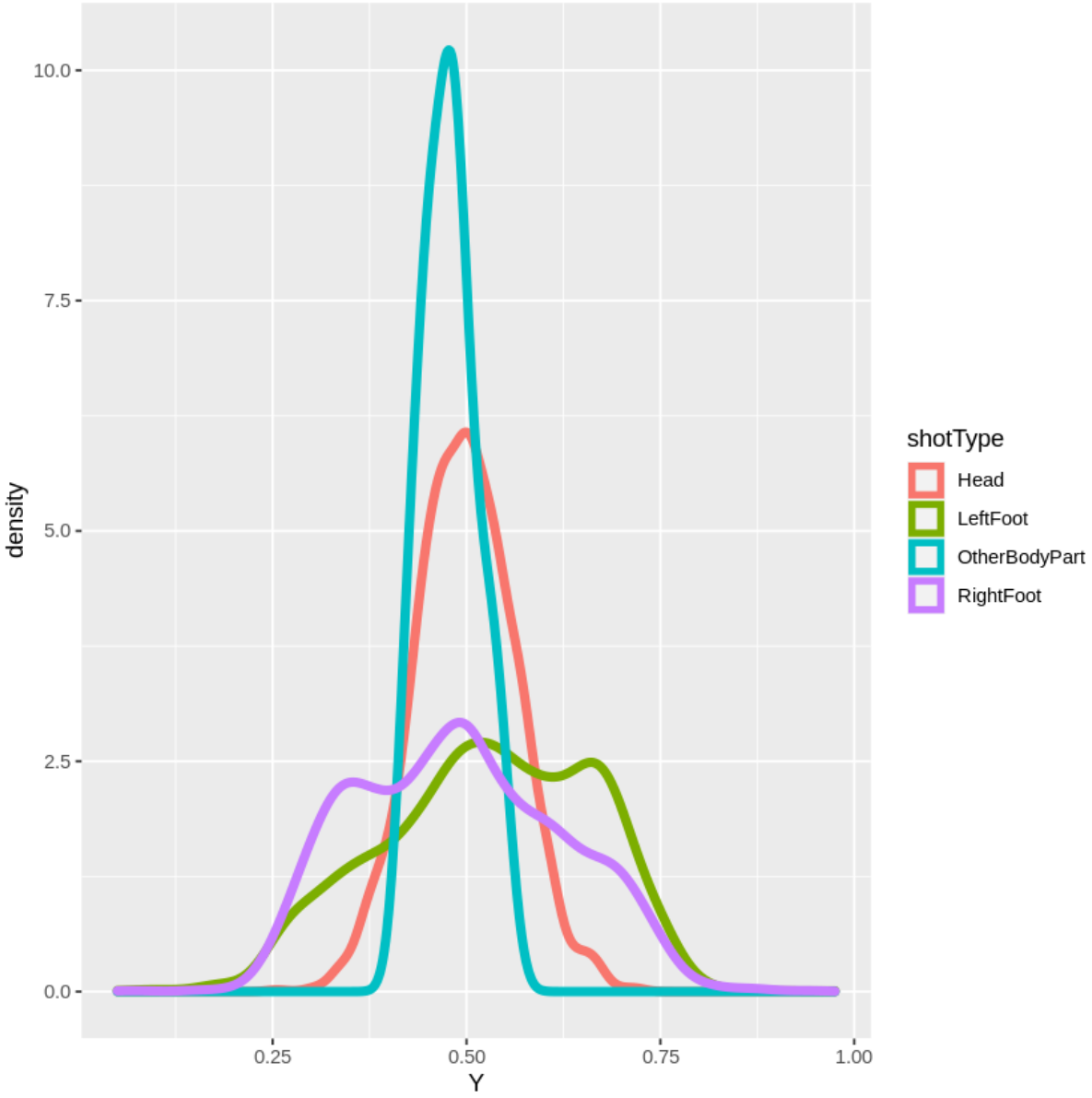


Figure 22: Horizontal density of shot types

We can see on the graph

4 Hypothesis Testing

4.1 Predicting the effectiveness of a striker based on ambidexterity

One of the widely accepted facts in football is, that if a player is good with both of his legs and with his head, too, than he scores a lot of goals.

To prove this fact we introduced a new parameter which shows how complete is a given player. Completeness of a player is a number between 0 and 1, and it is calculated from the shots of a given player taken by his right foot, left foot and with his head.

We called this metric "difference".

Difference is the normalized version of the standard deviation of the number of shots with right foot, left foot and with head for a given player.

If the number is close to zero, then the player is complete, he uses both of his legs and his head well - the number of shots taken with his right foot, left foot and with his head is very close.

If the number is close to one, then the player uses only one of his legs, or his head to shoot.

Beside that we introduced two more metrics of completeness of a given player: difference_shots, difference_goals.

difference_shots is almost the same metric as difference, however we do not take into account the shots taken with head and the metric is calculated with traditional subtraction method. Which means that if the number is close to zero, then the player uses his left leg to shoot, if the number is close to 0.5 then the player uses both of his legs and if the number is close to one, then the player mainly shoots with his right foot.

difference_goals is calculated from the number of goals with right or left foot of a given player with the same process as difference_shots. If the number is close to 0, then the player manly scores with his left leg, if the number is close to 0.5 then the player uses equivalently good both of his legs to score goals and if the number is close to 1, then the player tends to use his right foot to score.

We calculated these parameters for strikers with more then 5 goals in Bundesliga season 2019/2020.

The results of the calculation are very strange. If we show the results on graphs, then we can see, that players with high completeness tend to be less successful strikers and the players with low completeness - who are using only their left or right foot to score tend to be better in front of the goal.

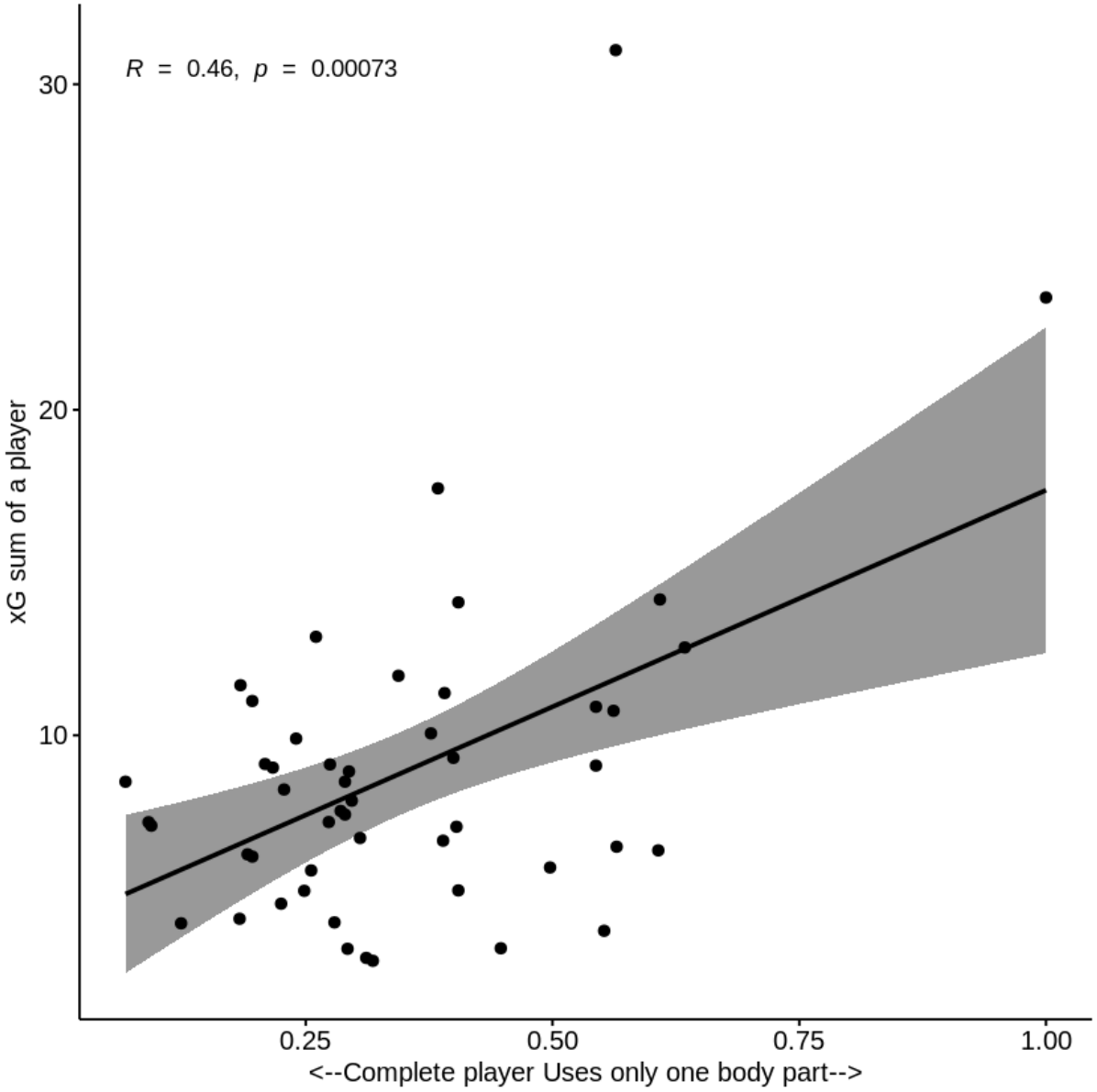


Figure 23: difference vs. sum of xG

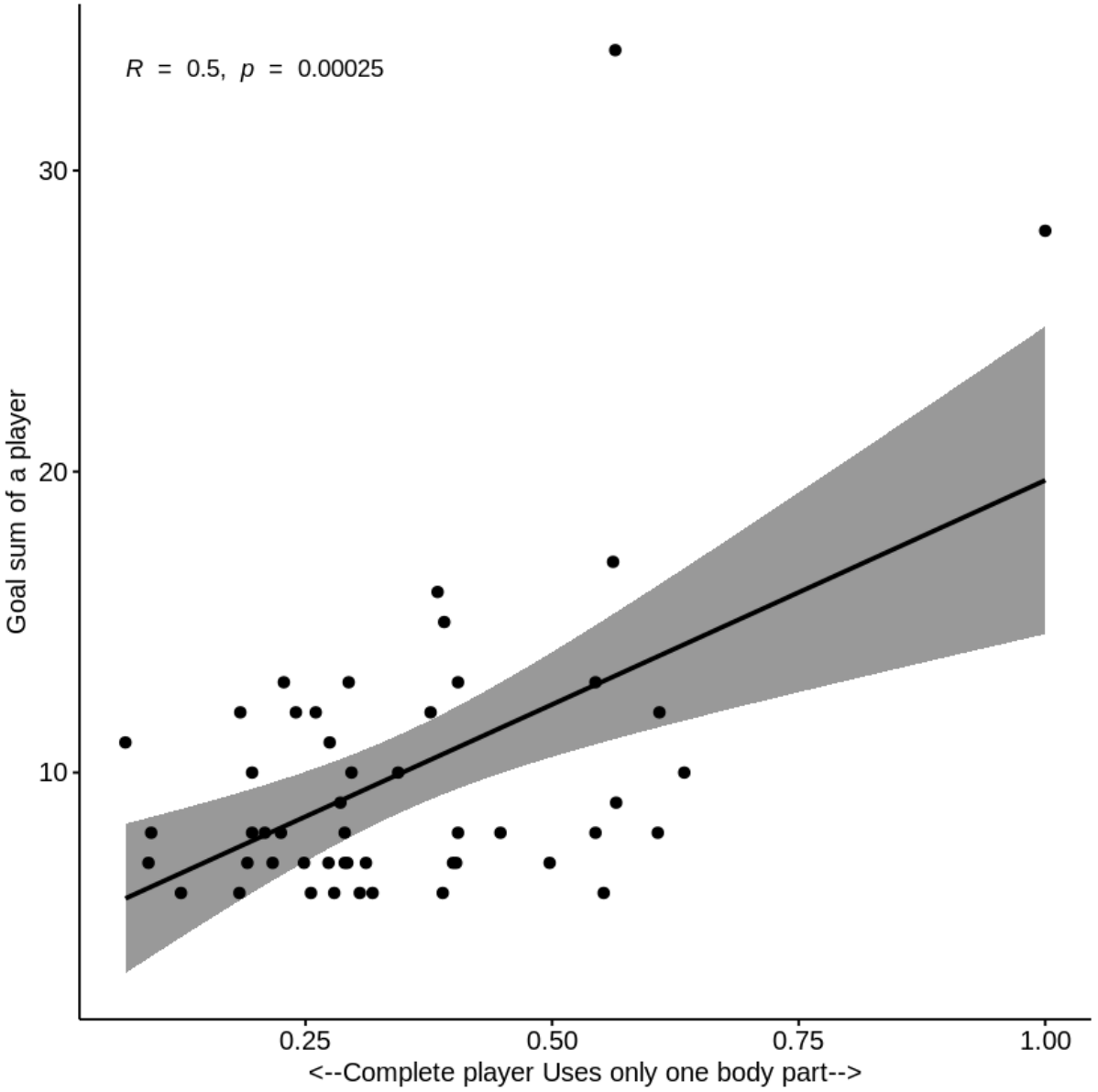


Figure 24: difference vs. sum of goals

The results of the calculation show, that players who are very good with only one foot in

front of the goal are better strikers overall, and tend to score more goals.

4.2 Predicting the outcome of a shot using logistic regression

The xG variable (expected goals) in our data is calculated by Understat from a large dataset (over 100,000 shots, 10 parameters for each) using a neural newtork. Since the algorithm Understat uses is not public, we do not exactly know what parameters they use to predict the outcome of a shot. In this section we create a model similar to Understat's xG.

Predicting the outcome of a shot

Prep

Imports

```
Bundesliga_shots_2019 <- read.csv("data/shots/Bundesliga_shots_2019.csv")
Bundesliga_shots_2020 <- read.csv("data/shots/Bundesliga_shots_2020.csv")
EPL_shots_2020 <- read.csv("data/shots/EPL_shots_2020.csv")
SerieA_shots_2020 <- read.csv("data/shots/SerieA_shots_2020.csv")
LaLiga_shots_2020 <- read.csv("data/shots/LaLiga_shots_2020.csv")
Ligue1_shots_2020 <- read.csv("data/shots/Ligue1_shots_2020.csv")
```

Helper functions

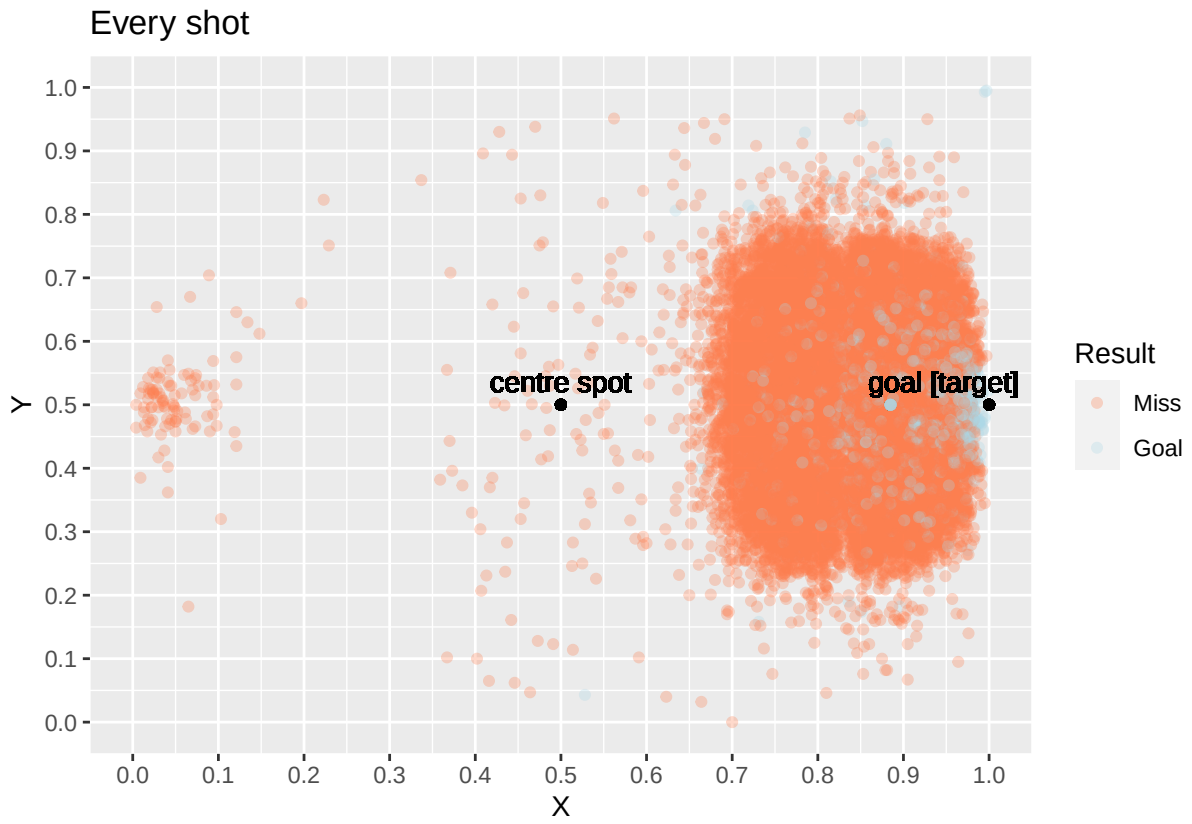
```
# calculates shot angle
calc_angle <- function(X, Y){
  y <- as.matrix(c(X - 1, Y - 0.5))
  x <- t(as.matrix(c(-1, 0)))
  dot.prod <- x %*% y
  norm.x <- norm(x, type="2")
  norm.y <- norm(y, type="2")
  theta <- acos(dot.prod / (norm.x * norm.y))
  return (as.numeric(theta) * (180 / pi))
}

'%ni%' <- Negate('%in%')
```

MEDA

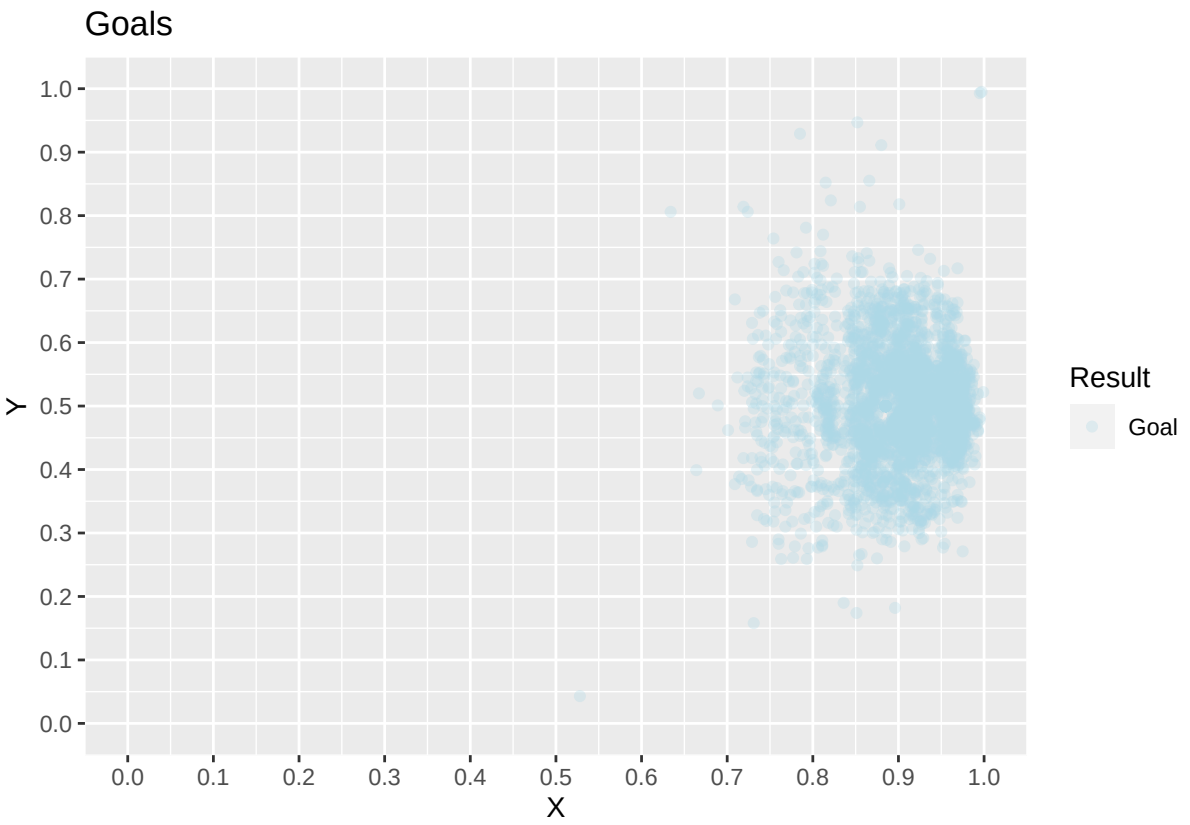
```
# converts result to binary (1 - goal, 0 - miss)
shots <- rbind(
  Bundesliga_shots_2020,
  EPL_shots_2020,
  SerieA_shots_2020,
  LaLiga_shots_2020,
  Ligue1_shots_2020) %>%
  mutate(result = ifelse(result == 'Goal', 1, 0))

# str(shots)
```

Mapping the positions of shots.

```
shots %>%
  ggplot(.) +
  geom_point(mapping = aes(x = X, y = Y, color = factor(result)), alpha = 0.3) +
  geom_point(mapping = aes(x = 0.5, y = 0.5), alpha = 1) +
  geom_text(aes(x = 0.5, y = 0.5, label = 'centre spot'), hjust = 0.5, vjust = -0.6) +
  geom_point(mapping = aes(x = 1, y = 0.5), alpha = 1) +
  geom_text(aes(x = 1, y = 0.5, label = 'goal [target]'), hjust = 0.8, vjust = -0.6) +
  scale_color_manual(labels = c('Miss', 'Goal'), values = c('coral', 'lightblue')) +
  scale_x_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  labs(title = 'Every shot', color = 'Result')
```




```
shots %>%
  filter(result == 1) %>%
  ggplot(.) +
  geom_point(mapping = aes(x = X, y = Y, color = factor(result)), alpha = 0.3) +
  scale_color_manual(labels = c('Goal'), values = c('lightblue')) +
  scale_x_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  labs(title = 'Goals', color = 'Result')
```



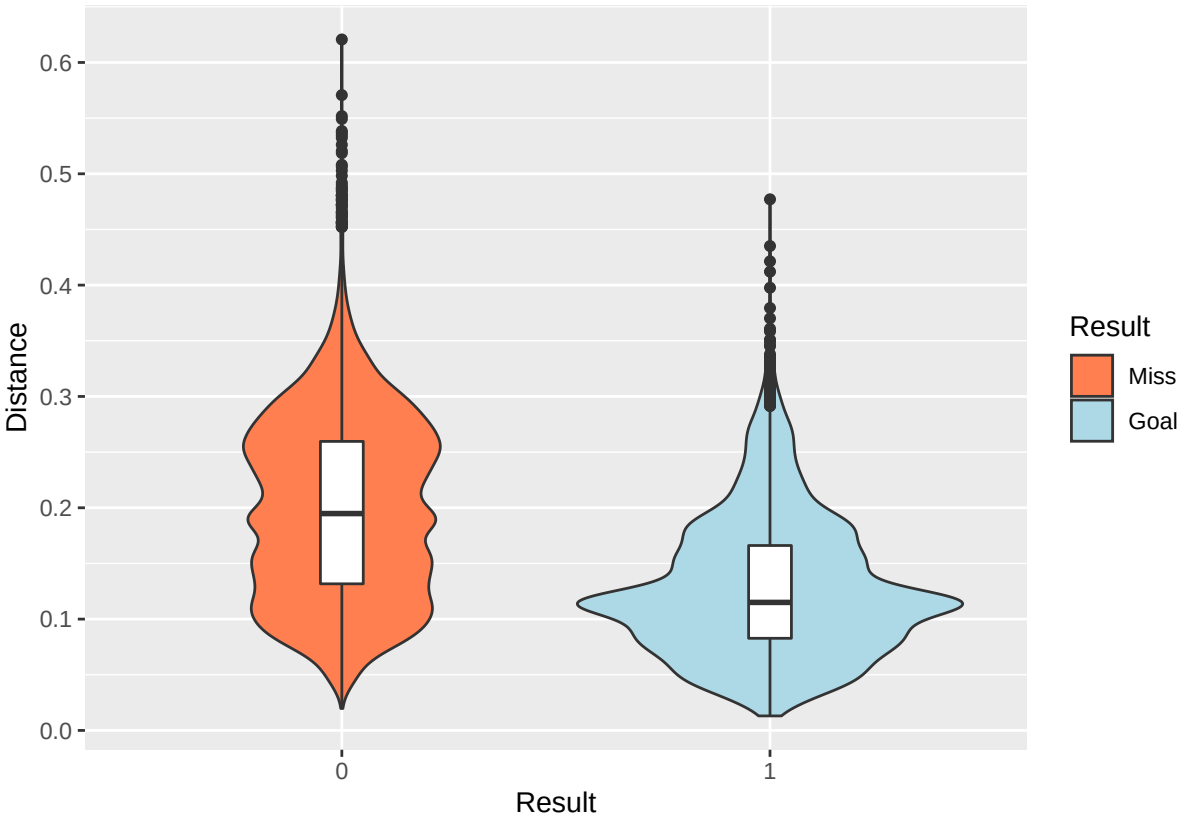
To make our predictions more precise we filtered out shots from the range $0 < X < 0.5$, since every goal comes from the opposite's half. There are 5 goals in our dataset in the range $Y < 0.1 \mid\mid Y > 0.9$. After manually checking these 5 goals we decided to remove shots from this range, because the goals were conceded as a result of goalkeeper errors.

```
shots <- shots %>%
  # filters out shots from over the half-way line and shots close to the sideline
  filter(X > 0.5, Y < 0.9 & Y > 0.1) %>%
  # calculates the distance
  mutate(distance = sqrt((1 - X)^2 + (0.5 - Y)^2))

# calculates shot angle
for(i in seq(1:nrow(shots))) {
  shots[i, 'shot_angle'] <- calc_angle(as.numeric(shots[i, 'X']), as.numeric(shots[i, 'Y']))
}

goals <- shots %>% filter(result == 1)
# str(shots)
```

Distance

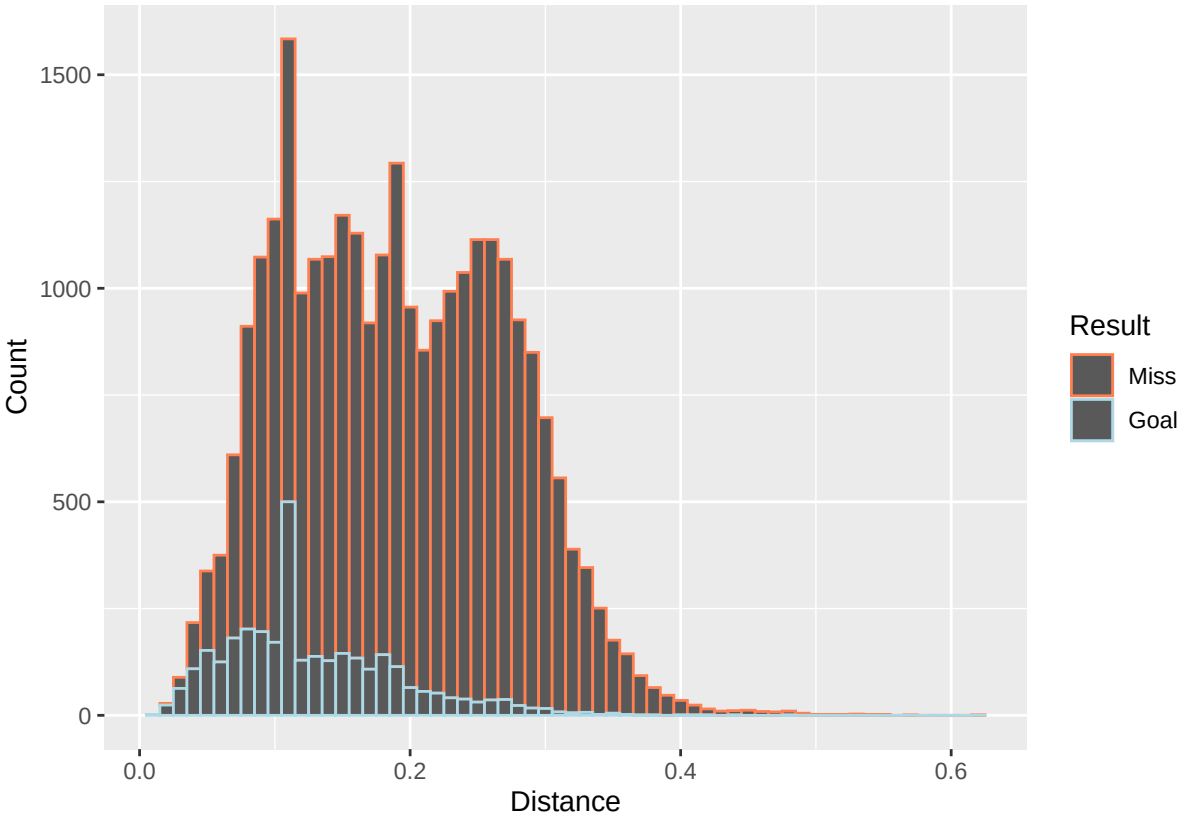
The combined boxplot below shows the distribution of goals and missed shots - we can see, that missed shots are distributed more evenly and most of the goals come from the penalty area (cca. distance < 1.5).

```
shots %>%
  ggplot(., aes(x = factor(result), y = distance)) +
  geom_violin(aes(fill = factor(result))) +
  geom_boxplot(width = 0.1) +
  scale_y_continuous(breaks = seq(0, 1, 0.1)) +
  scale_fill_manual(labels = c('Miss', 'Goal'), values = c('coral', 'lightblue')) +
  labs(y = 'Distance', x = 'Result', fill = 'Result')
```



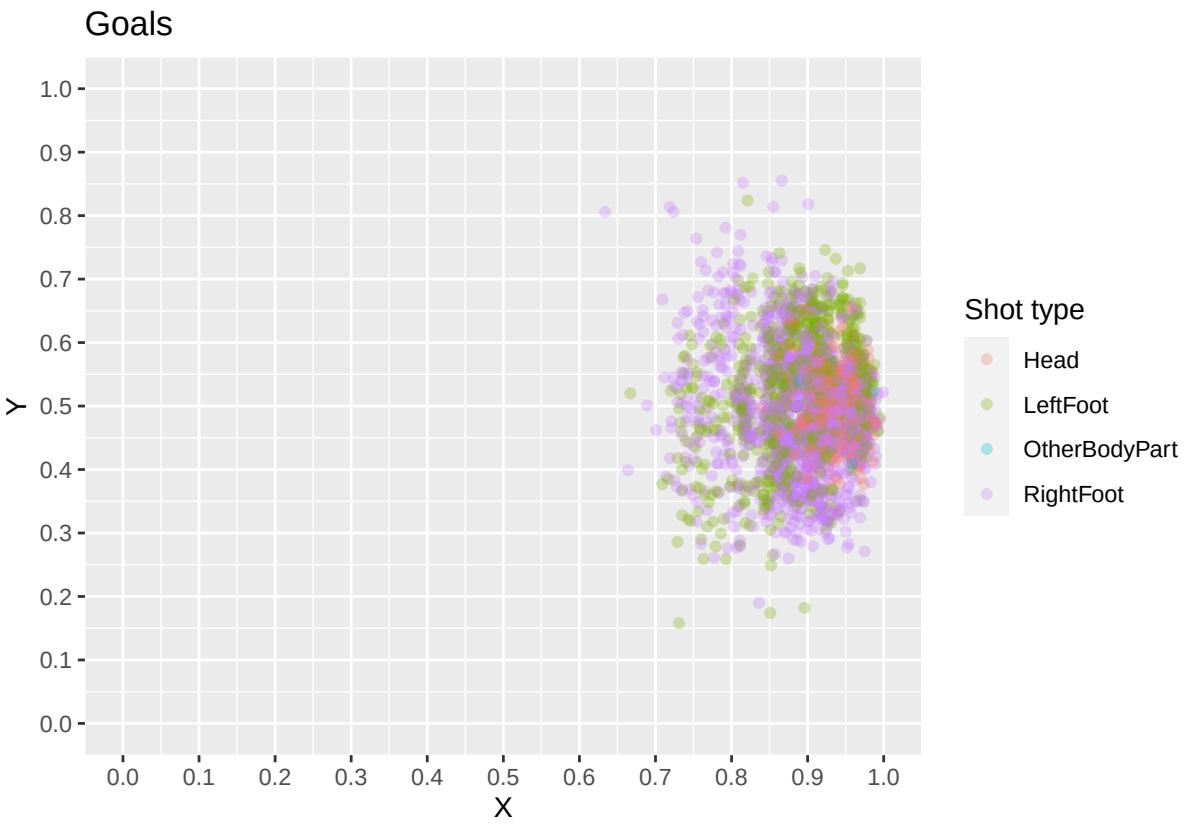
The histogram below shows the total of missed shots and goals. As we expected, the closer the shot is from the goal (target), the more likely it is going to be goal.

```
shots %>%
  ggplot(., aes(x = distance, color = factor(result))) +
  scale_color_manual(labels = c('Miss', 'Goal'), values = c('coral', 'lightblue')) +
  geom_histogram(binwidth = 0.01) +
  labs(x = 'Distance', y = 'Count', color = 'Result')
```



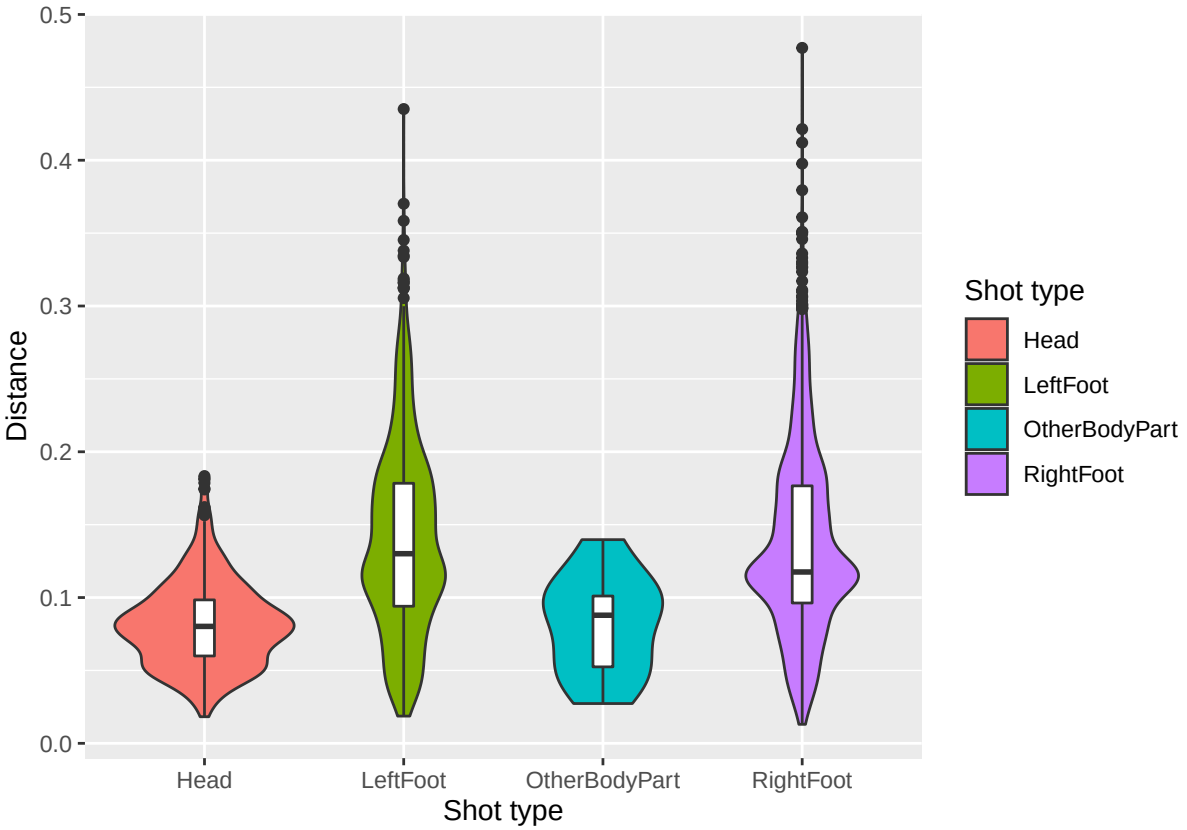
Mapping the position of goals based on shot type.

```
goals %>%
  ggplot(.) +
  geom_point(mapping = aes(x = X, y = Y, color = factor(shotType)), alpha = 0.3) +
  scale_x_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  labs(title = 'Goals', color = 'Shot type')
```



The type of the shot (head, left foot, right foot, or other body part) strongly affects the distribution of goals.

```
goals %>%
  ggplot(., aes(x = factor(shotType), y = distance)) +
  geom_violin(aes(fill = factor(shotType))) +
  geom_boxplot(width = 0.1) +
  scale_y_continuous(breaks = seq(0, 1, 0.1)) +
  labs(y = 'Distance', x = 'Shot type', fill = 'Shot type')
```

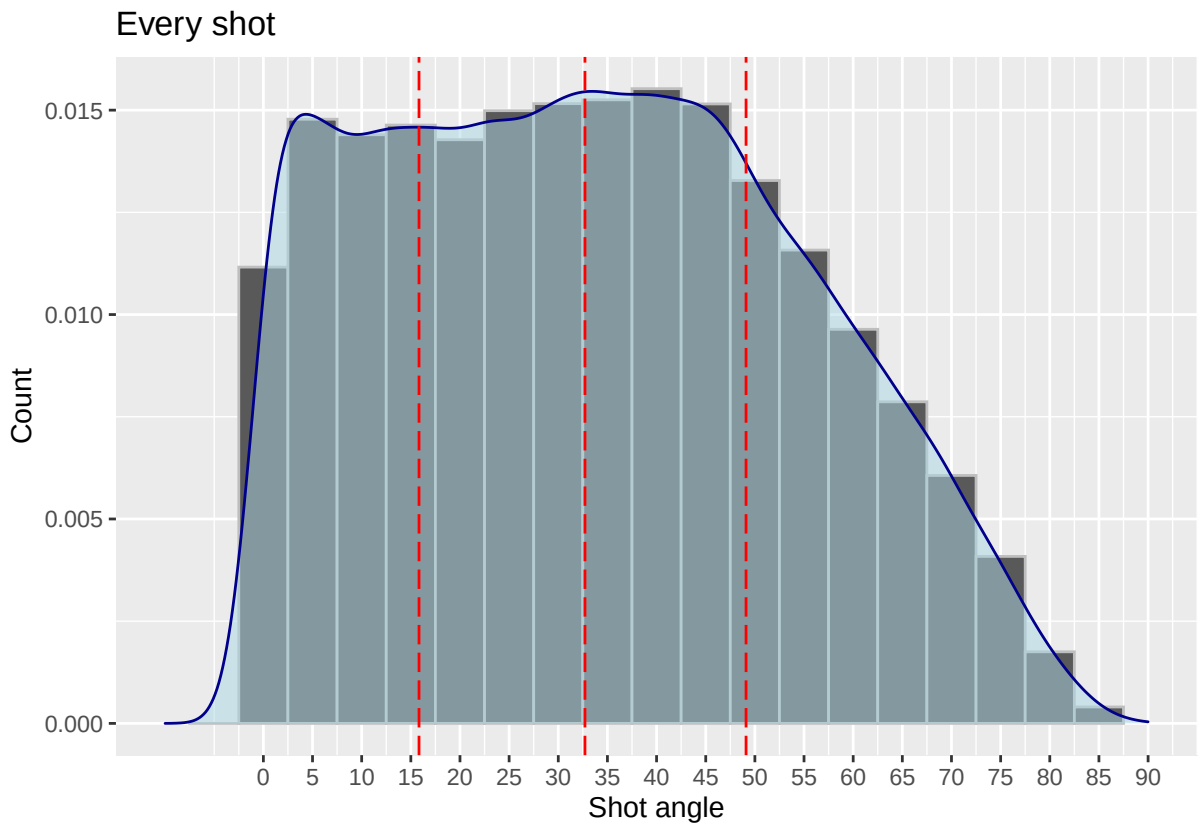


Shot angle

The histogram below shows the distribution of shots based on the angle of the shot. Missed shots are more evenly distributed compared to goals: 25% of goals has a shot angle smaller than 15.8°. We can see that a lot of goals is in the range: <0°, 5°>

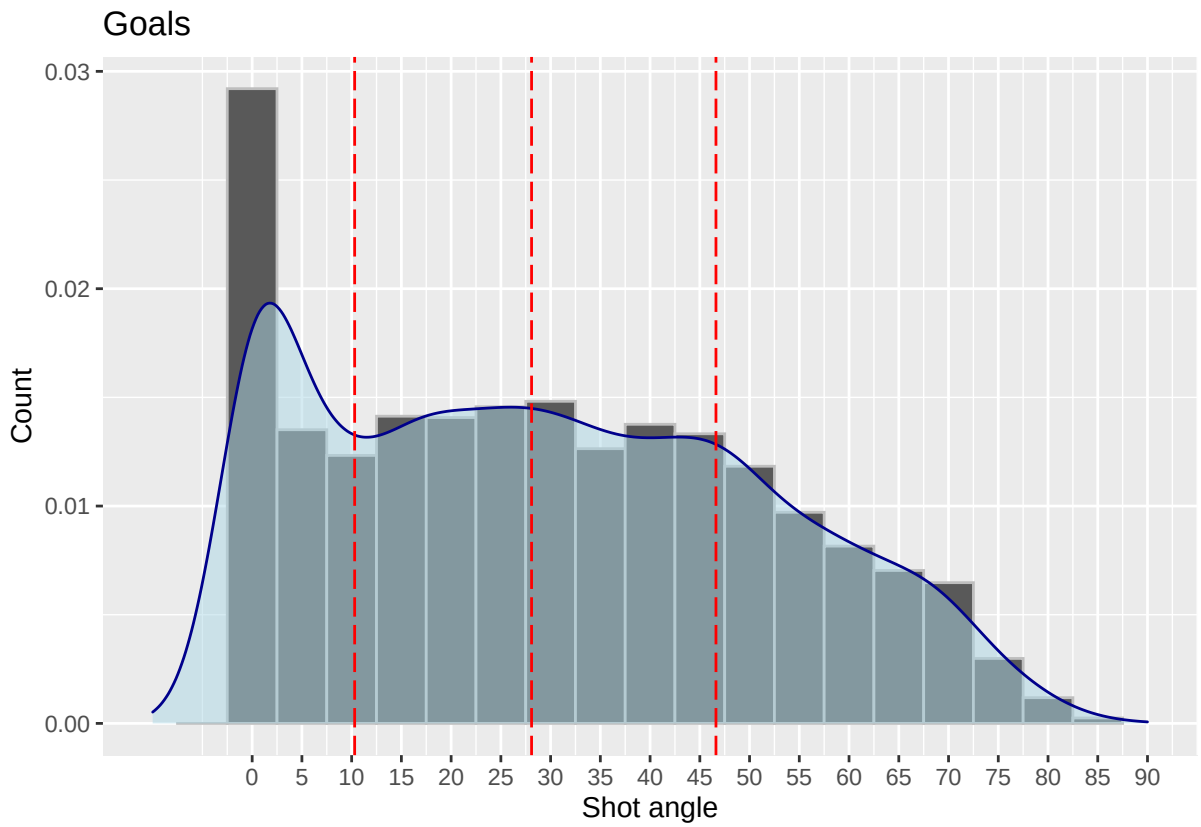
```
# every shot
quantiles <- quantile(shots$shot_angle)
# quantiles

shots %>%
  ggplot(., aes(x = shot_angle)) +
  geom_histogram(aes(y = ..density..), binwidth = 5, colour = 'grey') +
  geom_density(color='darkblue', fill='lightblue', alpha = 0.5) +
  geom_vline(xintercept = quantiles[2:4], colour = 'red', linetype ='longdash') +
  labs(title = 'Every shot', x = 'Shot angle', y = 'Count', fill = 'Result') +
  scale_x_continuous(limits = c(-10, 90), breaks = seq(0, 90, 5))
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```




```
# goals
quantiles <- quantile(goals$shot_angle)
# quantiles

goals %>%
  ggplot(., aes(x = shot_angle)) +  # fill = factor(shotType)
  geom_histogram(aes(y = ..density..), binwidth = 5, colour = 'grey') +
  geom_density(color='darkblue', fill='lightblue', alpha = 0.5) +
  geom_vline(xintercept = quantiles[2:4], colour = 'red', linetype ='longdash') +
  labs(title = 'Goals', x = 'Shot angle', y = 'Count', fill = 'Result') +
  scale_x_continuous(limits = c(-10, 90), breaks = seq(0, 90, 5))
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



Logistic regression

We used logistic regression to predict the outcome of a shot.

Data preparation

```
# train + test dataset
shots <- rbind(
  Bundesliga_shots_2020,
  EPL_shots_2020,
  SerieA_shots_2020,
  LaLiga_shots_2020,
  Ligue1_shots_2020) %>%
  filter(X > 0.5, Y < 0.9) %>%
  mutate(distance = sqrt((1 - X)^2 + (0.5 - Y)^2)) %>%
  # since shot types 'Head' and 'OtherBodyPart' had similar characteristics, we combined them
  mutate(shotType = ifelse(shotType == 'OtherBodyPart', 'Head', shotType)) %>%
  mutate(Class = ifelse(result == 'Goal', 1, 0))
shots$Class <- factor(shots$Class, levels = c(0, 1))
shots$situation <- as.factor(shots$situation)
shots$shotType <- as.factor(shots$shotType)
for(i in seq(1:nrow(shots))) {
  shots[i, 'shot_angle'] <- calc_angle(as.numeric(shots[i, 'X']), as.numeric(shots[i, 'Y']))
```

```
}

# validation dataset
validationShots <- rbind(Bundesliga_shots_2019) %>%
  filter(X > 0.5, Y < 0.9) %>%
  mutate(distance = sqrt((1 - X)^2 + (0.5 - Y)^2)) %>%
  mutate(shotType = ifelse(shotType == 'OtherBodyPart', 'Head', shotType)) %>%
  mutate(Class = ifelse(result == 'Goal', 1, 0))
validationShots$Class <- factor(validationShots$Class, levels = c(0, 1))
validationShots$situation <- as.factor(validationShots$situation)
validationShots$shotType <- as.factor(validationShots$shotType)
for(i in seq(1:nrow(validationShots))) {
  validationShots[i, 'shot_angle'] <- calc_angle(
    as.numeric(validationShots[i, 'X']),
    as.numeric(validationShots[i, 'Y']))
}
```

Creating training and testing datasets. Considering that the amount of missed shots is much higher the the amount of goals (cca. 1:7), we had to downsample missed shots to eliminate distortion in training.

```
set.seed(44)

sample <- sample(c(TRUE, FALSE), nrow(shots), replace = T, prob = c(0.7, 0.3))
trainShots <- shots[sample, ]
testShots <- shots[!sample, ]

# downsamples missed shots
trainShots <- downSample(x = trainShots[, colnames(trainShots) %ni% 'Class'],
                         y = trainShots$Class)

# table(trainShots$Class)
```

Building the model

Based on the exploratory data analysis we decided to use the distance and angle of the shot in our model, as well as the situation and the type of the shot. The summary shows that the model has good fit: most importantly the deviance residuals are balanced and the p-value is lower than 0.05 (at every parameter).

```
# builds the model
set.seed(44)
model <- glm(Class ~ distance + shot_angle + situation + shotType,
             data = trainShots,
             family = 'binomial')

summary(model)
```

```
##
## Call:
## glm(formula = Class ~ distance + shot_angle + situation + shotType,
##     family = "binomial", data = trainShots)
##
## Deviance Residuals:
```

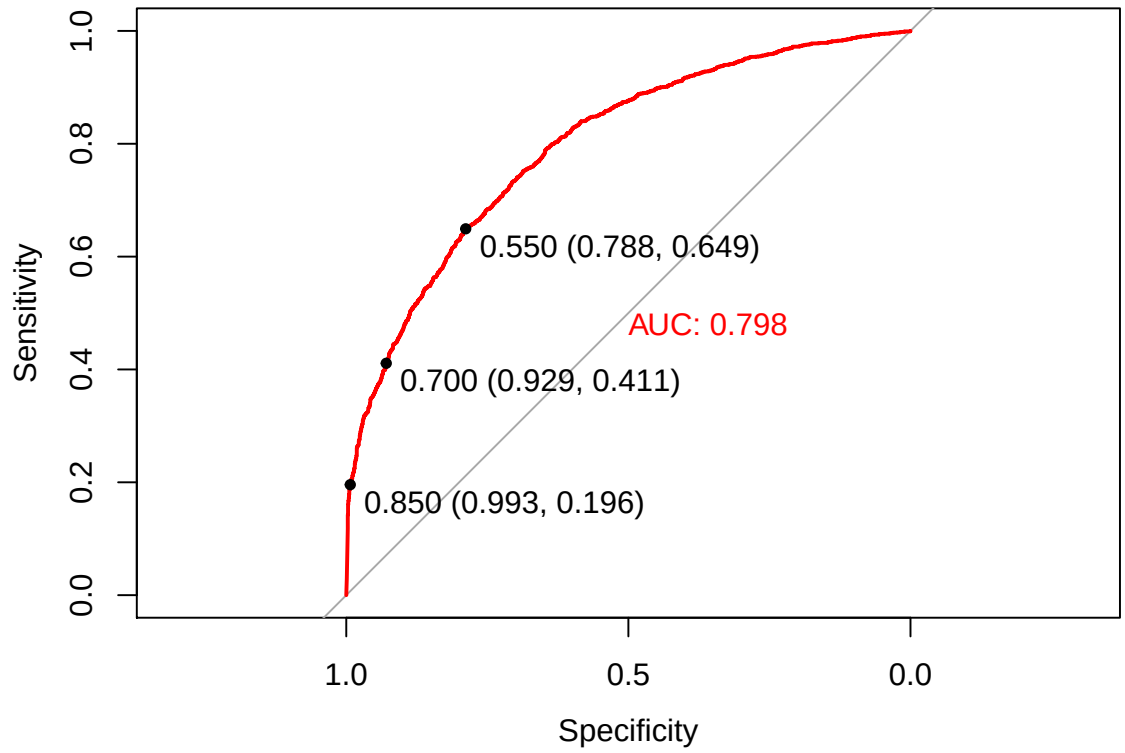
```
##     Min       1Q   Median       3Q      Max
## -2.7867  -0.9353   0.0595   0.9517   3.9540
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)           2.150958   0.259779   8.280  < 2e-16 ***
## distance            -16.200507   0.613346 -26.413  < 2e-16 ***
## shot_angle            0.003250   0.001697   1.916 0.055418 .
## situationFromCorner  -1.085173   0.246284  -4.406 1.05e-05 ***
## situationOpenPlay    -0.504981   0.223656  -2.258 0.023955 *
## situationPenalty      2.444188   0.473740   5.159 2.48e-07 ***
## situationSetPiece    -0.922232   0.267312  -3.450 0.000561 ***
## shotTypeLeftFoot      1.129875   0.115893   9.749  < 2e-16 ***
## shotTypeRightFoot     1.037684   0.107904   9.617  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6260.5  on 4515  degrees of freedom
## Residual deviance: 4906.4  on 4507  degrees of freedom
## AIC: 4924.4
##
## Number of Fisher Scoring iterations: 6
```

Chosing the threshold using ROC. We decided to go with higher specificity (true negative rate): 0.85.

```
invisible(plot(
  roc(trainShots$Class, fitted(model)),
  print.thres = c(0.55, 0.7, 0.85),
  col = 'red',
  print.auc = T))
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
# (specificity, sensitivity)
```

Testing the model

```
# creates predictions for the test data
testShots$pred <- predict(model, newdata = testShots, type = 'response')
```

The confusion matrix below shows how our model predicts the outcome of shots.

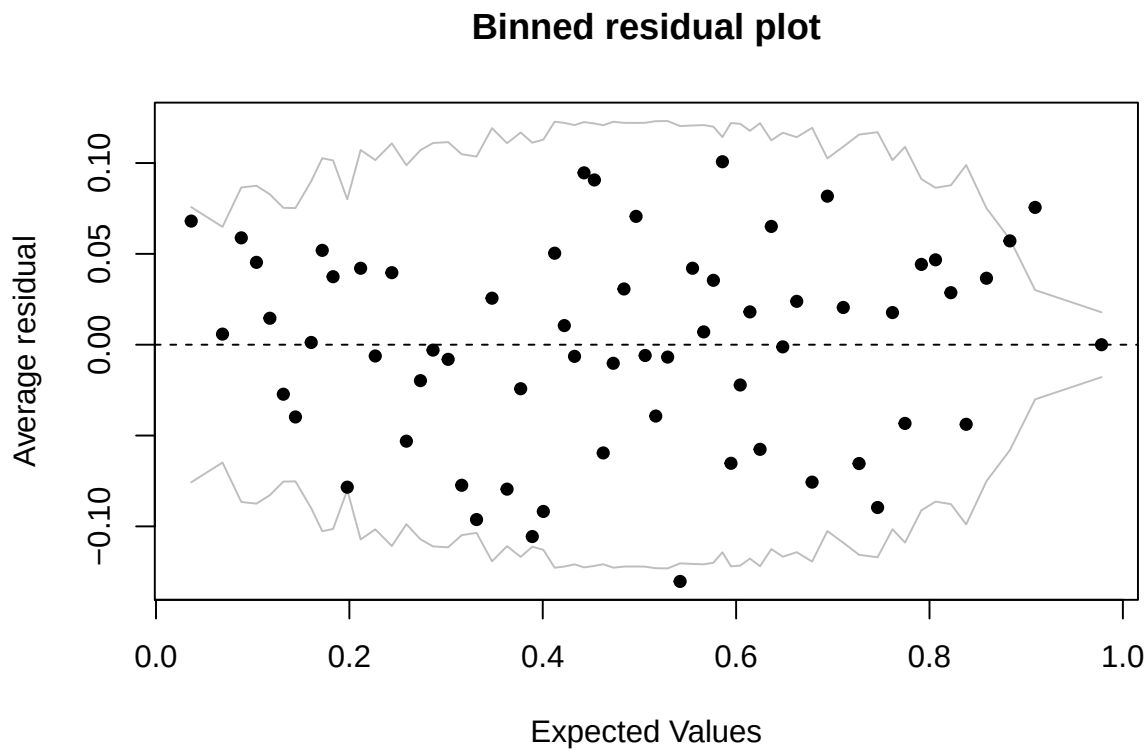
```
# confusionMatrix(factor(ifelse(testShots$pred > 0.55, 1, 0)), testShots$Class)$table
# confusionMatrix(factor(ifelse(testShots$pred > 0.7, 1, 0)), testShots$Class)$table
confusionMatrix(factor(ifelse(testShots$pred > 0.85, 1, 0)), testShots$Class)$table
```

```
##            Reference
## Prediction    0    1
##          0 7261  763
##          1   93  192
```

```
# tests prediction accuracy
y_pred_num <- ifelse(testShots$pred > 0.85, 1, 0)
y_pred <- factor(y_pred_num, levels = c(1, 0))
mean(y_pred == testShots$Class)
```

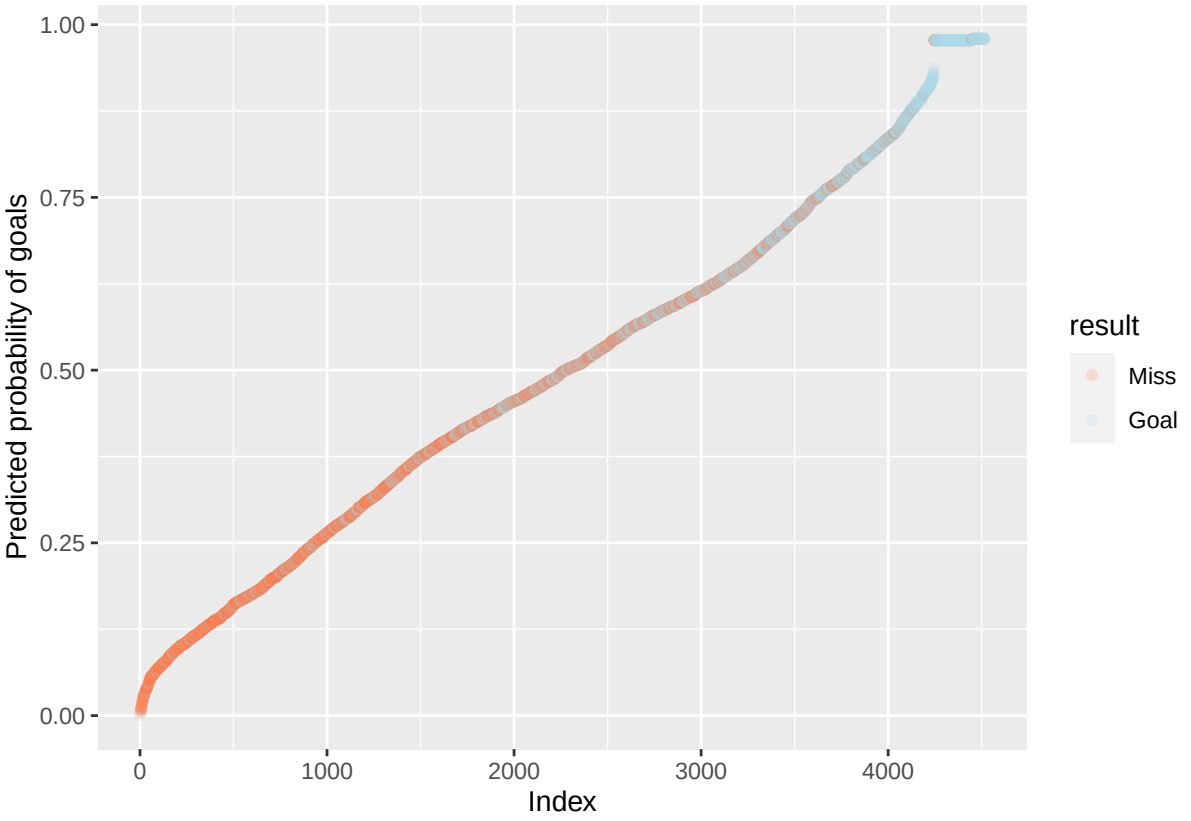
```
## [1] 0.8969792
```

```
binnedplot(fitted(model),
           residuals(model, type = "response"),
           nclass = NULL,
           xlab = "Expected Values",
           ylab = "Average residual",
           main = "Binned residual plot",
           col.int = "gray")
```



Plotting the prediction.

```
prediction <- data.frame(probability = model$fitted.values, result = trainShots$Class)

prediction <- prediction[order(prediction$probability, decreasing = FALSE), ]
prediction$rank <- 1:nrow(prediction)

prediction %>%
  ggplot(., aes(x = rank, y = probability)) +
  geom_point(aes(color = result), alpha = 0.2) +
  scale_color_manual(labels = c('Miss', 'Goal'), values = c('coral', 'lightblue')) +
  xlab('Index') +
  ylab('Predicted probability of goals')
```



Validation

We used shots from a different season (2019/2020) as validation dataset.

```
# create prediction
validationShots$pred <- predict(model, newdata = validationShots, type = 'response')

# test the accuracy of our prediction
y_pred_num <- ifelse(validationShots$pred > 0.85, 1, 0)
y_pred <- factor(y_pred_num, levels = c(1, 0))

mean(y_pred == validationShots$Class)
```

```
## [1] 0.8841456
```

Evaluation

The accuracy of the model

The fact that goals in football are rare events makes it much harder to predict the outcome of a shot than in any other sport. In our validation dataset we have 1095 goals and 7053 missed shots, which means that if we would predict that every shot is a miss, the accuracy of our prediction would be 86,5%. Compared to this, the accuracy of our model was 88,4%, therefore the improvement is only 1,9%.

The outcome of a shot in football is heavily influenced by randomness. The original idea behind the xG model was to create a metric, that describes the situation (the shot) more precisely. The same thing is true in our case: even if the model can not predict the outcome of a shot accurately, it can reveal dangerous shots.

To illustrate how xG works, we can compare the predictions on understat.com for this year's season in Premier League with the final results. Based on the xG metric, Brighton should be on the 5th place in the league, however they are 16th. This means that Brighton either wasted their chances or they were simply unlucky.

№	Team	M	W	D	L	G	GA
1	Manchester City	38	27	5	6	83	32
4	Chelsea	38	19	10	9	58	36
3	Liverpool	38	20	9	9	68	42
2	Manchester United	38	21	11	6	73	44
16	Brighton	38	9	14	15	40	46
5	Leicester	38	20	6	12	68	50
6	West Ham	38	19	8	11	62	47
8	Arsenal	38	18	7	13	55	39
11	Aston Villa	38	16	7	15	55	46
7	Tottenham	38	18	8	12	68	45
9	Leeds	38	18	5	15	62	54
10	Everton	38	17	8	13	47	48
13	Wolverhampton Wanderers	38	12	9	17	36	52
15	Southampton	38	12	7	19	47	68
12	Newcastle United	38	12	9	17	46	62
18	Fulham	38	5	13	20	27	53
17	Burnley	38	10	9	19	33	55
14	Crystal Palace	38	12	8	18	41	66
20	Sheffield United	38	7	2	29	20	63
19	West Bromwich Albion	38	5	11	22	35	76

Figure 1: Predictions on understat.com

This means that our model can be used mainly to show how dangerous a shot is, and not for predicting it's outcome.

Comparison with the xG model

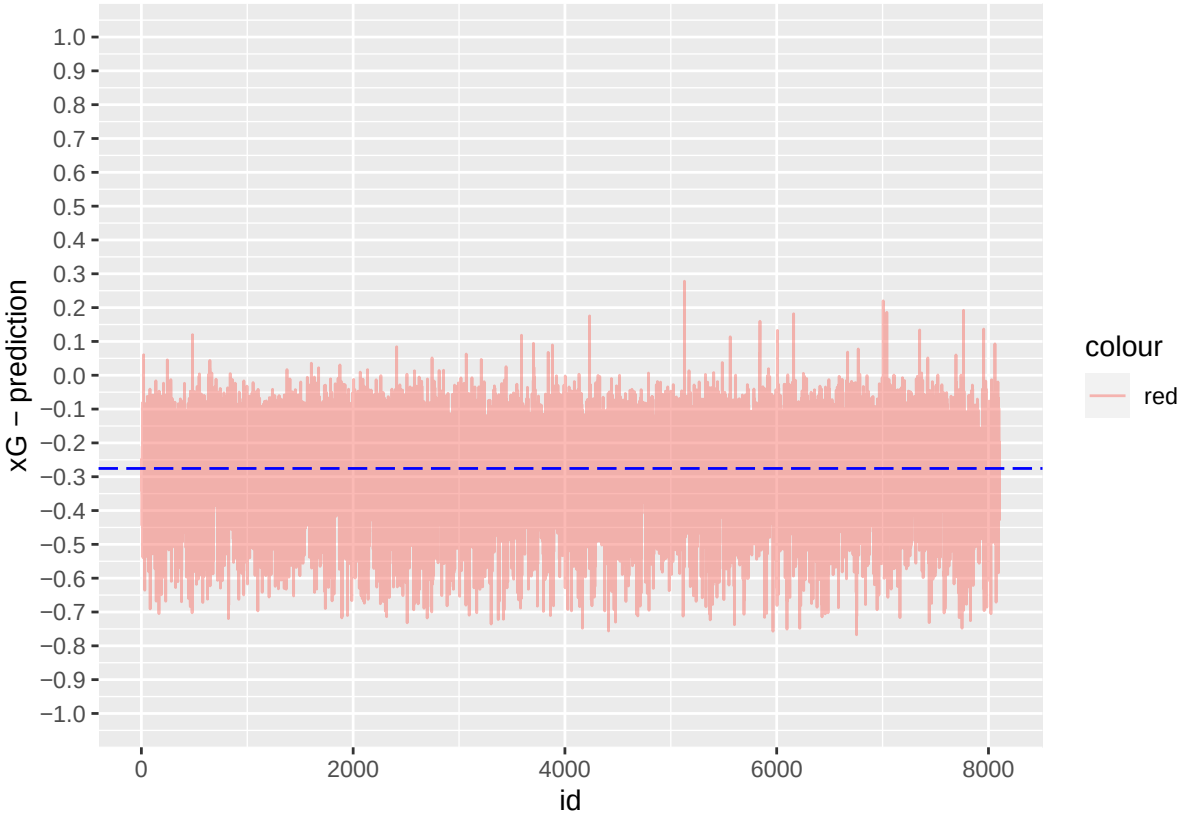
We can predict the outcome of shots in our validation dataset using the xG metric from Understat. For 0.5 threshold it shows 91,4% accuracy.

```
# test xG accuracy
y_xG_num <- ifelse(validationShots$xG > 0.5, 1, 0)
y_xG_pred <- factor(y_xG_num, levels = c(1, 0))
mean(y_xG_pred == validationShots$Class)
```

```
## [1] 0.914744
```

On the graph below we can see the difference between our model and the xG. We can see that our model overevaluates shots (with cca. 0.27) compared to the xG model.

```
validationShots <- validationShots %>% mutate(serial = row_number())

validationShots %>%
  ggplot(., aes(x = serial)) +
  geom_line(aes(y = xG - pred, color = 'red'), alpha = 0.5) +
  geom_hline(yintercept = mean(validationShots$xG - validationShots$pred),
             colour = 'blue',
             linetype ='longdash') +
  xlab("id") +
  ylab("xG - prediction") +
  scale_y_continuous(limits = c(-1, 1), breaks = seq(-1, 1, 0.1))
```

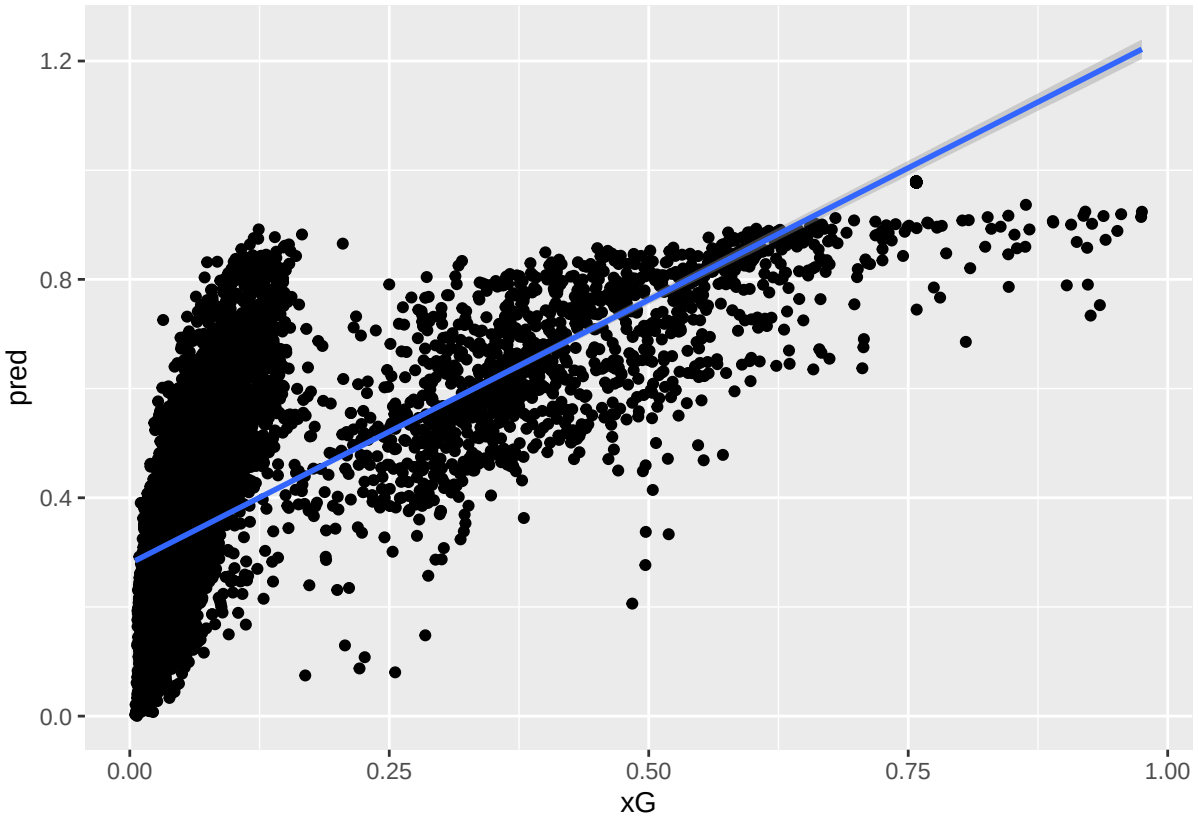


```
# mean(validationShots$xG - validationShots$pred)
```

The graphs below show the correlation between our prediction and Understat's xG. We can see that our model evaluates smaller chances (under 0.2 xG) slightly differently than Understat's model.

```
validationShots %>%
  ggplot(., aes(x = xG, y = pred)) +
  geom_point() +
  geom_smooth(method = lm)  # formula = y ~ poly(x, 2)
```

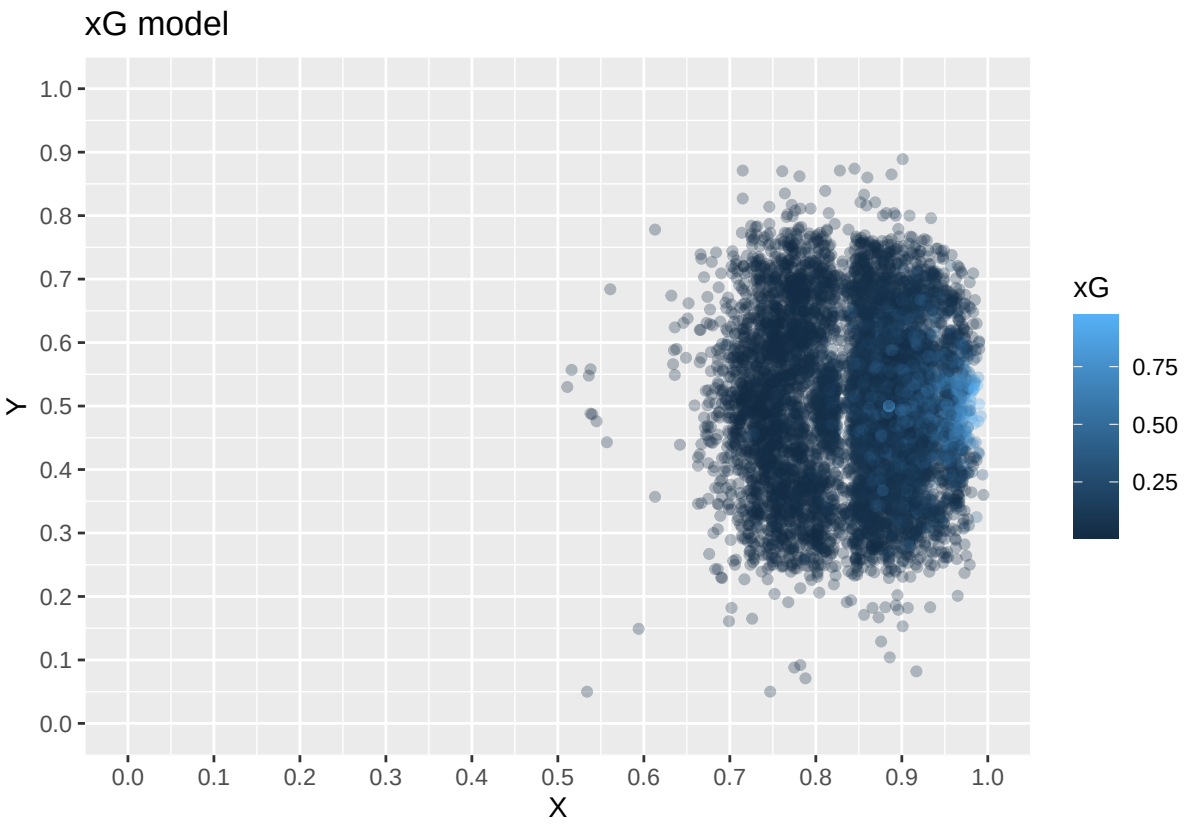
```
## `geom_smooth()` using formula 'y ~ x'
```



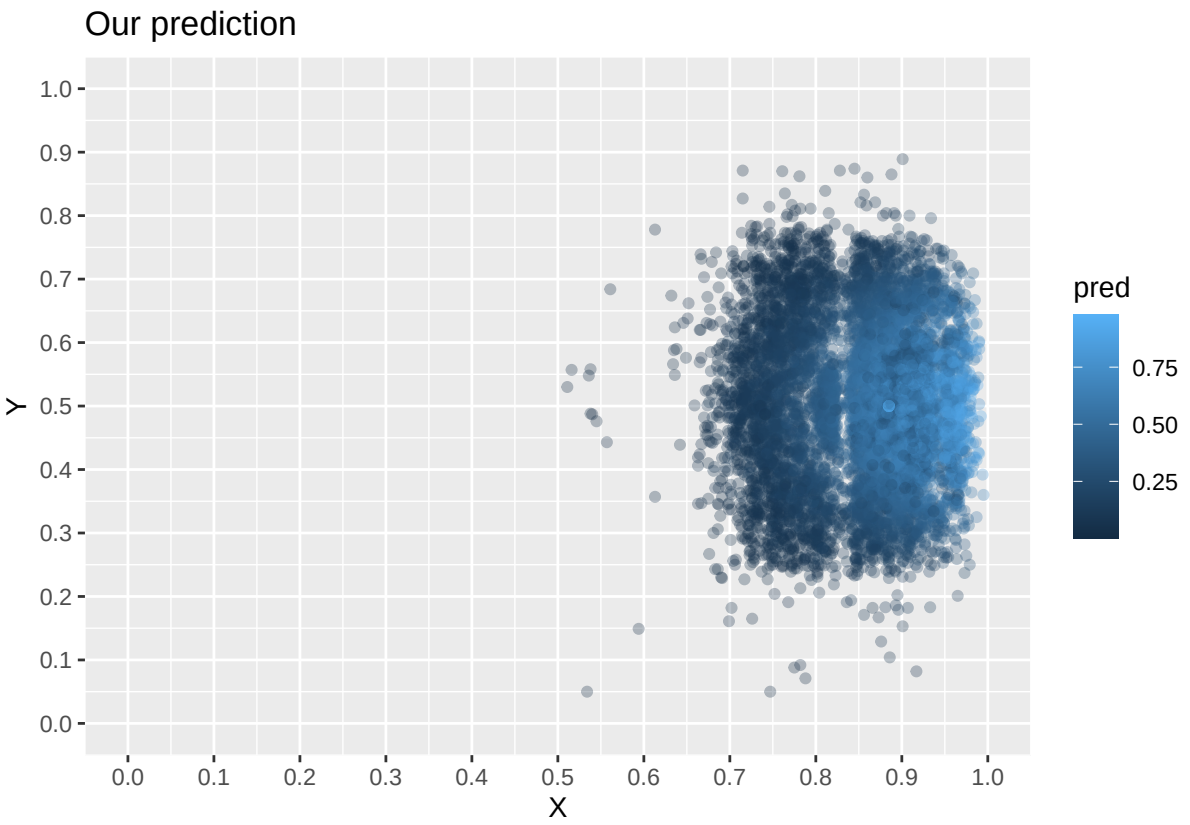

```
validationShots %>%
  mutate(corr = ifelse(xG < 0.2, 1, 2)) %>%
  ggplot(., aes(x = xG, y = pred, color = factor(corr))) +
  geom_point() +
  geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```


```
validationShots %>%
  ggplot(.) +
  geom_point(mapping = aes(x = X, y = Y, color = xG), alpha = 0.3) +
  scale_x_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  labs(title = 'xG model', color = 'xG')
```




```
validationShots %>%
  ggplot(.) +
  geom_point(mapping = aes(x = X, y = Y, color = pred), alpha = 0.3) +
  scale_x_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  scale_y_continuous(limits = c(0, 1), breaks = seq(0, 1, 0.1)) +
  labs(title = 'Our prediction', color = 'pred')
```



Different xG models (eg. Opta's xG, InfoGoal's xG) use different methods, therefore it is not unusual to get different predictions.

Our dataset does not provide deeper understanding of the situation. Other xG models use more precise data that contains more detailed information, such as the speed of the shot or the number of defensive players between the ball and the goal. This might be one of the reasons why our model over overevaluates some shots, compared to Understat's xG.

5 Conclusion

In the first part of our project, we examined how footedness affects the player's performance. Our hypothesis was, that players who can use both left and right legs are performing better. We rejected this hypothesis, because our analysis showed, that players that are strong with their left or right foot are performing much better. For the first part of the project was responsible Dávid Varga.

In our second hypothesis, we assumed that the outcome of a shot is heavily affected by the shot distance and shot angle. While the outcome of shots in football is greatly influenced by random moments, we concluded that this statement is true. For this part of the project was responsible Márton Németh.

Since our dataset contains mainly continuous variables, and the categorical variables are not related to each other, we could not use Bayesian network in our project.