Predicting football results of the german bundesliga

HarvardX project submission
PH125.9x Data Science
-
Capstone "CYO"

Author: Martin Haitzmann[*]

Last updated September 20, 2020

Contents

1 Overview

Football (commonly used wording in Europe, in the US rather referred to as Soccer)[1] is one of the worlds most popular sports games. It is a low scoring game, where two opposing teams, each consisting of 11 team members (10 players and one goalkeeper), try to score goals in 90 minutes of playing time (two half's, each 45 minutes with a 15 minute break in between). In case of knockout games, like in cup competitions, the 90 minutes can be extended to another 2x15 minutes of playing time and possible penalty shootout to get a winner. In common national league competitions, however, ties resp. draws[2] are also possible. National league competitions in Europe are organized in seasons, usually lasting from September to Mai (some leagues have a one month winter break in January).

Predicting the results of games is a common hobby of fans, some of them also trying to earn extra money with bets. Thus, it is about predicting individual games results, but also e.g. giving bets on who will be the next winner of the complete league competition. These questions became of special

[*] martin.c.haitzmann@gmail.com

[1] https://en.wikipedia.org/wiki/Association_football

[2] both teams scoring the same amount of goals after 90 minutes

importance in the most recent season (2019/2020) for another reason. Almost all European league competitions needed to be suspended in the middle of march 2020 due to COVID-19 pandemic related lock-downs. Therefore, at that point in time, the season was roughly spoken to an extent of 2/3 finished. With the open question in April and May if leagues could be continued, it was not any more only about the pure personal interest, who would have won the league. Rather, as the final standings of the leagues also have impacts on future competitions (teams get relegated/promoted or are allowed to participate in further competitions the subsequent season, like the European champions league), it became a question of more common interest. The more, as such competitions have a significant impact on the participating clubs income and prestige as well.

> So, who would have won the league? Who would have made it in the group of teams qualified for the European champions league competition? Who would have needed to relegate?

Very interesting questions from a data scientist's and sports fan's point of view! This report tries to find answers to these questions focusing on the German bundesliga. The future remains unpredictable, as we cannot really model a competition that never existed that way before (matches without viewers in the stadium after a complete lock-down). However, it is about analyzing and developing ways on how to predict results of soccer game. Exciting enough!

The approach was to first *find suitable data* on past match results. Kaggle offered some interesting soccer related data-sets, partly with plenty of features regarding team and player performances, but the analysis here required very recent data. Furthermore, as this report should only be the starting point for the authors personal interest in analyzing and predicting soccer games, regularly updated data sources were searched. Such a source was found in openfootball. It seems to be a nicely organized open access project with some contributors. Having decided for this data source, an exploratory analysis of the data revealed for the german bundesliga only results of past matches to be available. There are tables in the SQLLite database that would refer to players as well, but unfortunately these tables were not filled with information. However, as the database structure is there, the data will perhaps be extended at some point in the future. Perhaps also an opportunity then to bring this project to the next step . ***Long story short, soccer predictions based solely on past results is the main focus in this analysis.***

During *data exploration and model development* two possibilities of possible predictions became clearer: It seemed obvious from the beginning that one could not only predict the expected league ranks, but that it would rather be necessary to predict the results of the remaining matches and to construct the final table then. It turned out that predicting the results only (team1 wins, draw, team2 wins) using e.g. classification trees leads to the successful building of a final table, but in the end there is the theoretical possibility that ranks in the final table can be decided by scored and received goals in case of teams with the same amount of points. Thus, the possibility of simulating the scores of each game were explored as well, mainly based on the idea, that football scores might be modeled with Poisson distribution, as pointed out by several analysis in the past[3].

Although this report successfully built a model for predicting soccer games, luckily the leagues were able to finish with games without spectators and the answers to the questions were found in reality and not by simulations.

2 Analysis and methods

This project starts as a one time analysis for the HavardX data science course, but the personal intention is to further dig into that topic. For that reason the preference was not to build the analysis on a once uploaded data-set like the ones in Kaggle, but to rely on data that are regularly updated. openfootball seemed to be very promising in that respect. It is a privately initiated project offering open public domain football data-sets in a well defined data structure.

The relevant data regarding the german bundesliga is downloaded from openfootball as a SQLLite database. The database contains different tables for a flexible use of data. In general, a table structure

[3] Maher (1982) and Dixon, Coles (1997) were among those preparing the statistical analysis of soccer predictions. The ideas are also nicely summarized in the blogs of Sheehan (2018) and Hickman (2019)

Tab. 1: Available variables for analysis.

rel_vars	rel_vars_txt
team1	home
team2	away
score1	goals home team
score2	goals away team
round	round
season	season
date	date game played
winner	winner

as shown in Fig. 1 is available.

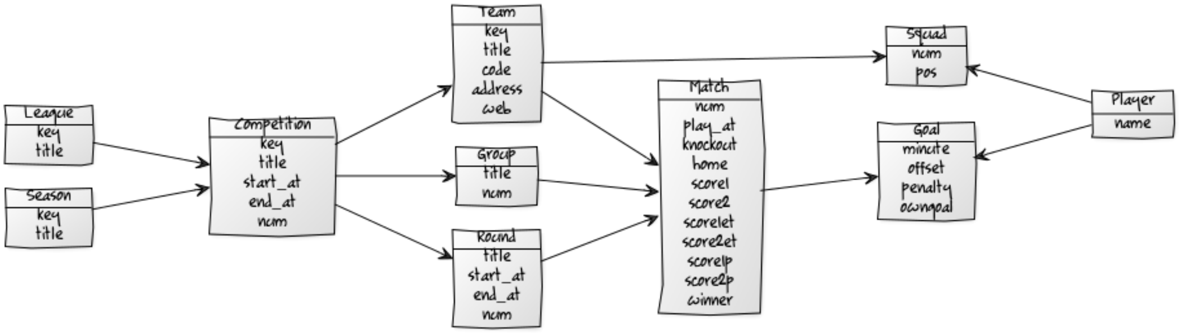


Fig. 1: Database structure as published by openfootball on github

An SQL query joining the different relevant pieces of information together finally leads to a single data.frame with extracted match results containing seasons from 2000 on-wards. To be more precise, in each season 306 matches need to be played.

```
##
## 2000/01 2001/02 2002/03 2003/04 2004/05 2005/06 2006/07 2007/08 2008/09 2009/10
##     306     306     306     306     306     306     306     306     306     306
## 2010/11 2011/12 2012/13 2013/14 2014/15 2015/16 2016/17 2017/18 2018/19 2019/20
##     306     306     306     306     306     306     306     306     306     306
```

18 teams compete in home and away games, resulting in 9 games per round. Altogether 34 rounds are necessary, to complete the season (every team competes against 17 others, thus 17*2 = 34).

As already outlined, openfootball provides information on the result of each game. The variables are shortly described in Tab. 1.

```
## # A tibble: 6 x 8
##   team1      team2       score1 score2 round season  date       winner
##   <chr>      <chr>        <dbl>  <dbl> <dbl> <chr>   <date>      <dbl>
## 1 dortmund   rostock          1      0     1 2000/01 2000-08-11      1
## 2 bayern     herthabsc        4      1     1 2000/01 2000-08-12      1
## 3 freiburg   stuttgart        4      0     1 2000/01 2000-08-12      1
## 4 hsv        tsvmuenchen      2      2     1 2000/01 2000-08-12      0
## 5 klautern   bochum           0      1     1 2000/01 2000-08-12      2
## 6 leverkusen wolfsburg        2      0     1 2000/01 2000-08-12      1
```

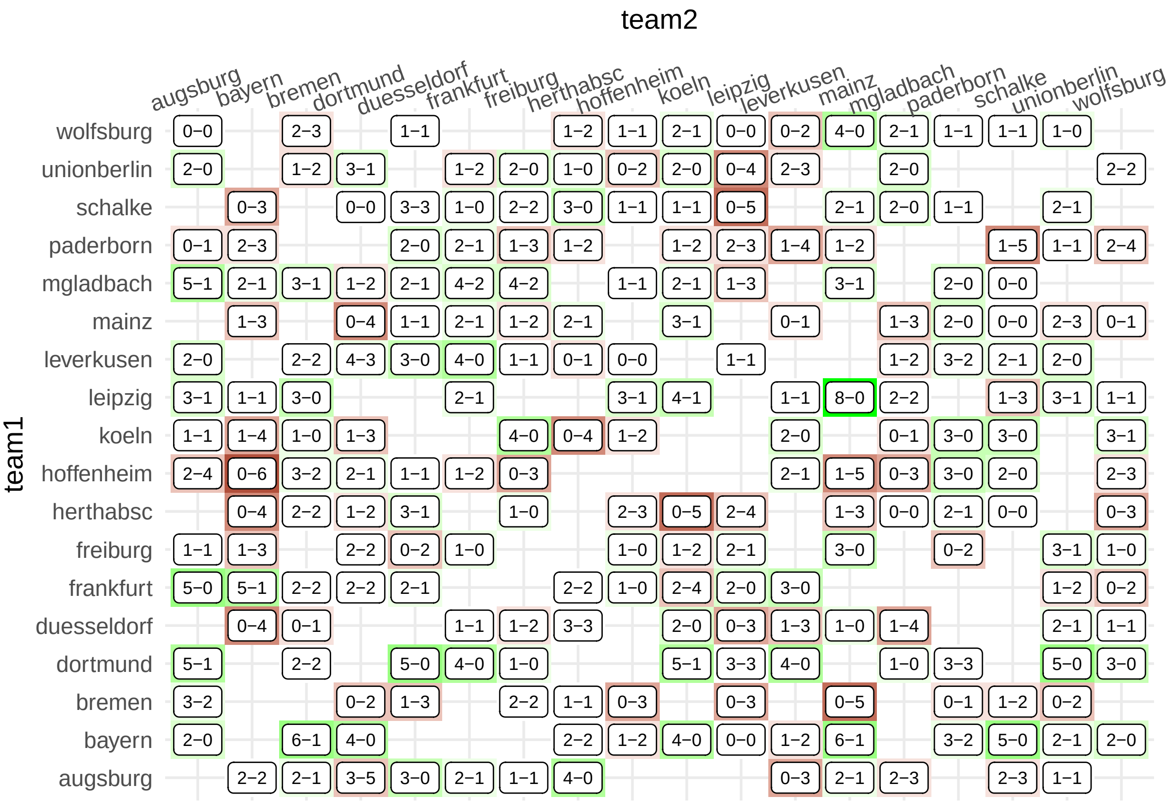
Unfortunately openfootball does not provide other than results information (shots on goals statistic, or even players belonging to teams, . . .) regarding the german bundesliga, although the table structure would principally foresee that; thus, for this analysis, the model to predict results needs to be solely based on information from past games. This will be a good start into the topic of soccer predictions anyways, as there is a need to construct team performance indicators out of the available data only. For the future the basis is then hopefully built to extend the model with additional features, either by combining with other data sources or ideally being at some pint in the future available in the openfootball tables.

2.1 Explore the data

To recapitulate from the introduction, the goal of this report is to predict the rest of the most recent season 2019/20, as the COVID-19 related lock-down lead to a suspension of the league. At the time when the report was written, it was already clear, that the league had luckily been finished. Without spectators, but with real sports.

2.1.1 Season 2019/20 and the COVID related lockdown

As mentioned, a part of the season 2019/20 was already played before the lock-down. Fig. 2 provides an overview of the results.



team2

team1	augsburg	bayern	bremen	dortmund	duesseldorf	frankfurt	freiburg	herthabsc	hoffenheim	koeln	leipzig	leverkusen	mainz	mgladbach	paderborn	schalke	unionberlin	wolfsburg
wolfsburg	0–0		2–3		1–1			1–2	1–1	2–1	0–0	0–2	4–0	2–1	1–1	1–1	1–0	
unionberlin	2–0		1–2	3–1		1–2	2–0	1–0	0–2	2–0	0–4	2–3		2–0				2–2
schalke		0–3		0–0	3–3	1–0	2–2	3–0	1–1	1–1	0–5		2–1	2–0	1–1		2–1	
paderborn	0–1	2–3			2–0	2–1	1–3	1–2		1–2	2–3	1–4	1–2			1–5	1–1	2–4
mgladbach	5–1	2–1	3–1	1–2	2–1	4–2	4–2		1–1	2–1	1–3		3–1		2–0	0–0		
mainz		1–3		0–4	1–1	2–1	1–2	2–1		3–1		0–1		1–3	2–0	0–0	2–3	0–1
leverkusen	2–0		2–2	4–3	3–0	4–0	1–1	0–1	0–0		1–1			1–2	3–2	2–1	2–0	
leipzig	3–1	1–1	3–0			2–1			3–1	4–1		1–1	8–0	2–2		1–3	3–1	1–1
koeln	1–1	1–4	1–0	1–3			4–0	0–4	1–2			2–0		0–1	3–0	3–0		3–1
hoffenheim	2–4	0–6	3–2	2–1	1–1	1–2	0–3					2–1	1–5	0–3	3–0	2–0		2–3
herthabsc		0–4	2–2	1–2	3–1		1–0		2–3	0–5	2–4		1–3	0–0	2–1	0–0		0–3
freiburg	1–1	1–3		2–2	0–2	1–0			1–0	1–2	2–1		3–0		0–2		3–1	1–0
frankfurt	5–0	5–1	2–2	2–2	2–1			2–2	1–0	2–4	2–0	3–0					1–2	0–2
duesseldorf		0–4	0–1			1–1	1–2	3–3		2–0	0–3	1–3	1–0	1–4			2–1	1–1
dortmund	5–1		2–2		5–0	4–0	1–0			5–1	3–3	4–0		1–0	3–3		5–0	3–0
bremen	3–2			0–2	1–3		2–2	1–1	0–3		0–3		0–5		0–1	1–2	0–2	
bayern	2–0		6–1	4–0				2–2	1–2	4–0	0–0	1–2	6–1		3–2	5–0	2–1	2–0
augsburg		2–2	2–1	3–5	3–0	2–1	1–1	4–0				0–3	2–1	2–3		2–3	1–1	

Fig. 2: Fixtures already played in relevant season

The standings at the time of the suspension of the league on 2020-03-13 can be seen in Tab. 2[4]:

We see from the table (variable: games_played) that the league was actually suspended at the 25th round. The date of the announcement of the suspension was around 2020-03-13. Only frankfurt and bremen had 1 game less, meaning that these two teams have only played 24 rounds at the time of the lock-down.

During the lock-down the remaining fixtures were rescheduled to dates in Mai and June, as shown in Tab. 3[5].

[4] every win leads to 3 points for the winning team and 0 points for the losing team; a draw results in 1 point for each of the two teams

[5] the games needed to be played without the support of visitors

Tab. 2: League standings at time of league suspension.

team	games_played	wins	draws	lost	g_scored	g_received	g_diff	points
bayern	25	17	4	4	73	26	47	55
dortmund	25	15	6	4	68	33	35	51
leipzig	25	14	8	3	62	26	36	50
mgladbach	25	15	4	6	49	30	19	49
leverkusen	25	14	5	6	45	30	15	47
schalke	25	9	10	6	33	36	-3	37
wolfsburg	25	9	9	7	34	30	4	36
freiburg	25	10	6	9	34	35	-1	36
hoffenheim	25	10	5	10	35	43	-8	35
koeln	25	10	2	13	39	45	-6	32
unionberlin	25	9	3	13	32	41	-9	30
frankfurt	24	8	4	12	38	41	-3	28
herthabsc	25	7	7	11	32	48	-16	28
augsburg	25	7	6	12	36	52	-16	27
mainz	25	8	2	15	34	53	-19	26
duesseldorf	25	5	7	13	27	50	-23	22
bremen	24	4	6	14	27	55	-28	18
paderborn	25	4	4	17	30	54	-24	16

Tab. 3: Rescheduled fixtures after lock-down.

	24	26	27	28	29	30	31	32	33	34	Sum
2020-05-16	0	6	0	0	0	0	0	0	0	0	6
2020-05-17	0	2	0	0	0	0	0	0	0	0	2
2020-05-18	0	1	0	0	0	0	0	0	0	0	1
2020-05-22	0	0	1	0	0	0	0	0	0	0	1
2020-05-23	0	0	5	0	0	0	0	0	0	0	5
2020-05-24	0	0	3	0	0	0	0	0	0	0	3
2020-05-26	0	0	0	4	0	0	0	0	0	0	4
2020-05-27	0	0	0	5	0	0	0	0	0	0	5
2020-05-29	0	0	0	0	1	0	0	0	0	0	1
2020-05-30	0	0	0	0	5	0	0	0	0	0	5
2020-05-31	0	0	0	0	2	0	0	0	0	0	2
2020-06-01	0	0	0	0	1	0	0	0	0	0	1
2020-06-03	1	0	0	0	0	0	0	0	0	0	1
2020-06-05	0	0	0	0	0	1	0	0	0	0	1
2020-06-06	0	0	0	0	0	5	0	0	0	0	5
2020-06-07	0	0	0	0	0	3	0	0	0	0	3
2020-06-12	0	0	0	0	0	0	1	0	0	0	1
2020-06-13	0	0	0	0	0	0	6	0	0	0	6
2020-06-14	0	0	0	0	0	0	2	0	0	0	2
2020-06-16	0	0	0	0	0	0	0	4	0	0	4
2020-06-17	0	0	0	0	0	0	0	5	0	0	5
2020-06-20	0	0	0	0	0	0	0	0	9	0	9
2020-06-27	0	0	0	0	0	0	0	0	0	9	9
Sum	1	9	9	9	9	9	9	9	9	9	82

All in all, it is necessary to predict the outcomes of 82 games while 224 were already played in the season.

2.1.2 Historic data

In this context of the exploratory data analysis, historic data are supposed to be all available results until the lock-down resp. The date when the suspension of the league competition was announced (2020-03-13).

2.1.2.1 Results The three possible results of the outcome of one football match are:

```
##   num      char                           desc
## 1   0      draw Draw (both teams same score)
## 2   1 win_team1        Team1 wins (home win)
## 3   2 win_team2        Team2 wins (away win)
```

Over some seasons the observed probabilities of the results are:

```
##       season
## winner   2013/14   2014/15   2015/16   2016/17   2017/18   2018/19   2019/20
##      0 0.2091503 0.2679739 0.2320261 0.2418301 0.2712418 0.2385621 0.2187500
##      1 0.4738562 0.4738562 0.4411765 0.4901961 0.4542484 0.4509804 0.4330357
##      2 0.3169935 0.2581699 0.3267974 0.2679739 0.2745098 0.3104575 0.3482143
```

All in all, a draw is less likely than a win for one of the opposing teams. The more, it is evident, that team1 (the home team) was in each season likely to win more more often according to the descriptive statistics. This “home-advantage” is a common known effect in soccer games and is discussed a bit later.

Coming back to the distribution of past results, one very basic model could therefore introduce a simple guess for each game’s result according to these prevalent probabilities. However, there is need to obey a further piece of information, as we want to predict after-lock-down-results resp. results after the 25th round of a season. It should be noted that the occurrences for the possible outcomes differ if the series is split at the 25th game-week:

```
## 
## up to 25th round:
```

```
##       season
## winner   2013/14   2014/15   2015/16   2016/17   2017/18   2018/19   2019/20
##      0 0.2266667 0.2711111 0.2400000 0.2311111 0.2933333 0.2444444 0.2187500
##      1 0.4444444 0.4666667 0.4266667 0.4888889 0.4266667 0.4400000 0.4330357
##      2 0.3288889 0.2622222 0.3333333 0.2800000 0.2800000 0.3155556 0.3482143
```

```
## 
## after 25th round:
```

```
##       season
## winner   2013/14   2014/15   2015/16   2016/17   2017/18   2018/19
##      0 0.1604938 0.2592593 0.2098765 0.2716049 0.2098765 0.2222222
##      1 0.5555556 0.4938272 0.4814815 0.4938272 0.5308642 0.4814815
##      2 0.2839506 0.2469136 0.3086420 0.2345679 0.2592593 0.2962963
```

The last part of the season (after 25th round) showed fewer draws and more wins of team1 (home win) regarding the most recent seasons. This makes also sense from a theoretical point of view. In the last phase of each league competition it is about the final position leading to relegation, championship and other consequences. Thus plenty of teams are encouraged to really win their games, because with a win one earns 3 points, while a draw only is worth one point.

A basic model to randomly predict the result according to the observed probability of past occurrences of a win of the home team, a draw or a win of the away team, should therefore be based on the part of past seasons that reflects the league finish (~round 25 of relevant past season).

2.1.2.2 Teams Fig. 3 provides an overview on the composition of the league. The color of the tiles reveals the points each team could gain throughout each season. Remember, in german bundesliga a team earns 3 points with a win, 1 point with a draw and 0 points if the opponent wins.

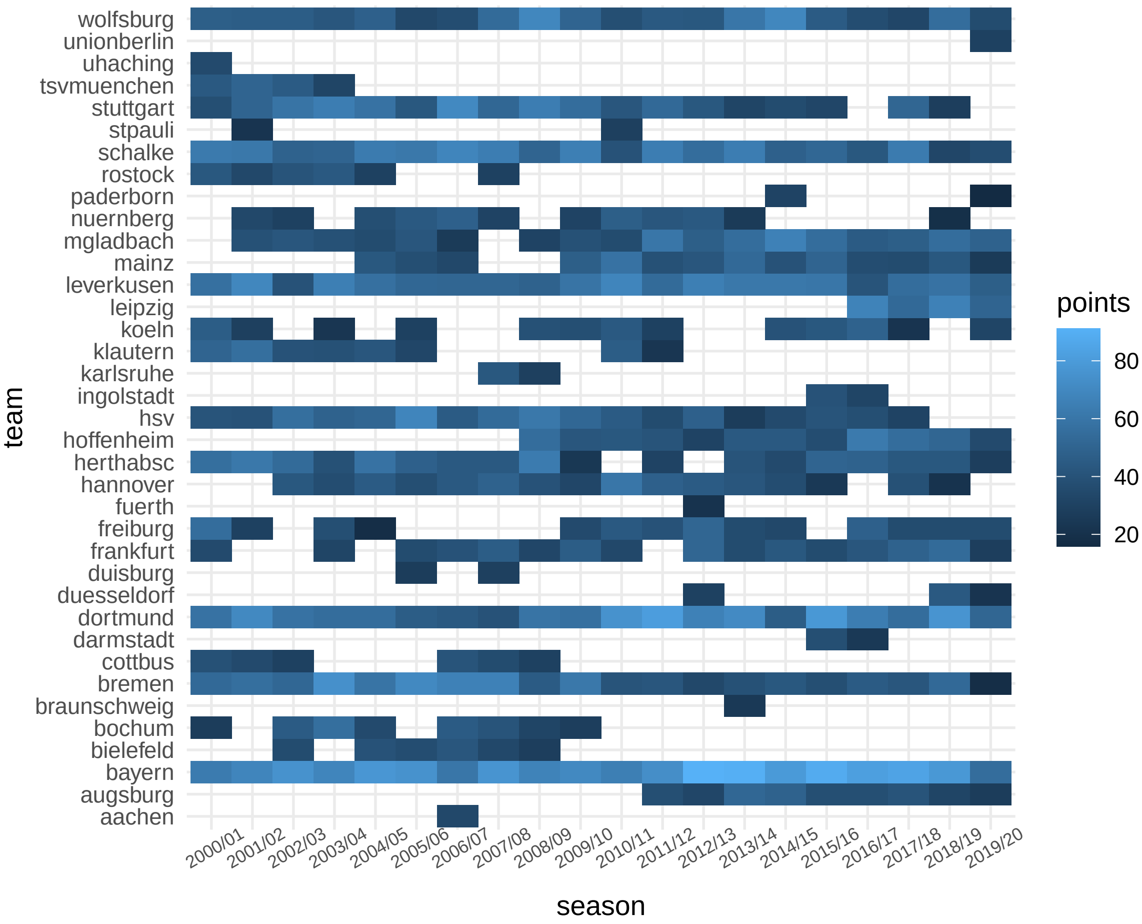


Fig. 3: Team perfromance throughout past seasons.

It is evident that not always the same teams are part of the league in each season. This is quite different to U.S. professional sports leagues with its license and franchise system, whereas promotion and relegation is very common in European league competition.

While some teams like bayern and dortmund were consistently part of the first german league since 2000, unionberlin, paderborn and koeln were promoted from the 2nd bundesliga to the 1st league in the season 2019/20, whereas hannover, nuernberg and stuttgart were relegated. This fluctuation of course hampers the possibilities of taking the results of past seasons into consideration for model building. However, each season might also be influenced by transfers, . . . so the results in the current season should be of main interest anyway, as this seem to be the best proxy for each teams general ability to perform in the season 2019/20. However, some teams like especially bayern and dortmund consistently performed very well during the last years, meaning that these are teams with constant success. Constant success of course makes it easier to buy good players, . . . , whereas not so good teams always have the problem of facing a loss in income in case they get relegated. Such clubs are of course not primary target for star players, e.g. such players having proven to perform very well over several seasons.

It is to find a compromise regarding model building and the inclusion of past seasons. In order not to lose too much information and acknowledging the fact that teams performance is not likely to change considerably on broad base, not the whole time series since 2000, but rather the last few seasons are kept for model building.

In order to build a suitable model one needs to know about the strength of each team. Due to a lack of other data, one of the only remaining possibilities is to extract information on the strength based on past results.

2.1.2.3 Scores/Goals Results are a consequence of scores resp. goals. In football it is desirable to score many goals and receive only a few. In terms of a team performance indicator it would be good to have a high mean in scored goals and a low mean value of received goals. Transferred to a graph, it is desirable to be at the right lower corner.

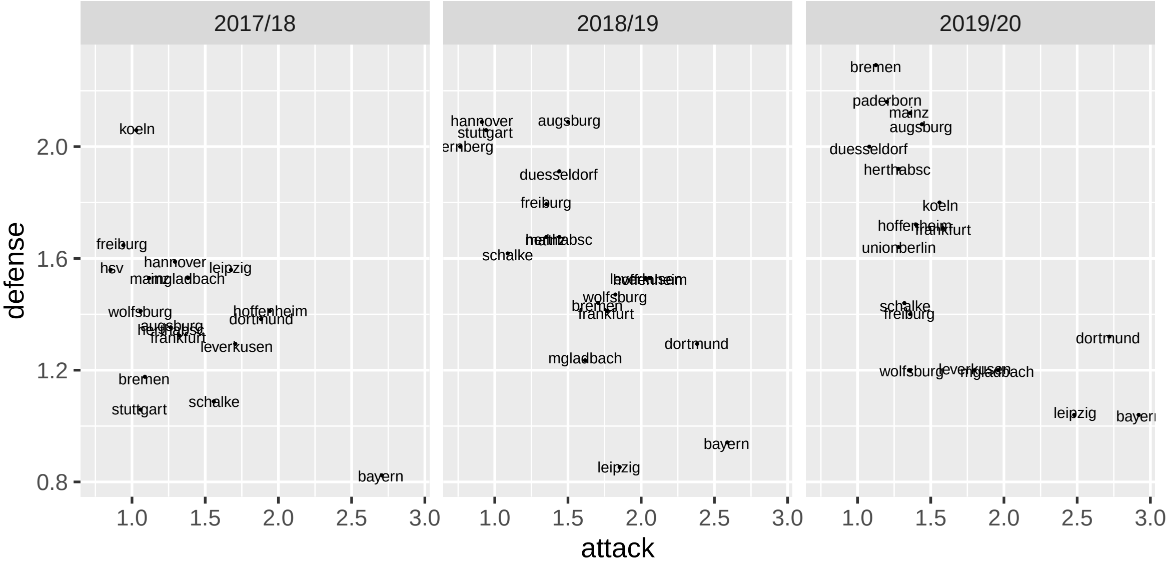


Fig. 4: Attack and defense score throughout past seasons.

In the season 2017/18 *bayern* did a very great job, scoring on average more than 2.5 goals per game over the season. The defense parameter with a value of about 0.8 received goals per game on the other hand was very low. So far, in the season 2019/20 (remember only 25 rounds played) *bayern* has as well the best combination of attack/defense score. **Attack and defense scores** give important insights in the ability of teams and should be further investigated in model building.

Another intrinsic property of scoring goals in football needs attention. The number of goals a team scores is a discrete value. Taking the results from the most recent seasons, a few interesting properties of the data become clearer.

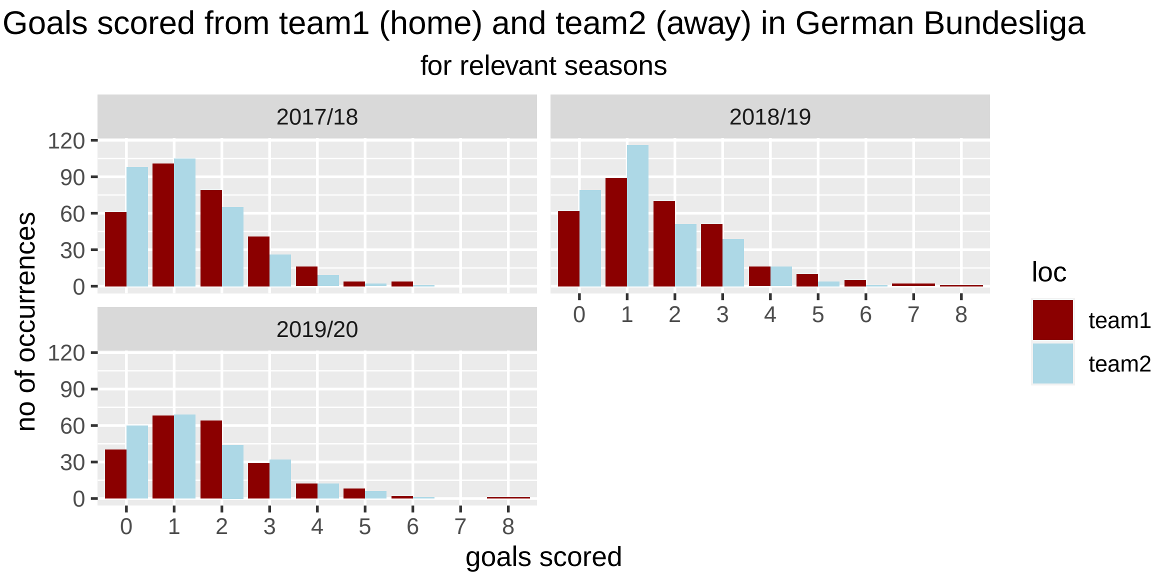


Fig. 5: Goals scored per home and away team in resent seeasons.

The maximum number of scored goals for one team was 8. This was achieved by a “team1” in the season 2018/19 as well as in season 2019/20[6]. Generally, team1 tends to to score a higher number of goals than team 2. This can be attributed to the so-called home-advantage

2.1.2.4 Home advantage It is common knowledge among football fans that there is some sort of an advantage for the home team, may it be because the players are more motivated, need not to travel to another city, do not get intimidated by the atmosphere, are backed by their own fans. . .

The effect becomes more visible after slightly changing the above illustration to show an overall density plot.

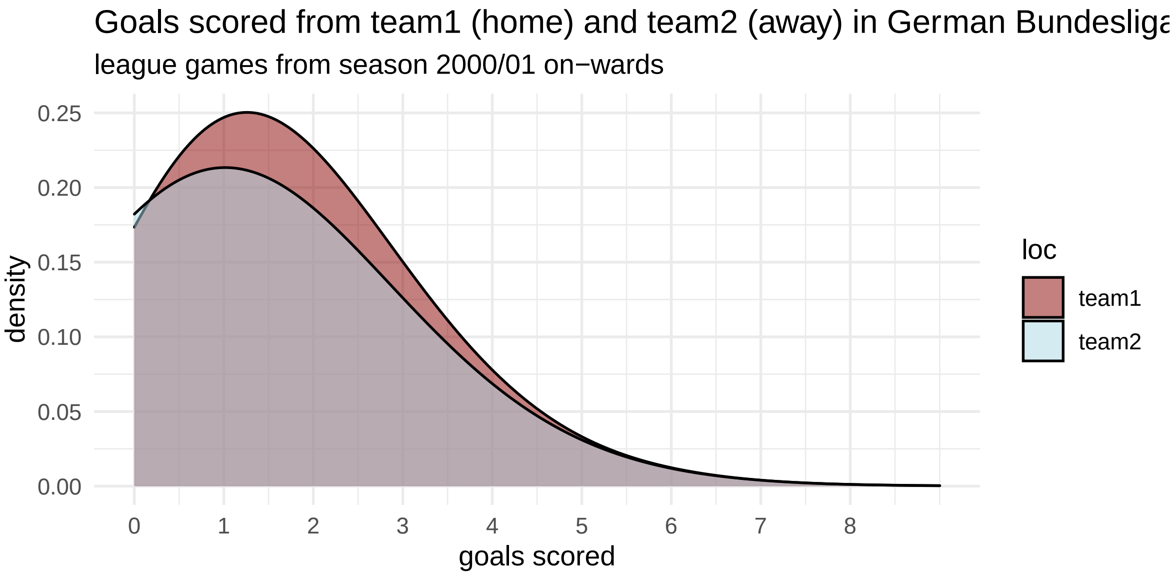


Fig. 6: Density plot for goals scored.

[6] be aware we are still assuming that we do not know the outcomes of that specific season for games after the lock-down. This is also the reason why the no of occurrences plotted on the y-axis are a bit lower.

To quantify the home advantage effect we can simply calculate a ratio of ***all goals scored form team1 / all goals scored form team2*** over the relevant seasons. Regarding the seasons from 2000/01 on-wards, this general effect results in an advantage of:

```
## [1] 1.41357
```

Home teams score on average 1.41 goals more.

The above density plot further reveals similarity to the Poisson distribution. Thanks to the work of e.g Maher (1982) and Dixon, Coles (1997) one can use the quite commonly used approximation for soccer games, that scores follow a Poisson distribution.

2.1.2.5 Scores and the Poisson distribution

The "Poisson distribution, named after French mathematician Siméon Denis Poisson, is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time or space if these events occur with a known constant mean rate and independently of the time since the last event."[7].

In the context of the current analysis, the number of goals is expressed as a function of an average rate of goals. Furthermore goals occur in a fixed interval of time (90 minutes) and are independent of each other, meaning that they do no become more/less likely by the number of goals already scored in the match. Regarding this report, we consider this assumption to be sufficient, although there is also doubt on the suitability of a Poisson distribution to model soccer goals[8]

The mathematical representation of the Poisson distribution is

$$P(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Sheehan (2018) and Hickman (2019) did a great job in their blogs explaining the details on how this distribution is useful for predicting soccer scores, being based itself on scientific work mentioned above ((Maher 1982) and (Dixon, Coles 1997)).

In short, the number of goals scored for the opposing teams in a match are assumed to follow a Poisson distribution and can therefore be predicted with a specific probability. The abilities of different teams to perform is based on information of past games, the parameter λ can be used to incorporate team parameters like attacking strength and defense capabilities (calculated by the mean values of scored and received goals per each team). λ might be further specified as a term like $\lambda = \alpha * \beta * \gamma$, with α representing attacking capabilities, β defense skills and γ the home advantage.

Dixon, Coles (1997) further developed such a basic Poisson model and introduced an interaction term to correct underestimated frequency of low scoring matches and applied time decay component so that recent matches are weighted more strongly.

2.2 Develop models

Regarding model development, it is assumed here to have the knowledge at the time of the lock-down. Thus the train and test set are defined in a way that all games after the 25th game-week in the season 2018/19[9] need to be predicted.

- test_set: all games after round 25 of season 2018/19
- train_set: all games up to round 25 in season 2018/19 and all past games from seasons 2015/16, 2016/17, 2017/18, 2018/19.

The predictions need to result in values of 0, 1 and 2, referring to

```
##   num     char                        desc
## 1   0     draw Draw (both teams same score)
```

[7] see https://en.wikipedia.org/w/index.php?title=Poisson_distribution#cite_note-Haight1967-1

[8] see Greenhough, Birch, Chapman, Rowlands (2002) and also the conclusion of David Sheehan's blog on Dixon-Coles processes (Sheehan 2018).

[9] and NOT the season 2019/20

```
## 2   1 win_team1        Team1 wins (home win)
## 3   2 win_team2        Team2 wins (away win)
```

2.2.1 Guessing results based on past distribution

The exploratory analysis revealed to use probabilities specifically after the 25th game-week, as there tend to be fewer draws. The average chance of results to occur based on the results for the seasons 2015/16, 2016/17, 2017/18 results in:

```
##        draw win_team1 win_team2
## 1 0.2304527 0.5020576 0.2674897
```

Guessing the results of the test_set games gives an accuracy of 0.308642. The confusion matrix is listed below.

```
##           Reference
## Prediction  0  1  2
##          0  1 13  2
##          1 10 19 17
##          2  7  7  5
```

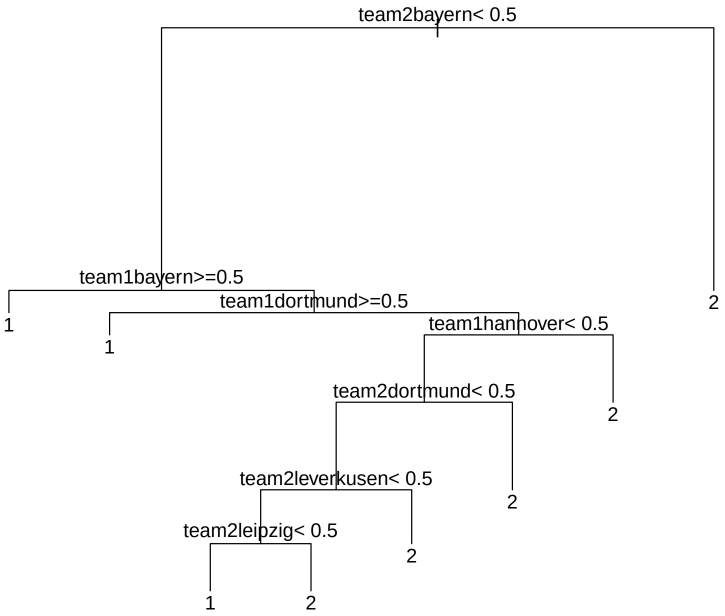
Hopefully a simple machine learning algorithm can perform better.

2.2.2 Classification (decision) tree

The prediction of results can be seen as a classification problem with categorical outcome, with possible solutions derived from prediction trees. The simplest possible model to think of is just to predict the winner using the information on the team opponents of each game.

```
tr.control <- trainControl(method = "repeatedcv",
                           number = 25, repeats = 10)
fit <- train(winner ~  team1 + team2,
             data = train_set,  method = "rpart",
             trControl = tr.control)
```

The resulting decision tree can be seen in the figure below.



On top of the tree is bayern, actually illustrating that they will win all their away games and in the second root also all of their home games. With the exception of games against bayern, dortmund is also supposed to win all their home games.

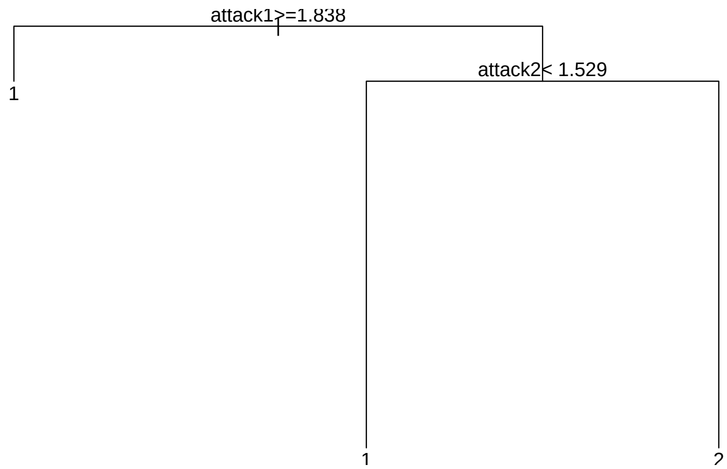
All in all, the confusion matrix of this model shows the following properties:

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1  2
##          0  0  0  0
##          1 13 35 12
##          2  5  4 12
##
## Overall Statistics
##
##                Accuracy : 0.5802
##                  95% CI : (0.4654, 0.6891)
##     No Information Rate : 0.4815
##     P-Value [Acc > NIR] : 0.04762
##
##                   Kappa : 0.2591
##
##  Mcnemar's Test P-Value : 6.523e-05
##
## Statistics by Class:
##
##                      Class: 0 Class: 1 Class: 2
## Sensitivity            0.0000   0.8974   0.5000
## Specificity            1.0000   0.4048   0.8421
## Pos Pred Value            NaN   0.5833   0.5714
## Neg Pred Value         0.7778   0.8095   0.8000
## Prevalence             0.2222   0.4815   0.2963
## Detection Rate         0.0000   0.4321   0.1481
## Detection Prevalence   0.0000   0.7407   0.2593
## Balanced Accuracy      0.5000   0.6511   0.6711
```

The accuracy is 0.5802469, but as we only predict wins for either team1 or team2, especially the sensitivity of this model for class 0 (draw) is the lowest possible value namely 0, whereas specificity is 1.

Does it help to include attack and defense scores as outlined in the exploratory analysis?

```
tr.control <- trainControl(method = "repeatedcv",
                           number = 25, repeats = 10)
fit <- train(winner ~ attack1 + defense1 + attack2 + defense2,
             data = train_set, method = "rpart",
             trControl = tr.control)
```



```
##
## Confusion Matrix:
```

```
##           Reference
## Prediction  0  1  2
##          0  0  0  0
##          1 10 33 10
##          2  8  6 14
```

```
##
## Accuracy:
```

```
## [1] 0.5802469
```

Well, accuracy is actually the same. This model predicts a home win if the attack score of the home team is greater or equal to 1.83. If the attack1 score is lower than a home win is only predicted if the attack score of the opposing team is lower than 1.52.

So this model is definitely simpler than out first try with team names, so let's see what is the result of multiple such trees.

2.2.3 Random forests

Irizarry (2019, 594) points out:

> Random forests are a very popular machine learning approach that addresses the shortcomings of decision trees using a clever idea. The goal is to improve prediction performance and reduce instability by averaging multiple decision trees (a forest of trees constructed with randomness). It has two features that help accomplish this.

Applied to our model the results are:

```
tr.control <- trainControl(method = "repeatedcv",
                           number = 25, repeats = 10)
fit <- train(winner ~ attack1 + defense1 + attack2 + defense2,
             data = train_set, method = "rf",
             trControl = tr.control)
fit
```

```
## Random Forest
##
## 1143 samples
##    4 predictor
##    3 classes: '0', '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (25 fold, repeated 10 times)
## Summary of sample sizes: 1097, 1096, 1097, 1098, 1097, 1097, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.4654599  0.1488669
##   3     0.4627640  0.1479298
##   4     0.4649882  0.1522597
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
res <- confusionMatrix(predict(fit, newdata = test_set), test_set$winner)
res$table
```

```
##           Reference
## Prediction  0  1  2
##          0  3  5  4
##          1  8 25  4
##          2  7  9 16
```

The accuracy is 0.5432099. One common drawback of these classification tree based models is the prediction of the three categories of outcome (win_team1, win_team2, draw). Although this is the most important task to do, there might be in the end of the league competitions teams having the same amount of points. Then, the number of scored and received goals decides on who is ranked before.

So, it is time to turn to another model approach relying on the Poisson distribution for scored goals, shortly introduced in the exploratory analysis.

2.2.4 Model based on Poisson distribution for scores

Hickman (2019) explains the advantages of using a Poisson distribution like $x_{ij} \sim Poisson(e^{\alpha_i \breve{\ } \beta_j+\gamma})$ where no matter what values α (attacking strength), β (defensive power), or γ (home advantage) take, the exponent of their sum will never be negative. When playing a very strong away teams, the mean goals will tend towards 0 (though will never actually reach it). Furthermore an iterative process is shown to optimize the parameters for the teams, and actually also keeping them within a normalized range of +/- 1.

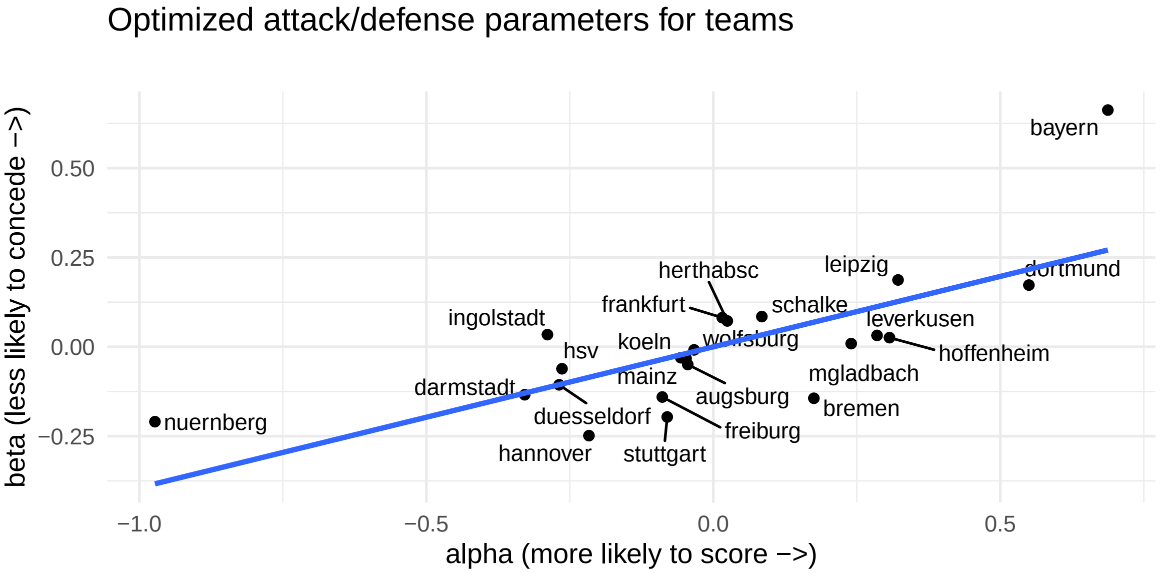


Fig. 7: Optimized attack/defense parameters for teams.

```
##           Reference
## Prediction  0  1  2
##          0  0  0  0
##          1 11 32 14
##          2  7  7 10
```

The accuracy of the model is 0.5185185. The Poisson model does not bring an improvement as far as the accuracy is concerned. The advantage of applying this Poisson model, however, is the availability of iteratively simulated most likely scoring results.

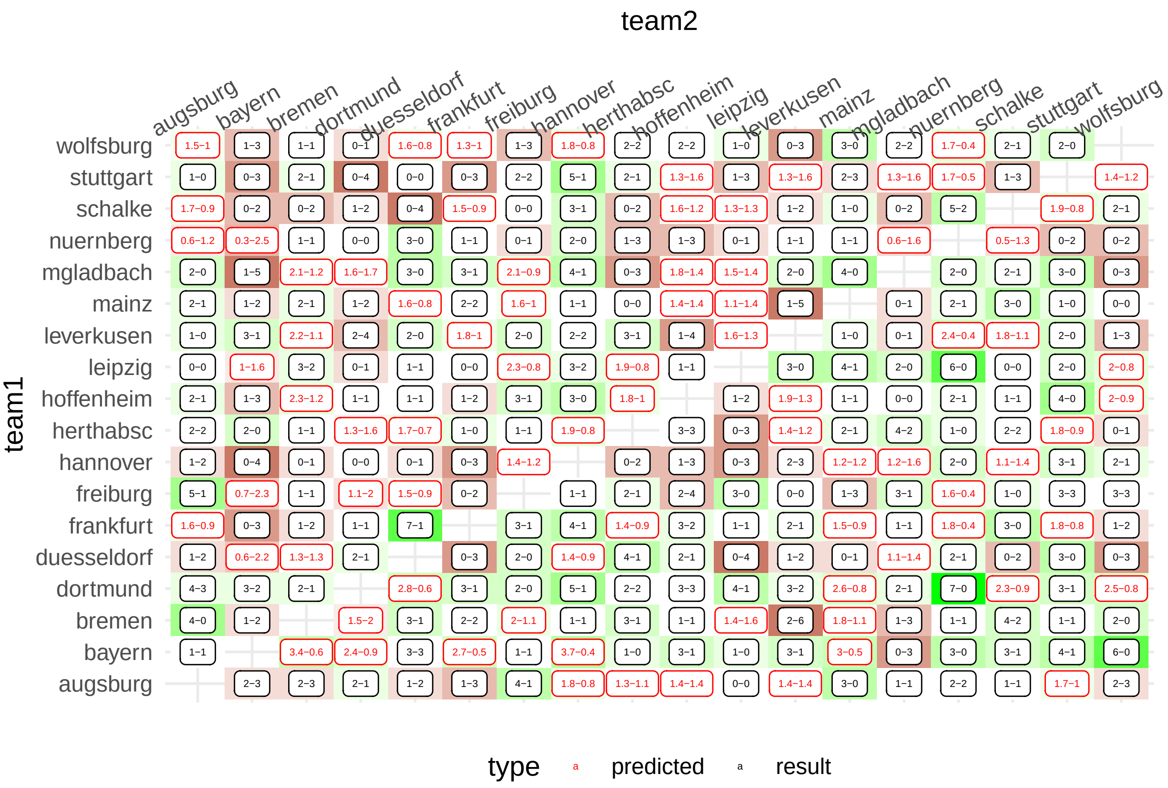


Fig. 8: Poisson model scores.

For final modeling the approach using the poisson distribution[10] is found to be most suitable, as it allows the prediction of goals, what might become essential for final league standings.

3 Results

Despite the lower accuracy compared to the tested classification trees, the decision was taken to model the final data[11] with the Dixon-Coles model, as it offers the possibility to predict specific team scores for each game, instead of only the overall 3 categories of results (win team1, win team2, draw). To further develop the chosen approach here, an existing package with an implementation of the Dixon/Coles approach[12] is used on the initial prediction problem.

Redefining train and test sets accordingly, results in updated attack and defense parameters for each team.

[10] actually as optimized aatack and defense parameters wer introduced one might rather refer to it as a Dixon-Coles model (Dixon, Coles 1997)

[11] test_set: all games after round 25 of season 2019/20; train_set: all games up to lockdown and all past games from seasons 2016/17, 2017/18, 2018/19, 2019/20.

[12] regista package

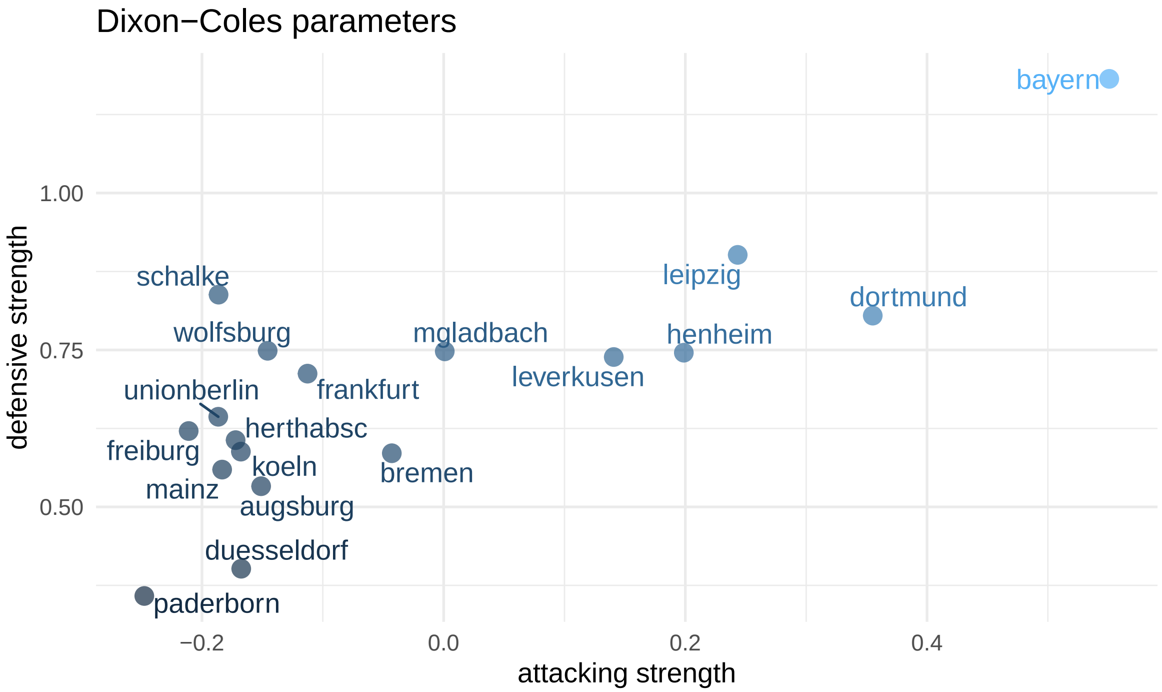


Fig. 9: Optimized attack/defense parameters (Dixon-Coles).

The goal is still to predict the results of season 2019/20 after the temporary lock-down. Rerunning a simulation on each open league game several times (to be precise the simulation was rerun 10^4 times) allows to get a probabilistic view on possible final league standings. The dotted lines in Fig. 10 already show the final place that each team achieved when the season could be finally finished after the lock-down related break.

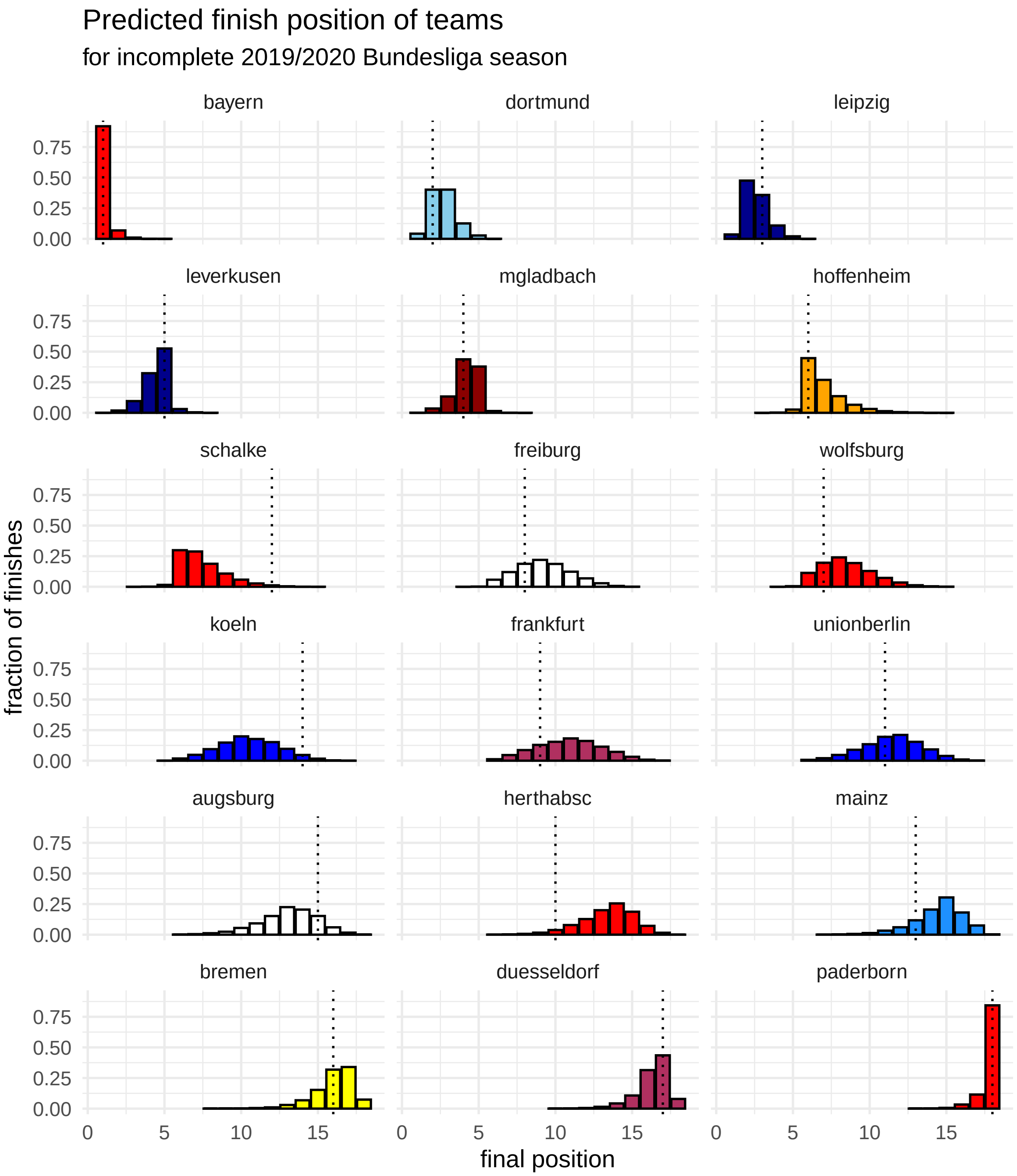


Fig. 10: Simulated league finish.

According to the simulations, bayern is very likely to finish the league as champion. However, there are also other teams with a champions chance.

```
## # A tibble: 5 x 6
##   team       predicted_finish   perc mean_finish final_rank prediction
##   <fct>                 <int>  <dbl>       <dbl> <chr>      <chr>
## 1 bayern                    1 0.9199         3   1          Champion chance
## 2 dortmund                  1 0.0422         3.5 2          Champion chance
## 3 leipzig                   1 0.0359         3.5 3          Champion chance
## 4 leverkusen                1 0.0009         4.5 5          Champion chance
## 5 mgladbach                 1 0.0011         4.5 4          Champion chance
```

At the lower side of the table, paderborn has a very high probability of finally occupying the last position. As the relegation places in german first bundesliga are the ranks 16 to 18, other teams in trouble as well.

```
## # A tibble: 9 x 3
##   team          perc prediction
##   <fct>        <dbl> <chr>
## 1 paderborn   0.9918 Relegation chance
## 2 duesseldorf 0.8286 Relegation chance
## 3 bremen      0.7319 Relegation chance
## 4 mainz       0.2586 Relegation chance
## 5 herthabsc   0.088  Relegation chance
## 6 augsburg    0.0772 Relegation chance
## 7 unionberlin 0.0112 Relegation chance
## 8 frankfurt   0.0096 Relegation chance
## 9 koeln       0.0031 Relegation chance
```

4 Conclusion and outlook

The purpose of this analysis was to get started in the highly interesting topic of soccer predictions using the knowledge and tools learnt throughout the data science course, but also already looking at tools that are commonly used for this purpose by other data scientists. Sports predictions are an interesting field in general, but it became of even greater relevance against the backdrops of the temporary suspension of whole leagues and the resulting uncertainty with questions: who would have won the league? Who would be relegated? …

This analysis was built on open public domain data-sets from openfootball. Such projects are very important as they provide recent data to the interested user free of charge. The drawback of course is the lack of individual team and player features, which could of course been helpful to get better predictions. That said, an integration of more features could be interesting especially to improve the performance of the classification trees model. Long story short, luckily plenty of interesting possibilities ahead, the more, as the new seasons already started as well.

5 Appendix

```r
print(sessionInfo())
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19041)
##
## Matrix products: default
##
## Random number generation:
##  RNG:     Mersenne-Twister
##  Normal:  Inversion
##  Sample:  Rounding
##
## locale:
## [1] LC_COLLATE=German_Austria.1252  LC_CTYPE=German_Austria.1252
## [3] LC_MONETARY=German_Austria.1252 LC_NUMERIC=C
## [5] LC_TIME=German_Austria.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
```

```
## other attached packages:
##  [1] regista_0.4.1.9000 devtools_2.3.0     usethis_1.6.0      ggrepel_0.8.2     
##  [5] RSQLite_2.2.0      DBI_1.1.0          data.table_1.12.8  caret_6.0-86      
##  [9] lattice_0.20-38    forcats_0.5.0      stringr_1.4.0      dplyr_0.8.5       
## [13] purrr_0.3.3        readr_1.3.1        tidyr_1.0.2        tibble_3.0.0      
## [17] ggplot2_3.3.0      tidyverse_1.3.0    captioner_2.2.3    kableExtra_1.1.0  
## [21] knitr_1.28        
## 
## loaded via a namespace (and not attached):
##  [1] colorspace_1.4-1     ellipsis_0.3.0       class_7.3-15        
##  [4] rprojroot_1.3-2      fs_1.4.1             rstudioapi_0.11     
##  [7] farver_2.0.3         remotes_2.1.1        bit64_0.9-7         
## [10] prodlim_2019.11.13   fansi_0.4.1          lubridate_1.7.8     
## [13] xml2_1.3.0           codetools_0.2-16     splines_3.6.3       
## [16] pkgload_1.0.2        jsonlite_1.6.1       pROC_1.16.2         
## [19] broom_0.5.5          dbplyr_1.4.2         compiler_3.6.3      
## [22] httr_1.4.1           backports_1.1.6      assertthat_0.2.1    
## [25] Matrix_1.2-18        lazyeval_0.2.2       cli_2.0.2           
## [28] htmltools_0.4.0      prettyunits_1.1.1    tools_3.6.3         
## [31] gtable_0.3.0         glue_1.4.0           reshape2_1.4.3      
## [34] Rcpp_1.0.4           cellranger_1.1.0     vctrs_0.2.4         
## [37] nlme_3.1-144         iterators_1.0.12     timeDate_3043.102   
## [40] gower_0.2.1          xfun_0.12            ps_1.3.2            
## [43] testthat_2.3.2       rvest_0.3.5          lifecycle_0.2.0     
## [46] MASS_7.3-51.5        scales_1.1.0         ipred_0.9-9         
## [49] hms_0.5.3            yaml_2.2.1           memoise_1.1.0       
## [52] rpart_4.1-15         stringi_1.4.6        desc_1.2.0          
## [55] foreach_1.5.0        randomForest_4.6-14  e1071_1.7-3         
## [58] pkgbuild_1.0.6       lava_1.6.7           rlang_0.4.5         
## [61] pkgconfig_2.0.3      evaluate_0.14        recipes_0.1.10      
## [64] labeling_0.3         bit_1.1-15.2         tidyselect_1.0.0    
## [67] processx_3.4.2       plyr_1.8.6           magrittr_1.5        
## [70] R6_2.4.1             generics_0.0.2       pillar_1.4.3        
## [73] haven_2.2.0          withr_2.1.2          mgcv_1.8-31         
## [76] survival_3.1-8       nnet_7.3-12          modelr_0.1.6        
## [79] crayon_1.3.4         utf8_1.1.4           rmarkdown_2.1       
## [82] grid_3.6.3           readxl_1.3.1         blob_1.2.1          
## [85] callr_3.4.3          ModelMetrics_1.2.2.2 reprex_0.3.0        
## [88] digest_0.6.25        webshot_0.5.2        stats4_3.6.3        
## [91] munsell_0.5.0        viridisLite_0.3.0    sessioninfo_1.1.1
```

References

Dixon, Coles, Mark. J., Stuart G. 1997. "Modelling Association Football Scores and Inefficiencies in the Football Betting Market." https://tolstoy.newcastle.edu.au/R/e8/help/att-6544/dixoncoles97.pdf.

Greenhough, Birch, Chapman, Rowlands. 2002. "Football Goal Distributions and Extremal Statistics." https://arxiv.org/pdf/cond-mat/0110605.pdf.

Hickman, Robert. 2019. "An Introduction to Modelling Soccer Matches in R." https://www.robert-hickman.eu/post/dixon_coles_1/.

Irizarry, Rafael A. 2019. "Introduction to Data Science." https://leanpub.com/datasciencebook.

Maher, M. J. 1982. "Modelling Association Football Scores." http://www.90minut.pl/misc/maher.pdf.

Sheehan, David. 2018. "Predicting Football Results with Statistical Modelling." https://dashee87.github.io/football/python/predicting-football-results-with-statistical-modelling-dixon-coles-and-time-

weighting/.