Preprint submitted to Machine Learning SI manuscript No.
(will be inserted by the editor)

A Data- and Knowledge-Driven Framework for Developing Machine Learning Models to Predict Soccer Match Outcomes

Daniel Berrar · Philippe Lopes · Werner Dubitzky

First draft: / Revision:

Abstract Predicting the outcome of soccer matches with advanced data analytics is a fascinating endeavor. The 2023 Soccer Prediction Challenge invited the machine learning community to develop innovative machine learning methods to predict the outcome of 736 future soccer matches. The Challenge posed two tasks to the participants. Task 1 was to forecast the exact match *score*, i.e., the number of goals scored by each team. Task 2 was to predict the match outcome as probability vector over the three possible *result* categories: victory of the home team, draw, and victory of the away team. We present a new data- and knowledge-driven framework for building machine learning models from readily available data to predict soccer match outcomes. A key component of this framework is a new approach to modeling *interdependent time series data of competing entities*. Using this framework, we developed various models based on k-nearest neighbor, artificial neural networks, naive Bayes, and ordinal forests models, which we applied to the two tasks of the 2023 Soccer Prediction Challenge. We also developed three reference models to gauge the performance of the resulting models. Among all submissions to the Challenge,

D. Berrar (corresponding author)
Machine Learning Research Group, School of Mathematics and Statistics
The Open University, Milton Keynes, United Kingdom, and
Department of Information and Communications Engineering, School of Engineering,
Tokyo Institute of Technology, Tokyo, Japan
E-mail: daniel.berrar@open.ac.uk

P. Lopes
Sport and Exercise Science Department
Laboratoire de Biologie de l'Exercice pour la Performance et la Santé (LBEPS)
University of Evry-Paris Saclay, Évry-Courcouronnes, France
E-mail: philippe.lopes@univ-evry.fr

W. Dubitzky
Freelance Data Scientist
Meitingen, Germany
E-mail: werner@dubitzky.com

our machine learning models based on k-nearest neighbor and neural networks achieved top performances. Our main insights from the Challenge are that relatively simple learning algorithms perform remarkably well compared to more complex algorithms, and that the key to successful predictions lies in how well soccer domain knowledge can be incorporated in the modeling process. Our data- and knowledge-driven framework can be used in combination with a plethora of supervised learning algorithms and is an important contribution to the new field of machine learning in soccer.

Keywords 2013 Soccer Prediction Challenge; data- and knowledge-driven framework; feature engineering; k-NN; ordinal forests; naive Bayes classifier; neural networks; match outcome prediction; soccer analytics.

1 Introduction

Unlike individual sports, such as running, tennis or golf, team sports are characterized by a much higher degree of complexity due to the vast number of possible player interactions, moves, tactics, and strategy. Thus, predicting the outcomes of team sports games is considered extremely difficult. Soccer, arguably one of the most popular team sports worldwide, has developed into a multi-billion dollar business. The modern game of *association soccer* is governed by the rules set forth by the Football Association Board and organized by bodies like FIFA (the Fédération Internationale de Football Association) and various continental and national federations. Predicting the outcome of soccer matches has been a subject of research since at least the late 1960s (Reep and Benjamin, 1968; Hill, 1974; Maher, 1982; Dixon and Coles, 1997; Angelini and De Angelis, 2017). Over recent years, soccer match outcome prediction has gained increased attention from the machine learning community, and spurred by international data mining competitions like the 2017 Soccer Prediction Challenge (Berrar et al., 2019a). The beauty of soccer match outcome prediction is that the fundamental task can be understood by practically anyone, and it is therefore also an excellent vehicle to showcase machine learning research to a wider audience. At the same time, it provides a truly exciting challenge for machine learning.

While match outcome predictions are of interest to clubs, soccer associations, sports equipment and services companies, etc., it is certainly also incentivized by the betting industry (Malamatinos et al., 2022). Part of the fascination of soccer is explained by the difficulty to predict the outcome of a match. If we viewed a soccer match as a scientific experiment to determine which team is better, we would realize that the number of robust measurements is rather limited. A soccer match involves hundreds of skillful moves and a wide variety of strategic and tactical plans, but the outcome is typically decided by a handful of quick and often random events, for example, a free kick, a mistake by the defender or goal keeper, an own goal, and so on. Yet the average number of shots per game is typically only between 20 and 30, and only 10% to 15% of these lead to a goal. In approximately 73.4% of the games, no more than three goals are scored, and about 66.2% of the matches end in a draw or with a margin of victory of only a single goal.

On a more abstract level, league soccer could be a viewed as a pool of contenders competing against each other in an attempt to gain an advantage over their competitors. There are analogies to many other domains; for example, in political science (e.g., the outcomes of debates between political candidates is scored to create a win-loss-draw record at different time points), the entertainment industry (e.g., two movies released at the same time compete for box office revenue, and the "winner" is determined at weekly intervals), the technological sector (e.g., "OS platform wars"), the tertiary education sector (e.g., two similar universities compete for a higher ranking), medicine and public health (e.g., the effects of two different interventions are monitored over time), and many more. In all these contexts, the common thread is the competition between two entities whose outcomes can be tracked over time.

This study presents a fundamentally new data- and knowledge-driven framework for buidling machine learning models from readily available data to predict soccer match outcomes. A key component of this framework is a new approach to modeling *interdependent time series data of competing entities*, that is, data of soccer teams that compete against each other over time. To illustrate and validate our framework, we developed a variety of machine learning models for the 2023 Soccer Prediction Challenge. This Challenge asked participants to develop machine learning approaches to predict the scores and results of 736 soccer league matches played in the second half of April 2023. While the participants were allowed the use any publicly available data for model development, the study presented here is deliberately based exclusively on the training set provided by the Challenge. This dataset consists of more than 300 000 league soccer matches covering 51 leagues in 34 countries. Each entry in this dataset describes a match in terms of league, season, date, team names, and final score. Other, more comprehensive datasets do exist, but they are not readily available. Since the authors of this study are also the organizers of the Challenge, our contributions were not in direct competition with the other Challenge participants. However, we do compare our results with those of the other participants.

The match records captured by the training set are "competitive" time series that do not lend themselves to a straightforward machine learning approach. How to construct useful predictive features from these data to be used for machine learning is not straightforward. Our framework incorporates soccer domain knowledge in the machine learning process by focusing on the knowledge-driven engineering of predictive features from interdependent time series of competing entities. We used artificial neural networks, k-nearest neighbor (Berrar et al., 2006), ordinal forests (Hornung, 2020), and naive Bayes learning (Berrar, 2018) to illustrate and validate the framework. We also developed three types of reference models to gauge the performance of the machine learning models. Two of the three reference models are based on simple statistics. The remaining reference model makes use of match outcome odds provided by bookmakers.

The novel contributions of our work can be summarized as follows: (1) We present a new data- and knowledge-driven framework for developing machine learning models from readily available match data. This framework includes fundamentally new approaches to predictive feature engineering from interdependent time series of competing entities. In principle, this framework can be used in combination with various supervised learning algorithms. We believe that our research is a milestone in the burgeoning field of machine learning in soccer and is likely to spur further work in this fascinating domain. (2) We validated our framework by building machine learning models that achieved top performances in the 2023 Soccer Prediction Challenge. Our main insights from this application are that relatively simple learning algorithms, such as k-nearest neighbor, perform remarkably well compared to more complex algorithms. The key to successful predictions clearly lies in how well soccer domain knowledge can be incorporated into the development process.

This article is organized as follows. In order to aid those readers who are not too familiar with the relevant soccer concepts, Section 2 briefly introduces some of the important notions and terms reoccurring in this paper. Section 3 summarizes the related work. In Section 4, we briefly describe the 2023 Soccer Prediction Challenge. In Section 5, we present the data- and knowledge-driven framework, giving a comprehensive rationale for the methods and approaches. In Sections 6 to 9, we apply and validate the framework by developing several machine learning models and reference models for the 2023 Soccer Prediction Challenge. The paper ends with a discussion (Section 10) and conclusion (Section 11).

2 Background

There are various types of soccer competitions at club and national team levels. However, the grassroots foundation of association soccer is *league soccer*. League soccer refers to soccer competitions that are structured in a league format. This includes professional leagues like the English Premier League or the Major League Soccer in the United States as well as amateur leagues. In these leagues, teams play a set schedule of games against one another, and points are awarded based on the outcomes of these games (three points for a victory, one point for a draw). Rankings within the league are determined by the number of points teams have accumulated (with goal difference and goals scored serving as tiebreakers).

A *league soccer* match refers to a competitive game that takes place between two *teams* within the structure of a specific soccer league. A league comprises a fixed number of teams (e.g., the English Premier league has 20, the German Bundesliga 18 teams). The first named team of a match is referred to the as the *home team*, the second as the *away team*. A match always takes place at the home team's venue or ground (some rare exceptions exist). It is well-known that the home team enjoys a competitive advantage called *home* *advantage.* Approximately 45% of all games end in a win by the home team. A scheduled match is sometimes also referred to as a *fixture.*

There are normally two ways to state the outcome of a league soccer match: *score* and *result.* The score gives the outcome of a match in terms of the number of goals scored by the home and the goals scored by the away team. We use the notation *n-m* as a shorthand for stating a match score. For example, 2-1 means that the home team scored two goals and the away team one goal. The match results states the outcome of a match in terms of a *home win*, a *draw*, and an *away win.* We often use the terms *win*, *draw* and *loss* as a kind of shorthand for stating a match result. The result is derived from the score as follows: A "win" means that the home team scored more goals than the away team, for example, a score of 2-1 represents a "win" (by the home team). The result is a draw if both teams score the same number of goals, e.g., a score of 2-2 is a "draw". Finally, a "loss" (win by the away team) occurs if the away team scores more goals than the home team, e.g., a 1-4 score represents a "loss".

The team that won the match is awarded 3 points. In case of a draw, each team is awarded 1 point. Thus, using above shorthand for result, a "win" means the home team gets 3 and the away team gets 0 points, a "draw" means both teams get 1 point, and a "loss" means the away team gets 3 and the home team gets 0 points. Based on the number of points, a league ranking table is determined. Ties are typically resolved by goal difference and goals scored. By the end of the season, the top-ranked team is normally crowned the "league champion". In leagues below the highest league, the top-ranked teams are promoted to the league above. The bottom-ranked teams are relegated to the next league below their league.

While match formats vary, in many leagues the matches are scheduled to take place over the course of a *season*, which normally lasts around eight months during which each team plays each other team twice, once at its home venue or ground and once at the opponents' venues.

The feature engineering approach presented in this paper revolves around the teams' past performances within a given league. Here, we distinguish three aspects referred to as *total*, *home*, and *away.* "Total" means that we include all considered past performances of a team, regardless of whether they have been achieved at the team's home venue or at the venues of its opponents. The home and away view, on the other hand, focus separately on a team's past performance for home matches and away matches, respectively. This captures explicitly the home and away aspects, but each view has only half the data points of the total view.

Our approach to feature modeling breaks up the natural season boundaries found in league soccer and introduces three separate league concepts: *league*, *super league*, and *meta league.* Under the league view, we maintain the natural structure of league soccer and process the data of a given league strictly in a season-by-season fashion. Under the super league view, we combine all data of a given league over all seasons covered in the Challenge datasets. This leads to an artificial league that we call "super league". Super leagues consist of more teams than the underlying league would allow within a single season. The meta league concept is even more radical, as it combines the data of all leagues across all covered season in the Challenge data into a single league called "meta league".

3 Related work

One of the first studies on soccer data analysis concluded that chance dominates the outcome of a match (Reep and Benjamin, 1968). Maher (1982) had more success with a Poisson model for the number of goals that a team scores during a match, which was applied to four English football league divisions for the seasons 1973 to 1974. As Dixon and Coles (1997) pointed out, it is comparatively easier to predict which teams will perform well in the long run, whereas reliable predictions for individual matches are far more difficult. Jurman (2020) showed that the match outcomes in longer competitions, such as a national league, essentially follow a linear trend, which can be exploited for match outcome prediction. Still, to what extent the outcome of match is predictable remained largely unknown.

Over the last 20 years, machine learning methods have been increasingly used for sports outcome prediction. In one of the first studies focusing on soccer, O'Donoghue et al. (2004) used machine learning methods to predict the results of the 2002 FIFA World Cup, but the best predictions were obtained from a simulation study involving a commercial game console. More recently, Malamatinos et al. (2022) used k-NN, LogitBoost, support vector machines, random forests, and CatBoost to predict the outcomes of the Greek Super League. Among the investigated models, CatBoost achieved the best performance. Similarly, Kundu et al. (2021) used different learning algorithms to predict the outcomes of matches of the English Premier League and obtained the overall best performance with a gradient boosting regressor model. Hubáček et al. (2019) and Berrar et al. (2019b) also reported promising results from gradient boosted trees for the 2017 Soccer Prediction Challenge.

Ievoli and Palazzo (2021) used passing network indicators quantifying player interactions as explanatory variables. They showed that network-based variables are related to a team's offensive actions and can improve the performance of forecasting models. Like in most real-world applications of machine learning, the key to success seems to lie in how well domain knowledge can be modeled and incorporated into the machine learning process (Berrar et al., 2019b).

Ren and Susnjak (2022) used the Kelly index to first categorize football matches of different predictability and then applied a variety of machine learning algorithms, which were benchmarked against bookmaker odds. Razali et al. (2022) developed a model for soccer match outcome prediction based on the pi-rating system using TabNet, which is a deep neural network for tabular data. The researchers re-analyzed the data from the 2017 Soccer Prediction Challenge and reported a better performance than the top-rated participants.

Stübinger et al. (2020) developed an ensemble of machine learning models to predict outcomes based on match and player attributes. In a simulation study, they included all matches of the top five European football leagues and the corresponding second leagues between 2006 and 2018. They benchmarked their predictions against the odds from one of the leading online bookmakers. The ensemble model resulted in economically and statistically significant returns of betting investments.

Predicting the outcome of a soccer match is extremely difficult. First, there are many diverse factors that interact in highly complex ways to produce goals in soccer. Second, compared to other team sports like basketball, ice hockey, or volleyball, soccer is a sport with a very low number of scoring attempts (shots on goal) and actual scores (goals) in a typical match. In the study presented here, we used a dataset that provides highly limited information about soccer matches. Predicting the outcome based on such limited data represents a formidable challenge. More comprehensive data about soccer matches may exist in commercial databases, but these databases are not readily available. However, as the mentioned studies illustrate, it is indeed possible to predict the outcomes, at least to some extent. The goal of the 2017 Soccer Prediction Challenge was to explore to what extent the outcome of a soccer match could be predicted with machine learning based on readily available match data (Dubitzky et al., 2019; Berrar et al., 2019a). The overall best performance was achieved by a k-nearest neighbor model trained on rating features (Berrar et al., 2019b).

4 The 2023 Soccer Prediction Challenge

Central to the 2023 Soccer Prediction Challenge is a dataset describing the data of 736 league soccer fixtures from 44 leagues in 28 countries. This dataset is referred to as *prediction set*. All fixtures in the prediction set were scheduled to take place in the period between 14th and 30th of April 2023. The participants of the Challenge were asked to predict the outcomes of all 736 fixtures by the strict deadline of 23:59 C.E.T. on 13th April 2023, i.e., prior to the start of any of the matches. Specifically, the challenge was to predict the match outcomes in terms of *exact scores* (Task 1) and a *probability forecasts of the results* (Task 2). Participants were requested to address at least one of the two tasks. To compare the submitted predictions, two error measures were defined and communicated to the participants: the *root-mean-square error* (*RMSE*) and the *ranked probability score* (*RPS*). The RMSE was used to evaluate the score predictions (Task 1), and the RPS to evaluate the result predictions (Task 2).

The RMSE defined by Equation 1 computes the total error of score predictions across a number of predicted matches,

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}[(\hat{s}_i(H) - (s_i(H))^2 + (\hat{s}_i(A) - s_i(A))^2]} \qquad (1)$$

where n is the number of matches in the prediction set; $\hat{s}_i(H)$ denotes the predicted and $s_i(H)$ the observed number of goals scored by the home team in the i^{th} match, and $\hat{s}_i(A)$ and $s_i(A)$ are the respective predictions for the goals scored by the away team, such that $\hat{s}_i(H), s_i(H), \hat{s}_i(A), s_i(A) \in \mathbb{N}_0$.

The smaller the RMSE, the better the predictions. In this definition, each individual error is the sum of the squared difference of the predicted and observed number of goals scored by the home, and the squared difference of the predicted and observed number of goals scored by the away team.

Notice, for the evaluation of the Challenge score predictions, only positive whole numbers including 0 were permitted. For some models in the present study, we allow positive real numbers including 0 in the model training phase, but round the final predictions to whole numbers including zero.

The RPS is a scoring function designed for ranked or ordinal categories (Epstein, 1969; Constantinou and Fenton, 2012). The participants of the Challenge were required to provide their result predictions as a probability vector, $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, \hat{y}_3)$, where $\hat{y}_1$ denotes the predicted probability of a win, $\hat{y}_2$ of a draw, and $\hat{y}_3$ of a loss, such that $\sum_{j=1}^{3} y_j = 1$. To compute the RPS of such a probabilistic forecast, we need to represent the *observed* result as a hotvector $\mathbf{y} = (y_1, y_2, y_3) | y_j \in \{0, 1\} \wedge \sum_{n=1}^{3} y_j = 1$. Hence, the hotvector $(1, 0, 0)$ denotes an observed win, $(0, 1, 0)$ a draw and $(0, 0, 1)$ a loss. For example, given the result prediction $(0.6, 0.3, 0.1)$ and an observed draw, $(0, 1, 0)$, the RPS for this prediction works out to be 0.185. The lower the RPS, the better the prediction. The RPS is defined as shown in Equation 2.

$$\text{RPS} = \frac{1}{2} \sum_{i=1}^{2} \left(\sum_{j=1}^{i} (\hat{y}_j - y_j) \right)^2 \tag{2}$$

The *average ranked probability score*, RPS_{avg}, was used to provide a combined score of all predictions in the prediction set,

$$\text{RPS}_{\text{avg}} = \frac{1}{n} \sum_{i=1}^{n} \text{RPS}_i \tag{3}$$

In addition to the prediction set, the Challenge also provided a training set consisting of the basic match data of 302 691 completed league soccer matches from 51 leagues in 34 countries. All matches in the training set were played after the 18/03/2000 and *before* the cut-off date of 05/04/2023. The cut-off year of 2000 was chosen because from 2000 onward all leagues covered in the training set have had adopted the three-points-for-a-win rule. The Challenge participants were allowed to use any publicly available datasets to train their machine learning models, or the training set provided by the organizers, or both.

The approach and models in this study are based exclusively on the training set (and prediction set) provided by the Challenge.

Notice, due to late changes in the fixture schedules, the prediction set that was finally used to evaluation the submission to the 2023 Soccer Prediction Challenge was reduced to 714 matches. These matches originate from 43 leagues in 27 countries.

5 Data- and knowledge-driven framework

A machine learning approach to predicting the outcomes of soccer matches based on readily available match data like those provided by the 2023 Soccer Prediction Challenge entails the following: (1) use soccer domain knowledge to understand the problem and inform the modeling process; (2) process and transform the time series data from competing teams, with the goal of determining optimal predictive features; (3) generate and evaluate machine learning models based on test data; (4) apply selected models to the Challenge prediction set for an independent evaluation on real future data.

The subsections below describe the various aspects of the data and the framework.

5.1 Readily available match data—the Challenge data

The training data provided for the 2023 Soccer Prediction Challenge captures basic information about soccer matches. Each entry includes the date on which the match took place, the names of the two teams facing off in the match, the final score, the name of the (association) soccer league, and the season in which the match was played. Table 1 shows the first five entries of the training set.

Table 1: Excerpt of the Challenge training set (Sea=season, Lge=league, HT=home team, AT=away team, HS=home score, AS=away score, GD=goal difference, WDL=win(W), draw (D), or loss(L))

Sea	Lge	Date	HT	AT	HS	AS	GD	WDL
00-01	GER1	11/08/2000	Dortmund	Hansa Rostock	1	0	1	W
00-01	GER1	12/08/2000	Bayern Munich	Hertha Berlin	4	1	3	W
00-01	GER1	12/08/2000	Freiburg	VfB Stuttgart	4	0	4	W
00-01	GER1	12/08/2000	Hamburger SV	Munich 1860	2	2	0	D
00-01	GER1	12/08/2000	Kaiserslautern	Bochum	0	1	-1	L

Note, the season values in Table 1 are a shorthand. For example, “00-01” is an abbreviation for the “2000/2001” season (sometimes also written “2000/01”).

Clearly, this data is rather limited in terms of variables that are potentially predictive of the outcome of a soccer match. However, one of the advantages of the training data is that they are widely and readily available, both in terms of historic records and timely availability before future matches take place.

The Challenge training set includes only two quantitative variables (positive whole numbers including 0), i.e., the final score of completed matches. Each score variable could at the same time be viewed as a characteristic of the home team as well as the away team. For example, a match score of 4-1 is simultaneously indicative of the home team's attacking strength (the home team scored 4 goals) and the away team's defending weakness (the away team conceded 4 goals).

In addition to the quantitative score variables, the training data include the names of the teams facing off against each other. Potentially, these nominal categorical variables could serve as predictive features. For example, the correlation of the team name pair $(Liverpool, X)$ with a home win might be stronger than that of the pair $(Fulham, X)$ (where X is any other team in the league).

One aspect of league soccer is that all matches are confined to within a league and a season. This means that the league (nominal categorical) and season (nominal ordinal) variables are identical for all matches in a given league and season. Thus, these variables do not seem to be good predictors of individual matches.

The date variable does not seem to by highly predictive in terms of the outcome of individual matches, either. Typically, on a given date, multiple league matches take place. However, the date variable is crucial, as it imposes a total (chronological) order relation on the matches from the oldest to the most recent matches. Therefore, each team in the Challenge training set could be viewed as a subject or individual of a longitudinal time series, because certain characteristics of a team could be "measured" at multiple points in time. Consider Table 4. Focussing on Liverpool, we can track the goals that Liverpool has scored over time as follows: 2 goals in the match on 13/02/2023, 2 on 18/02/2023, 0 on 25/03/2023, and so on. However, we need to keep in mind that such quantities not only depend on the team under consideration, but also on the team's opponent. It is obviously easier to score goals against an opponent with a weak defense. If we computed the average of such quantities over a number of time points, then the influence of individual opponents should average out. The feature engineering approach presented in this study is based on the concept of longitudinal team performance time series.

5.2 The team perspective

It is not immediately obvious how the Challenge training set could be used to train machine learning models. In order to make the data amenable to machine learning, we need to determine a data feature representation which is predictive of the match outcome. Here, we explain our approach to feature modeling from the perspective of teams.

Given the two teams facing off in a match, our main concept of predictive modeling revolves around the idea that prior to a match, each of the two opposing teams could be characterized by a set of quantitative features that are prognostic of the match outcome. Such features could be grouped into the *team performance*, such as the team's ability to score goals, concede no goal over a sequence of matches, achieve certain victory rate at the team's home venue, and so on. These categories are necessarily linked to the two quantitative score variables of the training data. Even with the highly limited data we have in the training set, a considerable number of such categories are conceivable. According to soccer domain knowledge, the following *team performance* aspects should important for predicting the outcome of a competitive soccer league match: attacking and defending ability, ability to maximize results, and performance or success in the league. Based on these, we focus on the specific team performance categories depicted in Table 2.

Table 2: Eight important team performance categories

Performance category	Description
Scoring	Team's ability to score goals (attacking performance)
Conceding	Team's ability to prevent goals (defensive performance)
Scoring/conceding	Team's aggregate ability of scoring and conceding goals
Winning	Team's ability to win matches
Drawing	Team's ability to hold a draw
Losing	Team's propensity to lose matches
Points	Team's ability to earn points
League	Team's league table position

The *Scoring* and *Conceding* team performance categories relate directly to the two score variables in the training set. A soccer team that scores many goals (in relation to its competitors) could be viewed as having a strong attack. Similarly, a team that consistently prevents its opponents from scoring many goals is likely to have a strong defense. The stronger a team's attack and defense compared to its opponent's attack and defense, the more likely it is to prevail. The *Scoring/Conceding* category refers to an aggregate or combination of the team's Scoring (attack) and Conceding (defense) categories.

The *Winning*, *Drawing*, and *Losing* performances of a team relate directly to the result of matches (win, draw, loss), the points that a team earns, and how the team performs in the league as a whole. The *Points* category is essentially similar but provides a different "view" of the same aspects. This is relevant, given the points awarding scheme in league soccer (3 points for a win, 1 for a draw, and 0 for a loss).

The *League* category provides a kind of "league-calibrated" view of a team's overall performance or success in the league. How successful a team is in the league is expressed by the team's position or rank in the league table. The higher a team is ranked in the table, the more competitive the team is in relation to teams ranked below it. Soccer league tables are normally ordered by points first and then by goal difference (and by the number of goals scored).

The team performance categories discussed so far do not explicitly differentiate a critical factor called *home advantage*. The home advantage in soccer (and indeed other team sports) is a well-known phenomenon whereby teams experience a considerable competitive advantage by playing at their home venue (Wunderlich et al., 2021; Nevill and Holder, 1999). Based on the Challenge training set of over 300 000 matches, we can provide a quantitative estimate of the home advantage as follows: 44.83% matches are won by the home team (compared to 28.14% wins by the away team), and the average number of goals scored by the home and away team is 1.47 and 1.12, respectively. The team performance categories discussed above could be constructed separately for a team's *home*, or *away* and *total* (home and away matches combined) performances. Thus, our feature modelling framework is capable of explicitly capturing the home advantage dimension.

So with the eight feature candidates listed in Table 2, we could potentially represent the performance of a single team by a maximum of 24 features—eight features for each of the total, home, and away performances, respectively. Notice, this approach to data feature representation would be highly redundant because six out of the eight basic feature categories are ultimately derived from the scoring and conceding performances, and the eight total performances are derived from the home and away performances, respectively. Since each match involves two teams, the maximum number of features per match would be 48, that is, 24 for the home team and 24 for the away team. Ideally, a layered automated feature learning approach like deep learning could then be employed to construct the optimal feature representation.

Having identified potential predictive features that could be constructed from the training data to facilitate machine learning, the question is: "How do we construct these?" It turns out that there are a number of issues that need to be considered before we can proceed.

Table 3: Man City matches on and prior to 01/04/2023

Date	HT	AT	HS	AS
12/02/2023	Man City	Aston Villa	3	1
15/02/2023	Arsenal	Man City	1	3
18/02/2023	Nottingham Forest	Man City	1	1
25/02/2023	Bournemouth	Man City	1	4
04/03/2023	Man City	Newcastle United	2	0
11/03/2023	Crystal Palace	Man City	0	1
01/04/2023	**Man City**	**Liverpool**	4	1

Table 4: Liverpool matches on and prior to 01/04/2023

Date	HT	AT	HS	AS
13/02/2023	Liverpool	Everton	2	0
18/02/2023	Newcastle United	Liverpool	0	2
25/02/2023	Crystal Palace	Liverpool	0	0
01/03/2023	Liverpool	Wolverhampton	2	0
05/03/2023	Liverpool	Man United	7	0
11/03/2023	Bournemouth	Liverpool	1	0
01/04/2023	**Man City**	**Liverpool**	4	1

Let's say we are considering an upcoming match between the home team H and the away team A. For each team, we compute an estimate of a feature f by aggregating the n recent performances corresponding to the feature f. Obvious candidates for aggregation include the mean and median. To illustrate this idea, consider Tables 3 and 4: for the ENG1 league match on 01/04/2023 between Man City and Liverpool (which ended in a 4-1 win for Man City), we determine that Man City scored 2.33 and Liverpool 2.17 goals on average in the their $n = 6$ recent encounters prior to the match. So before the teams are facing off, the scoring performance of Man City was considerably higher than that of Liverpool.

Tables 3 and 4 illustrate two interesting aspects of this feature modeling scheme. First, we see that the number of home and away matches that the two teams played in their six recent matches prior to 01/04/2023 is not the same—Man City played two home and four away matches, whereas Liverpool played three home and three away matches. Second, the time stamps in the date columns of Tables 3 and 4 show that the six recent matches of the two teams were not all played on the same dates. This leads to a situation where one team may have longer recovery times between matches than the other. However, the longer the time series becomes, the lower will be the effect of the home/away and time stamp discrepancies.

This leads us to the critical question: "How far back should one look into the history of recent matches of the teams to construct optimal features?" To answer this question, we need to consider various aspects of the league/season structure of league soccer.

5.3 The league/season perspective

The Challenge training set comprises a total of 302 691 matches covering 51 leagues in 34 countries. In each country, league soccer is structured into three separate levels of organization, each level providing its own context. The first level of organization is the country level. The leagues within a country are structured along a hierarchy of divisions or tiers. The higher the division, the higher the quality of the teams playing in that division. The second level is the league level. Within a league, the number of teams is fixed. From league to league the number may vary, e.g., the SCO1 league consists of 12 and the ENG2 league of 24 teams. Each league organizes its matches into seasons. The duration of a soccer season varies by country and league. A typical season is played over a period of about eight to ten months. The number of seasons covered for each league in the training set varies from a minimum of 10 to a maximum 24. The earliest season covered is the 2000/2001 season and the most recent one is the 2023/2024 season.

Across all leagues and seasons, the data format in the training set is the same. Thus, one might be tempted to model the features by combining all league-season subsets into a single large dataset. However, domain knowledge as well as the data tell us that the characteristics of the leagues may vary from one league to another. In statistical terms, one cannot necessarily assume that data from different leagues come from the same population. We refer to this as the *cross-league compatibility problem*. We briefly illustrate this issue based on the home win proportion and away goal average for the DZA1 and JPN2 leagues in the Challenge training set. The home win proportion is 53.21% for DZA1 (Algerian top division) and 39.12% for JPN2 (2nd division of Japan), respectively. Given these league-to-league variations, one must assume that constructing features by simply merging all leagues into a single large dataset is probably not wise.

Processing the data for each league separately raises the question of how to deal with the seasons within a league. Soccer domain knowledge informs us that each season within a league is unique. This implies that a team in one season may not be necessarily directly comparable with the same team in the next season. We refer to this as the *season-transition problem*. There are two main components to the season-transition problem: (1) the change of teams that feature in a league from season to season; (2) the change within teams, in particular with regard to their players, from season to season.

Most soccer leagues see a change of teams featuring in the league due to relegation and promotion. This means that by the end of a season, some of the lowest ranked teams in the league will be relegated to the league below, while some of the highest ranked teams will be promoted to the league above. For top league in each country, teams leave the league only via relegation route. In the Challenge training set, 34 of the 51 leagues represent top-tier leagues in their respective countries. This means the team fluctuation across seasons is limited, because the lowest ranked teams will be replaced by the highest ranked teams from the league below. For example, the GER1 league comprises 18 teams. After each season, two or three teams are relegated to the lower GER2 league and replaced by promoted teams from GER2. This means that 15 or 16 out of 18 teams that played in one season in GER1 play also in the following season. So there is a reasonable consistency from season to season. 17 out of the 51 leagues in the Challenge training set have a league above and below them. For these leagues, there is a greater shake-up in terms of the teams featuring in the league from season to season. For example, between 4 to 6 out of a total of 18 teams featuring in the GER2 league will be replaced as a result of promotion (from GER3 to GER2) and relegation (from GER1 to GER2). Although somewhat less than in the 34 top-ranked leagues, there is still a majority of teams that remain in the league from season to season. The following example from the GER1 league further illustrates the relative "stability" of teams belonging to a league across season boundaries. Over the 23 seasons (from 2000/2001 to 2022/2023 season) covered for the GER1 league (which always comprises 18 teams) in the training set, no more than 37 teams featured in the league. This relative stability of teams within a league is central to our feature modeling approach.

Transiting from season to season, another—and perhaps even more important—problem crops up. At the months-long break between seasons, soccer clubs usually change many things, from coaching staff, to players, to new business arrangements with sponsors, etc. Perhaps one of the most critical changes is due to players leaving the team and new players arriving. This is critical because the players are the most valuable assets that professional soccer clubs have. And there are regulatory restrictions on when players can be recruited. Usually, there are only two windows in a season where this happens. Therefore, errors made when changing the composition of the players of a team cannot be immediately corrected. Also, it is always a challenge to integrate new players into a team.

After investigating the main aspects arising from cross-league differences and from the season to season transitions within a league, we view (for the time being) the matches played within a particular league and season as the most "natural" unit. Based on this assumption, feature engineering should be performed separately for each of such units before any further data integration or model training should occur. This view is consistent with common soccer domain knowledge. While this assumption seems absolutely sound, we ultimately abandoned it in our feature engineering approach, for the following reasons.

Computing features separately for each season within each league does raise some issues; especially, matches at the start and end of the season pose problems.

First, towards the end of a season, many decisions (promotion, relegation, championship, qualification for other competitions, etc.) have normally been settled before the final matches are being played. Thus, a number of matches in the final stages of a season may no longer be fully competitive, casting doubt on the robustness of features derived from such matches. This is what we refer to as the *end of season problem.* Ideally, such matches should be identified and perhaps removed from the training data altogether. However, the disadvantage is that dropping matches will reduce the size of the training set.

Second, at the beginning of the season, we face what we refer to as the *start of season problem.* As a season for a given league gets underway, all teams start with zero goals and zero points. We have to wait until each team has played a decent number of matches before we can compute reliable performance indicators. The problem is that this leads to a considerable reduction of the instances in training data. To illustrate this idea, let's say we require each team to have played a minimum of six matches before we compute performance features. Consider the total of 380 matches played over a single season in the ENG1 league (20 teams). We already lose the last 10 matches from the dataset, as we do not have any features to compute for any matches after the last match day in the season. Depending on the concrete match schedule, additional 60 matches would be lost when we require each team to have played a minimum of six matches. So in total, we lose about 70 (18.4%) of the 380 matches from a single season of ENG1.

The *strength of the opposition problem* is perhaps one of the more subtle aspects relevant to feature engineering. As we aggregate various past performances of a team into a predictive feature value, we should factor in the strength of the opponent against which the team has achieved these performances. For example, a 2-1 win against a top team should carry more weight than a 3-1 win against a lesser opponent.

Finally, we consider the *recency problem.* If we model features based on the performances of the teams over their n recent matches, we need to determine an optimal value for n. The first obvious choice is to include all matches a team has played in the current season prior to a given match. The idea is that the features would become more robust the further the season has progressed. However, one could argue that performances too far back in the season are obsolete and no longer predictive for the current match. So in a sense, we are looking at a compromise between long-term and short-term performance trends. Adopting a within league-season approach for feature modeling means that the number n of recent matches that one could consider for a team changes gradually from 0 (for the team's first match in a season) to the maximum of N (for the team's last match) over a season. The value of N depends on the format of the league schedule, which varies from league to league, an the number of teams, T, featuring in a league. A common schedule format consists of each team playing the other twice, once at home and once at the away venue. For such a format, $N = 2T - 1$, where T denotes the number of teams in a league. For example, for the ENG1 league, $T = 20$, hence $N = 2 \times T - 1 = 2 \times 20 - 1 = 37$. This means that right before a team's last (38th) match in the season in the ENG1 league, a team has already played $n = 37$ matches. These 37 matches would be the basis for calculating the actual features indicative of the outcome of the current match (in this case the last match of the team in the season). However, the problem is that this is the maximum value for n. In particular, in the beginning of the season, n is small and perhaps not ideal for calculating robust predictive features.

5.4 The feature model

Having considered various aspects relevant to feature modeling, we performed an exploratory data analysis and model prototyping using the k-nearest neighbor algorithm. This has led us to adopt the following feature modeling assumptions and decisions:

Table 5: Nine selected teams of the ENG1 super league over nine consecutive seasons, including the season 2022/23 up to 4th April 2023. The cell numbers state the number of games a team has played in the corresponding season. The Sum column states the total number of matches a team has played since the start (2000/01) of the super league.

Team	14-15	15-16	16-17	17-18	18-19	19-20	20-21	21-22	22-23	Sum
Norwich City	0	38	0	0	0	38	0	38	0	266
Portsmouth	0	0	0	0	0	0	0	0	0	266
Fulham	38	38	38	38	38	0	38	0	28	266
Watford	0	38	38	38	38	38	0	38	0	266
Burnley	38	0	38	38	38	38	38	38	0	304
Nottingham Forest	0	0	0	0	0	0	0	0	28	28
Middlesbrough	0	0	38	0	0	0	0	0	0	380
Newcastle United	38	38	0	38	38	38	38	38	27	787
Man City	38	38	38	38	38	38	38	38	28	826

– Compute features strictly separately for each league.
– For each league, merge matches from all seasons into a “super league”.
– Let data analysis and model prototyping decide the best recency depth n.
– Explore a limited number of “plausible” (based on domain knowledge) feature sets.
– Select promising features sets and develop machine learning models.

Based on our analyses, we decided to compute features strictly on a league-by-league basis. Furthermore, we merged the data from all seasons within a league and sorted the resulting dataset chronologically based on the date variable. This process generated a kind of “super league”. Such a super league approach has two main advantages. First, it explicitly captures the idiosyncratic context of each league, thus mitigating the cross-league compatibility problem. Second, for most teams, it allows us to construct much longer time series than would be possible with a season-by-season approach. This should reduce the impact caused by the beginning-of-season-problem.

Conceptually, as discussed above, the super league approach contradicts common soccer wisdom because the composition of the teams may change drastically from one season to the next due to change of players, coaching staff, club ownership, sposorship, etc. Other problematic issues include a potential limitation of the length of the time series for some teams and “gaps” in the time series for teams that feature in the super league for only a few seasons.

Merging all seasons of a league into a single super league means that all teams that have ever played in the corresponding league make up the super league. For instance, the training set covers 23 seasons for the ENG1 league from the 2000/01 to 2022/23 season. After merging the matches of all 23 seasons into one super league, the resulting dataset contains a total of 8639 matches and includes the 45 teams that have ever played in ENG1 in these 23 seasons. Table 5 depicts a selection of nine teams over nine seasons of the ENG1 league. Notice, all seasons except the last (labelled 22-23) are complete. Since there are 20 teams in the ENG1 league, each team plays a total of 38 matches over a complete seasons. First, we notice that Portsmouth has not played in any of the nine seasons depicted in the table. But the Sum column suggests that Portsmouth has featured 266 times in the ENG1 league in the seasons from 2001/01 to 2013/14 (prior to the nine seasons in the table). Second, we can infer from the table that Man City has featured in 22 of the 23 seasons covered by the training set (the team missed out on the 2001/02 season). Third, we see that most teams depicted in the table (except Man City) did not feature in *all* of the nine seasons shown in Table 5. This causes gaps in their team performance time series. For example, Norwich City appeared only in 3 of the 9 seasons depicted in the table, with 3 one-season and one three-season gaps. After some experimentation, we decided that from the perspective of a given match, we can simply concatenate all matches prior to the current match in chronological order, so that we have a continuous time series for each team spanning all seasons the team has featured in the league. For example, if we are interested in the average number of goals that Watford scored in their $n = 50$ matches prior to their last (38th) match of the 2021/22 season, we would consider Watford's remaining 37 matches of the 2021/22 and the last 13 matches in the 2019/20 season (i.e., we simply ignore that Watford did not feature in ENG1 in the 2020/21 season).

Another subtle issue with the super league approach is linked to teams that feature only a small number of seasons in the time frame covered by the Challenge dataset (from 2000/01 to 2022/23 seasons). For these teams, the length of the team performance time series is limited to the number of times they featured in the league. For example, there are four teams that featured only in a single season in the ENG1 league over the considered time frame. For these teams, the maximum length of the team performance time series is limited to 38 time points. Nottingham Forest is one of those teams. Since they have entered the ENG1 for the first time in the incomplete Challenge season 2022/23, their situation is even worse, as they have appeared only in 28 matches in the entire training set.

One of the major advantages of the super league approach is that it affords the computation of much longer time series than would be possible with a within-season approach. Another advantage is the flexibility it gives us in terms of the datasets that we use for model training. First, since the datasets created by the super league approach capture the characteristics of each league separately, we have the flexibility to *combine* such datasets before model training. For example, we could combine all three super league feature datasets from the three Germany leagues (GER1, GER2, GER3) into a single "country league" dataset, or we could indeed combine *all* super league feature datasets arising from the Challenge training set into a single "meta league" dataset. Second, we could also keep the super league feature datasets separate and train our models strictly on a league-by-league basis, i.e., for a given league, we develop a machine learning based exclusively on the corresponding super league dataset.

Unfortunately, the end-of-season-problem is not explicitly addressed by the super league approach presented here. Our modeling assumption is to ignore this issue, as it affects only a handful of matches per league and season. One way to address this problem would be to identify the affected matches and remove them altogether from the dataset. To realize this, a highly manual and extremely time-consuming procedure would be required.

As described above, our approach to feature modeling could be subject to gaps if we considered very long time series that cut across seasons. For the Challenge, we encountered the unexpected situation that the prediction set featured a couple of teams whose team performance time series were limited to only a handful of matches. This situation came about because the affected teams have never been a member of the league before and that the seasons in those leagues commenced only a few weeks prior to the Challenge deadline. So for those teams, the features are based on very short time series. Feature value estimates derived from such short time series may not be very robust.

In this study, we deal with the home advantage aspect in two ways. First, we use two features sets, one containing total, home as well as away, and one containing only total team performance features. The former feature set, referred to as "homeaway", provides an explicit representation of the home advantage. Second, for total team performance features (which merges home and away performances into a total performance category), we have a less explicit representation of the home advantage, as the home team's features always come before the away team's features in the feature training set.

The feature engineering concept presented here could produce a certain amount of "information leak" and therefore lead to a slight underestimation of the generalization error produced by machine learning models. In our feature modeling approach, each match is characterized by the past performances of the two teams facing off in a match. Once the features for all matches in the training set are determined, we usually split the data randomly into training and test set in order to estimate the model prediction performance. This could lead to the odd situation that some aspects of a team's future performance (seen from the data of the current match in the training set) could be leaked from the test set to the training set. By adopting various cross-validation procedures, we hope to mitigate the effect that this has on the overall modeling process. We accept that this may result in a slight underestimation of the model's generalization error. However, the feature calculation itself does not have this problem, as the features are strictly based on prior performance. Also note that the information leak is not possible for the Challenge prediction set, which is essentially the ultimate "test set", as the outcomes of the matches in the prediction set are not known at the time of model construction.

5.5 Feature generation

Our feature engineering procedure is firmly rooted in the somewhat counter-intuitive super league approach outlined above. This means that we compute features solely on a league-by-league basis, and we merge all seasons of a league covered in the datasets into a super league. Based on our reflections on feature modeling and some initial data analysis and model prototyping, we explored several feature sets. We ended up with two concrete choices. Each of the two chosen feature sets rests on three of the eight basic team performance categories shown in Table 2: *Scored* and *Conceded* goals and *League* success. Our choice is justified as follows. First, the goals that the teams score and concede are the only two primary quantitative variables in the Challenge data. They provide a direct competitive view of the teams facing off in a match. All other team performance categories are derived from these two variables. Second, the team performance category *League* success provides a context of the two opposing teams in the league as a whole. In a sense, the *League* success category "qualifies" the goal performance categories.

The two feature sets upon which our modeling and evaluation is based are referred to as *total* and *homeaway* feature sets, respectively. Total features are based on the combined home and away matches that a team has played. Homeaway team features are computed separately for the home and away matches of the team.

Let T denote either the home or away team facing off in a match. Further, let $v \in \{h, a, t\}$ refer to either the home (h), the away (a), or the total (t) (i.e., home and away combined) "venue", respectively, where the team's performance was made. Then, we define the feature $f_v(T, n_v)$ as the mean value of a team's performance over a team's n_v recent matches at the corresponding venue v as shown in Equation 4.

$$f_v(T, n_v) = \frac{1}{n_v} \sum_{i=1}^{n_v} f_{v,i}(T) \tag{4}$$

To illustrate this calculation, consider the current fixture Man City against Liverpool on 01/04/2023 depicted in Table 4. If feature f represents the goals scored (expressed as an average per match) and T stands for team Liverpool, we determine the feature values $f_h(\mathit{Liverpool}, n_h) = 3.67$ and $f_a(\mathit{Liverpool}, n_a) = 0.67$ over the $n_a = n_a = 3$ recent home and away matches, respectively, and $f_t(\mathit{Liverpool}, n_t) = 3.00$ over the $n_t = 3$ total matches of Liverpool. If feature f represents Liverpool's match winning performance (expressed as average wins per match) over the $n_h = n_a = 3$ recent matches, we obtain $f_h(\mathit{Liverpool}, n_h) = 1.00$ (3 wins out of recent 3 home matches), $f_a(\mathit{Liverpool}, n_a) = 0.33$ (1 win out of 3 away matches) and $f_t(\mathit{Liverpool}, n_t) = 0.66$ (2 wins out of $n_t = 3$ recent matches).

Conceptually, we should always use an even number for the n_t recent total performances of a team and $n_h = n_a = n_t/2$ for the n_h and n_a recent home and away performances of a team. The reason for this choice is explained by the match schedule adopted in most leagues, where normally a team plays in an alternating fashion on the home and away venues. Thus, focusing on even values for n_t, we generally make sure that we capture the same number of home and away matches of a team. Using $n_t/2$ for both n_h and n_a has two reasons. First, this choice makes sure that the time period covered by n_t is roughly the same as the combined time period covered by the $n_h = n_t/2$ and $n_a = n_t/2$ recent matches. Second, the matches of a team covered by n_t recent total and $n_h = n_t/2$ and $n_a = n_t/2$ combined are normally the same.

In the remainder of this text, we use n sometimes in a generic way to refer to the number of recent matches independent on venue, and sometimes it is implied that n refers only to n_t, the total team performances (Table 7).

For the *League* success feature, which represents the average league position of a team over n recent matches, we do not use the absolute ranks from the league table to compute the aggregate (average) feature values. Absolute rank positions could be used if the data were kept separate from other league data for the entire modeling cycle. Since we combine the data from all leagues for the homeaway feature set (meta league), using absolute ranks would be a problem because the number of teams in a league varies. To address this issue, we first calculate the normalized rank for each team and then determine the average over n recent matches. The normalized rank of a team is expressed as a value from the unit interval. A team ranked first has a normalized rank of 1, and a team ranked last has a normalized rank of 0.

Given the absolute rank $R(T, N, n)$ of a team T in a league table derived from the n recent matches of N teams, the team's normalized rank $r(T, N, n)$ is calculated using Equation 5. Notice that we may apply this calculation either to home or away matches only, or to all (total) matches, to make the result dependent on the venue in the same way as the features discussed above. Here, for conciseness, we present only the generic formula.

$$r(T, N, n) = \frac{N - R(T, N, n)}{N - 1} \tag{5}$$

To illustrate Equation 5, consider Table 6. The table depicts two league tables covering the top 7 ranks prior to the match between Man City and Liverpool on 01/04/2023 (see also Tables 3 and 4). The top league table covers all matches from the start of the season 2022/23 to 19/03/2023. The bottom league table is based on the $n = 6$ most recent matches of each team prior 01/04/2023. The total average normalized rank, $r(ManCity, 20, 27)$, of Man City over the entire season up to 19/03/2023 was 0.95 (based on 27 matches), that of Liverpool, $r(Liverpool, 20, 26)$, only 0.68 (based on 26 matches). However, when we consider only the $n = 6$ most recent matches, we see that Liverpool fared much better than for the entire season: $r(ManCity, 20, 6) = 0.95$ and $r(Liverpool, 20, 6) = 0.89$, respectively.

With these considerations, we now present the two feature sets chosen for this study: the *total* and *homeaway* feature sets, respectively. The total feature set consists of the six features depicted in the top six rows, and the homeaway feature set consists of all 18 features depicted in Table 7.

For the total feature set, n recent matches of the teams are used to compute the aggregated values for goals scored, goals conceded, and normalized rank.

Table 6: League tables from rank 1 to 7 of ENG1 2022/23 season. Top: Based on all matches of a team from start of season. Bottom: Based on the 6 most recent matches of each team. (R=Absolute rank, Pld=Matches played, Scr=Goals scored, Con=Goals conceded, r=Normalized rank, scr=Average goals scored, con=Average goals conceded

Team	R	Pld	Scr	Con	r	scr	con
Arsenal	1	28	66	26	1.00	2.36	0.93
Man City	2	27	67	25	**0.95**	2.48	0.93
Man United	3	26	41	35	0.89	1.58	1.35
Newcastle United	4	26	39	19	0.84	1.50	0.73
Tottenham Hotspur	5	28	52	40	0.79	1.86	1.43
Brighton	6	25	46	31	0.74	1.84	1.24
Liverpool	7	26	47	29	**0.68**	1.81	1.12

Team	R	Pld	Scr	Con	r	scr	con
Arsenal	1	6	19	5	1.00	3.17	0.83
Man City	2	6	14	4	**0.95**	2.33	0.67
Liverpool	3	6	13	1	**0.89**	2.17	0.17
Brighton	4	6	9	4	0.84	1.50	0.67
Man United	5	6	9	10	0.79	1.50	1.67
Tottenham Hotspur	6	6	11	9	0.74	1.83	1.50
Aston Villa	7	6	10	8	0.68	1.67	1.33

For example, given the ENG1 match on 01/04/2023 between Man City and Liverpool (see also Tables 3 and 4), the values for all six features based on the $n = 6$ recent total matches are as follows (see bottom part of Table 6): $scr_{t,6}(ManCity) = 2.33$, $con_{t,6}(ManCity) = 0.67$, $r_{t,6}(ManCity) = 0.95$, $scr_{t,6}(Liverpool) = 2.17$, $con_{t,6}(Liverpool) = 0.17$ and $r_{t,6}(Liverpool) = 0.89$.

The homeaway features set includes the total feature set (upper part of Table 7) based on the teams' n recent total matches as well as the aggregated values for goals scored, goals conceded, and normalized rank of each team over $n/2$ recent home and away matches, respectively. The reason to have both the total as well as the home and away records in this feature set is that we capture both the total performance and have also an explicit representation of the teams' home and away performance. Moreover, we strike a compromise between the venues and time periods covered for total and home and away team performances. The basic intention of such a redundant feature representation is that the learning algorithm should find the optimal balance between these important soccer aspects.

The value for the total features based on n recent matches is not necessarily a combination of the two sets of $n/2$ matches for home and away matches of a team. To illustrate this point, consider Tables 3 and 4. The $n = 6$ recent total matches of Liverpool (Table 4) cover exactly Liverpool's $n/2 = 3$ home and $n/2 = 3$ away matches. Thus, the aggregated features of the recent three home

Table 7: Top 6 rows: Total feature set consisting of 6 features based on all (home and away combined) n recent matches of a team. All 18 rows: Homeaway feature set adding features separately calculated for a team's $n/2$ recent home and away matches, respectively.

Feature	Description
$scr_{t,n}(H)$	Home team's average goals scored over n recent *total* matches
$con_{t,n}(H)$	Home team's average goals conceded over n recent *total* matches
$r_{t,n}(H)$	Home team's average normalized rank over n recent *total* matches
$scr_{t,n}(A)$	Away team's average goals scored over n recent *total* matches
$con_{t,n}(A)$	Away team's average goals conceded over n recent *total* matches
$r_{t,n}(A)$	Away team's average normalized rank over n recent *total* matches
$scr_{h,n/2}(H)$	Home team's average goals scored over $n/2$ recent *home* matches
$con_{h,n/2}(H)$	Home team's average goals conceded over $n/2$ recent *home* matches
$r_{h,n/2}(H)$	Home team's average normalized rank over $n/2$ recent *home* matches
$scr_{h,n/2}(A)$	Away team's average goals scored over $n/2$ recent *home* matches
$con_{h,n/2}(A)$	Away team's average goals conceded over $n/2$ recent *home* matches
$r_{h,n/2}(A)$	Away team's average normalized rank over $n/2$ recent *home* matches
$scr_{a,n/2}(H)$	Home team's average goals scored over $n/2$ recent *away* matches
$con_{a,n/2}(H)$	Home team's average goals conceded over $n/2$ recent *away* matches
$r_{a,n/2}(H)$	Home team's average normalized rank over $n/2$ recent *away* matches
$scr_{a,n/2}(A)$	Away team's average goals scored over $n/2$ recent *away* matches
$con_{a,n/2}(A)$	Away team's average goals conceded over $n/2$ recent *away* matches
$r_{a,n/2}(A)$	Away team's average normalized rank over $n/2$ recent *away* matches

and three away matches correspond directly to the aggregated features over $n = 6$ recent total matches. This is different for Man City, though (Table 3). While the $n = 6$ recent total matches include the $n/2 = 3$ recent away matches of Man City, Man City's three recent home matches are not fully covered by the $n = 6$ total matches (only 2 of these 3 matches are included). Therefore, the total feature aggregates are not directly obtained from the home and away aggregates.

Finally, the two feature sets (total, homeaway) used in our study depend on the number n of recent matches used to compute the feature aggregate (mean) values. In our modeling approach, n is viewed as a hyperparameter. In order to find an optimal value for n, we have employed two procedures: Pearson's correlation and k-nearest neighbor (k-NN).

To determine the optimal n, we generated the full homeaway feature set consisting of 18 features (Table 7) for each of the 51 super leagues from the training set for all n-values from $n = 9$ to $n = 100$. This means, for each of the 51 super leagues, we have generated 92 datasets, each containing 18 features based on the corresponding n-value. In these datasets, the total features are computed based on the n recent total, $n/2$ recent home and away team performances, respectively. For each of the 51 collections of 92 n-dependent feature datasets, we determined the optimal values of n using two approaches (Pearson, k-NN) as follows.

The *Pearson* correlation approach was realized first because of its conceptual simplicity and computational efficiency. This meant that some of our machine learning models could be completed and deployed in time for the Challenge deadline. For each of the 51 super league training sets, each consisting of 92 n-specific feature training sets, we computed the following two "derived" features:

1. Difference of the average total goal difference: the home team's average total goal difference minus the away team's average total goal difference, $\Delta gdi_{t,n}(HA) = gdi_{t,n}(H) - gdi_{t,n}(A)$.
2. Difference of the average total goals scored: the home team's average total goals scored minus the away team's average total goals scored, $\Delta scr_{t,n}(HA) = scr_{t,n}(H) - scr_{t,n}(A)$.

We chose these two features after some experimentation. The difference of the (average) goal difference, $\Delta gdi_{t,n}(HA)$, captures the combined goal scoring and conceding performance of the two teams facing off in a match. A positive value is an indicator for the home team to score more goals than the away team in the current match (i.e., a home win). A negative value suggests that the away team will score more goals than the home team (i.e., an away win). A value near zero points to the same number of goals scored by each team (i.e., a draw). The difference of the (average) goals scored, $\Delta scr_{t,n}(HA)$, provides a complementary and somewhat redundant view. However, it captures one of the most essential performance dimensions of a soccer team, namely the ability to score goals. Similar to $\Delta gdi_{t,n}(HA)$, a positive value favors a win of the home and a negative value a win of the away team, respectively, and a value near zero is an indicator for a draw. Notice, both derived features are expected to be positively correlated with the observed goal difference in soccer matches.[1] We exploit this positive correlation property in the Pearson approach to determining an optimal recency value $n_{Pearson,L}$ for a super league L. This is done by computing the Pearson correlation of the sum of $\Delta gdi_{t,n}(HA)$ and $\Delta scr_{t,n}(HA)$ with the observed goal difference $\Delta s(H, A)$ across each n-specific datasets in a given league. For example, for the ENG1 super league, we determined the optimal value as $n_{Pearson}(ENG1) = 60$. Table 8 shows the statistics of the n values for the 43 super leagues in the prediction set that were subject to the final Challenge evaluation. Notice that for two super leagues (CHL1 and POR1), the maximum preset range value of $n = 100$ was reached.

The k-NN approach to determine the optimal n for each super league was performed after the deadline of the Challenge. Here, we did not use derived features but employed all six total features (top 6 rows in Table 7). Also, only the 44 super leagues featuring in the Challenge prediction set were subject to

[1] The goal difference of a single match, $\Delta s(H, A)$, is defined in an "asymmetric" way as follows: $\Delta s(H, A) = s(H) - s(A)$, where $s(H)$ and $s(A)$ denote the observed goals scored by the home (H) and away (A) team, respectively.

this procedure, because the subsequent modeling phase was performed separately for each super league. For each of the 44 super leagues, each of the 92 n-specific datasets was used to learn the optimal k in a leave-one-out cross-validation procedure. All k-values from 3 to 350 were tested. The datasets corresponding to the values of n that produced the lowest error (RPS_{avg} and RMSE, respectively) were selected for downstream modeling. This process resulted in an optimal n and k value both for score and result prediction for each super league. For example, for the ENG1 super league, we determined the optimal value as $n_{kNN,Score}(ENG1) = 42$ and $n_{kNN,Result}(ENG1) = 44$, respectively. The latter is illusrated in Figure 2. Table 8 shows some statistics for the n values for both the result and score predictions derived with the k-NN approach across all 44 super leagues. The maximum value of $n = 100$ was reached once for result (POR1 league) and for score (CHL1 league).

Table 8: Statistics from determining the optimal value for n using the Pearson correlation and k-NN approaches.

Statistic	$n_{Pearson}$	$n_{kNN,Result}$	$n_{kNN,Score}$
Minimum	13	13	15
Maximum	100	100	100
Mean	55.98	50.98	49.07
Standard deviation	25.32	24.44	23.06

The Challenge training set consists of 302 691 matches. After generating the feature sets (homaway and total) separately for each of the 51 super leagues in the training set, the total number of matches were reduced to 292 325. The reason for this is that the time series of each team in a super league starts at some point in all the seasons covered in a super league. This means that for the first few matches of each team there is only a handful of recent matches in the database. Thus, after the calculating the features for a super league, we discard those matches in which one or both teams have less than 6 prior matches. This leads to a loss 10 366 (3.43%) matches from the training set.

6 Overview of reference models and machine learning models

Using the feature datasets (total and homeaway) generated by the framework, we have investigated various models for both Challenge tasks, that is, score and result prediction. These models can be divided into two groups: machine learning and reference models. The machine learning models are designated by the prefix “DeepFoot”, which was our chosen team name. The machine learning models are the following:

– DeepFoot-KNN and DeepFoot-KNN2 are k-nearest neighbor models based on the total feature set (top 6 rows Table 7).

– DeepFoot-ANN refers to feedforward artificial neural networks based on the total feature set.
– DeepFoot-OF128-18 and DeepFoot-OF128-6 models, which are based on ordinal forests with 128 trees using the homeaway and the total feature set, respectively (cf. Table 7);
– DeepFoot-NB6 and DeepFoot-NB18 models, which are based on a naive Bayes classifier using the homeaway and the total feature set, respectively;
– DeepFoot-NB-$18_{3|816}$ models, which collectively denotes 816 different naive Bayes classifiers that use all possible combinations of three features from the homeaway feature set;
– DeepFoot-NB-$18_{4|3060}$ models, which collectively denotes 3060 different naive Bayes classifiers that use all possible combinations of four features from the homeaway feature set.

For each super league, the final DeepFoot-KNN models are based on the n-specific datasets identified by the Pearson method. The final DeepFoot-KNN2 and DeepFoot-ANN models are based on the n-specific datasets identified by the k-NN method. All other final models are based on the n-specific datasets identified by the Pearson recency method. But instead of developing separate models for each super league, these models combined the n-specific datasets into a single large meta league dataset to develop a single model for each task.

We also developed three types of reference models:

– Null-1 and Null-N reference models are essentially null models based on the distribution of result probabilities and average scores derived from the original Challenge training set;
– Baseline-1 and Baseline-N reference models use statistics of some team performance categories computed from the original Challenge training set;
– Bookmakers' reference models are based on odds from multiple bookmakers for scores and results of the matches in the prediction set.

Neither the null nor the baseline models use any form of optimization.

We first discuss the three types reference models that we implemented to provide a baseline for the machine learning predictions. Then, we describe the machine learning models that we developed.

7 Reference models

Based on the ENG1 matches with the IDs 148 and 154 from the prediction set, Leeds vs. Liverpool, and Liverpool vs. Nottingham, we illustrate the various predictions from the reference models in Table 9.

7.1 Null models

The null models Null-1 and Null-N derive their predictions on a league-by-league basis. Null-1 considers only the *last* season (which corresponds to the

Table 9: Examples of reference model predictions for two ENG1 matches. $s(H)$ and $s(A)$ denote observed home/away goals; $\hat{s}(H)$ and $\hat{s}(A)$ predicted home/away goals; Result the observed result; $\hat{y}_1$, $\hat{y}_2$ and $\hat{y}_3$ the predicted probabilities for win, draw and loss

Ref. Model	Home Team (H)	Away Team (A)	$s(H)$	$s(A)$	$\hat{s}(H)$	$\hat{s}(A)$	Result	$\hat{y}_1$	$\hat{y}_2$	$\hat{y}_3$
Null-1	Liverpool	Nottingham	3	2	2	1	win	0.48	0.24	0.28
Null-1	Leeds	Liverpool	1	6	2	1	loss	0.48	0.24	0.28
Null-N	Liverpool	Nottingham	3	2	2	1	win	0.46	0.25	0.29
Null-N	Leeds	Liverpool	1	6	2	1	loss	0.46	0.25	0.29
Baseline-1	Liverpool	Nottingham	3	2	2	1	win	0.45	0.27	0.27
Baseline-1	Leeds	Liverpool	1	6	1	2	loss	0.27	0.25	0.47
Baseline-N	Liverpool	Nottingham	3	2	2	1	win	0.51	0.28	0.21
Baseline-N	Leeds	Liverpool	1	6	1	2	loss	0.29	0.23	0.48
Bookmakers	Liverpool	Nottingham	3	2	2	0	win	0.81	0.13	0.06
Bookmakers	Leeds	Liverpool	1	6	1	2	loss	0.22	0.22	0.55

season of the matches in the prediction set) of each league in the training set. Null-N takes into account *all* seasons of a given league covered in the training set. Theses choices, last and all seasons per league, are two obvious candidates. We could have performed certain analyses to determine the optimal number of seasons (or recent matches per team) to find some optimal prediction, but we wanted to avoid any type of learning or optimization in the null and baseline reference models. Each of the two null models performs the same type of calculation to predict the scores and results in the prediction set, the only difference is the number of seasons covered (last and all).

The *score* prediction null models simply take the average number for the goals scored by the home and away team over the considered season(s) and round these to the nearest integer. For example, for the Null-1 model, we consider the most recent (incomplete) season of the ENG1 league in the training set. In this season, the average goals scored by the home and away teams were 1.62 and 1.13, respectively. Rounding these to the next integers, we obtain 2 and 1, respectively. Thus, the Null-1 model predicts a 2-1 score for all ENG1 matches in the prediction set (see top two rows in Table 9). The Null-N model works in a similar way, except that it processes all 23 ENG1 seasons covered in the training set and obtains the average scores of 1.52 and 1.16, respectively. Like the Null-1 model, the Null-N (after rounding) predicts a 2-1 home win for all ENG1 matches in the prediction set (Table 9 rows 3 to 4).

Similar to the score prediction null models, the *result* prediction null models simply determine the proportion of home wins, draws, and away wins in the corresponding training set (last season for Null-1 and all seasons for Null-N model) to estimate the win, draw and loss probabilities. For example, for all ENG1 matches in the prediction set we get the following result prediction for win, draw, loss: $(0.48, 0.24, 0.28)$ from the Null-1 and $(0.46, 0.25, 0.29)$ from the Null-N model, respectively (top 4 rows in Table 9).

7.2 Baseline models

Like the null models, the two baseline models we employed consider the *last*, i.e., most recent season (Baseline-1), and *all* seasons (Baseline-N) separately for each league in the training set. However, unlike the null models which predict the same outcome for all prediction set matches within a given league, the baseline models take into account the performances of each team to predict the outcome of each match in the prediction set individually.

For *score* predictions, the baseline models determine the average goals scored and conceded of each team in the corresponding training set and calculate the average expected (predicted) score.

Let H denote the home team and A denote the away team facing off in a match. Furthermore, let $scr_t(H)$ and $scr_t(A)$ denote the total number of goals scored, and $con_t(H)$ and $con_t(A)$ the total number of goals conceded (over the seasons covered in the training sets) by the home and away teams, respectively. The terms $n_t(H)$ and $n_t(A)$ refer to the total number of matches each team has played. Then, the predicted match *score*, expressed as the pair $(\hat{s}(H), \hat{s}(A))$, is calculated using Equations 6 and 7. The notation $\lfloor \rceil$ denotes the rounding to the next integer.

$$\hat{s}(H) = \left\lfloor \frac{1}{2} \left(\frac{scr_t(H)}{n_t(H)} + \frac{con_t(A)}{n_t(A)} \right) \right\rceil \tag{6}$$

$$\hat{s}(A) = \left\lfloor \frac{1}{2} \left(\frac{con_t(H)}{n_t(H)} + \frac{scr_t(A)}{n_t(A)} \right) \right\rceil \tag{7}$$

Equations 6 and 7 apply to both Baseline-1 and Baseline-N. The predictions that these models make are different because they are using different subsets from the training set. Rows 5 to 8 in Table 9 illustrate the score predictions of the two baseline models for two ENG1 league matches. Unlike the null models, the baseline models do not predict the same score for all matches within a league. Also, in the case of the two matches, both baseline models agree on the identical final (rounded) score per match.

The *result* prediction of the baseline models follows a similar rationale as the score prediction. Instead of looking at the goals, the baseline models for result prediction calculate the proportions of the three possible result categories (win, draw, loss) based on the teams' winning, drawing, and losing performance in the corresponding league-specific training sets.

Let H and A refer to the home and away team, respectively, facing off in a match. Further, let $win_t(H)$, $drw_t(H)$ and $los_t(H)$ denote the total number of matches the *home* team, and $win_t(A)$, $drw_t(A)$ and $los_t(A)$ the total number of matches the *away* team has won, drawn and lost, respectively, within the considered seasons in the corresponding training set. The terms $n_t(H)$ and $n_t(A)$ refer to the total number of matches of each team in the training set. Then, the predicted match *result*, expressed as the probability vector $(\hat{y}_1, \hat{y}_2, \hat{y}_3)$, is calculated using Equations 8 to 10. In the equations $\hat{y}_1$ denotes the predicted probability for a win, $\hat{y}_2$ a draw, and $\hat{y}_3$ a loss.

$$\hat{y}_1 = \frac{1}{2}\left(\frac{win_t(H)}{n_t(H)} + \frac{los_t(A)}{n_t(A)}\right) \tag{8}$$

$$\hat{y}_2 = \frac{1}{2}\left(\frac{drw_t(H)}{n_t(H)} + \frac{drw_t(A)}{n_t(A)}\right) \tag{9}$$

$$\hat{y}_3 = \frac{1}{2}\left(\frac{los_t(H)}{n_t(H)} + \frac{win_t(A)}{n_t(A)}\right) \tag{10}$$

Rows 5 to 8 in Table 9 illustrate the result predictions of the two baseline models for two ENG1 league matches. The examples illustrate that unlike the null models' result predictions, the baseline models' result predictions are not identical for all matches within a given league. Also, the two baseline models produce different result predictions in the case of the two example matches.

7.3 Bookmakers' models

We investigated how predictions based on the odds that bookmakers provide would fare in the two tasks of the Challenge. We use the term "bookmakers' model" to refer to the score and result reference models derived from bookmaker odds.

The *score* predictions of the bookmakers' score model are based on the *average decimal odds* from multiple bookmakers.[2] Typically, bookmakers cover all scores involving up to five goals by each team. For example, for the ENG1 match between Man City against Liverpool on 01/04/2023, the five scores with the lowest average decimal bookmaker odds and the derived implied probabilities are depicted in Table 10.

For a given match, we identified the score with the lowest average decimal odds (corresponding to the highest implied probability) as the predicted score for that match. For example, for ENG1 match between Man City and Liverpool, the lowest averaged decimal odds were 8.80 for a 1-1 score (top row in Table 10). Thus, we determined the score of 1-1 as the prediction of the bookmakers' model.

[2] Decimal odds are a common way to express odds in sports betting. They represent the potential return on a bet for every unit staked, including the original stake. Decimal odds are expressed as a decimal number greater than or equal to 1. For example, if one bets \$10 on a 1-1 score for which the decimal odds of 8.80 are given, one receives \$88 in case the prediction is correct.

Table 10: Top 5 lowest average decimal odds from at least 10 bookmakers for ENG1 match Man City vs. Liverpool.

Score	Average odds	Implied probability	# Bookmakers
1-1	8.80	0.1136	12
2-1	9.40	0.1064	12
1-0	10.00	0.1000	13
2-0	10.00	0.1000	11
3-1	14.00	0.0714	10
...	...	...	...

For the bookmakers' *result* predictions, we processed the average odds from multiple bookmakers for the three result classes win, draw, and loss. The mean number of bookmakers' odds per match for the 714 matches in the final prediction set were 13.11, with a standard deviation of 3.23. For 653 of the 714 matches, we had the decimal odds of at least 10 bookmakers. Thus, the odds that we used are thought to be reasonably robust. Given the average bookmaker odds, o_j, for the result category j, we normally obtain an estimate of the "implied" probability, $\hat{y}_j$, as follows: $\hat{y}_j = 1/o_j$, where the index j corresponds to the result categories win, draw and loss, respectively. However, generally probabilities derived in this way are deliberate overestimates to ensure that the bookmakers make a profit. For example, the bookmakers' average decimal odds for the result of the match Leeds vs. Liverpool from the prediction set were given as $o_1 = 4.37$, $o_2 = 4.30$ and $o_3 = 1.74$. Based on these, the sum of the implied probabilities would be approximately 1.04. To account for this effect, and to ensure a probability sum of 1, we calculated the probabilities for the three result classes by dividing the implied probability of each class by the sum of all three implied probabilities.

For a given match, let o_j denote the average decimal odds from multiple bookmakers for the result, such that $j = 1$ corresponds to the result category win, $j = 2$ to draw, and $j = 3$ to loss, respectively. Then the predicted probability, $\hat{y}_j$, for the result category j is calculated using the Equation 11.

$$\hat{y}_j = \frac{1/o_j}{\sum_{i=1}^{3}(1/o_i)} \tag{11}$$

See rows 9 to 10 in Table 9 for an illustration of the score and result predictions made by the baseline models, and Tables 13 and 14 on how these models fared in terms of their prediction performance in the Challenge (Rank column) and in relation to our machine learning models (RMSE and RPS_{avg} columns).

8 Machine learning models

8.1 k-nearest neighbor classifier

The *k-nearest neighbor* (k-NN) algorithm is one of the simplest and oldest supervised learning algorithms (Cover and Hart, 1967; Wu et al., 2008; Berrar et al., 2006). To classify an unknown test case, the k-NN identifies the k nearest neighbors from the training set, that is, those k training cases that are closest to the test case based on some measure of distance or similarity of the predictive features. The main advantages of the k-NN include its conceptual simplicity, its straightforward approach to determine both the score and result prediction from the identified k nearest neighbors, and its approach to learning. Essentially, learning in k-NN takes place when new cases with known outcomes are added to the case or knowledge base. The final crucial step in using k-NN is the optimization of the hyperparamter k. Depending on the size of the data and the concrete tuning approach, this step can be quite expensive computationally.

In this study, we produced two sets of k-NN models, DeepFoot-KNN and DeepFoot-KNN2, each consisting of a score and result prediction model. Both k-NN model sets use the total feature set produced in the feature generation phase for each of the 51 super leagues in the Challenge training set. Because both types of k-NN models as well as the ANN model are trained separately for each super league, only the 44 datasets corresponding to the leagues in the prediction set were needed.

For the DeepFoot-KNN models, which were completed in time for the Challenge deadline, the dataset with the optimal recency depth $n_{Pearson}(L)$, corresponding to the 44 super leagues (L) in the prediction set, was used. For the DeepFoot-KNN2 models, which were completed after the deadline, we used the k-NN algorithm itself to determine two separate recency values for score and result prediction, respectively ($n_{kNN,Score}(L)$ and $n_{kNN,Result}(L)$).

For each of the 44 DeepFoot-KNN models, we had exactly one single training set with six total features, each corresponding to a single recency value $n_{Pearson}(L)$. For these n-specific training sets, we carried out a leave-one-out cross-validation (LOOCV) procedure for all k-valuees from 3 to 350. These k-limits were chosen after some exploratory experimentation. After this process, we had one k-NN model for each of the 51 super leagues each made of the following components: (1) A knowledge base or casebase consisting of the n-specific feature dataset determined by the Pearson method. (2) Optimal k-values, $k_{Score}(L)$ and $k_{Result}(L)$, for predicting match scores and results, respectively. For example, for the ENG1 super league, we used the feature training set corresponding to the recency value $n_{Pearson}(ENG1) = 60$ and determined the two optimal k-values for the score and result model as follows: $k_{Score}(ENG1) = 211$ and $k_{Result}(ENG1) = 154$.

For each of the 44 DeepFoot-KNN2 models, we employed LOOCV to determine both optimal n-values for both score and result prediction, $n_{kNN,Score}(L)$ and $n_{kNN,Result}(L)$, while at the same time determining optimal k-values for score and result, $k_{Score}(L)$ and $k_{Result}(L)$. However, unlike in the DeepFoot-KNN case, which is based on a single training set per super league, we applied LOOCV for all k-values from 3 to 350 for each of the 92 training sets which had been generated for recency values from $n = 9$ to $n = 100$. After this procedure we had the following elements for each of the 44 super leagues: (1) Two knowledge bases or casebases, each consisting of the n-specific feature dataset determined by k-NN for score and result prediction, respectively, based on optimal values of $n_{kNN,Score}L$ and $n_{kNN,Result}(L)$. 2) Optimal k-values, $k_{Score}(L)$ and $k_{Result}(L)$, for predicting match scores and results, respectively. It took approximately two weeks to complete this process for all 44 super leagues on a high-spec PC. For example, for the ENG1 super league, we obtained the following results: (1) Recency values: $n_{kNN,Score}(ENG1) = 42$ and $n_{kNN,Result}(ENG1) = 44$ (for the latter, see also Figure 2). (2) Optimal k-values: $k_{Score}(ENG1) = 59$ and $k_{Result}(ENG1) = 298$.

Since the procedure used for the DeepFoot-KNN2 models is computationally very expensive, the DeepFoot-KNN2 models were not completed in time for the Challenge deadline.

Tables 13 and 14 show the final prediction performance of the k-NN models. We used the K-D algorithm implemented in the R package Fast Nearest Neighbor Search Algorithms and Applications (FNN) to implement the k-NN models.

8.2 Neural networks

Artificial neural networks (ANNs) are a fundamental class of machine learning models. One common type of ANN is the feedforward neural network, also referred to as multilayer perceptron (Rumelhart et al., 1986; Hornik et al., 1989), which is widely used due to its capacity to approximate complex functions, making it suitable for tasks like pattern recognition, classification, regression, and optimization.

We developed two ANN models tailored specifically to the score and result prediction tasks. The models were trained separately for each of the 44 super leagues of the prediction set based on the total feature set consisting of 6 features (top part of Table 7).

The architecture of the ANN *score* model has 6 nodes in the input layer, 2 in the single hidden layer, and 2 in the output layer. Each node in the hidden layer receives 6 inputs from the input layer plus one input from a bias node x_0. The 2 nodes in the output layer receive their inputs from the outputs of the corresponding hidden layer. The 2 output layer nodes represent the decimal score prediction which are rounded to the next integer after the model training is complete.

Let the vector $(x_0, x_1, ..., x_6)|x_i \in \mathbb{R}^+$ define the bias node ($x_0 = 1$) and the six input nodes $(x_1, ..., x_6)$ of the ANN score model, such that $(x_1, ..., x_6)$ correspond to the six total features. Further, let the vector $(w_{0,H}, w_{1,H}, ..., w_{6,H})|w_{i,H} \in \mathbb{R}$ and $a_H \in \mathbb{R}^+$ denote the parameters of the hidden node corresponding to the predicted home score, and the vector $(w_{0,A}, w_{1,A}, ..., w_{6,A})|w_{i,A} \in \mathbb{R}$ and $a_A \in \mathbb{R}^+$ to the predicted away score, respectively. Then, the predicted decimal home $\hat{s}(H) \in \mathbb{R}^+$ and away $\hat{s}(A) \in \mathbb{R}^+$ scores of a match are calculated using the activation functions shown in Equations 12 and 13.

$$\hat{s}(H) = \frac{a_H}{1 + \exp\left(-\sum_{i=0}^{n=6} w_{i,H}\, x_i\right)} \tag{12}$$

$$\hat{s}(A) = \frac{a_A}{1 + \exp\left(-\sum_{i=0}^{n=6} w_{i,A}\, x_i\right)} \tag{13}$$

The values of the six inputs in Equations 12 and 13 refer to the six total features shown in the top part of Table 7. These features always assume positive values including zero. Similarly, the parameters a_H and a_A represent the maximal values for the goals scored and therefore are limited to positive numbers including zero.

The ANN score model predicts a match score consisting of two positive decimal numbers including zero. These decimal scores are used in the training phase to optimize the model parameters using the RMSE criterion. Once training is complete, the final predicted scores are obtained by rounding the decimal scores to the next integer. Thus, the final predicted score is defined as $\hat{s}(H) := \lfloor \hat{s}(H) \rceil$ and $\hat{s}(A) := \lfloor \hat{s}(A) \rceil$.

The *result* ANN model architecture is similar to the score ANN model but uses 3 nodes in the hidden layer and output layer, respectively, each corresponding to one of the three result categories. Furthermore, the three outputs are subject to a softmax conversion to scale the outputs to a proper probability vector. In the training phase, the probability vector for result prediction is evaluated against the observed result, which is represented as a hotvector, using the RPS scoring rule (Constantinou and Fenton, 2012) defined by Equation 2.

Let the vector $(x_0, x_1, ..., x_6)|x_i \in \mathbb{R}^+$ define the bias node $(x_0 = 1)$ and the six input nodes $(x_1, ..., x_6)$ of the ANN result model, such that $(x_1, ..., x_6)$ correspond to the six total features. Further, let the vector $(w_{0,1}, w_{1,1}, ..., w_{6,1})|w_{i,j} \in \mathbb{R}$ and $a_R \in \mathbb{R}$ denote the parameters of a hidden node corresponding to the result category j, such that $j = 1$ denotes the result category win, $j = 2$ draw, and $j = 3$ loss, respectively. Then, the predicted probability, $\hat{y}_j$, for the result category j is calculated using Equations 14 and 15.

$$h_j = \frac{a_j}{1 + \exp\left(-\sum_{i=0}^{6} w_{i,j}\, x_i\right)} \tag{14}$$

$$\hat{y}_j = \frac{\exp(h_j)}{\sum_{i=1}^{3} \exp(h_i)} \tag{15}$$

We adopted a straightforward "ensemble parameter averaging" approach to train the ANN models for score and result prediction. One of our main motivations for adopting this approach was the high variance of the test error in cross-validation. This technique is conceptually different from bagging (Breiman, 1996). In bagging, multiple independent base models are trained, and their predictions are averaged or voted upon. In model averaging, the parameters of the models themselves are averaged to create a single final model (Wortsman et al., 2022). The advantage of this approach is its conceptual simplicity and the expectation that the final model achieves a good bias-variance tradeoff. Another advantage is that there is no final model learning phase based on the entire dataset, as it would be the case in a conventional n-fold cross-validation procedure. The final model is calculated from the average of the parameters from each individual model training fold.

Our approach to model training and validation was as follows. For each of the 44 super leagues featuring in the prediction set, we performed the four main steps based on the same ANN model architecture for score and result prediction, respectively.

Step 1: Repeat Steps 1a to 1c ten times.
Step 1a: Split the data randomly into training and test set.
Step 1b: Train model until at least one convergence criterion is met.
Step 1c: Record train and test errors and individual model parameters.
Step 2: Calculate average training and test errors.
Step 3: Determine final parameters as average over 10-fold training.
Step 4: Use final model to predict prediction set matches.

Step 1a makes sure that each of the ten individual model training runs had a different setup in terms of training and test data. In addition, all model weights were initialized randomly in the line with the parameter constraints mentioned above.

Step 1b trains the model based on a random split of the entire dataset into training and test set using a split ratio of 0.85/0.15. We implemented the back-propagation training algorithm of our ANN models as a stochastic gradient descent (SGD) optimization procedure (Rumelhart et al., 1986; Tian et al., 2023). The SGD method computes the gradient of the cost with respect to the model's output (score or result prediction), which essentially represents how much the output would need to change to minimize the error. This gradient of the cost is propagated backward through the network and used to update each ANN model parameters. SGD performs an update of each model parameter θ_j for each training instance using the following update rule:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta; x^{(i)}; y^{(i)}) \tag{16}$$

where θ_j represents a model parameter, α is the step size or learning rate, and J is the cost function. Unlike batch and mini-batch gradient descent, SGD updates the weights using only a single training example in each iteration, hence the notation $x^{(i)}$ and $y^{(i)}$ denoting the predictors and observed outcome of instance i, respectively. The learning rate hyperparameter in our implementation was fixed to the value $\alpha = 0.25$.

As the algorithm sweeps through the training set, it performs the parameter update for each training sample. Several passes can be made over the training set until the algorithm converges. After some experimentation, we employed two convergence criteria; the training was terminated once one of the two criteria was met. Criterion 1: stop if for 10 iterations the model error has not decreased. Criterion 2: stop after a maximum of 200 iterations has been reached. Most of the time, criterion 1 was reached first.

Step 1c memorizes the optimized parameters of each iteration and the relevant overall model errors. For the score model, the optimized RMSE based on decimal scores was kept for both training and test sets, as well as the RMSE resulting after rounding of the decimal scores to the next integer for the test set. For the result model, the RPS_{avg} for both training and test sets at each iteration was memorized.

In Step 2, the average errors over the ten training/test runs were computed and stored. Step 3 simply calculates the average of all parameters obtained in the individual runs and stores these. These parameter averages constitute the parameters of the final model (Step 4).

8.3 Ordinal forests

Ordinal forests are a special type of regression forests in which the ordinal class variable is treated as a continuous variable (Hornung, 2020). In contrast to regression trees (Breiman, 2001), the class values are replaced by score values that maximize the out-of-bag prediction performance. Training an ordinal forest involves the following steps. First, several thousands of candidate score sets are generated by repeated random sampling. For each score set, a regression forest is fitted for the class values of the target class, and the out-of-bag prediction error is calculated based on a user-defined performance metric. Second, the final score is calculated based on those score sets that resulted in the best out-of-bag performance. Finally, using the final score set for the target class values, a regression forest is fitted. In our ordinal forests models, we used the RPS as the cost function (Epstein, 1969; Constantinou and Fenton, 2012).

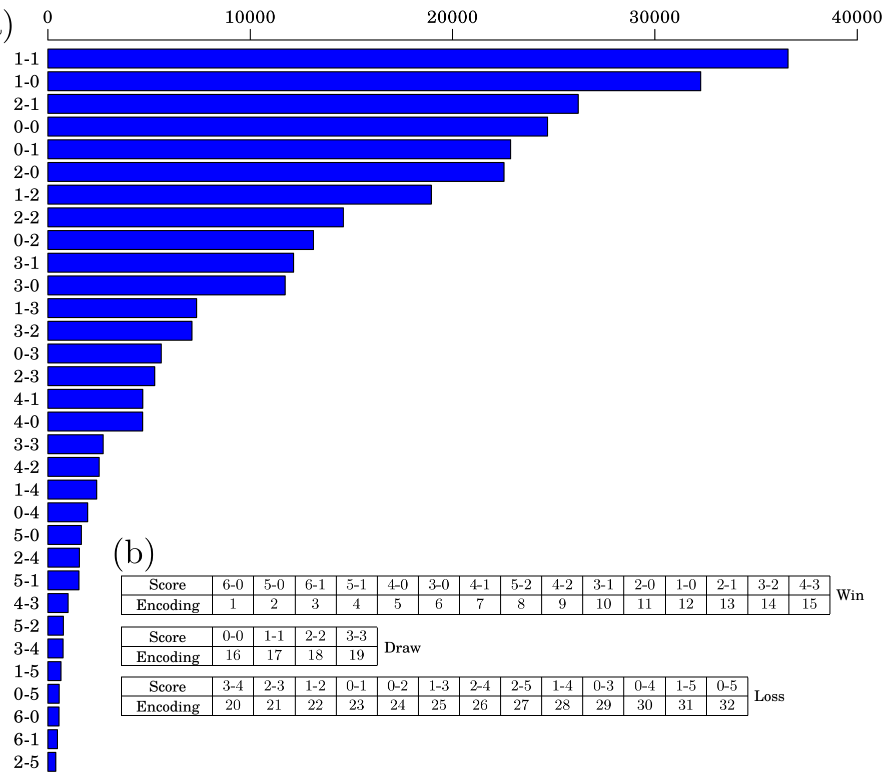
We chose ordinal forests for the following two reasons. First, regression trees, or random forests more generally, have shown remarkable performance across a wide range of applications with tabular data, for which they generally outperform even deep learning (Grinsztajn et al., 2022). Tree ensembles are therefore one of the state-of-the-art algorithms for classification and regression tasks. Second, we required an algorithm that can deal with ordinal class values. In ordinal classification problems, misclassifications can be more or less severe. For example, if the possible class labels are “win”, “draw”, and “loss”, misclassifying a real “win” as “draw” is obviously less severe than misclassifying it as “loss”.

To train the ordinal forest models, we used the feature training set (consisting of 18 homeaway features) based on the optimal n-value from the Pearson method. This data set contains $n = 292\,325$ matches. We pre-processed the data as follows. First, the score was converted into a discrete class label. This resulted in a total of 85 discrete classes, including some extremely rare classes, such as 0-13, which occurred only once. We decided to exclude such extremely unlikely events, as our available computing hardware was insufficient for an effective training of ordinal forests with that many classes. We discarded all matches whose outcomes occurred in less than 293 of 292 325 matches or 0.1%, leaving 289 960 matches with 32 discrete outcomes for further analysis. Figure 1a shows a frequency plot of the classes. By far the most frequent classes are 1-1 and 1-0, making up almost one quarter of all outcomes.

The prediction of match outcomes is of an ordinal nature; clearly, predicting a real 1-1 outcome as 1-0 is better than predicting it as 0-1. Our next problem was therefore to find an appropriate encoding of the scores that adequately reflects the ordinal relation. Finding such an encoding turned out to be more difficult than expected, as there exists no unique ordering of score classes that is consistent with soccer domain knowledge. Among the filtered 32 classes, 6-0 and 0-5 are as far apart as possible, and therefore it makes sense to encode them as 1 and 32. But it is not obvious how to order the outcomes between those two extremes. We eventually decided to encode the outcomes as shown in Figure 1b, as this encoding represents a meaningful ordinal relation.

From the feature training set, we then randomly sampled 30 000 matches (approximately 10%) for the test set and used the remaining 259 960 matches for the training set. The ordinal forest was trained with $n_{\text{score}} = 100$ score sets, from which the 10 best were then selected for the score set to train the final regression forest. For each score set, a regression forest with $n_{\text{tree},1} = 10$ regression trees was built. Using the optimized score set, a larger regression forest with $n_{\text{tree},2} = 128$ trees was fitted to the training set and then applied to the test set. Training an ordinal forest with these parameters took more than 12 hours on a standard PC with 32 GB RAM and seven Intel i7-7700T CPUs. A larger number of trees generally improves model stability, but more than 128 trees led to system crashes. For the Challenge, we therefore did not perform any further fine-tuning of the model parameters. We used the six total features (cf. Table 7, top six rows) for DeepFoot-OF128-6 and 18 homeaway features for DeepFoot-OF128-18.

(a)



(b)

Score	6-0	5-0	6-1	5-1	4-0	3-0	4-1	5-2	4-2	3-1	2-0	1-0	2-1	3-2	4-3
Encoding	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Win

Score	0-0	1-1	2-2	3-3
Encoding	16	17	18	19

Draw

Score	3-4	2-3	1-2	0-1	0-2	1-3	2-4	2-5	1-4	0-3	0-4	1-5	0-5
Encoding	20	21	22	23	24	25	26	27	28	29	30	31	32

Loss

Fig. 1: (a) Frequency of the 32 filtered match score in the Challenge training set. (b) Encoding of the outcomes for ordinal classification.

For each match in the test set, the ordinal forest models produce probabilities for each of the 32 outcomes, which were used to address score and result prediction. The prediction of exact scores was based on the maximum class probability. For example, if class 11 achieved the highest probability, then the outcome 2-0 was predicted (cf. Figure 1b). For the prediction of match results in terms of probabilities of win, draw, and loss, the calculated probabilities were simply added accordingly: the probability of win is the sum of all probabilities for the outcomes 6-0, 5-0, 6-1, ..., 4-3; the probability of draw is the sum of all probabilities for the outcomes 0-0, 1-1, 2-2, and 3-3; and the probability of loss is the sum of all probabilities for the outcomes 3-4, 2-3, 1-2, ..., 0-5. This approach, however, can lead to predictions that at first seem inconsistent. For example, the match Forest Green vs. York City (ENG5 league, played on 02/12/2006) ended 0-1. The ordinal forest produced the highest class probability of 0.1051 for the outcome 0-0, a draw. However, the probabilities of win, draw, and loss are 0.2603, 0.2434, and 0.4963, respectively—the probability of loss is the largest, although 0-0 is the most probable outcome.

Table 11: Results of the ordinal forests on the tests set and prediction set.

	Test Set		Prediction Set	
	RMSE	RPS_{avg}	RMSE	RPS_{avg}
DeepFoot-OF128-6	1.8261	0.2140	1.8805	0.2186
DeepFoot-OF128-18	1.7681	0.2117	1.8277	0.2150

Based on the predictions of the test set, the performance measures RMSE and RPS_{avg} were then calculated and used as estimates of the performance on the prediction set. Then, to predict the matches of the prediction set, the ordinal forest was trained on the entire feature training set consisting of 292 325 matches. Table 11 shows the results on the test set and the prediction set.

DeepFoot-OF128-18 performed slightly better than DeepFoot-OF128-6 on both tasks and both the test set and the prediction set.

8.4 Naive Bayes classifier

The naive Bayes classifier is based on Bayes' theorem and belongs to the family of generative classifiers (Berrar, 2018). By making the naive assumption that the predictive features are statistically independent from each other, the naive Bayes classifier calculates a class posterior probability for each case according to Equation 17,

$$\mathbb{P}(y_j|\mathbf{x}_{\mathrm{i}}) = \frac{\prod_{k=1}^{p} \mathbb{P}(x_k|y_j)\mathbb{P}(y_j)}{\mathbb{P}(\mathbf{x}_{\mathrm{i}})} \tag{17}$$

where $\mathbf{x}_i$ denotes a case consisting of p predictive features, i.e., $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{ip})$. Each case is assumed to belong to exactly one class $y \in \{y_1, y_2, ..., y_c\}$. The predicted class is the maximum a posteriori class, and it is calculated as $\hat{y}$ for the case $\mathbf{x}_i$ as

$$\hat{y} = \underset{y_j}{\text{argmax}} \prod_{k=1}^{p} \mathbb{P}(x_k|y_j)\mathbb{P}(y_j) \tag{18}$$

We included the naive Bayes classifier because it is one of the oldest workhorses of machine learning and has shown remarkable performance compared to more complex models in a range of applications (Webb et al., 2005; Berrar, 2018), even the naive assumption does not hold; clearly, for our feature sets, this is the case.

The naive Bayes classifier was built using the same pre-processed data that we used for the ordinal forests (Section 8.3). Table 12 shows the results on the test set and the prediction set. Only the results of DeepFoot-NB-6 were submitted to the Challenge, as DeepFoot-NB-18 could not be completed before the Challenge deadline.

After the Challenge deadline, we performed further experiments as follows. For the naive Bayes classifier, the only tuning parameter is the number of features. Therefore, from the set of $p = 18$ predictive features, we selected all $\binom{18}{3} = 816$ combinations of 3 features and built 816 naive Bayes classifiers, collectively referred to as DeepFoot-NB-$18_{3|816}$, which we then applied to the test set. From this set of 816 models, the model with the lowest RMSE was selected for *score* prediction, and the model with the lowest average RPS was selected for *result* prediction. We proceeded analogously for all combinations of four features and built and applied $\binom{18}{4} = 3060$ models, collectively referred to as DeepFoot-NB-$18_{4|3060}$. From the set of 3060 naive Bayes models, the two models with lowest average RPS and RMSE on the test set were selected and then used to predict the matches in the prediction set.

Table 12: Results of the naive Bayes classifier on test and prediction set. From DeepFoot-NB-$18_{3|816}$ and DeepFoot-NB-$18_{4|3060}$, the models that performed best on the test set were selected and then applied to the prediction set. Models marked by * were completed after the Challenge deadline.

	Test Set		Prediction Set	
	RMSE	RPS_{avg}	RMSE	RPS_{avg}
DeepFoot-NB-6	1.7798	0.2183	1.8406	0.2247
DeepFoot-NB-18*	2.1309	0.2546	2.2573	0.2689
DeepFoot-NB-$18_{3\|816}$*	1.6895	0.2150	1.7461	**0.2155**
DeepFoot-NB-$18_{4\|3060}$*	1.7023	0.2137	**1.7417**	0.2159

Using only 3 to 4 predictive features improves the performance of the naive Bayes classifier considerably, compared to the performance based on the full set of 18 features. This shows that the computational costs due to the exhaustive search are justified. The worst performance is achieved by the naive Bayes classifier using all 18 features. This result is consistent with our theoretical understanding of the naive Bayes classifier: including correlated features generally leads to a performance degradation.

9 Results

Table 13 shows the results of our machine learning models and reference models for Task 1, the prediction of exact *scores*. Table 14 shows the corresponding results for Task 2, the prediction of *results* in terms of probabilities for the result categories win, draw, and loss. The columns "Feature Set" and "# Features" describe the feature sets that the models were trained on and the corresponding number of features, respectively. The column labeled "League" relates to the league/season combination found in the training set. Recall, each of the 51 leagues featuring in the Challenge training set was processed separately in the feature generation phase by pooling all seasons within the league into a super league. In the model generation phase, there are two basic options for how to use these 51 separate super league feature datasets: (1) Construct the models separately for each of 44 leagues featuring in the prediction set, each corresponding to one of the 51 super league datasets. This means that we discard the data from 7 of the 51 super leagues in the Challenge training set. In Tables 13 and 14, this situation relates to the value *Super League* in the "League" column. (2) Merge all 51 super league datasets into a single dataset referred to as "meta league" (corresponding to the *Meta League* entries in the "League" column). This option yields a much larger training set and exploits the data from all 51 leagues, even though seven of these do not occur in the prediction set. Finally, the value *League* in the "League" column occurs only for the reference models Null-1 and Baseline-1. Here, the baseline predictions are performed separately for each of the 44 leagues featuring in the prediction set but only the data of the last season of each league (corresponding to the season in which the prediction set matches occur) was used. Clearly, the latter approach is very limited in the number of data points that are exploited. Finally, the column "Rank" shows the rank that our models achieved in the leader board that contains all valid submissions to the 2023 Soccer Prediction Challenge.

Overall, our model DeepFoot-KNN2 performed the best for the score prediction Task 1, whereas DeepFoot-NB-18 performed the worst. On Task 2, the overall best performance was achieved by the bookmakers' model, and the worst overall performance was achieved again by DeepFoot-NB-18. Surprisingly, our reference models performed remarkably well on Task 1 where they achieved the 7th, 8th, 9th, and 10th rank.

10 Discussion

Soccer outcome prediction has become a challenging but fascinating new area for machine learning research and development. Here, we presented a new data- and knowledge-driven framework for building machine learning models from readily available soccer data to predict match outcomes. We used our framework to build predictive models for the 2023 Soccer Prediction Challenge.

In Task 1 of the Challenge, prediction of exact *scores*, our k-nearest neighbor model DeepFoot-KNN2 achieved the top performance. Furthermore, DeepFoot-KNN and DeepFoot-ANN were also ranked very high on 3rd and 4th place, respectively. These three models are all based on the super league approach, i.e., for each of the 44 leagues featuring in the prediction set, the data of all seasons of the league were merged into one dataset an processed as if it was a single continuous season. The entire feature engineering and model development process was carried out separately for each super league dataset. All three models used the total feature set consisting of six features based on the combination of home and away matches. Unlike DeepFoot-KNN and DeepFoot-ANN, which are based on the optimal recency value derived with

Table 13: The *score* prediction (Challenge Task 1) models sorted by the leader board position (Rank) among all Challenge participants. Models marked by the star symbol (*) were completed after the Challenge deadline. In total, 26 models are ranked.

Team	RMSE	League	Feature Set	# Features	Rank
DeepFoot-KNN2*	1.6227	Super League	Total	6	1
DeepFoot-KNN	1.6339	Super League	Total	6	3
DeepFoot-ANN*	1.6479	Super League	Total	6	4
Baseline-N	1.6653	Super League	n/a	n/a	7
Baseline-1	1.6682	League	n/a	n/a	8
Null-1	1.6757	League	n/a	n/a	9
Null-N	1.6862	Super League	n/a	n/a	10
DeepFoot-NB-$18_{4\|3060}$*	1.7417	Meta League	Homeaway	4	13
Bookmakers*	1.7433	n/a	n/a	n/a	14
DeepFoot-NB-$18_{3\|816}$*	1.7461	Meta League	Homeaway	3	15
DeepFoot-OF128-18	1.8277	Meta League	Homeaway	18	19
DeepFoot-NB-6	1.8406	Meta League	Total	6	20
DeepFoot-OF128-6	1.8805	Meta League	Total	6	23
DeepFoot-NB-18*	2.2573	Meta League	Total	6	26

Table 14: The *result* prediction (Challenge Task 2) models sorted by the leader board position (Rank) among all Challenge participants. Models marked by the star symbol (*) were completed after the Challenge deadline. In total, 28 models are ranked.

Team	RPS_{avg}	League	Fature Set	# Features	Rank
Bookmakers	0.2063	n/a	n/a	n/a	1
DeepFoot-ANN *	0.2113	Super League	Total	6	4
DeepFoot-KNN	0.2117	Super League	Total	6	6
DeepFoot-KNN2 *	0.2122	Super League	Total	6	7
DeepFoot-OF128-18	0.2150	Meta League	Homeaway	18	13
DeepFoot-NB-$18_{3\|816}$*	0.2155	Meta League	Homeaway	3	14
DeepFoot-NB-$18_{4\|3060}$*	0.2159	Meta League	Homeaway	4	16
DeepFoot-OF128-6	0.2186	Meta League	Total	6	19
Baseline-1	0.2199	League	n/a	n/a	21
Baseline-N	0.2234	Super League	n/a	n/a	23
Null-N	0.2238	Super League	n/a	n/a	24
Null-1	0.2242	League	n/a	n/a	25
DeepFoot-NB6	0.2247	Meta League	Total	6	26
DeepFoot-NB18*	0.2689	Meta League	Total	18	28

the Pearson method, DeepFoot-KNN2 is based on the k-NN approach for determining the optimal recency hyperparamter n.

In Task 2 of the Challenge (prediction of *results*), the three models that featured in the top 4 in the score prediction task did also reasonably well: DeepFoot-ANN, DeepFoot-KNN, and DeepFoot-KNN2 were ranked 4th, 6th and 7th, respectively. Surprisingly, DeepFoot-KNN fared a little bit better than DeepFoot-KNN2 in this task. The naive Bayes classifier using the entire homeaway feature set performed worst in both tasks. The main reason is probably that naive Bayes does not take into account the ordinal relation of the classes.

We expected that ordinal forests would perform much better in the 2023 Soccer Prediction Challenge, for the following reasons. Ordinal forests are ensembles of regression trees, which have shown superior performance in a range of applications with tabular training data, even when compared to deep learning (Grinsztajn et al., 2022). One possible explanation for the relatively poor performance might be our chosen encoding of scores as discrete classes (cf. Figure 1b). As ordinal forests were trained with the RPS as objective function, it is crucial that the encoding preserves the relative ordering of similar classes. For example, the scores 6-0 and 0-5 are the two most opposite scores in the considered training set of 32 outcomes. Encoding these scores as 1 and 32 makes sense, as it maximizes the distance between these classes. However, there exists no obviously “correct” ordering of the scores in-between; for example, how should the scores 2-5, 1-4, and 0-3 be ordered and encoded? The goal difference is 3 in each example, but it is not clear which result is a more decisive victory for the away team. There is no total order relation between scores. In our ongoing research, we are working on a ranking method for soccer scores. Another possible explanation for the relatively poor performance of ordinal forests is that we discarded those matches with extremely unlikely scores, that is, scores that occurred with a frequency of 0.1% or less, such as 0-7. This filtering discarded 2365 (0.8%) matches with 53 unlikely scores from the training set; hence, the ordinal forest model was unable to predict extremely rare outcomes. The filtering was necessary, however, as training with all 85 classes was not computationally possible with our available hardware.

The super league approach to feature engineering developed in this study seems to contradict common soccer domain knowledge. Yet, it led to highly competitive results in both Challenge tasks. A central element of the super league approach revolves around the recency value n, which determines how many recent team performances should be taken into account to characterize a team prior to a match. Common soccer domain knowledge would suggest that perhaps the recent 5 to 15 matches should be considered. However, the results that we obtained paint a counter-intuitive picture. To illustrate this, consider Figure 2.

Figure 2 illustrates the result from a leave-one-out cross-validation of the result prediction model with the k-NN model for all n values from 9 to 100. The triangles indicate the optimal n-values. For example, for the FRA3 league, the optimal recency value was $n_{FRA3} = 18$, whereas for GER1 it was $n_{GER1} = 64$. It is rather astonishing that for GER1 the recent 64 games of a team would

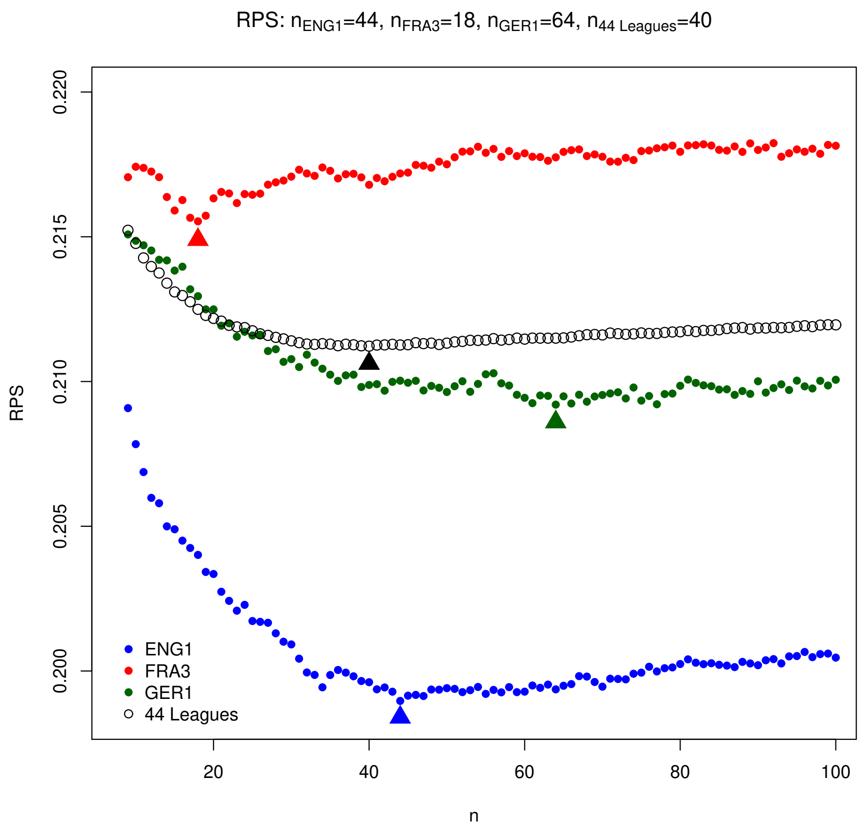


Fig. 2: Illustration of the dependency of the recency value n for *result* prediction (RPS) for $n = 9, 10, ..., 100$. ENG1, FRA3, and GER1 refer to the English Premier League, the French National League, and the German Bundeliga, respectively. *44 Leagues* shows the average for the 44 leagues in the Challenge prediction set.

yield the optimal depth to calculate the features. For GER1, this means that almost two complete seasons covering a period of nearly two years are covered.

Figure 2 also shows how the values for n vary across leagues. The average n-value for the 44 leagues from the prediction set is 40. Notice that this value is different from the mean value of 50.98 in the $n_{kNN,Result}$ column of Table 8. First, the $n_{kNN,Result}$ values in the table are based on all 51 leagues found in the training set. Second, the data for 44 leagues shown in Figure 2 are averaging at each n-value the RPS_{avg} error. Given that the average league size of the 44 leagues of the prediction set is 17.68 teams, each team, on average, would play 33.36 matches per season. This means that, on average, the best

n-value is found well beyond a season's worth of matches played by each team. This result was unexpected.

Because of the surprising but promising results we obtained with the super league approach to feature engineering and model development, we will further investigate this in the future based on datasets that provide more data on a soccer match, including corners, fouls, shots, yelow/red cards, and so on.

Another surprising result from our study relates to the overall predictive performance based on the total and homeaway features depicted in Tables 13 and 14. Since the homeaway features explicitly capture the home advantage *and* include the six total features, one would have expected that this feature set leads to a superior performance, but this was not the case. We have no explanation for this observation. In our future work, we will further investigate this finding by using a wider range of machine learning models.

Another surprising finding of our study is the relative good performance of our simple null models and baseline models. These models outperformed a number of far more complex machine learning models and the bookmakers' model in the score prediction task (Task1, Table 13) and also a small number of machine learning models in the result prediction task (Task 2, Table 14). This is quite surprising, given that the null and baseline models are relatively simplistic, compared to the machine learning models. We did not even give these models the chance to benefit from the recency processing of the data, which we used for the machine learning models. It is quite possible that their performances would have been even better if we had chosen the null and baseline models based on an optimal n-value. There might be a theoretical explanation for this unexpected observation. It is often tacitly assumed that there exists a trade-off between model complexity and predictive performance: the more complex a model is, the higher its expected predictive performance, and vice versa (Rudin, 2019). For a given task, however, there might be countless models whose performances do not differ by a lot, an observation for which Leo Breiman coined the term "Rashomon effect" (Breiman, 2001). Among the set of models with a similar performance, there might be some that are surprisingly simple. There are numerous examples in other domains, for example, bioinformatics (Dudoit and Fridlyand, 2002; Berrar et al., 2006), which illustrate that complex models are indeed not always better (Hand, 2006; Gosiewska et al., 2021).

The bookmakers' model came out top in the result prediction task (Task 2). This was not unexpected, as bookmakers have access to comprehensive match datasets and are able to tap into the wisdom of very large crowds of people—some with profound understanding of soccer—who bet on the outcomes of matches through the betting platforms. Given the minimal information about matches that we used in our study, one could have expected that the margin between the bookmakers' model and the machine learning models would be even much higher, but it wasn't—and this is rather motivating for machine learning researchers.

Considering that the bookmakers' model was the clear winner among all Challenge participants for the result prediction task, it is surprising that the bookmakers' model did not fare too well in the score prediction task. It is difficult to say what the reasons for this observation might be. For the bookmakers' score predictions, we only had the odds for the score with the lowest odd value, so we could not determine if the best score prediction models would have been profitable if their predictions had been to bet on the prediction set matches. As some of our score prediction models fared well against the bookmakers' score model, one might ask the question: "Would one have made a profit by betting money on the scores predicted by our models for the 736 matches from the prediction set?" The answer is: "We do not know." The reason is that prior to the Challenge deadline, we determined the bookmakers' score predictions for each of the 736 matches by identifying only the single score for which the bookmakers offered the lowest odds, which corresponds to the highest implied probability. The odds of all other scores of each match offered by the bookmakers were not collected. After the 736 matches had been completed, the odds were no longer available, so we do not know what the odds for the 736 observed scores were. Therefore, we could not test if the predictions of any of our top-performing score models could have been used in a profitable way in sport betting.

11 Conclusions

Soccer match outcome prediction offers a formidable challenge for machine learning. Despite the seeming unpredictability of match outcomes, our research has demonstrated that it is possible to use readily available match data and machine learning to build predictive models with remarkable performance. One of our main insights is that the key to successful predictions lies in how well domain knowledge can be formalized and included in the modeling process. Our data- and knowledge-driven framework is an important novel contribution to the burgeoning field of machine learning in soccer.

Declarations

– Funding: Not applicable.
– Conflicts of interest/Competing interests: The authors have no competing interests to declare.
– Ethics approval: Not applicable.
– Consent to participate: Not applicable.
– Consent for publication: Not applicable.
– Availability of data and material: The data were provided by the 2023 Soccer Prediction Challenge.
– Code availability: Code will be made available upon request.
– Authors' contributions: Conceptualization: WD, DB, PL; Methodology: WD, DB; Implementation: WD, DB; Writing (original draft): WD, DB, PL; Writing (review and editing): WD, DB, PL.

References

Angelini G, De Angelis L (2017) PARX model for football match predictions. Journal of Forecasting 36(7):795–807

Berrar D (2018) Bayes' theorem and naive Bayes classifier. In: Ranganathan S, Nakai K, Schönbach C, Gribskov M (eds) Encyclopedia of Bioinformatics and Computational Biology, Elsevier, pp 403–412

Berrar D, Bradbury I, Dubitzky W (2006) Instance-based concept learning from multiclass DNA microarray data. BMC Bioinformatics 7(73)

Berrar D, Lopes P, Davis J, Dubitzky W (2019a) Guest editorial: special issue on machine learning for soccer. Machine Learning 108:1–7

Berrar D, Lopes P, Dubitzky W (2019b) Incorporating domain knowledge in machine learning for soccer outcome prediction. Machine Learning 108(1):97–126

Breiman L (1996) Bagging predictors. Machine Learning 24(2):123–140

Breiman L (2001) Statistical modeling: The two cultures (with comments and a rejoinder by the author). Statistical Science 16(3):199–231

Constantinou A, Fenton N (2012) Solving the problem of inadequate scoring rules for assessing probabilistic football forecast models. Journal of Quantitative Analysis in Sports 8:1–14

Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Transactions on Information Theory 13(1):21–27

Dixon M, Coles S (1997) Modelling association football scores and inefficiencies in the football betting market. Applied Statistics 46(2):265–280

Dubitzky W, Lopes P, Davis J, Berrar D (2019) The Open International Soccer Database for machine learning. Machine Learning 108(1):9–28

Dudoit S, Fridlyand J (2002) Introduction to classification in microarray experiments. In: Berrar D, Granzow M, Dubitzky W (eds) A Practical Approach to Microarray Data Analysis, Springer, pp 132–149

Epstein ES (1969) A scoring system for probability forecasts of ranked categories. Journal of Applied Meteorology 8(6):985–987

Gosiewska A, Kozak A, Biecek P (2021) Simpler is better: Lifting interpretability-performance trade-off via automated feature engineering. Decision Support Systems 150:113,556

Grinsztajn L, Oyallon E, Varoquaux G (2022) Why do tree-based models still outperform deep learning on typical tabular data? In: Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022) Track on Datasets and Benchmarks, pp 1–48

Hand D (2006) Classifier technology and the illusion of progress. Statistical Science 21(1):1–15

Hill I (1974) Association football and statistical inference. Applied Statistics 23(2):203–208

Hornik K, Stinchcombe M, White H (1989) Multilayer feedforward networks are universal approximators. Neural Networks 2(5):359–366

Hornung R (2020) Ordinal forests. Journal of Classification 37:4–17

Hubáček O, Šourek G, Železný F (2019) Learning to predict soccer results from relational data with gradient boosted trees. Machine Learning 108(1):29–47

Ievoli R, Palazzo G L Ragozini (2021) On the use of passing network indicators to predict football outcomes. Knowledge-Based Systems 222:106,997

Jurman G (2020) Seasonal linear predictivity in national football championships. Big Data 7(1):21–34

Kundu T, Roy A, Rai C (2021) Predicting english premier league matches using classification and regression. In: Proceedings of International Conference on Communication and Computational Technologies, Springer, pp 555–568

Maher M (1982) Modelling association football scores. Statistica Neerlandica 36(3):109–118

Malamatinos MC, Vrochidou E, Papakostas G (2022) On predicting soccer outcomes in the Greek league using machine learning. Computers 11(133):1–24

Nevill A, Holder R (1999) Home advantage in sport: an overview of studies on the advantage of playing at home. Sports Medicine 28(4):221–236

O'Donoghue P, Dubitzky W, Lopes P, Berrar D, Lagan K, Hassan D, Bairner A, Darby P (2004) An evaluation of quantitative and qualitative methods of predicting the 2002 FIFA World Cup. Journal of Sports Sciences 22(6):513–514

Razali N, Mustapha A, Arbaiy N, Lin PC (2022) Deep learning for football outcomes prediction based on football rating system. In: 10th International Conference on Applied Science and Technology, pp 1–7

Reep C, Benjamin B (1968) Skill and chance in association football. Journal of the Royal Statistical Society, Series A (General) 131(4):581–585

Ren Y, Susnjak T (2022) Predicting football match outcomes with eXplainable machine learning and the Kelly index. URL `https://arxiv.org/abs/2211.15734`, `2211.15734`

Rudin C (2019) Stop explaining black box machine learning models for high stakes decisions and use interpretable models instea. Nature Machine Intelligence 1:206–215

Rumelhart D, Hinton G, Williams R (1986) Learning representations by back-propagating errors. Nature 323:533—536

Stübinger J, Mangold B, Knoll J (2020) Machine learning in football betting: Prediction of match results based on player characteristics. Applied Sciences 10(1):46

Tian Y, Zhang Y, Zhang H (2023) Recent advances in stochastic gradient descent in deep learning. Mathematics 11(3)

Webb G, Boughton J, Wang Z (2005) Not so naive Bayes: Aggregating one-dependence estimators. Machine Learning 58:5–24

Wortsman M, Ilharco G, Gadre S, Roelofs R, Gontijo-Lopes R, Morcos A, Namkoong H, Farhadi A, Carmon Y, Kornblith S, Schmidt L (2022) Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In: Chaudhuri K, Jegelka S, Song L, Szepesvari C, Niu G, Sabato S (eds) Proceedings of the 39th International Conference on Machine Learning, Proceedings of Machine Learning Research, vol 162, pp 23,965–23,998

Wu X, Kumar V, Quinlan J, Ghosh J, Yang Q, Hea M (2008) Top 10 algorithms in data mining. Knowledge and Information Systems 14(1):1–37

Wunderlich F, Weigelt M, Rein R, Memmert D (2021) How does spectator presence affect football? home advantage remains in european top-class football matches played without spectators during the covid-19 pandemic. PLoS ONE 16(3):e0248,590