technische universität dortmund

fakultät statistik

Classic statistical and modern machine learning methods for modeling and prediction of major tennis tournaments

by:

Nourah Buhamra

A thesis submitted in fulfilment of the requirements
for the Doctor of Statistic

of the

Department of Statistic,
TU Dortmund University

Dortmund, April 2025

Amtierender Dekan:

Prof. Dr. Philipp Doebler

Gutachter:

Prof. Dr. Andreas Groll (Technische Universität Dortmund)

Prof. Dr. Markus Pauly (Technische Universität Dortmund)

Tag der Prüfung:

11.06.2025

Abstract

We propose a comprehensive approach for predicting outcomes in Grand Slam tennis tournaments from 2011 to 2022, focusing on the probability that the first-named player would win. Our study includes both regression and machine learning models, evaluated using cross validation and external validation strategies through performance measures such as classification rate, predictive likelihood, and Brier score. Some specific aspects are examined in greater detail: non-linear effects, court-specific abilities and global player abilities. Additionally, we introduce the concept of "newcomer" players to estimate ability effects for individuals participating in only one tournament. Among regression models, spline-based approaches with P-splines yielded positive betting returns while placing fewer bets, suggesting a more conservative and potentially safer approach, confirmed as top performers in 2022 validation, while LASSO models were less effective. Notably, all models outperformed the benchmark model.

For the machine learning (ML) models, we investigate classification trees (CT), random forests (RF), extreme gradient boosting (XGBoost), support vector machines (SVM), and artificial neural networks (ANNs) to model and predict match outcomes over the same period. These models were compared with regression approaches using the same predefined validation strategies. However, within the framework of machine learning, a sequential validation technique, namely the rolling window strategy, was employed to assess model stability and accuracy. The findings highlighted that, among the different validation strategies considered, XGBoost achieved the highest classification rate, linear SVM was the most effective in predictive likelihood, and spline models with Points and Prob covariates consistently delivered the lowest Brier scores, emphasizing the strength of these methods across various validation techniques.

Furthermore, we apply the concept of statistically enhanced learning to improve predictive performance in our models. We incorporate covariates derived from separate modeling approaches, such as the Elo rating, along with covariates based on meaningful mathematical transformations, including two Age-related variables. Evaluating the predictive power of these statistically enhanced covariates involves assessing their contribution to model accuracy, often resulting in improved performance compared to models that rely solely on classic covariates. By integrating advanced statistical techniques, we can enhance the model's ability to generalize to new data. This approach has been successfully applied in football, where team ability parameters derived from separate statistical models have improved predictive

performance.

To enhance interpretability, we employed interpretable machine learning (IML) tools such as partial dependence plots (PDPs) and individual conditional expectation (ICE) plots, particularly for models like RF. These visualization tools, combined with feature engineering, allow us to assess the individual and combined impact of key variables, such as Age and Elo ratings, on prediction accuracy. The results confirm that enhanced variables contribute positively to model performance and provide deeper insights into predictors of match outcomes in sports analytics.

Acknowledgments

They say life is a river, and we must learn to go with the flow and embrace its surprises. Learning to go with the flow would not have been possible without the support of the two most important people in my life. My deepest gratitude goes to my wonderful mother and my soulmate, my incredible husband. Thank you both for being my pillars of strength and encouragement. I am truly grateful for everything you have done to help me succeed.

I extend my heartfelt thanks to my beloved children, whose love has always been a source of strength and inspiration.

I am also profoundly grateful to my supervisor Professor Andreas Groll for guiding me throughout my studies. I deeply appreciate his trust, mentorship, and the opportunity to be one of his students. Thank you for your valuable feedback, for dedicating your time, and for handling essential matters, including the visa process and follow-ups with the Kuwait embassy, which I realize may have been inconvenient. I would also like to sincerely thank Prof. Dr. Markus Pauly for dedicating a substantial amount of time to carefully reading my thesis and providing valuable feedback in the report. I truly appreciate your support throughout this process.

Many thanks also go to my colleagues, Hendrik van der Wurp and Guillermo Briseno. Special thanks to my co-authors, Susanne Brunner and Alexander Gerharz, for their assistance in proofreading my papers and their help with R code. I am deeply thankful for all your help.

This work would not have been possible—or even completed—without the support and confidence of everyone mentioned here. Finally, I am proud to have accomplished this, and if given the chance, I would choose to relive this journey all over again. Thank you all for being a part of this journey.

"The unexamined life is not worth living"

— Socrates

List of main publications

This cumulative thesis is based on the following three manuscripts:

Article 1: Buhamra, N., Groll, A., and Brunner, S. (2024). Modeling and prediction of tennis matches at grand slam tournaments. *Journal of Sports Analytics*, 10(1):17–33.

Contribution of the author:

The author of this thesis played a leading role in the preparation and organization of the manuscript. She was also responsible for implementing the simulation studies and conducting the analysis of the data. S. Brunner was involved in data preparation and preprocessing. Moreover, the co-authors contributed through scientific discussions, feedback, and revision support during the development of the manuscript.

Article 2: Buhamra, N., Groll, A., and Gerharz, A. (2025). Comparing modern machine learning approaches and different forecasting strategies for modeling tennis matches at grand slam tournaments. *Journal of Sports Analytics*. to appear.

Contribution of the author:

The author of this thesis was responsible for the conceptualization, structuring, and writing of the manuscript. She carried out the simulation studies and performed the statistical analyses of the studies. The literature review, methodology selection, and interpretation of the results were also primarily handled by the author. Input from collaborator and supervisor was incorporated during the writing. In particular, A. Gerharz also provided continuous support throughout the development of the manuscript, including advise during the simulation studies, the revision process, and other technical tasks.

Article 3: Buhamra, N. and Groll, A. (2025). Statistical enhanced learning for modeling and prediction tennis matches at grand slam tournaments. arXiv preprint arXiv:2502.01613.

Contribution of the author:

While this manuscript benefited from the guidance of supervisor, the author took a leading role in the development of its structure and content. She was mainly responsible for conducting simulations, analyzing datasets, and drafting the text. Collaborative discussions helped shape the direction of the study, but the implementation and interpretation were carried out independently by the author. The final manuscript reflects her substantial personal contribution to the research.

Contents

Abbreviations

ANNs	Artificial neural networks
CART	Classification and regression tree
cp	Complexity parameter
CT	Classification tree
CV	Cross validation
c-ICE	Centered ICE plot
d-ICE	Derivative ICE plot
GAM	Generalized additive model
ICE	Individual conditional expectation
IML	Interpretable machine learning
LASSO	Least absolute shrinkage and selection operator
ML	Machine learning
MLP	Multilayer perceptron
P-splines	Penalizes splines
PDP	Partial dependence plot
RBF	Radial basis function kernel
RF	Random forest
RSS	Residual sum of squares
SEL	Statistical enhanced learning
SVM	Support vector machine, a machine learning method
XGBoost	Extreme gradient boosting

Part I

Summary of thesis work

Chapter 1

Introduction

In recent years, sports data analysis has gained significant attention and importance in both professional and amateur athletics. With advancements in data collection technologies and the increasing availability of sophisticated analytical tools, sports organizations and teams have turned to data analytics to gain a competitive edge. These sports data cover various aspects, such as tracking player movements, analyzing game strategies, monitoring physical conditions, and even predicting injury risks. The impact of data analytics is particularly evident in professional leagues and competitions, where every decision can influence the outcome of a match or season. A notable example is tennis, which is undoubtedly one of the most popular sports globally, played by millions of players, with its most prestigious tournaments attracting significant international attention. This widespread popularity, combined with the potential for profit in betting markets, makes tennis an appealing subject for research.

In tennis, data is gathered from multiple sources, analyzed to identify patterns, to optimize player performance, and to develop winning strategies. By using such data, both players and coaches can assess opponents' strengths and weaknesses, refine their training programs, and make informed decisions during a match.

In this cumulative dissertation, a summary of all my publications in this field (sports analytics on tennis) is presented. Since it is a cumulative dissertation, only concise overviews of the included articles are provided. For exhaustive results and further details, I recommend reading the respective articles themselves which can be found in Part II.

First, we present a statistical approach for modeling and predicting the outcomes of Grand Slam tennis tournaments using regression models. While linear regression is commonly employed for this purpose, it may not be suitable when the relationships between variables are nonlinear or complex. In such cases, non-linear methods, such as splines or decision trees, can be used to develop reliable models without necessarily providing explicit insights into the relationships between the input variables and the target outcomes.

In addition to selecting appropriate models, which is crucial for accurately predicting match results, we also evaluate the proposed models from a betting perspective by assessing the gains or losses of the centrality-based specification.

Furthermore, nowadays, machine learning (ML) has shown promising results in sports prediction. Therefore, we propose several machine learning approaches to accurately predict tennis match outcomes. Within this context, the dissertation also explores how interpretable machine learning (IML) tools can be applied to sports analytics, focusing on the ability to understand and interpret the predictions made by machine learning models. IML emphasizes building models that not only make accurate predictions but also provide explanations or justifications for their decisions.

Finally, we provide examples of how statistical techniques are applied to tennis matches at Grand Slam tournaments, including feature engineering to enhance predictive modeling and derive insights from tennis data. Thus, the aim of this work is to investigate which model approaches yield the most accurate and interpretable results.

State of the art in tennis analytics

Nowadays, there is growing interest in predicting sports match outcomes using statistical and machine learning methods. Recent studies highlight the increasing application of machine learning techniques in tennis analytics, aiming to enhance predictive accuracy and to better understand match dynamics.

In Bunker et al. (2024b), a comparative evaluation of Elo ratings and machine learning-based methods for tennis match result prediction is presented. The authors evaluate five ML models: Alternating Decision Trees (ADTrees), Logistic Regression, Support Vector Machines (SVMs), Random Forests (RFs), and Artificial Neural Networks (ANNs). The findings indicate that ADTrees and Logistic Regression achieve higher accuracies than Elo ratings and comparable accuracies to predictions derived from betting odds. Notably, ADTrees provide an interpretable decision tree model that accommodates variations in the average betting odds difference threshold, highlighting their potential in this domain. The study suggests that ADTrees, with their balance of accuracy and interpretability, could be particularly beneficial for applications requiring both high predictive performance and transparency. Lv et al. (2024) explore the impact of momentum in tennis matches, with a particular focus on the 2023 Wimbledon men's final between Carlos Alcaraz and Novak Djokovic. The authors developed a Composite Bayesian Regression Framework (CBRF), integrating CatBoost and Random Forest regression models to predict momentum shifts throughout the match. Their analysis revealed significant non-zero autocorrelation coefficients, indicating a strong correlation between momentum and match success. Additionally, the study proposed winning strategies based on key influencing factors and conducted predictive analyses for six specific time intervals. The models demonstrated strong generalization capabilities when applied to women's matches, different tournaments, and various playing surfaces.

Candila and Palazzo (2020) investigated the application of artificial neural networks (ANNs) to predict the outcomes of men's tennis matches and assess the profitability of various betting strategies based on these predictions. The study not only demonstrates ANN's superior predictive performance compared to traditional models but also highlights

its practical applicability in developing profitable betting strategies. Their work contributes to the broader field of sports analytics by illustrating the advantages of integrating advanced machine learning techniques into predictive modeling and decision-making processes.

LeCun et al. (2015) provided a comprehensive overview of deep learning, particularly focusing on artificial neural networks (ANNs) and their advancements. Their work highlighted how deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have revolutionized various fields, including computer vision, speech recognition, and natural language processing. They discussed the underlying principles of neural networks, optimization techniques, and the role of large-scale datasets and computational power in improving model performance. The paper also explored practical applications and the potential future developments of deep learning in artificial intelligence. Additionally, Hastie et al. (2009) and James et al. (2013) provide comprehensive introductions to statistical learning methods, including ANNs.

Another study highlighting the growing application of ANNs and deep learning in tennis analytics is Almarashi et al. (2024). This study evaluated the effectiveness of the Neural Network Auto-Regressive (NNAR) model in forecasting tennis players' performance indicators, such as wining probability, aces, double faults, and game dominance. The researchers compared the NNAR model with traditional linear stochastic time series models, finding that the NNAR model outperformed others based on metrics like Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). The study emphasized the importance of considering time-dependent performance measures over solely relying on official rankings, suggesting that players can utilize these insights to enhance future performance.

The ANNs are also used for injury forecasting, strategy optimization, and even real-time decision-making during matches, helping coaches and analysts gain deeper insights into the game. For Example, Sun et al. (2023) employs a deep neural network leveraging Long-Short-Term Memory (LSTM) cells to predict tennis match outcomes based on various statistics. De Seranno (2020) utilizes neural networks to forecast the probability of winning in tennis matches, incorporating a vast number of input variables to handle non-linear relationships effectively. Sampaio et al. (2024) explore the role of machine learning in their research, including neural networks, to enhance tennis performance through psychological state monitoring, talent identification, match outcome prediction, spatial and tactical analysis, as well as injury prevention. Fernando et al. (2019) presents a novel framework for predicting shot location and type in tennis, incorporating neural memory modules to model the episodic and semantic memory components of a tennis player.

In Hovad et al. (2024), the authors explored the application of deep learning techniques to classify tennis actions from video data. Utilizing the Slow Fast deep learning architecture, they trained three models of varying sizes on the THETIS tennis dataset. The best-performing model achieved a generalization accuracy of 74%, demonstrating the potential of deep learning in accurately identifying specific tennis actions. The study also conducted an error

analysis and discussed limitations in current publicly available tennis datasets, providing directions for future improvements in tennis action classification.

Zhang et al. (2024) tackles the challenge of assessing and predicting tennis players' scoring abilities using in-match data. The authors present a comprehensive approach that includes modeling real-time scoring capabilities, establishing a predictive framework, and enhancing model applicability. Using data from the 2023 Wimbledon men's singles final, they apply the Entropy Weighting Method (EWM) and Analytic Hierarchy Process (AHP) to calculate weights, followed by the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) to evaluate player performance. To improve real-time scoring assessment, they introduce a Principal Component Analysis Stepwise Regression Model, which combines Principal Component Analysis (PCA) with stepwise linear regression. The model accurately captures players' performance during matches. Additionally, the authors propose a scoring ability migration model based on time series concepts to predict match trends, achieving a prediction accuracy of up to 95.27% in women's tennis matches. This research provides valuable insights for future advancements in tennis match outcome prediction.

The literature overview in the remainder of this section is mainly based on Buhamra et al. (2024) and Buhamra et al. (2025). In recent years, various approaches have also been proposed for the statistical modeling of tennis matches and tournaments, leading to advancements in existing methods for predicting match-winning probabilities. Once individual match outcomes can be predicted, it becomes possible to estimate the winning probabilities for an entire tournament. For example, Clarke and Dyte (2000) utilized the official Association of Tennis Professionals (ATP) rankings to estimate a player's probability of winning using logistic regression. Arcagni et al. (2022) expanded the methodology for rating calculations to determine match-winning probabilities. Their approach incorporated a centrality measure, allowing player ratings to dynamically adjust with each new match. These updated ratings were then used as covariates in a simple logit model. Klaassen and Magnus (2003) utilized a large live dataset from Wimbledon to make predictions during the tournament, making their approach particularly relevant for betting markets. Easton and Uylangco (2010) built upon Klaassen and Magnus' model, comparing it with bookmakers' odds on a point-by-point basis. Their findings confirmed that bookmakers' odds serve as strong predictors of match outcomes for both men's and women's tennis. Gu and Saaty (2019) analyzed Grand Slam, ATP, and the Women's Tennis Association (WTA) matches, combining statistical data with subjective judgments. Their study identified and systematically prioritized multiple factors—both objectively and subjectively—to enhance predictive accuracy. McHale and Morton (2011) introduced a Bradley-Terry-type model to forecast the outcomes of top-tier ATP and WTA competitions. Their approach incorporated surface types (hard court, carpet, clay, and grass) and demonstrated that including match scores, play data, and surface influence improved prediction accuracy compared to ranking-based models. However, they did not account for dependencies between factors.

Ma et al. (2013) applied a logistic regression model to predict match outcomes (win/loss)

based on 16 variables, including player skills, performance metrics, player characteristics, and match conditions. More recently, Yue et al. (2022) proposed a statistical framework for predicting Grand Slam match outcomes, incorporating exploratory data analysis. Their approach introduced new variables using the Glicko rating model, a Bayesian method commonly used in professional chess, to refine prediction accuracy. Del Corral and Prieto-Rodríguez (2010) estimated separate probit models for men and women, using Grand Slam tennis match data from 2005 to 2008. The explanatory variables were categorized into three groups: a player's past performance, physical characteristics, and match characteristics. The accuracy of the models was assessed both in-sample and out-of-sample by computing Brier scores, comparing predicted probabilities with actual outcomes from 2005 to 2008 and the 2009 Australian Open. Additionally, they employed bootstrapping techniques to evaluate the out-of-sample Brier scores for the 2005–2008 data.

Somboonphokkaphan et al. (2009) proposed a method to predict the winner of tennis matches by utilizing both match statistics and environmental data, based on a Multi-Layer Perceptron (MLP) equipped with a backpropagation learning algorithm. MLP, a type of ANN, is an effective technique for solving real-world classification problems, especially when relying on large datasets and capable of handling incomplete or noisy data. Whiteside et al. (2017) developed an automated stroke classification system to quantify the hitting load in tennis, employing machine learning models such as a cubic kernel support vector machine. Additionally, Wilkens (2021) focused on machine learning approaches and expanded upon previous research by applying a broad range of machine learning techniques. He used various models, including neural networks and random forests, along with one of the most extensive datasets in professional men's and women's tennis singles matches. The author also demonstrated that the average prediction accuracy could not exceed approximately 70%.

As well, Gao and Kowalczyk (2021) developed a model that predicts tennis match outcomes with an accuracy exceeding 80%, significantly large than predictions based solely on betting odds. The model identified serve strength as a key predictor of match outcomes. Using a RF classifier, the study emphasizes the importance and effectiveness of simple models in the era of deep learning. Moreover, the model incorporates a wide range of features, capturing physical, psychological, court-related, and match-related variables, all compiled and processed from ATP data spanning 2000 to 2016. Additionally, in past few years, there has been an increasing interest in feature engineering, as it plays a critical role in boosting the performance of machine learning models by identifying and capturing relevant patterns and relationships within the data. As an example, Felice et al. (2023) introduce Statistically Enhanced Learning (SEL), a framework that formalizes existing feature engineering and extraction techniques in machine learning. This approach aims to tackle challenges in ML tasks by optimizing both feature selection and representation. Furthermore, numerous studies have explored methods to understand and interpret complex (black box) ML models. For instance, Auret and Aldrich (2012) utilized variable importance measures, directly

derived from RF models, along with partial dependence plots. Their work demonstrated that RF models can reliably identify the influence of individual variables, even when there is significant additive noise.

Guidelines and descriptions for the thesis

The dissertation focuses on modeling and predicting tennis matches at Grand Slam tournaments by covering various classical statistical and machine learning approaches. Within this work, different strategies are applied, including leave-one-tournament-out cross validation, expanding window [1], and rolling window. Additionally, it covers some other main aspects such as statistical enhanced learning (SEL) and interpretable machine learning tools, including Partial Dependency Plots (PDP; Friedman, 2001) and Individual Conditional Expectation (ICE; Apley and Zhu, 2020).

Dataset

The data includes information on 5,013 matches from men's Grand Slam tournaments which took place between 2011 and 2022. Several potential covariates were included, such as the players' Age, ATP ranking and Points, Odds, Elo rating, and two additional Age variables. These additional age variables were designed to capture the "optimal" age range for a tennis player, typically between 28 and 32 years (Weston, 2014). Note that, in our models, we will consider these covariates in the form of differences. For example, the difference in the players' ranking positions was calculated by subtracting the Rank of the second player from that of the first player. Similarly, for the Age variable, the age of the second player was subtracted from that of the first player, and this method was applied to all other covariates. For a detailed explanation and definition of those covariates we refer to Buhamra et al. (2024). For this analysis, we compiled a dataset using variables obtained from the R, package deuce (Kovalchik, 2019).

Classical regression models for tennis matches

First, we present a statistical approach for predicting the tennis matches of Grand Slam tennis tournaments using regression models. While linear regression is commonly employed for this purpose, it may not be suitable when the relationships between variables are non-linear or complex. In such case, splines and LASSO are utilized as an extension to the logistic model. So, we examined whether the assumption of non-linear effects or the inclusion of surface- and player-specific abilities is reasonable (i.e., we account for the incorporation of court-specific abilities). Furthermore, we address how to handle players who have participated in

[1] This strategy is used in Buhamra et al. (2024) and Buhamra et al. (2025), however in Buhamra et al. (2024) was referred to as a rolling window. Hence, it should not be confused with the rolling window approach from Buhamra et al. (2025).

only one tournament by introducing the concept of the "newcomer".

Along with selecting the most suitable models, we evaluate their performance from a betting perspective by examining the potential gains or losses each model may yield.

In this part of our work, we aim to fully leverage the flexibility of modern regression approaches, utilizing their high interpretability to uncover insights into specific associations and relationships in professional tennis. We introduce various regression methods and compare them based on some performance measures. A detailed description of these performance measures can be found in Buhamra et al. (2024).

Machine learning approaches

Next, we switch to a ML perspective and introduce those methods to demonstrate how modern machine learning approaches can help improve predictive performance. However, since these methods tend to be less interpretable, we also show how this can be addressed using interpretable ML techniques such as PDP and ICE. We introduce several machine learning approaches suitable for binary responses, such as classification trees (Breiman et al., 1984), random forests (Breiman, 2001), extreme gradient boosting (Chen and Guestrin, 2016), support vector machines (Cortes and Vapnik, 1995), and ANNs (Rosenblatt, 1958). SVMs have the advantage of being more effective in high-dimensional spaces where the number of features is very large. According to Sipko and Knottenbelt (2015), they often achieve better accuracy compared to ANNs. However, the latter will still be investigated in this thesis.

Additionally, a comparison of these approaches with the regression models examined in our previous work (Buhamra et al., 2024) is provided. Both the regression and machine learning models are compared. Here, the same dataset and performance measures are used as for the regression approaches in the previous paragraph. The evaluation is conducted separately using the same leave-one-tournament-out cross validation strategy, as well as an expanding window approach. However, this time, a rolling window approach is also introduced (see Buhamra et al., 2025).

Statistical enhanced learning aspect

Finally, in this part, we aim to explore the relevance of specific enhanced variables for both regression and machine learning approaches. To achieve this, we examine the predictive potential of "statistically enhanced covariates"[2] in more detail. For instance, in Buhamra and Groll (2025), we examined enhanced variables derived from separate approaches, such as the Elo rating, as well as those obtained through advanced transformations of original variables. One example is the covariate Age.30, which is based on the idea proposed by Weston (2014), emphasizing that the optimal age of tennis players falls between 28 and 32

[2]This concept is closely related to feature engineering, which involves selecting, transforming, or creating input variables to improve a model's performance. Feature engineering helps machine learning algorithms extract meaningful patterns from raw data by structuring it in a way that enhances predictive accuracy. Effective feature engineering can significantly impact model performance and enhance a model's predictive accuracy.

years. Therefore, the middle of this interval (i.e., 30 years) was used as the reference age in our analysis, which is how the name of this variable also originated. Additionally, we considered more conventional features directly available on certain websites, such as the players' ATP world ranking. Then, all enhanced covariates were analyzed alongside these conventional covariates. Our objective is to determine whether these enhanced covariates can improve statistical learning approaches, particularly in terms of predictive performance. The remainder of the thesis is structured as follows. Chapter 2 provides an overview of all the statistical methods used in this work. Each article is summarized in Chapter 3, including their introductions and conclusions. In Chapter 4, the overall conclusion of the thesis is presented. Finally, Part II consists of the three articles that form the core of this thesis.

Chapter 2

Statistical methods

This chapter introduces the reader to the statistical methods used in our analysis. It begins with a classical logistic regression approach for a binary outcome. We then extend the regression model, particularly focusing on regularization techniques like Least Absolute Shrinkage and Selection Operator (LASSO). Non-parametric models, such as splines, are also introduced. Finally, we cover five machine learning approaches, emphasizing interpretability through IML tools, which help illustrate and understand 'black box' models, such as RF. Note that some parts of this chapter are adapted from the main articles (Buhamra et al., 2024; 2025).

2.1 Logistic regression model

Although linear models are well-suited for regression analyses when the response variable is continuous and may show an approximate normal distribution (conditional on the covariates), if necessary after an appropriate transformation, in many cases the response is not continuous but binary, as is the case in our study. Generalized linear models (GLMs) combine various regression methods for response variables that do not need to follow a normal distribution, including the logit model for binary responses, which will be the focus of our analysis.

For n individuals, let be given observations $(y_i, x_{i1}, \ldots, x_{ip})$, $i = 1, \ldots, n$, of a binary target variable y and covariates $x_1, \ldots, x_p$. In the logistic regression model, the relationship between y and metric, categorical or binary covariates is examined. Here, $y = 1$ denotes the occurrence of a particular event (typically defined as "success") and y = 0 that the event does not occur (also defined as "failure"). Then,

$$\pi_i = P(y_i = 1|x_{i1}, \ldots, x_{ip}) = E[y|x_{i1}, \ldots, x_{ip}]$$

is the (conditional) probability for the occurrence of $y_i = 1$, given the covariate values $x_{i1}, \ldots, x_{ip}$. The aim is to model π_i appropriately as a function of the feature variables.

The associated linear predictor is defined as

$$\eta_i = \boldsymbol{x}_i^\top \boldsymbol{\beta}, \tag{2.1}$$

with $\boldsymbol{x}_i = (1, x_{i1}, \ldots, x_{ip})^\top$ and $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_p)^\top$. Since $\pi_i \in [0, 1]$ has to hold for all possible choices of $\boldsymbol{x}$, the linear model approach $\hat{\pi}_i = y_i$ cannot be chosen.

Therefore, the linear predictor η_i is related to the probability π_i by a strictly monotonically increasing function $h\colon \mathbb{R} \to [0, 1]$, i.e.

$$\pi_i = h(\eta_i) = h(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ip}) .$$

The function $h(\cdot)$ is also called the *response function*. With the help of the inverse function $g = h^{-1}$, we can also write $\eta_i = g(\pi_i)$. Equation (2.1) can be written as

$$\eta_i = g(\pi_i).$$

The function g is called *link function* (see, e.g., Nelder and Wedderburn, 1972, or Fahrmeir et al., 2013). In the *logit* model, the logistic response function

$$\pi = h(\eta) = \frac{\exp(\eta)}{\exp(1 + \eta)}$$

is chosen and the corresponding *logit link function* is given by

$$g(\pi) = \ln\left(\frac{\pi}{1 - \pi}\right) = \eta = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p.$$

Parameter estimation

The regression parameters of the logit model are typically estimated via maximum likelihood (Fahrmeir et al., 2013). We write the given data as $(y_i, \boldsymbol{x}_i)$, $i = 1, \ldots, n$. Assuming that the observations are stochastically independent, the likelihood function is given by

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} f(y_i|\boldsymbol{\beta}; x_i).$$

Here, $f(y_i|\boldsymbol{\beta}; \boldsymbol{x}_i)$ denotes the likelihood contributions of $y_i|\boldsymbol{x}_i$, which depend on the unknown parameter vector $\boldsymbol{\beta}$ via $\pi_i = E[y_i|\boldsymbol{x}_i] = h(\boldsymbol{x}_i^\top \boldsymbol{\beta})$. Thus, for Bernoulli data the likelihood can also be represented as

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} L_i(\boldsymbol{\beta}) = \prod_{i=1}^{n} \pi_i^{y_i}(1 - \pi_i)^{1-y_i}$$

where $L_i(\boldsymbol{\beta})$ is the individual likelihood contribution of the i-th observation. With

$$\pi_i = \frac{\exp(\boldsymbol{x}_i^\top \boldsymbol{\beta})}{1 + \exp(\boldsymbol{x}_i^\top \boldsymbol{\beta})}$$

and by logarithmizing we obtain the log-likelihood

$$l(\boldsymbol{\beta}) := \ln(L(\boldsymbol{\beta})) = \sum_{i=1}^{n} \left[y_i(\boldsymbol{x}_i^\top \boldsymbol{\beta}) - \ln(1 + \exp(\boldsymbol{x}_i^\top \boldsymbol{\beta})) \right].$$

The estimators for $\beta_0, \ldots, \beta_p$ are obtained by numerical maximization of the log-likelihood, e.g. by using the Fisher scoring or the Newton-Raphson method, (see, e.g., Nelder and Wedderburn, 1972). Generally, for more details on GLMs, see also Fahrmeir and Tutz (2001). The standard logit model can be fitted in R via the `glm` function available in the basic `stats` package (R Core Team, 2024).

2.2 Extension for the regression models

In the following, we will present extensions of the classical regression model. First, in Section 2.2.1 we will explain the idea of LASSO penalization. Then, splines will be introduced in more detiales in Section 2.2.2.

2.2.1 Least absolute shrinkage and selection operator (LASSO)

Regularization is a statistical technique used to prevent overfitting in predictive models, e.g. by adding a penalty term to the loss function, discouraging overly complex models. This regularization can take various forms, such as L1 penalization (LASSO), which encourages sparsity by forcing some coefficients to be exactly zero, or L2 penalization (ridge), which shrinks coefficients toward zero but does not eliminate them entirely. Additionally, spline smoothing is a form of regularization that uses piecewise polynomial functions to create flexible models while controlling excessive wiggliness in the fitted curve. Component-wise boosting (see, e.g., Bühlmann and Yu, 2003, Bühlmann and Hothorn, 2007, or Schmid and Hothorn, 2008) is another regularization approach that builds models incrementally by adding weak learners sequentially, focusing on correcting errors made by previous iterations while maintaining model simplicity. Penalization, as a specific concept of regularization, helps create simpler and more interpretable models while improving their ability to generalize to unseen data. The strength of the penalty is controlled by a tuning parameter, which balances the trade-off between model complexity and predictive accuracy. For example, ridge regression (L2 penalization) adds a penalty proportional to the sum of the squared coefficients, shrinking them toward zero without completely eliminating any. In contrast, LASSO regression (L1 penalization) applies a penalty based on the absolute values of the coefficients, encouraging sparsity by forcing some coefficients to be exactly zero, effectively performing variable selection. Throughout this work, we will mainly focus on LASSO penalization and spline smoothing.

Overall, LASSO is a powerful tool for improving prediction accuracy and simplifying models by reducing the number of predictor variables (Tibshirani, 1996). It is particularly applicable to binary response variables. LASSO enhances the traditional linear regression model by incorporating an ℓ_1-norm penalty on the coefficients. This penalty shrinks some of the coefficient estimates toward zero, effectively performing variable selection.

Figure 2.1 by James et al. (2013) provides a geometrical interpretation of the constraints imposed by LASSO and ridge regression, exemplarily in two-dimensional coefficient space (β_1, β_2).

The left chart shows the LASSO constraint region, which is as a diamond-shaped area, representing the ℓ_1-norm constraint $|\beta_1| + |\beta_2| \leq s$. The chart on the right illustrates the ridge regression constraint, which is a circular region corresponding to the ℓ_2-norm constraint $\beta_1^2 + \beta_2^2 \leq s$. In the context of LASSO and ridge regression, s represents the constraint or shrinkage parameter, which determines the maximum allowable size of the coefficient vector within a specified norm. Thus, s controls the degree of regularization—a smaller s forces coefficients to be closer to zero, while a larger s allows more flexibility in coefficient values.

The red ellipses in both plots represent the contours of the residual sum of squares (RSS), which indicate the direction of the optimal solution in an unconstrained regression setting. The estimated coefficients are obtained where these ellipses first touch the constraint region. For LASSO, the diamond-shaped constraint leads to solutions that often lie exactly on the axes, meaning some coefficients are set to zero. This occurs because the diamond has sharp corners, which increases the likelihood that the RSS contours will first intersect at these points—enabling automatic feature selection.

In contrast, the circular constraint of ridge regression does not favor solutions where coefficients are exactly zero. Instead, it shrinks all coefficients toward zero continuously, but without completely eliminating any feature. This explains why ridge regression is effective for shrinkage but not for variable selection. Thus, the fundamental difference between the two approaches is that LASSO encourages sparsity by setting some coefficients to zero, whereas ridge regression retains all predictors but shrinks their magnitudes to prevent overfitting.

Note that in Figure 2.1, the authors explored the case where $p = 2$. Where p represents the number of features in the regression model. When $p = 3$, the ridge regression constraint region becomes a sphere, while the LASSO constraint forms a polyhedron. For $p > 3$, the ridge regression constraint becomes a hypersphere, and the LASSO constraint becomes a polytope. Despite these changes in the constraint shapes, the fundamental concepts shown in the coreponding figure still apply. Specifically, the LASSO continues to perform feature selection when $p > 2$, due to the sharp corners of the polyhedron or polytope. According to James et al. (2013), the LASSO has a major advantage over ridge regression in that it produces simpler and more interpretable models, involving only a subset of the predictors. Further information and a comparison between the two methods, along with insights into

which method yields better predictions, can be found e.g. in Hastie et al. (2015).

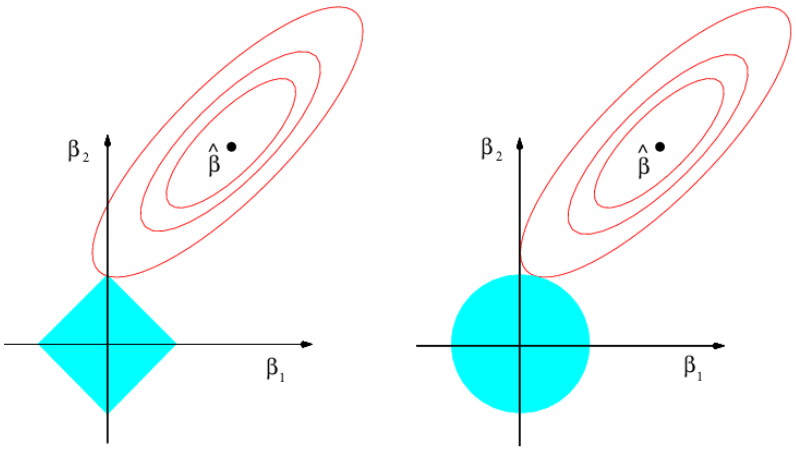

Figure 2.1: LASSO (ℓ_1-norm) vs. ridge regression (ℓ_2-norm) constraints: The LASSO constraint forms a diamond-shaped region promoting sparsity, while the ridge constraint forms a circular region, leading to coefficient shrinkage without selection as provided by James et al. (2013) in the online resources (see `https://www.statlearning.com/resources-second-edition`).

When the number of covariates p becomes very large, estimation becomes numerically unstable (see, e.g., Fahrmeir et al., 2013). This can also be the case if there is some substantial mulitcollinearity between the columns of the design matrix $\boldsymbol{X} = \left(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\right)^T$. To address this problem, a penalty term pen($\boldsymbol{\beta}$) is added to the negative log-likelihood in the logit model. According to Park and Hastie (2007), the estimator is then obtained by minimizing

$$\hat{\boldsymbol{\beta}}_{\text{pen}} = \arg\min_{\boldsymbol{\beta}} \left(-l(\boldsymbol{\beta}) + \lambda \cdot \text{pen}(\boldsymbol{\beta})\right),$$

where λ is the *penalty parameter* that controls the influence of the penalty term on the parameters estimated by the maximum likelihood method (Fahrmeir et al., 2013). Here, the penalty term is given by

$$\text{pen}(\boldsymbol{\beta}) = \sum_{j=1}^{p} |\beta_j|.$$

By keeping the intercept free from penalization, regularization methods help maintain a balance between model simplicity and predictive accuracy while preserving essential aspects of the prediction. Penalizing the intercept could reduce model interpretability and make it more challenging to understand the model's baseline behavior.

Thus, the penalty term allows model estimation and variable selection to be performed in one step, as small coefficients are shrunk to 0. According to Park and Hastie (2007), the estimator $\hat{\boldsymbol{\beta}}_{\text{LASSO}}$ is given by

$$\hat{\boldsymbol{\beta}}_{\text{LASSO}} = \arg\min_{\boldsymbol{\beta}} \left(-l(\boldsymbol{\beta}) + \lambda \cdot \sum_{j=1}^{p} |\beta_j| \right).$$

There is no closed-form representation for solving this minimization problem. Therefore, numerical optimization methods are used to obtain the optimal LASSO estimator $\hat{\boldsymbol{\beta}}_{\text{LASSO}}$ (see, e.g., Friedman et al., 2010). In order to determine an optimal value for λ, typically for a sequence of different λ values K-fold cross validation[3] is performed. The mean cross validated (CV) error is then calculated for each of these λ. The error is typically compared on the basis of the *deviance* on the omitted data, which is defined as the negative of twice the log-likelihood. To estimate $\boldsymbol{\beta}_{\text{LASSO}}$, the λ that minimizes mean CV error is used (Friedman et al., 2010). This method is implemented in the `glmnet` R package by Friedman et al. (2010) and a notable update is detailed in Tay et al. (2023).

Figure 2.2 provides an illustrative example on an artificial dataset to explain the concept of LASSO. The LASSO model is fitted using the `glmnet` function from the eponymous R package, and the coefficient paths are then plotted. This visualization is useful for understanding how LASSO selects features and penalizes complexity to improve predictive performance. It displays the coefficient paths of a LASSO-regularized logistic regression model, fitted using the `glmnet` package on a simulated dataset with 500 observations and 10 predictors generated from a normal distribution. The coefficient paths illustrate how feature selection is influenced across different values of the regularization parameter λ. The plot helps identify which predictors remain important after applying LASSO regularization. The x-axis represents the λ values, which control the strength of penalization. As λ increases, many coefficients are shrunk to zero, leaving only a few important features (blue lines). Conversely, as λ decreases, the model allows larger coefficient magnitudes, capturing more complexity. The y-axis shows the values of the estimated regression coefficients ($\hat{\beta}_j$) for different predictors at each λ value. If a coefficient remains nonzero, it means the corresponding predictor is more relevant for classification in the respective logistic regression model. Each line represents how a specific predictor's coefficient changes as λ varies. Black lines represent coefficients that are strongly shrunk toward zero (less important predictors). Blue lines represent the three most significant predictors (true nonzero coefficients), which retain nonzero values even under strong penalization. As λ increases, most black lines move toward zero, meaning those features are being eliminated. The red dashed vertical line represents λ_{opt}, the optimal penalty parameter selected via tuning, in particular here via 10-fold cross validation. This is the point where the model achieves the best balance between accuracy and generalization. At λ_{opt}, some coefficients remain nonzero (important predictors), while others have been eliminated (unimportant features). The optimal model at λ_{opt} includes only the strongest predictors, improving interpretability and avoiding overfitting.

[3] K-fold cross validation is a technique used to assess the performance of a machine learning model. It involves dividing the dataset into K equally-sized subsets (or folds). The model is trained on K-1 folds and tested on the remaining fold. This process is repeated K times, with each fold used as the test set once. The results are then averaged to provide a more reliable estimate of model performance.

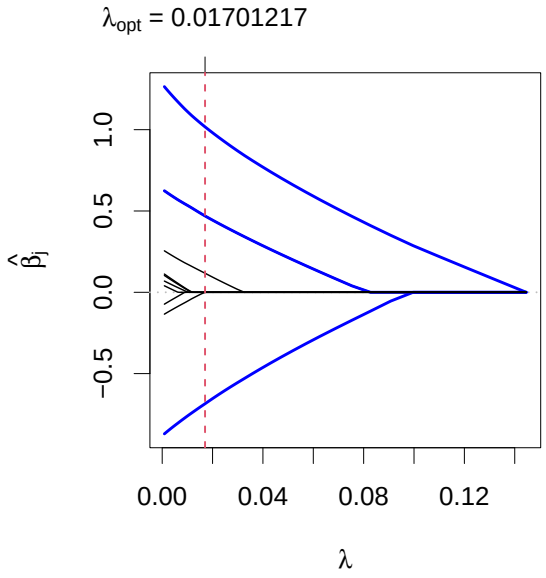

Figure 2.2: Exemplary coefficient paths for the LASSO model across λ values on an artificial dataset, with the three key nonzero coefficients in blue and others in black. The vertical red dashed line marks the optimal λ from cross-validation.

2.2.2 Non-parametric models (splines)

In the methods discussed above, the relationship between the covariates and the target variable is assumed to be linear. However, real-world data often exhibit non-linear patterns that cannot be adequately captured by linear models. To address this, we begin with one of the simplest ways to introduce non-linearity—polynomial regression. While polynomials offer more flexibility, they can lead to issues such as overfitting and instability. A more advanced and robust approach is the use of splines, which provide a piecewise polynomial representation that balances flexibility and interpretability. In this context, we employ *B-splines* (Eilers and Marx, 1996) to model non-linear relationships effectively.

Let y_i be the i-th observation of the response variable and x_i the corresponding value of a single (metric) covariate. The influence of x_i on y_i can then be modeled as a polynomial function $f(\cdot)$ of degree l, i.e.,

$$f(x_i) = \gamma_0 + \gamma_1 x_i + \ldots + \gamma_l x_i^l.$$

For more flexible modeling, the range of values of the covariates is also divided into intervals, so that one obtains several locally estimated polynomials. However, since this leads to a non-steady function, *polynomial splines* are defined. A polynomial spline of degree $l \geq 0$ to the nodes $a = \kappa_1 < \cdots < \kappa_m = b$ is a function $f : [a, b] \to \mathbb{R}$ with the following features (see, e.g., Fahrmeir et al., 2013):

1. $f(\cdot)$ is $(l-1)$ times continuously differentiable.

2. On every interval $[\kappa_j, \kappa_{j+1})$ $f(\cdot)$ is a polynomial of the l-th degree.

The range of values is thus decomposed by the knots $\kappa_1, \ldots, \kappa_m$. For the constructive representation of polynomial splines, various equivalent approaches exist (Fahrmeir et al., 2013). Here, we use the so-called *B-splines* basis proposed by (Eilers and Marx, 1996).

B-splines

The function $f(x)$ from above can be represented as a linear combination of $d = m + l - 1$ so-called *basic-spline* or *B-spline* basis functions. Such a basis function consists of $l+1$ polynomial pieces of l-th degree, which are composed in a $l-1$ times continuously differentiable manner. $f(\cdot)$ can then be represented according to Fahrmeir et al. (2013) as

$$f(x) = \sum_{j=1}^{d} \gamma_j B_j(x),$$

where γ_j are the coefficients to be estimated, and $B_j(x)$ represents a set of basis functions.

Following Fahrmeir et al. (2013), the B-spline basis functions can be recursively defined as following:

$$B_j^l(x) = \frac{x - \kappa_j}{\kappa_{j+l} - \kappa_j} B_j^{l-1}(x) + \frac{\kappa_{j+l+1} - x}{\kappa_{j+l+1} - \kappa_{j+1}} B_{j+1}^{l-1}(x),$$

where

$$B_j^0(x) = \mathbb{1}_{[\kappa_j, \kappa_{j+1})}(x) = \begin{cases} 1, & \text{if } \kappa_j \leq x < \kappa_{j+1}, \\ 0, & \text{otherwise.} \end{cases} \qquad j = 1, \ldots, d-1.$$

To estimate a B-spline, the corresponding basis functions have to be first computed at a given number of knots. At this point, we introduce the B-spline design matrix $\mathbf{B}$, which plays a crucial role in representing smooth functions through B-splines. It is constructed by evaluating a set of B-spline basis functions at specified data points. Each row of the matrix corresponds to an observation, while each column represents the values of a specific B-spline basis function evaluated at all n observations. The matrix $\mathbf{B}$ serves as the foundation for spline-based regression and smoothing techniques, allowing flexible, piecewise polynomial fitting. This matrix forms the foundation for representing the spline function and is later used in the estimation process. It is given as

$$\mathbf{B} = \begin{bmatrix} B_1(x_1) & B_2(x_1) & \cdots & B_d(x_1) \\ B_1(x_2) & B_2(x_2) & \cdots & B_d(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ B_1(x_n) & B_2(x_n) & \cdots & B_d(x_n) \end{bmatrix},$$

each row corresponds to an observation x_i, each column represents the evaluation of a B-spline basis function $B_j(\cdot)$ at the corresponding data points, d is the number of basis functions, and n is the number of data points.

This process is shown in Figure 2.3, where basis functions are computed to approximate

the simulated data (colored gray here). In Figure 2.4, each basis function is assigned a weight $\hat{\gamma}_j$ for scaling by the estimator $\hat{\gamma}$.

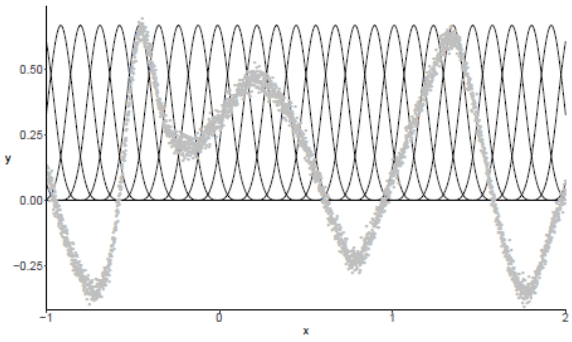

Figure 2.3: B-spline basis

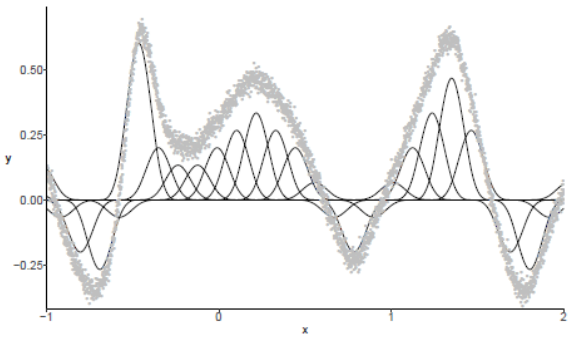

Figure 2.4: Weighted basis functions

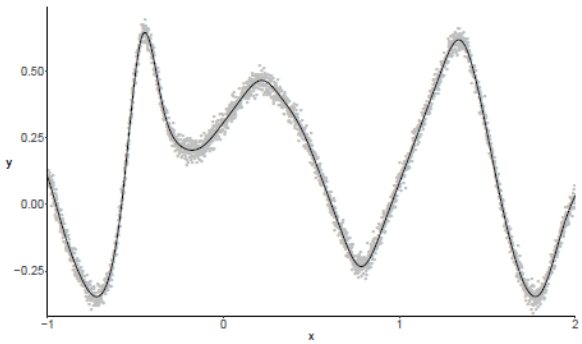

Figure 2.5: Final spline estimation

The final estimation is calculated by summing up the single weighted basis functions (see Figure 2.5).

As an unpenalized estimation of a non-linear B-spline effect often overfits, typically the non-linear effect is smoothed by using *penalized B-splines*, i.e. *P-splines*, which will be introduced in the next paragraph.

P-splines

Beside the problem of potential overfitting, the goodness-of-fit of the B-spline approach depends on the number of selected nodes. To avoid this problem, various penalization methods exist in the form of P-splines (Eilers and Marx, 2021). Here, a penalized estimation criterion, which is extended by a penalty term, is used instead of the usual estimation criterion (Fahrmeir et al., 2013). For P-splines based on B-splines (see, e.g., Eilers and Marx, 1996), the function $f(x)$ is first approximated by a polynomial spline with many nodes (typically about 20 to 40). The penalty term then results in

$$\lambda \int (f''(x))^2 \, dx \, .$$

This is motivated by the fact that the second derivative is used as a measure of the curvature of a function. For approximation of the second derivative exist simple representations, so that the penalty term results in

$$\lambda \sum_{j=3}^{d} \left(\Delta^2 \gamma_j\right)^2 ,$$

where $\Delta^2 \gamma_j = \gamma_j - 2\gamma_{j-1} + \gamma_{j-2}$ (see again Eilers and Marx, 1996).

The second order difference $\Delta^2 \gamma_j$ can be represented as a matrix operation involving the vector of coefficients $\gamma = (\gamma_1, \gamma_2, \ldots, \gamma_d)^T$. The penalty on the second differences is equivalent to:

$$\lambda \sum_{j=3}^{d} (\gamma_j - 2\gamma_{j-1} + \gamma_{j-2})^2.$$

This can be written in matrix form as:

$$\lambda \boldsymbol{\gamma}^T \mathbf{Q} \boldsymbol{\gamma}$$

where the penalty matrix $\mathbf{Q}$ is a symmetric matrix. Specifically, $\mathbf{Q}$ is a tridiagonal matrix that captures the second-order differences between the coefficients γ_j, where the diagonal entries are 2 and the off-diagonal entries are -1, as follows:

$$\mathbf{Q} = \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \cdots & 0 \\ 0 & -1 & 2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 2 \end{bmatrix}.$$

To fit the P-spline model e.g. in the simple case of a Gaussian response, i.e. a classic linear model setup, we minimize the following objective function, which combines the least squares loss with the penalty term:

$$L(\boldsymbol{\gamma}) = \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \boldsymbol{\gamma}^T \mathbf{Q} \boldsymbol{\gamma}$$

the penalized least squares estimate then has the form

$$\hat{\gamma} = \left(\mathbf{B}^T \mathbf{B} + \lambda \mathbf{Q}\right)^{-1} \mathbf{B}^T \mathbf{y}$$

where $\mathbf{B}$ is the design matrix formed by the B-spline basis functions, and $\mathbf{Q}$ is the penalty matrix (which here exemplarily encodes second order differences between the coefficients γ_j). The penalty matrix $\mathbf{Q}$ encourages smoothness by shrinking differences between adjacent coefficients. As shown by Eilers and Marx (2021), this approach results in a computationally efficient and stable smoothing method, widely used in practical applications.

The optimal smoothing parameter λ is again a tuning parameter of the procedure and is

determined using generalized CV. For more details on the methodology, refer to Eilers and Marx (2021). Information on the corresponding software implementation in R can be found in Wood (2017).

2.3 Machine learning (ML) approaches

In this section, we present the machine learning methods employed throughout this work. While some of the mathematical formulas are derived from the original articles, additional details and explanations are provided here for further clarity.

2.3.1 Decision trees

The fundamental concept behind the *decision trees* is to recursively partition the data into subsets based on the values of the input features. At each node of the tree, a decision is made about which feature to split on and what threshold to use. This process is repeated until a stopping criterion is met, resulting in a tree structure with leaf nodes containing the predicted values.

The Classification and Regression Tree (CART) algorithm, introduced by Breiman et al. (1984), is a class of non-parametric algorithms capable of handling both classification and regression tasks. While it has powerful capabilities in regression, where it can predict continuous outcomes, this thesis primarily focuses on classification tasks, as our target variable is binary. This type of learning is widely applied across various fields, e.g. Angelini et al. (2008); Conneau et al. (2017). In medical diagnosis, the goal is often to assess whether a particular disease is present in a patient, based on a combination of clinical data, test results, and symptoms Kourou et al. (2015). Another example is provided by Bishop and Nasrabadi (2006), which offers a comprehensive resource for understanding classification problems and pattern recognition. This work is applicable to a wide range of fields, including computer vision, speech recognition, and bioinformatics.

In the case of classification trees, the focus is on predicting categorical outcomes. In the context of sport domain such as tennis, a classification tree can predict the winner of a match based on features like player ranking, age, experience, recent performance, surface type, and more. Unlike traditional linear regression models, which assume a linear relationship between variables, classification trees are capable of capturing complex non-linear patterns and interactions within the data. The target (response) variable is typically binary (e.g., in our case, $y = 1$ if Player A wins, $y = 0$ if Player B wins). The tree grows by iteratively splitting the dataset based on values of the covariates. These splits are selected to create more homogeneous groups with respect to the response variable. Common splitting criteria include metrics such as Gini impurity or entropy (information gain). Hence, decision trees seek to find the best split to subset the data.

CARTs use the Gini index in the process of splitting the dataset into a decision tree, which is defined by

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}),$$

a measure of total variance across the K classes. Here, $\hat{p}_{mk}$ represents the proportion of training observations in the m-th region that are from the k-th class. The Gini index is minimized when all values of $\hat{p}_{mk}$ are close to zero or one. Therefore, the Gini index serves as a measure of node purity, where a low value indicates that a node is largely composed of observations from a single class (James et al., 2013).

Another preferable measure which is an alternative to the Gini index is the entropy, given by

$$E = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}.$$

Similar to the Gini index, entropy takes on a low value when the m-th node is pure. In fact, the Gini index and entropy are often numerically quite close (James et al., 2013). Note that, in this work, we focus on using the Gini index. Some mathematical background is provided, assuming a basic knowledge of tree-based methods. For more detailed description see James et al. (2013) and Hastie et al. (2009).

One notable drawback of classification trees is their tendency to overfit the data, especially when the tree grows too deep. A fully grown decision tree often overfits the training data, leading to poor generalization on new, unseen test data. This issue can be addressed by pruning the tree to reduce its complexity or by using ensemble methods such as random forest or gradient boosting. In the `rpart` package (Therneau et al., 2015), this is managed by the complexity parameter (*cp*), which sets a threshold that an improvement must exceed for a split to remain in the tree. A very small *cp* value leads to overfitting, while a large *cp* results in an overly simple tree, causing underfitting. Hence, *cp* is the main tuning parameter of the method. Both extremes reduce the model's predictive performance. The optimal *cp* can be identified by testing different values and using cross validation (CV) to determine the corresponding prediction accuracy of the model (we implemented this CV approach to ensure that for all methods requiring tuning, the same data folds are used, allowing for better comparability). The best *cp* is defined as the one that maximizes the CV accuracy. Another issue is the instability of classification trees, as small changes in the data can lead to different splits and significantly different trees. This instability is related to the high variance of decision trees, making them highly sensitive to even small variations in the training data and potentially leading to inconsistent predictions.

An example of a classification tree in practice is shown in Figure 2.6. To fit the tree and produce a plot that reflects the specified rules accurately, the `rpart` package is used along with the function `rpart.plot` for visualization. In a binary classification tree, the goal is to predict which of two possible outcomes will occur. In the context of tennis matches, the focus would be on predicting the winner of a match, such as:

- $Y = 1$: Player A wins the match.
- $Y = 0$: Player B wins the match.

At each internal node of the tree, a decision is made (e.g., "Is Player A's ranking larger than 10?"). Based on the answer (Yes/No), the data is split and moves to the next node, continuing until a prediction is made at the leaf node. The final prediction at the leaf node (e.g., Player A wins or Player B wins) is determined by the majority class in the training data that reaches that node. Based on the proportions of observations in the final nodes, the respective probabilities for $Y = 1$ and $Y = 0$ are also obtained.

Thus, the classification tree splits the data based on input features (such as player statistics) to classify the outcome into one of the predefined categories. This means that we can train a classification tree using features like player rankings, their recent performance, age, surface type, etc. Each node in the tree represents a decision point based on one of the features (e.g., Elo_A, Elo_B, Age_A, etc.). The splits in the tree indicate how the model partitions the data to make predictions. For example, it may first split based on whether Elo_A is larger than a certain value. The terminal nodes (leaves) represent the final predictions (outcomes) of matches. Each leaf will show the predicted class (0 or 1) and the proportion of instances that belong to each class. The graph also shows the probability of winning for each player. This provides insight into how confident the model is about its predictions.

By examining the splits, one can understand which features (Elo ratings, Age, Surface) are most influential in determining the match outcome. If Elo_A is frequently used for splits, it may indicate its importance. By following the branches from the root to the leaves, we can identify the criteria that lead to each prediction, making the model interpretable for tennis match outcomes. Note that later in our work, we modify this approach by using the differences between the covariate values of the first-named and second-named players to better capture their relative advantages. Specifically, the value of the second player was always subtracted from the value of the first player.

In the following sections, we introduce both random forest and extreme gradient boosting (XGBoost) as extensions and enhancements to this approach. Random forests are a parallel ensemble method that improves prediction accuracy by averaging the outcomes of multiple decision trees. Similarly, gradient boosting machines build trees sequentially, with each new tree attempting to correct the errors of the previous ones. Classification trees, particularly when combined with these ensemble methods, provide a robust framework for predicting tennis matches by incorporating some factors such as player form, surface type, or ranking into the model. While the literature on employing trees to model and forecast tennis matches is still scarce (Gao and Kowalczyk, 2021; Svensson, 2021 and Ünal, 2023), this is different in soccer. Bunker et al. (2024a) demonstrates such applications in predicting soccer match results. Groll et al. (2019) introduced a hybrid modeling approach that combines random forests with Poisson ranking methods to predict soccer match outcomes in international tournaments. This innovative model integrates team ability parameters derived from historical

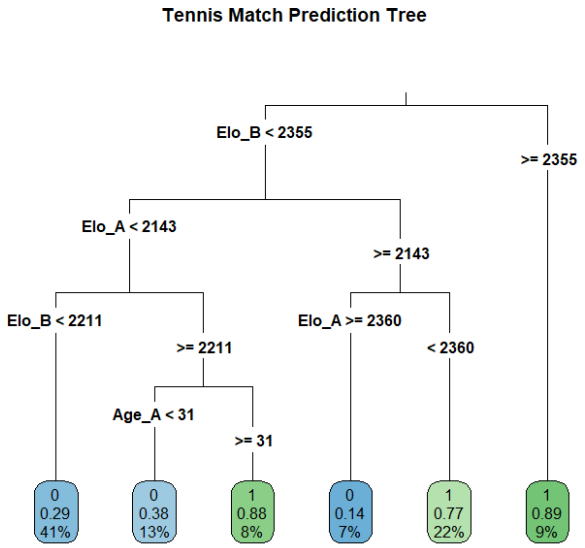

Figure 2.6: Example of a classification tree model to predict the outcome of a tennis match based on players' Elo ratings and Ages

match data into the random forest framework, enhancing its predictive accuracy. The results confirmed the model's strong predictive capabilities, outperforming traditional methods, including betting odds.

2.3.2 Random forest (RF)

A random forest (RF) is a machine learning algorithm that can be used for both classification and regression tasks, first proposed by Breiman (2001). It is based on a parallel ensemble of regression or decision trees, depending on the type of outcome. Here, we'll focus on the version with decision trees. The algorithm is called "random" due to two key aspects of randomness in its construction. First, in each tree it randomly selects subsets of predictors at each split during training, ensuring diversity across the trees. Second, each decision tree in the ensemble is trained on a randomly drawn bootstrap sample from the original dataset, meaning that different subsets of the data are used for training each tree. It is referred to as a "forest" because it builds an ensemble of multiple decision trees, which collectively improve predictive accuracy and robustness.

Random forests are typically trained using the 'bagging' method, which reduces model variance by introducing randomness through bootstrap sampling with replacement from the training data set. Nevertheless, sampling without replacement is often assumed, as it simplifies analysis (Ramosaj, 2020; Scornet et al., 2015).

To apply bagging in a classification tree, a large number B of trees are generally generated commonly with default values of $B = 500$ or $B = 1000$ (Probst et al., 2019), using B bootstrapped training sets. The final prediction is then determined by majority voting among the resulting trees. In our study, we found that using $B = 400$ was sufficient to achieve

good performance (i.e., we set `ntree` = 400 and, hence, did not tune this parameter). Note that, the number of trees in a forest is typically not tuned, as increasing the number of trees generally enhances performance (Scornet, 2017; Díaz-Uriarte and Alvarez de Andrés, 2006). Additional details on bagging are available in James et al. (2013).

The main extension compared to bagging is the idea of drawing randomly only a subset of "`mtry`" features at each node for splitting in the individual trees, which involves selecting a random subset of predictors from the set of all available features at each decision node in a decision tree. This introduces diversity among the trees in the ensemble, decreasing the variance of the ensemble and, hence, reducing the risk of overfitting and enhances the model's generalization performance by ensuring that each tree is built on a different subset of features. For fitting random forest models, we use the `ranger` implementation in R (Wright and Ziegler, 2017). In addition, there are other R packages that provide an automatic tuning procedure for random forests, such as `mlr3` and `caret` packages (see Bischl et al., 2023 and Kuhn, 2008).

The choice of hyperparameters is task-dependent and can be determined using techniques such as CV or random search (Bartz et al., 2023). Here, `mtry` and the `ntree` are the two parameters that most likely have a big effect on our final accuracy and are defined as follows:

- `mtry`: number of variables randomly sampled as candidates at each split.
- `ntree`: number of trees to grow.

The `mtry` hyperparameter controls the split-variable randomization feature of the random forest and it helps to balance low tree correlation with reasonable predictive strength. According to Hastie et al. (2009), for classification $mtry = \sqrt{p}$ is recommended, where p is the number of predictors. However, when there are fewer relevant predictors a larger value of `mtry` tends to perform better because it makes it more likely to select those features with the strongest signal. When there are many relevant predictors, a lower `mtry` might perform better. For this reason, we decided to properly tune this parameter using (self-implemented) 10-fold CV. Regarding the minimum node size, it is recommended to set the minimum node size to one for classification and five for regression (Díaz-Uriarte and Alvarez de Andrés, 2006).

There are no specific criterias when it comes to choosing the number of trees that the random forest should use, as long as `ntree` is large enough. However, there is a threshold where adding more trees does not give any significant gain to the random forest (Oshiro et al., 2012), but computational costs would unnecessarily increase.

Like a decision tree, using random forest for tennis match prediction is advantageous because they can handle non-linear relationships between variables and naturally account for complex interactions, such as how a player's performance might depend on the match surface, Elo or their age. Additionally, random forest reduces the risk of overfitting by averaging predictions across many trees, making it robust and highly accurate for real-world predictions. This method also provides insights into which factors are most important in

influencing match outcomes through variable importance metrics. For further details about variable importance, the reader can refer to Hastie et al. (2009).

2.3.3 Extreme gradient boosting (XGBoost)

As alternative to the random forest approach from above (parallel ensemble methods) we also consider sequential ensembles. A well-known approach in this context is boosting, a technique derived from the machine learning field (Freund et al., 1996), which was subsequently adapted to estimate predictors in statistical models (Friedman et al., 2000, and Friedman, 2001). Generally, the concept of an iterative boosting algorithm is to combine multiple weak learners additively into a single, strong ensemble model, which leads to high predictive accuracy (Schapire, 1990). One notable advantage of statistical boosting algorithms is their flexibility for high-dimensional data and their ability to incorporate variable selection in the fitting process (Mayr et al., 2014). The boosting model f^{boost} has the form

$$f^{\text{boost}}(\boldsymbol{x}) = \sum_{m=1}^{M} f_m(\boldsymbol{x_i})$$

where $f_1, \ldots, f_M$, $M \in \mathbb{N}$ are weak learners, which will be referred to as base functions in the following. According to Friedman et al. (2000), boosting can be considered as an additive expansion in a set of elementary functions. Hence, f^{boost} can be rewritten as

$$f^{boost}(\boldsymbol{x}) = \sum_{m=1}^{M} \alpha_m h(\boldsymbol{x}; \mathbf{a_m}),$$

where $\alpha_1, \ldots, \alpha_M \in \mathbb{R}$ are expansion coefficients and $h : \mathbb{R}^d \to \mathbb{R}$ with $\boldsymbol{x} \to h(\boldsymbol{x}; \mathbf{a})$ are described in terms of simple basis functions by a set of parameters $\mathbf{a_m} \in \mathbb{R}^D$. Fitting such a model involves selecting the basis functions and then their parameters $\{\alpha_m, \mathbf{a_m}\}_{m=1}^{M}$ are determined by minimizing a loss function L averaged over the training dataset

$$\{\hat{\alpha}_m, \hat{\mathbf{a}}_\mathbf{m}\}_{m=1}^{M} = \arg \min_{\{\alpha_m, \mathbf{a_m}\}_{m=1}^{M}} \frac{1}{n} \sum_{i=1}^{n} L\left(Y_i, \sum_{m=1}^{M} \alpha_m h(\boldsymbol{X_i}, \mathbf{a_m})\right).$$

Here, a forward stagewise additive modeling approach is commonly applied. In this method, optimization is carried out by sequentially adding new basis functions to the additive model, without modifying the parameters and coefficients of those already included. At each iteration m, $m = 1, \ldots, M$, we identify the following

$$(\alpha_m, \mathbf{a_m}) = \arg \min_{\alpha, \mathbf{a}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, g_{m-1}(\boldsymbol{X_i}) + \alpha h(\boldsymbol{X_i}, \mathbf{a})), \tag{2.2}$$

with

$$g_0(x) = 0 \quad \text{and} \quad g_{m-1}(x) = \sum_{k=1}^{m-1} \alpha_k h(x; \mathbf{a_k}),$$

which results in

$$g_m(\boldsymbol{x}) = g_{m-1}(\boldsymbol{x}) + \alpha_m h(\boldsymbol{x}; \mathbf{a}_m).$$

Further details can be found in Hastie et al. (2009). The boosting framework was later generalized to support the optimization of any differentiable loss function (Friedman, 2001), where a functional gradient descent approach is used to solve equation (2.2). For a differentiable loss function L, the corresponding negative gradient on the training data is given by

$$-\gamma(\boldsymbol{x}_i) = \left[\frac{\partial L(Y_i, g(\boldsymbol{x}_i))}{\partial g(\boldsymbol{x}_i)}\right]_{g(x)=g_{m-1}(x)}.$$

Since the negative gradient is available only at the data points given in the training dataset, it is approximated by $h(\boldsymbol{x}; \mathbf{a}_m)$ to enable generalization to other points, where

$$\{\alpha_m, \mathbf{a_m}\} = \arg\min_{\alpha, \mathbf{a}} \sum_{i=1}^{n} \left(-\gamma_m(\boldsymbol{x}_i) - \alpha h(\boldsymbol{x}_i; \mathbf{a})\right)^2.$$

The step length ρ_m is determined using a line search of the form

$$\rho_m = \arg\min_{\rho} \sum_{i=1}^{n} L\left(Y_i, g_{m-1}(\boldsymbol{x}_i) + \rho h(\boldsymbol{x}_i; \mathbf{a_m})\right).$$

Furthermore, Friedman (2001) proposed a regularization parameter $0 < \nu \leq 1$, which is applied as a multiplicative factor to the step length at each iteration. This factor, also known as the learning rate, is typically small (e.g., $\nu = 0.1$), and it determines the step size taken at each iteration m. The corresponding update can then be defined as

$$\nu\rho_m h(\boldsymbol{x}; \mathbf{a_m}).$$

See Hastie et al. (2009) for more details about the algorithm of the gradient boosting method. Extreme Gradient Boosting, or XGBoost, is a powerful and efficient machine learning algorithm built upon the principles of gradient boosting. It was introduced by Chen and Guestrin (2016) and incorporates a Newton step instead of the standard gradient step, which enhances performance and helps reduce overfitting (Friedman et al., 2000). Here, the second-order derivatives are considered in addition to the negative gradient, which are given by

$$\delta_m(\boldsymbol{x}_i) = \left[\frac{\partial^2 L(Y_i, g(\boldsymbol{x}_i))}{\partial g(\boldsymbol{x}_i)^2}\right]_{g(x)=g_{m-1}(x)}.$$

The Newton step can be obtained by solving

$$\mathbf{a_m} = \arg\min_{\mathbf{a}} \sum_{i=1}^{n} \left(\gamma_m(\boldsymbol{x}_i) h(\boldsymbol{x}_i, \mathbf{a}) + \frac{1}{2}\delta_m(\boldsymbol{x}_i) h(\boldsymbol{x}_i, \mathbf{a})^2\right)$$

The step length is subsequently defined as $\nu h(\boldsymbol{x}; \mathbf{a_m})$.

The extreme gradient boosting method is widely used for both regression and classifica-

tion tasks. In the field of sports, XGBoost has been successfully applied in areas like tennis and football prediction (Pan et al., 2024; Markopoulou, 2024; Yigit et al., 2020 and Groll et al., 2021).

One important aspect of using XGBoost is that it involves several tuning parameters to control the model's behavior and performance. In our study, we tuned the following parameters:

- **`max_depth`**: The maximum depth of the trees in the boosting ensemble. Higher values allow the trees to capture more complex patterns but may lead to overfitting, while lower values may underfit the data. Typical range is 3 to 10 (default is 6) (Chen and Guestrin, 2016).
- **`nthread`**: Specifies the number of threads to be used for processing (Chen and Guestrin, 2016).
- **`learning rate`**: Controls the step size in each boosting step and the contribution of each tree to the final prediction. A smaller learning rate reduces the risk of overfitting by shrinking the contribution of each tree but requires more boosting rounds (`nrounds`) to converge. Typical range values are 0.01 to 0.3, default is 0.3 (Contributors, 2024).
- **`nrounds`**: The number of boosting rounds (or trees) to train. A larger number of rounds increases model complexity but can also lead to overfitting, especially if the learning rate is high. The range depends on the dataset, but it usually falls between 50 and 1000 (Contributors, 2024).

The parameters `max_depth`, `eta`, and `nrounds` are critical for controlling the complexity, speed, and convergence of the XGBoost model. Regularization and parallelization (using `nthread`) further help to improve training efficiency and reduce the risk of overfitting. Finding optimal values for these parameters often involves tuning techniques like grid search or random search. For our approach, we specified reasonable, discrete grids and used a self-implemented 10-fold cross validation to determine the optimal tuning parameters. The approach is implemented using the `xgboost` function from the `xgboost` R package (Chen et al., 2019).

2.3.4 Support vector machine (SVM)

Support vector machine (SVM) is a supervised learning algorithm for classification tasks, introduced by Cortes and Vapnik (1995). The fundamental idea behind SVM is to find an optimal decision boundary, or hyperplane, which separates data points into distinct classes with the maximum possible margin.

In a binary classification task, SVM identifies a boundary that best separates data points of one class from those of the other. The algorithm maximizes the distance between this hyperplane and the nearest data points from each class, known as support vectors. Only

these closest points influence the position and orientation of the hyperplane, determining the optimal boundary that maximizes separation.

When the data is linearly separable, SVM finds a straight hyperplane that maximizes the margin between classes. For non-linearly separable data, SVM uses kernel functions to transform the original data into a higher-dimensional space where a separating hyperplane can be found. Commonly used kernels include:

- Linear Kernel: suitable for linearly separable data.
- Polynomial Kernel: adds polynomial terms, enabling more complex boundaries.
- Radial Basis Function (RBF) Kernel: popular for capturing complex relationships in non-linear data.
- Sigmoid Kernel: similar in behavior to neural networks.

SVM also includes a regularization parameter C, which controls the trade-off between maximizing the margin and minimizing classification error. A higher C value attempts to classify every point correctly, while a lower C allows more misclassifications, helping the model generalize better.

There are several references that cover a range of SVM applications, illustrating the versatility of SVM in handling both linear and non-linear data across various fields. For example, in image classification and facial recognition, Osuna et al. (1997) applies SVM to facial recognition, specifically in distinguishing faces from non-face images, demonstrating that SVM's margin-maximizing approach can improve accuracy in image classification tasks. In the field of medical diagnostics, Guyon et al. (2002) applies SVM for feature selection in cancer classification, highlighting its effectiveness in bioinformatics and showing SVM's capability in analyzing complex biological data to improve diagnostic accuracy. In sports analytics, Wei et al. (2013) uses SVM to classify different types of tennis serves based on player style and motion, showing that by analyzing shot characteristics, SVM can accurately classify serve types and contribute to more advanced player analytics. Additionally, Hughes et al. (2007) explore advancements in computerized notational analysis, showing how SVM can support coaching strategies by identifying winning patterns and key performance indicators in tennis, highlighting SVM's role in analyzing historical match data to uncover predictive patterns for strategic improvement. Finally, in Wei et al. (2015), SVM is again employed to classify types of tennis serves based on player style and motion, reinforcing the model's effectiveness in advanced player analytics. Figure 2.7 illustrates an exemplary data situation in a binary classification problem, where two different classes are represented by circles and diamond shapes (Savareh et al., 2018). The support vectors, which are the points closest to the hyperplane, are highlighted in dark shade. A hyperplane separates the classes, and the margins, indicated by dashed lines, represent the distance between the hyperplane and the support vectors.

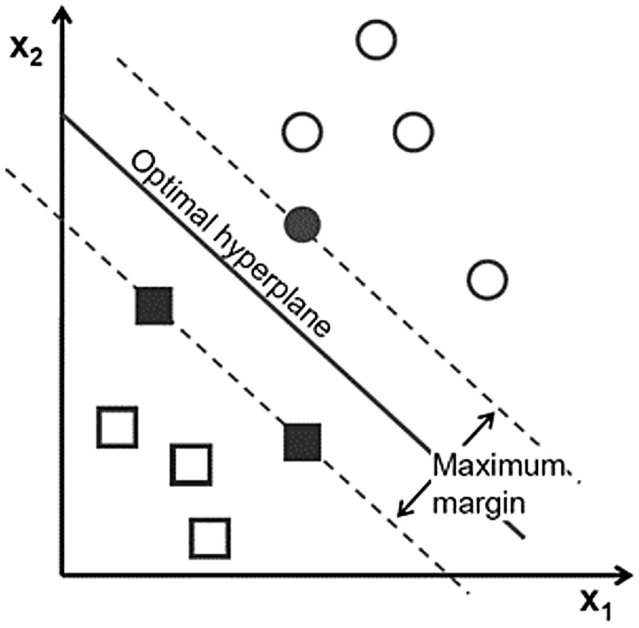

Figure 2.7: Graphical illustration of classification problem for two classes (Savareh et al., 2018). The solid line corresponds to the decision boundary (hyperplane), and the dashed lines visualize the margin.

The mathematical definition of a hyperplane in two dimensions is given by James et al. (2013) as

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0\,, \tag{2.3}$$

for parameters β_0, β_1 and β_2. Any $\boldsymbol{x} = (x_1, x_2)^T$ for which equation (2.3) holds is a point on the hyperplane. Equation (2.3) can be extended to the p-dimensional case by

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p = 0\,, \tag{2.4}$$

with $\boldsymbol{x} = (x_1, x_2, ..., x_p)^T$ being a vector of length p. Again, if $\boldsymbol{x}$ satisfies equation (2.4), then $\boldsymbol{x}$ lies on the hyperplane. In case that $\boldsymbol{x}$ does not satisfy equation (2.4), then this indicates that $\boldsymbol{x}$ lies on one side of the hyperplane, e.g.

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p > 0.$$

Otherwise, if

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p < 0\,,$$

then $\boldsymbol{x}$ lies on the other side of the hyperplane.

Suppose that we have an $n \times p$ data matrix $\boldsymbol{X}$ that consists of n training observations in the p-dimensional space with corresponding binary responses falling into two classes, that is, $y_1, ..., y_n \in \{-1, 1\}$, where -1 represents one class and 1 the other class.

Then, a separating hyperplane has the property that

$$y_i \cdot (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) > 0$$

for all $i = 1, ..., n$.

The maximal marginal hyperplane is the separating hyperplane for which the margin is largest. For a set of n training observations $\boldsymbol{x}_1, ..., \boldsymbol{x}_n \in \mathbb{R}^p$ and associated class labels $y_1, ..., y_n \in \{-1, 1\}$, the maximal margin hyperplane is e.g. defined by James et al. (2013) as the solution to the following optimization problem. Let $M > 0$ represent the margin of our hyperplane. Then, we want to maximize M and optimize $\beta_0, \beta_1, \ldots, \beta_p$ in

$$y_i \cdot (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M\,, \qquad \forall i = 1, ..., n\,,$$

subject to $\sum_{j=1}^{p} \beta_j^2 = 1$. Further details about SVM can be found e.g. in James et al. (2013).

Often classes are not perfectly separable, then so-called slack variables $\xi_1, \ldots, \xi_n$ are introduced in additional constraints

$$\begin{aligned} y_i \cdot \left(\boldsymbol{x}_i^\top \boldsymbol{\beta} + \beta_0\right) &\geq M\,(1 - \xi_i)\,, i = 1, \ldots, n \\ \xi_i &\geq 0, i = 1, \ldots, n \\ \sum_{i=1}^{n} \xi_i &\leq C, \end{aligned}$$

with $C \geq 0$ constant. These slack variables allow some flexibility compared to perfect separation, and ξ_i represents the extent to which observation i lies on the wrong side of its margin line, relative to the total margin violations M (see, e.g., Hastie et al., 2009). For $0 < \xi_i \leq 1$, observation i lies between its margin line and the hyperplane, and for $\xi_i > 1$ it lies on the wrong side of the hyperplane. Observations on the correct side of the margin yield $\xi_i = 0$. Hence, C is a tuning parameter that restricts the total amount of observations that may lie on wrong sides. Only those observations with $\xi_i \geq 0$ affect the hyperplane and are called support vectors.

In this work, we employ two types of SVM models: the linear SVM and the non-linear SVM with a radial basis function (RBF) kernel.

In order to define a proper kernel, the kernel function k must be a proper inner product for an introduction to inner product spaces (see Young, 1988). In particular, the kernel matrix $\mathbf{K}$ with $K_{i,j} := k(x_i, x_j)$ must be positive semi-definite. One of the most popular kernels is the RBF kernel, defined by

$$k^{\text{RBF}}(x_1, x_2) := \exp\left(-\frac{\|x_1 - x_2\|^2}{2\gamma^2}\right),$$

where γ is an additional kernel hyperparameter. A manual grid search over hyperparameter values is conducted to fine-tune the parameters for the RBF kernel, identifying the optimal SVM model. The performance of this tuned SVM model is then assessed using the testing dataset, with the process carried out using the `svm` function from the `e1071` R package (Dimitriadou et al., 2009).

In Buhamra et al. (2025), we conduct a comparative analysis to predict and evaluate the

performance of machine learning methods versus traditional regression techniques obtained from Buhamra et al. (2024) in the context of sports data, specifically focusing on men's Grand Slam tennis tournaments.

2.3.5 Artificial neural networks (ANNs)

An alternative approach that can model highly complex functions of the input features is artificial neural networks (ANNs). The term "neural network" has its origins in the field of neuroscience and early computational models inspired by biological neural systems. The concept dates back to McCulloch and Pitts (1943), who developed the first mathematical model of artificial neurons. Their work laid the foundation for ANNs, influencing later developments like perceptrons (Rosenblatt, 1958) and deep learning models. ANNs are composed of layers of interconnected nodes, or neurons, which process input data through weighted connections and activation functions. Typically, an ANN consists of an input layer, one or more hidden layers, and an output layer, where each layer transforms the data before passing it to the next.

In this subsection, we focus on neural networks as effective models for statistical pattern recognition. Specifically, we concentrate on the class of neural networks that have demonstrated the highest practical value, namely the multilayer perceptron. This representation has the potential to reveal significant relationships between features that a lower-dimensional model might overlook. However, employing ANNs also introduces new challenges.

ANNs can capture complex relationships between various match features (in the field of sport analytics). However, they are often considered 'black box' models, as they do not provide transparency into the decision-making process, making interpretation difficult. In addition to tuning hyperparameters, developing an effective tennis prediction model requires a careful selection of these parameters to achieve optimal performance. Moreover, ANNs are especially prone to overfitting, particularly when the number of hidden layers or neurons is large compared to the number of training examples. So one could say that an ANN's ability to model non-linear relationships comes at the cost of increased susceptibility to overfitting, a larger number of hyperparameters to tune, and higher computational expense. Nevertheless, this interesting model merits our attention and further investigation.

In such a model, each neuron computes a value based on its inputs, which is then passed on to other neurons. A feedforward network is represented as a directed acyclic graph. Typically, ANNs are structured into multiple layers, with neurons in non-input layers being fully connected to all neurons in the preceding layer. Each connection in the network has an associated weight, and a neuron uses these inputs and their corresponding weights to calculate an output value. A common approach for this calculation is a non-linear weighted sum:

$$f(x) = K\left(\sum_{i=1}^{n} w_i x_i\right),$$

where w_i is the weight of input x_i. The non-linear activation function $K(\cdot)$ enables the network to solve complex problems using a relatively small number of neurons. Among the various sigmoid functions, the logistic function is commonly used for this purpose. Match prediction can be performed by feeding the player and match features into the neurons of the input layer and propagating the values through the network. When a logistic activation function is applied, the network's output can represent the probability of winning the match. Which means if a logistic activation function is used, the output of the network can represent a probability or a classification score. Specifically, the logistic function maps input values to a range between 0 and 1, making it suitable for binary classification tasks where the output can be interpreted as the probability of one class (e.g., class 1) occurring. The closer the output is to 1, the larger the probability that the input belongs to the positive class, and the closer it is to 0, the larger the probability it belongs to the negative class.

There are various training algorithms designed to optimize the network's weights in order to produce the most accurate outputs for a given set of training examples. For instance, the backpropagation algorithm utilizes gradient descent to minimize the mean square error between the target values and the network's outputs. Somboonphokkaphan et al. (2009) used a three-layers feedforward ANN for match prediction, employing the backpropagation algorithm. Multiple networks with different sets of input features were trained and compared. The best-performing model had 27 input nodes, representing both player and match features, and achieved an average accuracy of approximately 75% in predicting match outcomes during the 2007 and 2008 Grand Slam tournaments.

Considering the network structure, the ANN used in this work follows a multilayer perceptron (MLP) architecture, which is a type of feedforward network where information flows in one direction from the input layer through one or more hidden layers to the output layer. The network is trained using backpropagation, a gradient-based optimization method that adjusts weights to minimize prediction errors.

This structure enables MLPs to learn complex patterns and make accurate predictions. These networks are widely used in tasks such as image recognition, speech processing, and predictive analytics due to their ability to model complex patterns effectively. Within the frame work of sport analytics, particularly in tennis, they are used to predict match outcomes, assess player performance, and analyze playing styles. By processing features such as player rankings, recent form, head-to-head records, and match conditions, ANNs can identify patterns and generate probabilities for match predictions.

One key aspect of the network structure is hyperparameter tuning, which involves adjusting key parameters to optimize model performance and improve its ability to generalize to new data. These hyperparameters control the learning behavior and impact the accuracy of the neural network. In this work, we focused on the following parameters:

- **`size`**: The number of neurons in the hidden layer, which defines the model's complexity. A larger number of neurons can result in a more complex model but also increases the risk of overfitting. Therefore, grid search is used to select the values that minimize the

error.

- **`decay`**: This parameter acts as a regularization factor. A smaller decay value will correspond to a larger learning rate, while a larger decay value corresponds to a slower learning rate. Tuning decay helps control overfitting and the rate at which the network learns. The best values for decay is obtained by defining grid search.

- **`maxit`**: The maximum number of iterations (epochs) for training. If the decay is large, fewer iterations may be required for convergence. If the decay is low, more iterations may be needed for the model to converge. The convergence-based is used for tuning the `maxit` parameter. It adjusts the number of iterations dynamically based on whether the loss is still improving, helping to find a suitable stopping point for the neural network training process. This approach avoids the need for exhaustive searching over a range of `maxit` values. I.e., the method adapts the number of iterations based on the model's convergence, ensuring the training process is computationally efficient while still trying to improve model performance.

- **`activation function`**: The most commonly used activation functions in neural networks are sigmoid-based functions. One example is the logistic (sigmoid) function, which restricts output values to the range [0,1] and is frequently used in logistic regression. Another option is the hyperbolic tangent (tanh) function, which outputs values between [-1,1]. According to LeCun et al. (2012), symmetric activation functions like tanh(·) can improve training efficiency by reducing training time. However, here we employed the sigmoid function for the classification task, as it is well-suited for predicting binary outcomes such as match wins and losses.

As shown by Hornik (1991), a single hidden layer with a finite number of neurons is sufficient to approximate any continuous function. In this work, we used a single hidden layer. This concept is known as the universal approximation theorem for neural networks Csáji et al. (2001). However, the number of neurons in the hidden layer remains a hyperparameter that needs to be optimized. Note that a neural network without hidden layers is essentially equivalent to logistic regression. It is the inclusion of hidden layers that enables the network to perform non-linear classification.

Further background and detailed information on ANN models can be found in Bishop (1995) and Bishop and Nasrabadi (2006).

A visualization of a fully connected feedforward neural network is shown in Figure 2.8. A network is considered feedforward as the connections between neurons do not form a cycle. A neural network always consists of at least an input layer and an output layer. Additionally, the network may include one or more intermediate layers that connect the input and output layers. These intermediate layers are called *hidden* layers. Figure 2.8 illustrates the structure of the ANN, which includes input nodes in the input layer, a variable number of nodes in the hidden layer, and a single output node.

Each layer consists of a specific number of neurons, with the number of neurons in the input and output layers determined by the requirements of the problem or the specific case being investigated. In our example, seven input neurons are used, each corresponding to a feature. The output layer consists of a single neuron, as we only need to predict the probability of a win. The values of the neurons in the input layer are equal to the corresponding feature. The values of a neuron h_j in the subsequent layer is computed as follows:

$$h_j = \phi\left(b_j + \sum_{i=1}^{n} \boldsymbol{x}_i w_{ij}\right), \tag{2.5}$$

where b is trainable bias weight unique to every neuron and $\phi(\boldsymbol{x}_i)$ is an activation function. The bias weight fulfills a similar role to the intercept in logistic regression and is not strictly required for our problem. The activation function introduces non-linearity into the model. For this, we used commonly chosen and suitable activation functions, specifically the logistic sigmoid function (Bishop and Nasrabadi, 2006; Goodfellow et al., 2016 and Orr and Müller, 1998).

By calculating equation (2.5) for each neuron beyond the input layer, the neuron values are propagated through the network layers until they reach the output layer. The values of the output layer neurons represent the neural network's predictions. For binary classification task, a single output neuron with the logistic (sigmoid) activation function is used. During training, the network weights are optimized using a gradient descent approach, with backpropagation employed to compute the gradients (Hecht-Nielsen, 1992).

The implementation is done in R using the `nnet` package by Ripley (2002). For visualization, the `plotnet` function from the `NeuralNetTools` package (Beck, 2018) is used.
In the following, we will describe the layers (structure) of the ANN corresponding to Figure 2.8.

Input Layer

The input layer represents the features used to predict the outcome. In this model, the input features are Prob, Elo, Points, Age, Age.30, Age.int and Rank.

Hidden Layer

The hidden layer processes input features and extracts meaningful patterns. This model uses **3 hidden nodes** (`size = 3`), which act as intermediate computational units.

Output Layer

The output layer generates the final prediction. Since this is a binary classification problem (win or loss), the output layer consists of a single neuron, which means that the output is a value between 0 and 1, representing the probability of winning.

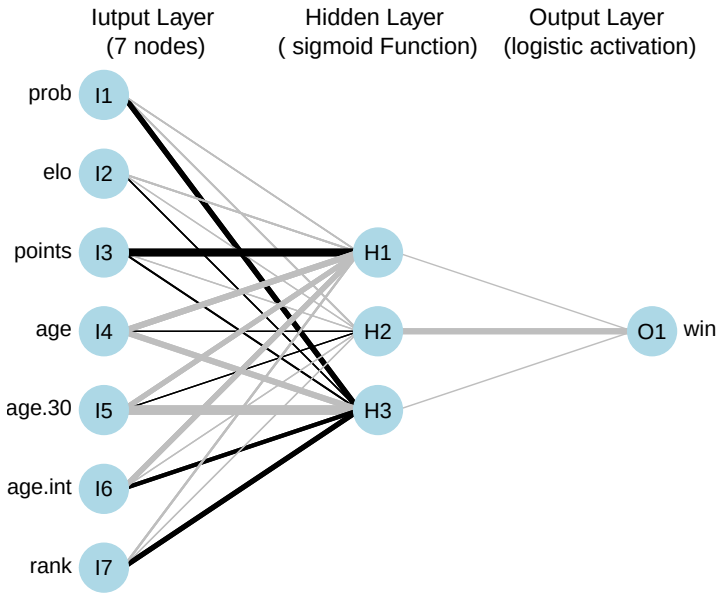

Figure 2.8: Example of a three layers fully connected feedforward neural network.

The thickness of the arrows in the plot indicates the strength of the connection between neurons, which is based on the estimated weights from the trained neural network model.

The arrows between nodes (connections) represent the weights, which are adjusted during training. The thickness of the arrows in the plot indicates the strength of the connection between neurons, which is based on the estimated weights from the trained neural network model. Thicker arrows represent higher-weighted connections, while thinner ones indicate lower weights. Activation functions introduce non-linearity, enabling the network to learn complex patterns. Sigmoid with outputs values between (0,1) is used as it useful for binary classification.

2.4 Interpretable machine learning (IML)

Simple models, such as linear regression or decision trees, are inherently more interpretable due to their straightforward relationships between input and output. However, this is not the case for more complex models like RF or XGBoost, which lack this inherent transparency. Interpretability in machine learning refers to the ability to understand and explain how a model makes its predictions or decisions. In other words, interpretability enables one to identify the factors influencing a model's outcomes and the relationships within the data that shape those results. Therefore, interpretable machine learning (IML) tools, such as partial dependence plot (PDP) and individual conditional expectation (ICE) plot, are used to provide insights into the decision-making process of complex models. The goal of using IML is to create models that not only yield accurate predictions but also offer explanations

for those predictions.

2.4.1 Partial dependence plot (PDP)

The partial dependence plot (PDP) is a global interpretation tool that represents expected values based on the data distribution (Friedman, 2001). It illustrates the average effect of a specific feature on predictions while averaging out the influence of other features. As a global interpretation method, it reveals the overall behavior of the model by considering all instances and providing insights into the relationship between a feature and the predicted outcome. Consequently, PDP is particularly useful for understanding general patterns in the data and for diagnosing model performance.

The PDP is a feature effect method, which illustrates the marginal effect of one or two features on the predicted outcome of a machine learning model (Friedman, 2001). The partial dependence function for regression is expressed as

$$\hat{f}_S(\boldsymbol{x}_S) = \mathbb{E}_{\boldsymbol{x}_C}[\hat{f}(\boldsymbol{x}_S, \boldsymbol{x}_C)] = \int \hat{f}(\boldsymbol{x}_S, \boldsymbol{x}_C)\, dP(\boldsymbol{x}_C)\,.$$

where $P(\cdot)$ is the distribution of the features in set C and the features in $\boldsymbol{x}_S$ are those for which the partial dependence function is plotted, while $\boldsymbol{x}_C$ represents the other features used in the machine learning model $\hat{f}(\cdot)$, which are considered as random variables in this context. Typically, the set S contains only one or two features, namely those whose effect on the prediction one aims to understand. The feature vectors $\boldsymbol{x}_S$ and $\boldsymbol{x}_C$ together form the complete feature space $\boldsymbol{x}$. Partial dependence operates by marginalizing the model's output over the distribution of the features in set C, allowing the function to reveal the relationship between the features in S and the predicted outcome. By doing so, we obtain a function depending solely on the features in S, while still accounting for interactions with other features. The partial function $\hat{f}_S$ is given by

$$\hat{f}_S(\boldsymbol{x}_S) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(\boldsymbol{x}_S, \boldsymbol{x}_{i,C})\,,$$

i.e., it is estimated by calculating averages on the training data. The partial function shows the average marginal effect on the prediction for given values of the features in S. Here, $\boldsymbol{x}_{i,C}$ denotes the actual values from the dataset for the features not of interest, and n represents the number of instances in the dataset. A key assumption of the PDP is that the features in C are uncorrelated with those in S. If this assumption is not met, the computed averages may incorporate data points that are very unlikely or even impossible.

In classification tasks where the machine learning model produces probabilities, the partial dependence plot shows the probability of a specific class based on various values of the feature(s) in S.

In addtion, Greenwell et al. (2018) introduced a simple measure of feature importance that relies on partial dependence. The core concept is that a flat PDP implies the feature is

not significant, whereas increased variability in the PDP signifies greater importance of the feature.

The importance for numerical features is determined by the deviation of each unique feature value from the average curve (Greenwell et al., 2018) and is given by

$$I(\boldsymbol{x}_S) = \sqrt{\frac{1}{K-1}\sum_{k=1}^{K}\left(\hat{f}_S(\boldsymbol{x}_S^{(k)}) - \frac{1}{K}\sum_{k=1}^{K}\hat{f}_S(\boldsymbol{x}_S^{(k)})\right)^2}.$$

Note that here the $\boldsymbol{x}_S^{(k)}$ are the K unique values of feature x_S. The range of the PDP values for the unique categories or categorical features is

$$I(\boldsymbol{x}_S) = \frac{\max_k(\hat{f}_S(\boldsymbol{x}_S^{(k)})) - \min_k(\hat{f}_S(\boldsymbol{x}_S^{(k)}))}{4}.$$

This method for calculating deviation is referred to as the range rule. It provides a rough estimate of deviation when only the range is known. The denominator of four is derived from the properties of the standard normal distribution.

Finally, this PDP-based feature importance measure only captures the primary effect of the feature while ignoring any potential interactions with other features. Another, drawback of this measure is that it is based solely on unique values. Consequently, a unique feature value with only one instance is assigned the same importance as a value that has many instances when computing feature importance. For further details and examples see Molnar (2020).

2.4.2 Individual conditional expectation (ICE)

The individual counterpart to a PDP is known as the individual conditional expectation (ICE) plot (Goldstein et al., 2015). Unlike partial dependence plots, which show an average effect across all data points, ICE plots are especially useful for observing individual-level variations in predictions. ICE plots display separate curves for each instance in the dataset, illustrating how predicted values change as the feature of interest is varied. They are particularly effective in cases where feature interactions exist, as they allow us to see how different instances might respond differently even under the same feature values.

The purpose of examining individual expectations alongside partial dependencies is to gain a deeper understanding of how a feature affects predictions on an individual basis. PDPs reveal the average relationship between a feature and the prediction, which works well when interactions between the target feature and other features are minimal. However, in cases where interactions are present, ICE plots offer significantly more insight by displaying how each instance uniquely responds to changes in the feature. This means that in ICE plots, for each instance in $\{(\boldsymbol{x}_S^{(i)}, \boldsymbol{x}_C^{(i)})\}_{i=1}^N$, the curve $\hat{f}_S^{(i)}$ is plotted against $\boldsymbol{x}_S^{(i)}$, while $\boldsymbol{x}_C^{(i)}$ remains fixed.

A limitation of ICE plots is that it can sometimes be challenging to discern differences between individual curves due to variations in their starting predictions. A straightforward

solution is to center the curves at a specific value of the feature, showing only the difference in predictions relative to this point. This modified plot is known as a centered ICE plot (c-ICE). Anchoring the curves at the lower end of the feature range is often effective. According to Goldstein et al. (2015), the centered curves are then defined as follows

$$\hat{f}_{\text{cent}}^{(i)} = \hat{f}_i - \mathbf{1}\hat{f}(x^a, x_C^{(i)}),$$

where $\mathbf{1}$ is a vector of ones with the appropriate dimensions (typically one or two), $\hat{f}$ represents the fitted model, and x_a denotes the anchor point.

An alternative approach to enhance the visual identification of heterogeneity is to examine the individual derivatives of the prediction function concerning a feature. This results in the development of a derivative ICE plot (d-ICE). The derivatives of a function (or curve) reveal whether changes are occurring and their direction. By using a derivative ICE plot, it becomes easier to pinpoint ranges of feature values where the predictions from the black box model vary for some instances.

If there is no interaction between the feature being analyzed, x_S and the other features, x_C, then the prediction function can be formulated as

$$\hat{f}(x) = \hat{f}(x_S, x_C) = g(x_S) + h(x_C),$$

with

$$\frac{\delta \hat{f}(x)}{\delta x_S} = g'(x_S).$$

When there are no interactions, the individual partial derivatives should remain consistent across all instances. If there are differences, it indicates the presence of interactions, which can be observed in the d-ICE plot. Generally, The derivative ICE plot takes a long time to compute and is rather impractical (Molnar, 2020).

Chapter 3

Summary of the articles

3.1 Article 1: Modeling and prediction of tennis matches at Grand Slam tournaments

In the following, a summary of Buhamra et al. (2024) is presented. Through this work, we explore various approaches for modeling and predicting the outcomes of Grand Slam tennis tournaments from 2011 to 2022, specifically focusing on the probability that the first-named player wins. In our analysis, we account for covariate differences by calculating the difference between the corresponding values of the two players. For instance, the ranking difference is obtained by subtracting the second player's Rank from the first player's Rank, and the same approach is applied to other covariates such as Age and Elo rating.

Our goal is to determine which modeling approaches are best suited for this purpose. All methods are based on regression models and incorporate multiple potential covariates.

The evaluation of these regression models is conducted using both leave-one-tournament-out cross validation and external validation, with respect to three performance measures: classification rate, predictive Bernoulli likelihood, and Brier score.

To predict and compare the results of the 2022 tournaments with actual outcomes, we apply a "rolling window" strategy with a continuously updating dataset.

Additionally, we investigate the impact of incorporating non-linear effects and additional court- and player-specific attributes to assess their relevance. Those two specific aspects are examined in more detail. This analysis aims to leverage the flexibility and interpretability of modern regression methods to gain insights into key associations and relationships in professional tennis. Furthermore, we introduce the concept of "newcomer" players to estimate the ability effects for those who have participated in only one tournament.

Our main focus is on the logistic regression model, where a binary target variable y and covariates $x_1, \ldots, x_p$ are given. In our case, $y = 1$ denotes the occurrence of a particular event (defined as "win") and y = 0 that the event does not occur (defined as "lose"). We aim to model π_i appropriately as a function of the feature variables.

$$\pi_i = P(y_i = 1|x_{i1}, \ldots, x_{ip}) = E[y|x_{i1}, \ldots, x_{ip}]$$

hence, the linear predictor η_i is related to the probability π_i by a strictly monotonically increasing function $h\colon \mathbb{R} \to [0, 1]$, we can also write

$$\pi_i = h(\eta_i) = h(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ip})\,.$$

When the number of covariates p becomes very large, estimation may become numerically unstable. To address this issue, regularization is applied using the least absolute shrinkage and selection operator (LASSO; Tibshirani, 1996). Here, the penalty term is expressed by

$$\text{pen}(\boldsymbol{\beta}) = \sum_{j=1}^{p} |\beta_j|.$$

It allows model estimation and variable selection to be performed in one step, as small coefficients are shrunk to 0. In our case, this approach was applied to both surface-specific player skills and global player skills. For the former, since tennis is played on various surfaces, players' performance may vary depending on the surface. To capture this effect, a factor variable was introduced, leading to a large set of new (dummy-encoded) covariates. However, this can result in an excessive number of parameters and unstable estimates. To address this issue, logistic regression was combined with LASSO regularization. A similar approach was applied to global player skills. Further details can be found in Section 4.1 of the corresponding article. As the effect of covariates on the target variable can also be non-linear, splines can be used to capture this flexibility. Therefore, we introduce P-splines, which incorporate a penalty term to enhance regularization. Specifically, we use P-splines based on B-splines (Eilers and Marx, 1996). Then, all proposed linear and non-linear models, including the benchmark model based on probabilities calculated from the average odds of different bookmakers, are compared with respect to their predictive performance using leave-one-tournament-out cross validation strategy to select the best model in terms of the perviously mentioned performance measures. Subsequently, an external validation method "rolling window" is conducted to evaluate the best models identified in the corresponding cross validation approach. We also consider betting profit as an additional method to compare the predictive quality of different models.

The results indicate that within the leave-one-tournament-out cross validation approach, all models performed very similarly in terms of classification rate, likelihood, and Brier score. However, when comparing betting profits and the number of bets placed, it was found that only the spline models yielded positive betting returns, though they placed fewer bets. Therefore, it can be inferred that these models are more conservative and potentially safer in terms of betting (see Table 3.1).

The 2022 tournaments were then used as an external validation dataset. The findings from the cross validation comparison for the rolling window approach were largely confirmed,

Table 3.1: The values of betting profits and the number of bets placed for leave-one-tournament-out cross validation approach for different models approaches including linear, splines and LASSO. In brackets is the ratio of all matches in which a bet was placed. The values of the winner models are highlighted in bold font.

	Explanatory variables	Betting profit	Amount of bets
Linear	Prob	−128.8	2696 (57.1%)
	Points, Prob	−99.9	2395 (50.7%)
	Rank, Prob	−110.9	2707 (57.3%)
	Age, Prob	−150.3	2716 (57.5%)
	Rank, Points, Prob	−104.3	2418 (51.2%)
Splines	Prob	**113.7**	1170 (24.8%)
	Rank, Prob	**113.7**	1170 (24.8%)
	Points, Prob	101.3	1298 (27.5%)
	Rank, Points, Prob	101.8	1299 (27.5%)
	Elo, Prob	106.9	1175 (24.9%)
LASSO	Surface specific	−225.3	1891 (40.1%)
	General skills	−199.3	2002 (42.4%)
	Benchmark	−311.4	1359 (28.8%)

with performance measure values generally showing slight improvements over those from the CV. Regarding betting returns, only the spline models achieved a profit, except for the model using ranking Points and betting Odds, which resulted in a small loss. The proportion of bets placed increased for all models, although the benchmark model placed significantly fewer bets than in the cross validation comparison. Notably, the benchmark model again performed no better than the other models across most performance measures. Additionally, it was observed that among the linear models, the covariates Rank and Prob yielded the best results. Within the spline approach, all three models performed nearly equally well. The LASSO models generally performed worse than the other models and therefore are not recommended.

In summary, we propose several regression models for predicting Grand Slam tennis tournaments from 2011 to 2022, focusing on the probability that the first-named player would win. Models were evaluated using leave-one-tournament-out CV and rolling window approach across particular performance measures. The study also considered non-linear effects, player and court-specific abilities, and the inclusion of newcomer players. Spline models using P-splines showed positive betting returns and demonstrated a conservative approach to betting. Validation on 2022 data confirmed spline models as top performers, while LASSO models were less effective. Also, it was notable that all models performed at least as well as the benchmark model.

3.2 Article 2: Comparing modern machine learning approaches and different forecast strategies on Grand Slam tennis tournaments

Machine learning has recently demonstrated promising results in sports prediction. In Buhamra et al. (2025), we provide various machine learning approaches for modeling and predicting outcomes of tennis matches in Grand Slam tournaments.

This study focuses on multiple machine learning approaches, including CT, RF, XGBoost, SVM and ANNs, applied to matches in men's Grand Slam tournaments from 2011 to 2022. We then compare these approaches with the regression models examined in our previous work from Article 1 (Buhamra et al., 2024).

The models are evaluated using the same validation strategies, dataset, and performance measures as in Article 1. However, within this work, we introduce an additional validation strategy with a fixed three-year training window.

This analysis considers a similar selection of potential covariates as in Article 1 (Buhamra et al., 2024), including player Age, ATP Ranking and Points, betting Odds, Elo rating, and two additional Age variables that take into account the "optimal" age range for tennis players, which is typically between 28 and 32 years (Weston, 2014).

Similar to Article 1, the first step involves evaluating the models using Grand Slam tournaments from 2011 to 2021. The 2022 tournaments are initially excluded and later utilized as an external validation dataset. To validate the models from the preceding CV phase, we assess their predictive performance on new, unseen data in a more realistic setting by applying an expanding validation forecasting approach. In this approach, each remaining tournament is used sequentially as a test dataset in chronological order, while the training dataset is updated and enlarged continuously. This scheme, as previously discussed in Buhamra et al. (2024), provides a progressive view of model performance. An alternative strategy is to use only recent data for prediction through a rolling window approach. Here, a fixed three-year training window is applied, with the training dataset rolling forward each year to include only the most recent three years of data. The results show that within the CV approach, the selected linear and spline models performed similarly in terms of classification rate, likelihood, and Brier score, whereas Brier score values for the machine learning models were more volatile.

For the tournaments from 2022, which were used as an external validation set in an expanding window validation approach, the performance measures generally showed slight improvement. However, performance varied across those performance measures: XGBoost achieved the largest classification rate, the linear SVM model excelled in predictive likelihood, and the spline model including both Points and Prob covariates attained the lowest Brier score, outperforming all other models.

In the rolling window approach, with a training dataset of 12 tournaments (three-years training window) used to predict each subsequent tournament, the same models as in the

expanding window validation were evaluated. The results were mostly consistent with the expanding window strategy, although performance measure values were generally slightly lower. Performance differed slightly between models and performance measures: the linear SVM model performed best in both classification rate and predictive likelihood, while the linear regression model with Points and Prob achieved the lowest Brier score. Table 3.2 provides a summarized overview for the results of ML models based on the performance measures with respect to the proposed validation strategies.

According to the results obtained from the ML models, only minor differences were observed across the three validation approaches, and the models performed similarly in terms of the performance measures.

Within the leave-one-tournament-out CV, the ANNs model achieved the best classification rate and Brier score among all models corresponding to this validation strategy, while the largest likelihood value was obtained by another model, namely SVM (linear).

In the expanding window approach, the XGBoost model achieved the best classification rate among all models and validation strategies, followed closely by the SVM (linear) with a value of 0.8295. Also, the SVM (linear) model outperformed the others in terms of predictive log-likelihood and Brier score.

For the rolling approach with a fixed 3-years window, the SVM (linear) model achieved the largest classification rate and also yielded the best likelihood value. However, the ANNs model outperformed others with respect to the Brier score measure.

Table 3.2: Results of the different proposed validation approach for the ML models; best performer in bold font.

	ML models	Class. rate	Likeli-hood	Brier score
Leave-one-tournament-out CV	ANNs	**0.8021**	0.6828	**0.1560**
	Random forest	0.7978	0.6817	0.1584
	SVM (Linear)	0.7961	**0.7415**	0.1675
	SVM (Radial kernel)	0.7829	0.6149	0.1807
	XGBoost	0.7777	0.7234	0.1901
	Classification tree	0.7486	0.6703	0.2178
Expanding validation approach	SVM (Linear)	0.8295	**0.7637**	**0.1446**
	XGBoost	**0.8306**	0.7526	0.1562
	Random forest	0.8162	0.6734	0.1511
	SVM (Radial kernel)	0.7956	0.5858	0.1882
	ANNs	0.7918	0.6904	0.1466
	Classification tree	0.7577	0.6803	0.2061
Rolling approach with a fixed 3-years window	SVM (Linear)	**0.8012**	**0.7424**	0.1709
	Random forest	0.7971	0.6727	0.1608
	XGBoost	0.7755	0.7231	0.2074
	SVM (Radial kernel)	0.7729	0.5448	0.2139
	ANNs	0.7726	0.6837	**0.1554**
	Classification tree	0.7256	0.6900	0.2678

To sum up, this study explores machine learning methods, including CT, RF, XGBoost, SVM and ANNs, to model and predict outcomes of men's Grand Slam tennis matches from

2011 to 2022, the same period as in Article 1 (Buhamra et al., 2024). These models are evaluated and compared with regression models using the same strategy as in Buhamra et al. (2024), with the addition of a rolling window validation strategy using a fixed 3-years window, focusing on the same performance measures. The results of the considered strategy (expanding window) indicate that XGBoost achieved the largest classification rate, SVM (linear) performed best in predictive likelihood, and spline models (with Points and Prob covariates) consistently attained the lowest Brier score.

3.3 Article 3: Statistical enhanced learning for modeling and prediction tennis matches at Grand Slam tournaments

Here, we demonstrate how feature engineering techniques can be effectively applied in sports analytics to enhance predictive models and derive valuable insights from sports data.

As our main focus is to analyze whether "statistically enhanced features" improve prediction performance, we particularly examine the use of covariates such as Elo, Age.30, and Age.int, which are not directly available but are derived either through separate modeling techniques (Elo) or meaningful mathematical transformations (Age.30, Age.int). These covariates are used to further enhance statistical models for tennis. Note that, the additional age variables were created to capture the "optimal" age range for a tennis player, typically considered to be between 28 and 32 years (Weston, 2014). Our objective is to investigate whether the enhanced variables have substantial potential to improve the performance of our proposed models, specifically when compared to more classical covariates, which are directly available, such as e.g. (Rank and Points).

Furthermore, as machine learning algorithms grow increasingly sophisticated, the need for interpretability grows as well, especially for models like random forests (RF), whose predictions can be challenging to understand.

Interpretable machine learning (IML) is a branch of machine learning focused on making complex models more transparent and understandable. In our work, we enhance model transparency by applying specific IML tools that provide insights into how these models make predictions, thereby making them more accessible and reliable.

We introduced combined PDP and ICE plots to comprehensively visualize the relationship between predictor variables and the predicted outcome. This dual perspective is essential for understanding the complexity of machine learning models: overlaying the PDP on the ICE plots reveals general trends (captured by the PDP) while also highlighting the variability in how individual observations respond to changes in each predictor (captured by the ICE lines). Individual Conditional Expectation (ICE) plots display the effect of a feature on the predicted outcome for each individual instance by plotting one line per observation. Each line shows how the prediction changes as the feature value varies, keeping all other features constant. This allows us to observe heterogeneity in predictions—different instances may respond differently to changes in the same feature, which is crucial for identifying interactions or non-linear effects (Goldstein et al., 2015). PDP shows the average effect of a feature on the model's prediction by averaging over the effects of other features. This helps to better understand the relationship between each feature and the prediction.

There are a wide range of applications across various domains within IML area, such as criminal justice Dressel and Farid (2018), Doshi-Velez and Kim (2017), finance Ustun and Rudin (2019), healthcare and medical fields Hu et al. (2022), Ouyang et al. (2023), and diabetes management Payrovnaziri et al. (2020), where IML, particularly Local interpretable model-agnostic explanations (LIME) by Ribeiro et al. (2016), is leveraged to provide individ-

ualized insights into factors affecting blood glucose levels. This analysis incorporates various methods from Buhamra et al. (2024; 2025), including linear regression, splines, and random forests. These approaches were then compared within an expanding window strategy to analyze tennis data from Grand Slam tournaments using the same performance measures as defined before. When analyzing the results, we find that values vary across different approaches and performance measures. In general, models tend to perform better when at least one of these enhanced variables is included. Examining PDPs provided insights into how each predictor variable influences the model's predictions. For instance, the PDP for the differences of Points shows a general upward trend, suggesting that accumulating more Points positively influences the predicted outcome. Likewise, the PDP for Age.30 indicates that being closer to the optimal age of 30 is slightly associated to large winning probabilities. Figure 3.1 and Figure 3.2 show the PDP and ICE plots for the covariate differences. These include the differences between both players in age (Age.30), ranking positions (Rank), ranking points (Points), and Elo ratings (Elo).

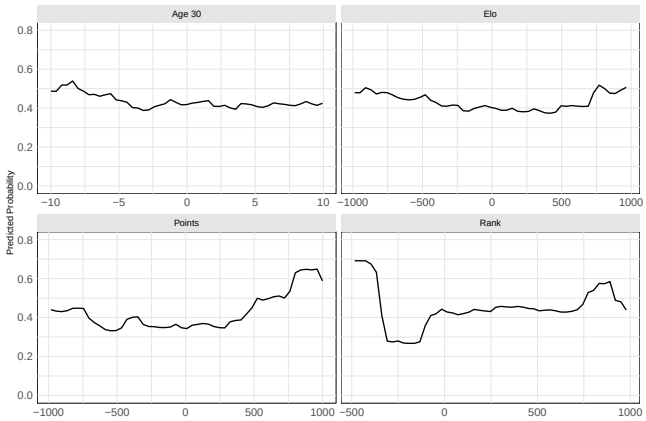

Figure 3.1: PDP plot for the random fores model includes Age.30, Elo, Points and Rank as covariates

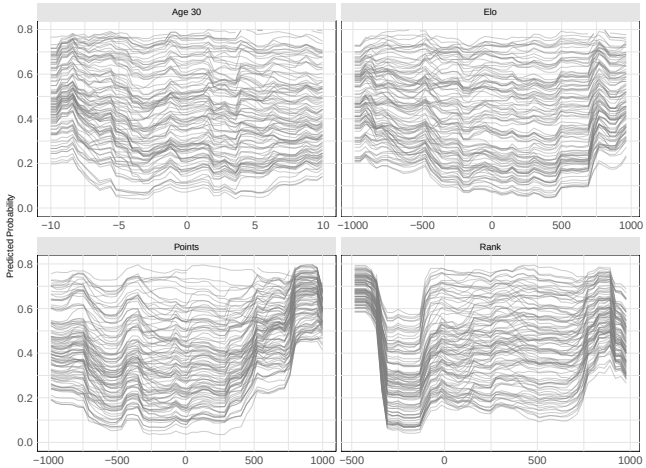

Figure 3.2: ICE plot for the random forest model including the covariates Age.30, Elo, Points and Rank

To sum up, we tested various models, including linear regression and random forests, on

tennis data to assess the impact of enhanced variables, such as Age and Elo, on predictive accuracy. Additionally, we used IML tools, such as PDP and ICE plots, to improve the interpretability of machine learning models like RF. In sports analytics, these tools, along with feature engineering, enhance model insights and performance.

Chapter 4

Discussion and outlook

This work presents a comprehensive comparison of classical regression models with modern machine learning approaches for modeling and predicting outcomes of men's Grand Slam tennis matches. Moreover, we analyze the predictive potential of so-called statistically enhanced covariates, and take procedures from the field of interpretable machine learning into account to make complex machine learning models more understandable.

As a starting point, logistic regression models with linear effects are presented. Given that the relationship between covariates and the target variable may not always be strictly linear, we also incorporate non-linear effects using spline-based approaches, which are based on penalized B-splines, i.e. P-splines. Additionally, we explore a regularization method, LASSO, as an extension to regression models. This approach is particularly useful when a large number of covariates are constructed, as it helps mitigate issues related to an excessive number of parameters and potential estimator instability. This regularization method enables model estimation and variable selection to be performed in one step by shrinking irrelevant coefficients to zero. Therefore, we investigate the inclusion of both surface-specific and global player skill parameters within this regularization framework. We also define the concept of a "newcomer", identifying players who have competed in only one Grand Slam tournament. In our analysis, we also consider betting returns based on the average probability of a win for each respective player, calculated from the average odds provided by several different betting providers, as included in the `deuce` package obtained from `https://www.oddsportal.com/`. We continue our analysis using classification rate, predictive Bernoulli likelihood, and Brier score through leave-one-tournament-out cross validation (CV) followed by external validation.

Our main finding from the classical regression comparison is that all models performed similarly in terms of classification rate, predictive likelihood, and Brier score within the cross-validation framework. However, in terms of betting profits and the number of bets placed, only the spline models generated positive returns, despite placing fewer bets. Regarding external validation, the performance measure values typically showed slight improvements compared to those from the cross validation. In terms of betting returns, all spline models generated a profit, except for the one that included Points and Prob as covariates, which

resulted in a loss. The LASSO models generally performed worse than the other models and are therefore not recommended. It is worth mentioning that the benchmark model performed no better than the other models across most performance measures.

Machine learning (ML) has gained enormous popularity and widespread usage in recent years, becoming an integral part of various applications. By enabling systems to learn from data and improve their performance over time, ML has demonstrated the ability to analyze large datasets, identify patterns, and make predictions, driving innovation and efficiency across sectors. Furthermore, ML is now more accessible than ever, allowing organizations to harness the power of data for better decision-making and enhanced user experiences. Therefore, we continue our analysis and extend our work by using the same dataset and performance measures, this time applying machine learning methods, with the aim of investigating whether these methods can enhance the predictions from our previous work in Buhamra et al. (2024). As a result, we conduct a comparative analysis of various machine learning models, including classification trees, random forests, XGBoost, SVM, and ANNs alongside the classical regression models used in our previous work. The models are validated through different strategies, namely cross validation, expanding window, and rolling window.

The main findings indicate that in the CV strategy, the selected linear and spline models exhibited similar performance in terms of classification rate and predictive likelihood. However, the Brier score values for the ML models exhibited larger volatility, except for the RF and ANN models, which produced acceptable values comparable to those of the other models. In the expanding window approach, the performance measure values demonstrated slight improvements. However, variations were observed among these measures: XGBoost recorded the highest classification rate, the linear SVM model excelled in predictive likelihood, and the spline model that incorporated both the Points and Prob covariates achieved the lowest Brier score, outperforming all other models. Within the rolling window approach, the performance measure values generally somewhat deteriorated, with slight differences across models and measures; this time, the linear SVM model outperformed the others, achieving higher results in both classification rate and predictive likelihood, whereas the linear regression model that included Points and Prob attained the lowest Brier score.

Then, in order to improve predictive performance, we investigated enhanced covariates, particularly Elo ratings and Age variables. This approach demonstrates that models incorporating these enhanced variables yield better accuracy. Furthermore, to enhance the interpretability of complex ML models, specifically the RF, we focus on IML tools. By employing techniques like PDPs and ICE plots, we visually clarify the effects of predictor variables on model predictions, thereby increasing transparency and accessibility. PDPs and ICE plots provide valuable insights into both the individual and average impacts of features on outcomes.

Future considerations involve extending the analysis to female tennis data, though information availability on female players might be limited. Additionally, other types of tour-

naments beyond the four Grand Slams could be included, such as world cups and smaller tournaments. Another avenue for exploration is to conduct 1,000 simulation runs of new tournaments to estimate the probability of a player winning a tournament. Moreover, one could explore more advanced machine learning models, such as deep learning techniques, which could be a promising direction for future research (Bishop, 1995; LeCun et al., 2015). Finally, further statistically enhanced covariates could be developed. For instance, similar to the historical match abilities created for soccer teams by Ley et al. (2019), analogous abilities could be formulated for tennis players.

References

Almarashi, A. et al. (2024). A novel comparative study of nnar approach with li...

Angelini, E., Di Tollo, G., and Roli, A. (2008). A neural network approach for credit risk evaluation. *The quarterly review of economics and finance*, 48(4):733–755.

Apley, D. W. and Zhu, J. (2020). Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 82(4):1059–1086.

Arcagni, A., Candila, V., and Grassi, R. (2022). A new model for predicting the winner in tennis based on the eigenvector centrality. *Annals of Operations Research*, pages 1–18.

Auret, L. and Aldrich, C. (2012). Interpretation of nonlinear relationships between process variables by use of random forests. *Minerals Engineering*, 35:27–42.

Bartz, E., Bartz-Beielstein, T., Zaefferer, M., and Mersmann, O. (2023). *Hyperparameter tuning for machine and deep learning with R: A practical guide*. Springer Nature.

Beck, M. W. (2018). Neuralnettools: visualization and analysis tools for neural networks. *Journal of statistical software*, 85:1–20.

Bischl, B., Binder, M., Lang, M., Pielok, T., Richter, J., Coors, S., Thomas, J., Ullmann, T., Becker, M., Boulesteix, A.-L., et al. (2023). Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(2):e1484.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees. wadsworth int. *Group*, 37(15):237–251.

Buhamra, N. and Groll, A. (2025). Statistical enhanced learning for modeling and prediction tennis matches at grand slam tournaments. *arXiv preprint arXiv:2502.01613*.

Buhamra, N., Groll, A., and Brunner, S. (2024). Modeling and prediction of tennis matches at grand slam tournaments. *Journal of Sports Analytics*. 10(1), 17-33.

Buhamra, N., Groll, A., and Gerharz, A. (2025). Comparing modern machine learning approaches and different forecasting strategies for modeling tennis matches at grand slam tournaments. *Journal of Sports Analytics*. to appear.

Bühlmann, P. and Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting.

Bühlmann, P. and Yu, B. (2003). Boosting with the l 2 loss: regression and classification. *Journal of the American Statistical Association*, 98(462):324–339.

Bunker, R., Yeung, C., and Fujii, K. (2024a). Machine learning for soccer match result prediction. *arXiv preprint arXiv:2403.07669*.

Bunker, R., Yeung, C., Susnjak, T., Espie, C., and Fujii, K. (2024b). A comparative evaluation of elo ratings-and machine learning-based methods for tennis match result prediction. *Proceedings of the Institution of Mechanical Engineers, Part P: Journal of Sports Engineering and Technology*, 238(4):305–316.

Candila, V. and Palazzo, L. (2020). Neural networks and betting strategies for tennis. *Risks*, 8(3):68.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Chen, T., He, T., Benesty, M., and Khotilovich, V. (2019). Package 'xgboost'. *R version*, 90:1–66.

Clarke, S. R. and Dyte, D. (2000). Using official ratings to simulate major tennis tournaments. *International transactions in operational research*, 7(6):585–594.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*.

Contributors, X. (2024). Xgboost documentation. `https://xgboost.readthedocs.io/en/stable/parameter.html`. Accessed: 2024-11-08.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20:273–297.

Csáji, B. C. et al. (2001). Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24(48):7.

De Seranno, A. (2020). Predicting tennis matches using machine learning. *Master of Science in Computer Science Engineering*.

Del Corral, J. and Prieto-Rodríguez, J. (2010). Are differences in ranks good predictors for grand slam tennis matches? *International Journal of Forecasting*, 26(3):551–563.

Díaz-Uriarte, R. and Alvarez de Andrés, S. (2006). Gene selection and classification of microarray data using random forest. *BMC bioinformatics*, 7:1–13.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., Weingessel, A., and Leisch, M. F. (2009). Package 'e1071'. *R Software package, avaliable at http://cran. rproject. org/web/packages/e1071/index. html*.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Dressel, J. and Farid, H. (2018). The accuracy, fairness, and limits of predicting recidivism. *Science advances*, 4(1):eaao5580.

Easton, S. and Uylangco, K. (2010). Forecasting outcomes in tennis matches using within-match betting markets. *International Journal of Forecasting*, 26(3):564–575.

Eilers and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, 11:89–121.

Eilers and Marx, B. D. (2021). *Practical smoothing: The joys of P-splines*. Cambridge University Press.

Fahrmeir, L., Kneib, T., Lang, S., and Marx, B. (2013). *Regression. Modells, Methods and Applications*. Springer, Berlin.

Fahrmeir, L. and Tutz, G. (2001). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer-Verlag, New York, 2nd edition.

Felice, F., Ley, C., Groll, A., and Bordas, S. (2023). Statistically enhanced learning: a feature engineering framework to boost (any) learning algorithms. *arXiv preprint arXiv:2306.17006*.

Fernando, T., Denman, S., Sridharan, S., and Fookes, C. (2019). Memory augmented deep generative models for forecasting the next shot location in tennis. *IEEE Transactions on Knowledge and Data Engineering*, 32(9):1785–1797.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer.

Friedman (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Friedman, Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407.

Friedman, Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Gao, Z. and Kowalczyk, A. (2021). Random forest model identifies serve strength as a key predictor of tennis match outcome. *Journal of Sports Analytics*, 7(4):255–262.

Goldstein, A., Kapelner, A., Bleich, J., and Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *journal of Computational and Graphical Statistics*, 24(1):44–65.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep learning*, volume 1. MIT press Cambridge.

Greenwell, B. M., Boehmke, B. C., and McCarthy, A. J. (2018). A simple and effective model-based variable importance measure. *arXiv preprint arXiv:1805.04755*.

Groll, A., Hvattum, L. M., Ley, C., Popp, F., Schauberger, G., Van Eetvelde, H., and Zeileis, A. (2021). Hybrid machine learning forecasts for the uefa euro 2020. *arXiv preprint arXiv:2106.05799*.

Groll, A., Ley, C., Schauberger, G., and Van Eetvelde, H. (2019). A hybrid random forest to predict soccer matches in international tournaments. *Journal of quantitative analysis in sports*, 15(4):271–287.

Gu, W. and Saaty, T. L. (2019). Predicting the outcome of a tennis tournament: Based on both data and judgments. *Journal of Systems Science and Systems Engineering*, 28(3):317–343.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46:389–422.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). Statistical learning with sparsity. *Monographs on statistics and applied probability*, 143(143):8.

Hecht-Nielsen, R. (1992). Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. Elsevier.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2):251–257.

Hovad, E., Hougaard-Jensen, T., and Clemmensen, L. K. H. (2024). Classification of tennis actions using deep learning. *arXiv preprint arXiv:2402.02545*.

Hu, C., Tan, Q., Zhang, Q., Li, Y., Wang, F., Zou, X., and Peng, Z. (2022). Application of interpretable machine learning for early prediction of prognosis in acute kidney injury. *Computational and Structural Biotechnology Journal*, 20:2861–2870.

Hughes, M., Hughes, M. T., and Behan, H. (2007). The evolution of computerised notational analysis through the example of racket sports. *International Journal of Sports Science and Engineering*, 1(1):3–28.

James, G., Witten, D., Hastie, T., Tibshirani, R., et al. (2013). *An introduction to statistical learning*, volume 112. Springer.

Klaassen, F. J. and Magnus, J. R. (2003). Forecasting the winner of a tennis match. *European Journal of Operational Research*, 148(2):257–267.

Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., and Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. *Computational and structural biotechnology journal*, 13:8–17.

Kuhn, M. (2008). Building predictive models in r using the caret package. *Journal of statistical software*, 28:1–26.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

LeCun, Y., Bottou, L., Orr, G., and Müller, K. (2012). Efficient backprop, lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), lectu (1998) 7700: 9–48. *DOI: https://doi.org/10.1007/978-3-642-35289-8-3*.

Ley, C., Wiele, T. V. d., and Eetvelde, H. V. (2019). Ranking soccer teams on the basis of their current strength: A comparison of maximum likelihood approaches. *Statistical Modelling*, 19(1):55–73.

Lv, X., Gu, D., Liu, X., Dong, J., and Li, Y. (2024). Momentum prediction models of tennis match based on catboost regression and random forest algorithms. *Scientific Reports*, 14(1):18834.

Ma, S. C., Ma, S. M., Wu, J. H., and Rotherham, I. D. (2013). Host residents' perception changes on major sport events. *European Sport Management Quarterly*, 13(5):511–536.

Markopoulou, C. (2024). Sport analytics algorithms for football performance prediction.

Mayr, A., Binder, H., Gefeller, O., and Schmid, M. (2014). The evolution of boosting algorithms. *Methods of information in medicine*, 53(06):419–427.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133.

McHale, I. and Morton, A. (2011). A bradley-terry type model for forecasting tennis match results. *International Journal of Forecasting*, 27(2):619–630.

Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.

Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society*, A 135:370–384.

Orr, G. B. and Müller, K.-R. (1998). *Neural networks: tricks of the trade*. Springer.

Oshiro, T. M., Perez, P. S., and Baranauskas, J. A. (2012). How many trees in a random forest? In *Machine Learning and Data Mining in Pattern Recognition: 8th International Conference, MLDM 2012, Berlin, Germany, July 13-20, 2012. Proceedings 8*, pages 154–168. Springer.

Osuna, E., Freund, R., and Girosit, F. (1997). Training support vector machines: an application to face detection. In *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pages 130–136. IEEE.

Ouyang, Y., Cheng, M., He, B., Zhang, F., Ouyang, W., Zhao, J., and Qu, Y. (2023). Interpretable machine learning models for predicting in-hospital death in patients in the intensive care unit with cerebral infarction. *Computer Methods and Programs in Biomedicine*, 231:107431.

Pan, H., Luan, Q., and Cong, F. (2024). An integrated tennis momentum exploration model based on logistic regression and xgboost. In *Proceedings of the 2024 International Conference on Smart City and Information System*, pages 535–540.

Park, M. Y. and Hastie, T. (2007). L_1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society Series B*, 19:659–677.

Payrovnaziri, S. N., Chen, Z., Rengifo-Moreno, P., Miller, T., Bian, J., Chen, J. H., Liu, X., and He, Z. (2020). Explainable artificial intelligence models using real-world electronic health record data: a systematic scoping review. *Journal of the American Medical Informatics Association*, 27(7):1173–1185.

Probst, P., Wright, M. N., and Boulesteix, A.-L. (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: data mining and knowledge discovery*, 9(3):e1301.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ramosaj, B. (2020). Analyzing consistency and statistical inference in random forest models.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Ripley, B. D. (2002). *Modern applied statistics with S*. springer.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Sampaio, T., Oliveira, J. P., Marinho, D. A., Neiva, H. P., and Morais, J. E. (2024). Applications of machine learning to optimize tennis performance: a systematic review. *Applied Sciences*, 14(13):5517.

Savareh, B. A., Bashiri, A., Behmanesh, A., Meftahi, G. H., and Hatef, B. (2018). Performance comparison of machine learning techniques in sleep scoring based on wavelet features and neighboring component analysis. *PeerJ*, 6:e5247.

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5:197–227.

Schmid, M. and Hothorn, T. (2008). Boosting additive models using component-wise p-splines. *Computational Statistics & Data Analysis*, 53(2):298–311.

Scornet, E. (2017). Tuning parameters in random forests. *ESAIM: Proceedings and surveys*, 60:144–162.

Scornet, E., Biau, G., and Vert, J.-P. (2015). Consistency of random forests.

Sipko, M. and Knottenbelt, W. (2015). Machine learning for the prediction of professional tennis matches. *MEng computing-final year project, Imperial College London*, 2.

Somboonphokkaphan, A., Phimoltares, S., and Lursinsap, C. (2009). Tennis winner prediction based on time-series history with neural modeling. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, pages 18–20. Citeseer.

Sun, X., Wang, Y., and Khan, J. (2023). Hybrid lstm and gan model for action recognition and prediction of lawn tennis sport activities. *Soft Computing*, 27(23):18093–18112.

Svensson, W. (2021). Can statistical models beat benchmark predictions based on rankings in tennis?

Tay, J. K., Narasimhan, B., and Hastie, T. (2023). Elastic net regularization paths for all generalized linear models. *Journal of statistical software*, 106.

Therneau, T., Atkinson, B., Ripley, B., and Ripley, M. B. (2015). Package 'rpart'. *Available online: cran. ma. ic. ac. uk/web/packages/rpart/rpart. pdf (accessed on 20 April 2016)*.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society*, B 58:267–288.

Ünal, T. (2023). Predicting tennis match outcome: a machine learning approach using the srp-crisp-dm framework. Master's thesis, Middle East Technical University.

Ustun, B. and Rudin, C. (2019). Learning optimized risk scores. *Journal of Machine Learning Research*, 20(150):1–75.

Wei, X., Lucey, P., Morgan, S., Carr, P., Reid, M., and Sridharan, S. (2015). Predicting serves in tennis using style priors. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2207–2215.

Wei, X., Lucey, P., Morgan, S., and Sridharan, S. (2013). Predicting shot locations in tennis using spatiotemporal data. In *2013 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–8. IEEE.

Weston, D. (2014). Using age statistics to gain a tennis betting edge. `http://www.pinnacle.com/en/betting-articles/Tennis/atp-players-tipping-point/LMPJF7BY7BKR2EY`.

Whiteside, D., Cant, O., Connolly, M., and Reid, M. (2017). Monitoring hitting load in tennis using inertial sensors and machine learning. *International journal of sports physiology and performance*, 12(9):1212–1217.

Wilkens, S. (2021). Sports prediction and betting models in the machine learning age: The case of tennis. *Journal of Sports Analytics*, 7(2):99–117.

Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, London, 2nd edition.

Wright, M. N. and Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17.

Yigit, A. T., Samak, B., and Kaya, T. (2020). An xgboost-lasso ensemble modeling approach to football player value assessment. *Journal of Intelligent & Fuzzy Systems*, 39(5):6303–6314.

Young, N. (1988). *An introduction to Hilbert space*. Cambridge university press.

Yue, J. C., Chou, E. P., Hsieh, M.-H., and Hsiao, L.-C. (2022). A study of forecasting tennis matches via the glicko model. *Plos one*, 17(4):e0266838.

Zhang, J., Zhu, Q., Xu, L., and Gong, H. (2024). Research on tennis match evaluation and prediction model based on data mining. In *Fourth International Conference on Applied Mathematics, Modelling, and Intelligent Computing (CAMMIC 2024)*, volume 13219, pages 1169–1177. SPIE.

Part II

Publications

Journal of Sports Analytics 10 (2024) 17–33
DOI 10.3233/JSA-240670
IOS Press

Modeling and prediction of tennis matches at Grand Slam tournaments

N. Buhamra, A. Groll* and S. Brunner
Department of Statistics, TU Dortmund University, Dortmund, Germany

Received 7 September 2022
Accepted 11 December 2023
Published 29 February 2024

Abstract. In this manuscript, different approaches for modeling and prediction of tennis matches in Grand Slam tournaments are proposed. The data used here contain information on 5,013 matches in men's Grand Slam tournaments from the years 2011–2022. All regarded approaches are based on regression models, modeling the probability of the first-named player winning. Several potential covariates are considered including the players' age, the ATP ranking and points, odds, elo rating as well as two additional age variables, which take into account that the optimal age of a tennis player is between 28 and 32 years. We compare the different regression model approaches with respect to three performance measures, namely classification rate, predictive Bernoulli likelihood, and Brier score in a 43-fold cross-validation-type approach for the matches of the years 2011 to 2021. The top five optimal models with highest average ranks are then selected. In order to predict and compare the results of the tournaments in 2022 with the actual results, a comparison over a continuously updating data set via a "rolling window" strategy is used. Also, again the previously mentioned performance measures are calculated. Additionally, we examine whether the assumption of non-linear effects or additional court- and player-specific abilities is reasonable.

Keywords: Grand Slam tournaments, Tennis matches, prediction, model selection, cross validation, penalization

1. Introduction

In recent years, several approaches to the statistical modeling of tennis matches and tournaments have been proposed and the existing methods for predicting the probability of winning matches in tennis have been expanded. Then, when all matches can be predicted, also winning probabilities for a whole tournament could potentially be calculated. For instance, Clarke and Dyte (2000) used the official Association of Tennis Professionals (ATP) computer tennis rankings to predict a player's chance of winning via logistic regression. Arcagni et al. (2022) extended the approach of rating calculations to determine the probability that a player will win a match. The usage of this centrality measure allows the ratings of the whole set of players to vary every time there is a new match, and the resulting ratings are then used as a covariate in a simple logit model. Klaassen and Magnus (2003) used a large (live) data set from the Wimbledon predictions during the event. Hence, their work is suitable for the betting market. Easton and Uylangco (2010) used Klaassen and Magnus' model and compared it with bookmakers' odds on a point-by-point basis. They verified that bookmakers' odds are a good predictor of outcomes of both men's and women's tennis matches. Gu and Saaty (2019) predicted the outcome of tennis matches of Grand slam tournaments as well as of the ATP and the Women's Tennis Association (WTA) using both data and (unqualified, subjective) judgments, and this way identified numerous factors and systematically prioritized them subjectively and objectively, so as to improve the accuracy of the prediction. In McHale and Morton (2011), a Bradley-Terry type model was proposed for forecasting the top tier of the WTA and ATP competition. They considered surface (hardcourt, carpet, clay or grass) influence on match outcomes. They

*Corresponding author: A. Groll, Department of Statistics, TU Dortmund University, Vogelpothsweg 87, 44221 Dortmund, Germany. E-mail: groll@statistik.tu-dortmund.de.

found that a model incorporating information on match score, play data and surface can give a higher accuracy of forecasting than ranking-based models. However, they did not consider the dependence between factors. Ma et al. (2013) applied another logistic regression model on 16 variables representing player skills and performance, player characteristics and match characteristics. Yue et al. (2022) proposed a statistical approach for predicting the match outcomes of Grand Slam tournaments, using exploratory data analysis. The proposed approach introduces new variables via the Glicko rating model, a Bayesian method commonly used in professional chess.

Recently, machine learning models have been utilized to predict the winner of tennis matches. Somboonphokkaphan et al. (2009) proposed a method to predict the winner of tennis matches using both match statistics and environmental data based on a Multi-Layer Perceptron (MLP) equipped with a back-propagation learning algorithm. MLP is a basic sort of Artificial Neural Networks (ANN). ANNs are a powerful technique to solve real world classification problems and are particularly effective for predicting outcomes when the networks rely on a large database and are able to deal with incomplete information or noisy data. In addition, there are several studies for predictions based on machine learning approaches (see, for example, Whiteside et al., 2017). Also Wilkens (2021) focused on machine learning approaches and extended previous research by conducting and applying a wide range of machine learning techniques. He used a variety of models such as neural networks and random forests in combination with one of the most extensive data sets in the area of professional men's and women's tennis singles matches. Moreover, the author showed that the average prediction accuracy cannot be increased to more than about 70%. Bayram et al. (2021) defined a new method based on network analysis to extract a new feature that represents the player's skill on each surface considering the variation of his performance over time, which is believed to have a big effect on the match outcome. In addition, advanced machine learning paradigms such as Multi-Output Regression and Learning using privileged information have been applied, and the results were compared with standard machine learning approaches, such as regression tree- and forest-based methods as well as single- and multi-target regression techniques. Evaluating the results showed that the proposed methods provide more accurate predictions of tennis match outcomes than classical approaches frequently used in the literature. Chitnis and Vaidya (2014) considered performance assessments of professional tennis players using Data Envelopment Analysis in historical matches played in ATP world tour rankings. Radicchi (2011) novel evidence of the utility of tools and methods of network theory in real application. Del Corral and Prieto-Rodrıguez (2010) estimated separate probit models for men and women using Grand Slam tennis match data from 2005 to 2008. The explanatory variables are divided into three groups: a player's past performance, a player's physical characteristics, and match characteristics. The accuracies of the different models were evaluated both in-sample and out-of-sample by computing Brier scores and comparing the predicted probabilities with the actual outcomes from 2005 to 2008 and from the 2009 Australian Open. In addition, they used bootstrapping techniques, and evaluated the out-of-sample Brier scores for the 2005–2008 data.

Statistical and machine learning techniques have also been applied in other racket sports. For example, Lennartz et al. (2021) focused on international table tennis and analyzed matches of recent holdings of the Men's World Cup and the Grand Finals of the Men's ITTF World Tour. Also, they applied statistical and machine learning methods on table tennis tournaments for prediction with a correct classification rate of around 75% by a random forest and 74% by a penalized generalized linear logit model. Even though both models based their predictive power mainly on the official table tennis rankings and points, variables like age, playing handedness or individual strength turned out to be important additional factors.

In the present work, we concentrate on several regression-based modeling approaches with a focus on tennis Grand Slam tournament data. While complex machine learning models often have the capacity to further increase the predictive performance, they also come with the substantial draw-back of loosing interpretability. Hence, in this work we want to fully exploit the flexibility of modern regression approaches, using their high level of interpretability to gain some knowledge to understand certain associations and relations in professional tennis. For this purpose, a data set was compiled using the R package `deuce` (Kovalchik, 2018). It contained information on 5,013 matches at men's Grand Slam tournaments from 2011 to 2022. Several potential covariates are considered including the players' age, the ATP ranking and points, odds, Elo rating as well as two additional age variables, which take into account that the "optimal" age of a tennis player is between 28 and 32 years (Weston, 2014). We present different

regression approaches which are then compared with respect to various performance measures. Two specific aspects that we will investigate in more detail are whether it makes sense to (i) allow for non-linear covariate effects or to (ii) incorporate additional court-specific abilities.

The rest of the article is structured as follows. Section 2 introduces the present data set, explains the variables and defines the objectives. Then, in Section 3, different modeling approaches are introduced, including logistic regression, regularization using the Least Absolute Shrinkage and Selection Operator (LASSO) and non-parametric spline regression. In Section 4, these modeling approaches are compared via 43-fold cross-validation (CV) for the Grand Slam tournaments from 2011 to 2021. For this comparison, various performance measure are defined. Then, the performance measures are calculated on the Grand Slam tournament 2022 using a rolling window approach. We discuss the obtained results in Section 5. Section 6 summarizes the main results and gives a final overview.

2. Data

In the following, both the used data set and the variables it contains are described in more detail. Subsequently, the objective of this work is specified.

The underlying data set was compiled using the R package `deuce` (Kovalchik, 2018). It contains information on 5,013 matches in men's Grand Slam tournaments from 2011 to 2022. The variables included in the data set are listed and described below. Unless stated otherwise, the variables were directly included in the data sets of the package.

Player1: The name of the (randomly chosen) first-named player.

Player2: The name of the (randomly chosen) second-named player.

Year: The year when the match took place (ranging from 2011 to 2022).

Tournament: The Grand Slam tournament where the players met (Australian Open, French Open, Wimbledon, US Open).

Surface: A factor variable describing the surface on which the match was played (either "hard", "clay" or "grass").

Victory: A dummy variable capturing whether the first-named player did win the match (1: yes, 0: no).

Age: A metric predictor collecting the age difference of the players in years; age of the 2nd player was subtracted from the age of the 1st player. Note that players' ages were not given directly and had to be calculated from the player's date of birth as well as the date of the relevant match.

Prob: Difference in the probabilities that the respective player will win. These were calculated from the average odds for both players (see AvgProb1 and AvgProb2 below). The probability of the 2nd player winning was subtracted from the probability of the 1st player winning. This variable is later used for modeling.

Rank: Difference in the players' ranking positions. These were calculated by subtracting the rank of the 2nd player from the rank of the 1st player. For this, the rank of the player at the start of the tournament was used. The position in the ranking is based on the ATP ranking points.

Points: Difference in the ATP ranking points. The points of the 2nd player were subtracted from the points of the 1st player. World ranking points are awarded for each match won per tournament. Wins in later rounds of a tournament are valued higher than wins in the first rounds of a tournament. Points earned in a tournament expire after 52 weeks.

Elo: Difference of the Elo-numbers. The Elo-number of the 2nd player was subtracted from the Elo-number of the 1st player. The Elo-number takes into account whether a player played against a higher or lower ranked player. The Elo-number increases more if a player wins against a player with a high Elo-number than if he wins against a player with a lower Elo-number. It is updated after each match of a player.

Age.30: To calculate this variable, first the distance between the age of the players and reference age 30 was calculated and then the corresponding difference was calculated as for the variable *Age*. It is assumed that the standard *Age* variable introduced above does not contain enough information. For example, a 25-year-old player typically has an advantage over a 20-year-old one, while a 40-year-old player typically has a disadvantage over a 35-year-old one. However, in

both cases, the age difference is 5 years. As Weston (2014) argued, the optimal age of tennis players is between 28 and 32 years. Therefore, the middle of the interval, i.e. 30, was used as reference age here.

Age.int: For this feature, the distance to the closer limit of the interval was used, i.e. for players younger than 28 the distance to 28 was calculated and for players older than 32 the distance to 32 was calculated. For players between 28 and 32 the distance was set to 0. Then, the difference was calculated as for the variable *Age*.

AvgProb1: Average probability for a win by *Player1*, calculated from the average odds of several different betting providers, which were included in the `deuce` package as obtained from https://www.oddsportal.com/.

AvgProb2: Average probability for a win by *Player2*, calculated from the average odds of the betting providers. Together with *AvgProb1*, it sums up to 1 per match.

B365.1 Winning odds for *Player1* obtained from the specific bookmaker Bet365. For example, with winning odds of 2.31 for *Player1*, one gets back 2.31 money units if he wins, if previously one money unit was placed on this event. The odds from this specific bookmaker are later used to calculate the betting returns.

B365.2 Same as *B365.1*, but from the perspective of *Player2*.

It should be noted that the data set does not include matches in which one of the two players retired or was unable to compete, e.g. due to injury, such that the other player won without actually playing the match. These matches do not contain any information and could distort the results and are therefore excluded. Furthermore, the data set does not contain any missing values.

In addition, it should also be noted that there are some players which only participated in a single Grand Slam tournament. For instance, Camilo Ugo Carabelli participated only at the French Open 2022 and did not participate in any of the Grand Slam tournaments from 2011 to 2021. Also, as another example, Jan Satral participated for the very first time at the US Open 2016 and never again after this. Altogether, there are 70 players which participated in only one single Grand Slam tournament. Therefore, in our leave-one-tournament-out strategy for comparing the predictive power of the different modeling approaches, for these players no estimates of their abilities are available, if their matches are part of the tournament which is currently used as test data. In order to obtain nonetheless reasonable estimates for the player ability effects of such players, which can then be used for the prediction of the currently left-out Grand Slam tournaments, we group all players that have only participated in a single tournament in a group called "newcomer". Hence, these players share the same player-specific ability parameters.

Based on this data set, the best possible regression model for predicting tennis matches at Grand Slam tournaments is sought. We investigate which model approaches are particularly suitable for this purpose. Among other things, it will be examined whether the assumption of non-linear influences or additional surface- and player-specific abilities is reasonable. For the different modeling approaches, we then determine models which are optimal with respect to certain performance measures, and compare those with each other.

3. Statistical methods

In the following, the statistical methods used in this work are briefly introduced. These include logistic regression and parameter estimation using maximum likelihood. Based on this, we shortly motivate regularization and define the so-called LASSO-estimator. Finally, spline regression with P-splines is described.

3.1. The logistic regression model

For n individuals, let observations $(y_i, x_{i1}, \ldots, x_{ip})$, $i = 1, \ldots, n$ of a binary target variable y and covariates $x_1, \ldots, x_p$ be given. In the logistic regression model, the relationship between y and metric, categorical or binary covariates is examined. Here, $y = 1$ denotes the occurrence of a particular event (typically defined as "success") and y = 0 that the event does not occur (also defined as "failure"). Then,

$$\pi_i = P(y_i = 1|x_{i1}, \ldots, x_{ip}) = E(y|x_{i1}, \ldots, x_{ip})$$

is the (conditional) probability for the occurrence of $y_i = 1$, given the covariate values $x_{i1}, \ldots, x_{ip}$. The aim is to model π_i appropriately as a function of the feature variables. Therefore, the linear predictor η_i is

related to the probability π_i by a strictly monotonically increasing function $h\colon \mathbb{R} \to [0, 1]$, i.e.

$$\pi_i = h(\eta_i) = h(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}) .$$

The function $h(\cdot)$ is also called the *response function*. With the help of the inverse function $g = h^{-1}$, we can also write $\eta_i = g(\pi_i)$.

The estimators for $\beta_0, \ldots, \beta_p$ are obtained by numerical maximization of the log-likelihood, e.g. by using the Fisher scoring or the Newton-Raphson method, see, e.g., Nelder and Wedderburn (1972). Generally, for more details on GLMs, see also Fahrmeir and Tutz (2001).

3.2. Regularization

If the number of covariates p becomes very large, estimation becomes numerically unstable (see, e.g., Fahrmeir et al., 2013). This can also be the case if there is some substantial mulitcollinearity between the columns of the design matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\top$. To address this problem, a penalty term pen($\boldsymbol{\beta}$) is added to the negative log-likelihood in the logit model. According to Park and Hastie (2007), the estimator is then obtained by minimizing

$$\hat{\boldsymbol{\beta}}_{pen} = \arg\min_{\boldsymbol{\beta}}(-l(\boldsymbol{\beta}) + \lambda \cdot pen(\boldsymbol{\beta})),$$

where λ is the *penalty parameter* that controls the influence of the penalty term on the parameters estimated by the ML method.

The Least Absolute Shrinkage and Selection Operator (LASSO)

One possibility for penalization is provided by the *Least Absolute Shrinkage and Selection Operator* (LASSO; Tibshirani, 1996). Here the penalty term is given by

$$pen(\boldsymbol{\beta}) = \sum_{j=1}^{p} |\beta_j|.$$

It allows model estimation and variable selection to be performed in one step, as small coefficients are shrunk to 0. There is no closed-form representation for solving this minimization problem. Therefore, numerical optimization methods are used to obtain the optimal LASSO estimator $\hat{\boldsymbol{\beta}}_{\text{LASSO}}$ (see, e.g., Friedman et al., 2010). To optimize the penalty parameter λ typically K-fold cross validation can be is used.

3.3. Splines

In the methods introduced above, the influence of the covariates on the target variable is assumed to be strictly linear. However, often also non-linear influences are worthwhile. In order to model these appropriately and flexibly, so-called *splines* can be used. Here, the so-called *B-splines* (Eilers and Marx, 1996) are used.

B-splines

In principle, with B-splines a non-linear effect $f(x)$ of a metric predictor can be represented as

$$f(x) = \sum_{j=1}^{d} \gamma_j B_j(x).$$

As an unpenalized estimation of a non-linear B-spline effect often overfits, typically the non-linear effect is smoothed by using *penalized B-splines*, i.e. *P-splines*.

P-splines

Beside the problem of potential overfitting, the goodness-of-fit of the B-spline approach depends on the number of selected nodes. To avoid this problem, various penalization methods exist in the form of P-splines. Here, a penalized estimation criterion, which is extended by a penalty term, is used instead of the usual estimation criterion. For P-splines based on B-splines (see, e.g., Eilers and Marx, 1996), the function $f(x)$ is first approximated by a polynomial spline with many nodes (typically about 20 to 40). The penalty term then results in

$$\lambda \int \left(f''(x)\right)^2 dx .$$

This is motivated by the fact that the second derivative is used as a measure of the curvature of a function. If this becomes too large, it will be penalized by the term above. For approximation of the second derivative exist simple representations, so that the penalty term results in

$$\lambda \sum_{j=3}^{d} \left(\Delta^2 \gamma_j\right)^2 ,$$

where $\Delta^2 \gamma_j = \gamma_j - 2\gamma_{j-1} + \gamma_{j-2}$ (see again Eilers and Marx, 1996).

The optimal smoothing parameter λ is determined using generalized CV. For more details on the methodology, see also Eilers and Marx (2021),

and for more details on the corresponding software implementation in R, see Wood (2017).

4. Evaluation

In the following, some model approaches which seem to be suitable to adequately model tennis matches well, are investigated. These are compared with each other in a CV-type strategy in order to be able to select the best model with respect to a selection of performance measures. In an external validation with previously unused data, the best models from the preceding CV-type approach are evaluated. All calculations and evaluations are performed with the statistical programming software R (R Core Team, 2022).

4.1. Model selection

To model the outcome of a tennis match appropriately, different regression models with different assumptions can be used. Since the target variable y (win/loss) is a binary variable, the methods are all based on logistic regression. Therefore, the model can be generally formulated as

$$\ln\left(\frac{\pi_i}{1-\pi_i}\right) = \eta_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ip}\beta_p.$$

Since the $x_{i1}, \ldots, x_{ip},\ i = 1, \ldots, n$, are differences of the covariate values of the players, where the value of the second player is always subtracted from the value of the first player, β_0 would correspond to a kind of "home effect" for the first-named player. However, since the data are composed in such a way that one of the two players is named first randomly, β_0 cannot be meaningfully interpreted here and is therefore excluded and set to zero in the following considerations. Furthermore, it is assumed that the $y_i|x_{i1}, \ldots, x_{ip}$ are independent for $i = 1, \ldots, n$.

Linear effects

The simplest and most straight-forward model approach is to assume linear covariate effects in the linear predictor. Here, nevertheless, covariates must be selected appropriately. To ensure this, all possible combinations of the available variables are compared in the CV-type approach. Since there are seven covariates in the data set, there are $\sum_{i=1}^{7}\binom{7}{i} = 127$ different combinations of these. However, of these seven covariates, three reflect specific age differences. Therefore, it is additionally assumed that the combinations always include a maximum of one age variable. This results in 63 different combinations.

Non-linear effects (splines)

The assumption of linear effects can possibly be very limiting and lead to insufficient results. Therefore, spline models based on B-splines are also considered. Here, all 63 possible combinations are again compared with respect to various performance measures. To obtain smoothness, penalization of the splines is performed and, hence, P-splines are used. Furthermore, an additional regularization approach is used, i.e. an additional variable selection is performed on the spline effects. This is done by setting the spline coefficients to zero and is implemented in the `gam`-function from `mgcv` (Wood, 2017) via setting the `select` argument to `TRUE`. For the underlying methodology of this additional penalization and variable selection approach, see e.g. Marra and Wood (2011). We will later on see that exemplarily the effect of the variable probs is non-linear for very low and high values (see Fig. 1), which also seems to be reasonable (see our explanation in Section 5).

Surface-specific player skills (LASSO)

Since tennis is played on different surfaces (grass, hard court and clay court) and these surfaces have specific characteristics, so that each surface is played differently, it is plausible to assume that not every player copes equally well on every surface. For example, the Spaniard Rafael Nadal is considered as the "king of clay", as he has won 14 French Open titles on this surface. At the same time, however, he was "only" able to win the Wimbledon title twice, which is played on grass. In contrast, the Swiss Roger Federer has already won Wimbledon eight times, but the French Open only once. In order to take into account these specific features of players and surfaces, a corresponding factor variable is created. Technically, this requires effect coding. Actually, to the data set artificially columns are added, whose entries are either 0, 1 or –1. Each column represents a combination between a player and one of the corresponding surfaces, i.e. for each player there are a maximum of three such columns. If a player has not played on one of the three surface types, the corresponding columns are omitted. Each row consists of exactly one entry of –1 and 1, respectively, while all other entries of the row are 0. The entry 1 is in the column of the combination of the first player and the corresponding surface on which the match took place. Analogously, the entry

–1 is in the column of the combination of the second player and the surface. For example, the line

⋯	*Nadal.Hard*	*Nadal.Grass*	*Nadal.Clay*	⋯	*Federer.Hard*	*Federer.Grass*	*Federer.Clay*	⋯
⋯	0	1	0	⋯	0	−1	0	⋯

means that Rafael Nadal played against Roger Federer on grass, Nadal was named as the first player and Federer as the second. The remaining entries in the row are 0, because the match was played on grass and only these two players were involved. All these columns are appended to the existing data set. The newly constructed variables as well as the already existing variables, i.e. *Age, Ranking, Points, Elo, Prob, Age.30* and *Age.int*, are then jointly used as covariates.

As a result, a large number of new covariates is constructed, namely 1,024, which generally leads to an extremely large number of parameters being estimated and the associated estimators becoming unstable. Therefore, for this approach the logistic regression model is combined with LASSO regularization. The influences of all covariates is assumed to be linear here. Via the surface-specific player skill parameters, the model can detect when a player has won or lost more often than average on a surface.

Global player skills (LASSO)

Analogously, it can be argued that there are also players who overall perform even better or worse than the information of their covariate values would suggest, i.e. who have a generally great or substandard talent and are therefore more likely to be assessed as winners or losers. For this purpose, similar to the previous paragraph, a corresponding factor variable is created, again using effect coding. In this case, we add only one column per player. The added columns again only have the entries 0, 1 or –1, where the values 1 and –1 occur exactly once per row. In each row, the value 1 appears at the position belonging to the first-named player and the value –1 at the position belonging to the second-named player, all remaining entries are 0. The following exemplary row

⋯	*Novak.Djokovic*	⋯	*Rafael.Nadal*	⋯	*Roger.Federer*	⋯
⋯	0	⋯	−1	⋯	1	⋯

indicates that Rafael Nadal played as the second named player against the first named player Roger Federer. The column for the player Novak Djokovic (as well as for all other players), for example, is then 0, since he did not play. Again, those global player-specific abilities are then added to the design matrix and used along with the covariates defined in Section 2. The columns are then appended to the already existing record.

Again, a large number of new covariates is constructed, namely 426, so again a large number of player-specific skill parameters has to be estimated. And, hence, again we extend the logistic regression model with LASSO penalization and assume linear covariate effects only.

Benchmark model

As a benchmark model for prediction, we solely use the probabilities (probs) calculated from the average odds of the different bookmakers included in the `deuce` package. The winning probabilities in the i-th match $\hat{\pi}_{i1}$ and $\hat{\pi}_{i2}$ for player 1 and player 2, respectively, can be derived from the two winning odds, i.e. odd_{i1} and odd_{i2}, respectively, according to Schauberger and Groll (2018) as follows:

$$\hat{\pi}_{i1} = \frac{\frac{1}{odd_{i1}}}{\frac{1}{odd_{i1}} + \frac{1}{odd_{i2}}} \quad \text{or} \quad \hat{\pi}_{i2} = \frac{\frac{1}{odd_{i2}}}{\frac{1}{odd_{i1}} + \frac{1}{odd_{i2}}}$$

Note that naturally those probs fullfill $\hat{\pi}_{i1} + \hat{\pi}_{i2} = 1$. Moreover, this way, it is automatically adjusted for the bookmaker's margin. These margins result from the fact that the betting providers artificially lower their betting odds to gain some profit. This means that when the inverse values of the odds are directly used as probabilities, they do not sum up to 1, but to a value slightly larger than 1. In order to calculate the margin, the sum of the reciprocals in the denominator is used for normalization. Here, it is implicitly assumed that the margin is equally distributed to both players.

4.2. Performance measures

To compare the selection of regression models in terms of their predictive performance on new, unseen data, the following criteria are considered. First, use $\tilde{y}_1, \ldots, \tilde{y}_n$ for the true binary outcomes of the n matches, i.e., $\tilde{y}_i \in \{0, 1\}, i = 1, \ldots, n$. Further, let $\hat{\pi}_{i1} =: \hat{\pi}_i$ denote the probability, predicted by a certain model, that player 1 wins the i-th match. The

probability that player 2 wins the match is given by $\hat{\pi}_{i2} = 1 - \hat{\pi}_{i1} = 1 - \hat{\pi}_i$, since y is binary.

Classification rate

The (mean) *classification rate* indicates how many matches on average are correctly predicted by a certain model. It is defined by

$$\frac{1}{n}\sum_{i=1}^{n} 1(\tilde{y}_i = \hat{y}_i), \text{ where } \quad \hat{y}_i = \begin{cases} 1, & \hat{\pi}_i > 0.5 \\ 0, & \hat{\pi}_i \leq 0.5 \end{cases},$$

see, e.g., Schauberger and Groll (2018). Here, large values indicate a good model. A mean classification rate of 0.5 would correspond to a random classification. It is therefore also desirable that the classification rate is much larger than 0.5.

Predictive Bernoulli likelihood

The *predictive Bernoulli likelihood* is based on the predicted probability for the true outcome and for n predictions is defined as

$$\hat{\pi}_i^{\tilde{y}_i}(1 - \hat{\pi}_i)^{1-\tilde{y}_i},$$

see again Schauberger and Groll (2018). Once again, a high value is an indicator of a good model. In the following, the average likelihood is used for model comparison.

Brier score

The *Brier score* is based on the squared distances between the predicted probability and the actual (binary) output from the i-th match. It is defined according to Brier (1950) as

$$\frac{1}{n}\sum_{i=1}^{n}(\hat{\pi}_i - \tilde{y}_i)^2.$$

This is an error measure, so low values indicate a good model.

Betting profit

Another way to compare the predictive quality of different models is the betting profit. Let odd_{i1} and odd_{i2} be the odds from a specific betting provider for a win of the first and second player in the i-th match, respectively. If one bets one monetary unit on a win of the respective player, the expected betting returns for the i-th match are given by

$$E[return_{i,\,player1}] = \hat{\pi}_{i1} \cdot odd_{i1} - 1 \quad \text{or}$$

$$E[return_{i,\,player2}] = \hat{\pi}_{i2} \cdot odd_{i2} - 1,$$

because, for instance, if player 1 wins the match (which due to the model at hand happens with predicted probability $\hat{\pi}_{i1}$), the better, who has previously invested one money unit (hence the –1), would receive odd_{i1} money units, if they has bet on this event (see Schauberger and Groll, 2018). Hence, if the player on whom the bet was placed wins, the betting return is calculated by the player's odds minus the invest of one monetary unit. If the other player wins, the better's loss is -1 monetary unit.

Principally, the bet should of course be placed on the match outcome with maximum positive expected return. If the return is not positive for either outcome, no bet is placed on the corresponding match. In this work, to calculate actual, realistic betting returns, the odds of the specific betting provider *Bet365* are used, i.e. it is assumed that the bets are placed with this provider. The total betting return is then the sum of all betting returns across all matches.

4.3. Leave-one-tournament-out cross validation

To evaluate the models, all Grand Slam tournaments from 2011 to 2021 are used, i.e. the tournaments that took place in 2022 are initially not considered and are used as external validation data later on. So, from each of the 11 years four tournaments are used. A 43-fold CV-type strategy is performed with these tournaments. The data set then still contains 4,720 of the original 5,013 matches. The following scheme is used:

1. From the 43 Grand Slam tournaments present in the data set, a training data set of 42 tournaments and a test data set of the remaining tournament are constructed.
2. Then, all regression models introduced above are fitted:
   - For the models with linear influences, the function `glm` from the R package `stats` (R Core Team, 2022) is used. As described in Section 4.1, there are 63 such models, each using at most one of the three variables for age.
   - For the calculation of the spline models the function `gam` from the R package `mgcv` (Wood, 2004) is used. Again, there are also 63 different models here.
   - In order to be able to compute the two LASSO-penalized models, first the design matrices have to be constructed as described in Section 4.1. For a proper usage of the LASSO, then all columns of the design matrix of the train-

ing data need to be standardized. The function `cv.glmnet` from the R package `glmnet` (Friedman et al., 2010) is used to compute both models. For this, a 10-fold (inner) CV is first performed on the current training data to find the optimal λ which provides the minimum deviance. The corresponding LASSO model is used afterwards for prediction. 10-fold CV is also recommended in Friedman et al. (2010).

- For prediction based on average betting odds, the probabilities calculated from odds are used as predicted probabilities.

3. After fitting the respective model, for each match of the test data the probabilities that the first player wins are predicted.
4. Steps 1–3 are repeated until each of the 43 tournaments has served once as a test data set.
5. Finally, the predicted results are compared with the actual results and the performance measures defined in Section 4.2 are calculated.

Table 1 on page 9 shows the results of the five best models with linear effects and the five best models with spline effects. Additionally, the results of the LASSO models and the benchmark model are shown. The top five models were selected from the 63 models by assigning ranks for each of the three performance measures, classification rate, predictive Bernoulli likelihood, and Brier score, with the best model receiving the highest rank. Average ranks were assigned if the models had equal performances. In order to select the best models in terms of all three measures, their ranks were also averaged per model and the five models with the highest average ranks were selected. The models are listed below in such a way that the best model is determined in first place, the second best in second place and so on. The overall betting performance is also given together with the per match betting return in brackets. Finally, the number of bets placed is provided in the last column, together with the ratio of all matches in which a bet was placed in brackets.

The classification rate is slightly above 77% for all models (see 1st column). Differences between the models can only be seen at the third decimal. The linear regression model with both rank and probs as covariates here performs best with a value of 0.7776, i.e. this model predicts the correct outcome for 77.76% of the matches.

The values of the predictive Bernoulli likelihood (2nd column) differ somewhat more. It is noticeable that within one group of model approaches the likelihood is almost the same (differences are only seen in the fourth decimal). The average likelihood of the models with both linear and non-linear effects is slightly more than 0.69, i.e. the models predict the correct outcome with an average probability of about 69%. The two LASSO models are just below 0.69. It is noticeable that the benchmark model has the lowest likelihood, which is 0.6709. The average betting provider correctly predicts the outcome of a match with an average probability of 67.09%.

The differences across model groups in the Brier scores are rather small, similar to the classification rate. The models with non-linear effects perform best in this regard with Brier scores between 0.1546 and 0.1547. The models with linear effects produce slightly larger values, followed by the LASSO models. In the case of the Brier score, the benchmark model again performs worst with a value of 0.1555.

Table 1

Results of the cross validation for the (at most) best five models per model class (best performing model in bold font). In brackets are the betting profits per match and the ratio of all matches in which a bet was placed

	Explanatory variables	Class. rate	Likeli-hood	Brier score	Betting profit	Amount of bets
Linear	Prob	0.7772	**0.6919**	0.1549	−128.8 (−0.05)	2696 (57.1%)
	Points, Prob	0.7772	0.6917	0.1549	−99.9 (−0.04)	2395 (50.7%)
	Rank, Prob	**0.7776**	0.6918	0.1550	−110.9 (−0.04)	2707 (57.3%)
	Age, Prob	0.7766	0.6918	0.1550	−150.3 (−0.05)	2716 (57.5%)
	Rank, Points, Prob	0.7772	0.6917	0.1550	−104.3 (−0.04)	2418 (51.2%)
Splines	Prob	0.7764	0.6912	**0.1546**	**113.7 (0.09)**	1170 (24.8%)
	Rank, Prob	0.7764	0.6912	**0.1546**	**113.7 (0.09)**	1170 (24.8%)
	Points, Prob	0.7764	0.6912	0.1547	101.3 (0.08)	1298 (27.5%)
	Rank, Points, Prob	0.7764	0.6912	0.1547	101.8 (0.08)	1299 (27.5%)
	Elo, Prob	0.7764	0.6911	0.1547	106.9 (0.09)	1175 (24.9%)
LASSO	Surface specific	0.7774	0.6816	0.1551	−225.3 (−0.12)	1891 (40.1%)
	General skills	0.7770	0.6838	0.1551	−199.3 (−0.10)	2002 (42.4%)
	Benchmark	0.7772	0.6709	0.1555	−311.4 (−0.23)	1359 (28.8%)

The betting returns are negative for all models except for the spline-based approaches, and here there are substantial differences between the various modeling approaches. If linear effects are assumed, the losses range from about 100 to 150 monetary units. For the spline models, gains are achieved which lie between about 101 and 114 monetary units. In particular, the model in which either only prob or *prob* and rank were included perform best with a gain of about 114 monetary units. The two models with LASSO perform even worse than the linear models, with a betting loss of almost 200 and 225 money units. Once again, the benchmark model performs worst with a loss of about 311 monetary units. This comparatively high loss is due to the fact that this model almost always bets on the underdog, but the underdog rarely wins. The betting profit must be seen in relation to the number of bets placed. Here, the tendency can be seen that models with a high betting loss have the tendency to bet more often. It is particularly noticeable that bets are placed much less frequently when using the non-linear models compared to the other models. It can therefore be assumed that these models are a little more conservative. This can also be seen from the betting returns per match (bet). In some cases, the corresponding losses are significantly lower than those of the other model approaches.

4.4. External validation

To validate the models from above with respect to their predictive performance on new, unseen test data, the three best models from the groups of modeling approaches with linear and non-linear effects are used, as well as the two LASSO models and the benchmark model. For this purpose, the performance measures are calculated on the four Grand Slam tournaments 2022. The data set then contains 293 matches. The validation is performed using a "rolling window"-type approach, i.e. one of the remaining tournaments is used as the test data set in chronological order. The training data set then continues to be constantly updated and enlarged, this scheme can be explained as follows:

1. First, all tournaments prior to 2022 are used as the training data set and then the models are fitted in a way as it is described in the second step of the CV approach described in Section 4.3. With those, then predictions can be obtained for the 2022 Australian Open, as this is the first Grand Slam tournament of the year 2022.
2. The new training data set will then contain all tournaments up to and including the Australian Open 2022, on which the models are fitted again and predictions are made for the French Open 2022, the 2nd Grand Slam tournament of the year 2022.
3. Now the French Open 2022 is added to the training data set and the models are fitted again. This will then be used to predict Wimbledon 2022.
4. Then, the Wimbledon 2022 matches will be added to the training data set, and again, the models are fitted and predictions are made for the final Grand Slam tournament, the US Open 2022.
5. Finally, the predicted results for all four tournaments are compared with the actual results and the performance measures are calculated.

The results of the external validation are shown in Table 2. Once again, in the last two columns additionally the average betting returns per match and the proportion of bets placed are given in brackets.

The classification rate is again above 77% for all models, and for some models even above 78% and, hence, slightly better compared to the classification rates from the CV-type strategy from above. The best value is 78.16% and is achieved by three models. It is striking here that now the benchmark model is among the best.

Similar trends as in Table 1 (cf. page 9) can be seen for the likelihood, although the results here are somewhat better. The linear models achieve the highest likelihood, where the model including the covariates *Rank* and *Prob* delivers the best value with 0.7037, only slightly behind are the models with non-linear effects (all between 0.7031 and 0.7033). The LASSO models yield an even slightly smaller likelihood. The benchmark model again performs worst.

The Brier score for all models is around 0.141–0.142. In Section 4.3, the values were slightly larger. The benchmark model again yields the largest, and thus worst value with 0.1441. The best value is achieved by the non-linear model including covariates *Points* and *Prob* (0.1411).

Looking at the results of betting returns, the larger part of the results are similar to those of Table 1. The LASSO models and the linear models yields the largest betting loss: approximately between 16 and 24 monetary units each, respectively. The models with non-linear effects again mostly yield profits (about four monetary units). However, here the benchmark model yields the largest loss with 39 monetary units.

Also in terms of the number of bets placed, the results from Table 1 are mostly confirmed. The lin-

Table 2
Results of the external validation for the (at most) best three models per model class from Section 4.3 (best performing model in bold font). In brackets are the betting profits per match and the ratio of all matches in which a bet was placed

	Explanatory variables	Class. rate	Likeli-hood	Brier score	Betting profit	Amount of bets
Linear	Prob	**0.7816**	0.7036	0.1419	−21.6 (−0.11)	197 (67.2%)
	Points, Prob	0.7747	0.7027	0.1421	−17.4 (−0.09)	191 (65.2%)
	Rank, Prob	**0.7816**	**0.7037**	0.1418	−20.5 (−0.01)	198 (67.6%)
Splines	Prob	0.7782	0.7031	0.1413	**4.2 (0.03)**	121 (41.3%)
	Rank, Prob	0.7782	0.7031	0.1413	**4.2 (0.03)**	121 (41.3%)
	Points, Prob	0.7782	0.7033	**0.1411**	−0.28 (−0.00)	126 (43.0%)
LASSO	Surface specific	0.7782	0.6934	0.1423	−24.4 (−0.15)	161 (54.9%)
	General skills	0.7782	0.6969	0.1416	−16.1 (−0.09)	171 (58.4%)
	Benchmark	**0.7816**	0.6810	0.1441	−39 (−1.00)	39 (13.3%)

ear models again bet most frequently (in almost every third match), the LASSO models bet second most frequently (in about half of the matches). If non-linear effects are included, betting is even less frequent (in about 40% of the cases), these models seem to be again the most conservative ones, but at a higher level, since the proportion of bets placed has increased for each of these three model approaches compared to the results from Section 4.3. The same applies to the betting returns per match, which is now 0.00 monetary units of loss in the worst case. It is noticeable that bets were placed much less frequently with the benchmark model than with all other models. This was not the case in Section 4.3. There, a bet was placed in 28.8% of the possible matches, while here the percentage is only 13.3%.

5. Discussion

In Section 4.3, a leave-one-tournament-out CV-type approach was performed with the 43 Grand Slam tournaments from the years 2011–2021. Since the tournaments actually were played one after the other in time, the CV-type strategy in a certain sense resulted in the setting that the past was predicted with information from the future. Initially, this argues against the normal intuition of prediction, since, for example, players with currently high rankings are also more likely to have had high rankings in the past tournament, i.e. the values are correlated. However, since the crucial assumption that the $y_i|x_{i1}, \ldots, x_{ip}$ are (conditionally on the covariate information) independent for $i = 1, \ldots, n$ is still realistic, CV was used here as a technical tool to compare performance across many different prediction models.

Instead of a CV-type approach, a performance comparison over a continuously updating data set ("rolling window") could be considered, as in Section 4.4. This would have the advantage that the temporal structure of the data could be preserved. However, the CV-type strategy here had the advantage that the models could be compared on more data. Thus, each tournament served once as a test data set and since this was only used for an initial comparison to find principally suitable models, CV was preferred to the rolling window approach.

For the models from Table 1 (page 9) and Table 2 (page 11) it is noticeable that the variable *Prob* is always selected. So the bookmaker information seems to be very important and to have a big influence on the prediction. This could also be a reason why the models all perform quite similarly. The number of ranking points or the rank itself are also partly selected in the three best linear and spline-based models. Hence, these variables also appear to be important to a certain extend, although not quite as influential as the odds.

It is hardly possible to filter out a clear winner amongst the regarded models, since they differ little with regard to the performance measures. If one initially compares only the three regression modeling approaches, tendencies can be identified. When considering the classification rate and predictive Bernoulli likelihood, the linear models might be very slightly preferred. Regarding the Brier score, the spline-based models perform slightly better.

However, since the two LASSO models never perform best with respect to any of the measures, and since also their respective betting loss is the largest, these models are rather not to be preferred. Within the first modeling approach, according to the results from Table 2 (page 11), the linear model with the ranking position and the betting odds is best suited to model a tennis match. Within the spline models, the model including only the betting odds and the number of

ranking points could be chosen as the winner with respect to the first three performance measures. But as all three models almost yield equal results, also the most sparse and simple model, here the first one just including the betting odds, could be chosen, as it also results in positive betting returns.

Between the two models selected in this way (*Rank* and *Prob* as linear effects or only *Prob* as a non-linear effect), the betting profit should also be considered. If this is taken into account, the spline model should be preferred; if the betting returns are not considered to be important, the linear model should be selected.

Spline models

Figure 1 shows the fitted spline of the variable *Prob* and its pointwise 95% confidence intervals. For this purpose, the model was fitted with the covariate *Prob* on all tournaments.

The estimated effect looks almost linear between -0.5 and 0.5, and then steeper at both edges. This suggests that betting providers use somewhat "unfair" or special odds in these regions. If the absolute difference in the winning probabilities is more than 0.5, it can be assumed that a strong player is playing against an extreme underdog, which occurs particularly often in the first round of a tournament. For example, the favorite in a match may have a 99% chance of winning the match according to the betting companies, while the underdog has a 1% chance of winning. This would result in odds of 1.01 for the favorite and 100 for the underdog (ignoring the bookmaker's margin for a moment). However, since an underdog's win is very unlikely, the bookmaker probably don't always withhold the same margin from an underdog's odds. For higher odds, they probably withdraw larger margins than for lower odds. Therefore, it seems reasonable to assume that the effect of *Prob* is not linear, but instead (slightly) non-linear.

LASSO models

In determining the optimal penalty strength λ_{opt} for the LASSO models, 10-fold CV was performed for a sequence of different λ values and the mean deviance was calculated. Figure 2 shows this process as an example with the 2011–2021 data for the model with general player skills.

Here, $\lambda_{opt} = 0.0161$ provided a minimum deviance of 0.9504. The LASSO model with this choice for λ_{opt} was then used to predict the Australian Open 2022. For the same setting, Fig. 3 on page 13 shows the corresponding coefficient paths as a function of λ, together with λ_{opt} as the vertical dashed line.

For any λ between about 0.02 and 0.3, every coefficient except that of the variable *Prob* is shrunk to zero. For decreasing λ, the coefficient of *Prob* becomes larger, again showing the importance of this variable. For λ smaller than 0.02, the coefficients of both variables *Points* and *Elo* are also positive. The coefficients estimated by the model can be read at the location of λ_{opt}. *Prob* has a coefficient estimate of 1.50, *Points* of 0.06 and *Elo* of 0.02. Among others,

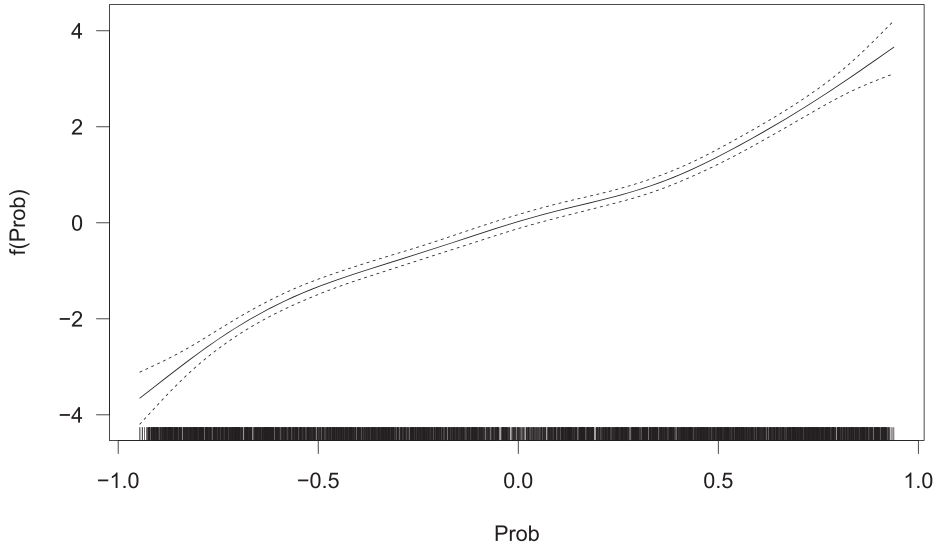

Fig. 1. Estimated non-linear effect of the variable *Prob*.

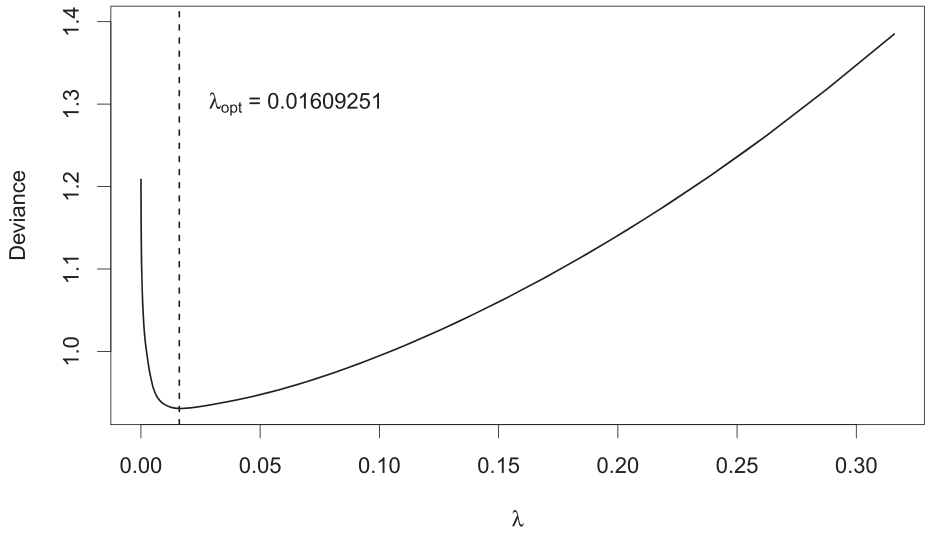

Fig. 2. CV-deviance of LASSO-model as a function of λ; vertical dashed line: λ_{opt}.

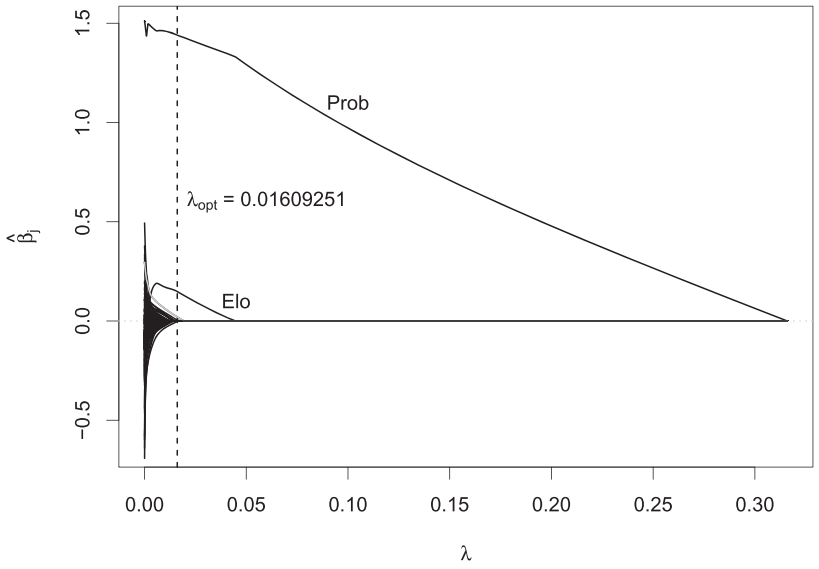

Fig. 3. Coefficient paths vs. penalty strength λ; vertical dashed line: λ_{opt}.

the coefficient estimate of the player Tennys Sandgren (plotted in gray), is also positive at this point with a value of 0.03.

This can be seen in more detail in Fig. 4, where it is zoomed into the range of the smaller λ values. At the optimal amount of penalization, the coefficients of most other players are 0, except for ten different players. Among those, Tennys Sandgren has the largest estimated regression coefficient with a value of 0.03, so if he is one of the two players competing in a match, the value of 0.03 is added to his linear predictor for modeling the probability of him winning the match, so he seems to perform a bit better than his covariate values indicate. If one chooses λ to be even smaller, the coefficient estimates of many other players are also no longer shrunken to zero, and they are given positive or negative abilities. As the LASSO estimator for smaller λ gets closer and closer to the maximum likelihood estimator, these coefficient estimates are numerically very unstable. This can also be seen in the path of the variable *Prob* (see Fig. 3), which shows a very wiggly behavior for λ close to 0.

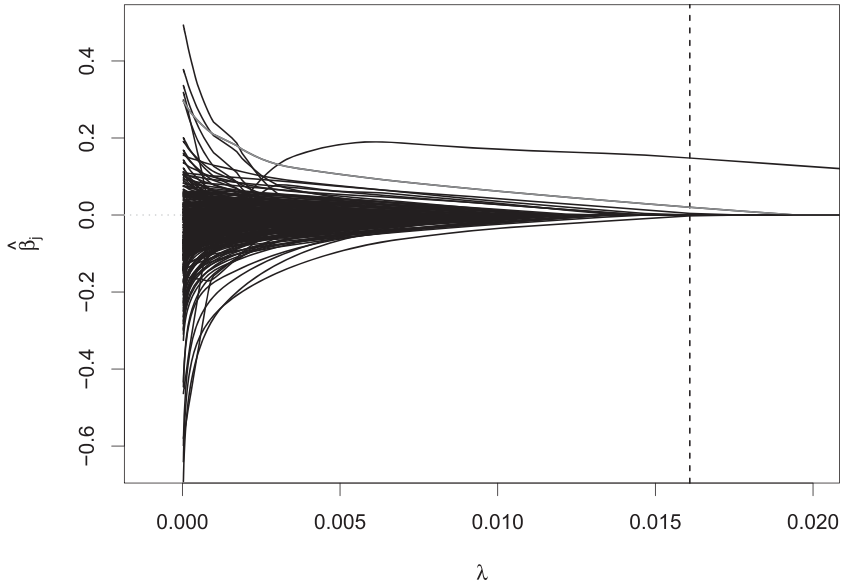

Fig. 4. Zoomed fragment of the coefficient paths for lower λ values.

6. Summary and overview

In this work, tennis matches in Grand Slam tournaments were modeled within the framework of regression. For this purpose, a data set that was compiled using the R package `deuce` (Kovalchik, 2018). This contained information on 5,013 matches in men's Grand Slam tournaments from the years 2011–2022. This included the age difference of both players (*Age*), the difference in their ranking positions (*Rank*) and ranking points (*Points*), in Elo numbers (*Elo*), in probabilities for the victory of the players, calculated from the average odds by the betting providers (*Prob*), as well as the two additional age variables *Age.30* and *Age.int*, which should take into account that the optimal age of a tennis player is between 28 and 32 years.

Different regression approaches were considered for modeling and prediction of tennis matches. As there are only two possible outcomes in tennis (win or loss), all models were based on a binary outcome and, hence, on logistic regression, modeling the probability of the first named player to win. It was discussed that modeling with intercept would not be useful as this would incorporate a kind of home effect for the first named player – a property which was not desired here.

The different modeling approaches were compared in a 43-fold leave-one-tournament-out CV-type strategy. Each of the 43 Grand Slam tournaments from 2011 to 2021 served once as a test data set. The following models were included:

- Models with linear effects: to find suitable covariates, all possible combinations of the seven covariates were considered such that at most one of the three age variables was incorporated. This resulted in 63 models.
- Models with non-linear effects (splines): Again, 63 models were considered.
- A model which took into account surface- and player-specific effects. Due to the large amount of unknown parameters, here LASSO penalization was used.
- A model that considered general player-specific abilities. Again, LASSO penalization was used.
- A benchmark model, where the predicted probabilities were derived from average betting odds.

Within the CV-type approach, the models were compared in terms of the classification rate, the predictive Bernoulli likelihood, the Brier score as well as betting returns. Since 63 different models resulted for each of the first two approaches, the five best models were selected in each case.

It was found that all models performed very similarly in terms of classification rate, likelihood and Brier score. The classification rate was slightly above 77% for all models, meaning that the models predicted the correct outcome in about 77% of the matches. The predictive Bernoulli likelihood was

about 0.69 for both linear and non-linear models, while the LASSO models and the benchmark model were slightly below. The models thus predicted with an average probability of about 69% the correct outcome of a match. The Brier score was slightly above 0.15 for all models, differences were mostly found in the third decimal. When comparing the betting profits and the amount of bets placed, it was found that only the spline models achieved positive betting returns, but also placed fewer bets. Therefore, it can be assumed that these models are more conservative (and thus probably safer) in terms of betting.

The tournaments in 2022 were then used as an external validation data set. The three best models with each linear and non-linear effects, the two LASSO models and the benchmark model were then compared again. The results of the preceding CV-type competition could be mostly confirmed, with the values of the performance measures generally being slightly better than for the CV: the classification rate was around $0.775 - 0.782$, the predictive likelihood yielded around $0.693 - 0.704$ and the Brier score was between $0.141 - 0.142$. With regard to the betting returns, again only the spline models achieved a betting profit, except for the model with the number of ranking points and the betting odds led to loss. The proportion of placed bets increased for all models, only the benchmark model placed considerably fewer bets than in the preceding CV-type competition. The most striking here was that, again the benchmark model was no better than the other models for most performance measures.

In a more detailed discussion, it was pointed out that the CV-type strategy here was preferable to a "rolling window approach" in finding models, since on the one hand the assumption of independence of the observations of the target variable, given the covariates, is fulfilled. Secondly, this allowed the models to be compared on more data, as the initial aim was to find suitable models. Furthermore, it was emphasized that the betting odds are very important for the prediction. In addition, it could be worked out that within the linear models the covariates *Rank* and *Prob* provided the best results. Within the spline approach, all three models provided almost equal results. Due to simplicity, the model that only considered betting odds would be preferred here. The LASSO models tended to perform worse than the other models and therefore could not really be recommended. The obtained betting returns can be used to decide between the first two types of approaches. They turned out to be positive for the best spline model, while the linear model yielded a loss. However, the classification rate and predictive likelihood were better for the latter model. It was also notable that all models performed at least as well as the benchmark model.

Lastly, the spline model with the *Prob* variable and the model with general player-specific skills were examined in more detail. Based on the corresponding fitted smooth effect, the behavior of the bookmakers in setting odds for an extreme underdog were discussed. Using the LASSO model for general player-specific skills, the CV for finding an optimal λ via deviance minimization was illustrated. In addition, the corresponding coefficient paths for the different covariates were shown and explained.

In future research, an upcoming complete tournament could also be repeatedly simulated, and then the probability of a certain player to win the tournament could be determined. This can take advantage of the fact that the tournament course is completely drawn before the start, i.e. it can already be said on the basis of the tournament tree that two players can meet at earliest in a certain round. This means that is not necessary to take into account whether someone has finished first or second in a certain group stage, as it is the case in soccer, for example. With such an approach, however, only the match-specific betting odds for the first round would be available. Models that do not use the odds as covariates could then be preferable, but this could lead to a poorer prediction performance due to the large influence of this variable. Alternatively, one could look at models that do not use the odds for individual matches, but instead use odds set before the tournament on each player to win the whole tournament.

Moreover, another extension of the approach proposed here would be to allow for more flexible, time-varying player-specific ability parameters, similar e.g. to the approach proposed by Ley et al. (2019) for modeling soccer. This approach has also been used successfully and the resulting estimates have been incorporated as a so-called "hybrid" feature in a random forest model for predicting the FIFA World Cup 2018 in Groll et al. (2019), and hence, seems to be also promising in tennis. And the authors have already planned to carry over this idea to tennis. To do so, a different (and much larger) data set has to be collected and also a separate, quite complex model specifically designed for historic match data has to be built up.

Finally, as already stated, in this work only (directly interpretable) approaches within the frame-

work of regression were analyzed. In future research, we have planned to compare their performance with different complex machine learning models, which often have the capapbility to further increase the predictive performance, though coming with the substantial draw-back of loosing interpretability. For that reason, they typically should be equipped with certain methods of *interpretable machine learning* (IML), such as partial dependence plots Friedman, 2001, ICE plots Goldstein et al., 2015 and ALE plots Apley and Zhu, 2020. A first attempt to summarize the limited selection of available methods for interpretable ML appears in Molnar (2020).

References

Apley, D.W. & Zhu, J., 2020, Visualizing the effects of predictor variables in black box supervised learning models, *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *82*(4), 1059-1086.

Arcagni, A., Candila, V. & Grassi, R., 2022, A new model for predicting the winner in tennis based on the eigenvector centrality, *Annals of Operations Research*, pages 1-18.

Bayram, F., Garbarino, D. & Barla, A., 2021, Predicting tennis match outcomes with network analysis and machine learning, In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 505-518. Springer.

Brier, G.W., 1950, Verification of forecasts expressed in terms of probability, *Monthly Weather Review*, *78*, 1-3.

Chitnis, A. & Vaidya, O., 2014, Performance assessment of tennis players: Application of dea, *Procedia-Social and Behavioral Sciences*, *133*, 74-83.

Clarke, S.R. & Dyte, D., 2000, Using official ratings to simulate major tennis tournaments, *International Transactions in Operational Research*, *7*(6), 585-594.

Del Corral, J. & Prieto-Rodríguez, J., 2010, Are differences in ranks good predictors for grand slam tennis matches? *International Journal of Forecasting*, *26*(3), 551-563.

Easton, S. & Uylangco, K., 2010, Forecasting outcomes in tennis matches using within-match betting markets, *International Journal of Forecasting*, *26*(3), 564-575.

Eilers, P.H. & Marx, B.D., 2021, *Practical smoothing: The joys of P-splines*, Cambridge University Press.

Eilers, P.H.C. & Marx, B.D., 1996, Flexible smoothing with B-splines and penalties, *Statistical Science*, *11*, 89-121.

Fahrmeir, L., Kneib, T., Lang, S. & Marx, B., 2013, *Regression, Modells, Methods and Applications*, Springer, Berlin.

Fahrmeir, L. & Tutz, G., 2001, *Multivariate Statistical Modelling Based on Generalized Linear Models*, Springer-Verlag, New York, 2nd edition.

Friedman, J., Hastie, T. & Tibshirani, R., 2010, Regularization paths for generalized linear models via coordinate descent, *Journal of Statistical Software*, *33*(1), 1-22.

Friedman, J.H., 2001, Greedy function approximation: a gradient boosting machine, *Annals of Statistics*, *29*, 337-407.

Goldstein, A., Kapelner, A., Bleich, J. & Pitkin, E., 2015, Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation, *Journal of Computational and Graphical Statistics*, *24*(1), 44-65.

Groll, A., Ley, C., Schauberger, G. & Van Eetvelde, H., 2019, A hybrid random forest to predict soccer matches in international tournaments, *Journal of Quantitative Analysis in Sports*, *15*, 271-287.

Gu, W. & Saaty, T.L., 2019, Predicting the outcome of a tennis tournament: Based on both data and judgments, *Journal of Systems Science and Systems Engineering*, *28*(3), 317-343.

Klaassen, F.J. & Magnus, J.R., 2003, Forecasting the winner of a tennis match, *European Journal of Operational Research*, *148*(2), 257-267.

Kovalchik, S., 2018, deuce: resources for analysis of professional tennis data, *R package version*, 1.

Lennartz, J., Groll, A. & van der Wurp, H., 2021, Predicting table tennis tournaments: A comparison of statistical modelling techniques, *International Journal of Racket Sports Science*, *3*(2).

Ley, C., Wiele, T.V.d. & Eetvelde, H.V., 2019, Ranking soccer teams on the basis of their current strength: A comparison of maximum likelihood approaches, *Statistical Modelling*, *19*(1), 55-73.

Ma, S.C., Ma, S.M., Wu, J.H. & Rotherham, I.D., 2013, Host residents' perception changes on major sport events, *European Sport Management Quarterly*, *13*(5), 511-536.

Marra, G. & Wood, S.N., 2011, Practical variable selection for generalized additive models, *Computational Statistics & Data Analysis*, *55*(7), 2372-2387.

McHale, I. & Morton, A., 2011, A bradley-terry type model for forecasting tennis match results, *International Journal of Forecasting*, *27*(2), 619-630.

Molnar, C., 2020, *Interpretable machine learning*, Lulu. com.

Nelder, J.A. & Wedderburn, R.W.M., 1972, Generalized linear models, *Journal of the Royal Statistical Society, A*, *135*, 370-384.

Park, M.Y. & Hastie, T., 2007, L_1-regularization path algorithm for generalized linear models, *Journal of the Royal Statistical Society Series B*, *19*, 659-677.

R Core Team, 2022, *R: A Language and Environment for Statistical Computing,* R Foundation for Statistical Computing, Vienna, Austria.

Radicchi, F., 2011, Who is the best player ever? a complex network analysis of the history of professional tennis, *PloS One*, *6*(2), e17249.

Schauberger, G. & Groll, A., 2018, Predicting matches in international football tournaments with random forests, *Statistical Modelling*, *18*(5-6), 460-482.

Somboonphokkaphan, A., Phimoltares, S. & Lursinsap, C., 2009, Tennis winner prediction based on time-series history with neural modeling, In *Proceedings of the International Multi Conference of Engineers and Computer Scientists*, volume 1, pages 18-20. Citeseer,

Tibshirani, R., 1996, Regression shrinkage and selection via the Lasso, *Journal of the Royal Statistical Society, B*, *58*, 267-288.

Weston, D., 2014, Using age statistics to gain a tennis betting edge, http://www.pinnacle.com/en/betting-articles/Tennis/atp-players-tipping-point/LMPJF7BY7BKR2EY

Whiteside, D., Cant, O., Connolly, M. & Reid, M., 2017, Monitoring hitting load in tennis using inertial sensors and machine learning, *International Journal of Sports Physiology and Performance*, *12*(9), 1212-1217.

Wilkens, S., 2021, Sports prediction and betting models in the machine learning age: The case of tennis, *Journal of Sports Analytics*, *7*(2), 99-117.

Wood, S.N., 2004, Stable and efficient multiple smoothing parameter estimation for generalized additive models, *Journal of the American Statistical Association*, *99*(467), 673-686.

Wood, S.N., 2017, *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, London, 2nd edition,

Yue, J.C., Chou, E.P., Hsieh, M.-H. & Hsiao, L.-C., 2022, A study of forecasting tennis matches via the glicko model, *PloS One*, *17*(4), e0266838.

Comparing modern machine learning approaches and different forecast strategies on Grand Slam tennis tournaments

N. Buhamra[1] A. Groll[2] A. Gerharz[3]

[1]Department of Statistics, TU Dortmund University, Vogelpothsweg 87, 44221 Dortmund, email: nourah.buhamra@tu-dortmund.de

[2]*Corresponding author*, Department of Statistics, TU Dortmund University, Vogelpothsweg 87, 44221 Dortmund, email: groll@statistik.tu-dortmund.de

[3]Department of Statistics, TU Dortmund University, Vogelpothsweg 87, 44221 Dortmund, email: gerharz@statistik.tu-dortmund.de

Abstract

This manuscript builds upon the approach presented in Buhamra et al. (2024). Different modern machine learning and regression approaches for modeling and prediction of tennis matches in Grand Slam tournaments are investigated. Our data includes information on 5,013 matches in men's Grand Slam tournaments from the period 2011-2022. The investigated methods focus on modeling the probability of the first-named player to win the respective match. Moreover, different features are considered including the players' age, the ATP ranking and points, bookmakers' odds, Elo rating as well as two additional age variables, which take into account the optimal age of a tennis player. We compare the different regression approaches regarded in Buhamra et al. (2024) to modern machine learning approaches with respect to various performance measures. Moreover, we also investigate different forecast strategies. First of all, a cross-validation-type strategy for all matches between 2011 and 2021. We also use an "expanding window" strategy by continuously updating the training data to analyze the predictive performance of the approaches on the tournaments from 2022. Finally, a "rolling window" strategy is used with only three years of tournaments as training data. We then select small subsets of best models with largest average ranks, and investigate those in more detail by the help of interpretable machine learning techniques.

Keywords: Grand Slam tournaments, Tennis matches, machine learning, prediction, model selection, cross validation, rolling window.

1 Introduction

Recently, various statistical and machine learning methods for modeling both tennis matches and whole tournaments have been introduced, leading to an expansion of existing techniques for predicting match-winning probabilities in tennis. Consequently, once all individual matches can be predicted, it may also be possible to calculate the winning probabilities for an entire tournament.

In recent years, machine learning (ML) models were employed to forecast the winners of tennis matches. Somboonphokkaphan et al. (2009) introduced an approach

that is able to predict match winners by utilizing both match statistics and environmental data, which uses a Multi-Layer Perceptron (MLP) in combination with a back-propagation learning algorithm. MLP is a fundamental type of Artificial Neural Network (ANN), which are an effective method for addressing real-world classification task and are especially useful for predicting outcomes when they have access to extensive databases and can handle incomplete or noisy data. Whiteside et al. (2017), developed an automated stroke classification system in order to quantify the hitting load in tennis using machine learning models such as a cubic kernel support vector machine. Wilkens (2021) builds on prior research by exploring various ML techniques, including neural networks and random forests, using one of the largest datasets for professional men's and women's tennis matches. He found that the average prediction accuracy cannot exceed approximately 70%.

In Sipko and Knottenbelt (2015), the authors predicted the winner of the tennis match based on the probability of serve points won which in turn gives the probability of a player winning the match. They define a method of extracting 22 features from raw historical data, including abstract features, such as player fatigue and injury. Then, by using the resulting data set, they developed and optimized models which were able to outperform Knottenbelt's Common-Opponent model based on machine learning algorithms such as artificial neural networking. The authors believe that machine learning can bring new innovation in tennis betting as the neural networking generates a 4.35% return on investment in the betting market, which is an improvement of about 75% over the current state-of- the-art stochastic models. Furthermore, Gao and Kowalczyk (2021) constructed a model that predicts tennis match outcomes with high accuracy over 80%, much larger than predictions using betting odds alone, and identified serve strength as a key predictor of match outcome. This was done by using a random forest classifier that highlights the importance and relevance of simple models in the age of deep learning. Moreover, a wide variety of features to capture information about physical, psychological, court-related, and match-related variables are included in a large data set of tennis matches, which is compiled and processed from ATP data from 2000 to 2016. Finally, an overview of modeling and predicting tennis matches at Grand Slam tournaments by different regression approaches has been presented in Buhamra et al. (2024).

In the present work, we focus on several ML modeling approaches, which are applied on tennis Grand Slam tournament match data. Additionally, we proceed further with the comparison of those approaches to the regression models investigated in our previous work (see Buhamra et al., 2024). To achieve this, we use the same data analyzed in Buhamra et al. (2024), which was created from the `deuce` R package (Kovalchik, 2019), and includes information on 5,013 matches from men's Grand Slam tournaments between 2011 and 2022. Several features are available, such as the players' *age*, *ATP ranking* and *points*, *odds*, *Elo rating*, and two other *age*-based variables that account for the "optimal" player's age, which was found to be in the interval of $[28; 32]$ years (Weston, 2014). Various machine learning and regression approaches will be introduced that are then compared via different performance measures. Moreover, as it is a generally known result that feature engineering often also is quite a crucial aspect for the modeler (and sometimes even more relevant than the choice among different types of statistical or machine learning models), we analyze the impact of different types of forecasting strategies: a cross-validation-type strategy for all matches between 2011 and 2021, an "expanding window" strategy by continuously extending the training data, and a "rolling window" strategy based on only three years of tournaments as training data.

The rest of the article is structured as follows. Section 2 briefly introduces the data set and defines the objectives of this work. Then, in Section 3, different machine learn-

ing modeling approaches are introduced, including classification trees, random forest, XGBoost, support vector machine (SVM) and artificial neural networks (ANNs). Besides, some classical regression modeling approaches and corresponding regularization concepts such as the Least Absolute Shrinkage and Selection Operator (LASSO) and non-parametric spline techniques are defined. More details on the regression models and a benchmark model are introduced in Section 4. Moreover, three performance measure are defined and two approaches from the field of interpretable machine learning (IML) are described, which are then used to interpret the best machine learning model. In Section 5, all modeling approaches are compared via the different forecasting strategies and the introduced performance measures, on the one hand on all Grand Slam tournaments from 2011 to 2021 via a leave-one-tournament-out CV-type strategy. On the other hand, we employ an expanding window approach on the Grand Slam tournament 2022, as well as a rolling window approach with only three years of tournaments as training data in order to have very recent prediction results. Finally, Section 6 summarizes the main results and concludes.

2 Data

In this section, we shortly introduce the data set from Buhamra et al. (2024), which was compiled based on the R package `deuce` (Kovalchik, 2019). It contains the following information for 5,013 matches in men's Grand Slam tournaments from 2011 to 2022:

Player1: name of (randomly chosen) first-named player.

Player2: name of (randomly chosen) second-named player.

Year: year when the match took place (ranging from 2011 to 2022).

Tournament: Grand Slam tournament where the respective match took place (Australian Open, French Open, Wimbledon, US Open).

Surface: factor variable describing the surface on which the match was played ("hard","clay" or "grass").

Victory: binary target, determining whether the first-named player won the match (1: yes, 0: no).

Age: metric predictor capturing the age difference of the players (in years); age of the 2nd player was subtracted from that of 1st player.

Prob: difference in winning probabilities of both players, obtained from their average odds (see *AvgProb1* and *AvgProb2* below); winning probability of 2nd player is subtracted from that of 1st player.

Rank: difference in players' rankings, calculated by subtracting the rank of 2nd player from that of 1st player. For this, the rank of the players at the start of the tournament is used. The position in the ranking is determined by the ATP points.

Points: difference in ATP points; points of 2nd player are subtracted from those of 1st player; points are awarded for each match won per tournament. Wins in later rounds of a tournament are valued higher than wins in the first rounds of a tournament. Points expire after 52 weeks. Again, the players' points at the start of the tournament are used.

Elo: difference in Elo rating; Elo rating of 2nd player is subtracted from that of 1st player; Elo rating takes into account whether a player played against a higher or lower ranked player. It increases more if a player wins against a player with a large Elo rating than if he wins against a player with a lower one; it is updated after each new match of a player.

Age.30: first, the distance between the age of the players and reference age of 30 years is calculated, then the corresponding difference between both competing players is calculated, as e.g. for *Age*. This is because the standard *Age* variable from above does not contain enough information. For example, a 25-year-old player typically has an advantage over a 20-year-old one, while a 40-year-old player typically has a disadvantage over a 35-year-old one. However, in both cases, the age difference is 5 years. Following Weston (2014), who argued that the optimal age of tennis players is within $[28;32]$ years, we use 30 years as the mid point of the interval as reference age.

Age.int: first, the distance to the closer limit of the interval was used, i.e. for players younger than 28 the distance to 28 was calculated, while for players older than 32 the distance to 32 was computed. For players with an age within $[28;32]$ years the distance was set to 0. Then, the difference between both players was calculated.

AvgProb1: average winning probability of Player 1, based on the average odds from several different betting providers, which are provided in the `deuce` package (obtained from `https://www.oddsportal.com/`).

AvgProb2: Same as *AvgProb1*, but from the perspective of Player 2. Note that per match we have *AvgProb1* + *AvgProb2*=1.

B365.1 Winning odds for Player 1 obtained from the specific bookmaker Bet365. Exemplarily, winning odds of 2.31 for Player 1 yield a return of 2.31 money units if he wins, if previously one money unit is placed on this event. These odds are later used to compute betting returns.

B365.2 Same as *B365.1*, but from the perspective of Player 2.

A detailed description of the variables included in the data set can be found in Section 2 of Buhamra et al. (2024).

It is important to note that we exclude matches in which one player forfeited or was unable to compete, such as due to injury, resulting in a win for the other player without an actual match being played. These matches lack informative data and could skew the results, and hence are omitted. Additionally, the dataset contains no missing values. For this dataset, we aim to identify the most effective machine learning model for predicting tennis matches at Grand Slam tournaments and explore which modeling approaches are especially suitable for this task.

Among other things, it will be examined whether the machine learning approaches are powerful with regard to prediction of new matches compared to the classical regression approaches used in Buhamra et al. (2024). Then, for the different machine learning modeling approaches, based on a leave-one-tournament-out CV strategy we determine models which are optimal with respect to certain performance measures, and compare those with each other. Additionally, both an expanding window approach with a starting window of eleven years and a rolling window approach with short window of three years period are used.

3 Statistical and machine learning methods

In the following, closely following Buhamra et al. (2024), in Section 3.1 classical statistical methods are introduced, beginning with logistic regression. Based on this, we shortly motivate the concept of regularization and define the so-called LASSO-estimator. Moreover, spline regression with P-splines is described. Then, in Section 3.2 several machine learning methods such as classification trees, random forest, XGBoost and support vector machine (SVM) are introduced.

3.1 Regression approaches

Next, we will first introduce the classic logistic regression model for binary outcomes and then sketch different extensions, which will be used in this work. All of those have been introduced in detail already in Buhamra et al. (2024), but for better readability of this manuscript we show them again.

Logistic regression

For n individuals, let observations $(y_i, x_{i1}, \ldots, x_{ip})$, $i = 1, \ldots, n$ of a binary target variable y and covariates $x_1, \ldots, x_p$ be given. In the logistic regression model, the relationship between y and metric, categorical or binary covariates is examined. Here, $y = 1$ denotes the occurrence of a particular event (typically defined as "success") and y = 0 that the event does not occur (also defined as "failure"). Then,

$$\pi_i = P(y_i = 1 | x_{i1}, \ldots, x_{ip}) = E[y_i | x_{i1}, \ldots, x_{ip}]$$

is the (conditional) probability for the occurrence of $y_i = 1$, given the covariate values $x_{i1}, \ldots, x_{ip}$. The aim is to model π_i appropriately as a function of the features.

Therefore, the linear predictor η_i is related to the probability π_i by a strictly monotonically increasing function $h\colon \mathbb{R} \to [0,1]$, i.e.

$$\pi_i = h(\eta_i) = h(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ip}) .$$

The function $h(\cdot)$ is also called the *response function*. With the help of the inverse function $g = h^{-1}$, the so-called link function, we can also write $\eta_i = g(\pi_i)$. The logistic regression model is the most famous candidate within the framework of Generalized Linear Models (GLMs).

The estimators for $\beta_0, \ldots, \beta_p$ are obtained by numerical maximization of the log-likelihood, e.g. by using the Fisher scoring or the Newton-Raphson method, see, e.g., Nelder and Wedderburn (1972). Generally, for more details on GLMs, see also Fahrmeir and Tutz (2001).

Regularization

If the number of covariates p becomes very large, the estimation becomes numerically unstable (see, e.g., Fahrmeir et al., 2013). This can also be the case if there is some substantial mulitcollinearity between the columns of the design matrix $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^\top$. To address this problem, a penalty term pen($\boldsymbol{\beta}$) is added to the model's (negative) log-likelihood $l(\cdot)$, which in our case corresponds to a Bernoulli logit model. According to Park and Hastie (2007), the estimator is then obtained by minimizing

$$\hat{\boldsymbol{\beta}}_{pen} = \arg\min_{\boldsymbol{\beta}} (-l(\boldsymbol{\beta}) + \lambda \cdot pen(\boldsymbol{\beta})),$$

where λ is the *penalty parameter* that controls the influence of the penalty term on the parameters estimated by the ML method.

The Least Absolute Shrinkage and Selection Operator (LASSO)

One possibility for penalization is provided by the *Least Absolute Shrinkage and Selection Operator* (LASSO; Tibshirani, 1996). The corresponding penalty term is given by

$$pen(\boldsymbol{\beta}) = \sum_{j=1}^{p} |\beta_j|.$$

It allows model estimation and variable selection to be performed in one step, as small coefficients are shrunk to 0. No closed-form representation for solving this minimization problem does exist. Therefore, numerical optimization methods are used to obtain the optimal LASSO estimator $\hat{\boldsymbol{\beta}}_{\text{LASSO}}$ (see, e.g.,Friedman et al., 2010). To optimize the penalty parameter λ typically K-fold cross validation is used.

Spline-based approaches

In the methods introduced above, the influence of the covariates on the target variable is assumed to be strictly linear. However, often also non-linear influences are relevant. In order to model these appropriately and flexibly, so-called *splines* can be used. Here, the so-called *B-splines* (Eilers and Marx, 1996, 2021) will be employed.

B-Splines

In principle, with B-splines a non-linear effect $f(x)$ of a metric predictor can be represented as

$$f(x) = \sum_{j=1}^{d} \gamma_j B_j(x)\,,$$

where $B_j(x)$ represent the different B-spline basis functions, d denotes the number of basis functions used, and γ_j the corresponding spline coefficients. As an unpenalized estimation of a non-linear B-spline effect often overfits, typically the non-linear effect is smoothed by using *penalized B-splines*, i.e. *P-splines*.

P-splines

Beside the problem of potential overfitting, the goodness-of-fit of the B-spline approach depends on the number of selected nodes. To avoid this problem, various penalization methods exist in the form of P-splines. Here, a penalized estimation criterion, which is extended by a penalty term, is used instead of the usual estimation criterion. For P-splines based on B-splines (see, e.g., Eilers and Marx, 1996), the function $f(x)$ is first approximated by a polynomial spline with many nodes (typically about 20 to 40).

A suitable penalty term guaranteeing smoothness then would be given by

$$\lambda \int \left(f''(x)\right)^2 dx\,.$$

This is motivated by the fact that the second derivative is used as a measure of the curvature of a function. If this becomes too large, it will be penalized by the term above. For approximation of the second derivative exist simple representations, so that the simpler penalty term

$$\lambda \sum_{j=3}^{d} \left(\Delta^2 \gamma_j\right)^2$$

could be used instead, where $\Delta^2\gamma_j = \gamma_j - 2\gamma_{j-1} + \gamma_{j-2}$ (see again Eilers and Marx, 1996).

The optimal smoothing parameter λ is determined using generalized CV. For more details on the methodology, see also Eilers and Marx (2021), and for more details on the corresponding software implementation in R, see Wood (2017).

3.2 Machine learning approaches

In the following, we'll introduce different machine learning approaches that are suitable for binary responses and will be used in this work for comparison against the regression approaches, which were introduced before in Section 3.1.

Classification trees

Decision trees are a supervised machine learning algorithm. A tree starts with its root node and then follows binary splits based on the features' values until a leaf node is reached and the final binary result is given. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the given features. The splitting of nodes (including the root node) to sub-nodes happens based on impurity metrics. Hence, decision trees seek to find the best split to subset the data, and they are typically trained through the Classification And Regression Tree (CART) algorithm. Metrics, such as Gini impurity and information gain can be used to evaluate the quality of the splits. CARTs were introduced by Breiman et al. (1984) and use Gini impurity in the process of splitting the data set into a decision tree. Gini impurity measures how often a randomly chosen attribute is misclassified and can be represented as follows

$$1 - \sum_{i=0}^{c} (p_i)^2 ,$$

where p_i is the probability of class i with total number of classes c. The values of the Gini index range from 0 to 1, a lower value being more ideal. It should be noted that in this manuscript we will focus on the classification trees rather than the regression ones, as in our case the target is a binary variable (i.e. the target variable is categorical).

However, a fully grown tree typically will overfit the training data and the resulting model will not generalize well for predicting the outcome of new (unseen) test data. Therefore, techniques such as pruning (stopping the tree to grow) are used to control this problem. In the `rpart` package Therneau et al. (2015), this is controlled by the complexity parameter (cp), which gives a threshold which an improvement has to pass that a split stays in the tree. A too small value of cp leads to overfitting, and a too large cp value will result in a too small tree and, hence, underfitting. Both cases decrease the predictive performance of the model. An optimal cp value can be estimated by testing different cp values and using cross-validation (CV) approaches to determine the corresponding prediction accuracy of the model[1]. The best cp is then defined as the one that maximizes the CV accuracy.

Random forest

Individual trees, as Breiman (1996b) pointed out, suffer from instability, which is inherent in the tree model and can not be fixed through changing the way a tree is built. A first solution that does not interfere in the tree construction process is an ensemble method called *bootstrapping and aggregating* (bagging; Breiman, 1996a), which in general improves the predictive performance compared to a single regression tree and is easy to implement.

Similar to bagging, a *random forest* also consists of multiple decision trees which are aggregated for a more accurate prediction (Breiman, 2001). The logic behind the random forest model is that multiple uncorrelated models (i.e., the individual decision

[1] We self-implemented this CV approach such that for all methods, where tuning was necessary, the same folds are used and, hence, better comparability is achieved.

trees) perform much better as a group than they do alone, as this reduces the variance and yields more accurate predictions than an individual model. Depending on the type of problem, the type of prediction will vary. For a regression task, the individual decision trees will be averaged, and for a classification task, a majority vote, (i.e. the most frequent categorical variable) will yield the predicted class.

Again, in this manuscript, we will consider the classification task for the random forest since our target variable is binary. The main extension compared to bagging is the idea of drawing randomly only a subset of "`mtry`" features at each node for splitting in random forest, which involves selecting a random subset of predictors from the set of all available features at each decision node in a decision tree. This introduces diversity among the trees in the ensemble, reducing the risk of overfitting and enhances the model's generalization performance by ensuring that each tree is built on a different subset of features. For fitting random forest models, we use the `ranger` implementation in R (Wright and Ziegler, 2017).

Here, in the random forest, `mtry` and the `ntree` are the two parameters that most likely to have a big effect on our final accuracy and are defined as follows:

- `mtry`: number of variables randomly sampled as candidates at each split.
- `ntree`: number of trees to grow.

The `mtry` hyperparameter controls the split-variable randomization feature of the random forests and it helps to balance low tree correlation with reasonable predictive strength. For classification $mtry = \sqrt{p}$ is recommended, where p is the number of predictors. However, when there are fewer relevant predictors a larger value of `mtry` tends to perform better because it makes it more likely to select those features with the strongest signal. When there are many relevant predictors, a lower `mtry` might perform better. For this reason, we decided to properly tune this parameter using (self-implemented) 10-fold CV (see footnote 1 for details).

There are no specific criterias when it comes to choosing the number of trees that the random forest should use, as long as `ntree` is large enought. However, there is a threshold where adding more trees does not give any significant gain to the random forest (Oshiro et al., 2012), but computational costs would unnecessarily increase. Here, we decided to set `ntree`$= 400$ (and, hence, did not tune this parameter).

Extreme gradient boosting

Boosting originates in the machine learning community and proved to be a successful and practical strategy to improve classification procedures. Boosting is a sequential ensemble method that combines the predictions of multiple weak models to produce a stronger prediction, resulting in a final, aggregated model. I.e., first, a single model is built for the task and prediction errors are computed. Then, the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models are added.

Since Freund et al. (1996) have presented their famous Ada-Boost algorithm, many extensions have been proposed and boosting approaches were updated to estimate predictors for statistical models (e.g., gradient boosting by Friedman et al., 2000, generalized linear and additive regression based on the L2-loss by Bühlmann and Yu, 2003).

Hence, boosting is considered as a sequential process; i.e., if e.g. (simple) trees are used as learners, those are grown using the information from a previously grown tree one after the other. This process slowly learns from data and tries to improve its prediction in subsequent iterations. A main advantage of statistical boosting algorithms

is their flexibility for high-dimensional data and their ability to incorporate variable selection in fitting process (Mayr et al., 2014). An extensive and enlightening, general overview on gradient boosting algorithms can be found in Bühlmann and Hothorn (2007). Friedman (2001) introduced the idea of gradient tree boosting, using decision trees as learners. The decision trees are repeatedly fitted on the residuals of the previous fit and, hence, are combined to a sequential ensemble.

This technique was then further improved by Chen and Guestrin (2016) via introducing additional regularization in the objective function. The regularization terms make the single trees weak learners to avoid over fitting. In a certain boosting iteration, the next tree is additively incorporated into the ensemble after multiplication with a rather small learning rate which makes the learners even weaker. The method is called *extreme gradient boosting*, in short *XGBoost*, and is known in the machine learning community for its high predictive power. The approach is implemented in the `xgb.train` function from the `xgboost` R package (Chen et al., 2023). One important aspect is that XGBoost involves several tuning parameters. For this purpose, we specified reasonable, discrete grids and again used (self-implemented) 10-fold cross validation to determine optimal tuning parameters, see footnote 1 for further details.

Support vector machine (SVM)

Support vector machine (SVM) is another supervised machine learning technique, developed by Cortes and Vapnik (1995). The theory of SVM can be explained by defining four concepts, namely the hyperplane, the optimal margin, the soft margin and the kernel function (Noble, 2006): (i) the hyperplane, is the optimal line in a high dimensional space that separates the data points; (ii) the optimal margin is defined as the distance between the hyperplane and the closest support vectors; (iii) the soft margin which allows for separation of data points with a minimal number of misclassifications; (iv) as data points in a classification problem cannot always be separated linearly, the fourth concept, the kernel function, was developed. The kernel function changes the shape of space to a higher dimension and makes it possible to find a non-linear decision boundary. A kernel function that does not suggest too many unnecessary dimensions is preferred because too many dimensions can lead to over fitting. When the data is perfectly linearly separable only then we can use Linear SVM. Perfectly linearly separable means that the data points can be classified into two classes by using a single straight line. However, in case the data is not linearly separable we can use non-linear SVM. So if the data points cannot be separated into two classes by using a straight line, we use kernel functions to classify them. In most applications we do not find linearly separable data points, hence, we use kernel functions in order to solve them. In this work, two different SVM types are used for our models, namely the linear SVM and the non-linear SVM with radial basis function (RBF) kernel.

The mathematical definition of a hyperplane in two dimensions is given by James et al. (2013) as

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 = 0\,, \tag{3.1}$$

for parameters β_0, β_1 and β_2. Any $\mathbf{x} = (x_1, x_2)^T$ for which equation (3.1) holds is a point on the hyperplane.

Equation (3.1) can be extended to the p-dimensional case by

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p = 0\,, \tag{3.2}$$

with $\mathbf{x} = (x_1, x_2, ..., x_p)^T$ being a vector of length p. Again, if $\mathbf{x}$ satisfies equation (3.2), then $\mathbf{x}$ lies on the hyperplane. In case that $\mathbf{x}$ does not satisfy equation (3.2), then this

indicates that **x** lies on one side of the hyperplane, e.g.

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p > 0.$$

Otherwise, if

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p < 0,$$

then **x** lies on the other side of the hyperplane.

Suppose that we have an $n \times p$ data matrix $\mathbf{X}$ that consists of n training observations in the p-dimensional space with corresponding binary responses falling into two classes, that is, $y_1, ..., y_n \in \{-1, 1\}$, where -1 represents one class and 1 the other class.

Then, a separating hyperplane has the property that

$$y_i \cdot (\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_p X_{ip}) > 0$$

for all $i = 1, ..., n$.

The maximal marginal hyperplane is the separating hyperplane for which the margin is largest. For a set of n training observations $\mathbf{x}_1, ..., \mathbf{x}_n \in \mathbb{R}^p$ and associated class labels $y_1, ..., y_n \in \{-1, 1\}$, the maximal margin hyperplane is e.g. defined by James et al. (2013) as the solution to the following optimization problem. Let $M > 0$ represent the margin of our hyperplane. Then we want to maximize M and optimize $\beta_0, \beta_1, \ldots, \beta_p$ in

$$y_i \cdot (\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M, \qquad \forall i = 1, ..., n,$$

subject to $\sum_{j=1}^{p} \beta_j^2 = 1$. Further details about SVM can be found e.g. in James et al. (2013).

Often classes are not perfectly separable, then so-called slack variables $\xi_1, \ldots, \xi_n$ are introduced in additional constraints

$$\begin{aligned} y_i \cdot \left(\mathbf{x}_i^\top \beta + \beta_0\right) &\geq M(1 - \xi_i), i = 1, \ldots, n \\ \xi_i &\geq 0, i = 1, \ldots, n \\ \sum_{i=1}^{n} \xi_i &\leq C \end{aligned}$$

with $C \geq 0$ constant. Those slack variables allow some leeway compared to perfect separation, and ξ_i corresponds to the proportion of M that observation i lies on the wrong side of its margin line (see, e.g., Hastie et al., 2009). For $0 < \xi_i \leq 1$, observation i lies between its margin line and the hyperplane, and for $\xi_i > 1$ it lies on the wrong side of the hyperplane. Observations on the correct side of the margin yield $\xi_i = 0$. Hence, C is a tuning parameter that restricts the total amount of observations that may lie on wrong sides. Only those observations with $\xi_i \geq 0$ affect the hyperplane and are called support vectors. Note that in addition to linear SVM, there also exists a non-linear version known as the radial SVM, which will also be used in the following.

The main advantage of using SVM is that they are more effective in high dimensions where the number of features can be very large. The SVM commonly achieves better accuracy in comparison to Artificial Neural Networks (ANNs; Sipko and Knottenbelt, 2015). Furthermore, Valero (2016) explains that SVM also gives a sparse solution, which reduces the risk of overfitting. A detailed overview of the SVM approach for classification is given in Wang and Casasent (2008) and Steinwart and Christmann (2008). A manual grid search of hyperparameter values is used. Based on tuning results of the parameters for the radial basis function (RBF) kernel, the best SVM model is identified. Then the performance of the tuned SVM model is evaluated using the testing data set. This is done by using the `tune.svm` function from the `e1071` R package (Dimitriadou et al., 2009).

Artificial neural networks (ANNs)

An artificial neural network (ANN) is a computational model inspired by the way biological neural networks in the human brain process information (Rosenblatt, 1958). ANNs were developed separately in different fields, e.g. statistics and artificial intelligence, andand reach back to at least 1986 (see, e.g., Rumelhart et al., 1986a,b; McClelland et al., 1987). They consist of interconnected nodes, or neurons, organized into layers: an input layer, one or more hidden layers, and an output layer. Each connection between neurons has an associated weight that adjusts as learning proceeds. In a nutshell, an ANN relies on the following components and works as follows:

- *Input Layer*: Data is fed into the network through the input layer.
- *Weighted Sum*: Each neuron calculates a weighted sum of its inputs.
- *Activation Function*: The weighted sum is passed through an activation function (e.g. ReLU or sigmoid). It can introduce non-linearity and determines the neuron's output.
- *Forward Propagation*: The output from one layer becomes the input for the next layer until reaching the output layer.
- *Loss Function*: The difference between the predicted output and actual target values is calculated using a loss function.
- *Backpropagation*: The network adjusts its weights based on this error using optimization algorithms like gradient descent, effectively "learning" from the data.

Through repeated iterations over training data, ANNs can learn complex patterns and make predictions or classifications on new data. A sound introduction to ANNs can be found for example in Hastie et al. (2009).

In this work, we train our ANNs using the R function `neuralnet` (Fritsch et al., 2019). The function allows to use backpropagation, resilient backpropagation (with or without weight backtracking) or a modified globally convergent version. Moreover, it enables flexible settings through custom-choice of error and activation function.

4 Further modeling details and evaluation aspects

In the following, some more details on the regression model approaches, which seem to be suitable to adequately model tennis matches, are provided. Then, both the regression and machine learning approaches are investigated and compared with respect to their predictive performance on new, unseen matches. To achieve this, a leave-one-tournament-out CV-type approach is employed to select the best model based on various performance measures. In an external validation using previously unused data, the top models from the earlier CV-type strategy are assessed through an expanding window method. Subsequently, a rolling window with a recent three-year training dataset is utilized to make predictions for the selected models. For all computations and evaluations the statistical programming software R (R Core Team, 2023) was used.

4.1 Modeling details for the regression approaches

In this section, we provide more details for the regression approaches. These can also be found in Buhamra et al. (2024), but for better readability of this manuscript we describe them here again.

Principally, to model the binary outcome y (win/loss) of a tennis match, the logistic regression models introduced in Section 3.1 can be employed, which generally are given by

$$\ln\left(\frac{\pi_i}{1-\pi_i}\right) = \eta_i = \beta_0 + x_{i1}\beta_1 + \cdots + x_{ip}\beta_p\,.$$

Since the $x_{i1},\ldots,x_{ip},\ i = 1,\ldots,n$, are differences of the covariates of both players, i.e. the value of the second player always being subtracted from the one of the first player, β_0 would represent a "home effect" of the first-named player. However, since the ordering of the two players in our dataset is random, β_0 is not meaningful and, hence, excluded (i.e., β_0 is set to zero). Finally, the $y_i|x_{i1},\ldots,x_{ip}$ are assumed to be independent for $i = 1,\ldots,n$.

Linear effects

The simplest model approach assumes linear covariate effects in the predictor, but requires careful selection of covariates. For this purpose, all possible combinations of the available variables are compared in the CV-type strategy. With seven covariates in the dataset, there are initially $\sum_{i=1}^{7}\binom{7}{i} = 127$ combinations; however, since three of these reflect specific age differences, we ensure that maximum one age variable can be incorporated in each combination. This leads to a total of 63 distinct combinations.

Non-linear effects (splines)

As the assumption of strictly linear effects can possibly be very restrictive, spline models using B-splines are also considered. These enable to incorporate non-linear covariate effects. Again, we regard the same 63 covariate combinations, and again compare the respective models in terms of various performance measures. To obtain smoothness, the splines are penalized via using P-splines. Moreover, an additional regularization approach is used, i.e. additional variable selection is performed on the spline effects by potentially setting the corresponding spline coefficients to zero. This is implemented in the `gam`-function from `mgcv` (Wood, 2017) (via setting the `select` argument to `TRUE`). The underlying methodology of this additional penalization and variable selection approach is described e.g. Marra and Wood (2011).

Surface-specific player skills

Tennis matches are played on different surfaces (grass, hard court, and clay), each influencing match outcomes differently. Hence, it is reasonable to assume that players' performances vary across these surfaces. For instance, Rafael Nadal, known as the "king of clay", has won 14 French Open titles but only two Wimbledon titles on grass. Conversely, Roger Federer has won Wimbledon eight times but just once at the French Open. As already described in Buhamra et al. (2024), we account for these player-surface dynamics by creating a corresponding factor variable using effect encoding. This involves adding artificial columns to the dataset with entries $\in \{-1,0,1\}$ for each player-surface combination; a player can have up to three such columns. If a player has not competed on a surface, the respective column is omitted. Each row contains one entry of 1 and of -1, while all other entries are 0. The entry 1 indicates the first player's combination with the match surface, while -1 corresponds to the second player's combination. For example, the line

$\cdots$	*Nadal.Hard*	*Nadal.Grass*	*Nadal.Clay*	$\cdots$	*Federer.Hard*	*Federer.Grass*	*Federer.Clay*	$\cdots$
$\cdots$	0	1	0	$\cdots$	0	-1	0	$\cdots$

represents a match where Rafael Nadal played against Roger Federer on grass, with Nadal being first-named and Federer second-named. All other entries in that row are 0, as the match was played on grass and only these two players were involved. All these columns are added to our data set, and are used as new features in combination with the existing variables *age, ranking, points, Elo, prob, age.30* and *age.int*.

This results in a large number of new covariates, namely 1,024, and hence leads to an extremely large number of parameters being estimated, and the associated estimators become unstable. Hence, we combine the logistic regression model with LASSO regularization (see Section 3.1). Consequently, this modling approach is restricted to linear covariate effects. These surface-specific player abilities allow the model to detect players who have won or lost more often than average on a specific surface.

Global player skills (LASSO)

Analogously, and again following Buhamra et al. (2024), one could simply allow for global player-specific abilities, to account for players who overall perform even better or worse than the information of their covariate values would suggest (independent from a specific surface type). Similar to the previous paragraph, this can be done by introducing a corresponding factor variable, again via effect coding. Now, only one column per player is added. These columns again only have entries $\in \{-1, 0, 1\}$, with 1 and -1 occurring exactly once per row. Again, in each row, the values 1 and -1 appear at the positions corresponding to the first-named and the second-named player, respectively, and all remaining entries are 0. The following exemplary row

$\cdots$	*Novak.Djokovic*	$\cdots$	*Rafael.Nadal*	$\cdots$	*Roger.Federer*	$\cdots$
$\cdots$	0	$\cdots$	-1	$\cdots$	1	$\cdots$

indicates that Rafael Nadal played as second-named player against the first-named player Roger Federer. For example, the entry in Novak Djokovic's column is then 0 (as well as for all other players), since he was not involved in that particular match. Again, those global player-specific abilities are then added to the design matrix and used along with the basic covariates defined in Section 2. Consequently, again a large number of new covariates is obtained, namely 426, and, hence, a large number of player-specific skill parameters has to be estimated. Once more, we extend the logistic regression model with LASSO penalization and assume linear covariate effects only.

Modeling details for the machine learning approaches

For all the ML models introduced in Section 3.2, no player-specific dummies have been included. Also note that all tree-based approaches implicitly incorporate both non-linear as well as interaction effects. While linear SVM only accounts for linear effects, radial SVM also allows for a certain extend of non-linearity.

4.2 Benchmark model

Similar to Buhamra et al. (2024), we built a benchmark model for prediction based on the probabilities (probs) calculated from the average odds of the different bookmakers included in the `deuce` package. According to e.g. Schauberger and Groll (2018), estimates for the winning probabilities of both players, $\hat{\pi}_{i1}$ and $\hat{\pi}_{i2}$, for match i can be derived from the two corresponding winning odds, odd_{i1} and odd_{i2}, as follows:

$$\hat{\pi}_{i1} = \frac{\frac{1}{odd_{i1}}}{\frac{1}{odd_{i1}} + \frac{1}{odd_{i2}}} \qquad \text{or} \qquad \hat{\pi}_{i2} = \frac{\frac{1}{odd_{i2}}}{\frac{1}{odd_{i1}} + \frac{1}{odd_{i2}}}$$

Naturally, we have $\hat{\pi}_{i1} + \hat{\pi}_{i2} = 1$, adjusting for the bookmaker's margins, which result from the fact that the betting providers artificially lower their betting odds to gain some profit. This means that when the inverse values of the original odds are directly used as probabilities, they do not sum up to 1, but to a value slightly larger than 1. Hence, the sum of the reciprocals in the denominator is used for normalization, assuming implicitly that the margin is equally distributed to both players.

4.3 Performance measures

To compare the predictive performance of the used regression and machine learning models on new, unseen data, several measures are considered (see also Buhamra et al., 2024). First, let $\tilde{y}_1, \ldots, \tilde{y}_n$ denote the true binary outcomes of the n matches, i.e., $\tilde{y}_i \in \{0,1\}, i = 1, \ldots, n$. Second, let $\hat{\pi}_{i1} =: \hat{\pi}_i$ denote the probability, predicted by a certain model, that Player 1 wins the i-th match. Since $\tilde{y}_i$ is binary, the probability that Player 2 wins the match then simply yields $\hat{\pi}_{i2} = 1 - \hat{\pi}_{i1}$.

Classification rate

The (mean) *classification rate* gives the proportion of matches correctly predicted by a certain model, i.e.

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = \hat{y}_i), \text{ where } \quad \hat{y}_i = \begin{cases} 1, & \hat{\pi}_i > 0.5 \\ 0, & \hat{\pi}_i \leq 0.5 \end{cases},$$

see, e.g., Schauberger and Groll (2018). Hence, large values indicate a good predictive performance.

Predictive Bernoulli likelihood

The (mean) *predictive Bernoulli likelihood* is based on the predicted probability $\hat{\pi}_i$ for the true outcome $\tilde{y}_i$, and for n observations is defined as

$$\frac{1}{n}\sum_{i=1}^{n} \hat{\pi}_i^{\tilde{y}_i}(1-\hat{\pi}_i)^{1-\tilde{y}_i},$$

see again Schauberger and Groll (2018). Once again, a large value is an indicator of a good model.

Brier score

The *Brier score* is based on the squared distances between the predicted probability $\hat{\pi}_i$ and the actual (binary) output $\tilde{y}_i$ from match i, and according to Brier (1950) is defined as

$$\frac{1}{n}\sum_{i=1}^{n} (\hat{\pi}_i - \tilde{y}_i)^2.$$

As this is an error measure, here low values indicate a good model.

4.4 Interpretable machine learning

Next, we briefly introduce two methods from the field of interpretable machine learning (IML), namely *partial dependence plots* (PDP; see Friedman, 2001) and *individual conditional expectation* (ICE) plots (Goldstein et al., 2015). These allow to make complex, black box-type machine learning models more interpretable by understanding the

marginal effects of individual features (see R packages `pdp` by Greenwell et al., 2017). More details on IML are provided in Molnar (2020).

Partial Dependence Plot (PDP)

Following Friedman (2001), PDPs display the marginal effect of one or two predictors on the predicted response of a ML model, which might be linear, monotonic, or strongly non-linear. In the notion of Molnar (2020), the partial dependence function is given by

$$\hat{f}_S(\mathbf{x}_S) = \mathbb{E}_{\mathbf{x}_C}[\hat{f}(\mathbf{x}_S, \mathbf{x}_C)] = \int \hat{f}(\mathbf{x}_S, \mathbf{x}_C)\, dP(\mathbf{x}_C)\,,$$

where $P(\cdot)$ is the density of the features in set C, $\mathbf{x}_S$ are the predictors from the (typically small) set S for which $\hat{f}_S(\cdot)$ is desired, while $\mathbf{x}_C$ denotes the remaining predictors included in the underlying ML model $\hat{f}(\cdot)$. by integrating out the influence of the features from C, we obtain the marginal effects of the features of interest in S, still taking into account potential interactions.

For given training data of size n and an arbitrary value of the features from S, $\hat{f}_S(\cdot)$ can be estimated via

$$\hat{f}_S(\mathbf{x}_S) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(\mathbf{x}_S, \mathbf{x}_{i,C})\,,$$

i.e. by their marginal effect on the response, averaging over the observed values $\mathbf{x}_{i,C}$ of the remaining predictors. An important underlying assumptions is that the covariates from the two different set C and S are uncorrelated in order to avoid unrealistic estimates.

Individual Conditional Expectation (ICE)

Closely related to PDPs, ICE plots illustrate the effect of a certain predictor on the individual observation level (Goldstein et al., 2015). The plot contains one line per observation, representing the predicted response for this observation as the covariate of interest varies. Hence, the ICE plot reveals the variability and heterogeneity of the covariate effects across different all individual obersvations. This way, interpretability and transparency of a complex ML model can be increased and outliers identified. Moreover, possible interactions between different covariates may be detected.

5 Results

In the following, the proposed regression models along with the machine learning techniques are compared using the various performance measures introduced in Section 4.3. This is done separately via a leave-one-tournament-out cross validation strategy (Section 5.1), as well as both via an expanding window (Section 5.2) and a rolling window external validation scenario (Section 5.3).

5.1 Leave-one-tournament-out cross validation

In a first step, to evaluate the models only the Grand Slam tournaments from 2011 to 2021 are used, i.e. the tournaments that took place in 2022 are initially not considered and are used as external validation data later on (see Section 5.2). So, from each of the eleven years four tournaments are used with the exception of year 2020, where Wimbledon did not take place due to the COVID-19 pandemic. This results in a sub data set, which contains 4,720 observations, on which a 43-fold CV-type strategy is performed based on the following scheme (similar to Buhamra et al., 2024):

1. From the 43 Grand Slam tournaments present in the data set, a training data set of 42 tournaments and a test data set of the remaining tournament are constructed.
2. Then, all regression and machine learning models introduced above are fitted:
   - For the models with linear influences, the function `glm` from the R package `stats` (R Core Team, 2022) is used. As described in Section 4.1, there are 63 such models, each using at most one of the three variables for age.
   - For the calculation of the spline models the function `gam` from the R package `mgcv` (Wood, 2004) is used. Again, there are also 63 different models here.
   - In order to be able to compute the two LASSO-penalized models, first the design matrices have to be constructed as described in Section 4.1. For a proper usage of the LASSO, then all columns of the design matrix of the training data need to be standardized (including also those columns corresponding to the surface-specific and global player skills).
     The function `cv.glmnet` from the R package `glmnet` (Friedman et al., 2010) is used to compute both models. For this, a 10-fold (inner) CV is first performed on the current training data to find the optimal λ which provides the minimum deviance. The corresponding LASSO model is used afterwards for prediction. 10-fold CV is also recommended in Friedman et al. (2010).
   - The random forest model is fitted via the R package `ranger` (Wright et al., 2019), wheres for the XGBoost model the respective `xgboost` implementation is used (Chen et al., 2019). Finally, for SVM the `e1071` package is employed, see Dimitriadou et al. (2006). Note that for the aforementioned ML models, no player-specific dummies have been included, as the results did not improve (result not shown). Actually, they even partly deteriorated. Also note that the tree-based approaches implicitly incorporate both non-linear and interaction effects.
3. After fitting the respective model, for each match of the test data the probabilities that the first player wins are predicted.
4. Steps 1–3 are repeated until each of the 43 tournaments has once been used as test data set.
5. Finally, the predicted results are compared with the actual results and the performance measures defined in Section 4.3 are calculated.

Table 1 shows the results of the five best models with linear effects and the five best models with spline effects[2]. Additionally, the results of the LASSO models, all machine learning approaches and the benchmark model are shown. The models are listed below in such a way that the best model is determined in first place, the second best in second place and so forth.

The classification rate is above 77% for all models, except for the classification tree (see 1st column). It is noticeable that within each group of model approaches the classification rate is almost the same (differences are only seen in the fourth decimal). However, for the machine learning models the values vary more. The neural network performs best with a value of 0.8021, i.e. this model predicts the correct outcome for

[2]For the models with linear effects and the spline models, the top five models were selected from the 63 models by assigning ranks for each of the three performance measures, classification rate, predictive Bernoulli likelihood, and Brier score, with the best model receiving the highest rank. If the models had equal performances, average ranks were assigned. Moreover, in order to select the best models in terms of all three measures, the three corresponding ranks were averaged per model. Finally, only the five models with the highest average ranks were selected.

Table 1: Results of the leave-one-tournament-out CV approach for the five best models per regression model class, as well as for the machine learning, LASSO approaches and the benchmark model; best performer in bold font.

	Explanatory variables	Class. rate	Likeli-hood	Brier score
Linear	Prob	0.7772	0.6919	0.1549
	Rank, Prob	0.7776	0.6918	0.1550
	Points, Prob	0.7772	0.6917	0.1549
	Rank, Points, Prob	0.7772	0.6917	0.1550
	Age, Prob	0.7766	0.6918	0.1550
Splines	Prob	0.7764	0.6912	**0.1546**
	Rank, Prob	0.7764	0.6912	**0.1546**
	Points, Prob	0.7764	0.6912	0.1547
	Rank, Points, Prob	0.7764	0.6912	0.1547
	Elo, Prob	0.7764	0.6911	0.1547
Machine learning	ANN	**0.8021**	0.6828	0.1560
	Random forest	0.7978	0.6817	0.1584
	SVM (Linear)	0.7961	**0.7415**	0.1675
	SVM (Radial kernel)	0.7829	0.6149	0.1807
	XGBoost	0.7777	0.7234	0.1901
	Classification tree	0.7486	0.6703	0.2178
LASSO	Surface specific	0.7774	0.6816	0.1551
	General skills	0.7770	0.6838	0.1551
	Benchmark	0.7772	0.6709	0.1555

80.21% of the matches. In second place is the RF model, which has a correct prediction value of 0.7978. Also, it is noticeable that all spline models perform (almost) equal with a value of 0.7764. In addition, with respect to the classification rate, four different models deliver the same value of 0.7772.

For the (average) predictive Bernoulli likelihood (2nd column) we obtain a somewhat similar picture. The mean likelihood of the models with both linear and non-linear effects is slightly larger than 0.69, i.e. the models predict the correct outcome with an average probability of about 69%. The two LASSO models are just below 0.69. It is noticeable that the linear SVM model achieves the largest predictive likelihood, which is 0.7415. Again, the spline models performed almost identical with a value of about 0.691.

Across model groups, similar to the classification rate the differences in the Brier scores are rather small. The models with non-linear effects perform best in this regard, with Brier scores between 0.1546 and 0.1547. The models with linear effects produce slightly larger values, followed by the LASSO models and then the machine learning models. Here, the two LASSO models delivered the same value of 0.1551. The classification tree performs clearly worst with a value of 0.2178, followed by XGBoost and the SVM (with radial kernel) with values of 0.1901 and 0.1807, respectively.

Overall, the results show that the spline models with only *prob* or both *rank* and *prob* as covariates, the random forest and the linear SVM perform best across all performance measures.

5.2 Expanding window validation

Next, we will validate the models from above with respect to their predictive performance on new, unseen test data in a more realistic way. For this purpose, we investigate

the three best models from the groups of modeling approaches with linear and non-linear effects, as well as the two LASSO models, the machine learning models and the benchmark model. Again, all performance measures are calculated on the four Grand Slam tournaments 2022, which contains overall 293 matches. To mimic a more realistic prediction use case, this time the validation is performed using an expanding window forecasting approach, i.e. each time one of the remaining tournaments is used as the test data set in chronological order, and the training data set is constantly updated and enlarged. Altogether, this scheme has already been used in Buhamra et al. (2024) and can be explained as follows:

1. First, all tournaments prior to 2022 are used as the training data set. Then, all models are fitted as described in step 2 of the CV approach described in Section 5.1. Based on those, predictions are derived for the 2022 Australian Open matches, as this was the 1st Grand Slam tournament in 2022.
2. Then, the training data is updated, by adding the matches of the Australian Open 2022 on which the models are fitted again and predictions are made for the French Open 2022, which is the 2^{nd} Grand Slam tournament in 2022.
3. Next, the matches of the French Open 2022 are added to the training data set and the models are fitted again. Based on those fits, Wimbledon 2022 is predicted.
4. Then, the Wimbledon 2022 matches will be added to the training data set, and again, the models are fitted and predictions are made for the final Grand Slam tournament in 2022, the US Open 2022.

Finally, the prediction results for all four tournaments are compared with the actual match outcomes, and the corresponding performance measures are calculated (see Table 2 for results).

Table 2: Results of the expanding window approach for the best three regression models out of those five best models found Section 5.1, as well as machine learning, LASSO approaches and the benchmark model; best performer in bold font.

	Explanatory variables	Class. rate	Likeli-hood	Brier score
Linear	Rank, Prob	0.7816	0.7037	0.1418
	Prob	0.7816	0.7036	0.1419
	Points, Prob	0.7747	0.7027	0.1421
Splines	Points, Prob	0.7782	0.7033	**0.1411**
	Prob	0.7782	0.7031	0.1413
	Rank, Prob	0.7782	0.7031	0.1413
Machine learning	SVM (Linear)	0.8295	**0.7637**	0.1446
	XGBoost	**0.8306**	0.7526	0.1562
	Random forest	0.8162	0.6734	0.1511
	SVM (Radial kernel)	0.7956	0.5858	0.1882
	ANN	0.7918	0.6904	0.1466
	Classification tree	0.7577	0.6803	0.2061
LASSO	General skills	0.7782	0.6969	0.1416
	Surface specific	0.7782	0.6934	0.1423
	Benchmark	0.7816	0.6810	0.1441

Regarding the classification rate, most of the models performed better here compared to the classification rates from the CV-type strategy from above. However, here the classification tree model delivers the worst results with a value of 0.7577, while

the best value is 83.06% and is achieved by the XGBoost model. Now, the benchmark model is not among the best.

Similar trends as in Table 1 (cf. page 17) can be seen for the predictive likelihood, although the results here have generally slightly improved. XGBoost together with linear SVM achieve the largest likelihood, where the linear SVM model delivers the best value with 0.7637, only slightly behind are the linear and spline regression models that perform almost equally (all around between 0.7031 and roughly 0.7036): in case of the linear models the differences appeared in the third decimal, while in case of non-linear effects the differences can be seen only in the fourth decimal. Also, it should be noted that two of the spline models deliver the same value of 0.7031. Both the benchmark model and the LASSO models yield an even slightly smaller likelihood. The LASSO models perform almost equally with differences that can be seen only in the third decimal place. The SVM model with radial kernel here performs worst.

For most of the model the Brier score is $\in [0.1411 - 0.1441]$, except for the machine learning models, where the values differ more. Compared to Section 5.1, the values are slightly smaller. The classification tree model again yields the largest, and thus worst value with 0.2061. The best value is 0.1411 and is achieved by the non-linear model including covariates *points* and *prob*.

Overall, interestingly almost all approaches have improved, some even quite substantially. This indicates that the predictions should occur in chronological order. This time the results show that the spline models with both *points* and *prob* as covariates, XGBoost and linear SVM perform best, each with respect to one of the performance measures. Overall, also the linear model with only *prob* performs quite well. Altogether, the expanding window strategy seems to be clearly superior compared to the CV-type strategy.

5.3 Rolling window validation

An alternative strategy is to only use recent data for prediction, but in a rolling window manner. Hence, only 3 years of training data are used, i.e. 12 tournaments in total, starting from the first tournament in 2011, which is Australian open, until we reach the last tournament of 2013, which is US open. This means we use the 12 tournaments from 2011 to 2013 as first training data set and then predict the 13th tournament (i.e., the first tournament of 2014, which is Australian Open). Then, a rolling window approach is performed and the training data set constantly changes with a fixed short time window of three years. We again focus on the three best models from the groups of linear and non-linear regresion models, as well as the two LASSO models, the machine learning models and the benchmark model. In detail, this scheme can be explained as follows:

1. The tournaments of the first 3 years in our data set, 2011 - 2013, are used as training data, on which the models are fitted. Then, predictions can be obtained for the 2014 Australian Open, the 13^{th} tournament in our data set and first Grand Slam tournament in 2014.
2. The new training data set then shifts by one tournament and starts from the 2^{nd} tournament for 2011, which is French Open 2011 and ends with the 13^{th} tournament, i.e. Australian Open 2014. Again, all models are fitted and used to predict the next tournament, which is now the 14^{th} tournament in our data set (and 2^{nd} Grand Slam tournament in 2014).
3. This procedure is repeated until finally, the last training data consists tournaments 35-46 from our full data set, on which again all models are fitted and used to predict the very last tournament in our data, which is the 47^{th} tournament, namely the final Grand Slam tournament in 2022, US Open 2022.

Finally, the predicted results for all tournaments, on which predictions have been made, starting from Australian open 2014 until the US open 2022, are compared with the actual results and the respective performance measures are calculated. The results of the rolling window are shown in Table 3.

Table 3: Results of the rolling window approach for the best three regression models out of those five best models found Section 5.1, as well as machine learning, LASSO approaches and the benchmark model; best performer in bold font.

	Explanatory variables	Class. rate	Likeli-hood	Brier score
Linear	Prob	0.7743	0.6893	0.1562
	Points, Prob	0.7740	0.6890	**0.1546**
	Rank, Prob	0.7714	0.6884	0.1563
Splines	Rank, Prob	0.7727	0.6895	0.1564
	Points, Prob	0.7719	0.6887	0.1566
	Prob	0.7708	0.6884	0.1563
Machine learning	SVM (Linear)	**0.8012**	**0.7424**	0.1709
	Random forest	0.7971	0.6727	0.1608
	XGBoost	0.7755	0.7231	0.2074
	SVM (Radial kernel)	0.7729	0.5448	0.2139
	ANN	0.7726	0.6837	0.1554
	Classification tree	0.7256	0.6900	0.2678
LASSO	Surface specific	0.7735	0.6735	0.1564
	General skills	0.7727	0.6753	0.1565
	Benchmark	0.7743	0.6683	0.1567

Principally, the results are quite similar compared to the previous results, but have all deteriorated a bit. The classification rate is again slightly above 77% for all models, with differences mostly only in the third decimals, except for the machine learning models which partly perform better (random forest and linear SVM). Here, linear SVM performs best with a value of 0.8012, i.e. this model predicts the correct outcome for 80.12% of the matches. Also, the random forest model here delivered good performance with a value of 0.7971. All non-linear models performed very similar, with differences that can bee found only in the third decimal place. The two LASSO models also achieved very similar values around 0.773. While three other models deliver the value 0.774.

Compared to the results in Table 2, the predictive likelihood results are somewhat lower here. The best value is achieved by the linear SVM model (0.7424), followed by XGBoost with a value of 0.7231. The differences in both linear and non-linear regression model can only be seen in the third decimal. Both LASSO models yield very similar results around 0.67. The SVM with radial kernel and the benchmark model perform worst with values of 0.5448 and 0.6683, respectively.

Also the Brier score values are slightly worse (i.e., larger) compared to the previous results from Table 2. This clearly indicates that for a good predictive performance on Grand Slam matches, training data should be rather large, and a collection of three years of Grand Slam matches data as used here is not sufficient. Hence, besides the precise choice of statistical or machine learning approach, a sound basis of informative covariates and large enough sample sizes are very important. The classification tree model once again yields the worst value with 0.2678. The linear model including covariates *points* and *prob* is achieved the best value of 0.1546. However, all linear and non-linear regression models, the tow LASSO models and the benchmark model deliver roughly

similar values with differences that can be seen only in the fourth decimal. All machine learning models instead result in rather large values, with linear SVM, random forest and neural network still being acceptable.

5.4 Interpretable machine learning

In the following, interpretable machine learning (IML) is used to analyze the performance of XGBoost model, which achieved the largest classification rate among all models, with a value of 0.8306, as discussed in Section 5.2. For this purpose, combined partial dependence plots (PDPs; Friedman, 2001) and Individual Conditional Expectation (ICE; Goldstein et al., 2015 plots are presented to enhance the interpretability and visualization of the respective complex model. For the implementation in R, we rely on the `pdp` package (Greenwell et al., 2017). Also, we use the `ggplot2` (Hadley, 2016) to create custom PDPs and ICE plots. In Figure 1, a multi-panel plot is presented where each panel represents the combined PDP and ICE plot for a different covariate (namely, *age.30*, *Elo*, *points* and *rank*). The x-axis of each panel represents the respective covariate (recall that we account for differences), while the y-axis shows the corresponding predicted winning probabilities. This visualization helps in understanding how each predictor variable individually influences the predicted outcome, highlighting potential non-linear relationships captured by the model. The ICE plot (gray lines) shows how predictions vary for individual instances, while the PDP (bold black line) displays the overall average effect of the variable.

The steep increasing trend for the *Elo* panel in the PDP line indicates that as *Elo* differences increase, the predicted probability of Player 1 winning also increases. Regarding the ICE lines, as they vary widely, the impact of *Elo* difference is not consistent across all matches, possibly due to interactions with other features. For the *rank*, PDP line decreases with increasing *rank* difference, which means that when Player 1 has a worse ranking, his predicted probability of winning decreases. The PDP for *age.30* shows a slightly increasing trend at the beginning, then a slight decrease, indicating that *age* difference does not have a substantial impact on predictions. Toward the end, we observe another obvious fluctuating trend. This could mean that the impact of extreme *age* differences is inconsistent.

For covariate *points*, the PDP line also fluctuates rather strongly, indicating that the effect of the *points* difference on the predicted probability of winning is not straightforward, i.e., there could be ranges where having a larger points advantage helps, but other ranges where the effect is weaker or even reversed. Another explanation could be that the impact of the *points* difference correlates with other covariates like *surface*, *age*, or *Elo* rating. For example, a player with a small *points* advantage might have a higher winning probability on clay courts, but not necessarily on grass courts (i.e., there is an interaction effect with other features).

6 Summary and overview

In this manuscript, we continued the work from Buhamra et al. (2024) and modeled tennis matches in Grand Slam tournaments within the framework of regression and modern machine learning approaches. For this purpose, the same data set as used in Buhamra et al. (2024) was analyzed, which was build based on the `deuce` R package (Kovalchik, 2019). The final data set contains information on 5,013 matches in 47 men's Grand Slam tournaments from the years 2011-2022, and includes covariate information on the *age* difference of both players, the difference in their *rank* positions and ATP *points*, in *Elo* rating, in probabilities for the victory of the players, calculated from the average

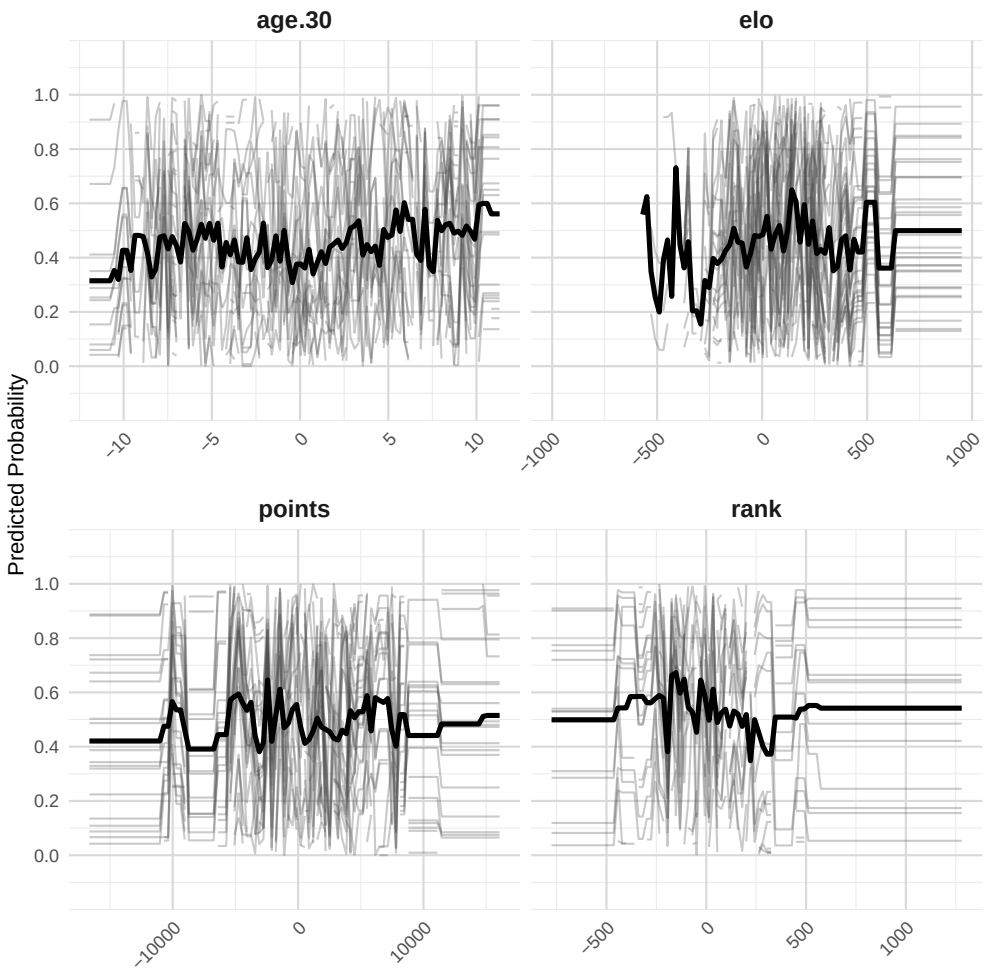

Figure 1: Combined PDP (thick black line) and ICE plots (gray lines) for XGBoost model in combination with the expanding window approach, including covariates *age.30*, *Elo*, *points* and *rank*

odds by the betting providers (*prob*), as well as the two additional age variables, *age.30* and *age.Int*, which take into account that the optimal age of a tennis player is between 28 and 32 years.

Different regression models, which were already considered in Buhamra et al. (2024), were compared to modern machine learning approaches for modeling and prediction of tennis matches. As the outcome in tennis is binary (win or loss), the probability of the first-named player to win is modeled. It was discussed that for the logistic regression approaches, modeling with intercept is not suitable, as this would incorporate a kind of home effect for the first-named player - a property which was not desired here.

The different modeling approaches were compared in (i) a 43-fold leave-one-tournament-out CV-type strategy, (ii) an expanding window, and (iii) a rolling window validation strategy (with a short window for a period of three years). The following regression and ML approaches were included:

- Logistic regression with linear effects: all possible combinations of the seven covariates, such that at most one of the three age variables was incorporated, were considered. This resulted in 63 models.
- Logistic regression with non-linear effects (splines): Again, the same 63 models were considered.
- A model taking into account surface- and player-specific effects. Due to the large

amount of unknown parameters, here LASSO penalization was used.

- A model taking into account general player-specific abilities. Again, LASSO penalization was used.
- Different machine learning approaches: classification trees, random forest, XGBoost, and support vector machine with linear and radial kernel.
- A benchmark model, where the predicted probabilities were derived from average betting odds.

Within the CV-type approach, the models were compared in terms of the classification rate, the predictive Bernoulli likelihood, and the Brier score. Since 63 different models resulted for each of the first two approaches, the five best models were selected in each case.

It was found that the selected linear and spline models performed very similarly in terms of classification rate, likelihood and Brier score. The classification rate was slightly above 77% for all models, meaning that the models predicted the correct outcome in about 77% of the matches. The predictive Bernoulli likelihood was about 0.69, while the LASSO models and the benchmark model were slightly below. The models thus predicted with an average probability of about 69% the correct outcome of a match. The Brier score was above 0.154 for all models, and differences were mostly found either in the third or fourth decimal. Regarding the machine learning models the values for the Brier score were more volatile.

The tournaments in 2022 were then used as an external validation data set in an expanding window validation approach. The three best models with each linear and non-linear effects, the two LASSO models, all machine learning models and the benchmark model were then compared again. Principally, the results of the preceding CV-type competition could be mostly confirmed, but surprisingly the expanding window strategy turned out to be clearly superior compared to the CV-type strategy: the classification rate was around $0.757 - 0.831$, the predictive likelihood yielded around $0.681 - 0.764$, and the Brier score was between $0.141 - 0.206$. Obviously, performances of the models vary across measures. The XGBoost has the best performance among all others when considering the classification rate. The (linear) SVM model achieves the best predictive performance in terms of the predictive likelihood. Meanwhile, the spline model, including both covariates *points* and *prob*, outperformed all other models by yielding the lowest Brier score value.

In order to use more recent data sets for prediction, we also investigated a rolling window approach, which was based on a training data containing 12 tournaments and which was used to always predict the next tournament. The same selection of models as for the expanding window validation approach was analyzed. Principally, the results of the preceding expanding window-strategy competition could be mostly confirmed, with the values of the performance measures generally being slightly worse. The classification rate was around between $0.771 - 0.801$, the predictive likelihood yielded around $0.545 - 0.742$, and the Brier score was between $0.154 - 0.268$. Again, the values somewhat differ between models and proposed performance measures. If we consider the classification rate and the predictive likelihood, then the (linear) SVM model outperforms all other models. However, for the Brier score, the linear regression model including *points* and *prob* achieves the best performance.

So, interestingly, our analyses show that for the prediction of Grand Slam tennis matches, while there are slight differences across different statistical and machine learning approaches, the specific forecast strategy used seems to play an even more important role. It turned out that on our data, an expanding window approach was clearly superior compared to two other forecast strategies. The results show that on the one hand,

it seems to be important to do the predictions in chronological order, as the expanding window approach clearly outperforms the CV-type strategy. On the other hand, the training data needs to be rather large, and three years of training data as used in our rolling window approach are not sufficient. Particularly, complex machine learning methods such as SVM and XGBoost, which involve an extensive tuning process, seem to benefit from these two aspects to fully exploit their potential, as they exhibit rather good predictive performance (see Table 2). These findings are consistent with the generally known result that feature engineering tends to be the more critical choice for the modeler than the choice among strongly related models.

In the same direction heads a certain feature engineering-type approach, namely *statistical enhanced learning* (SEL), which is based on so-called "statistical enhanced covariates". These are covariates which are not directly observed but obtained by performing certain steps of feature engineering, or even as statistical estimators from separate statistical model, see Felice et al. (2023) for more details. For example, in the context of modeling soccer, Ley et al. (2019) proposed a model to estimate flexible, time-varying team-specific ability parameters. The resulting estimates were then added to the set of (conventional) features in a random forest model, which turned out to be quite sucessful for predicting the FIFA World Cup 2018 in Groll et al. (2019). A first attempt to carry over this idea to tennis is sketched in Buhamra and Groll (2025). Moreover, we are currently working on extending the approach from Ley et al. (2019) to professional tennis (see Bartmann et al., 2025). In future research, we plan to investigate the predictive potential of such "statistical enhanced covariates" in more detail and to combine the methods proposed here with the player-specific ability parameters obtained with approach from Bartmann et al. (2025).

References

Bartmann, H., Groll, A., and Michels, R. (2025). Estimation of match abilities for tennis players via a maximum likelihood approach. Submitted at the IES 2025 conference proceedings.

Breiman, L. (1996a). Bagging predictors. *Machine learning*, 24:123–140.

Breiman, L. (1996b). Heuristics of instability and stabilization in model selection. *The annals of statistics*, 24(6):2350–2383.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees. wadsworth int. *Group*, 37(15):237–251.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78:1–3.

Buhamra, N. and Groll, A. (2025). Statistical enhanced learning for modeling and prediction tennis matches at grand slam tournaments. *arXiv preprint arXiv:2502.01613*.

Buhamra, N., Groll, A., and Brunner, S. (2024). Modeling and prediction of tennis matches at grand slam tournaments. *Journal of Sports Analytics*, 10(1):17–33.

Bühlmann, P. and Hothorn, T. (2007). Boosting algorithms: Regularization, prediction and model fitting.

Bühlmann, P. and Yu, B. (2003). Boosting with the l 2 loss: regression and classification. *Journal of the American Statistical Association*, 98(462):324–339.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Chen, T., He, T., Benesty, M., and Khotilovich, V. (2019). Package xgboost. *R version*, 90:1–66.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y., and Yuan, J. (2023). xgboost: Extreme gradient boosting. R package version 1.7.5.1.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20:273–297.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., Weingessel, A., and Leisch, M. F. (2006). The e1071 package. *Misc Functions of Department of Statistics (e1071), TU Wien*, pages 297–304.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D., Weingessel, A., and Leisch, M. F. (2009). Package e1071. *R Software package, avaliable at http://cran. rproject. org/web/packages/e1071/index. html*.

Eilers, P. H. and Marx, B. D. (2021). *Practical smoothing: The joys of P-splines*. Cambridge University Press.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, 11:89–121.

Fahrmeir, L., Kneib, T., Lang, S., and Marx, B. (2013). *Regression. Modells, Methods and Applications*. Springer, Berlin.

Fahrmeir, L. and Tutz, G. (2001). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer-Verlag, New York, 2nd edition.

Felice, F., Ley, C., Groll, A., and Bordas, S. (2023). Statistically enhanced learning: a feature engineering framework to boost (any) learning algorithms. *arXiv preprint arXiv:2306.17006*.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *icml*, volume 96, pages 148–156. Citeseer.

Friedman, J., Hastie, T., and Tibshirani, R. (2000). Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Fritsch, S., Guenther, F., and Wright, M. N. (2019). *neuralnet: Training of Neural Networks*. R package version 1.44.2.

Gao, Z. and Kowalczyk, A. (2021). Random forest model identifies serve strength as a key predictor of tennis match outcome. *Journal of Sports Analytics*, 7(4):255–262.

Goldstein, A., Kapelner, A., Bleich, J., and Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *journal of Computational and Graphical Statistics*, 24(1):44–65.

Greenwell, B. M. et al. (2017). pdp: An r package for constructing partial dependence plots. *R J.*, 9(1):421.

Groll, A., Ley, C., Schauberger, G., and Van Eetvelde, H. (2019). A hybrid random forest to predict soccer matches in international tournaments. *Journal of Quantitative Analysis in Sports*, 15:271–287.

Hadley, W. (2016). *Ggplot2: Elegrant graphics for data analysis*. Springer.

Hastie, T., Tibshirani, R., Friedman, J. H., and Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer.

James, G., Witten, D., Hastie, T., Tibshirani, R., et al. (2013). *An introduction to statistical learning*, volume 112. Springer.

Kovalchik, S. (2019). deuce: resources for analysis of professional tennis data. *R package version 1.4*.

Ley, C., Wiele, T. V. d., and Eetvelde, H. V. (2019). Ranking soccer teams on the basis of their current strength: A comparison of maximum likelihood approaches. *Statistical Modelling*, 19(1):55–73.

Marra, G. and Wood, S. N. (2011). Practical variable selection for generalized additive models. *Computational Statistics & Data Analysis*, 55(7):2372–2387.

Mayr, A., Binder, H., Gefeller, O., and Schmid, M. (2014). The evolution of boosting algorithms. *Methods of information in medicine*, 53(06):419–427.

McClelland, J. L., Rumelhart, D. E., Group, P. R., et al. (1987). *Parallel distributed processing, volume 2: Explorations in the microstructure of cognition: Psychological and biological models*, volume 2. MIT press.

Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.

Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society*, A 135:370–384.

Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567.

Oshiro, T. M., Perez, P. S., and Baranauskas, J. A. (2012). How many trees in a random forest? In *Machine Learning and Data Mining in Pattern Recognition: 8th International Conference, MLDM 2012, Berlin, Germany, July 13-20, 2012. Proceedings 8*, pages 154–168. Springer.

Park, M. Y. and Hastie, T. (2007). L_1-regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society Series B*, 19:659–677.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986a). Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Rumelhart, D. E., McClelland, J. L., Group, P. R., et al. (1986b). *Parallel distributed processing, volume 1: Explorations in the microstructure of cognition: Foundations*. The MIT press.

Schauberger, G. and Groll, A. (2018). Predicting matches in international football tournaments with random forests. *Statistical Modelling*, 18(5-6):460–482.

Sipko, M. and Knottenbelt, W. (2015). Machine learning for the prediction of professional tennis matches. *MEng computing-final year project, Imperial College London*, 2.

Somboonphokkaphan, A., Phimoltares, S., and Lursinsap, C. (2009). Tennis winner prediction based on time-series history with neural modeling. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, pages 18–20. Citeseer.

Steinwart, I. and Christmann, A. (2008). *Support vector machines*. Springer Science & Business Media.

Therneau, T., Atkinson, B., Ripley, B., and Ripley, M. B. (2015). Package rpart. *Available online: cran. ma. ic. ac. uk/web/packages/rpart/rpart. pdf (accessed on 20 April 2016)*.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society*, B 58:267–288.

Valero, C. S. (2016). Predicting win-loss outcomes in mlb regular season games–a comparative study using data mining methods. *International Journal of Computer Science in Sport*, 15(2):91–112.

Wang, Y.-C. F. and Casasent, D. (2008). New support vector-based design method for binary hierarchical classifiers for multi-class classification problems. *Neural Networks*, 21(2-3):502–510.

Weston, D. (2014). Using age statistics to gain a tennis betting edge. `http://www.pinnacle.com/en/betting-articles/Tennis/atp-players-tipping-point/LMPJF7BY7BKR2EY`.

Whiteside, D., Cant, O., Connolly, M., and Reid, M. (2017). Monitoring hitting load in tennis using inertial sensors and machine learning. *International journal of sports physiology and performance*, 12(9):1212–1217.

Wilkens, S. (2021). Sports prediction and betting models in the machine learning age: The case of tennis. *Journal of Sports Analytics*, 7(2):99–117.

Wood, S. N. (2004). Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association*, 99(467):673–686.

Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, London, 2nd edition.

Wright, M. N., Wager, S., Probst, P., and Wright, M. M. N. (2019). Package ranger. *Version 0.11*, 2.

Wright, M. N. and Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17.

Statistical enhance learning for modeling and prediction tennis matches at Grand Slam tournaments

N. Buhamra[1] A. Groll[2]

[1]Department of Statistics, TU Dortmund University, Vogelpothsweg 87, 44221 Dortmund, email: nourah.buhamra@tu-dortmund.de

[2]*Corresponding author*, Department of Statistics, TU Dortmund University, Vogelpothsweg 87, 44221 Dortmund, email: groll@statistik.tu-dortmund.de

Abstract

In this manuscript, we concentrate on a specific type of covariates, which we call statistically enhanced, for modeling tennis matches for men at Grand slam tournaments. Our goal is to assess whether these enhanced covariates have the potential to improve statistical learning approaches, in particular, with regard to the predictive performance. For this purpose, various proposed regression and machine learning model classes are compared with and without such features. To achieve this, we considered three slightly enhanced variables, namely elo rating along with two different player age variables. This concept has already been successfully applied in football, where additional team ability parameters, which were obtained from separate statistical models, were able to improve the predictive performance.

In addition, different interpretable machine learning (IML) tools are employed to gain insights into the factors influencing the outcomes of tennis matches predicted by complex machine learning models, such as the random forest. Specifically, partial dependence plots (PDP) and individual conditional expectation (ICE) plots are employed to provide better interpretability for the most promising ML model from this work. Furthermore, we conduct a comparison of different regression and machine learning approaches in terms of various predictive performance measures such as classification rate, predictive Bernoulli likelihood, and Brier score. This comparison is carried out on external test data using cross-validation, rolling window, and expanding window strategies.

Keywords: Grand Slam tournaments, tennis matches, machine learning, prediction, statistical enhance covariates, interpretable machine learning, expanding window.

1 Introduction

In recent years, various methodologies for statistical and machine learning based modeling of tennis matches and tournaments have emerged, and the existing methods for predicting the probability of winning matches in tennis have been expanded. Moreover, there is potential to calculate winning probabilities for an entire tournament when all individual matches can be predicted.

Recently, machine learning (ML) models have been employed to predict the outcomes of tennis matches. Somboonphokkaphan et al. (2009) introduced a method utilizing match statistics and environmental data to predict winners using a Multi-Layer Perceptron (MLP) with a back-propagation learning algorithm. MLP, a type of Artificial Neural Network (ANN), is effective for solving real-world classification problems and predicting outcomes, especially when handling large databases with incomplete or noisy data. Whiteside et al. (2017) proposed an automated stroke classification system to quantify hitting load in tennis, using machine learning techniques like a cubic kernel support vector machine. Wilkens (2021) expanded previous research by applying various ML techniques, including neural networks and random forests, with extensive datasets from professional men's and women's tennis singles matches. Despite these efforts, he found that the average prediction accuracy does not exceed 70%.

Sipko and Knottenbelt (2015) predicted match winners based on the probability of winning serve points, which subsequently indicates the overall probability of winning the match. They extracted 22 features from historical data, including abstract features like player fatigue and injury, and optimized models that outperformed Knottenbelt's Common-Opponent model using ML approaches such as artificial neural networks (ANNs). They suggest that ML-based techniques can innovate tennis betting, noting that ANNs generated a 4.35% return on investment, a 75% improvement in the betting market. Moreover, Gao and Kowalczyk (2021) developed a model that predicts tennis match outcomes with over 80% accuracy, surpassing predictions based on betting odds alone, and identifying serve strength as a crucial predictor. Their model used a random forest classifier, highlighting the importance of simple models even in the age of deep learning. Their comprehensive data set, compiled from ATP data from 2000 to 2016, includes a variety of features capturing physical, psychological, court-related, and match-related variables. Finally, a comprehensive overview of modeling and predicting tennis matches at Grand Slam tournaments by different regression approaches has been presented in Buhamra et al. (2024).

The main focus of this manuscript, however, is to analyze, whether so-called enhanced covariates have the potential to improve statistical and machine learning approaches, in particular, with regard to predictive performance. Generally, in recent years, there has been a growing interest in feature engineering. Effective feature engineering plays a crucial role in enhancing the performance of machine learning models by identifying and capturing relevant patterns and relationships within the data. This enables models to improve their predictive accuracy and extract meaningful insights from the data. For instance, Felice et al. (2023) introduce the concept of Statistically Enhanced Learning (SEL), a formalization framework for existing feature engineering and extraction tasks in ML. This approach has the potential to address challenges in ML tasks by optimizing feature selection and representation.

For example, in the context of modeling soccer, Ley et al. (2019) proposed a model to estimate flexible, time-varying team-specific ability parameters. The resulting estimates were then added to the set of (conventional) features in a random forest model, which turned out to be quite successful for predicting the FIFA World Cup 2018 in Groll et al. (2019a).

When using modern and complex ML models, another important aspect is interpretability of the fitted model. Hence, several studies have been conducted in the field of understanding and interpreting complex (black box-type) ML models. For example, Auret and Aldrich (2012) used variable importance measures, directly generated by the random forest models, and partial dependence plots, indicating that random forest models can reliably identify the influence of individual variables, even in the presence of high levels of additive noise.

Moreover, Goldstein et al. (2015) present both individual conditional expectation (ICE) plots and classical partial dependence plots (PDPs) on three different real data sets, namely depression clinical trial, white wine and diabetes classification in Pima Indians. They demonstrate how ICE plots can shed light on estimated models in ways PDPs cannot. Accordingly, ICE plots refine the PDP by graphing the functional relationship between the predicted response and the feature for individual observations. In particular, ICE plots highlight the variation in the fitted values across the range of a certain selected covariate, suggesting where and to what extent heterogeneities might exist.

More generally, Molnar et al. (2023) investigated the relationship between Partial Dependence Plots (PDPs) and Permutation Feature Importance (PFI) methods in understanding the data generating process in machine learning models. They explored how these two techniques can provide complementary insights into the importance and effects of features on model predictions. Consequently, they formalize PDP and PFI as statistical estimators representing the ground truth estimands rooted from the data generating process. Their analysis reveals that PDP and PFI estimates can deviate from this ground truth not only due to statistical biases but also due to variations in learner behavior and errors in Monte Carlo approximations. To address these uncertainties in PDP and PFI estimation, they introduce the learner-PD and learner-PFI approaches, which involve model refits, and propose corrected variance and confidence interval estimators.

Unlike traditional black-box models, interpretable machine learning (IML) models aim to provide insights into the decision-making process, enabling users to make informed decisions and understand the implications of model outputs. Therefore, in this work, we focused on IML models, such as partial dependency plots (PDP; Friedman, 2001) and Individual Conditional Expectation (ICE; Apley and Zhu, 2020), to make our employed random forest model more interpretable. Additionally, we demonstrate how feature engineering techniques can be applied in the context of sports analytics to enhance predictive modeling and gain insights from sports data. Specifically, we examine the use of covariates such as *Elo*, *Age.30* and *Age.int*, which all are not directly available, but are obtained from either a separate modeling approach (*Elo*) or via meaningful mathematical transformations (*Age.30*, *Age.int*), in order to enhance statistical models for tennis. For this application, various regression and machine learning approaches were considered, including linear regression, spline models, and random forest. These approaches were then compared based on various performance measures within an expanding window strategy for analyzing tennis data from Grand Slam tournaments. To conduct this analysis, a dataset was compiled using the R package `deuce` (Kovalchik, 2019). This data set included information on 5,013 matches from men's Grand Slam tournaments spanning the years 2011 to 2022. Several potential covariates were considered, including the players' age, ATP ranking and points, odds, elo rating, as well as two additional age variables. These additional age variables were designed to reflect the "optimal" age range of a tennis player, which is typically between 28 and 32 years (Weston, 2014).

The remainder of the article is structured as follows. Section 2 briefly introduces the data set and defines the objectives of this work. Then, in Section 3, different modeling approaches are introduced, including classical regression approaches and ML methods such as random forest. Besides, some interpretable machine learning techniques like partial dependence plots (PDP) and individual conditional expectation (ICE) plots are discussed, and various performance measures are defined. In Section 4, the modeling approaches are compared in terms of various performance measures, using an expanding window strategy. Additionally, interpretations for different model classes considering enhanced covariates and IML techniques are provided. Finally, Section 5

summarizes the main results and gives a final overview.

2 Data

Next, we shortly introduce our data set, which was compiled using the R package `deuce` (Kovalchik, 2019), and contains information on 5,013 matches from men's Grand Slam tournaments from 2011 to 2022. It was already used in Buhamra et al. (2024) and Buhamra et al. (2025), and contains the following variables.

Victory: A dummy variable indicating whether the first-named player won the match (1: yes, 0: no), used as the response variable in all models.

Conventional covariates

The following three covariates are conventional covariates, which could be extracted more or less directly from public sources. They are all incorporated in our final data set in the form of differences, i.e. for each feature the value of the 2nd player is subtracted from the one of the 1st player.

Age: A metric predictor collecting the age of the players in years. Note that players' ages were not given directly and had to be deduced from the player's date of birth as well as the date of the respective match.

Rank: For each player, the rank at the start of the respective tournament was collected. The position in the ranking is based on the ATP ranking points.

Points: The ATP world ranking points are awarded for each match won per tournament. Wins in later rounds of a tournament are valued higher than wins in the first rounds of a tournament. Points earned in a tournament expire after 52 weeks.

Enhanced covariates

Next, we introduce three covariates which we call enhanced, as they are not directly available, but are obtained from either a separate modeling approach (*Elo*) or via meaningful mathematical transformations (*Age.30, Age.int*), in order to enhance statistical models for tennis. Again, also these features are all incorporated as differences (value of 2nd player minus value of 1st player).

Elo: For each player, the overall Elo rating before a certain match is collected. The idea of the Elo rating system is that one can more accurately track and predict player performances over time, taking into account the relative strength of opponents and the outcomes of matches. Each player starts with an initial rating (often set around 1500 points, though this can vary depending on the specific implementation). After each match, the players' ratings are updated based on the match outcome. If a higher-rated player wins, they gain fewer points than if a lower-rated player wins. The amount of points exchanged in a match depends on the difference in ratings between the two players. Hence, this covariate is considered "enhanced" as it involves complex calculations and provides a more delicate measure of player performance. Further details on the Elo rating in tennis can be found in Angelini et al. (2022) and Vaughan Williams et al. (2021)

Age.30: This variable is given by the absolute distance between the age of the players and reference age 30. This is based on the assumption that the standard *Age* variable introduced above does not contain enough information. For example,

while a 25-year-old player typically has an age-advantage over a 20-year-old one, a 40-year-old player rather has a disadvantage over a 35-year-old one; and, in both cases, the age difference between the two players equals 5 years. Following Weston (2014), who argued that the optimal age of tennis players is between 28 and 32 years, we chose the mid-point as the reference age.

Age.int: This feature is based on the assumption that the optimal age of a tennis player lies within the interval $[28;32]$. Then, the smaller distance to the limits of this interval was derived, i.e. for players younger than 28 the distance to 28 was calculated and for players older than 32 the distance to 32 was calculated. For players between 28 and 32 the distance was set to 0.

A detailed description of the variables included in the data set can be found in Section 2 of Buhamra et al. (2024) and Buhamra et al. (2025) .

Note at this point that the data set does not include matches in which one of the two players gave up or was unable to compete, e.g. due to injury, such that the other player won without actually playing the match. These matches do not contain any relevant information for the present analysis and, hence, in order not to distort the results, are excluded. Moreover, the data set does not contain any missing values.

Based on this data set, the best possible statistical enhance learning model for predicting tennis matches at Grand Slam tournaments is sought. Also, it will be examined whether a machine learning approach, namely a random forest, incorporating enhanced statistical covariates, is more powerful in predicting tennis matches compared to classical regression approaches that also incorporate the corresponding enhanced covariates. Then, for all proposed modeling approaches, including machine learning and classical regression methods, optimal models are determined based on certain performance measures in terms of an expanding window prediction approach. Here, our focus will be only on the expanding window strategy, which reveals a clear winner model, but also other technique such as leave-one-tournament-out cross-validation and a rolling window approach have been considered. The results for those approaches can be found in the appendix. Our main objectives are (i) to quantify how the predictive performance can be improved by incorporating enhanced variables, and (ii) to provide better interpretations for the random forest model using IML tools such as PDP plots and ICE plots.

3 Methods

In the following, first the statistical and machine learning methods used in this work are briefly introduced in Section 3.1. We focus on both standard logistic regression, and generalized additive models based on P-splines. Moreover, the random forest as a representative from the field of machine learning is shortly presented. Then, in Section 3.2 several interpretable machine learning methods are explained, including partial dependence plots (PDP) and individual conditional expectation (ICE) plots. Finally, in Section 3.3 various performance measures are defined.

3.1 Statistical and machine learning methods

In this section, we introduce the classic logistic regression model for binary outcomes, followed by its extension to non-linear effects via spline-based approaches.

Logistic regression

Given observations $(y_i, x_{i1}, \ldots, x_{ip})$ for $i = 1, \ldots, n$ tennis matches,

$$\pi_i = P(y_i = 1 | x_{i1}, \ldots, x_{ip}) = E[y_i | x_{i1}, \ldots, x_{ip}]$$

is the (conditional) probability for $y_i = 1$, i.e. Player 1 winning the match, given covariate values $x_{i1}, \ldots, x_{ip}$.

We further specify a strictly monotonically increasing *response function* $h\colon \mathbb{R} \to [0,1]$,

$$\pi_i = h(\eta_i) = h(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}), \tag{3.1}$$

which relates the linear predictor η_i to π_i.

The logistic regression model, which uses the sigmoid function as response function, is the most famous candidate within the framework of Generalized Linear Models (GLMs).

The corresponding estimates $\hat{\beta}_0, \ldots, \hat{\beta}_p$ are obtained by numerical maximization of the underlying log-likelihood, e.g. by using Fisher scoring or the Newton-Raphson algorithms (see, e.g., Nelder and Wedderburn, 1972). This approach is implemented in the `glm` function from the `stats` package in R. For more details on GLMs, see Fahrmeir and Tutz (2001).

Spline-based approaches

In the classic logistic regression model introduced above, the covariates effects are strictly linear, see equation (3.1). However, in practice also non-linear influences might be relevant. In order to model these appropriately and flexibly, the GLM from above be extended to a so-called Generalized Additive Model (GAM; Wood, 2017a). For this purpose, so-called *splines* can be used. In this work, we focus on *B-splines* (see, e.g., Eilers and Marx, 2021).

So instead of linear effects like the $\beta_j x_{ij}$ in equation (3.1), with B-splines a non-linear effect $f(x)$ of a metric predictor can be represented as

$$f(x) = \sum_{j=1}^{d} \gamma_j B_j(x),$$

where $B_j(x)$ are different B-spline basis functions, d denotes the number of basis functions used, and γ_j the corresponding spline coefficients. As an unpenalized estimation of a non-linear B-spline effect often overfits, typically the non-linear effect is smoothed by using *penalized B-splines*, i.e. *P-splines*.

Beside the problem of potential overfitting, the goodness-of-fit of the B-spline approach depends on the number of selected nodes. To avoid this problem, various penalization methods exist in the form of P-splines. Here, a penalized estimation criterion, which is extended by a penalty term, is used instead of the usual estimation criterion. For P-splines based on B-splines see, e.g., Eilers and Marx (1996). For this approach, we rely on the `gam` function from the `mgcv` package (Wood, 2017b) in R. Note that such P-spline based approaches have also been used in Buhamra et al. (2024) and Buhamra et al. (2025).

Random forest

In the following, a random forest will be used for comparison with the linear and non-linear regression approaches introduced above. This method is based on a (typically

large) ensemble of trees, which were introduced by Breiman et al. (1984). However, as individual trees suffer from instability (Breiman, 1996b) an ensemble method called *bootstrapping and aggregating* (bagging; Breiman, 1996a) was introduced, which in general improves the predictive performance compared to a single regression tree and is rather easy to implement.

A *random forest* aggregates multiple decision trees to enhance prediction accuracy (Breiman, 2001). The key idea is that combining uncorrelated models (individual trees) reduces variance and improves predictions compared to using a single model. In this manuscript, classifications trees are considered in the *random forest* (where the most frequent class among the trees determines the outcome), as our target variable is binary. For this purpose, the `ranger` package in R by Wright and Ziegler (2017) is used for fitting the random forest models. Also, the two hyperparameters `mtry` and`ntree` are quite important. For optimizing `mtry`, 10-fold cross-validation is used, and `ntree` is set to 400. Further details about these parameters can be found in Buhamra et al. (2025)

3.2 Interpretable machine learning

In the following, we discuss interpretable machine learning (IML) methods such as *partial dependence plots* (PDP; see Friedman, 2001) and *individual conditional expectation* (ICE) plots (Goldstein et al., 2015) in more detail. These methods aim to enhance the interpretability of complex, black box-type machine learning models. Particularly, they can be applied to such black box models to provide explanations for individual predictions or overall model behavior. For this purpose, the implementations from the R packages `pdp` by Greenwell et al. (2017) and `ggplot2` by Wickham et al. (2016) are used (the latter one being generally applied for plotting). A nice overview of standard approaches for IML can be found in Molnar (2020).

Partial Dependence Plot (PDP)

Following Molnar (2020), the partial dependence plot (or PDP) illustrates the marginal effect of one or two features on the predicted outcome of a machine learning model (Friedman, 2001). It can reveal whether the relationship between a feature and the target is linear, monotonic, or more complex. The partial dependence function for regression is defined as follows

$$\hat{f}_S(\mathbf{x}_S) = \mathbb{E}_{\mathbf{x}_C}[\hat{f}(\mathbf{x}_S, \mathbf{x}_C)] = \int \hat{f}(\mathbf{x}_S, \mathbf{x}_C)\, dP(\mathbf{x}_C)\,,$$

where $P(\cdot)$ is the distribution of the features in set C, $\mathbf{x}_S$ are the features for which the partial dependence function is plotted, while $\mathbf{x}_C$ represents the other features used in the machine learning model $\hat{f}(\cdot)$, which are considered as random variables in this context. Typically, the set S contains only one or two features, namely those whose effect on the prediction one aims to understand. The feature vectors $\mathbf{x}_S$ and $\mathbf{x}_C$ together form the complete feature space $\mathbf{x}$. Partial dependence operates by marginalizing the model's output over the distribution of the features $P(\cdot)$ in set C, allowing the function to reveal the relationship between the features in S and the predicted outcome. By doing so, we obtain a function depending solely on the features in S, while still accounting for interactions with other features. The partial function $\hat{f}_S$ is given by

$$\hat{f}_S(\mathbf{x}_S) = \frac{1}{n}\sum_{i=1}^{n} \hat{f}(\mathbf{x}_S, \mathbf{x}_{i,C})\,,$$

i.e. it is estimated by calculating averages on the training data. The partial function shows the average marginal effect on the prediction for given values of the features in S.

Here, $\mathbf{x}_{i,C}$ denotes the actual values from the dataset for the features not of interest, and n represents the number of instances in the dataset. A key assumption of the PDP is that the features in C are uncorrelated with those in S. If this assumption does not hold, the calculated averages may include data points that are highly unlikely or even impossible. Further details can be found in Molnar (2020).

Individual Conditional Expectation (ICE)

The counterpart to a PDP (Friedman, 2001), which illustrates the average effect of a feature, is referred to as individual conditional expectation (ICE) plot and is applied for individual data instances. The approach was first introduced by Goldstein et al. (2015). ICE plots are used in machine learning to analyze the relationship between a feature and the predicted outcome for individual instances within a dataset. Unlike PDPs, which focus on the average effect of a feature, ICE plots offer insight into how changes in a specific feature or feature set impact the model's predictions for individual instances. Each line in an ICE plot represents the predicted outcome for a single instance as the feature value varies, revealing the variability and heterogeneity in feature effects across different instances. This helps identifying interactions between features, understanding complex model behaviors, and detecting outliers, thereby improving model interpretability and transparency.

3.3 Performance measures

In the following, performance measures are defined which we use to select the best model based on the predictive performance with regard to those measures (see also Buhamra et al., 2024, and Buhamra et al., 2025). Let $\tilde{y}_1, \ldots, \tilde{y}_n$ denote the true binary outcomes of a set of n matches, i.e., $\tilde{y}_i \in \{0,1\}, i = 1, \ldots, n$. Moreover, let $\hat{\pi}_{i1} =: \hat{\pi}_i$ denote the probability, predicted by a certain model, that player 1 wins match i. Then, the probability that player 2 wins the match is directly given by $\hat{\pi}_{i2} = 1 - \hat{\pi}_{i1}$.

Classification rate

The (mean) *classification rate* is given by the proportion of matches correctly predicted by a certain model, i.e.

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(\tilde{y}_i = \hat{y}_i), \text{ where } \quad \hat{y}_i = \begin{cases} 1, & \hat{\pi}_i > 0.5 \\ 0, & \hat{\pi}_i \leq 0.5 \end{cases},$$

see, e.g., Schauberger and Groll (2018). Hence, large values indicate a good predictive performance.

Predictive Bernoulli likelihood

Following again Schauberger and Groll (2018), the (mean) *predictive Bernoulli likelihood* is based on the predicted probability $\hat{\pi}_i$ for the true outcome $\tilde{y}_i$. For n observations it is defined as

$$\frac{1}{n}\sum_{i=1}^{n} \hat{\pi}_i^{\tilde{y}_i}(1-\hat{\pi}_i)^{1-\tilde{y}_i},$$

Once again, large values indicate good predictive performance.

Brier score

The *Brier score* (Brier, 1950) is based on the squared distances between the predicted probability $\hat{\pi}_i$ and the actual (binary) output $\tilde{y}_i$ from match i, and is defined as

$$\frac{1}{n}\sum_{i=1}^{n}(\hat{\pi}_i - \tilde{y}_i)^2 .$$

It is an error measure and, hence, low values indicate a good predictive performance.

4 Results

In the next section, the predictive power of the enhanced variables is presented for both regression and machine learning approaches. For this purpose, we use an expanding window (EW), a rolling window (RW) as well as a leave-one-tournament-out cross-validation (CV) approach. The best models are identified with respect to the previously defined performance measures. Additionally, we provide better interpretability of the machine learning approach, i.e. the random forest model, by using IML tools such as partial dependence plot (PDP) and individual conditional expectation (ICE). All calculations and evaluations were performed using the statistical programming software R (R Core Team, 2024).

4.1 Enhanced variables predictive power

To investigate the predictive potential of so-called 'statistically enhanced covariates' in more detail, certain promising variables are considered, such as *Elo, Age.30* and *Age.int*, along with two conventional covariates (*Rank* or *Points*). All possible combinations of these covariates result in a total of 21 models for each of our proposed approaches, including linear effects, non-linear effects (splines), and the random forest. The results are given in Tables 1 and 2, with the best performers highlighted in bold. For the expanding window approach, results are presented here in detail, as this method clearly identifies the top-performing models among the 21 proposed. The results for the leave-one-tournament-out cross-validation and rolling window approaches are included in the appendix.

Expanding window validation

We validated all proposed 21 models with respect to their predictive performance on new, unseen test data. The validation is performed using an expanding window forecasting approach, i.e., each time one of the remaining tournaments is used as the test data set in chronological order, and the training data set is constantly updated and enlarged. This scheme has already been previously used in Buhamra et al. (2025) and can be explained as follows:

1. First, all tournaments prior to 2022 are used as the training data set. Then, all models are fitted. Based on those, predictions are derived for the 2022 Australian Open matches, as this was the 1^{st} Grand Slam tournament in 2022.
2. Then, the training data is updated, by adding the matches of the Australian Open 2022. Then, the models are fitted again o the extended training data, and predictions are made for the French Open 2022, which is the 2^{nd} Grand Slam tournament in 2022.

3. Next, the matches of the French Open 2022 are added to the training data set and the models are fitted again. Based on those fits, Wimbledon 2022 is predicted.
4. Then, the Wimbledon 2022 matches will be added to the training data set, and again, the models are fitted and predictions are made for the final Grand Slam tournament in 2022, the US Open 2022.

Finally, the prediction results for all four tournaments are compared with the actual match outcomes, and the corresponding performance measures are calculated . Table 1 presents the predictive performance for the regression models with linear and non-linear effects, resulting from all 21 possible combinations of the statistically enhanced variables. Moreover, in Table 2, the same covariate combinations are presented, but with respect to the random forest method.

Linear model: The classification rates range from 0.737 to 0.795. The best performance was achieved by the linear regression model including *Points*, *Rank*, and *Elo*, with a value of 0.795. For the predictive likelihood, the best value is 0.701, achieved by the model including the covariates *Points*, *Elo* and *Age.int*. The lowest Brier score, hence the best, is 0.153 and is delivered by eight different models.

Splines model: Unlike the models with linear effects, for the non-linear models are clear winning model can be identified based on the enhanced variables *Elo* and *Age.30*. Specifically, the splines model including covariates *Elo* and *Age.30* demonstrates the best performance with respect to all proposed performance measures. This model achieved a classification rate of 0.792, and for the predictive likelihood and Brier score, the corresponding values are 0.703 and 0.149, respectively. Additionally, it should be noted that with respect to the classification rate, the model incorporating *Points*, *Elo* and *Age.30* also yielded identical results to the winning model, namely the value 0.792.

Random forest: Similar to the case of the spline-based approaches, the winning model among the random forests can be clearly identified. The results show that the random forest based on *Points*, *Rank*, *Age.30*, and *Elo* covariates outperforms all other models. It yielded a classification rate of 0.820, a predictive likelihood of 0.667, and a Brier score of 0.151.

Overall, if we considered both the proposed model types (i.e., linear regression, spline-based regression and random forests) and different forecasting strategies (such as EW, CV, and RW), certain models consistently prevail when these enhanced variables are present. The results of the leave-one-tournament-out CV approach for the regression approaches (with linear and non-linear effects), as well as for the random forest are presented in Tables 4 and 5, respectively, in the appendix. Also, the corresponding results of the rolling window approach are available in appendix (see Tables 6 and 7). In general, across all the tables presented in the appendix, the results suggest a nearly identical trend: prediction accuracy improves when at least one of these enhanced variables (i.e., *Elo*, *Age.30*, *Age.int*) is included.

Table 1: Results of the expanding window approach incorporating the statistically enhanced covariates for the regression model class with linear and non-linear effects; best results are highlighted in bold font.

	Specific models	Class. rate	Likeli-hood	Brier score
Linear	Points	0.754	0.639	0.178
	Elo	0.782	0.695	**0.153**
	Rank	0.754	0.642	0.177
	Points, Rank	0.754	0.661	0.171
	Points, Elo	0.782	0.695	**0.153**
	Rank, Elo	0.775	0.696	**0.153**
	Elo, Age.30	0.778	0.697	**0.153**
	Rank, Age.30	0.747	0.649	0.174
	Points, Age.30	0.758	0.643	0.176
	Elo, Age.int	0.775	0.696	0.154
	Rank, Age.int	0.737	0.647	0.175
	Points, Age.int	0.737	0.642	0.177
	Points, Rank, Elo	**0.795**	0.696	**0.153**
	Points, Rank, Age.30	0.758	0.666	0.168
	Points, Rank, Age.int	0.754	0.665	0.169
	Points, Elo, Age.30	0.785	0.696	**0.153**
	Elo, Rank, Age.30	0.782	0.698	**0.153**
	Points, Elo, Age.int	0.764	**0.701**	0.1570
	Elo, Rank, Age.int	0.761	0.696	0.162
	Points, Rank, Age.30, Elo	0.785	0.698	**0.153**
	Points, Rank, Age.int, Elo	0.788	0.697	0.154
Splines	Points	0.741	0.651	0.175
	Elo	0.775	0.697	0.153
	Rank	0.747	0.643	0.179
	Points, Rank	0.744	0.655	0.174
	Points, Elo	0.775	0.698	0.153
	Rank, Elo	0.778	0.696	0.156
	Elo, Age.30	**0.792**	**0.703**	**0.149**
	Rank, Age.30	0.751	0.654	0.175
	Points, Age.30	0.761	0.659	0.171
	Elo, Age.int	0.768	0.699	0.157
	Rank, Age.int	0.741	0.649	0.180
	Points, Age.int	0.734	0.655	0.178
	Points, Rank, Elo	0.778	0.697	0.158
	Points, Rank, Age.30	0.768	0.665	0.169
	Points, Rank, Age.int	0.741	0.661	0.175
	Points, Elo, Age.30	**0.792**	0.701	0.150
	Elo, Rank, Age.30	0.785	0.698	0.156
	Points, Elo, Age.int	0.764	0.701	0.157
	Elo, Rank, Age.int	0.761	0.696	0.162
	Points, Rank, Age.30, Elo	0.778	0.697	0.156
	Points, Rank, Age.int, Elo	0.761	0.697	0.163

Table 2: Results of the expanding window approach for random forest approach; best results are highlighted in bold font.

	Specific models	Class. rate	Likeli-hood	Brier score
Random forest	Points	0.773	0.596	0.189
	Elo	0.800	0.615	0.174
	Rank	0.784	0.596	0.187
	Points, Rank	0.779	0.616	0.179
	Points, Elo	0.799	0.632	0.168
	Rank, Elo	0.804	0.634	0.164
	Elo, Age.30	0.793	0.616	0.173
	Rank, Age.30	0.797	0.596	0.186
	Points, Age.30	0.778	0.598	0.187
	Elo, Age.int	0.804	0.618	0.172
	Rank, Age.int	0.793	0.595	0.187
	Points, Age.int	0.778	0.595	0.188
	Points, Rank, Elo	0.807	0.638	0.164
	Points, Rank, Age.30	0.791	0.616	0.178
	Points, Rank, Age.int	0.788	0.616	0.178
	Points, Elo, Age.30	0.779	0.631	0.172
	Elo, Rank, Age.30	0.796	0.632	0.166
	Points, Elo, Age.int	0.808	0.632	0.162
	Elo, Rank, Age.int	0.794	0.634	0.165
	Points, Rank, Age.30, Elo	**0.820**	**0.667**	**0.151**
	Points, Rank, Age.int, Elo	0.800	0.637	0.166

4.2 Model interpretation

In this section, our interpretation is developed based on the refitting of the top-performing linear, spline, and random forest models introduced in Section 4.1. Consequently, we provide deeper insights for these models. Additionally, interpretable machine learning tools such as partial dependence plot (PDP) and individual conditional expectation (ICE) are used to enhance our understanding of the random forest model.

4.2.1 Linear model

The linear model can be directly interpreted based on the p estimated regression coefficients $\hat{\beta}_j$ from Equation 3.1. Their values indicate how much the outcome variable is expected to change when the corresponding predictor variable changes by one unit, assuming all other covariates in the model are held constant. Hence, they allow direct interpretation of both strength and direction of the relationship between the predictors and the outcome variable. Table 3 shows the coefficient estimates for the best candidate linear model based on the results from the previous Section 4.1. It incorporates *Points*, *Rank* as conventional covariates, and *Elo* as enhanced covariate.

Table 3: Estimated coefficients for the candidate linear model based on the results from Table 1

Variables	$\hat{\beta}_j$
Rank	-0.0012
Elo	0.0042
Points	0.0001

For the *Rank* difference variable, the negative sign indicates that as this variable increases (resulting in a larger numerical rank, which corresponds to a lower-ranked first-named player), the linear predictor decreases by approximately 0.0012 units. Hence, due to the the used (logistic) response function, also the winning probability for the respective player decreases.

Analogously, for every one-unit increase in the *Elo* rating difference, the linear predictor exhibits an increase of approximately 0.0042 units. Hence, also the winning probability for the respective first-named player increase. In practical terms, this means that larger *Elo* ratings are associated with larger probabilities of winning a tennis match, as the positive sign of the coefficient estimate indicates a positive relationship.

The same positive relationship also holds for the predictor variable *Points*. For every one-unit increase in the difference in points between players (again, from the perspective of the first-named player), the model predicts an increase of approximately 0.0001 units in the linear predictor and, hence, also an increase of the respective player's winning probability.

4.2.2 Splines

Graphical illustrations of spline effects are a useful way to understand and visualize potential non-linear effects of predictors on the response variable. The spline-based approaches from Section 3.1 are able to capture non-linear relationships and allow for a flexible, semi-parametric form of regression using smooth functions (splines) for predictors. Corresponding ploting functions provide interpretable visualizations. In this context, the `gam` function from the `mgcv` package (Wood, 2017a) in R is used.

Figure 1 displays the effects for the covariates *Elo* and *Age.30*, which actually appear to be linear. The graphs show positive effects for both features, i.e. larger *Elo* ratings associated with larger winning probabilities. The *Age.30* difference has a slightly positive effect, which however is not significant taking the point-wise confidence band into account.

4.2.3 Random forest with interpretable machine learning

Next, the results for the random forest model with the best predictive performance from Section 4.1 are discussed. This model, which included *Points*, *Rank*, *Age.30*, and *Elo* as covariates, is refitted to our entire data set. Then, both partial dependence and Individual Conditional Expectation (ICE) plots are presented for better interpretability and visualization of the respective complex model.

Since ICE plots—the individual equivalent of Partial Dependence Plots (PDPs)—are considered crucial in our case, the combined ICE and PDP plots were used to provide a comprehensive visualization of the relationship between predictor variables and the predicted outcome.

In Figure 2 we generated a combined ICE and PDP. Each ICE line represents how the predicted winning probability changes with the predictor value for a single observation. The spread of the lines indicates variability in the effect of the predictor across different observations. The thick black line represents the PDP, showing the average effect of the respective predictor on the predicted outcome. This PDP line provides context to the ICE lines by highlighting the overall trend across all observations.

For instance, in the *Rank* panel, we can observe how the predicted outcome is first very strongly decreasing and then slowly increasing. Moreover, in the *Rank* panel, the spread of grey lines illustrates variability in how an increasing *Rank* difference affects different observations. After a steep decrease, the general trend of the black line indicates a almost constant, very slightly increasing relationship between *Rank* and the

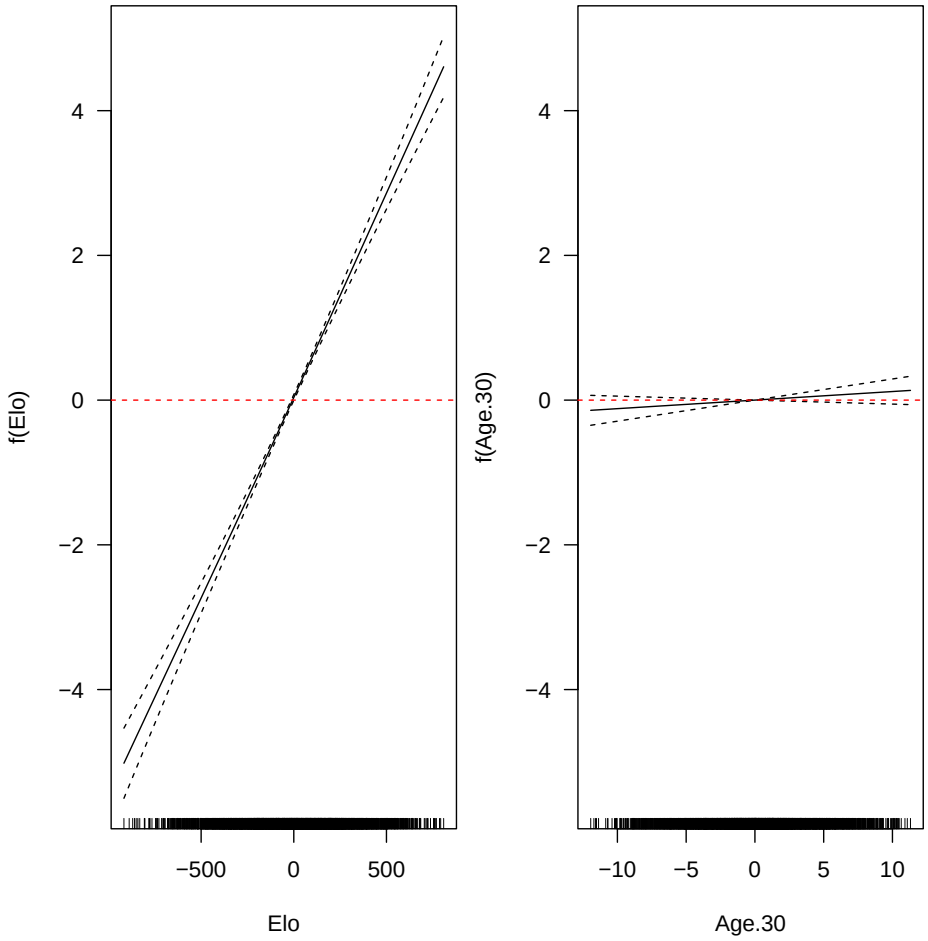

Figure 1: Spline effects for the enhanced covariates *Elo* and *Age.30*

predicted probability of winning. This means that as the *Rank* difference increases (i.e., the first-named player is ranked higher, meaning worse performance, when the rank of the second player is kept constant), the winning probability first strongly decreases, then increases a bit and finally remains almost constant, before it exhibits a small hill at the end.

In the *Elo* panel, the PDP line shows a slightly negative trend, suggesting that larger *Elo* differences first decrease the predicted probability of winning, before increasing again. For the *Points*, upward slope in the PDP suggests that accumulating more points correlates positively with increased chances of winning matches. Variability among ICE lines indicate that while most players benefit from accumulating points, some may experience diminishing returns at higher point levels or face challenges against specific opponents. The *Age.30* panel shows significant variability in individual responses, as indicated by the spread of the ICE lines. The ICE lines highlight differences among individual responses: some (first-named) players with an age closer to the optimal one of 30 years outperform the (second-named) player substantially. In addition, we can see a slightly decreasing trend of the *Age.30* difference, which then seems to then remain rather constant for small or positive differences.

Overall, this combined plot effectively conveys both the average influence of each predictor (PDP) and its varying impact on individual observations (ICE). It allows for precise interpretation of how predictors influence the outcome across different levels and individuals.

Finally, in Figure 3, a heatmap plot for the partial dependency of *Elo* and *Age.30* is

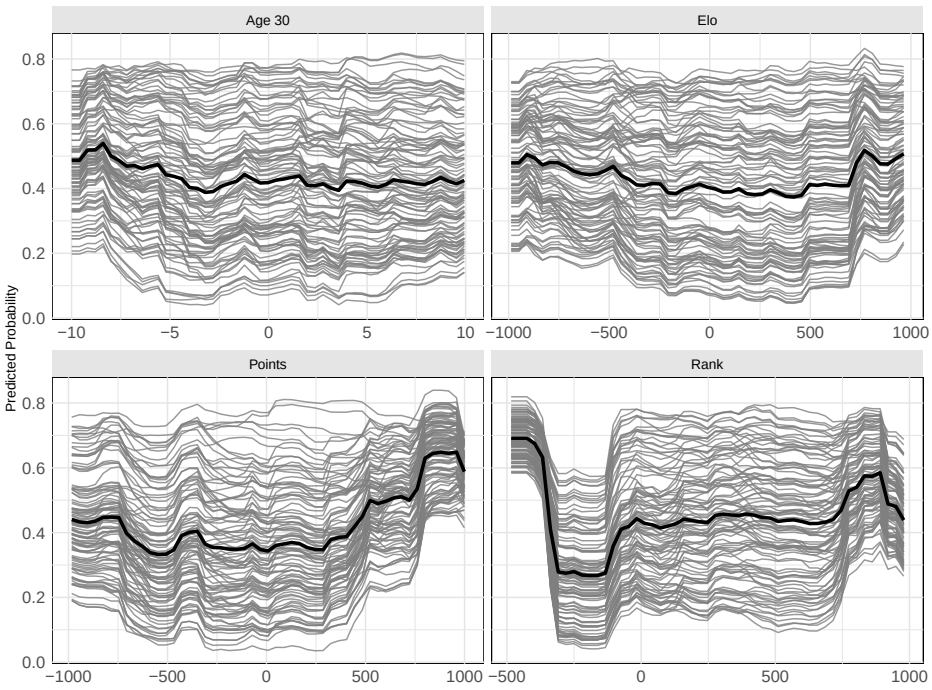

Figure 2: Combined PDP (thick black line) and ICE plots (grey lines) for the random forest model including covariates *Age.30*, *Elo*, *Points* and *Rank*

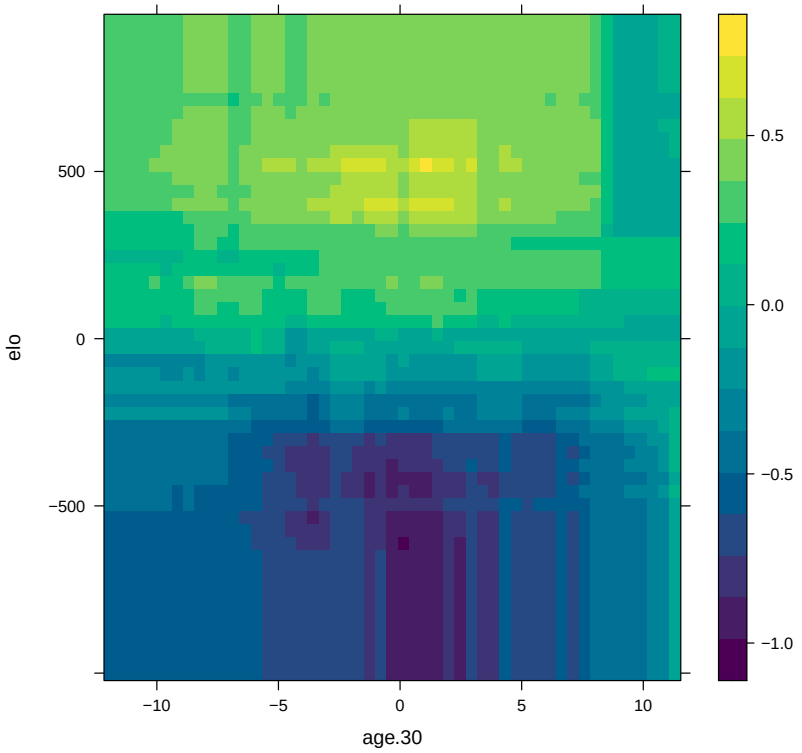

Figure 3: Heat map for the random forest model including *Elo* and *Age.30* as covariates

presented. The plot examines how the linear predictor (and, hence, the wining probability) changes jointly with *Age.30* and *Elo*. If the color changes are not uniform across the heat map, this suggests a complex interaction between *Elo* and *Age.30*. For exam-

ple, larger *Elo* have a stronger impact on the predictor for certain age ranges than for others (e.g., players with an age closer to 30 benefit more from a larger Elo ratings than younger or older players).

For example, a region that transitions from blue to yellow as *Elo* increases indicates that higher *Elo* scores are associated with higher winning probabilities, especially when *Age.30* is within a certain range.

5 Summary and overview

In this work, we compared different model approaches for modeling tennis matches in Grand Slam tournaments focusing on two main aspects: statistically enhanced covariates and interpretable machine learning tools. First, we demonstrated how these enhanced covariates can be applied in the context of sports analytics to improve predictive modeling performance and gain insights from sports data. Then, to better understand the interpretation of complex ML models, exemplarily for a random forest, we presented partial dependence plots (PDPs) to visualize the average partial relationship between the predicted response and one or more features, along with individual conditional expectation (ICE) plots, a tool for visualizing the model estimated by any supervised learning algorithm. For this purpose, we used a data set provided and analyzed in Buhamra et al. (2024, 2025), which was compiled based on the R package `deuce` (Kovalchik, 2019). It contains information on 5,013 matches in 47 men's Grand Slam tournaments from the years 2011-2022. It also includes covariate information on the age difference of both players (*Age*), the difference in their ranking positions (*Rank*) and ranking points (*Points*), in Elo numbers (*Elo*), as well as the two additional age-based variables, *Age.30* and *Age.int*, which were constructed such that they take into account that the optimal age of a tennis player is between 28 and 32 years.

Different regression models, which were already considered in Buhamra et al. (2024, 2025), were compared with machine learning approaches, in particular a random forest model, for modeling and predicting tennis matches. Since there are only two possible outcomes in tennis (win or loss), all models were based on a binary outcome and thus focused on modeling the probability of the first-named player winning.

The different modeling approaches were compared with respect to their prediction performance on unseen matches via an expanding window strategy. The following regression and ML approaches were included in this comparison:

- Logistic regression with linear effects: all possible combinations of the three enhanced covariates along with the conventional variables *Point* and *Rank* were considered. This resulted in 21 models.
- Logistic regression with non-linear effects (splines): Again, the same 21 combinations of enhanced and conventional covariates were considered.
- A random forest model as a machine learning approach: Also here, the same combinations of enhanced and conventional covariates were considered, resulting in 21 models.

Via the expanding window approach, the models were compared in terms of classification rate, predictive Bernoulli likelihood and Brier score. Since each approach resulted in 21 different models, the model with the best predictive performance measures was selected in each case. Overall, the values vary between the different approaches and over proposed performance measures. The random forest model, including the covariates *Points*, *Rank*, *Age.30*, and *Elo* achieved the best classification rate among all other models with a value of 0.8202. In contrast, the spline-based regression model based on

Elo and *Age.30* only yielded the best predictive performance in terms of predictive likelihood and Brier score compared to all other model approaches. Generally, one could say that models consistently perform better when at least one of the enhanced variables is included.

Additionally, we investigated a CV-type approach and a rolling window approach. The rolling window approached was based on a (varying) training dataset always containing 12 tournaments that were used to predict the outcome of the next tournament. The results for these two approaches are provided in the appendix. Principally, results are varying among the three different training-test-subdivision approaches, but generally led to similar results as the expanding window strategy. For example, the winning random forest model of the rolling window approach is similar to the one of the expanding window strategy, as both include *Points*, *Age.30*, *Elo*, and *Rank* as the best model predictors.

To gain a comprehensive understanding and interpretation of each approach, we analyze the coefficients of the linear regression model. Additionally, we employ spline graphs to visually represent and interpret the relationship between predictor variables and the response variable within the context of Generalized Additive Models (GAMs). These graphs can capture non-linear relationships, which is particularly advantageous for complex datasets where linear models may be insufficient. Furthermore, we introduce interpretable machine learning (IML) tools such as partial dependence plots (PDP) and individual conditional expectation (ICE) plots. These tools help in comprehending and interpreting the predictions made by complex (black box-type) machine learning models, in the present case a random forest model.

By examining PDP plots, we obtained insights into how each predictor variable affects the model's predictions. For instance, the PDP for *Points* exhibits a general upward trend, indicating that earning more *Points* has a positive effect on the predicted outcome. Similarly, the PDP for *Age.30* suggests that being closer to the optimal age of 30 years is only slightly associated with larger winning probabilities.

In addition, a heat map visualization illustrating the joint effect of the two covariates *Elo* and *Age.30* on the predicted outcome is presented. By examining the color gradient and regions in the heat map, we can infer how changes in these two covariates influence the outcome, highlighting any non-linear interactions and dependencies between those.

In future research, additional IML methods, such as Accumulated Local Effects (ALE) plots (Apley and Zhu, 2020) and Local Interpretable Model-agnostic Explanations (LIME; Ribeiro et al., 2016) can also be used. Furthermore, one could investigate more complex machine learning models, such as deep learning approaches (Bishop, 1995; LeCun et al., 2015). Additionally, similar to soccer (Groll et al., 2019b), one could also focus on tournament outcomes. For example, the probability of a certain player winning the tournament could be determined. This approach takes advantage of the fact that the tournament bracket is fully drawn before the start, allowing us to predict the earliest round in which two players could meet. Unlike in soccer, where group stage outcomes influence subsequent matches, this setup simplifies predictions. However, using only the match-specific betting odds for the first round presents a challenge. Models that exclude odds as covariates might be preferable, despite potentially lower prediction performance due to the significant influence of odds. Alternatively, models could be developed that use pre-tournament odds for each player to win the entire tournament, rather than odds for individual matches. Finally, additional statistically enhanced covariates could be produced. For example, similar to the historic match abilities for soccer teams developed by Ley et al. (2019), such abilities could also be developed for tennis players. We are currently working on such an approach and plan to include those ability parameters into our models in the future.

References

Angelini, G., Candila, V., and De Angelis, L. (2022). Weighted Elo rating for tennis match predictions. *European Journal of Operational Research*, 297(1):120–132.

Apley, D. W. and Zhu, J. (2020). Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 82(4):1059–1086.

Auret, L. and Aldrich, C. (2012). Interpretation of nonlinear relationships between process variables by use of random forests. *Minerals Engineering*, 35:27–42.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Breiman, L. (1996a). Bagging predictors. *Machine learning*, 24:123–140.

Breiman, L. (1996b). Heuristics of instability and stabilization in model selection. *The annals of statistics*, 24(6):2350–2383.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). Classification and regression trees. wadsworth int. *Group*, 37(15):237–251.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78:1–3.

Buhamra, N., Groll, A., and Brunner, S. (2024). Modeling and prediction of tennis matches at grand slam tournaments. *Journal of Sports Analytics*, 10(1):17–33.

Buhamra, N., Groll, A., and Gerharz, A. (2025). Comparing modern machine learning approaches for modeling tennis matches at grand slam tournaments. *Journal of Sports Analytics*. under review.

Eilers, P. H. and Marx, B. D. (2021). *Practical smoothing: The joys of P-splines*. Cambridge University Press.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, 11:89–121.

Fahrmeir, L. and Tutz, G. (2001). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer-Verlag, New York, 2nd edition.

Felice, F., Ley, C., Groll, A., and Bordas, S. (2023). Statistically enhanced learning: a feature engineering framework to boost (any) learning algorithms. *arXiv preprint arXiv:2306.17006*.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Gao, Z. and Kowalczyk, A. (2021). Random forest model identifies serve strength as a key predictor of tennis match outcome. *Journal of Sports Analytics*, 7(4):255–262.

Goldstein, A., Kapelner, A., Bleich, J., and Pitkin, E. (2015). Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *journal of Computational and Graphical Statistics*, 24(1):44–65.

Greenwell, B. M. et al. (2017). pdp: An r package for constructing partial dependence plots. *R J.*, 9(1):421.

Groll, A., Ley, C., Schauberger, G., and Van Eetvelde, H. (2019a). A hybrid random forest to predict soccer matches in international tournaments. *Journal of quantitative analysis in sports*, 15(4):271–287.

Groll, A., Ley, C., Schauberger, G., and Van Eetvelde, H. (2019b). A hybrid random forest to predict soccer matches in international tournaments. *Journal of Quantitative Analysis in Sports*, 15:271–287.

Kovalchik, S. (2019). deuce: resources for analysis of professional tennis data. *R package version 1.4*.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

Ley, C., Wiele, T. V. d., and Eetvelde, H. V. (2019). Ranking soccer teams on the basis of their current strength: A comparison of maximum likelihood approaches. *Statistical Modelling*, 19(1):55–73.

Molnar, C. (2020). *Interpretable machine learning*. Lulu. com.

Molnar, C., Freiesleben, T., König, G., Herbinger, J., Reisinger, T., Casalicchio, G., Wright, M. N., and Bischl, B. (2023). Relating the partial dependence plot and permutation feature importance to the data generating process. In *World Conference on Explainable Artificial Intelligence*, pages 456–479. Springer.

Nelder, J. A. and Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society*, A 135:370–384.

R Core Team (2024). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Schauberger, G. and Groll, A. (2018). Predicting matches in international football tournaments with random forests. *Statistical Modelling*, 18(5-6):460–482.

Sipko, M. and Knottenbelt, W. (2015). Machine learning for the prediction of professional tennis matches. *MEng computing-final year project, Imperial College London*, 2.

Somboonphokkaphan, A., Phimoltares, S., and Lursinsap, C. (2009). Tennis winner prediction based on time-series history with neural modeling. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1, pages 18–20. Citeseer.

Vaughan Williams, L., Liu, C., Dixon, L., and Gerrard, H. (2021). How well do Elo-based ratings predict professional tennis matches? *Journal of Quantitative Analysis in Sports*, 17(2):91–105.

Weston, D. (2014). Using age statistics to gain a tennis betting edge. http://www.pinnacle.com/en/betting-articles/Tennis/atp-players-tipping-point/LMPJF7BY7BKR2EY.

Whiteside, D., Cant, O., Connolly, M., and Reid, M. (2017). Monitoring hitting load in tennis using inertial sensors and machine learning. *International journal of sports physiology and performance*, 12(9):1212–1217.

Wickham, H., Chang, W., and Wickham, M. H. (2016). Package 'ggplot2'. *Create elegant data visualisations using the grammar of graphics. Version*, 2(1):1–189.

Wilkens, S. (2021). Sports prediction and betting models in the machine learning age: The case of tennis. *Journal of Sports Analytics*, 7(2):99–117.

Wood, S. N. (2017a). *Generalized additive models: an introduction with R*. chapman and hall/CRC.

Wood, S. N. (2017b). *Generalized Additive Models: An Introduction with R*. Chapman & Hall/CRC, London, 2nd edition.

Wright, M. N. and Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, 77(1):1–17.

A Appendix

Table 4: Results of the leave-one-tournament-out CV approach for regression model class; best results in bold font.

	Specific models	Class. rate	Likeli-hood	Brier score
Linear	Points	0.732	0.623	0.185
	Elo	0.747	0.656	0.172
	Rank	0.732	0.592	0.199
	Points, Rank	0.732	0.634	0.181
	Points, Elo	0.748	0.658	0.171
	Rank, Elo	**0.749**	0.657	0.172
	Elo, Age.30	0.747	0.656	0.172
	Rank, Age.30	0.731	0.592	0.199
	Points, Age.30	0.729	0.623	0.185
	Elo, Age.int	0.747	0.656	0.172
	Rank, Age.int	0.731	0.592	0.199
	Points, Age.int	0.731	0.623	0.185
	Points, Rank, Elo	0.747	0.659	**0.170**
	Points, Rank, Age.30	0.732	0.634	0.181
	Points, Rank, Age.int	0.733	0.634	0.181
	Points, Elo, Age.30	0.747	0.658	0.171
	Elo, Rank, Age.30	0.747	0.657	0.172
	Points, Elo, Age.int	0.745	0.657	0.171
	Elo, Rank, Age.int	0.746	**0.656**	0.172
	Points, Rank, Age.30, Elo	0.747	**0.659**	**0.170**
	Points, Rank, Age.int, Elo	0.748	**0.659**	**0.170**
Splines	Points	0.729	0.637	0.181
	Elo	0.746	0.656	0.172
	Rank	0.729	0.611	0.193
	Points, Rank	0.732	0.641	0.179
	Points, Elo	0.748	**0.657**	**0.171**
	Rank, Elo	**0.748**	0.656	0.172
	Elo, Age.30	0.746	0.656	0.172
	Rank, Age.30	0.729	0.611	0.193
	Points, Age.30	0.729	0.637	0.181
	Elo, Age.int	0.745	0.656	0.172
	Rank, Age.int	0.728	0.611	0.194
	Points, Age.int	0.729	0.637	0.181
	Points, Rank, Elo	0.747	**0.657**	**0.171**
	Points, Rank, Age.30	0.731	0.641	0.179
	Points, Rank, Age.int	0.730	0.641	0.179
	Points, Elo, Age.30	0.746	**0.657**	**0.171**
	Elo, Rank, Age.30	0.746	0.656	0.172
	Points, Elo, Age.int	0.745	**0.657**	**0.171**
	Elo, Rank, Age.int	0.746	0.656	0.172
	Points, Rank, Age.30, Elo	0.745	**0.657**	**0.171**
	Points, Rank, Age.int, Elo	0.744	**0.657**	**0.171**

Table 5: Results of the leave-one-tournament-out CV approach for random forest models; best results in bold font.

	Specific models	Class. rate	Likeli-hood	Brier score
Random forest	Points	0.757	0.611	0.186
	Elo	0.769	0.623	0.179
	Rank	0.757	0.604	0.190
	Points, Rank	0.759	0.629	0.181
	Points, Elo	0.772	0.641	**0.174**
	Rank, Elo	**0.773**	0.638	0.175
	Elo, Age.30	0.771	0.624	0.179
	Rank, Age.30	0.751	0.615	0.190
	Points, Age.30	0.755	0.608	0.187
	Elo, Age.int	0.769	0.620	0.179
	Rank, Age.int	0.753	0.606	0.191
	Points, Age.int	0.754	0.608	0.187
	Points, Rank, Elo	0.772	**0.645**	**0.174**
	Points, Rank, Age.30	0.758	0.626	0.182
	Points, Rank, Age.int	0.760	0.626	0.182
	Points, Elo, Age.30	0.769	0.638	0.175
	Elo, Rank, Age.30	0.772	0.635	0.176
	Points, Elo, Age.int	0.770	0.638	0.176
	Elo, Rank, Age.int	0.770	0.636	0.176
	Points, Rank, Age.30, Elo	0.770	0.643	**0.174**
	Points, Rank, Age.int, Elo	0.770	0.643	0.175

Table 6: Results of the rolling window approach for regression model class; best results in bold font.

	Specific models	Class. rate	Likeli-hood	Brier score
Linear	Points	0.645	0.498	0.309
	Elo	0.630	0.496	0.330
	Rank	0.645	**0.501**	**0.293**
	Points, Rank	0.645	0.499	0.317
	Points, Elo	0.633	0.496	0.332
	Rank, Elo	0.633	0.497	0.331
	Elo, Age.30	0.627	0.495	0.331
	Rank, Age.30	0.641	**0.501**	0.294
	Points, Age.30	0.642	0.498	0.309
	Elo, Age.int	0.631	0.495	0.331
	Rank, Age.int	0.644	**0.501**	0.294
	Points, Age.int	0.646	0.498	0.309
	Points, Rank, Elo	0.636	0.497	0.332
	Points, Rank, Age.30	0.644	0.499	0.317
	Points, Rank, Age.int	**0.647**	0.499	0.317
	Points, Elo, Age.30	0.628	0.495	0.332
	Elo, Rank, Age.30	0.632	0.496	0.331
	Points, Elo, Age.int	0.631	0.495	0.332
	Elo, Rank, Age.int	0.635	0.496	0.331
	Points, Rank, Age.30, Elo	0.633	0.496	0.332
	Points, Rank, Age.int, Elo	0.634	0.496	0.333
Splines	Points	**0.648**	0.500	0.319
	Elo	0.634	0.496	0.331
	Rank	0.643	**0.501**	**0.306**
	Points, Rank	0.646	0.499	0.323
	Points, Elo	0.633	0.496	0.332
	Rank, Elo	0.634	0.497	0.330
	Elo, Age.30	0.633	0.496	0.331
	Rank, Age.30	0.645	**0.501**	**0.306**
	Points, Age.30	0.644	0.499	0.320
	Elo, Age.int	0.632	0.496	0.331
	Rank, Age.int	0.644	**0.501**	**0.306**
	Points, Age.int	0.646	0.499	0.319
	Points, Rank, Elo	0.633	0.497	0.332
	Points, Rank, Age.30	0.643	0.499	0.323
	Points, Rank, Age.int	0.644	0.499	0.323
	Points, Elo, Age.30	0.630	0.496	0.332
	Elo, Rank, Age.30	0.634	0.496	0.331
	Points, Elo, Age.int	0.633	0.496	0.333
	Elo, Rank, Age.int	0.632	0.496	0.331
	Points, Rank, Age.30, Elo	0.632	0.496	0.332
	Points, Rank, Age.int, Elo	0.632	0.497	0.332

Table 7: Results of the rolling window approach for random forest models; best results in bold font.

	Specific models	Class. rate	Likeli-hood	Brier score
	Points	0.753	0.600	0.191
	Elo	0.771	0.611	0.183
	Rank	0.755	0.601	0.194
	Points, Rank	0.756	0.620	0.185
	Points, Elo	0.771	0.628	0.177
	Rank, Elo	0.773	0.627	0.179
	Elo, Age.30	0.771	0.610	0.184
	Rank, Age.30	0.748	0.600	0.195
	Points, Age.30	0.754	0.596	0.192
	Elo, Age.int	0.771	0.609	0.184
Random forest	Rank, Age.int	0.752	0.600	0.195
	Points, Age.int	0.755	0.599	0.192
	Points, Rank, Elo	0.773	0.636	0.177
	Points, Rank, Age.30	0.757	0.618	0.186
	Points, Rank, Age.int	0.755	0.616	0.186
	Points, Elo, Age.30	0.768	0.626	0.179
	Elo, Rank, Age.30	0.774	0.626	0.179
	Points, Elo, Age.int	0.765	0.625	0.179
	Elo, Rank, Age.int	0.769	0.625	0.179
	Points, Rank, Age.30, Elo	**0.789**	**0.659**	**0.165**
	Points, Rank, Age.int, Elo	0.771	0.634	0.177

Name: Nourah Buhamra Matrikelnummer: 227227

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Dissertation mit dem Titel

"Classic statistical and modern machine learning methods for modeling and prediction of major tennis tournament"

selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet sowie die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht habe und die Satzung der Technischen Universität Dortmund zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe. Ich versichere außerdem, dass ich die beigefügte Dissertation nur in diesem und keinem anderen Promotionsverfahren eingereicht habe und dass diesem Promotionsverfahren keine endgültig gescheiterten Promotionsverfahren vorausgegangen sind. Ferner erkläre ich, dass keine Aberkennung eines bereits erworbenen Doktorgrades vorliegt. Ich versichere an Eides statt, dass ich nach bestem Wissen die reine Wahrheit erklärt und nichts verschwiegen habe.

Dortmund, den .....

Nourah Buhamra