UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

DOTTORATO DI RICERCA IN
Modelli e Metodi per l'Economia e il Management
(Analytics for Economics and Management - AEM)

settore scientifico disciplinare
SECS-S/01 - Statistica

CICLO XXXIV

Statistical models and algorithms for data mining and machine learning - Applications to sports analytics and bibliometrics.

Manlio Migliorati

RELATORE:
Prof. Marica Manisera, Università degli studi di Brescia

CO-RELATORE:
Prof. Paola Zuccolotto, Università degli studi di Brescia

This page intentionally left blank.

To Cina.
To Cinina and Cininina, getting better every day.
And to Cinino, of course.
But, first of all and definitively, to Cina.

This page intentionally left blank.

Yesterday is history.
Tomorrow is a mystery.
But today is a gift, that is why it is called present.

Kung Fu Panda

And what is good, Phædrus,
And what is not good,
Need we ask anyone to tell us these things?

Zen and the Art of Motorcycle Maintenance

This page intentionally left blank.

Acknowledgments

I wish to express appreciation and gratitude to my supervisor, Marica Manisera, who helped and guided me throughout this PhD journey: grazie, Marica. Moreover, I would like to thank my secondary supervisor Paola Zuccolotto, that introduced me not only to the statistical learning, but also to the basketball interpretation. I wish to express my gratitude to Maurizio Carpita, for his advice and for the possibility he gave me to work together. I would like to thank Eugenio Brentari for helping me in finalizing this project, and Silvia Golia that was the first that introduced me to statistics.

At last, I would like to thank my family for their ∞ patience during this period.

This page intentionally left blank.

Abstract

This PhD dissertation summarizes research questions and related answers achieved in the context of the PhD program of “Analytics for Economics and Management” of University of Brescia, Department of Economics and Management, attended by the author in the years 2018-2021. Main research activity is related to applying data science methodologies, and machine learning in particular, to sport, focusing on basketball. Intended goals are both proposing tools to help coaching staff in matches’ preparation, and facing the classification problem of outcome prediction of a match. The second research activity is focused on natural language processing applied to the bibliometric field, to offer valuable tools in finding and evaluating papers in a specific research domain (e.g. data science and sport, or COVID-19). The set of techniques includes, among others, Decision Trees, Random Forests, Recursive Partitioning, Deep Learning, Topic Models. All implementations are written in R language, and source code is available on demand.

This page intentionally left blank.

Abstract *(in Italian)*

Questa tesi di dottorato riassume gli obiettivi di ricerca e le relative risposte ottenute nell'ambito del programma di dottorato "Modelli e Metodi per l'Economia e il Management" dell'Università di Brescia, Dipartimento di Economia e Management, cui l'autore ha partecipato negli anni 2018-2021. La principale attività di ricerca è relativa all'applicazione delle metodologie di data science, e in particolare il machine learning, allo sport, focalizzandosi sul basket, col duplice scopo di fornire allo staff tecnico strumenti per preparare e analizzare un match, e di affrontare il problema di classificazione della previsione del risultato di una partita. La seconda attività di ricerca affrontata è focalizzata sull'utilizzo del linguaggio naturale nel settore bibliometrico, al fine di offrire strumenti utili alla individuazione e valutazione di articoli rilevanti per uno specifico contesto scientifico (ad esempio data science e sport, oppure COVID-19). L'insieme di tecniche utilizzate include, tra le altre, gli Alberi di Decisione, le Random Forests, il Recursive Partitioning, il Deep Learning, i Topic Models. Tutte le implementazioni sono scritte nel linguaggio R, e il codice sorgente è disponibile su richiesta.

This page intentionally left blank.

Contents

This page intentionally left blank.

List of Figures

List of Tables

This page intentionally left blank.

Listings

This page intentionally left blank.

Chapter 1

Summary

This PhD dissertation summarizes the research questions and related answers obtained attending the PhD program of "Analytics for Economics and Management" of the University of Brescia, Department of Economics and Management.
The work is focused on the application of machine learning algorithms to the domain of sport, and in particular to basketball, with the goal of answering two main research questions:

1. Oliver's Four Factors (Oliver, 2004; Kubatko et al., 2007) are a small set of statistics concentrating a lot of information, and are considered the keys to success for a basketball match. How these statistics can be used in helping technical staff of National Basketball Association (NBA) championship teams in analysing and preparing matches?

2. NBA is the most important basketball championship in the world, with a total revenue in the last years of about 8 billions U.S. dollars (Gough, 2021). As a consequence, it is not surprising there are several attempts in facing the problem of outcome predictions with machine learning. These models use, in general, a huge number of inputs to have an accuracy index about 67%. Is it possible to find alternative approaches to outcome prediction, offering the same accuracy but using a really low number (also just one) of regressors?

The structure of this dissertation is as follows:
Chapter 2 contains a review of the scientific literature related to our research questions. Literature about data science applied to sport in general, and to basketball in particular, is analyzed, and more relevant references are summarized. Scientific literature analysis has been also approached via Natural

Language Processing (NLP), opening new research questions whose first results are reported in Appendix A.
Chapter 2 also includes some sections of general interest (NBA organization, approach to dataset definition, Oliver's Four Factors formal definition), useful in approaching other chapters.
At last, a summary of machine learning techniques used in this dissertation is reported.

Chapter 3 aims to identify the drivers leading to a victory in NBA matches. To do this, the prediction results from machine learning models using two different sets of information as independent variable are compared:

1. the so called *box score* analytics, i.e. the classical information (attempted shots, made shots, rebounds ...) summarizing a match;

2. the already mentioned Oliver's Four Factors.

Box scores and Four Factors are used as regressors in machine learning algorithms, to predict the winner of the matches involving the Golden State Warriors team, on a dataset containing data of regular seasons from 2004-2005 to 2017-2018. Outcome prediction is not our goal, but it is the tool used to identify the most important success keys via variable importance measurement: as data are ex-post (i.e. they also includes information about the match under analysis), accuracy is very high, and so variable importance measures from fitted models are affordable for success drivers identification.
The Random Forests algorithm in general offers good accuracy, but unfortunately few interpretability tools (only variable importance measures); so, to these purposes, Random Forests models are coupled to models fitted using CART (after an assessment of their high accuracy), offering both variable importance and great interpretability. That is an example of application of the approach called "Global Surrogate" in Interpretable Artificial Intelligence (Molnar, 2021), when a different machine learning algorithm is used to improve interpretability for a black box (or in any case less interpretable) machine learning method.
That analysis shows how, for Golden State Warriors, defense is the key factor to win a game.
At last, the fitted models are shown to be both suitable in game preparation, and affordable on basketball court too, for supporting the coaching staff decisions.

Chapter 4 includes an analysis about the weight of the Oliver's Four Factors as success keys in NBA matches. The MOdel-Based recursive partitioning (MOB) algorithm is applied to a dataset including 19138 matches

of 16 NBA regular seasons (from 2004-2005 to 2019-2020). MOB, instead of fitting one global Generalized Linear Model (GLM) to all observations, partitions the observations according to selected variables, and estimates several ad hoc local GLMs for subgroups of observations.
Using the difference in victories ratio between two teams (named $diff$) as partitioning variable, 2 models have been produced to rank the importance of Four Factors, for home and away played matches, respectively.
The partitioning variable has been calculated both ex post and ex ante (i.e. considering or not, respectively information about the match under analysis), and both separating home and away data or not; the best model is produced with ex ante calculation, without considering home/away data separation.
This work proposes two innovative topics:

1. in some circumstances, using particular partitioning variables, (quasi) separation situations are generated. To solve it, a methodological extension of GLM-based recursive partitioning, from standard Maximum Likelihood (ML) estimation to bias-reduced (BR) estimation, has been identified and implemented. In this way it is possible to solve (quasi) separation convergence problems in the numerical solution of ML estimation in MOB.

2. BR-based GLM trees are applied to basketball analytics; fitted models produced using BR-based MOB, different for home and away matches, are shown to be easily interpretable, constituting a useful tool in supporting the coaching staff's job.

In chapter 5, machine learning models fitted on the basis of single features are shown to produce accurate outcome predictions for NBA matches, exhibiting the same top accuracy (about 67%) found in literature for quite more complex set of features.
Features based on:

- Elo, the algorithm born for rating chess players (Elo, 1978);

- the difference in relative victories frequencies between two teams used for the analysis described in Chapter 4);

- The Oliver's Four Factors;

are ex ante calculated (i.e. only considering data before of the match under exam) for the dataset containing the data of 16 NBA regular seasons. The first two approaches are based on teams strength definition, the third one

summarizes in four indexes the box score statistics.
Particular attention has been payed to:

- temporal aspects, considering both historical (i.e. all prior games are considered, with regression to mean) and dynamic (considering rolling mean features on some prior matches) approaches.
- To home court factor, calculating features both with and without considering in a separated way home and away matches.

Prediction models are produced on that dataset via Deep Learning, a specific subfield of machine learning based on neural networks composed by several layers (that's why it is called *deep*, without any reference to a bigger level of problem comprehension), using k-fold validation.
Results are compared to accuracies obtained from applying boosting algorithms, both for validating output and for having variable importance, following the same "global surrogate" approach mentioned above for chapter 3. Best accuracy for predictions using just one feature is shown to be 0.6734 on the complete dataset, and 0.7063 on a single season, aligned to accuracy found in literature for outcome prediction models using many more regressors.

Appendix A contains some results obtained in applying NLP tools to the analysis of scientific literature. Following the classic approach of research, the first PhD activity is the analysis of the scientific literature related to research questions, trying to identify papers relevant to our domain.
To this purpose, a corpus of domain relevant publications selected using Web Of Science (WoS) service is analysed using `Bibliometrix` (Aria and Cuccurullo, 2017), an R open-source tool for quantitative research in scientometrics and bibliometrics, to retrieve pertinent papers information about:

- authors and groups;
- nations;
- years;
- milestones and references;
- approaches and data science models used in this domain.

This activity has evolved in a new research field, focused on bibliographic research via NLP tools, and the first results of this ongoing job are presented in Appendix A. In particular:

1. the first Appendix Section recaps bibliographic research activities strictly related to machine learning applied to basketball;

2. the second Appendix Section shows how, on a corpus of COVID-19 papers built using both bibliographics and ad hoc NLP tools implemented in R, it is possible to conduct an analysis about authors, institutions and journals using unsupervised papers' clustering where clusters are identified on the basis of paper contents;

3. the third Appendix Section describes a possible approach to address the problem of attributing a value to papers posted in not peer-reviewed archives (e.g. arXive), trying to help in joining dichotomy between contents reliability and publishing lag time. A new index, derived from the h-index, is defined, to be used for attributing a score, calculated automatically accessing Google Scholar for retrieving basic information, to papers. The proposed approach has been applied to COVID-19 literature published in several free access, not peer-reviewed archives, and first results are reported.

All implementations are done using the R programming language (R Core Team, 2021) via RStudio (RStudio Team, 2021), adopting the Tidyverse paradigm (Wickham, 2014; Wickham et al., 2020). Specific R packages adopted in facing the different research questions are directly referred in the related chapters.

This page intentionally left blank.

Chapter 2

Literature Review and general topics

This chapter includes:

- A review of the literature about data mining applied to sport in general, and to basketball in particular, with a deep analysis concerning the outcome prediction literature. The problem of analysing scientific literature for a specific domain has been approached also using Natural Language Processing (NLP) techniques together with ad hoc tools, and first results of this ongoing activity are included in Appendix A.

- Some information and concepts common to other chapters, and in particular:

  - National Basketball Association (NBA) organization
  - approach chosen for dataset definition
  - Four Factors definition

- A short description of the machine learning algorithms used in the rest of this dissertation, with references to the chapters where these algorithms are used.

2.1 Literature review: machine learning and basketball predictions via machine learning

2.1.1 Data mining and sport

It is several years that data analytics play a fundamental role in sport analysis and management: in the last decades, publications on statistics in sport have multiplied, and a data-based approach was adopted in each professional sport (Alamar, 2013; Albert et al., 2017), facing different kinds of problems. Analysis and applications of statistics to sport include performance measurement (Mackenzie and Cushion, 2013; Page et al., 2007; Passos et al., 2016; Sandri et al., 2020; Zuccolotto et al., 2017b, 2019), injuries prevention (see Eetvelde et al., 2021 for a review), optimal game strategies (Zuccolotto and Manisera, 2020), match preparation (Migliorati, 2020; Miller, 2015; Thabtah et al., 2019), players' selection (Lewis, 2003) and, of course, outcomes forecasting (Bunker and Thabtha, 2019; Wunderlich and Memmert, 2020).
In effect, it was with the application of the data-driven approach described in Lewis, 2003, centered on selection of players for Oakland Athletics baseball team, that analytics in sport actually entered the maturity phase.
Then, quickly, data mining in sport was widely adopted and adapted in all professional sports, such as baseball (maybe the sport with the greatest history in analytics, starting in 1977 with dedicated reports) (Koseler and Stephan, 2017), hockey (see Swartz, 2017 for a review), American football (Baker and Kwartler, 2015; Silver, 2014), football (Carpita et al., 2015, 2020; Sarmento et al., 2014) and, of course, basketball.
Basketball milestones of this analytics-based approach are pioneering works (Kubatko et al., 2007; Oliver, 2004), where the famous Oliver's "Four Factors to success" were introduced as four indexes containing a big amount of information. Then, a huge number of analyses have been done applying data mining to basketball data (see, for example,Bianchi et al., 2017; Groll et al., 2018; Metulini et al., 2018; Sandri et al., 2020; Zuccolotto et al., 2017b,a, 2019; Zuccolotto and Manisera, 2020).

2.1.2 Data mining and outcome prediction

Considering the large interest and the increasing volume in sport betting, it is easy to understand the reason why the number of attempts in predicting games' results is continuously increasing, see for instance Bunker and Thabtha, 2019; Hubáček et al., 2019.
Machine learning techniques for outcome prediction have been widely ap-

plied (Haghighat et al., 2013), covering all professional sports, from horse races (Davoodi and Khanteymoori, 2010) to hockey (Gu et al., 2016) and from American football (Beal et al., 2020; David et al., 2011; Kahn, 2003; Purucker, 1996) to football (Carpita et al., 2019; Min et al., 2008; Tax and Joustra, 2015), just to give some examples among others.

2.1.3 Data mining and basketball outcome prediction

Also basketball, of course, has been investigated under this perspective.
In Loeffelholz et al. (2009) authors worked on a dataset of 650 NBA games, and used several kinds of ANN (Artificial Neural Networks, Zhang, 2000) for outcomes prediction, correctly predicting the winning team 74.33 percent of the time (on average), higher than experts percentage claimed to be 68.67.
In Miljkovic et al. (2010) it is reported how, among several machine learning algorithms, best results in both predicting the outcomes and calculating the final match spread were produced by the Naïve Bayes approach. Authors used 778 NBA games of season 2009-2010, considering 141 features as input, and an accuracy of 67% is reported.
In Cao (2012) data of 5 NBA seasons were analyzed using ANN, Support Vector Machine (Cortes and Vapnik, 1995), Naïve Bayes and logistic regression, with the latter approach producing the best prediction accuracy (about 70%) for the classification problem of predicting the winner of a game.
In a similar way, in Beckler et al. (2013) authors used Linear Regression, Support Vector Machines, Logistic Regression and ANN for NBA outcomes' prediction, using a dataset including seasons from 1991-1992 to 1996-1997 and reporting an accuracy of 73%.
In Cheng et al. (2016) authors applied the principle of Maximum Entropy (Jaynes, 1957) to predict NBA playoff outcomes for seasons from 2007–08 to 2014–15, using box score information as features, reporting an accuracy of 74.4%.
At last, there are several betting sites suggesting NBA outcome predictions. As an example, teamranking (2021) proposes predictions about NBA match winners using 4 approaches, built on the basis of several sources (historical data, breaking news and trends). For regular season 2017-2018 the maximum accuracy is 74.3%, obtained using decision trees on data of games of March.
Outcome prediction is a classification problem (i.e. the prediction chooses one of two possible qualitative values) and it is central in this dissertation, under two perspectives:

1. it has been used as a tool in identifying success factors and their relevance: the models fitted for outcome predictions using ex-post data

ensure a really high accuracy, and as a consequence a good quality in ranking success factors.

2. it has been approached using ex-ante data, in a true forecast scenario:
   (a) to show how models using single features ensure an accuracy comparable to top literature accuracies, where a huge number of independent variables is often used
   (b) to claim how, in order to improve accuracy in outcome forecasting, it would be necessary to integrate new information, different from box score statistics, as covariates.

2.2 National Basketball Association (NBA)

National Basketball Association (NBA) is the professional basketball championship of North America, the most important in the world. In the last years, NBA teams generated combined revenues of around 7.92 billion U.S. dollars each year (Gough, 2021), and NBA teams average value for 2021 has been calculated in about 2.2 US dollars billions, ranging from 1.3 billions of Memphis Grizzlies (the 30th of this ranking) to 5.0 billions of New York Knicks (the richest one) (Badenhausen and Ozanian, 2021).

To compare these values to football, the richest European clubs (Forbes, 2021) (Barcellona, Real Madrid, Bayern Munich, Manchester United) are comparable in value (a bit less than 5 billions) to New York Knicks, but team values quickly decrease (the value of 20th, Ajax, is around 400 millions) compared to NBA teams value.

NBA championship is divided in 2 conferences (Est and West); each conference is composed of 3 divisions, and each division includes 5 teams (see Fig. 2.1), so there are 30 teams in NBA. Each season starts with a regular season involving all teams, and normally each team plays 82 games.

Regular season is followed by playoff, where only the best 16 teams fight. Playoff is a best of seven elimination tournament (first to 4 wins); from 2016, top eight teams in each conference are qualified, regardless of divisional alignment (see Fig. 2.2).

In this direct elimination competition, the winners of the conference finals gain the NBA final and fight to win the title (see Fig. 2.3). In the last 20-21 season, access rules have been modified again and the so-called *play-in* rule has been introduced, regulating the distribution of the last 2 playoff places for each conference via a mini-tournament involving teams with a final regular season ranking among 7th and 10th.

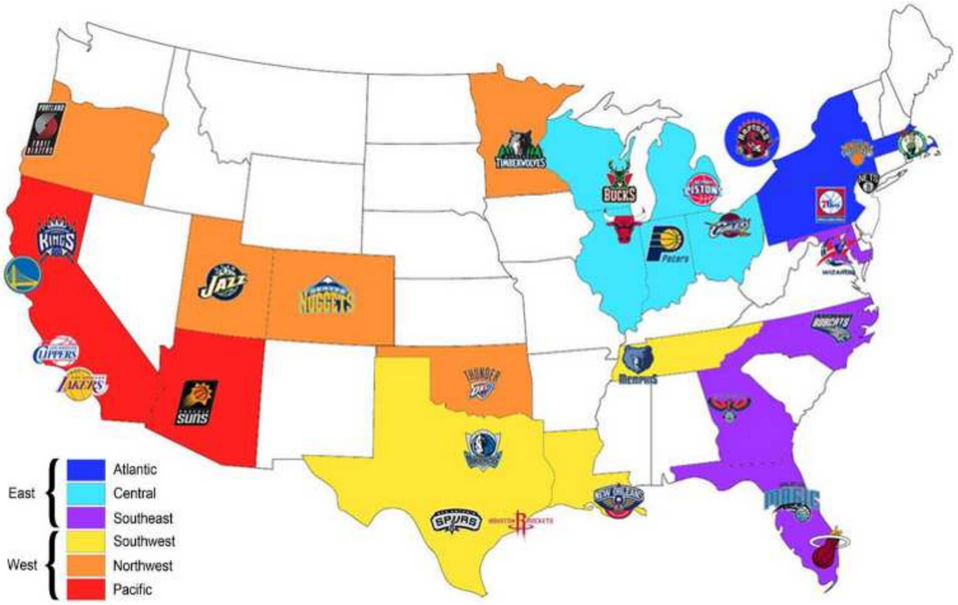


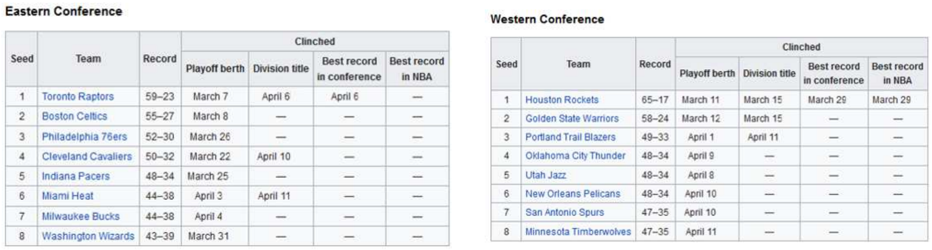
Figure 2.1: NBA geographic organization

2.3 Approach to dataset definition

The dataset used[1] in this dissertation includes only matches from NBA regular seasons. Usually, each season is seen as a uniform period, and teams are perceived as homogeneous entities during that season. The assumption is that in a single season there is continuity for a team, at least in its fundamental aspects, whereas changes occur between a season and the following one.

Actually, the situation seems to be different: it is true that teams can heavily change from one season to another, but several changes also occur during a single season. These changes can impact not only rosters (think to new free agents' contracts, new multi-year contracts, ten-day contracts for facing injuries, player exchanges, ...), but can involve coaches, managers and referees, too.

[1] Basketball dataset used in this dissertation has been obtained on the basis of play by play data kindly provided by BigDataBall (www.bigdataball.com), a data provider that leverages computer vision technologies to enrich and extend sports data sets with a number of unique metrics: since its establishment, BigDataBall has supported many academic studies as a reliable source of validated and verified statistics for NBA and several other sports.



Eastern Conference

Seed	Team	Record	Clinched			
			Playoff berth	Division title	Best record in conference	Best record in NBA
1	Toronto Raptors	59–23	March 7	April 6	April 6	—
2	Boston Celtics	55–27	March 8	—	—	—
3	Philadelphia 76ers	52–30	March 26	—	—	—
4	Cleveland Cavaliers	50–32	March 22	April 10	—	—
5	Indiana Pacers	48–34	March 25	—	—	—
6	Miami Heat	44–38	April 3	April 11	—	—
7	Milwaukee Bucks	44–38	April 4	—	—	—
8	Washington Wizards	43–39	March 31	—	—	—

Western Conference

Seed	Team	Record	Clinched			
			Playoff berth	Division title	Best record in conference	Best record in NBA
1	Houston Rockets	65–17	March 11	March 15	March 29	March 29
2	Golden State Warriors	58–24	March 12	March 15	—	—
3	Portland Trail Blazers	49–33	April 1	April 11	—	—
4	Oklahoma City Thunder	48–34	April 9	—	—	—
5	Utah Jazz	48–34	April 8	—	—	—
6	New Orleans Pelicans	48–34	April 10	—	—	—
7	San Antonio Spurs	47–35	April 10	—	—	—
8	Minnesota Timberwolves	47–35	April 11	—	—	—

Figure 2.2: Best eight conference franchises for regular season 2017/18, showing the ranking, the record and some other season's results

Sources as Marusek, 2021; Sports reference, 2021 confirm that fact: it is easy to verify how there is a huge number of transactions not only between seasons, but also during a season, invalidating the perspective of teams as homogeneous entities in that period. During season 2018-19, for instance, there were about 400 off-season signings, but about 300 in-season signings (particularly when playoffs are approaching and the admitted teams need to prepare them). So, our proposal is that a dataset could include matches selected on the basis of a homogeneous regulation framework. In sport, rules drive strategies (think to differences addressed by a championship without relegations, as NBA, with respect to a championship with relegation as normally football championships are) and tactics (think to football offside rule, or to NBA zone defense, prohibited until the 2001–2002 season), and it seems sound to consider these rules in a dataset definition.

NBA playoff rules are very different from regular season rules. In the perspective we are sketching (and depending of course on the analysis goals), it would be better to avoid including both playoff and regular season games in the same dataset.

Instead, also considering differences in rules as playoff access rules mentioned above, regular seasons' rules starting from season 2004-2005 are reasonably uniform, and analyses using a dataset built including their matches could be more suitable.

Of course, the same kind of analysis made in this dissertation about regular seasons could be replicated for playoff games which, in turn, have a not so variable frame of rules. In effect, it could be very interesting to verify the Four Factors weights in playoff games, and compare results to what we found in this dissertation for regular seasons. This job is left to future analyses.

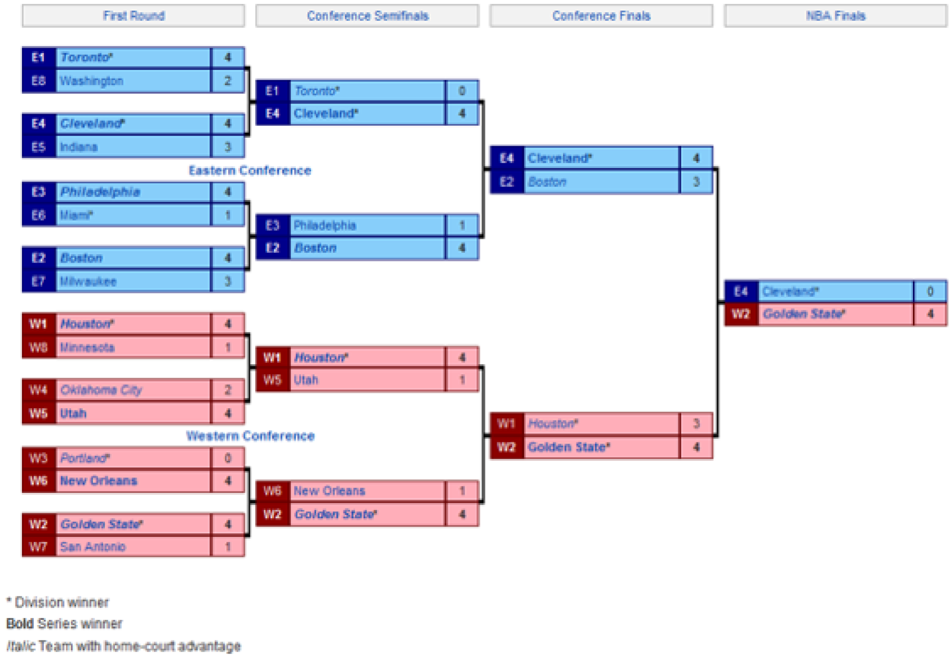


Figure 2.3: Playoff phase of season 2017/18

2.4 The Oliver's Four Factors

Basketball milestones of this analytics-based approach are pioneering works (Oliver, 2004; Kubatko et al., 2007), where famous "Four Factors to success" were introduced: few indexes concentrating a high number of information. Oliver identified Four Factors in trying to understand how basketball teams win games. They are a set of derived statistics considered fundamental for winning a match, used for summarising the attitude of a team with respect to shooting, turnovers, rebounds and free throws, and their empirical weight is reported in Table 2.1. Four Factors are defined beginning from the concept of *possession*, i.e. the number of times a team gains control of the ball during a match, formulated in its simplest and most common way as follows (refer to Table 2.2 for variable meaning):

$$Poss = (P2A + P3A) + 0.44 * FTA - OREB + TOV \tag{2.1}$$

In equation (2.1), FTA is weighted by 0.44, which is the fraction of free throws that end possessions, as estimated in the 2002–2003 through 2005–2006 seasons by Kubatko et al., 2007. Alternative and more compli-

Table 2.1: Empirical weight of Oliver's Four Factors

Factor	Weight
shooting	40%
turnovers	25%
rebounds	20%
free throws	15%

cated formulas have also been proposed in the specialised literature (Kubatko et al., 2007).

Table 2.2: Variables' acronym meaning

Acronym	Meaning
P2A	2-point field goals attempted
P3A	3-point field goals attempted
FTA	free throws attempted
P2M	2-point field goals made
P3M	3-point field goals made
FTM	free throws made
OREB	offensive rebounds
DREB	defensive rebounds
TOV	turnovers
POSS	possessions

It is possible to define the Four Factors in the following analytic way:

1. Shooting, measured by effective Field Goals percentage:
$$eFG\% = (P2M + 1.5 * P3M)/(P2A + P3A) \tag{2.2}$$

2. Turnover ratio, the number of turnovers (i.e. loss of ball) per possession:
$$TO\_ratio = TOV/POSS \tag{2.3}$$

3. Rebounds, defined by offensive rebounding percentage:
$$OREB\% = OREB/(OREB + DREB) \tag{2.4}$$
where, in the denominator, the team offensive rebounds and the opponent team defensive rebounds are considered, respectively.

4. Free throws rate:

$$FT\_rate = FTM/(P2A + P3A) \tag{2.5}$$

For each match, the Four Factors for both the home team (marked with *ht* in the following) and the away team (*at*) can be computed, leading in effect to eight factors.
In this dissertation the Four Factors have been calculated using the R package `BasketballAnalyzeR` (Sandri et al., 2018; Sandri, 2020).

2.5 Recap of machine learning methods used in this dissertation

In this Section, a list of machine learning methods used in this dissertation is proposed, including a short description of the methods, some basic references and some pointers to dissertation sections where they have been used.

- Classification And Regression Trees (CART) (Breiman et al., 1984).
  CART are a kind of binary decision tree, built binary splitting (in a recursive way) the predictor space in several regions. Splits are decided on the basis of predictors values minimizing a cost function (typically following a greedy approach, to save computational costs), and are recursively applied until a stop condition is fired; at the end of the process, observations result divided in subgroups, each one associated to a constant response. CART are used in chapter 3 for providing interpretable models.

- Random Forests (Breiman, 2001a; Ho, 1995).
  Random Forests is an ensemble learning technique, used for both classification and regression. It is based on the idea of building a huge number of different decision trees, where tree splits are evaluated considering a subset of randomly chosen predictors, to mitigate the classical trees' instability. Random Forests have been used in chapter 3.

- Logistic (logit) regression.
  Logistic Regression is a statistical model to be used in classification problems where the dependent variable is dichotomous, assuming only one of two values (for example in basketball: victory or defeat). It has been used in chapter 4.

- MOdel Based partitioning (MOB).
  MOB is an algorithm for recursive partitioning, as described in Zeileis et al., 2008, used in chapter 4. In this approach, fitting is not made via a unique global model for the complete dataset, but instead in a local way, estimating model parameters on subsets of data defined by recursive partitioning, possibly with a better fitting quality with respect to the global one, and generally with a better interpretability.

- Deep Learning (Goodfellow et al., 2016).
  The Deep Learning approach is based on Artificial Neural Networks (ANN) (Zhang, 2000); typically a Deep Learning architecture is composed by an input layer, an output layer and a number of hidden layers between the input and the output. A traversal process from one extremity to the other is repeated several times in both directions, driven by a loss function to be minimized. Deep Learning, in its Keras implementation (Chollet and Allaire, 2018), has been used in chapter 5.

- Topic model.
  Topic model (Deerwester et al., 1990) is a statistical model adopted in NLP and machine learning, aimed to classify documents in an unsupervised way identifying their hidden topics. It is often implemented via Latent Dirichlet Allocation (Blei et al., 2003), and in this dissertation has been used in applying NLP to results of bibliographic research as described in Appendix A.

Chapter 3

Analysing drivers of success in basketball matches using CART and Random Forests

Remarks Some of the contents of this chapter have been presented in:

Sports Analytics Workshop 2019
Athens University of Economics and Business (AUEB), 25th-26th Nov. 2019
in a presentation entitled:
Predicting winners in NBA basketball games by machine learning
M. Migliorati

Third international conference on Data Science & Social Research (DSSR), University of Bari, Italy, virtual conference, 10th-11th December 2020
in a presentation entitled:
I've seen things you people wouldn't believe: how data science can help coaching staff in basketball
M. Migliorati

and published in:
Electronic Journal of Applied Statistical Analysis
vol. 13, issue 2, pages 454-473, 2020
in a paper entitled:
Detecting drivers of basketball successful games: an exploratory study with machine learning algorithms
M. Migliorati

3.1 Introduction

This chapter aims to show how machine learning techniques can be profitably employed to identify drivers leading to success for basketball matches in NBA.
Data always played a fundamental role in sport management, constituting the starting point for suitable analysis in a field where a huge amount of money is invested, but where fortuity plays a great role. Moreover, considering the large interest in sport betting, it can not be surprising that several attempts were made to accurately predict games results, see references in Section 2.1.
The perspective of the work described in this chapter is different: we are not interested in predicting in advance the game winner, but in offering valuable tools to staff coaching, supporting them to decide which moves can be done both before a game and on the basketball court to increase success possibilities.
We are interested in applying data mining to basketball (Bianchi et al., 2017; Zuccolotto and Manisera, 2020), for identifying which are the drivers to victory. To do this, we will focus on the relationship between match outcomes (only win or loss, for basketball) and sets of features offering a snapshot of the match, as already studied in other sports (Carpita et al., 2015, 2020).
Our goal is not so far from Thabtah et al. (2019) where authors tried to identify the most important features in predicting outcomes for a dataset of 430 observations focused on NBA final games from 1980 to 2017. Three machine learning algorithms (Naïve Bayes, ANN and Logistic Model Trees, Landwehr et al., 2005) have been applied using box-score variables as predictors, being the number of defensive rebounds the most influential feature.
In this work we will use a large dataset (thousands of matches) to build prediction models that will be analyzed in terms of variable importance for identifying success drivers. A high fitting quality will be the guarantee that our analysis about variable importance is fair, so predictions will be made on the basis of ex-post information, with the goal of increasing the goodness of fit and consequently the goodness of the success drivers that will be identified.
To do that, we will use two different sets of predictors:

1. the so-called*box-score* analytics, i.e. the classical variables (number of attempted shots, made shots, ...) summarizing a match, see for instance (NBA, 2020a; ESPN, 2021)

2. the *Four Factors* as described in Section 2.4

A large dataset, including 14 NBA regular seasons (from 2004-2005 to 2017-2018), was built, and variables from both box scores and Four Factors have been used as regressors for machine learning classification, with the goal of building models predicting the winner of matches of Golden State Warriors (GSW), champion of the season 2017-2018.
Then, these models are analyzed in terms of quality of fit, and their variable importance studied to identify success drivers.
Classification models are built using:

- CART (Classification and Regression Trees, Breiman et al. (1984))
- Random Forests (Ho, 1995; Breiman, 2001a)

to couple trees easiness of interpretation, so important for data mining results acceptance and usage, to Random Forests stability and prediction quality. This chapter is structured as follows: the dataset is described in Section 3.2, followed by a description of CART and Random Forests algorithms in Section 3.3. Section 3.4 includes the results from applying CART and Random Forests to the dataset, with the goal of both identifying the most relevant predictors and proposing tools to facilitate coach staff in match interpretation. At last, conclusions are reported in Section 3.5.

3.2 The dataset

The current rules for regular season were adopted in season 2004-2005, and this uniformity of rules is the reason why our dataset starts from this season. The dataset includes 14 seasons from 2004-2005 to 2017-2018, for a total of about 17.000 matches of regular seasons.
Among several other statistics (see (NBA, 2020b)), basketball match analysis will be approached using:

1. box-score variables, a set of indicators summarizing the trends of a match. For our analysis purposes, we select 13 variables from the box-score information, to be used as predictors:

   1.1 PTS: number of points made

   1.2 P2A: number of 2-point field goals attempted

   1.3 P2M: number of 2-point field goals made

   1.4 P3A: number of 3-point field goals attempted

   1.5 P3M: number of 3-point field goals made

1.6 FTA: number of free throws attempted

1.7 FTM: number of free throws made

1.8 OREB: number of offensive rebounds

1.9 DREB: number of defensive rebounds

1.10 AST: number of assists (pass of the ball leading to a field goal score)

1.11 TOV: number of turnovers (loss of ball possession)

1.12 STL: number of steals (stealing ball to the opponent)

1.13 BLK: number of blocks (deflecting a field goal attempt)

These information are made available for both teams involved in a match.

2. Four Factors, the set of derived statistics described as “key to success” in Section 2.4.

We focus on a single team, Golden State Warriors (the winner of season 2017-2018):

- for the 14 regular seasons from 2004-2005 to 2017-2018 we have 1130 games involving GSW
- 90% of our observations, randomly chosen, will be used for training our classification models
- the remaining 10% will be used for testing.

3.3 Methods and Models: CART and Random Forests

In this section, Classification And Regression Tree (CART), (Breiman et al., 1984) and Random Forests (Ho, 1995; Breiman, 2001a), as applied to the above dataset, are described.

3.3.1 CART

CART (Breiman et al., 1984) are a kind of binary decision tree, a good example of the «algorithmic culture» (Breiman, 2001b). Trees are connected acyclic graphs, starting from one single node (named root) branching to children nodes. Each node has not more than 1 father, and can branch to several children. Nodes without children are called leaves. CART are built (typically

binary) splitting the predictors space in several regions. Splits are decided on the basis of the predictors values, minimizing a cost function (following a greedy approach to save computational efforts), and are recursively applied until a stop condition is fired. At the end of the process, observations will result divided in subgroups, one for each leaf of the tree, and associated to a constant value of the dependent variable.
CART can be used both for classification (when the output variable is categorical, as in our case in predicting the game winner; in this case the constant response associated to a leaf is the mode of the output variable of the observations belonging to it) and regression (when the output variable is quantitative; in this case, the constant response associated to a leaf is the mean of the output variable of the observations belonging to it).
The cost functions to be minimized typically are:

1. the Gini index for classification trees; for a generic node t:

$$G = 1 - \sum_{k=1}^{K} (p_k)^2 \tag{3.1}$$

   where K is the number of classes and p_k is the relative frequency of the class k in the node t.
   Gini index is a measure of node purity (where G=0 means a node totally pure, no incorrect classified observation)
   With the same meaning as Gini index, also information entropy:

$$E = -\sum_{k=1}^{K} (p_k * \log_2(p_k)) \tag{3.2}$$

   is used as cost function in other kinds of classification trees (ID3 Quinlan (1986), C4.5 Quinlan (1993)).

2. the Sum of Squared Errors (SSE) for regression trees. In this case the goal is to find J distinct and not overlapping regions $R_1, .., R_J$ that minimize the SSE defined as:

$$SSE = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2 \tag{3.3}$$

   where y_i is the observed value and $\hat{y}_{R_j}$ is the predicted value (the mean response) for the training observations within the jth region.

In a binary tree, predictions are made as follow:

- nodes represent partitions of the features' space, where a single predictor value is chosen to split the node in 2 branches
- for a new observation, a branch is chosen on the basis of its specific split predictor value
- the above action is repeated until the new observation reaches a tree leaf
- the constant value associated to that leaf will be the prediction for the new observation.

With CART, the attention must be payed to avoid *overfitting*, arising when a model is too much tailored on training data, so that it is not suitable in managing new observations. To this purpose trees are often pruned: the number of leaves is decreased, adding a penalization function depending on the number of leaves to the cost function to be minimized. CART play an important role in machine learning, not only because of their not so bad predictive power, but mainly because of the easiness in understanding it.

3.3.2 Random Forests

Random Forests (Breiman, 2001a; Ho, 1995) is an *ensemble* machine learning technique, i.e. a meta approach to machine learning, trying to improve predictive performance by combining predictions generated by several models (the so-called *weak learners*).

In particular, Random Forests is derived from bagging (bootstrap aggregating) algorithms, where a huge number (ensemble) of different decision trees are trained from bootstrap samples (random samples with replacement) of the training set. The ensemble of trained decision trees is then used to make predictions about new data. In effect, CART suffer from high variance, both in terms of lower prediction accuracy and instability (little changes can produce big impacts). It is known how, in general, average reduces variance, and bagging algorithms use this fact producing a more accurate final answer without increasing the bias.

With respect to bagging, Random Forests also draw random subsets of features for training the individual trees: in bagging, each tree is built considering the full set of features, instead in Random Forests the trees are more independent of each other and often produce better prediction accuracy.

In other words: in Random Forests, tree splits are evaluated on the basis of a

subset of randomly chosen predictors (building a model based on p features, in each split $\sqrt{p}$ features are considered for classification problems, $p/3$ for regression). In this way, it is possible to generate trees where also bagging neglected predictors can play a role, and consequently also rare models can be considered when the majority rule is applied, helping both to mitigate the classical tree overfitting problem, and to increase prediction performance.
The Random Forests' model can be used both for classification and regression, with a good predictive power. Unfortunately, the structure of the models produced by this machine learning algorithm is not so simple to be interpreted as, for instance, trees are.

In the following we will use both CART and Random Forests, looking for an interpretable answer to the classification question: ”will Golden State Warriors win this game?”.

3.4 Results

3.4.1 CART model: box-score variables as predictors

Our first model is a CART built using box-score variables as predictors. A first application using all the 13 box-score variables for both teams produces only an obvious conclusion: the number of points made is by far the most important variable, and Golden State Warriors will win a game when its score is greater than the opponent score.
Moreover, a correlation analysis shows how the number of assist (the variable AST) is strongly positively correlated (0.67) to points (PTS) (and it could not be different: an assist is a pass to a player realizing a successful throw; of course, the opposite is not true, for instance considering successful free throws).
Often it can happen that models address obvious or expected interpretations, and this is always a good point, confirming the soundness of the analysis. In any case, we are interested in finding interesting and unexpected insights, too. To this purpose, we exclude both PTS and AST from predictors, as shown in listing 3.1:

```
train.cart.bs=rpart(Result~
                    P2M.team+P2A.team+P3M.team+P3A.team+
                    FTM.team+FTA.team+OREB.team+DREB.team+
                    TOV.team+STL.team+BLK.team+
                    P2M.opp+P2A.opp+P3M.opp+P3A.opp+
                    FTM.opp+FTA.opp+OREB.opp+DREB.opp+
                    TOV.opp+STL.opp+BLK.opp,
                    data=train.bs,
                    method='class',
                    control = treeControl
)
```

Listing 3.1: CART using a subset of box-score predictors

The resulting tree (pruned to limit overfitting) seems more interesting, and is depicted in Figure 3.1:

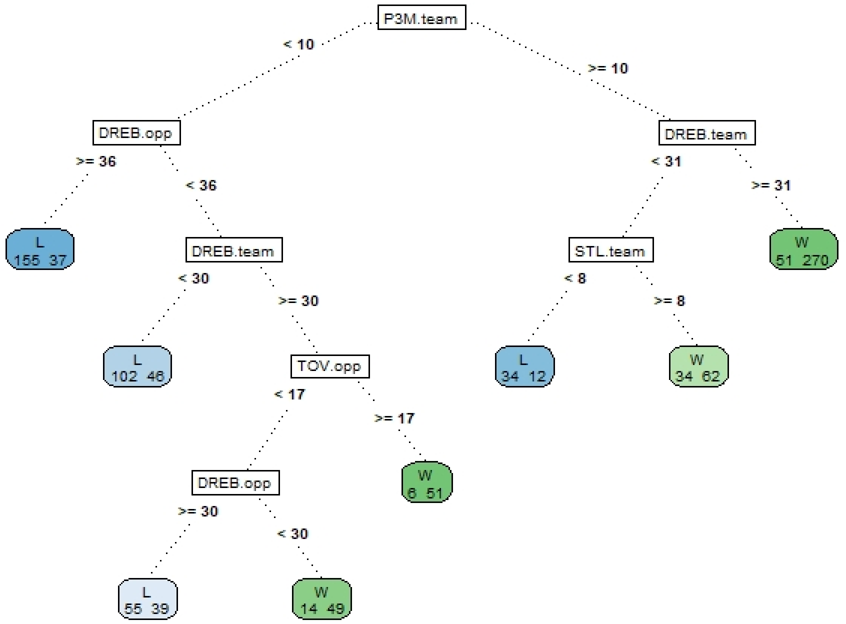


Figure 3.1: Training dataset CART based on box-score predictors without PTS and AST

In this tree:

- nodes represent predictors: so, the root of the tree, the *P3M.team* node, represents the*number of 3-point shots made* input
- for classifying a new observation, the tree is crossed on the basis of the value of its predictors: starting from the root of the tree, we will descend to the left if the number of 3-point shots made for the new

observation (match) is lower than 10, otherwise we will descend to the right

- in the leaves the classification result is highlighted: Win (green) or Lose (blue), together with frequency of the 2 possible classification results. Color is darker depending on how a frequency is greater than the other one.

So, a new observation is classified finding the most convenient leaf on the basis of its predictor values, crossing the tree starting from the root. The prediction for this new observation is given by the constant associated to that leaf.
As an example, the meaning of the pink path in Figure 3.2

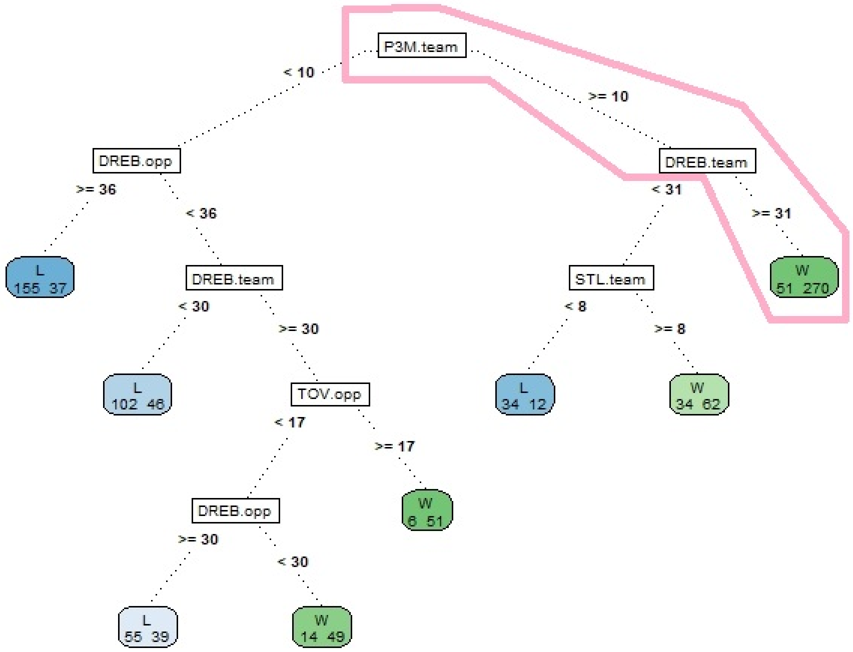


Figure 3.2: A path in the CART built using box-score predictors

is that Golden State Warriors will be predicted to win its match when they make at least ten 3-point shots ($P3M.team \geq 10$) and thirty-one defensive rebounds ($DREB.team \geq 31$): a good example of the easiness of tree

understanding mentioned before.
Moreover, that tree offers a clear example of the reason why this kind of tools can be really valuable to coaching staff.
Looking to the left subtree, we can see how the situation seems against GSW when the number of 3-point shots is lower than 10 ($P3M.team < 10$), because many leaves predict a GSW defeat with high probability. But, in effect, the tree shows how there are still 2 possibilities for winning, that can be tried if the opponent is not too strong in defensive rebounds ($DREB.opp < 36$). In this situation, GSW should pay great attention to defense, trying to get a high number of defensive rebounds ($DREB.team \geq 30$), together with an aggressive attitude inducing opponent or to lose many balls ($TOV.opp \geq 17$)), either to take few defensive rebounds ($DREB.opp < 30$).
So, this model suggests a possible strategy to GSW coaching staff for trying to bring the team to green (i.e. victory) leaves.
Only 5 predictors (3-point shots, team defensive rebounds, opponent defensive rebounds, team stolen balls, opponent lost balls), among the 22 available, were used to build up the tree: this is a good first step in the direction of identifying the success drivers we are looking for, but first we have to verify how much the model is suitable.
To assess quality of fit, the CART model is tested by predicting the final results for the remaining 10% of our observations, and comparing these predictions to actual results.
In other words, we built a confusion table for predictions and actual results on the test set (Table 3.1).

Table 3.1: Confusion matrix for CART box-score without PTS and AST, test dataset

		Prediction		
GSW winner?		no	yes	Total
Actual	no	34	10	44
	yes	22	47	69
	Total	56	57	113

Accuracy[1](defined as the ratio between correctly classified observations

[1] Several other indexes about prediction quality are reported in Table 3.5, where a comparison of quality of fit for the different models is reported

and overall number of observations) for this model is 0.7168, high enough (taking into account we are using a subset of available predictors) for safely analyzing variable importance to detect success drivers. Variable importance measures for CART models are calculated[2] not only considering the reduction in the loss function attributed to each variable for its effective splits, but also surrogate splits (i.e. possible predictors to be used in case of missing values) and competing splits (i.e. predictors evaluated but not chosen for splits), to have a view as complete as possible.
In this way, we can trace not only the variables appearing in tree plot (effectively selected for split), but also the other variables playing a meaningful role (also if not selected for splitting) in the tree building.
For the CART model above, variable importance is depicted in Figure 3.3.

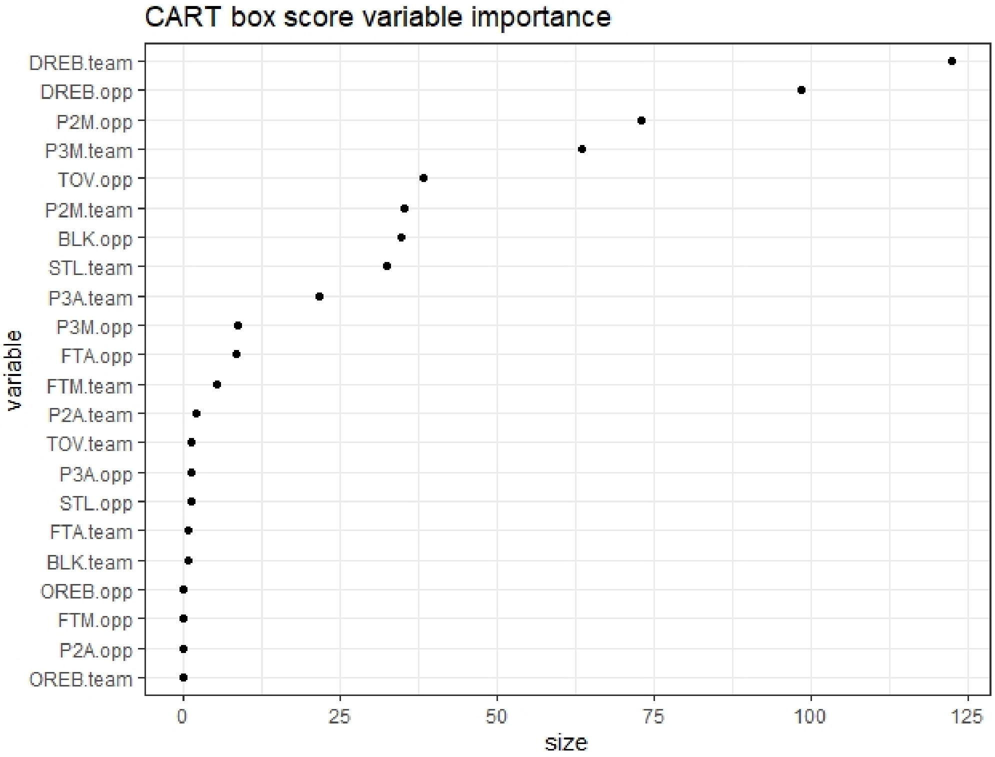


Figure 3.3: Variable importance measures for CART box-score without PTS and AST

[2] See R `caret` package documentation for details (Khun, 2020)

Figure 3.3 shows that defense is a key factor for GSW successes, and (but this is more obvious) to win it is necessary to made 2- and 3-point shots. Moreover, it is possible to note how offensive rebounds, usually considered as really important (as also shown in Zuccolotto et al., 2017b), are neglected in our model, also in our analysis where the largest definition of variable importance (including the evaluation of surrogate and competing splits, too) has been applied.

3.4.2 CART model: Four Factors as predictors

The second model is still built using CART, but with Four Factors as predictors as reported in Listing 3.2.

```
train.cart.ff=rpart(Result~
                        eFG.team + eFG.opp +
                        TO.team + Reb.off.team + FT.rate.team +
                        TO.opp + Reb.def.team + FT.rate.opp ,
                        data=train.ff,
                        method='class',
                        control = rpart.control(minbucket = 25)
)
```

Listing 3.2: CART using all Four Factors predictors

The (pruned) tree for our training dataset is depicted in Figure 3.4:

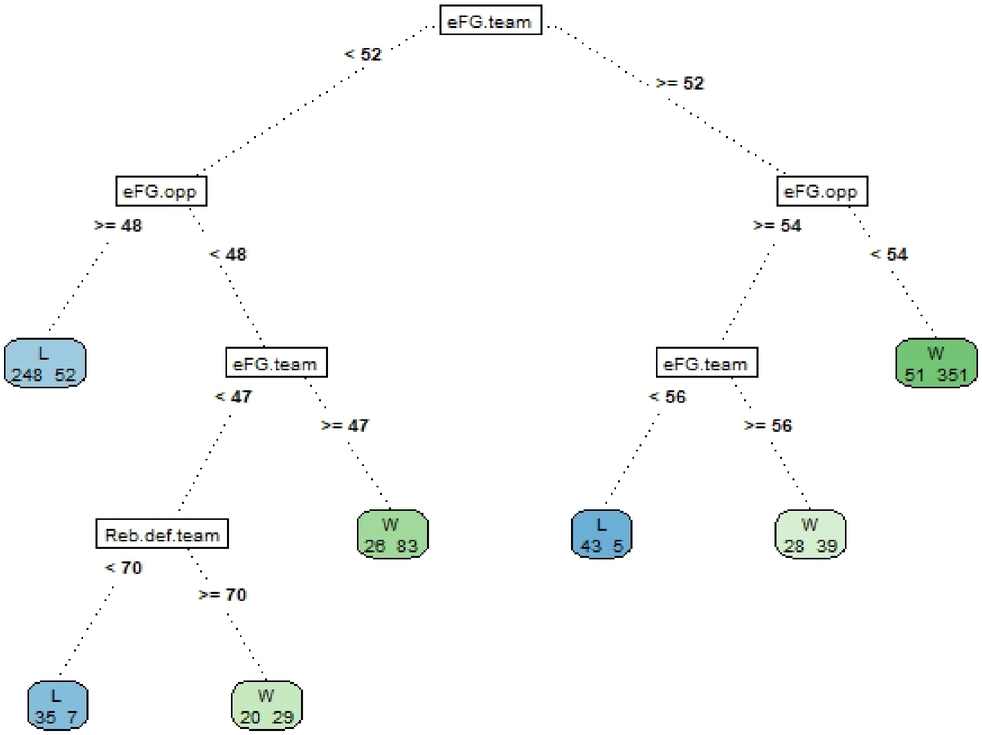


Figure 3.4: Training dataset CART built using all Four Factors

In this case the model is built using just three predictors among the 8 variables available: only (*eFG.team*, i.e. GSW shooting factor, *eFG.opp*, i.e. opponent shooting factor, and *Reb.def.team*, i.e. GSW defensive rebound factor) are effectively used for splitting.
This tree does not contain unexpected information: mainly, it shows how shooting factor plays a primary role as success driver (a first confirmation of Oliver's weights definition). An interesting point: defensive rebounds seem to have a high importance (node *Reb.Def.team* on the left), confirming what we observed analyzing CART model built using box-score variables.
In any case, the presence of shooting factors maybe hides other useful information, so we build a model without them (see Listing 3.3).

```
train.cart.ff2=rpart(Result~
                    TO.team + Reb.off.team + FT.rate.team +
                    TO.opp + Reb.def.team + FT.rate.opp ,
                    data=train.ff,
                    method='class',
                    control = rpart.control(minbucket =90)
)
```

Listing 3.3: CART using a subset of Four Factors predictors

obtaining the tree in Figure 3.5.

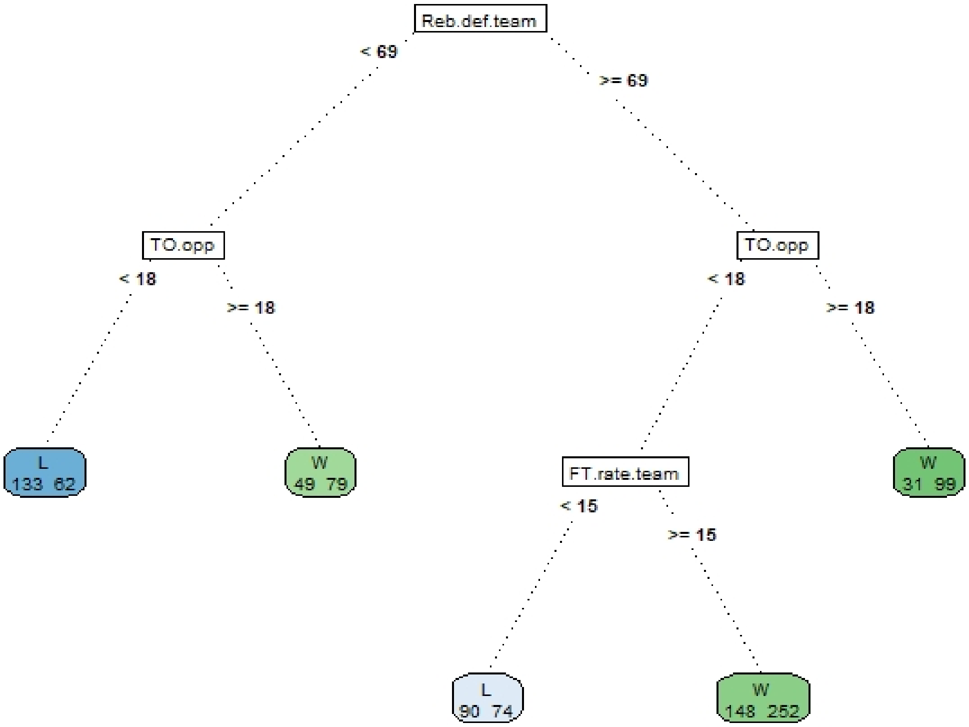


Figure 3.5: CART built using Four Factors without shooting

It confirms the conclusions about the importance of GSW defense made using the model built using box scores variables (Figure 3.2): they must fight for rebounds in their own area, and play in an aggressive way to induce

opponents to lose the ball.
This tree offers another interesting path to be interpreted: the *FT.rate.team* (i.e. the number of free throws respect to sum of 2- and 3- point shots) becomes important in situations where GSW is strong on defensive rebounds, but not strong enough in frequently inducing the opponent to lose the ball. In these situations, it can be appropriated to play for gaining free throws with respect to other play finalizations; how to get it depends on the contingent situation: for instance, the coaching staff could decide to play in attack and keep the ball, inducing the opponent players to foul.
Again, the tree can be a valuable tool to coaching staff, both to prepare the match and to react in the right way to situations happening on the court.
The confusion table addressing quality of fit for CART on the test dataset is reported in Table 3.2.

Table 3.2: Confusion matrix for CART Four Factors without shooting, test dataset

		Prediction		
GSW winner?		no	yes	Total
Actual	no	18	26	44
	yes	11	58	69
	Total	29	84	113

Accuracy is 0.6726: not so bad, taking into account we are not considering shooting, the most important factor, and high enough to let this analysis be used as a decision support tool for coaching staff.

3.4.3 Random Forests model: box-score variables as predictors

In this section we will analyze models built using Random Forests, to compare CART results to another high quality fit model. By construction, Random Forests enable emerging of lower weight variables, but the presence of variables *PTS* and *AST* is too cumbersome; so, as for CART, we prefer to make box-score based predictions without them (see Listing 3.4).

```
train.rf.bs.s=randomForest(Result~
                             P2M.team+P2A.team+
                             P3M.team+P3A.team+
                             FTM.team+FTA.team+
                             OREB.team+DREB.team+
                             TOV.team+STL.team+
                             BLK.team+
                             P2M.opp+P2A.opp+
                             P3M.opp+P3A.opp+
                             FTM.opp+FTA.opp+
                             OREB.opp+DREB.opp+
                             TOV.opp+STL.opp+
                             BLK.opp,
                         data=train.bs,
                         importance=TRUE)
```

Listing 3.4: Random Forests using a subset of box-score predictors

Results can be found in Table 3.3.

Table 3.3: Confusion matrix for Random Forests box-score without PTS and AST

		Prediction		
GSW winner?		no	yes	Total
Actual	no	40	4	44
	yes	6	63	69
	Total	46	67	113

Accuracy is about 0.9: the model is really suitable, also if we don't use PTS and AST predictors. Variable importance, in terms of mean decrease and accuracy node impurity, is depicted in Figure 3.6.

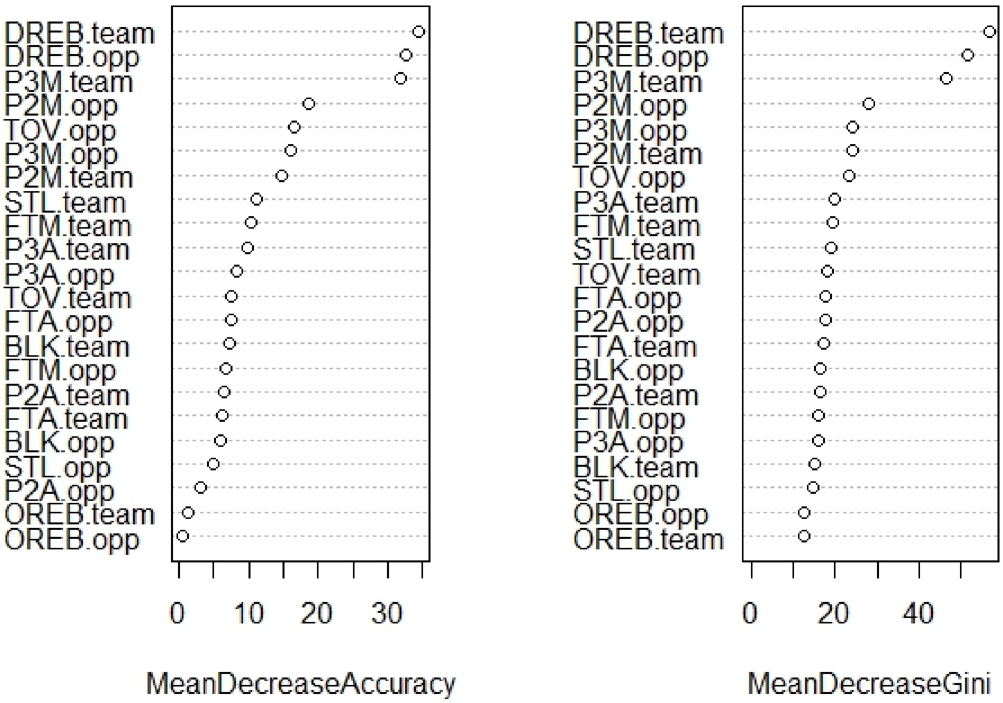


Figure 3.6: Random Forests using box-score variables as predictors, variable importance measures

Again, we find a confirmation that defense is really important for GSW to win: among the most important variables we have defensive rebounds and opponents turnover. As it is easy to guess, made shots are important, too. At last, it is confirmed the low importance of offensive rebounds.
So, this analysis is aligned with conclusions we made on the basis of CART model results.

3.4.4 Random Forests model: Four Factors as predictors

Our last model is built on Random Forests, again, but using Four Factors as features (see Listing 3.5).

```
train.rf.ff=randomForest(Result~
                        eFG.team + eFG.opp +
                        TO.team + Reb.off.team + FT.rate.team +
                        TO.opp + Reb.def.team + FT.rate.opp ,
                        data=train.ff,
                        importance=TRUE)
```

Listing 3.5: Random Forests using all Four Factors predictors

Prediction results are summarized in Table 3.4.

Table 3.4: Confusion matrix for Random Forests (Four Factors)

		Prediction		
GSW winner?		no	yes	Total
Actual	no	40	4	44
	yes	3	66	69
	Total	43	70	113

This model has the highest accuracy (0.94). In this case we used all the 8 predictors, because the model without shooting, the most important among the Four Factors, has an accuracy equal to 0.6018, not so high, without offering new infos about success drivers. Variable importance measures are reported in Figure 3.7.

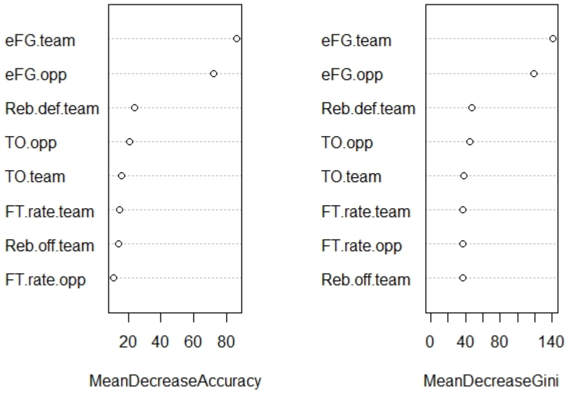


Figure 3.7: Random Forests using Four Factors as predictors, variable importance measures

As we can imagine shooting factors, both offensive and defensive, play a primary role; moreover, we have another confirmation about the importance of defense and low importance of offensive rebounds, as we argued interpreting other models.

3.5 Conclusions

In this chapter we have applied CART and Random Forests to a large dataset of NBA games, with the purpose of detecting success factors. In particular, we focused on Golden State Warriors regular seasons from 2004-2005 to 2017-2018, sharing the same rules framework, looking for GSW victories' drivers. Our classification models were built using both box-score variables and Four Factors as predictors, on the basis of ex-post data, showing high quality-of-fit measures, as reported in Table 3.5.

Table 3.5: Comparing prediction quality

	CART Box-Score no PTS and AST	CART Four Factors no shooting	Random Forests Box-Score no PTS and AST	Random Forests Four Factors
sensitivity	0.6812	0.8406	0.9130	0.9565
specificity	0.7727	0.4091	0.9091	0.9091
accuracy	0.7168	0.6726	0.9115	0.9381
recall	0.6812	0.8406	0.9130	0.9565
precision	0.8246	0.6905	0.9403	0.9429
F_measure	0.7461	0.7582	0.9264	0.9497

The goodness of fit of all models support us in using their variable importance measures as success drivers.
In the case of Golden State Warriors we detect how, after shooting factors, the most important success factors are related to defense (defensive rebounds and opponent turnovers). Instead, it seems that offensive rebounds are not so important, confirming the famous saying ‘offense sell tickets, defense wins championships”.
With respect to the Oliver’s Four Factors weighting, it is confirmed how the shooting factor is the most important success driver but, in our dataset, defensive rebounds seem to be more important than turnover. Instead, the results also confirmed the lower importance of free throws (but we verified how they become important in particular situations on the court).
The obtained models can be useful to the coaching staff in preparing games; moreover CART trees, thanks to their understandability, can be useful also on the court, to interpret the game and try to drive the team to advantageous situations.
Next steps will concern the application of the described approach to other teams and basketball championships (e.g. the Italian Lega Basket), to verify if and how success factors are team and championship specific, and to investigate models and drivers for score differences.

Chapter 4

Four Factors and the probability of winning a basketball game: MOdel-Based recursive partitioning

Remarks Some of the contents included in this chapter have been presented in:
13th International Conference of the ERCIM WG on Computational and Methodological Statistics
14th International Conference on Computational and Financial Econometrics (CFE-CMStatistics)
King's College, London, England, Virtual Conference, 19-21 December 2020
in a presentation entitled:
The impact of Four Factors on a basketball team's success: an approach with model-based recursive partitioning
M. Migliorati, M. Manisera, P. Zuccolotto

and have been submitted to:
AStA Advances in Statistical Analysis
Special Issue on "Statistics in Sports"
in a manuscript entitled:
The impact of Oliver's Four Factors on the probability of winning a basketball game: An approach with MOdel-Based recursive partitioning
M. Migliorati, M. Manisera, P. Zuccolotto

4.1 Introduction

This chapter focuses on the well-known Oliver's Four Factors (Oliver, 2004; Kubatko et al., 2007) described in Section 2.4, with the aim of investigating how these Factors, used as predictors, can have a different importance in driving a match result on the basis of the strength balance between the two basketball teams involved.

To investigate the impact of each factor in calculating the probability of winning a game (and, consequently, predicting the winner) and simultaneously accounting for differences in class between the two teams, we applied the MOdel-Based recursive partitioning (MOB) algorithm (Zeileis et al., 2008). This algorithm, instead of fitting one global Generalized Linear Model (GLM), estimates several local GLMs considering subgroups of observations built on the base of recursive partitioning.

The philosophy is the same as in Migliorati (2020), where both decision trees and random forests were applied to a large NBA dataset to assess the importance of two sets of predictors, constituted by box scores' statistics and the Four Factors, respectively.

To this purpose, we used real data from 19.138 matches of 16 National Basketball Association (NBA) regular seasons, from 2004–2005 to 2019–2020, and defined an index, used to partition observations with MOB, calculated as the difference between victory relative frequencies in previously played games for the two teams competing in a match.

The analysis highlighted (quasi) separation problems, introducing convergence problems in the numerical solution of the Maximum Likelihood (ML) estimation. To address this issue, we coupled MOB with Bias Reduction (BR) estimation (Firth, 1993), which ensures finite estimates also in the case of (quasi) separation situations (Kosmidis et al., 2020).

This chapter's original contribution is:

1. from a methodological point of view, an extension of GLM-based recursive partitioning from standard ML estimation to BR estimation by integrating MOB with BR GLM estimation, solving ML convergence problems and ensuring finite estimates when (quasi) separation occurs.

2. from an applied point of view, the diversification of the Four Factors' weights on the basis of the differences in class between the teams competing in a match, providing a valuable tool to coaching staff in analysing a match.

This chapter is structured as follows: Section 4.2 describes the dataset used in the application; Section 4.3 summarises the MOB algorithm and the computational issues we encountered, driving the definition of the original integrated solution mentioned above; Section 4.4 reports the results derived from the application of the BR-based GLM trees to basketball data, and proposes an analysis for both home and away matches, showing how the obtained results can be a valuable tool for coaching staffs. At last, Section 4.5 presents some conclusions.

4.2 The dataset

The dataset analysed in this chapter includes all the matches of the NBA regular seasons from 2004–2005 to 2019–2020 (until 11/03/2020, when the NBA was stopped for some months due to Covid–19).
Several analyses have been performed in order to find the best model for addressing the impact of Four Factors on victory probability. Different indexes (e.g. season, conference, victory percentage) have been considered, but at last we have chosen to base the analysis on a statistic, named *diff*, computed for each match in each season, and defined as the difference between the relative frequency of victories for home team *ht* and the relative frequency of victories for away team *at*. Considering the g-th match of a season composed of G matches we have:

$$diff_g = \frac{won\_matches_{ht,(g-k):(g-1)}}{played\_matches_{ht,(g-k):(g-1)}} - \frac{won\_matches_{at,(g-k):(g-1)}}{played\_matches_{at,(g-k):(g-1)}} \tag{4.1}$$

where $won\_matches_{ht,(g-k):(g-1)}$ and $played\_matches_{ht,(g-k):(g-1)}$ indicate, respectively, the global number of matches won and played by the home team considering the last k matches – until the $(g-1)$-th match – played against all the teams met in the championship (considering both home and away matches). Analogously, $won\_matches_{at,(g-k):(g-1)}$ and $played\_matches_{at,(g-k):(g-1)}$ indicate the global number of matches won and played, respectively, by the away team in the last k matches. For some g's, the two denominators in equation (4.1) may differ due to calendar differences in the championship schedule for the two considered teams *ht* and *at*.

The *diff* index ranges from -1 to 1:

1. It is equal to 1 when the home team is absolutely the strongest between the two competing teams in that match, because the home team won all the games played so far in the season (against all the teams it encountered so far), while the away team never won.

2. On the contrary, it is equal to -1 when the away team won all the games played so far and the home team never won.

In order to have a sufficient number of matches to compute the *diff* index, we modeled only the data from the second half of each season (so g starts from $(G/2)+1$ for every complete season) and computed via (4.1), for each g, considering the number of matches won and played starting from the first match of the season ($g-k$ fixed and equal to 1 with k increasing as g increases, according to the increase of the amount of information available at the g-th match).
The data from the second half of each season in the analysis include 9.569 observations. As equation (4.1) suggests, a possible alternative procedure to compute the *diff* index could be considering k adequately tuned in order to include only the past k matches (see, for example, chapter 5).

The *diff* index provides an easily interpretable way of summarising information about the results gained by the two competing teams in the last k matches of the season up to the match preceding the considered one, and can be interpreted as the difference in class between the two teams.

The dataset was then split into a training set, including matches from seasons from 2004–2005 to 2017–2018 (8,469 observations, 88.5%), and a test set, including matches from seasons 2018–2019 and (part of) 2019–2020 (1,100 observations, 11.5%).

Among more than 70 features available in the dataset, for studying the Four Factors weights we decided to use:

- the *result* of the match (referred to the home team victory and identified with the two categories 1 and 0 for won and lost, respectively) as dependent variable.
- The Four Factors (for both home and the away teams) as regressors.
- The *diff* index as MOB partitioning variable, in a way that will be clarified in the next section.

4.3 Methods and Models: MOdel Based recursive partitioning (MOB) and Bias Reduced GLM (BRGLM)

In this section, MOdel Based recursive partitioning (MOB) is described, together with the enhancement related to its integration with BRGLM, proposed to extend GLM-based recursive partitioning from standard ML to BR estimation, to have finite estimations in (quasi) separation conditions.

4.3.1 MOdel Based recursive partitioning (MOB)

MOB is an algorithm for recursive partitioning, as described in Zeileis et al. (2008).
In this approach, fitting is not made via a unique global model for the complete dataset, but instead in a local way, estimating model parameters on subsets of data defined by recursive partitioning and building in this way a segmented model.
The MOB algorithm is composed of four steps:

1. a parametric model $M(Y, \theta)$ (where Y is the set of observations and θ the vector of parameters) is fitted to all observations in the current node by estimating the parameter $\hat{\theta}$ via optimization of an objective function $\Psi(Y, \theta)$ (e.g. Ordinary Least Squares (OLS) or, with the appropriate changes, Maximum Likelihood Estimation (MLE))

$$\hat{\theta} = argmin \sum_{n=1}^{n} \Psi(Y_i, \theta) \tag{4.2}$$

2. estimates of models' parameters are tested for instability with respect to a set of partitioning variables $Z_1, .., Z_l$: if overall instabilities are detected, the variable Z_j associated to the highest instability is selected for split. Depending on the kind (numerical or categorical) of the partitioning variable Z_j, different instability tests can be used.

3. the split value of the selected partitioning variable that locally (greedy approach, for computational issues) optimizes the objective function Ψ is chosen; typically, for numerical partitioning variables the number of splits is fixed[1] equal to 2, instead it is fixed equal to C (the number of categories) for categorical partitioning variables.

4. the current node is split into child nodes, and the procedure is repeated for each child, until a stopping criterion (typically no more instabilities detected), is satisfied.

[1] In theory, the number of splits for numerical partitioning variables could be adaptively chosen, but typically, for computational issues, a binary splitting is carried out.

In this way a tree is built, where each leaf has associated a partition of the original observations. The subset of observations in each leaf is used for building a local model, possibly with a better fitting quality with respect to the global one. Differently from Decision Trees, exhibiting a single, constant fit in each leaf, MOB provides a complete model (e.g. linear or logit regression) in each leaf. Moreover, in regression MOB offers the possibility to have estimated coefficients tailored on subsets of observations, providing a higher interpretation value with respect to classical regression models.
MOB constitutes an interesting implementation of the Breiman's "two cultures" paradigm (Breiman, 2001a), where a dependent variable is addressed by two sets of variables:

1. variables $X_1, X_2, \ldots, X_p$, constituting the set of predictors to be used for the stochastic data model
2. variables $Z_1, Z_2, \ldots, Z_l$, constituting the set of variables to be considered for recursive partitioning in building the tree, i.e. the algorithmic leg.

In our analysis, as already mentioned in the end of Section 4.2, the dependent variable is the *result* of a match, assuming value 1 if the home team won the match and 0 otherwise, and we will use the logistic regression model. The MOB algorithm is provided by the `partykit` R package (Hothorn and Zeileis, 2015), and can be used both with predefined fitting function (e.g. linear regression or logistic regression) or integrating your own fitting function.
Quality of fit of the models will be measured using the Receiver Operating Characteristic (ROC) curve with the Area Under the Curve (AUC) (Bradley, 1997), together with the accuracy index.
Starting from the concepts of sensitivity (the proportion of true positives correctly identified) and specificity (the proportion of true negatives correctly identified), a ROC plot displays the performance of a binary classifier, showing how both sensitivity and specificity change when the classification threshold ranges over all possible values. In other words, it shows the trade off in sensitivity and specificity for all possible thresholds.
In ROC context, AUC ranges between 0 and 1 and measures the performance of a classifier, where an higher AUC means a better classification.
The optimal threshold, identified according to (Youden, 1950)[2] as the cut-off that maximizes the distance from the ROC curve to the identity (diagonal)

[2] Alternatively, the closest.topleft can be used; in this case the optimal threshold is the point closest to the top-left part of the plot.

line, will be used to classify observations in the *result* categories 1 or 0 according to the estimated probabilities.

4.3.2 A methodological extension of GLM-based recursive partitioning from standard MLE to bias reduction estimation

The logistic regression model states that

$$Pr(Y_i = 1|\boldsymbol{x}_i) = \frac{1}{1 + \exp(-\boldsymbol{\beta x_i})} \tag{4.3}$$

where Y_i is the dichotomous (with values of 0 and 1) dependent variable, $\boldsymbol{x_i}$ is the i-th element (i=1,..,n) of the independent variables vector and $\boldsymbol{\beta}$ is a (row) vector of coefficients (named θ in the generic description of MOB algorithm).
Expressed in "logit" form, it can be written as

$$ln\left[\frac{Pr(Y_i = 1|\boldsymbol{x}_i)}{Pr(Y_i = 0|\boldsymbol{x}_i)}\right] = \boldsymbol{\beta x_i}$$

To estimate $\boldsymbol{\beta}$ coefficients, generally logistic regression uses MLE, trying to maximize the log-likelihood function.
For some simple cases, this optimization problem can be deterministically solved: in the case of a dichotomous dependent variable and only one dichotomous independent variable, a $2x2$ crosstable can be build, with observed cell frequencies $f_{11}, f_{12}, f_{21}, f_{22}$, the intercept is 0 and the estimate of the β slope coefficient is

$$\hat{\beta} = ln\left[\frac{f_{11}f_{22}}{f_{21}f_{12}}\right]$$

As said above, this is true only in simple cases: in general it is not possible to analytically find a solution to that maximization problem, and iterative numeric algorithms (e.g. Gradient descent, Newton-Raphson, quasi-Newton, ...) must be used.
These iterative approaches can have problems in the presence of local maxima, but that is not the case for logistic regression because the log-likelihood function is concave, having at most 1 maximum. In any case, the situation can become problematic when log-likelihood function does not have a maximum, being monotonic increasing. In this case the maximum likelihood estimate would not exist, and so iterative methods used for finding MLE estimators do not converge: the model will always answer 1 (or 0) from a

certain predictor value on. More precisely, if we consider the realisations y_i, with $i = 1, \ldots, n$, of the n independent Bernoulli random variables and the p-dimensional covariate vector x_i associated with each y_i, we have (Allison, 2004, 2008):

- complete separation in the sample points $(y_1, x_1^T)^T, \ldots, (y_n, x_n^T)^T$ if there exists a vector $\gamma \in R^p$ such that $\gamma^T x_i > 0$ $\forall i$ with $y_i = 1$ and $\gamma^T x_i < 0$ $\forall i$ with $y_i = 0$;
- quasi-complete separation or (quasi) separation in the sample points if there exists a vector $\gamma \in R^p$ such that $\gamma^T x_i \geq 0$ $\forall i$ with $y_i = 1$ and $\gamma^T x_i \leq 0$ $\forall i$ with $y_i = 0$;
- overlap if neither complete nor quasi-complete separation occurs.

Data separation is a necessary and sufficient condition for the maximum likelihood estimate to have at least one infinite-valued component (Albert and Anderson, 1984). In (quasi) separation situations, standard maximum-likelihood estimation methods, such as the Iteratively Reweighted Least Squares (IRLS) (Green, 1984) commonly used in logistic regression, can be numerically unstable and inferential results can be wrong when procedures are based on the estimates and the estimated standard errors (Mansournia et al., 2018).
An example is the dataset reported in Figure 4.1, taken from (Allison, 2008), where y is the dependent variable and x is the single predictor, presenting a complete separation: a linear combination of the regressor is perfectly predictive of the outcome; taking into account the linear function

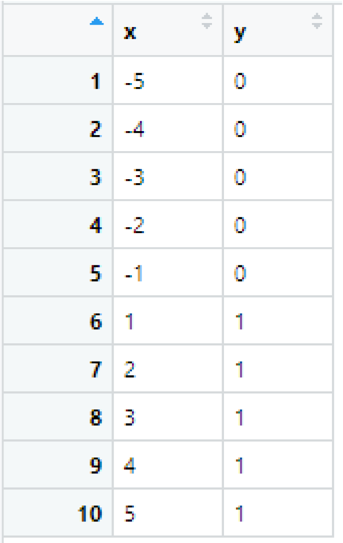
$$c = 0 + 1(x)$$

where the vector of coefficients β is equal to $[0, 1]$ and applying 4.3, we obtain

$$\hat{y} = \frac{1}{1 + \exp(-x)}$$

i.e. that for all negative x, y holds 0 and for all positive x, y holds 1, as plotted in figure 4.1.

An analogous situation for quasi complete separation, happening when complete separation condition holds, and moreover equality condition holds for at least one case in each category of the dependent variable.
In the dataset in Figure 4.2 we have the same rows as for complete separation, and 2 more rows related to value 0 for x, where y assumes values 0 and 1 (observations 6 and 7).



	x	y
1	-5	0
2	-4	0
3	-3	0
4	-2	0
5	-1	0
6	1	1
7	2	1
8	3	1
9	4	1
10	5	1

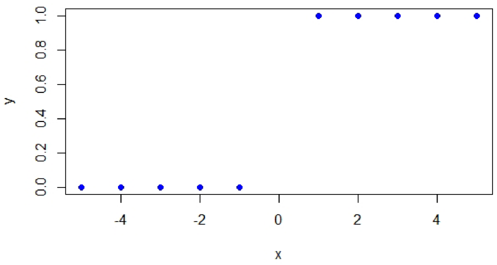
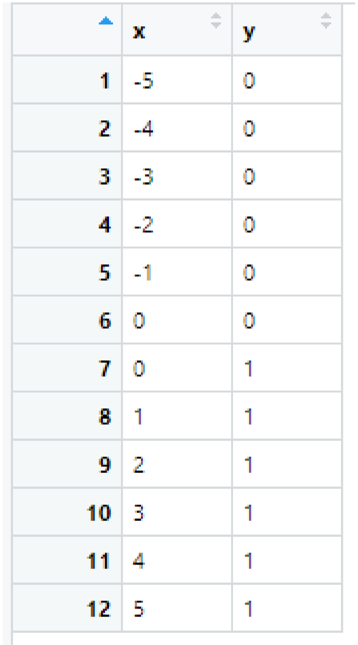


Figure 4.1: Complete separation example: the dataset is on the left, the dependent variable prediction on the right

We incurred in warnings about the (quasi) separation when both classical logistic regression and MOB (using logistic regression as a fitting function) were used to fit ex-post data with *result* (home team won or lost) as the dependent variable, the Four Factors as predictors and some variables in the dataset as partitioning variables.
As in MOB only the standard MLE procedure was available, to face such a problem two main approaches [3] have been investigated:

1. Exact logistic regression. Exact logistic regression was originally proposed by Cox (1970), but was not computationally feasible until relatively recently.
   It consists in abandoning MLE and instead using permutations of data for producing exact p-values for the null hypothesis that a specified predictor variable has a coefficient of 0, conditional on all the other predictors. These p-values are essentially unaffected by complete or quasi complete separation. The approach is computationally heavy and can be used only for small datasets. In Allison (2008), exact logistic regression is evaluated as follows: "Despite the attractiveness of

[3] we do not take into account the simplest solution consisting in identifying and deleting problematic observations, to avoid losing of information



	x	y
1	-5	0
2	-4	0
3	-3	0
4	-2	0
5	-1	0
6	0	0
7	0	1
8	1	1
9	2	1
10	3	1
11	4	1
12	5	1

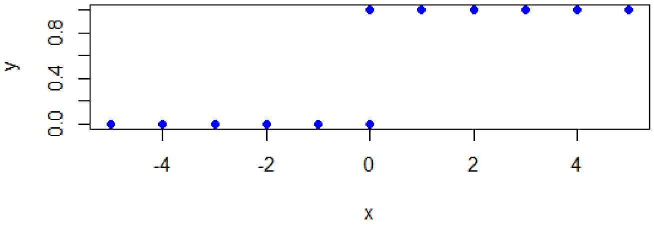


Figure 4.2: Quasi complete separation example: the dataset on the left, the dependent variable prediction on the right

exact logistic regression, it is essential to emphasize that it is computationally feasible only for rather small samples, or when the number of cases on one the two outcomes is small."

In R, a (prototypical) package (named `elrm` proposing an implementation for this approach was found, but at the date it wasn't maintained up-to-date and it is obsolete.

So, it was impossible (both taking into account the size of our dataset and the maturity of the R package for exact logistic regression) to adopt this approach.

2. Penalized Maximum Likelihood Estimation (PMLE). Based on Firth (1993), this approach produces estimates with a relatively little bias, even under extreme conditions: the BR estimator resulting by penalizing logistic regression log-likelihood by Jeffreys prior takes always finite values, even when some of the MLE are infinite (as in separation situations).

   The estimation bias can be reduced in several ways: by solving the mean BR adjusted score equations (Firth, 1993; Kosmidis and Firth,

2020), by solving the median BR adjusted score equations (Kenne Pagui et al., 2017) or by subtracting an estimate of the bias from the MLE (Cordeiro and McCullagh, 1991). Computationally, a quasi Fisher scoring algorithm is used.
Moreover, this second approach is computationally feasible even for large samples (differently from exact logistic regression). In R, there are a few libraries implementing PMLE:

(a) logistf: this package implement Firth's BR logistic regression via the function logistf.

(b) brglm2, a package directly implemented by Kosmidis, one of main authors in PMLE field.

After careful analyses and prototypes, we preferred to use brglm2 for the integration between MOB and PMLE, to adopt the mean bias-reducing adjusted score approach (Firth, 1993), ensuring finite estimates for some GLM (and among them Binomial response) (Kosmidis et al., 2020; Kosmidis and Firth, 2020), even in cases where MLE are divergent (as for separation problem). This adds another context to the list of the different application contexts where the reduced-bias estimator can favourably be used (see, for example, Kosmidis et al., 2020 for a list of diverse application areas).
Integration between brglm2 and partykit was easy in what concerns training phase: we simply defined a new fit function *mob.brglm2* (Listing 4.1):

```r
mob.brglm2 = function(y, x, start = NULL, weights = NULL, offset = NULL, ...) {
  glm(y ~ 0 + x,
    family=binomial,
    start = start, method = "brglm_fit",type="AS_mean", ...)
}
```

Listing 4.1: Defining brglm fitting function to be used in MOB

and executed MOB with it (Listing 4.2):

```r
home.diff.fit=mob(result~
              eFG.t +TOR.t + OReb.t + FTR.t | diff,
              data=dataset.training,
              fit=mob.brglm2
            )
```

Listing 4.2: Invoking MOB with brglm fitting function

For the testing phase, an ad hoc prediction function has been implemented (Listing 4.3) to get around some issues that should have been solved in the further releases of the package.

```
# MOB+brglm prediction: standard predicates with response type
    ‘‘node’’ works
home.diff.fit.node=partykit:::predict.glmtree(home.diff.fit,
    newdata = dataset.testing, type = "node")

# so we will use "composite" approach, starting from leaves
brglm2_probs=f.mob.calc.probs(home.diff.fit,home.diff.fit.node,
    dataset.testing)  # finding testing observations probs

# returns the dataframe of probabilities for test observations
f.mob.calc.probs=function(mob_tree,mob_df,test){
  leaves=nodeids(mob_tree, terminal = TRUE)  # leaves ID
  n=length(leaves)  # number of leaves
  ret <- data.frame(A= numeric(0), B= numeric(0))
  for (i in 1:n){
   # calculating node probs
    node.probs=f.mob.node.prob(mob_tree,mob_df,test,leaves[i])
    ret=rbind(ret,node.probs)
  }
  ret= arrange(ret,ID)  # sorting using observation ID
  ret=unlist(ret$value)  # we need just values
  return(ret)
}

# returns probs for observations classified in node n
f.mob.node.prob=function(tree,nodes_df,test,n){
  idx=filter(nodes_df,node==n)
  obs=test[rownames(idx),]
  coef=coef(tree,node = n)  # leaf predictors estimated coeffs
  # calculating node obs probs
  prob=f.logit.find_mob_diff_prob(obs,coef)
  aux =unlist(prob)
  ret=enframe(aux)
  return(ret)
}

# calculating node obs probabilities
# here there's reference to our custom  dataset
f.logit.find_mob_diff_prob=function(leaf_obs,leaf_coef){
  ff=leaf_obs[,c(10,12,14,16)]  # node obs Four Factors values
  intercept=as.double(leaf_coef[1])  # node estimated intercept
  coef.ff=leaf_coef[2:5]  # node estimated Four Factors
    coefficients
  num_obs=dim(leaf_obs)[1]  # number of obs in the node
  log_fun=rep(0,num_obs)   # initializing probabilities vector
  for (i in 1:num_obs){  # for all observations in the node
   # calculate logit exponent
     es=f.logit.calculate_diff_esp(intercept,coef.ff,ff[i,])
     # calculate logit probability
    log_fun[i]=exp(es)/(1+exp(es))
  }
  return(log_fun)
}

# applying logit exponential formula
f.logit.calculate_diff_esp=function(intercept,ff_coef,ff_values){
  esp=intercept
  for (i in 1:4){
    esp=esp+as.double(ff_coef[i])*as.double(ff_values[i])
  }
  return(esp)
}
```

Listing 4.3: Implementing custom MOB-brglm prediction

It can be meaningful to underline how, after some communications, Prof. Zeileis (one of MOB authors) realized it was easy to integrate `brglm2` in MOB, and natively made it.

4.4 Results

In this section, we summarise the results obtained by applying the MOB algorithm with brglmFit (from package `brglm2`) as fitting function to our NBA dataset.
We considered two models:

1. the first model aimed at helping the coaching staff when preparing the home match; the winning probability of the home team is modelled as function of the home team's Four Factors.

2. the second model, aimed at supporting the preparation of the away matches, based on the opponent's Four Factors as predictors. In this second model, the winning probability of the away team is modelled as function of the away team's Four Factors.

In both cases, the partitioning variable used in MOB is the *diff* index, which is the difference of victory relative frequencies between the home and the away team, as defined in equation (4.1).

The choice to model home and away matches separately is due to two main reasons: (1) we aimed to deliver two separate tools to the coach or team management as support when deciding the best game strategy, as explained above; and (2) including all the information for the home and away teams in one single dataset (and incorporating team information into the model) resulted in a model able to fit the data nearly perfectly (in particular, including the first factor of both teams makes it very easy to identify which team scored the most).

4.4.1 A model for the home matches

In this subsection we analyse a model suitable for preparing home matches. The dependent variable Y is the match *result*, with possible categories 1 = won and 0 = lost, referred to the home team; the predictors are the home team's Four Factors; partitioning is based on the *diff* index.

Fitting the 'home model' to the training set

The model estimated on the training data is fitted with a pruning asking for at least 1000 observations in a node to be splitted (Listing 4.4).

```
home.diff.fit=mob(result~ eFG.t +TOR.t + OReb.t + FTR.t|diff,
                data=dataset.training,
                control=mob_control(minsize=1000),
                fit=mob.brglm2
              )
```

Listing 4.4: Fitting MOB-BRGLM home model

Figure 4.3 displays the tree built using $diff$ as partitioning variable, as reported in each intermediate node together with the p-value of the instability test. The tree terminates with 5 leaves, each of them showing the number of observations classified as belonging to the leaf, together with the coefficients estimated for its own logistic regression model, which is fitted to the observations in that node.

Figure 4.4 shows the specific model associated to each leaf summarised via four spinograms, one for each factor (shooting, turnovers, rebounds and free throws, respectively), offering a simple tool to better interpret the results.

The spinogram is a plotting style provided by the `partykit` package for logistic regression, which shows how the success probability of the dependent variable $Pr(Y = 1)$ (on the y-axis) varies on the basis of an independent variable on the x axis. The stacked bars' widths correspond to the relative frequencies of the values of the independent variable (in classes), while the bars' heights correspond to the conditional relative frequencies of $Y = 1$, which are the success probability estimates, in every class of the independent variable (defined according to a quantile subdivision, here quartiles).

Looking at the first leaf on the left (Node 3) in Figure 4.4, which includes the observations with negative *diff* (the away team won more than the home

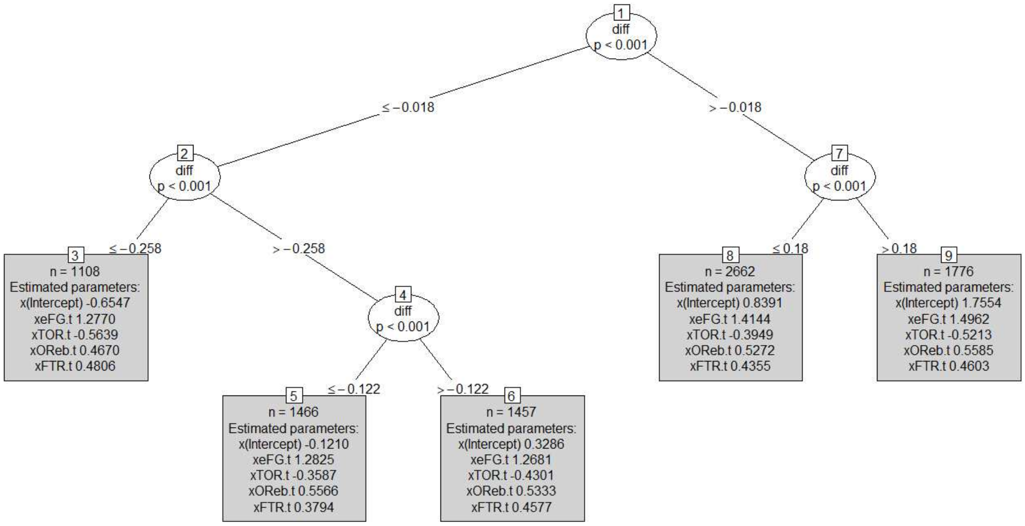


Figure 4.3: MOB tree for the home team model (training set): in the leaves, coefficient's estimates are shown together with each subset size n

team up to the current match, so we can suppose that the away team is stronger than the home team), we can observe that the model highlights the difficulty for the home team to find a winning strategy. Only a really high shooting percentage can give the home team a hope of victory. This suggests a really high weight of the shooting factor in the game strategy in this situation.

In the opposite extreme (first leaf on the right), Node 9 contains the observations with positive *diff* (> 0.18). For the games included in this node, we can suppose that the home team is stronger than the away team. Here, the model confirms that, as expected, winning probabilities are really high, regardless of the Four Factors. It is a good situation in which the coach can try new strategies and tactics, or let play second lines allowing first lines to stand.

Middle nodes include more balanced matches, with fighting teams having similar victory percentages in the matches played up to the current match. For example, looking at Nodes 5 and 6, the spinograms suggest that a good shooting factor is mandatory to win, otherwise, with high probability, the home team will be defeated; for the other three factors, winning probability tends to be substantially flat.

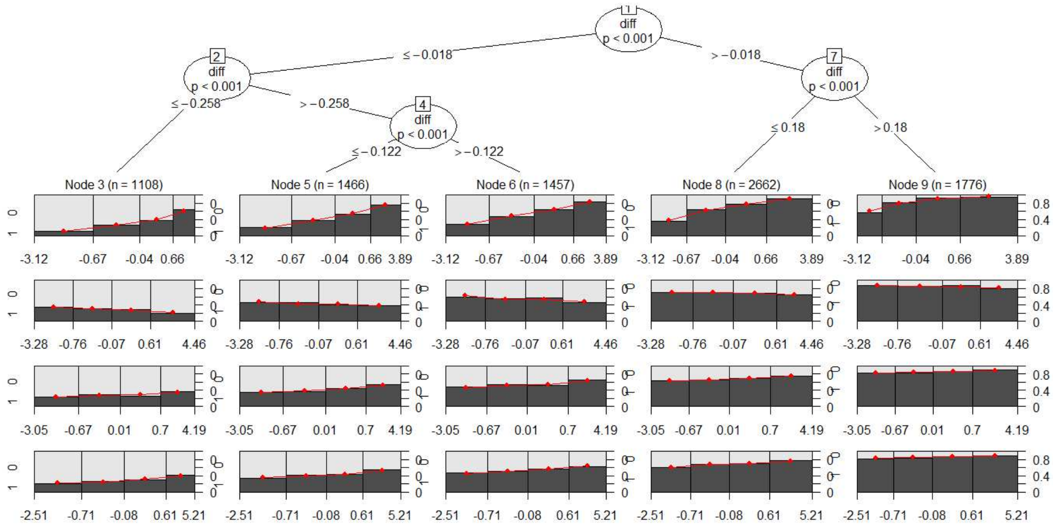


Figure 4.4: MOB tree for the home team model (training set): for each leaf, a set of four spinograms, one for each factor (from top to bottom: shooting, turnovers, rebounds and free throws, respectively), is depicted

Testing the 'home model'

Testing is based on the test set, containing 1,100 matches (11.5%) from the second half of the regular seasons 2018–2019 and 2019–2020, and probabilities have been calculated using custom implementation previously described.

Having an unbalanced dataset (in our training set the home teams win more often than the away teams, with 59.81% of wins, that is $Y = 1$), we considered that prevalence in prediction.

The results from the ROC/AUC analysis in the test set, performed using the `pROC` package (Robin et al., 2011), show a good fit, as displayed in Figure 4.5: AUC equals 0.85, and for the selected model (with a classification threshold equal to 0.688), the accuracy is 0.7445. This high value of the accuracy index, as well as those described in the followings, is justified because this model and the subsequent ones are not models for predicting the match result but for explaining the influence of the Four Factors on the match result; indeed, the covariates refer to to the current game.

Size and accuracy for single leaves, i.e. for each locally fitted model, are reported in Table 4.1.

As expected, accuracy is higher in extreme situations (when one of the two teams had much better performance than the other one up to the current match), and is lower (but still high) in balanced games, when the strength

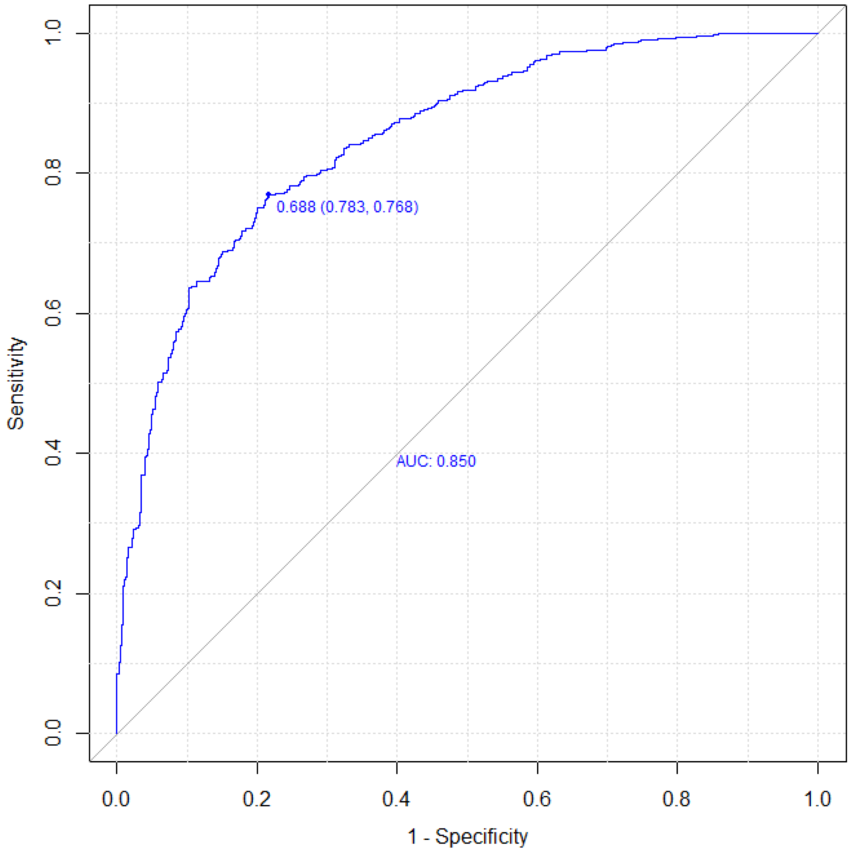


Figure 4.5: ROC/AUC analysis for the ‘home model’ (test set)

of the two teams is similar (*diff* is around 0).

4.4.2 A model for the away matches

In this subsection we fit a model suitable in approaching the away matches. The focus is here on the away team so the dependent variable is the away team *result* and the predictors are the away team’s Four Factors; partitioning is still based on the *diff* index.

Fitting the ‘away model’ to the training set

The model estimated on the training set for away matches is produced by following code:

```
opp.diff.fit=mob(result~eFG.o +TOR.o + OReb.o + FTR.o|diff,
```

Table 4.1: Size and accuracy of local models for the “home model” (test set)

node	num	accuracy
3	176	0.7841
5	169	0.7219
6	191	0.6387
8	334	0.7096
9	230	0.8043

```
                     data=dataset.training,
                     control=mob_control(minsize=1000),
                     fit=mob.brglm2)
```

Listing 4.5: Fitting MOB-BRGLM away model

and is displayed in Figures 4.6 and 4.7.

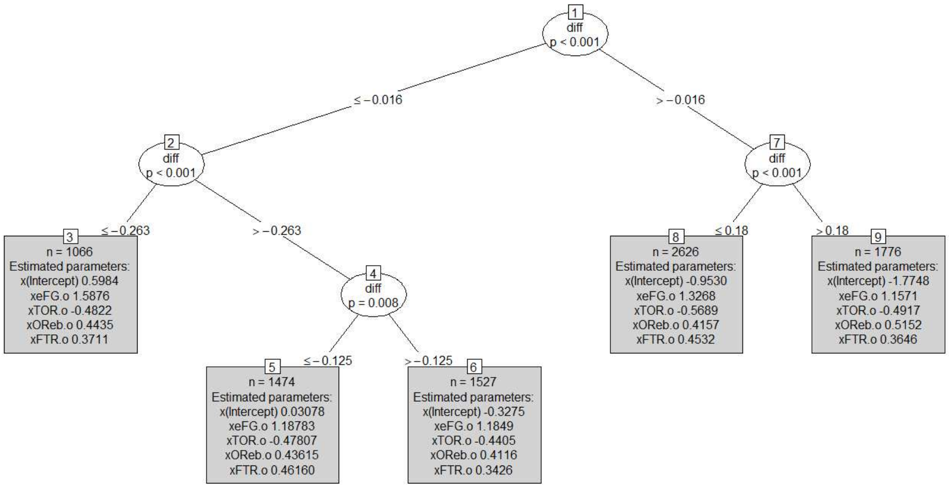


Figure 4.6: MOB tree for the away team model (training set): in the leaves, coefficient’s estimates are shown, together with each subset size n

Results from the home and away models are very similar, as expected, in terms of both partitioning and estimated models.

Testing the ‘away model’

Results from the ROC/AUC analysis in the test set is displayed in Figure 4.8,

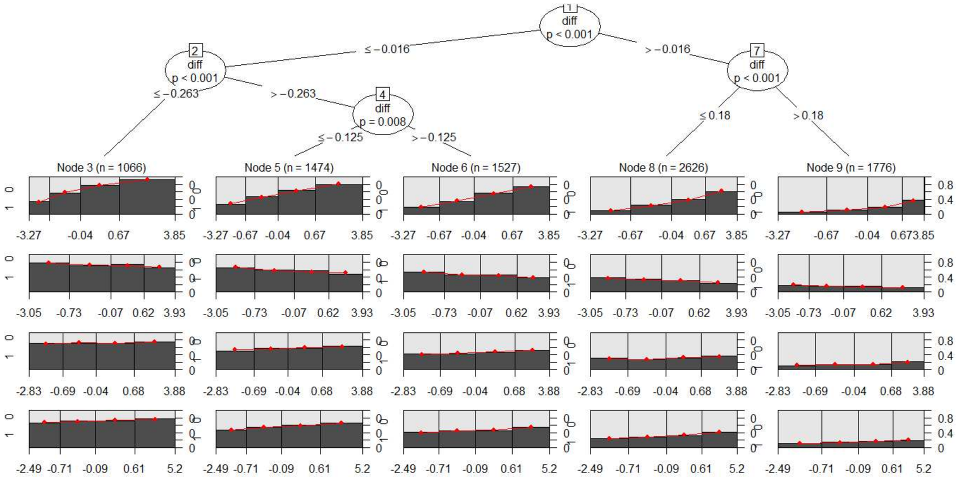


Figure 4.7: MOB tree for the away team model (training set): for each leaf, a set of four spinograms, one for each factor (from top to bottom: shooting, turnovers, rebounds, free throws, respectively), is depicted

with an AUC of 0.833; for the selected model (with classification threshold equal to 0.398), accuracy is 0.7509.

Size and accuracy for single leaves, i.e. for each local fitting model, are reported in Table 4.2.

Table 4.2: Size and accuracy of local models (testing dataset), the away team

node	num	accuracy
3	169	0.7515
5	172	0.7267
6	200	0.755
8	329	0.7173
9	230	0.813

4.4.3 The GLM tree with an offset term

The results in Subsections 4.4.1 and 4.4.2 suggest that the splits in both trees are mainly driven by differences in the intercept and that the remaining differences are mostly random variations. This is confirmed by Figure 4.9, representing the estimated Four Factors' coefficients as functions of the node

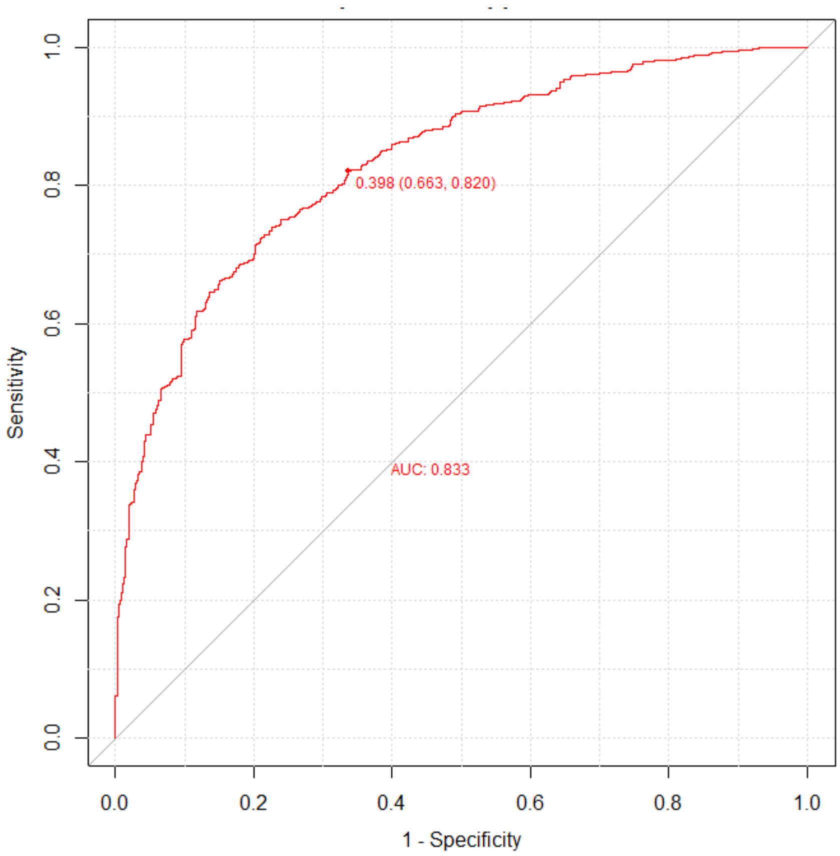


Figure 4.8: ROC/AUC analysis for the 'away model' (test set)

mid-points of the partitioning variable (the *diff* index).

Figure 4.9 suggests that the the splits in both the home and away models are driven by differences in the intercept and not by differences in the Four Factors' coefficients, which appear to be nearly constant across different nodes. This implies that the winning probability increases or decreases with the *diff* index, but the Four Factors do not seem to contribute much.

In order to reveal any structural difference among the impacts of the Four Factors on the winning probability in the tree leaves, we included the *diff* index as an offset term in the estimation of the GLM tree, that is keeping the parameter of the *diff* index fixed globally for all observations while investigating, by estimating the tree, if and how the Four Factors' parameters vary locally, that is across subgroups.

Indeed, an offset term can be used in GLMs in order to include an ad-

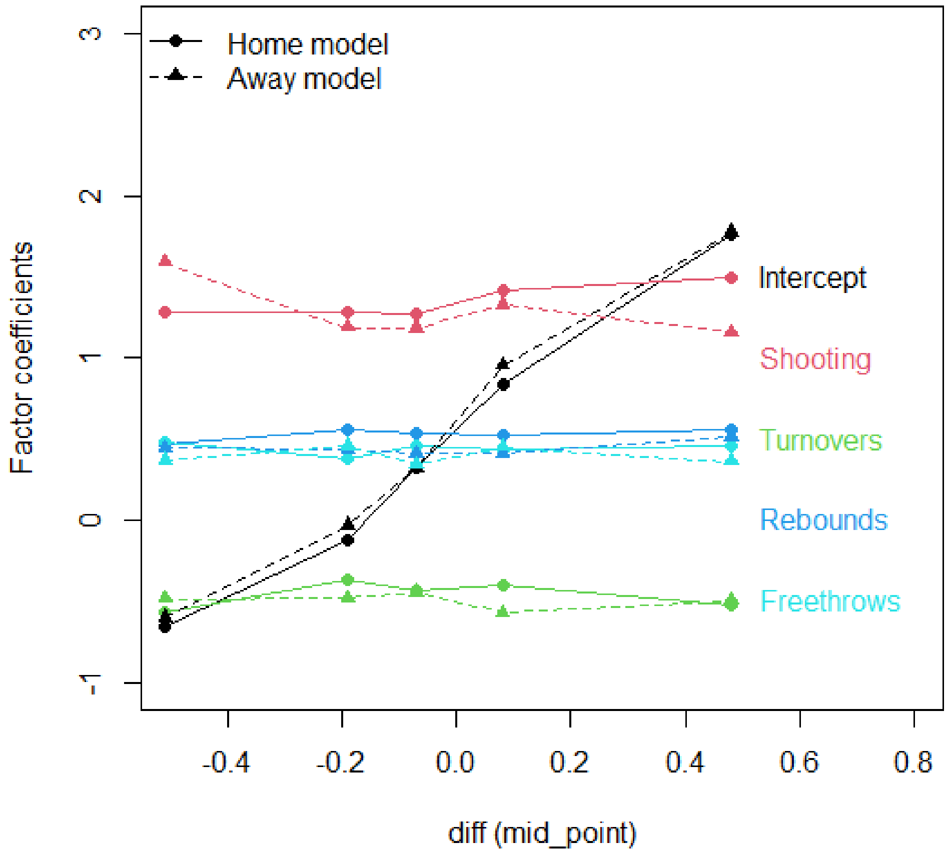


Figure 4.9: Estimated Four Factors' coefficients (y-axis) as a function of the node mid-points of the partitioning variable (*diff* index, x-axis)

ditional regressor in the linear predictor when the effect of this regressor is fixed or known a priori. Therefore, the coefficient of this additional term is not estimated but fixed. In our situation, in order to quantify the global effect of the *diff* index we first estimated, on all observations, a logistic regression model with the winning probability as the dependent variable and the *diff* index as the sole regressor. The resulting estimated coefficient of the *diff* index measures the (known a priori) contribution of *diff* and, multiplied by the *diff* index itself, has then been used as the offset term in the GLM tree, where the aim was to investigate the local effects of the Four Factors in subgroups of observations identified by the partitioning variable (see Listing 4.6

```
# a) estimate the additive model in a first step
log.fit_ea.diff.noha=glm(result~ ea.diff.noha,
                         family=binomial,
                         data=dataset.training)

# b) preparing offset term: product of diff index with its coefficient
log.coef_ea.diff.noha=coef(log.fit_ea.diff.noha)
offset_ea.diff.noha=log.coef_ea.diff.noha[2]*dataset.training$ea.diff.noha   # multiply ep.diff.noha by its coeff

# c) invoking glmtree with offset
m_ea.diff.noha_h=glmtree(result~ eFG.t+TOR.t+OReb.t+FTR.t|diff,
                         data =dataset.training,
                         offset=offset_ea.diff.noha,
                         minsize=1000,
                         family=binomial,
                         method="brglmFit")
```

Listing 4.6: Using offset in home model

This was repeated for both home and away GLM tree models; results are reported in Figures 4.10 and 4.11, which suggest that in both situations, only one split is enough to distinguish structural differences in the impact of Four Factors on the winning probability.

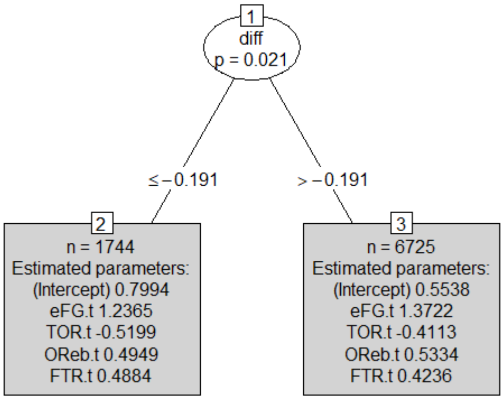


Figure 4.10: MOB tree for the home team model (training set) with the *diff* index as an offset term

Figure 4.12 shows the results from the ROC/AUC analysis in the test set for the home model (left) and the away model (right). AUC equals 0.817 and 0.814 in the home and in the away model, respectively. For the se-

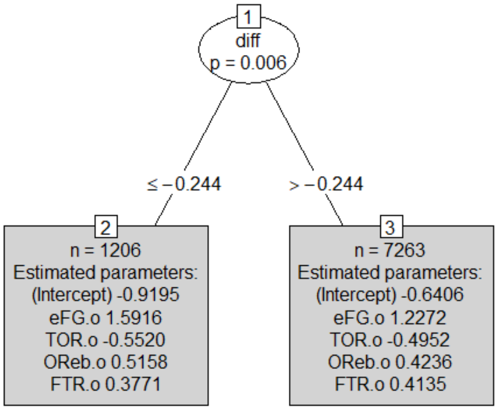


Figure 4.11: MOB tree for the away team model (training set) with the *diff* index as an offset term

lected models (classification thresholds equal 0.633 and 0.447, respectively), accuracy is 0.7418 in the home model and 0.7445 in the away model.

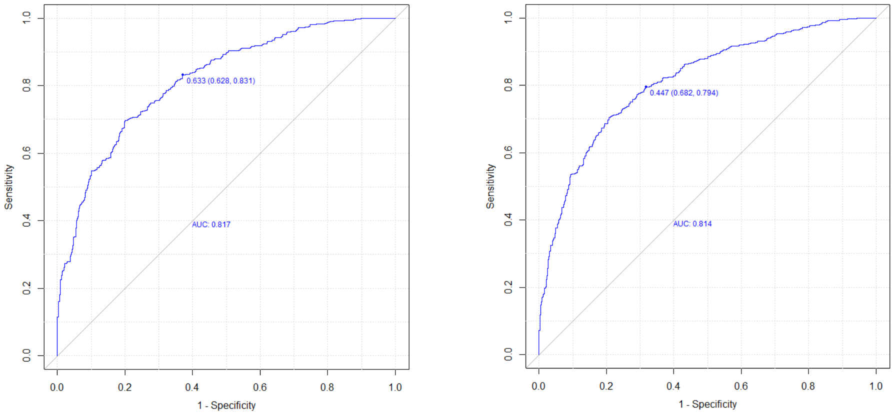


Figure 4.12: ROC/AUC analysis for the 'home model' (left) and the 'away model' (right), with the *diff* index as an offset term (test set)

The two trees are very easy to read and their straightforward interpretability is an added value when these tools are delivered to coaches or team management as support for their strategic decisions. The home model in Figure 4.10 will drive decisions when our team is playing a home match, while the away model in Figure 4.11 will support decisions if our team is playing away.

First of all, interestingly in both trees the threshold is not zero but is negative. This suggests that the threshold is favorable to the home team; indeed, in both trees, the leaf on the right contains matches where the home team is expected to have an advantage over the away team. On the contrary, in all the matches belonging to the left leaf, the home team is expected to have a disadvantage. The bonus for the home team, related to the negative threshold, is due to the presence, in the right leaf, of matches with a slightly negative *diff* deriving from a relative frequency of victories for the home team -up to that match- (slightly) lower than the relative frequency of victories for the away team.

As regards the estimated coefficients, results show that, in general, the first factor offers a high contribution in winning the match, as expected, while the contributions of other three factors are lower and roughly equal to each other. This contradicts, in some way, the weights proposed by Oliver (Table 2.1).

Looking for differences in the estimated coefficients of the right and left models, we see that in matches where our team is expected to have an advantage (right leaf in the home model in Figure 4.10 if we are playing at home and left leaf of the away model in Figure 4.11 if we are playing away), the impact of the first and third factors on our winning probability is higher than in matches where our team is expected to suffer more. On the contrary, the impact of the fourth factor is lower. In other words, the best strategy for winning a game when we are the favorite team (according to the past history of games as measured by the *diff* index) is to focus on offensive abilities. On the other side, in matches where we are not the favorite team (left leaf in Figure 4.10 and right leaf in Figure 4.11), offensive skills are less crucial and a defensive strategy, blocking the opponent's attack, and counting on free throws, can lead to win.

In future research, these simple models can be easily and profitably integrated with other partitioning variables, if available, for example, accounting for differences in teams' positions in the conference (or division) rankings or in some offensive or defensive characteristics.

4.5 Conclusions

In this chapter, we applied MOB to a large NBA dataset in order to understand how the Oliver's Four Factors importance changes on the basis of the balance of power between two teams competing in each match.

To do this, we analysed two different models with the result of the match as the dependent variable. The first model can be used by the coaching staff to prepare home matches (using the home team's Four Factors as predictors), the second one to make decisions about the best strategy in away matches (using the opponent team's Four Factors).

In both cases, we used the *diff* index as the partitioning variable; it is defined for each match as the difference between victory relative frequencies for the home and the away team considering previous games of the same season.

The whole dataset was used to calculate the *diff* index, but only matches of the second half of each season were used as observations in the model; the second half seasons from 2004–2005 to 2017–2018 were used for training, while the second half seasons 2018–2019 and (part of) 2019–2020 were used for testing.

In order to solve convergence problems of Maximum Likelihood estimation due to (quasi) separation issues, we proposed a solution that integrates MOB with Bias Reduction GLM estimation provided by the `brglm2` package. The proposed approach ensures finite estimates in (quasi) separation situations.

The results illustrate how the local models provided by MOB offer a great interpretation power: they show how the Four Factors' weights can change on the basis of the difference in class between the two teams involved in a match. Therefore, they can constitute a valuable tool for basketball coaching staff when preparing for a match.

Further research will be devoted to include other partitioning variables in the models in order to add information on the offensive or defensive abilities of the two teams, or external information, such as, for example, changes in the teams' rosters, changes in the coaching staff, position in the conference or division rankings. Other definitions of the *diff* index could also be explored, varying the k parameter in equation 4.1 in order to account for a rolling estimate of the *diff* index (like we did in Chapter 5).

This page intentionally left blank.

Chapter 5

Feature analysis for basketball outcome prediction by means of Deep Learning

Remarks Some of the contents of this chapter will be presented in:

IES 2022 - Innovation & Society 5.0
University of Campania "L. Vanvitelli", Italy, 27 January – 28 January 2022
in a presentation entitled:
Feature definition for NBA result prediction through Deep Learning
M. Migliorati, E. Brentari

and have been made public in arXiv:
http://arxiv.org/abs/2111.09695
in a manuscript entitled:
Features selection in NBA outcome prediction through Deep Learning
M. Migliorati

5.1 Introduction

This chapter is devoted to the problem of predicting the winner of matches in NBA, focusing on features selection. It is shown how, for outcome prediction classification problem, a careful definition of single features to be used in machine learning techniques, and Deep Learning (Goodfellow et al., 2016; Chollet and Allaire, 2018) in particular, can produce predictions with high quality, comparable to top quality predictions reported in literature, where a huge number of regressors is often used.
Three approaches have been selected for features' definition:

1. The Elo (from the name of its creator) rating system (Elo, 1978). Originally defined for rating chess players, today it is widely used in several board and card games, online games and sports. The Elo rating system, with some adjustments, has been applied to many sports, mainly for tournament rating predictions: football (Eetvelde and Ley, 2019; Hvattum and Arntzen, 2010; Leitner et al., 2010; World Football Elo Ratings, 2021), tennis (Angelini et al., 2021), Australian football (Ryall and Bedford, 2010), ice hockey (World Football Elo Ratings, 2021), American football (Silver, 2014) and, of course, to NBA basketball (Silver, 2015, 2020a).

2. the difference of the relative frequency of victories for the two teams involved in a match, named $diff$ (Migliorati et al., 2020).

3. Oliver's Four Factors (Oliver, 2004; Kubatko et al., 2007): few indexes aimed to identify success keys, as described in Section 2.4

The first two approaches are directly based on the teams' strength definition, the third one is based on a synthesis of several box score statistics, instead. All these approaches have been adapted to calculate the features' values before a match (ex-ante values), taking into account:

- Both historical (considering all past games) and dynamic (averaging a feature on some prior matches) definitions.

- Regression to mean (Galton, 1889), i.e. the trend for extreme values of getting closer to average in repeating experiments; this concept is particularly important in NBA championship, where after each season's end there is an explicit attempt of strength re-balancing among teams (draft mechanism).

- The home court factor, appearing so important in NBA (Harville and Smith, 1994; Jones, 2007).

The original NBA dataset described in previous chapters, including all the NBA regular seasons matches from 2004-2005 to 2019-2020 (until 11/03/2020, when NBA was stopped for some weeks due to Covid19), and counting 19.138 observations (one for each match), has been enriched with these features, and then used for both training and testing Deep Learning models.

Deep Learning is a specific subfield of machine learning, and is based on using neural networks composed by (possibly) several layers (this is the reason why the approach is called deep, no relevance to a deeper data understanding level).

The models have been developed in a particular Deep Learning echosystem (Chollet and Allaire, 2018) in R, using `Keras` package (Allaire and Chollet, 2021) via RStudio (RStudio Team, 2021). The nets, calibrated to produce models with a good prediction quality, are built considering the two hyperparameters (i.e. the number of layers and the numbers of units for each layer) small in size, a natural consequence of the small number of features. This chapter's original contributions are:

1. the comparison of single-feature and box-score based models, showing how the latter have a lower prediction quality (a possible symptom of the fact that they are close to their limit in outcome prediction). Moreover, the single-feature approach enables the usage of really simple neural networks (typically with just one hidden layer), asking for a reduced computational power and producing results comparable, for the specific problem, to more complex network architectures:

2. the idea of maintaining separated home and away data, obtaining new statistics (e.g. Elo home and Elo away ratings) to base our models on.

The chapter is organized as follows: Section 5.2 is devoted to describe the dataset and its features; Section 5.3 summarizes Deep Learning approach; Section 5.4 reports the outcome predictions produced in applying Deep Learning, and Section 5.5 ends the chapter proposing some conclusions and ideas for further enhancements.

5.2 The dataset

The dataset includes all NBA regular seasons matches from 2004-2005 to 2019-2020 (until 11/03/2020, when NBA was stopped for a period due to

Covid19), and it counts 19.138 observations, one for each match. During the 16 seasons taken into account, some franchises have changed name or court, so the total number of different teams in the dataset should be 34. For our analyses we adopted the most recent names (i.e. Brooklyn Nets, New Orleans Pelicans and Oklahoma City Thunder) for each franchise affected by changes in the considered period, reducing the number of teams in the dataset to the canonical 30.

5.2.1 Features' definition

5.2.1.1 The Elo rating

The Elo rating system (Elo, 1978) has been originally defined for calculating the strength of players in zero-sum games (i.e. games where a player gains exactly what its opponent loses) as chess, the sport for which this system was created by Arpad Elo.

Each player is assigned a rating constituted by a single number: new players have an initial default rating (that can change on the basis of the considered organization), and the difference in the ratings of the opponents of a match is used to establish the probability of the match result. After every match, the winning player will gain a certain quantity of points (and the defeated player will lose the same quantity) depending on their pre-match rating difference; moreover, the system is built in an "asymmetric" way: the gain for victory of the player with the highest rating is smaller than the eventual gain for victory of the player with the lowest rating.

More formally: if before a match Player1 has a rating $R1$ and Player 2 has a rating $R2$, the probability for Player 1 of winning the match (event $p1w$) is modeled as a logistic curve as follows:

$$P(p1w) = \frac{1}{1 + 10^{\frac{-(R1-R2)}{400}}} \tag{5.1}$$

and the probability of victory for Player 2 (event $p2w$) is modeled as:

$$P(p2w) = \frac{1}{1 + 10^{\frac{-(R2-R1)}{400}}} \tag{5.2}$$

where the value 400 is historically used in Elo (Fig. 5.1 shows the impact of that parameter on the slope of the sigmoid curve). Probabilities are 0.5 if Player1 and Player2 share the same rating, and in general $P(p1w) + P(p2w) = 1$.

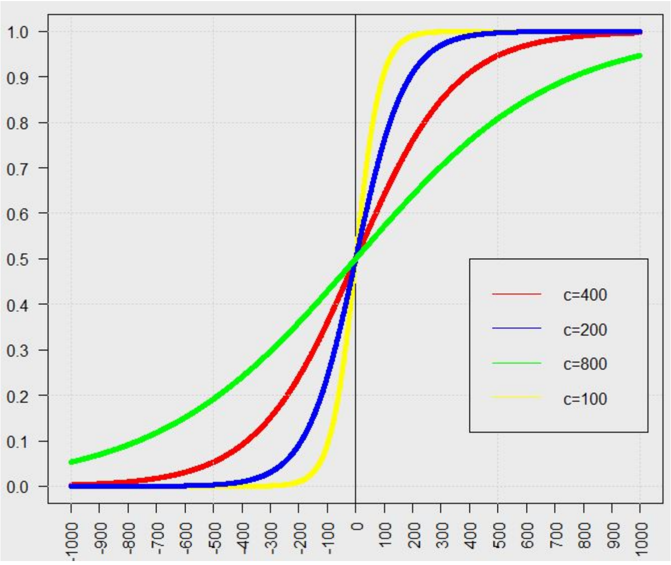


Figure 5.1: Elo logistic curves on the basis of different logistic parameter value (i.e. the denominator of the exponent in equations 5.1 and 5.2); 400 is the default in chess.

Let S be the result of a match: for games without the possibility of draws (as basketball is) S is 1 for Player1 victory, 0 instead when Player 2 won).[1] After the match, the ratings of the 2 players will be updated as follows:

$$R1^{'} = R1 + K * (S - P(p1w)) \tag{5.3}$$

$$R2^{'} = R2 + K * (S - P(p2w)) \tag{5.4}$$

where K is a parameter addressing how strongly a result will affect ratings' update: a small K means that ratings remain almost stable, a high K means strong impacts on rating change.

An example: if the ex-ante ratings are 1500 for Player1 and 1400 for Player2, and Player1 won the match, the updated ratings will be:

- for K=5, Player1 =1502 and Player 2=1398
- for K=50, Player1 = 1518 and Player2 =1382

Viceversa, if Player2 (i.e. the underdog) won the match, larger variations in ratings will be produced:

[1] For sports as football, where a draw is admitted too, S can assume the three possible values 1, 0.5, 0.

- for $K = 5$, Player1 =1497 and Player 2=1403
- for $K = 50$, Player1 = 1468 and Player2 =1432

In the chess world the logistic parameter is set equal to 400, to have P(p1w)= 0.75 and P(p2w) =0.25 when the difference in ratings between the two players is equal to 200, following the original suggestion by Elo. Moreover, in Elo system the initial rating is not important (the difference of rating between two players are considered) if there are not situations introducing inflation or deflation in the system; this is the case with our dataset, because the group of teams is closed. The difference in Elo ratings between the two teams fighting in a match will be the first feature used in the present study: for the initial ratings we will follow Silver (2015) and 1300 will be used, for the logistic parameter the classical value of 400 will be maintained.

5.2.1.2 The difference in relative victory frequencies

A second feature directly quantifying the strength of opposing teams is the difference of their relative victory frequencies, as expressed in chapter 4 in equation 4.1.
Named *diff* (Migliorati et al., 2020), it is a clear and concise way for showing the difference in class between the two teams, providing an analytical definition for a classic rule of thumb often used in naive fan predictions (the favorite is the team that won more in the past).

5.2.1.3 Four Factors

The Four Factors (Oliver, 2004; Kubatko et al., 2007) is the set of statistics considered keys to success, built on top of classic *box score* analytics (it is not a direct expression of the strength of a team) and already described in Section 2.4.

5.2.2 Features' characterization

The features introduced in Section 5.2.1 have been calculated ex ante, i.e. considering only information from prior matches, to make them suitable for outcome predictions. Moreover, they have been calculated in multiple ways, taking into account the periodicity (historical VS dynamic approach) and the separation of data on the basis of the court to calculate features.

5.2.2.1 Periodicity

The features used as covariates in models for predicting the outcome of a match have been calculated both from an historical (considering all the prior matches included in the dataset) and dynamic perspective (averaging on a subset of prior matches). Under the so-called historical perspective, if we have to predict the outcome of the game g, a generic feature $f(t, g)$ for a team t, with t ranging from 1 to the number of teams T, will be calculated as in Equation 5.5:

$$f_{(t,g)} = \frac{\sum_{i=1}^{g-1} f_{(t,i)}}{g-1} \tag{5.5}$$

under the dynamic perspective, $f(t, g)$ will be calculated as in Equation 5.6:

$$f_{(t,g)} = \frac{\sum_{i=g-d}^{g-1} f_{(t,i)}}{d} \tag{5.6}$$

where d is the depth, i.e. the number of prior games to be considered in calculating the average of f.

Historical approach: regression to mean When computing the historical $f(t, g)$ (equation5.5), the regression to mean was implemented: at each season starting, the features' values are reinitialized, taking into account a percentage of their past average.

Let us consider the last N regular seasons $s_1, .., s_N$, each one composed by m_k matches, where k renages over the number of seasons from 1 to N; moreover, let us consider the generic feature f of the team t, denoted as f_t, with t ranging over the number of teams, from 1 to T.

The value of a generic feature f_t for the team t for the first match of the new regular season s_{N+1}, denoted as f_{t,s_{N+1}^1}, is calculated as in 5.7, adding a proportion 1-P of the value of the feature after the last match of the previous season S, i.e. $f_{i,s_N^{m_N}}$ to a proportion P of the average of the values of the feature calculated considering all past matches for all teams:

$$f_{t,s_{N+1}^1} = f_{t,s_N^{m_N}} * (1-P) + \frac{\sum_{j=1}^{T} \sum_{k=1}^{N} \sum_{z=1}^{m_k} f_{j,s_k^z}}{T * \sum_{z=1}^{N} m_z} * P \tag{5.7}$$

where P is the proportion of regression to mean to be considered. A regression proportion equal to 0 reduces equation 5.7 to equation 5.8

$$f_{t,s_{N+1}^1} = f_{t,s_N^{m_N}} \tag{5.8}$$

where the first value of the feature for the new season of a team is equal to the last value of the feature of the previous season for that team; this means to have continuity among seasons, without any regression to mean: like a single, long season for the entire period 2004-2020.
At the other opposite, when the regression proportion P is equal to 1, equation 5.7 is reduced to equation 5.9

$$f_{t,s_{N+1}^1} = \frac{\sum_{j=1}^{T} \sum_{k=1}^{N} \sum_{z=1}^{m_k} f_{j,s_k^z}}{T * \sum_{z=1}^{N} m_z} \tag{5.9}$$

meaning that the starting features' values for each season are the same for every team, equal to the mean of all past values; it is a complete regression to mean for all teams.
The mechanism of regression to mean is suitable not only from a statistical point of view (Galton, 1889), because extreme values tend to become closer to the mean in new observations, but it seems particularly suitable for NBA (Silver, 2015), where the draft mechanism is adopted: at the end of each season, the worst classified teams will have the precedence in selecting new players; at the opposite, the best classified teams will be the last in such a choice. The draft mechanism is not perfect, but ensures a certain balancing among teams, see Figure 5.2 where the number of playoff accesses for each team is used as a measure of the efficiency of draft mechanism. For a good balancing, each team should have the same number of playoff accesses (about eight in the considered period), with a density curve (depicted in light blue) different from zero only around that value. Instead, there are teams with only one or two accesses, and other teams with 15 accesses. In this work we verified (see Section 5.4) how regression to mean is fundamental for having good predictions quality for models using historical features as regressors.

Dynamic approach In the dynamic approach, new statistics are built considering the average of a feature's value, taking into account only a subset of prior matches. The exact number of matches to be considered in the average[2] is to be identified for obtaining models with the best predictions quality.

[2] Some experiments using the exponential smoothing of some past matches as depth for rolling mean has been implemented, but the fitted models show no improvements on the quality of fit with respect to the approach used in the analysis

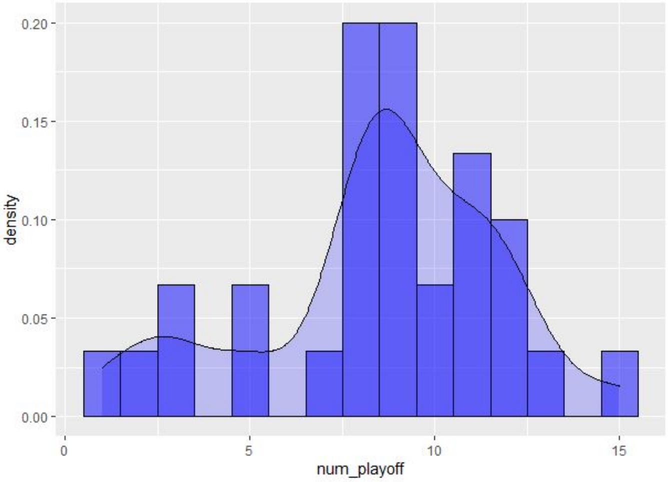


Figure 5.2: Efficiency of NBA draft re-balancing mechanism. On the vertical axis the number of playoff accesses bars and the density function (line) are depicted for seasons from 2004-2005 to 2019-2020.

5.2.2.2 Home VS away court data

In NBA, the home court factor plays an important role (Harville and Smith, 1994; Jones, 2007). The analysis about this topic on the dataset here considered confirms that: on average, home teams win 59.27% of the matches: a summary per season is reported in Table 5.1; home victories percentage is always greater than 57% (apart from the season 2019-2020, for which data are limited to 20/03/20), exceeding in several seasons the 60%.

This information can be useful in features' calculation. To this purpose:

- Elo definition is usually modified to take into account the home court advantage (chess does not care about the home factor, but many other sports, and basketball among them as demonstrated above, should): if home victory is more frequent, the impact on the ratings' update for the home team victory should be decreased.
- All features have been calculated:
  1. as usual, considering all past matches regardless the court where they have been played.
  2. Building two new variants for each feature, calculated considering either only data from the home played matches or only data

Table 5.1: Percentage of home victories per regular season (season 2019-2020 is limited to 20/03/20)

season	home victories %
2004-2005	60.43
2005-2006	60.46
2006-2007	59.15
2007-2008	60.18
2008-2009	60.73
2009-2010	59.40
2010-2011	60.33
2011-2012	58.59
2012-2013	61.19
2013-2014	58.05
2014-2015	57.48
2015-2016	58.86
2016-2017	58.37
2017-2018	57.89
2018-2019	59.27
2019-2020	55.10

from the away matches, respectively. This approach seems to be original for the features we are considering.

Modifying Elo calculation to account for the home advantage Probability definitions (5.1) and (5.2) must be revised to consider the home advantage. The classic approach to do that consists in adding a penalization parameter to the exponent, as in Formulas 5.10 and 5.11; in this way, a home victory will produce a smaller effect on rating updates, balancing the home court factor:

$$P(p1w) = \frac{1}{1 + 10^{\frac{-(R1-R2+HA)}{400}}} \tag{5.10}$$

$$P(p2w) = \frac{1}{1 + 10^{\frac{-(R2-R1-HA)}{400}}} \tag{5.11}$$

This penalization parameter must be carefully quantified because, as shown in Table 5.2, it can play an important role in Elo rating update[3]. The Table 5.2 contains some examples of rating updates for a match involving two teams with same Elo rating (1300) before their match. On the basis

[3] in Silver (2015) home-court advantage quantified as 100 Elo points

Table 5.2: Examples of the impact of home advantage penalization parameter on ratings update for two teams with same Elo rating (1300) before their match. Column Home_adv contains the home advantage, columns P(p1w) and P(p2w) the probability of victory for home and away team, respectively, Column result contains 1 in case of victory of home team and 0 otherwise, Columns newElo contain the update of Elo ratings for player 1 (p1) and player 2 (p2), respectively

Home_adv	$P(p1w)$	$P(p2w)$	result	newElo(p1)	newElo(p2)
0	0.50	0.50	1	1315.00	1285.00
50	0.57	0.43	1	1312.86	1287.14
100	0.64	0.36	1	1310.80	1289.20
150	0.70	0.30	1	1308.90	1291.10
0	0.50	0.50	0	1285.00	1315.00
50	0.57	0.43	0	1282.86	1317.14
100	0.64	0.36	0	1280.80	1319.20
150	0.70	0.30	0	1278.90	1321.10

of the value of penalization parameter (column Home_adv), the result of the application of Equations 5.10 and 5.11 to calculate the probability of victory for the home team (column$P(p1w)$) and for the away team (column $P(p2w)$) changes and consequently, on the basis of the result of the match (column result), the Elo rating values after the match (columns newElo(p1) and newElo(p2), respectively) can greatly change, too.
The Elo formula can be further generalized[4] as in equations 5.12 and 5.13: for each team, two parameters α_{adv} and β_{dis} (prize and penalization, respectively) are added to the numerator of the exponent, and the value assigned to them represents the sums of all advantages and disadvantages for that team:

$$P(p1w) = \frac{1}{1 + 10^{\frac{-(R1 - R2 + \alpha_{1_{adv}} - \beta_{1_{dis}})}{400}}} \tag{5.12}$$

$$P(p2w) = \frac{1}{1 + 10^{\frac{-(R2 - R1 + \alpha_{2_{adv}} - \beta_{2_{dis}})}{400}}} \tag{5.13}$$

In this way, in the Elo equations it is possible to take into account not only the home court factor, but also other factors, to be properly quantified, potentially offering some additional information: for instance player injuries

[4] In this contribution the `Elo` R package (Heinzen, 2020), already offering this possibility, has been used and preferred to other packages (as `Player Ratings` by Stephenson and Sonas (2020), for instance) for its completeness.

(as disadvantage for new injuries, or possible advantage when a top player returns to play, see Marusek, 2021; Hopkins, 2021 for good data sources), logistics (disadvantages due to travels or court altitude Silver (2020b)), number of days among consecutive matches Manner (2016)), referees (Price et al., 2009; Deutscher, 2015).
In this work, only home advantage has been considered, leaving the management of other information to future works.

Considering the court in features definition Typically, the statistics we are taking into account are calculated considering all matches, without reference to the court where matches are played. In effect, the performances of a team can be very different for matches played at home with respect to matches played away. As an example, in Table 5.3 few information about some Detroit Pistons (DET) matches are reported[5]. For each match the home team, the away team and the result (1 means victory for the home team) are specified. Moreover, the victory relative frequency (named *ratio*) is calculated, both as usual, considering all matches (DET ratio column), and differentiating these statistics on the basis of the court (column DET h ratio for the ratio calculated considering only the matches played at home, and column DET a ratio for the ratio calculated considering only the matches played away). After ten matches, the value of the ratio statistics calculated considering all past matches is equal to 0.5 (i.e. Detroit wins one game out of two); but looking to the ratio statistics calculated considering home/away data separation, we have some additional information: that team is really strong at home, with a home ratio equal to 0.8 (they won four out of five games played at home), but instead they are weak when playing away, with an away ratio only equal to 0.2 (they won only one game out of the five played away).
This approach, based on considering home/away data separation in features' calculation (called *the court issue* in the following of the chapter) seems promising, and it will be adopted in this work: besides features not considering the court issue, two new statistics based on the court issue will be calculated.

5.2.3 Dataset structure

Elo, $diff$ and Four Factors have been calculated and then properly lagged to ensure that only ex ante information involving the two teams are used to

[5] Information are taken from starting of season 2004-2005.

Table 5.3: Example of features calculation considering the court factor for some Detroit Pistons matches results. For each match the home team, the away team and the result (where 1 means home team victory) are reported. Moreover, the victory relative frequency is reported, both considering all matches (column DET ratio) and building two separate statistics considering only for home and away matches, respectively (columns DET h ratio and DET a ratio)

home team	away team	result	DET ratio	DET h ratio	DET a ratio
DET	HOU	1	1.00	1.00	
TOR	DET	1	0.50		0.00
DET	PHI	1	0.67	1.00	
LAC	DET	0	0.75		0.50
DEN	DET	1	0.60		0.33
UTA	DET	1	0.50		0.25
DET	MIN	1	0.57	1.00	
DET	IND	0	0.50	0.75	
DET	CHA	1	0.56	0.80	
CHA	DET	1	0.50		0.20

produce predictions.

The whole dataset is composed by 19138 observations, but both lagging and rolling mean calculus can introduce *Not Available* (NA) values, which can indicate either the absence of a value or the impossibility to calculate it [6]. Rows containing NA values have been discarded[7], arriving to a magnitude of about 18.000 observations (depending on how features are calculated).

The dataset has been classically split in training and testing subsets; the training dataset features' values have been standardized, and also the testing dataset features' values have been modified on the basis of mean and standard deviation used in standardizing corresponding training dataset features.

As a summary, Table 5.4 reports the ways adopted in features calculation, taking into account both periodicity (historical VS dynamic) and the court issue as discussed above. Moreover, the common information to be calibrated in features calculation are reported in column information.

[6] For instance, when rolling mean depth is set to n and less than n observations are available.

[7] In particular, the first season is affected by this issue.

Table 5.4: Features variants calculated for Elo, $diff$, Four Factors.

periodicity	court issue	information
historical	not considered	regression to mean %
historical	considered	regression to mean %
dynamic	not considered	rolling mean depth
dynamic	considered	rolling mean depth

5.3 Methods and Models: Deep Learning

5.3.1 Selecting the Neural Network Environment

The machine learning approach used to fit models and predict outcomes is Deep Learning (Goodfellow et al., 2016; Chollet and Allaire, 2018), i.e. Artificial Neural Networks (ANN) involving an (eventually high) number of hidden layers between the first (the input) and the last (the output) layers. Actually, there is a huge number of packages available in R offering ANN's functionalities in general and Deep Learning in particular, as verified exploring the *Comprehensive R Archive Network* (CRAN) repository.

To select the R package most suitable for the analysis, firstly a set of 30 potentially interesting packages has been identified, and then these packages have been compared via an R tool iad hoc implemented to parse CRAN download logs.

Figure 5.3 summarizes the results for packages related to ANN and Deep Learning, considering only the packages with at least 500 downloads in the period 12/02/21-13/01/21, and with a new release in the last year.

Among 30 R packages considered, there are six packages much more downloaded (magnitude of thousands VS hundreds) with respect to others; among these, `h2o` (H2O.ai, 2021, about 25000 downloads, really interesting) and `rminer` (Cortez, 2015, about 1500 downloads) are not specific for Deep Learning/ANN, and `NNet` (Venables and Ripley, 2002, about 100000 downloads) is a didactic tool, oriented to enable learning of low level ANN mechanisms.

Among the remaining ones, we decided to use the package `Keras` (Allaire and Chollet, 2021, about 25000 downloads), because this library is well documented, widely adopted in the AI field, focused on Deep Learning and offers a good abstraction on low-level ANN details (more than the `Torch` environment (Collobert et al., 2011; Falbel and Luraschi, 2021, about 1500 downloads) and `TensorFlow` (Abadi et al., 2015, about 25000 downloads). In effect, `Keras` package allows the users to focus on relevant data aspects,

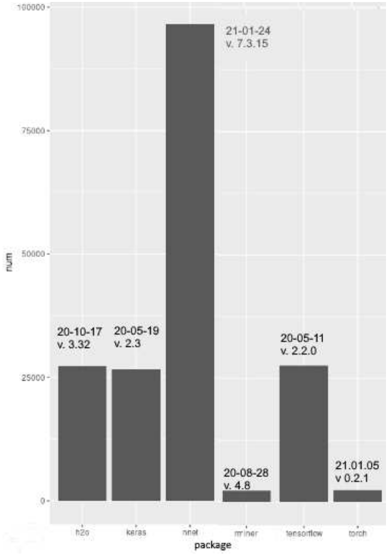


Figure 5.3: ANN/Deep Learning packages available on CRAN; on the vertical axis, the number of downloads in the period 12/02/21-13/01/21 is reported. Moreover, for each package it is reported the date and the number of the last release.

as the preparation of the dataset and the definition of the two hyperparameters defining a net (the number of layers and the number of units for each layer), without spending time on low level details.

5.3.2 Building Deep Learning models

All the models described in this work share the same Deep Learning sequential structure:

- one first input layer, with a number of input units /typically one) corresponding to the number of features to be considered in building the model
- one final output layer, with 1 output unit corresponding to the two possible results of a NBA match (basketball outcome prediction is a typical classification problem)
- a stack of several intermediate hidden sequential layers, connecting the input and output layers. Each hidden layer contains several elaboration units, to work on data received from the prior layer before sending them to the following layer.
  Data transformation on each layer is done by an $activationfunction$: a function, typically non linear, used to transform the weighted sum of layer inputs into outputs; in our models all layers use classic Rectified Linear Activation[8] *relu* (Goodfellow et al., 2016), apart from the output layer having a *sigmoid* activation function (the most suitable for a two-values classification problem).

This traversal process is repeated several times in both directions, with an optimizer updating weight values via a backpropagation algorithm, driven in its action by a loss function to be minimized. In our ANN:

- Adam (Kingma, 2014) is the optimizer.
- Binary_crossentropy is the loss function to be minimized.
- Accuracy is the metric to be used to verify the behavior of the net.

As all other machine learning mechanisms, ANN models can be affected by overfitting (i.e. the model is excessively tailored to the training data, and not able to generalize when applied to a different dataset); to verify and

[8] Rectified Linear Activation is a non-linear function defined as $relu(x) = max(0, x)$, i.e. returning 0 if its argument is negative, the argument itself otherwise.

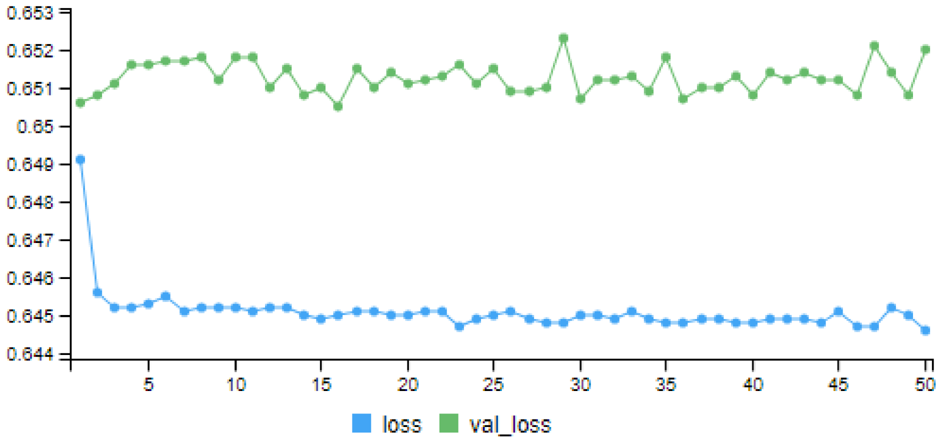


Figure 5.4: Deep Learning model on NBA dataset: example of overfitting (number of epochs on x-axis and measure of loss function on y-axis)

tune the behavior of the net before its application to a test dataset, in the `Keras` fit function it is possible to reserve a percentage of the training data (excluded from training, in our case 20%) to validation purposes. Figure 5.4, with the number of epochs (an epoch consists of one full training cycle on the training set) on x axis and a measure of the loss function on y axis, shows an example of overfitting problem, detectable comparing the distance of the loss functions for training (in blue) and for validation (in green) data: the loss function on the training data is decreasing with the increasing of the number of epochs as expected, while the loss function on validation data does not.

To reduce overfitting, dropout layers (i.e. hidden layers randomly setting some weight input units to 0, trying to explore not usual paths) and regularization parameters (i.e. penalizations introduced on the weights' matrix to let the network better generalize) are available (see Figure 5.5).

Prediction quality for our classification problem on NBA dataset is almost the same (see Figure 5.6 and Figure 5.7) using simple nets (with 1 input layer, 1 structural hidden layer and 2 drop layers, 1 output layer; AUC is 0.717, accuracy 0.6721 with a threshold of 0.489) and more complex networks (with ten hidden structural layers and much more units for each layer; AUC is 0.716, accuracy 0.6716 with a threshold of 0.432), and consequently the net with the simplest structure has been chosen.

Listing 5.1 reports the R code of the function invoking `Keras` to build a simple Deep Learning model, on the basis of the training data:

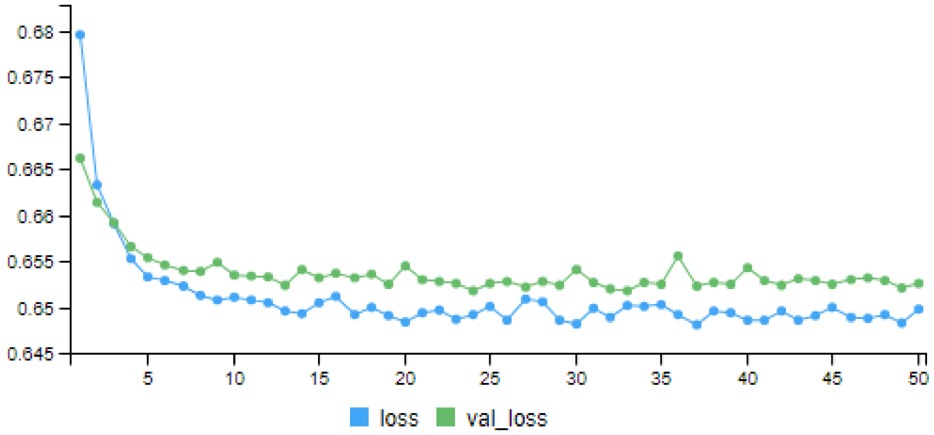


Figure 5.5: Deep Learning model on NBA dataset: usage of dropout layers and regularization parameters help in reducing overfitting

```
f.keras=function(X,Y, epochs_number,validation_perc){
    nf=dim(X)[2]                            # num of features
    k.fit = keras_model_sequential() %>%    # sequential model
    # input level: number of input equal to number of features
    layer_dense(units = 32, input_shape=nf, activation = "relu"
    ,
        kernel_regularizer = regularizer_l2(l = 0.001)) %>%
    layer_dropout(0.4) %>%  # to reduce overfitting
   # hiddel level
    layer_dense(units = 32, activation = "relu",
        kernel_regularizer = regularizer_l2(l = 0.001)) %>%
    layer_dropout(0.4) %>%  # to reduce overfitting
   # output level: 1 output and sigmoid activation
    layer_dense(1, activation = "sigmoid"
    k.fit %>%  compile( loss = ’binary_crossentropy’,
    optimizer =’’Adam’’
  metrics = "accuracy"  #  to verify net behavior
    # invoking fitting
    k.fit %>% fit( X,Y, epochs = epochs_number,
        validation_split = validation_perc,verbose=0)
    return(k.fit)}
```

Listing 5.1: Building a simple Deep Learning model via keras

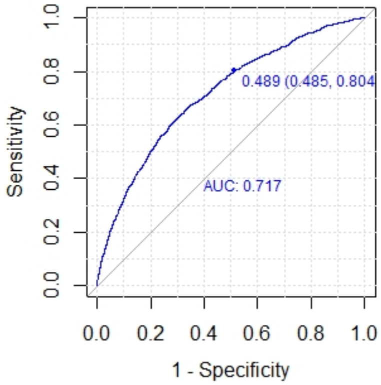


Figure 5.6: ROC curve for simple (just 1 hidden layers) Deep Learning model on NBA dataset

5.4 Results

The results reported in this section have been obtained using a v-fold cross-validation with v=4 (unless otherwise specified): for each validation, 75% of observations are randomly selected for training, and 25% for testing[9].

5.4.1 Using Elo based features

5.4.1.1 Historical Elo

Fit quality for Historical Elo based models depends on the three parameters used in calculating the feature: the percentage of regression to mean (see Subsection 5.2.2.1) and the two values of home advantage (see equations 5.10 and 5.11) and K (see equations 5.3 and 5.4) used in Elo rating calculation. To identify which parameters' values produce the best quality, several models have been fitted cycling on possible values of the parameters above, both considering or not the court issue.
Listing 5.3 shows the structure of the code implementing that cycling on the three parameters.

[9] A different approach based on time (using the first 14 regular seasons for training and the last 2 regular seasons for testing as in chapter 4), leads to similar results in terms of prediction quality.

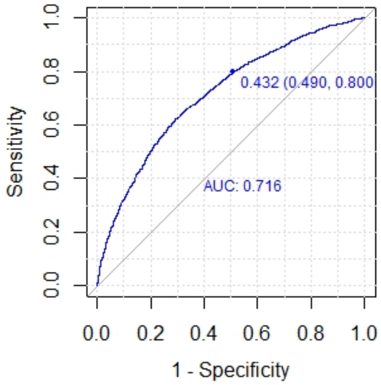


Figure 5.7: ROC curve for more complex (ten hidden layers with more units) Deep Learning model

```
# to find the best accuracy for models based on historical Elo
ans.elo.h.noha=NULL
for (prm in seq(0,100,by=10)){  # % regression to mean
  for (ha in seq(0,200,by=20)){  # home advantage value
    for (k in seq(10,70,by=10)){  # K
      nba.elo.h=f.ds.calc_elo(nba,prm,ha,k)    # preparing features
      nba.elo.h.noNA=na.omit( nba.elo.h)    # removing NA rows
      # building the model invoking keras
      ret=f.k.k_fold_acc( nba.elo.h.noNA,elo.h.,result, print=TRUE)
      acc=round(ret[1],4)            # accuracy
      auc=round(ret[2],4)            # area under curve
      # for each cycle, save parameters' values, accuracy and auc
      ans.elo.h.noha=rbind(ans.elo.h.noha,c(prm,ha,k,acc, auc))
    }
  }
}
```

Listing 5.2: Finding best accuracy for historical Elo

Execution results are reported in Table 5.5. The quality of predictions built using historical Elo without considering the court issue is the best one, with an AUC equal to 0.7117 and an accuracy equal to 0,6721 (using a

Table 5.5: Best quality of predictions for models based on historical Elo: for both approaches (considering or not the court issue), the best AUC measure, the corresponding threshold and the accuracy measure calculated using that threshold are reported, together with parameters' values used in Elo calculation

court issue	AUC	threshold	accuracy	regression to mean P%	home advantage	K
not considered	0.7117	0.5047	0.6721	20	40	30
considered	0.7001	0.5058	0.6650	60	40	30

threshold equal to 0.5047). These values have been obtained using a regression to mean percentage $P\%$ equal to 20, a home advantage parameter equal to 40, and K equal to 30, and are better than quality measures obtained for the best model considering the court issue (AUC equal to 0.7001, accuracy equal to 0.6650 using a threshold equal to 0.5058), built with a regression to mean percentage equal to 60, a home advantage equal to 40 and a K equal to 30.

5.4.1.2 Dynamic Elo

Predictions quality for models based on dynamic Elo depends on the depth used in averaging, as expressed in equation 5.6: a loop on the possible depths must be done, as reported in Listing 5.3. Results are reported in Table 5.6: the quality of predictions of the model built using dynamic Elo without considering the court issue is the best one, considering a depth equal to two; its AUC is equal to 0.7117 and its accuracy equal to 0.6736 (threshold equal to 0.5049), the best among the models we built in this work. Also predictions' quality for the best model built using dynamic Elo (depth equal to three) considering the court issue is good, with an AUC equal to 0.7103 and an accuracy equal to 0.6705 (threshold equal to 0.5148).

In general, models built on dynamic Elo have a quality slightly better than quality of models built using historical Elo.

Table 5.6: Quality of predictions for models based on dynamic Elo: for both approaches (considering or not the court issue), the best AUC measure, the corresponding threshold and the accuracy measure calculated using that threshold are reported, together with the depth used for averaging values.

court issue	AUC	threshold	best accuracy	depth
not considered	0.7117	0.5047	0.6736	2
considered	0.7103	0.5148	0.6705	3

```
## to find the best accuracy for models based on Elo rolling mean: cycling on depth
ans.elo.rm=NULL
for (depth in seq(2,90,by=1)){
  # calculating  Elo rolling mean features
  nba.elo_last=f.ds.calc_elo_last(nba,depth)
  feat=c(paste0("ea.elo.ha.delta.",depth),"result")
  # selecting only needed feature
  aux.elo.rm.ha=nba.elo_last_ha.bis[,feat]
  # removing NA rows
  aux.elo.rm.ha=na.omit(aux.elo.rm.ha)
  ret=f.k.k_fold_acc(aux.elo.rm.ha,feat,print=FALSE)
 # for each cycle, binding depth value, accuracy and auc
  ans.elo.rm=rbind(ans.elo.rm,c(dim(aux.elo.rm)[1],depth,round(ret[1],4)))
}

```

Listing 5.3: Finding best accuracy for dynamic Elo

5.4.1.3 Best accuracy model for Elo

The best Elo based model uses the dynamic Elo feature without considering the court issue, with a depth of 2; its AUC (calculated considering a single execution) is plotted in Figure 5.8; it is equal to 0.7117, the highest among our models. Predictions for that model considering single seasons have the accuracies reported in Table 5.7: season 2014-2015 shows best accuracy (0.7053), instead worst accuracy (0.6333) is for season 2019/20 (only partially played) and season 2008-2009 (0.6420).

Table 5.7: Elo best model prediction accuracy per single season

season	accuracy
2004-2005	0.6857
2005-2006	0.6772
2006-2007	0.6885
2007-2008	0.6753
2008-2009	0.6420
2009-2010	0.7028
2010-2011	0.7003
2011-2012	0.6545
2012-2013	0.6677
2013-2014	0.6829
2014-2015	0.7053
2015-2016	0.6768
2016-2017	0.6459
2017-2018	0.6633
2018-2019	0.6964
2019-2020	0.6333

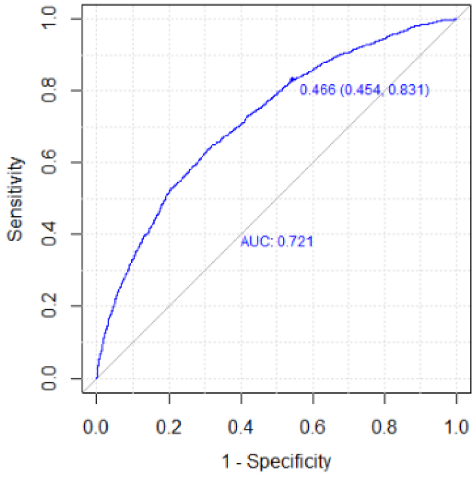


Figure 5.8: AUC for dynamic Elo (single execution). The figure reports the AUC value, together with the optimal threshold 0.466.

Table 5.8: Best quality of predictions for models based on historical $diff$: for both approaches (considering or not the court issue), the best AUC measure, the corresponding threshold and the accuracy measure calculated using that threshold are reported, together with the regression to mean percentage used for calculate the feature used in the model

court issue	AUC	threshold	accuracy	regression to mean P%
not considered	0.6925	0.5236	0.6626	90
considered	0.6775	0.4788	0.6572	78

5.4.2 Using $diff$ based features

5.4.2.1 Historical $diff$

In historical $diff$ approach, models' fit quality depends on regression to mean percentage value, as specified in equation 5.7. As a consequence, in order to identify the model with best predictions quality, all possible values for this parameter have been tried and results are reported in Table 5.8, where the quality of predictions of the model built using $diff$ without considering the court issue is the best one, with an AUC equal to 0.6925 and an accuracy equal to 0.6626 (using a threshold equal to 0.5236).

Figure 5.9 shows accuracy as a function of regression to mean percentage, for both $diff$ historical approaches. It can be important to underline how accuracy is relatively low (0.6139) when regression to mean percentage is equal to 0, being just slightly better than the naive strategy of assuming the home team always winning.

5.4.2.2 Dynamic $diff$

In $diff$ dynamic definition, models' quality in predictions depends on the depth employed for calculating rolling mean value, as expressed in equation 5.6. Results are reported in Table 5.9, where the quality of predictions of the model built using dynamic $diff$ without considering the court issue is the best one, with an AUC equal to 0.7020 and an accuracy equal to 0,663 (threshold equal to 0.5255).
Models built on dynamic $diff$ have a quality slightly better than quality of models built using historical $diff$.

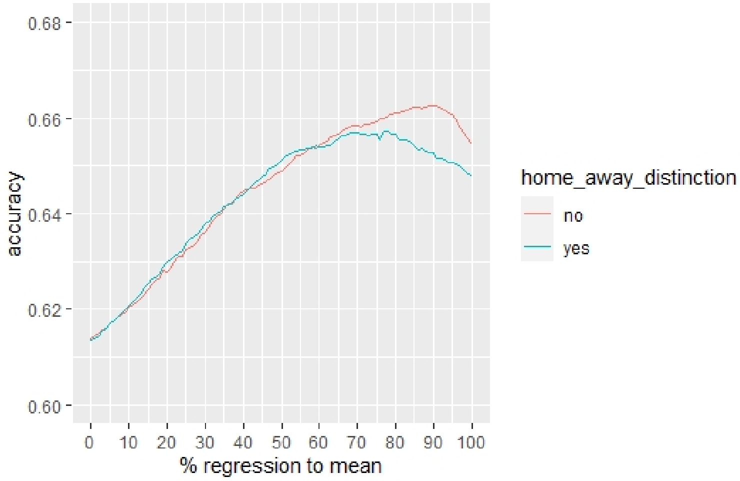


Figure 5.9: Historical $diff$: accuracy curve with respect to % of regression to mean

Table 5.9: Quality of predictions for models based on dynamic $diff$: for both approaches (considering or not the court issue), the best AUC measure, the corresponding threshold and the accuracy measure calculated using that threshold are reported, together with the depth used for averaging values

court issue	AUC	threshold	accuracy	depth
not considered	0.7020	0.5255	0.663	50
considered	0.6944	0.5057	0.6586	27

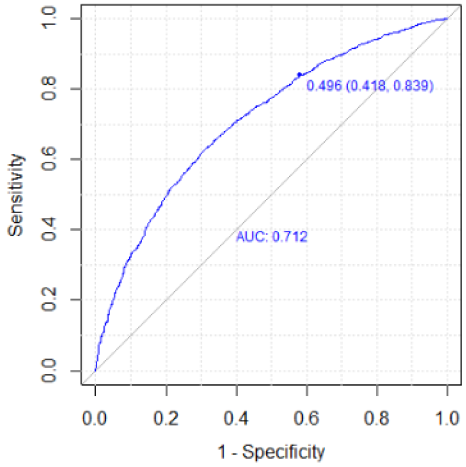


Figure 5.10: AUC for dynamic $diff$ (single execution). The figure reports the AUC value, together with the threshold (and its coordinates) to be considered in calculating accuracy

5.4.2.3 Best accuracy model for $diff$

The best model built on top of $diff$ feature is that using the dynamic $diff$ feature without considering the court issue, with a depth in averaging equal to 50; its AUC (calculated considering a single execution) is plotted in Figure 5.10. Predictions for that model considering single seasons have the accuracies reported in Table 5.10 (where first season 2004-2005 has been excluded because counting only 35 test observations due to NA omission in dataset preparation). Season 2010-2011 shows best accuracy (0.7034), instead worst accuracy (0.6222) is for season 2019/20 (only partially played) and season 2016-2017 (0.6230).

5.4.3 Using Four Factors based features

5.4.3.1 Historical Four Factors

As for other historical features, predictions quality for historical Four Factors based models depends on the percentage of regression to mean employed in dragging data from one season to the following one; Table 5.11 reports some results: the model built on historical Four Factors without considering the court issue is the best one, with an AUC equal to 0.6655 and an accuracy equal to 0.6400 (threshold equal to 0.5334).

Table 5.10: $Diff$ best model prediction accuracy per season

season	accuracy
2005-2006	0.6772
2006-2007	0.6689
2007-2008	0.6786
2008-2009	0.6636
2009-2010	0.6935
2010-2011	0.7034
2011-2012	0.6364
2012-2013	0.6553
2013-2014	0.6934
2014-2015	0.6623
2015-2016	0.6734
2016-2017	0.6230
2017-2018	0.6498
2018-2019	0.6568
2019-2020	0.6222

Table 5.11: Best quality of predictions for models based on historical Four Factors: for both approaches (considering or not the court issue), the best AUC measure, the corresponding threshold and the accuracy measure calculated using that threshold are reported, together with the regression to mean percentage used for calculate the feature used in the model

court issue	AUC	threshold	best accuracy	regression to mean P%
not considered	0.6655	0.5334	0.6400	78
considered	0.6527	0.4968	0.6347	74

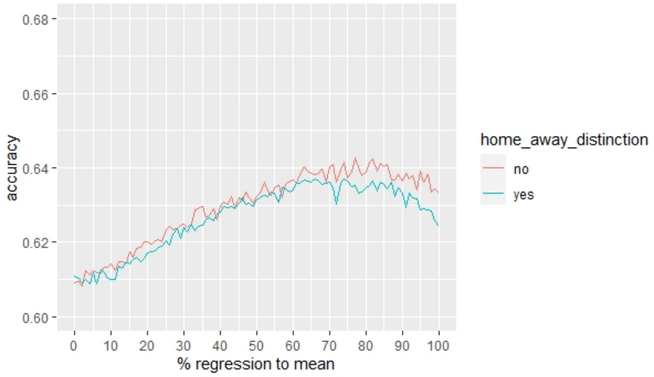


Figure 5.11: Historical Four Factors: accuracy VS regression to mean percentage; higher accuracy (0.6427) is found for the model without home/away data separation (considering a regression to mean percentage of 78%), against 0.6371 for model fitted considering home/away data separation.

Table 5.12: Best quality of predictions for models based on dynamic Four Factors: for both approaches (considering or not the court issue), the best AUC measure, the corresponding threshold and the accuracy measure calculated using that threshold are reported, together with the depth used for calculate the feature used in the model

court issue	AUC	threshold	best accuracy	depth %
not considered	0.6495	0.4934	0.6371	42
considered	0.6492	0.5091	0.6372	36

Also in this case (as for the corresponding situation for $diff$), prediction quality without applying regression to mean is actually lower (accuracy measure about 0.61), as reported in Figure 5.11.

5.4.3.2 Dynamic Four Factors

In Four Factors dynamic definition, models' quality in predictions depends on the depth employed for calculating rolling mean value, as expressed in equation 5.6. Results are reported in Table 5.12, where the quality measures of predictions for the two best models built considering or not the court issue are almost the same, with an AUC equal to 0.6495 considering the court issue and 0.6492 without considering it.

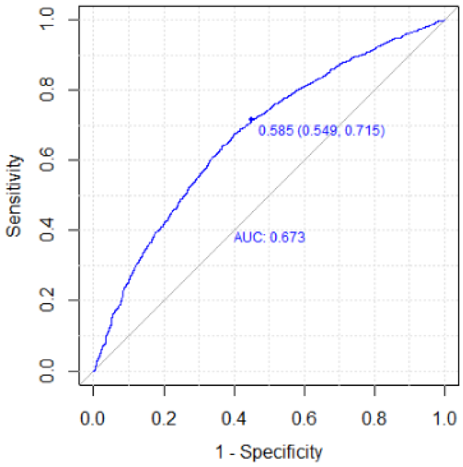


Figure 5.12: AUC for historical Four Factors (single execution). The figure reports the AUC value, together with the optimal threshold 0.585

5.4.3.3 Best accuracy model for Four Factors

Best predictions are generated using the model based on historical Four Factors without considering the court issue, with a regression to mean percentage P% of 78; its AUC (calculated considering a single execution) is plotted in Figure 5.12.

Predictions splitted on seasons (first season, 2004-2005, has been excluded, because counting only 35 observations due to NA management in dataset preparation) produces accuracies reported in Table 5.13, with regular season 2014-2015 showing best accuracy (0.6788) and worst accuracy (0.6175) reported for season 2005/06.

5.5 Conclusions and future works

In this chapter we showed how appropriately defined statistics can profitably be used as single features in fitting models for outcome predictions on a basketball dataset including 16 NBA regular seasons from 2004-2005 to 2019-2020.

The models quality is comparable (and in some cases better than) to results reported in the literature (with an accuracy about 0.67%-70%) for other models fitted using a huge number of statistics.

In this work, the best prediction quality for a model considering the whole

Table 5.13: Four Factors best model: accuracy per season (single execution)

season	accuracy
2005-2006	0.6175
2006-2007	0.6230
2007-2008	0.6494
2008-2009	0.6235
2009-2010	0.6502
2010-2011	0.6177
2011-2012	0.6455
2012-2013	0.6522
2013-2014	0.6760
2014-2015	0.6788
2015-2016	0.6566
2016-2017	0.6590
2017-2018	0.6532
2018-2019	0.6667
2019-2020	0.6370

period has been produced not considering the court issue and using a single dynamic Elo feature with an averaging depth equal to two (i.e. only Elo rating of prior two matches are considered in feature calculation). For this model, the AUC is equal to 0.7117 and the accuracy (using a threshold equal to 0.5047) is equal to 0.6736 (same AUC of the model built using historical Elo, but higher accuracy).

Comparing the accuracy of prediction on single seasons for the three models producing the best results, the dynamic Elo produces the best prediction in 9 seasons, the dynamic $diff$ in 5 and the Four Factors in 2. The best accuracy for a single season is equal to 0.7053 for the season 2014-2015.

Resultssuggest that, on the whole period, the court issue approach to features definition produces predictions comparable in the quality to models based on usual single feature, offering more interpretation details. Moreover, regression to mean can play a relevant role in prediction quality.

In general, quality of models built using $diff$ based features is close to quality of models built using Elo, and this is an expected result if we take into account how both these features express a direct measure of the strength of a team. Instead, the quality (remarkably the lowest among the three approaches) of models based on Four Factors seems a bit surprising. Maybe this fact suggests how the approaches based on box-score statistics are close to their limit in outcome prediction quality, and it is not easy to imagine

how to improve it.
In this perspective probably it can be more productive to deeply investigate approaches similar to those presented in this contribution, related to other information, considering:

1. a better management of seasons' change in dynamic feature definition. At the beginning of this work, regression to mean was thought just for historical features, supposing only a small number of prior matches have to be considered in dynamic features definition. Instead, results show that in some cases, as reported in Section 5.4, the best quality in dynamic models is often obtained considering a not so small depth (50 and 27, 36 and 42). In these cases, it is not difficult to cross two seasons, and regression to mean can play a role. It has been prototypically implemented also for the $diff$ feature in its dynamic form, and first outputs confirm how quality of fit is slightly improved by these strategies, and this aspect will be deeply investigated in future steps.

2. Analysis and integration of other kinds of information:

   (a) injuries, logistics, referees as proposed by several authors (see Sec. 5.2.2 about)

   (b) social networks (as proposed in Miller, 2015: today sources like Facebook (dated 2004) and Twitter (dated 2006) are old enough to offer information about several past years), breaking news, betting sites, market exchanges

   (c) players' characteristics and performances, both as single and with respect to other teammates (see Zuccolotto and Manisera, 2020 for a review).

Regarding Deep Learning: for this specific classification problem, we obtained good accuracy and limited overfitting maintaining small both the two hyperparameters (levels, units) of the net, heading towards the so called *shallow* learning: more complicated nets do not seem to offer great advantages in terms of quality of predictions, as reported in Sec. 5.3.2.
Few last words about `Keras`, the library we used to build Deep Learning models in R. Our activities have been really simplified by this package, offering a great abstraction on neural nets and enabling to focus just on relevant model aspects. At the moment, there is not a complete default explanation mechanisms associated to that library, but many researches are ongoing to offer explanation facilities (for example see Maksymiuk et al., 2020, Brandon

and Bradley, 2020 or Molnar et al., 2018), and it is easy to guess how this flaw will be early completely solved.

Bibliography

Abadi, M. et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Alamar, B. C. (2013). *Sports analytics: A guide for coaches, managers, and other decision makers*. Columbia University Press.

Albert, A. and Anderson, J. A. (1984). On the Existence of Maximum Likelihood Estimates in Logistic Regression Models. *Biometrika*, 71(1):1–10.

Albert, J., Glickman, M. E., Swartz, T. B., and Koning, R. H. (2017). *Handbook of Statistical Methods and Analyses in Sports*. CRC Press.

Allaire, J. and Chollet, F. (2021). *keras: R Interface to Keras*. R package version 2.4.0.

Allison, P. (2004). Convergence problems in logistic regression. In Altman M., Gill J., M. M., editor, *Numerical Issues in Statistical Computing for the Social Scientist*, pages 247–262. Wiley-Interscience, New York.

Allison, P. D. (2008). Convergence Failures in Logistic Regression. In *Proceedings of the* SAS *global forum*, pages 1–11. http://www2.sas.com/proceedings/forum2008/360-2008.pdf.

Angelini, G., Candila, V., and De Angelis, L. (2021). Weighted Elo rating for tennis match predictions. *European Journal of Operational Research*, 297(1):120–132.

Aria, M. and Cuccurullo, C. (2017). Bibliometrix: An R-tool for comprehensive science mapping analysis. *Journal of Informetrics*, 11(4):959–975.

Badenhausen, K. and Ozanian, M. (2021). NBA Team Values 2021. `https://www.forbes.com/sites/kurtbadenhausen/2021/02/10/nba-`

team-values-2021-knicks-keep-top-spot-at-5-billion-warriors-bump-lakers-for-second-place/?sh=1e4a4def645b, visited 19 Jul 2021.

Baker, R. and Kwartler, T. (2015). Using Open Source Logistic Regression Software to Classify Upcoming Play Type in the NFL. *Journal of Applied Sport Management*, 07(2).

Beal, R., Timothy, N., and Sarvapali, R. (2020). A Critical Comparison of Machine Learning Classifiers to Predict Match Outcomes in the NFL. *International Journal of Computer Science in Sport*, 19(2).

Beckler, M., Wang, H., and Papamichael, M. (2013). NBA oracle. Technical report. https://www.mbeckler.org/coursework/2008-2009/10701_report.pdf, visited 3 December 2020.

Bianchi, F., Facchinetti, T., and Zuccolotto, P. (2017). Role revolution: towards a new meaning of positions in basketball. *Electronic Journal of Applied Statistical Analysis*, 10(3):712–734.

Blei, D., Ng, A.Y, ., and Jordan, M. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Brandon, M. G. and Bradley, C. B. (2020). Variable Importance Plots—An Introduction to the vip Package. *The R Journal*, 12(1):343–366.

Breiman, L. (2001a). Random Forests. *Machine Learning*, 45(1):5–32.

Breiman, L. (2001b). Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199 – 231.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. (1984). *Classification and Regression Trees*. Chapman and Hall/CRC.

Bunker, R. P. and Thabtha, F. (2019). A machine learning framework for sport result prediction. *Applied Computing and Informatics*, 15(1):27–33.

Cao, C. (2012). Sports data mining technology used in basketball outcome prediction. Master dissertation, Technological University of Dublin. http://arrow.dit.ie/cgi/viewcontent.cgi?article=1040&context=scschcomdis, visited 3 Dec 2020.

Carpita, M., Ciavolino, E., and Pasca, P. (2019). Exploring and modelling team performances of the Kaggle European Soccer Database. *Statistical Modelling*, 19:1–29.

Carpita, M., Ciavolino, E., and Pasca, P. (2020). Players' Role-Based Performance Composite Indicators of Soccer Teams: A Statistical Perspective. *Social Indicators Research*, 156:1–16.

Carpita, M., Sandri, M., Simonetto, A., and Zuccolotto, P. (2015). Discovering the Drivers of Football Match Outcomes with Data Mining. *Quality Technology & Quantitative Management*, 12(4):537–553.

Cheng, G., Zhang, Z., Kyebambe, M. N., and Kimbugwe, N. (2016). Predicting the Outcome of NBA Playoffs Based on the Maximum Entropy Principle. *Entropy*, 18(12).

Chollet, F. and Allaire, J. (2018). *Deep Learning in R.* Manning.

Collobert, R., Kavukcuoglu, K., and Farabet, C. (2011). Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*.

Cordeiro, G. and McCullagh, P. (1991). Bias correction in generalized linear models. *J. R. Stat. Soc. Ser. B Methodol.*, 53(3):629–643.

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

Cortez, P. (2015). A tutorial on using the rminer R package for data mining tasks.

Cox, D. R. (1970). *Analysis of Binary Data.* Chapman and Hall, London.

David, J., Pasteur, R. D., Ahmad, M., and Janning, M. (2011). NFL Prediction using Committees of Artificial Neural Networks. *Journal of Quantitative Analysis in Sports*, 7(2):1–15.

Davoodi, E. and Khanteymoori, A. (2010). Horse racing prediction using artificial neural networks. *Recent advances in neural networks, fuzzy systems & evolutionary computing*, pages 155–160.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407.

Deutscher, C. (2015). No referee bias in the NBA: New evidence with leagues' assessment data. *Journal of Sports Analytics*, 1:91–96.

Eetvelde, H., De Michelis Mendonça, L., Ley, C., Seil, R., and Tischer, T. (2021). Machine learning methods in sport injury prediction and prevention: a systematic review. *Journal of Experimental Orthopaedics*, 8:27.

Eetvelde, H. and Ley, C. (2019). *Ranking Methods in Soccer*, pages 1–9.

Elo, A. E. (1978). *The Rating of Chess players, Past and Present.* Ishi Press International.

ESPN (2021). ESPN NBA statistics. `https://www.espn.com/nba/stats`, visited 2021-05-27.

Falbel, D. and Luraschi, J. (2021). *torch: Tensors and Neural Networks with 'GPU' Acceleration.* R package version 0.4.0.

Firth, D. (1993). Bias Reduction of Maximum Likelihood Estimates. *Biometrika*, 80(1):27–38.

Forbes (2021). The business of soccer - 2021 Ranking. `https://www.forbes.com/soccer-valuations/list/#tab:overall`, visited 19 Jul 2021.

Galton, F. (1889). *Natural Inheritance.* MacMillan.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning.* MIT Press.

Gough, C. (2021). National Basketball Association total league revenue from 2001/02 to 2019/20. `https://www.statista.com/statistics/193467/total-league-revenue-of-the-nba-since-2005/`, visited 15 Jul 2021.

Green, P. J. (1984). Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and Some Robust and Resistant Alternatives. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(2):149–170.

Groll, A., Manisera, M., Schauberger, G., and Zuccolotto, P. (2018). Guest Editorial Special Issue Statistical Modelling for Sports Analytics. *Statistical Modelling*, 18(5-6):385–387.

Gu, W., Saaty, T., and Whitaker, R. (2016). Expert System for Ice Hockey Game Prediction: Data Mining with Human Judgment. *International Journal of Information Technology & Decision Making*, 15(04):763–789.

H2O.ai (2021). *h2o: R Interface for H2O.* R package version 3.32.1.3.

Haghighat, M., Rastegari, H., and Nourafza, N. (2013). A Review of Data Mining Techniques for Result Prediction in Sports. *ACSIJ Advances in Computer Science: an International Journal*, 2.

Harville, D. A. and Smith, M. (1994). The home-court advantage: how large is it and does it vary from team to team. *The american statistician*, 48:22–29.

Heinzen, E. (2020). *Elo Ratings*. "https://cran.r-project.org/package=elo".

Ho, T. K. (1995). Random decision forests. In *Proceedings of the 3rd International Conference on Document, Analysis and Recognition*, pages 278–282. IEEE.

Hopkins, R. (2021). NBA injuries from 2010-2020. https://www.kaggle.com/ghopkins/nba-injuries-2010-2018, visited 2021-03-31.

Hothorn, T. and Zeileis, A. (2015). Partykit: A Modular Toolkit for Recursive Partytioning in R. *Journal of Machine Learning Research*, 16(118):3905–3909.

Hubáček, O., Sourek, G., and Železný, F. (2019). Exploiting sports-betting market using machine learning. *International Journal of Forecasting*, 35:783–796.

Hvattum, L. M. and Arntzen, H. (2010). Using Elo Ratings for Match Results Prediction in Association Football. *International Journal of Forecasting*, 26:460–470.

Jaynes, E. (1957). Information theory and statistical mechanics. *The physical review*, 106(4):620–630.

Jones, M. B. (2007). Home advantage in the NBA as a game-long process. *Quantitative analysis in sport*, 3(4).

Kahn, J. (2003). Neural network prediction of NFL football games. http://homepages.cae.wisc.edu/ ece539/project/f03/kahn.pdf.

Kenne Pagui, E. C., Salvan, A., and Sartori, N. (2017). Median bias reduction of maximum likelihood estimates. *Biometrika*, 104(4):923–938.

Khun, M. (2020). *package "caret"*. `https://cran.r-project.org/web/packages/caret/caret.pdf`.

Kingma, D.P.and Ba, j. (2014). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR).*

Koseler, K. and Stephan, M. (2017). Machine Learning Applications in Baseball: A Systematic Literature Review. *Applied Artificial Intelligence*, 31(9-10):745–763.

Kosmidis, I. and Firth, D. (2020). Jeffreys-prior penalty, finiteness and shrinkage in binomial-response generalized linear models. *Biometrika*, 108(1):71–82.

Kosmidis, I., Kenne Pagui, E. C., and Sartori, N. (2020). Mean and median bias reduction in generalized linear models. *Statistics and Computing*, 30(1):43–59.

Kubatko, J., Oliver, D., Pelton, K., and Rosenbaum, D. (2007). A starting point for analysing basketball statistics. *Journal of Quantitative Analysis in Sports*, 3(3):1–22.

Landwehr, N., Hall, M., and Frank, E. (2005). Logistic Model Trees. *Machine Learning*, 59:161–205.

Leitner, C., A. Zeileis, A., and Hornik, K. (2010). Forecasting Sports Tournaments by Ratings of (Prob)abilities: A Comparison for the EURO 2008. *International Journal of Forecasting*, 26:471–481.

Lewis, M. (2003). *Moneyball: the art of winning an unfair game.* W.W. Norton & Co Inc., New York.

Loeffelholz, B., Bednar, E., and Bauer, K. (2009). Predicting NBA games using neural networks. *Journal of Quantitative Analysis in Sports*, 5(1):1–17.

Mackenzie, R. and Cushion, C. (2013). Performance analysis in football: A critical review and implications for future research. *Journal of Sports Sciences*, 31(6):639–676.

Maksymiuk, S., Gosiewska, A., Biecek, P., and Staniak, M. (2020). *shapper: Wrapper of Python Library 'shap'*. `"https://cran.r-project.org/package=shapper"`.

Manner, H. (2016). Modeling and forecasting the outcomes of NBA basketball games. *Journal of Quantitative Analysis in Sports*, 12(1):31–41.

Mansournia, M., Geroldinger, A., Greenland, S., and Heinze, G. (2018). Separation in Logistic Regression: Causes, Consequences, and Control. *American Journal of Epidemiology*, 187(4):864–870.

Marusek, F. (2021). Pro Sports Transactions. `https://www.prosportstransactions.com/basketball/`, visited 2021-03-31.

Metulini, R., Manisera, M., and Zuccolotto, P. (2018). Modelling the dynamic pattern of surface area in basketball and its effects on team performance. *Journal of Quantitative Analysis in Sports*, 14(3):117–130.

Migliorati, M. (2020). Detecting drivers of basketball successful games: an exploratory study with machine learning algorithms. *Electronic Journal of Applied Statistical Analysis*, 13:454–473.

Migliorati, M., Manisera, M., and Zuccolotto, P. (2020). The impact of Four Factors on a basketball team success: An approach with model-based recursive partitioning. In *CFE-CMStatistics 2020 Book of Abstracts*, page 42. ISI.

Miljkovic, D., Gajic, L., Kovacevic, A., and Konjovic, Z. (2010). The use of data mining for basketball matches outcomes prediction. In *IEEE 8th international symposium on intelligent systems and informatics*, pages 309–312. IEEE.

Miller, T. W. (2015). *Sport analytics and data science: winning the game with methods and models.* FT press.

Min, B., Kim, J., Choe, C., Eom, H., and McKay, R. I. (2008). A compound framework for sports results prediction: a football case study. *Knowledge-Based Systems*, 12:551–562.

Molnar, C. (2021). *Interpretable Machine Learning; A Guide for Making Black Box Models Explainable.* `https://christophm.github.io/interpretable-ml-book/global.html`, visited 15/03/2021.

Molnar, C., Casalicchio, G., and Bischl, B. (2018). iml: An R package for Interpretable Machine Learning. *Journal of Open Source Software*, 3(26):786.

NBA (2020a). NBA statistics. https://stats.nba.com/teams/traditional.

NBA (2020b). NBA statistics Glossary. https://stats.nba.com/help/glossary/.

Oliver, D. (2004). *Basketball on Paper: Rules and Tools for Performance Analysis*. Potomac Books inc.

Page, G. L., Fellingham, G., and Reese, S. (2007). Using box-scores to determine a position's contribution to winning basketball games. *Journal of Quantitative Analysis in sport*, 3.

Passos, P., Araujo, D., and Volossovitch, A. (2016). *Performance analysis in team sports*. Taylor and Francis.

Price, J., Remer, M., and Stone, D. (2009). Subperfect Game: Profitable Biases of NBA Referees. *Journal of Economics & Management Strategy*, 21:271–300.

Purucker, M. (1996). Neural network quarterbacking. *IEEE Potentials*, 15:9–15.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco.

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning*, 1(1):81–106.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. `https://www.R-project.org/`.

Robin, X., Turck, N., Hainard, A., Tiberti, N., Lisacek, F., Sanchez, J., and Muller, M. (2011). pROC: an open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics*, 12(1).

RStudio Team (2021). *RStudio: Integrated Development Environment for R*. RStudio, PBC., Boston, MA. `http://www.rstudio.com/`.

Ryall, R. and Bedford, A. (2010). An optimized ratings-based model for forecasting Australian Rules football. *International Journal of Forecasting*, 26:511–517.

Sandri, M. (2020). *Zuccolotto P. and Manisera M.; Basketball data science – with Applications in R*, chapter 6, pages 185–196. Chapman and Hall/CRC.

Sandri, M., Zuccolotto, P., and Manisera, M. (2018). *BasketballAnalyzeR: An R package for the analysis of basketball data.* https://CRAN.R-project.org/package=BasketballAnalyzeR.

Sandri, M., Zuccolotto, P., and Manisera, M. (2020). Markov switching modelling of shooting performance variability and teammate interactions in basketball. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 69:1337–1356.

Sarmento, H., Marcelino, R., M.T., A., CampaniÇo, J., Matos, N., and Carlos LeitÃo, J. (2014). Match analysis in football: a systematic review. *Journal of Sports Sciences*, 32(20):1831–1843.

Silver, N. (2014). Introducing NFL Elo ratings. https://fivethirtyeight.com/features/introducing-nfl-elo-ratings, visited 2021-03-11.

Silver, N. (2015). How We Calculate NBA Elo Ratings. https://fivethirtyeight.com/features/how-we-calculate-nba-elo-ratings/, visited 2021-03-11.

Silver, N. (2020a). 2019-20 NBA Predictions. https://projects.fivethirtyeight.com/2020-nba-predictions, visited 2021-05-27.

Silver, N. (2020b). How our NBA predictions work. https://fivethirtyeight.com/methodology/how-our-nba-predictions-work/, visited 2021-05-27.

Sports reference (2021). NBA Transactions. "https://www.basketball-reference.com/leagues/NBA_2019_transactions.html", visited 2021-03-31.

Stephenson, A. and Sonas, J. (2020). *Player Ratings*. "https://CRAN.R-project.org/package=PlayerRatings".

Swartz, T. B. (2017). Hockey analytics. pages 1–10. Wiley StatsRef: Statistics Reference Online.

Tax, N. and Joustra, Y. (2015). Predicting the Dutch football competition using public data: A machine learning approach. *IEEE Transactions on Knowledge and Data Engineering*, 10(10):1–13.

teamranking (2021). teamranking predictions. https://www.teamrankings.com/nba/betting-models/detailed-splits/, visited 2021-05-27.

Thabtah, F., Zhang, L., and Abdelhamid, N. (2019). NBA Game Result Prediction Using Feature Analysis and Machine Learning. *Annals of Data Science*, 6(1):103–116.

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*. Springer, New York, fourth edition.

Wickham, H. (2014). Tidy data. *The Journal of Statistical Software*, 59.

Wickham, H., François, R., Henry, L., and Müller, K. (2020). *dplyr: A Grammar of Data Manipulation*. `https://CRAN.R-project.org/package=dplyr`.

World Football Elo Ratings (2021). World Football Elo Ratings. `https://www.eloratings.net/icehockey/`, visited 2021-05-27.

Wunderlich, F. and Memmert, D. (2020). Forecasting the outcomes of sports events: A review. *European Journal of Sport Science*.

Youden, M. J. (1950). Index for rating diagnostic tests. *Cancer*, (3):32–35.

Zeileis, A., Hothorn, T., and Hornik, K. (2008). Model-Based Recursive Partitioning. *Journal of Computational and Graphical Statistics*, 17(2):492–514.

Zhang, P. (2000). Neural Networks for Classification: A Survey. *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)*, 30(4):451–462.

Zuccolotto, P. and Manisera, M. (2020). *Basketball Data Science – with Applications in R*. Chapman and Hall/CRC.

Zuccolotto, P., Manisera, M., and Kenett, R. S. (2017a). Guest Editorial Special Issue Statistics in Sports. *Electronic Journal of Applied Statistical Analysis*, 10(3):1–2.

Zuccolotto, P., Manisera, M., and Sandri, M. (2017b). Big data analytics for modeling scoring probability in basketball: the effect of shooting under high-pressure conditions. *International Journal of Sports Science & Coaching*, 13(4):569–589.

Zuccolotto, P., Sandri, M., and Manisera, M. (2019). Spatial Performance Indicators and Graphs in Basketball. *Social Indicators Research*, 156:1–14.

Appendix A

Tools for bibliographic search

As a classic in approaching research questions, the first activity is related to analyze scientific literature, in order to identify papers relevant to the research domain and to collect information about publications:

- Authors and groups
- Nations
- Years
- Milestones and references
- Approaches and data science models to be tested.

This activity has evolved in a new research field, focused on bibliographic research coupled to Natural Language Processing (NLP). We adopted this approach in the field of machine learning applied to basketball, and first results are summarized in this Appendix, which is structured as follows: Section A.1 introduces the problem of identifying papers relevant to a specific domain using ad hoc bibliographic tools. Section A.2 shows some results obtained in applying bibliographic and NLP tools to COVID-19 scientific publications. Section A.3 summarizes first results obtained in applying a new index (named h^*), for assigning a credibility rate to not peer reviewed articles (e.g. those published on arXiv repository) on the basis of the authors' information, automatically retrieved accessing Google Scholar.

A.1 Retrieving and analysing papers focused on basketball and machine learning

The first step in identifying meaningful papers focused on machine learning applied to basketball has been the selection of a corpus of publications to be analyzed using `Bibliometrix` (Aria and Cuccurullo, 2017), an R open-source tool particularly suitable for quantitative research in scientometrics and bibliometrics.

A.1.1 Selecting a source

The two most influential scientific abstract and citation indexing services, Clarivate's Web Of Science (WoS) and Elsevier's Scopus, have been compared to identify the source to be used in the following analysis. The same simple query

machine learning AND basketball

has been executed on both indexing services, producing 29 results in WoS and 39 results in Scopus (as July 2019), classified as reported in Table A.1. In terms of number of meaningful results, the two sources seems to be similar, also if some more elements have been retrieved by Scopus. But looking to the fields available for the retrieved documents, WoS has been verified offering some more useful information (fields about research areas and categories classification), so we decided to adopt this source for our bibliographic search.

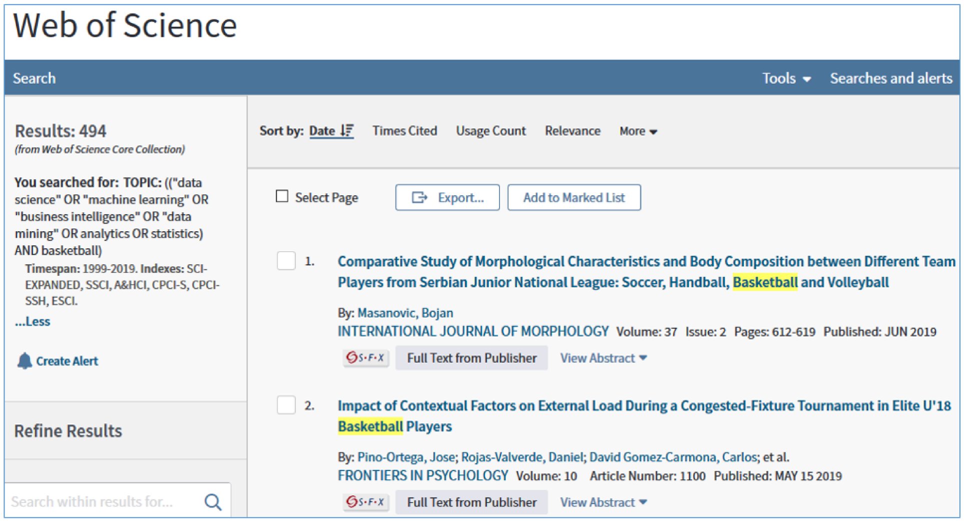
Table A.1: Results of the query *machine learning and basketball* in both Web of Science and Scopus

no.	description
25	elements in common between Scopus and Wos
4	WoS exclusive items: all articles, relevant and with many references
14	Scopus exclusive items: 5 out of scope, 3 related but not relevant, 6 (1 review paper and 5 conference papers) relevant

A.1.2 A more sophisticated query

As a second step, the following more complex query:
("data science" OR "machine learning" OR "business intelligence" OR

"data mining" OR analytics OR statistics) AND basketball
has been executed (July 2019) in WoS, taking into account a time period of 20 years and limiting the search to title, abstract, keywords, as reported in Figure A.1.



Results: 494
(from Web of Science Core Collection)

You searched for: TOPIC: (("data science" OR "machine learning" OR "business intelligence" OR "data mining" OR analytics OR statistics) AND basketball)
Timespan: 1999-2019. **Indexes:** SCI-EXPANDED, SSCI, A&HCI, CPCI-S, CPCI-SSH, ESCI.

□ 1. **Comparative Study of Morphological Characteristics and Body Composition between Different Team Players from Serbian Junior National League: Soccer, Handball, Basketball and Volleyball**
By: Masanovic, Bojan
INTERNATIONAL JOURNAL OF MORPHOLOGY Volume: 37 Issue: 2 Pages: 612-619 Published: JUN 2019

□ 2. **Impact of Contextual Factors on External Load During a Congested-Fixture Tournament in Elite U'18 Basketball Players**
By: Pino-Ortega, Jose; Rojas-Valverde, Daniel; David Gomez-Carmona, Carlos; et al.
FRONTIERS IN PSYCHOLOGY Volume: 10 Article Number: 1100 Published: MAY 15 2019

Figure A.1: Query executed in Wos, producing 494 results

obtaining 494 results.

In the third step, the 494 papers have been manually analyzed to select just pertinent papers, studying title, source, categories and reading the abstract. The number of papers decreased to 117, our dataset[1], to be analyzed using `Bibliometrix`.

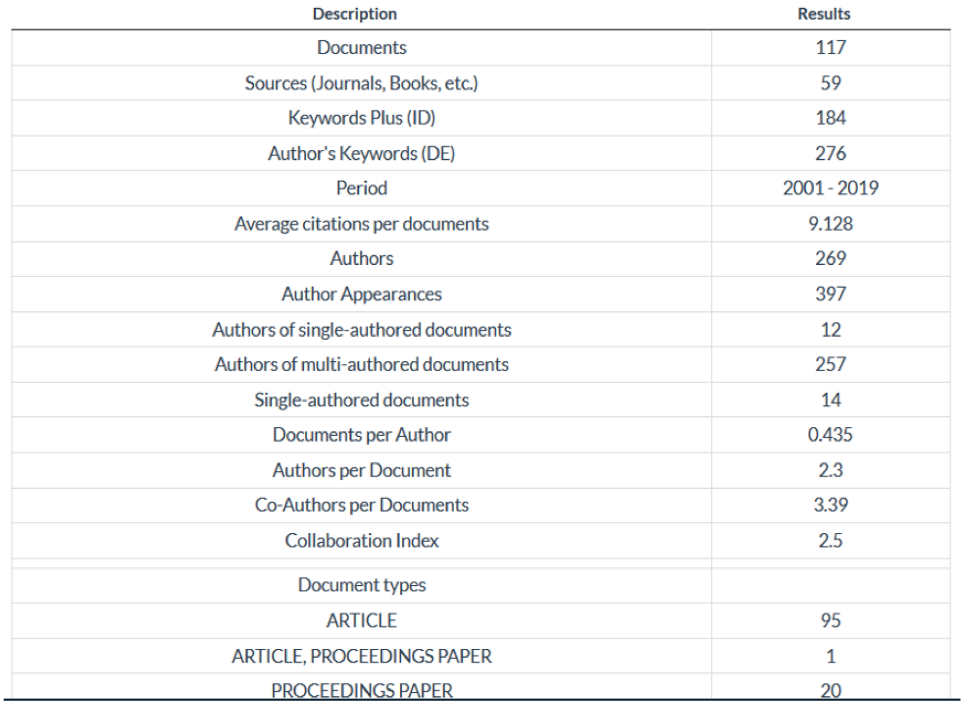
In some cases, a single selection considering only the last 20 years was not sufficient to highlight the evolution over time of the aspect under analysis, and a comparison with a second selection considering only the last 5 years has been done.

A.1.3 Applying **`Bibliometrix`**

This section is devoted to illustrate the results obtained invoking `Bibliometrix` (used via its web interface named `Biblioshiny`) on the dataset of 117 papers.

[1] It would be interesting to compare this corpus manually selected to clusters produced applying NLP methodologies (as Topic Models) to original set of 494 papers, to verify if it is possible to find relations among results.

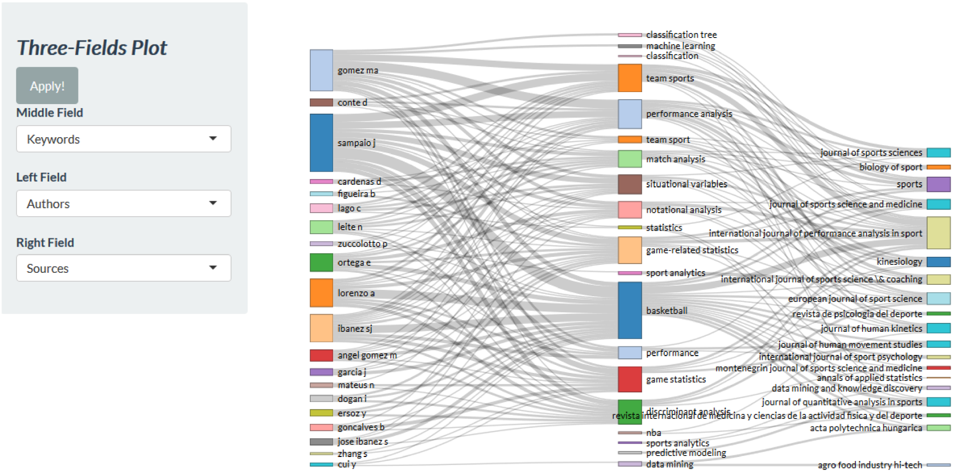
Let's start with a summary of the dataset, as reported in Figure A.2, including several information and statistics about the selected papers (among others: the keywords, the period, the number of citations, the kind of document)



Description	Results
Documents	117
Sources (Journals, Books, etc.)	59
Keywords Plus (ID)	184
Author's Keywords (DE)	276
Period	2001 - 2019
Average citations per documents	9.128
Authors	269
Author Appearances	397
Authors of single-authored documents	12
Authors of multi-authored documents	257
Single-authored documents	14
Documents per Author	0.435
Authors per Document	2.3
Co-Authors per Documents	3.39
Collaboration Index	2.5
Document types	
ARTICLE	95
ARTICLE, PROCEEDINGS PAPER	1
PROCEEDINGS PAPER	20

Figure A.2: Bibliographic dataset information

The Three-fields plot of Figure A.3 enables to create correspondence among three information classes; in our case, class *authors* is related to class *keywords*, and class *keywords* to class *journals*. In this way, it is possible navigate the plot, to analyze who published what and where. Instead, Figure A.4 enables a quantitative analysis of the distribution of papers in the 10 most important journals, comparing 20- and 5-year old views. In that two periods, first three reviews remain the same, but it can be interesting to verify other changes in these rankings: for instance, *Sports* is 10th in the 20-year ranking, and 4th in 5-year ranking, improving its weight; instead, *European Journal of Sport Science* is 4th in 20-year ranking, and it does not appear in the 5-year ranking. Moreover, the journal titled *Agro food industry hitech* is present in both rankings; it is a bi-monthly publication, indexed by both WoS and Scopus, normally not related to data science and sports (as per title); it is



Three-Fields Plot

Apply!

Middle Field: Keywords

Left Field: Authors

Right Field: Sources

gomez ma
conte d
sampaio j
cardenas d
figueira b
lago c
leite n
zuccolotto p
ortega e
lorenzo a
ibanez sj
angel gomez m
garcia j
mateus n
dogan i
ersoz y
goncalves b
jose ibanez s
zhang s
cui y
classification tree
machine learning
classification
team sports
performance analysis
team sport
match analysis
situational variables
notational analysis
statistics
game-related statistics
sport analytics
basketball
performance
game statistics
discriminant analysis
nba
sports analytics
predictive modeling
data mining
journal of sports sciences
biology of sport
sports
journal of sports science and medicine
international journal of performance analysis in sport
kinesiology
international journal of sports science \& coaching
european journal of sport science
revista de psicologia del deporte
journal of human kinetics
journal of human movement studies
international journal of sport psychology
montenegrin journal of sports science and medicine
annals of applied statistics
data mining and knowledge discovery
journal of quantitative analysis in sports
revista internacional de medicina y ciencias de la actividad fisica y del deporte
acta polytechnica hungarica
agro food industry hi-tech

Figure A.3: An application of three-fields plot: relations among authors, keywords and journals

present because our topic is the focus of two issues (1 and 3, vol 28, 2017), containing 25 articles (mainly from Chinese authors) about basketball and data science.

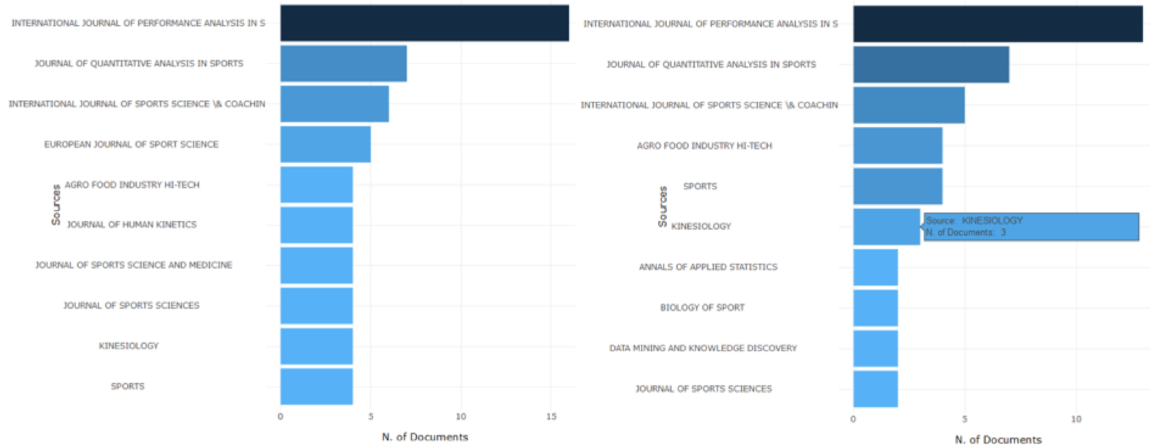


Figure A.4: Comparing main bibliographic sources (20 VS 5 years)

Figure A.5 enables a quantitative view about authors, comparing the first 9 more prolific authors in the considered domain, still taking into account the two 20- and 5-year old views. Also in this case the top of the rankings do not change, but there are several new entries in the last five years, highlighting the liveliness of the sector.

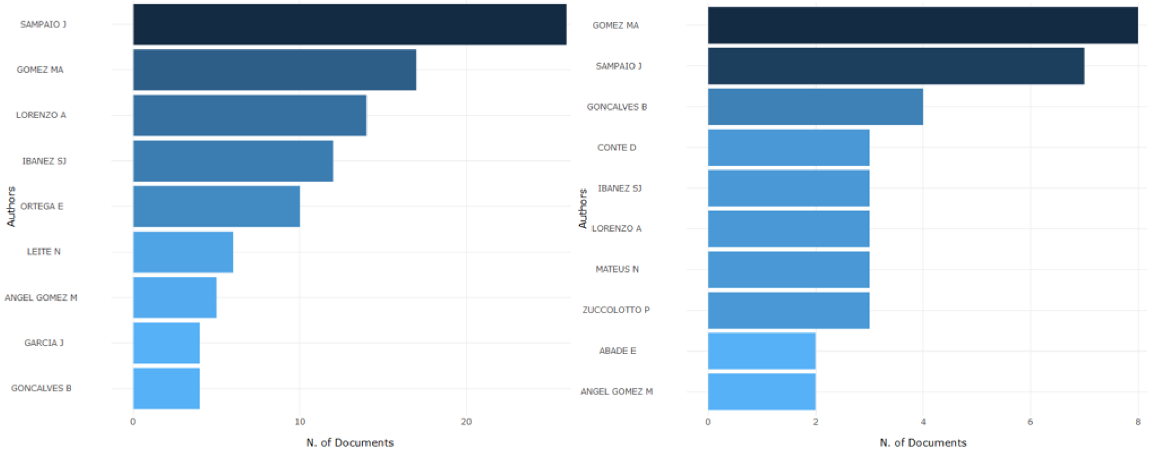


Figure A.5: Comparing authors (20 VS 5 years)

Authors affiliations (always 20 VS 5 years) are reported in Figure A.6, showing how Iberian Universities are central for the domain (but also our University of Brescia entered 5-year ranking).

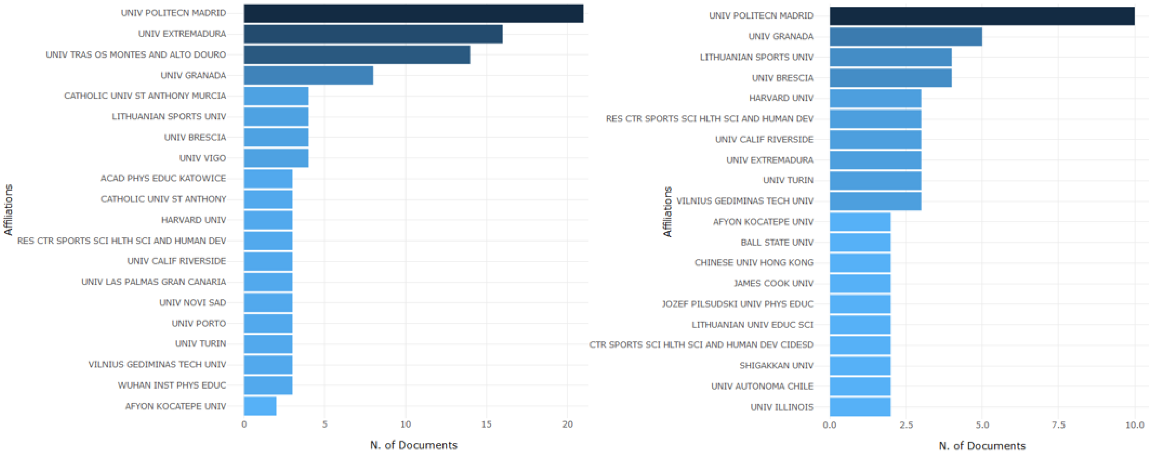


Figure A.6: Comparing affiliations (20 VS 5 years)

Figure A.7 shows how the conceptual structure in the domain evolved in time: the increased number of concepts and relations demonstrates the increased interest for the field.

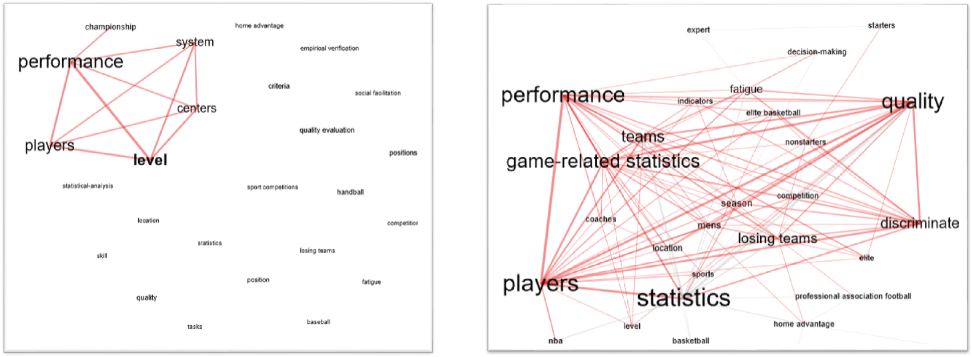
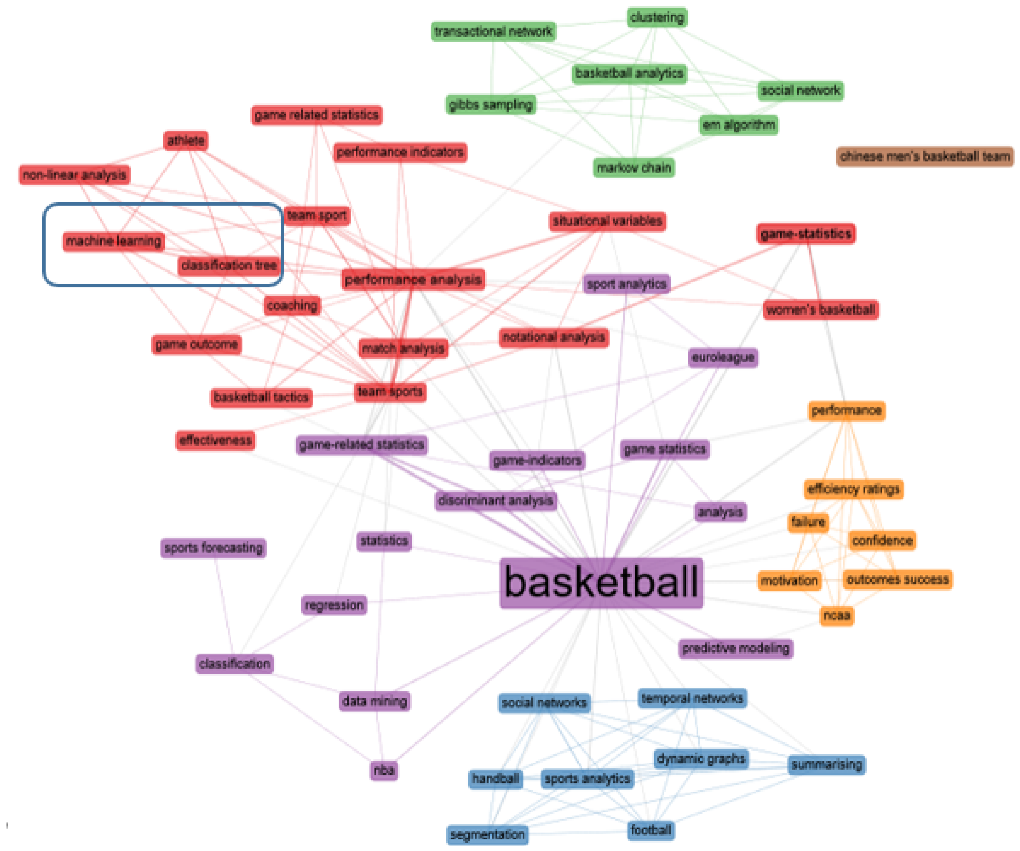


Figure A.7: Evolution of domain conceptual structure(20 VS 5 years)

Figure A.8 shows the thematic map for the domain under investigation, based on papers' co-word network analysis and clustering (considering the 50 most important authors' keywords, with a minimum of 20 words for each cluster). The red cluster in the top left region seems the more interesting for our purposes, being characterized by data science typical terms.

Figure A.9 show the graph of co-citations among authors, where a larger area means a higher number of citations. Both Four Factors' references (Kubatko 2007, Oliver 2004) are among the most important.

Figure A.10 shows the network collaboration among authors, and it is interesting to note how the more prolific authors as reported in Figure A.5 are greatly cooperating. Moreover, among others it is present in the chart the collaboration link between two professors of University of Brescia actively working in the domain.



- author's keywords
- Number of words: 50
- Min cluster frequency: 20

Figure A.8: Thematic map based on papers' keywords

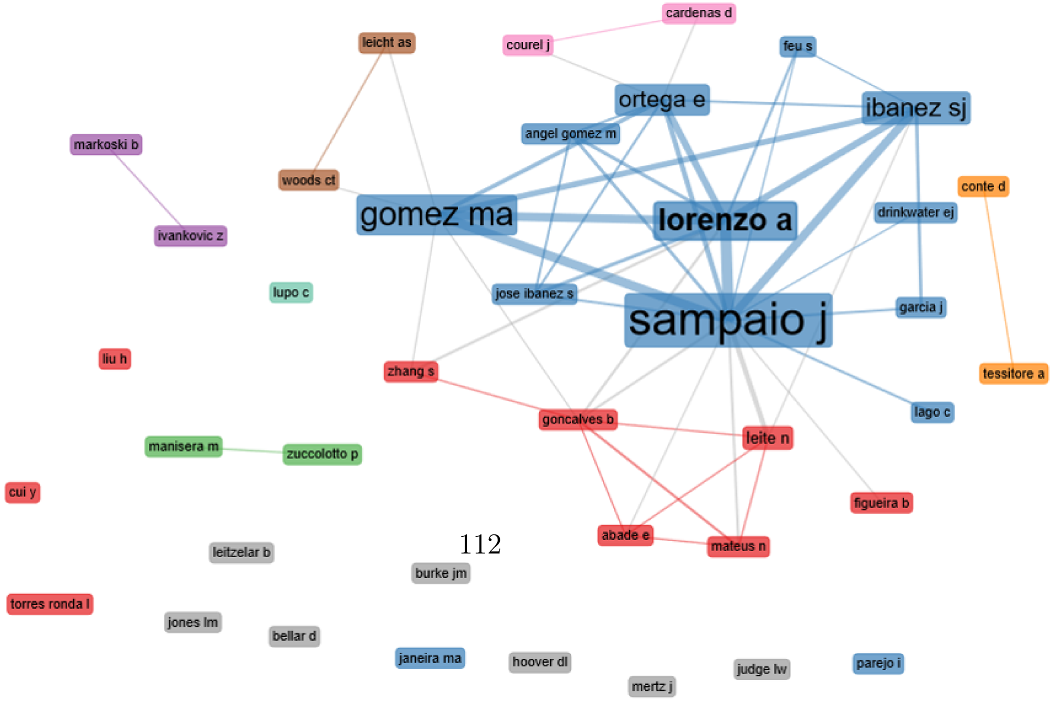


Figure A.10: Network of collaborations among authors

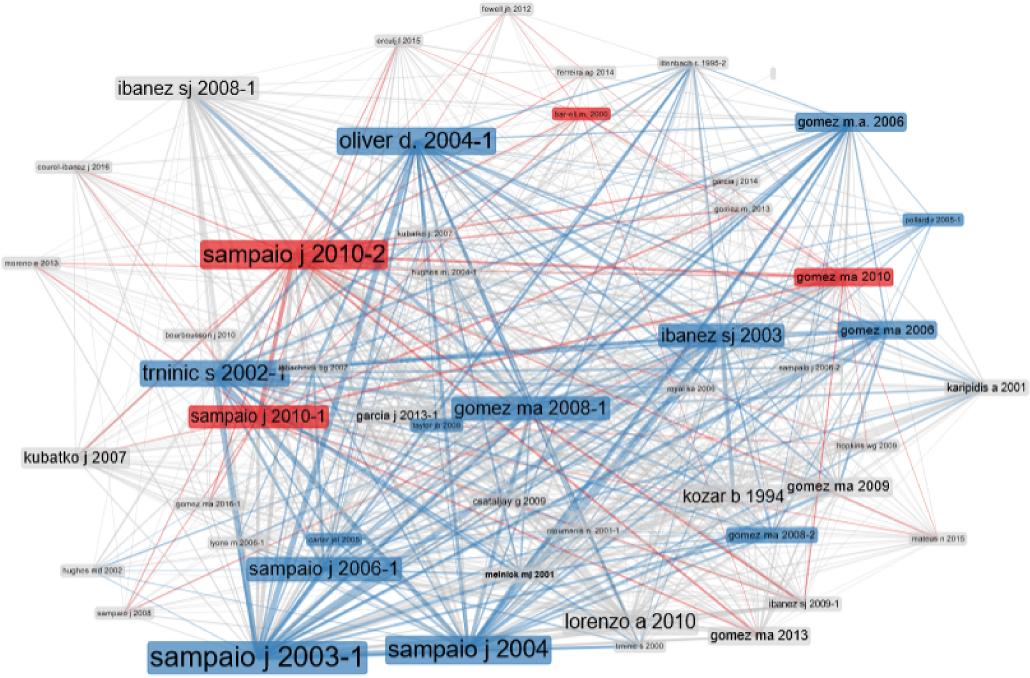


Figure A.9: Network of citations

Figure A.11 shows the network collaboration among Universities: again, Iberian institutions are playing a central role.

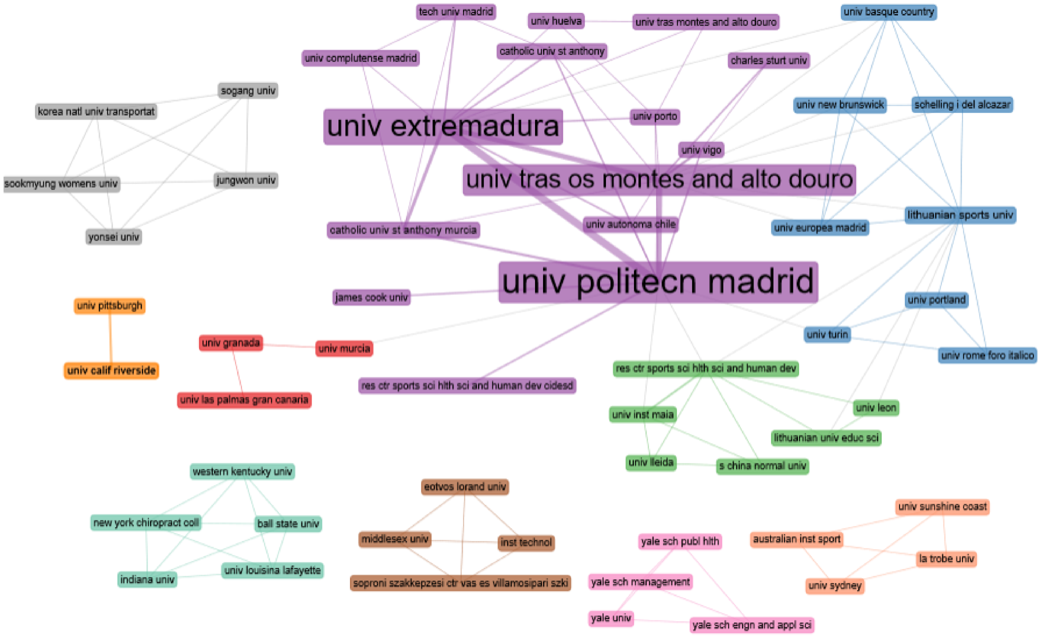


Figure A.11: Network of collaboration among institutions

In conclusion, Bibliometrix has demonstrated to be highly valuable in identifying papers, authors, journals, institutions to be investigated in approaching a domain, helping in selecting and focusing on more important items and their relationships.

A.2 Text Mining of searches about COVID-19

Remarks Some of the contents included in this section have been published (in Italian) in:
Statistica & Società n. 1/2020
Text Mining delle Ricerche su COVID-19: Cosa, Chi e Dove.
M. Migliorati, M. Carpita
www.rivista.sis-statistica.org/cms/?p=1159

A.2.1 Introduction

The bibliographic search about basketball and data science summarized in Appendix A.1 showed how it is not so simple to identify the subset of papers effectively relevant to address specific research questions. In this perspective, we started to study the application of NLP to bibliographic items, to make that goal easier. In the meanwhile, the new coronavirus COVID-19 pandemic was broken out, and the global health emergency caused in the first months of 2020 by the COVID-19 made the need for reliable scientific information even most pressing, both to facilitate contacts between research groups and to reduce fake news spread.
In a full epidemic scenario, one of main concerns is to quickly spread research results[2], enabling people facing pandemic to take into account new ideas. In this scenario, we decided to apply NLP and `Bibliometrix` [2] to COVID-19 bibliography. In this appendix the results of applying those technologies to COVID-19 bibliography with the goal of identifying the most relevant topics, the main scientific information sources and the more active countries are reported.

A.2.2 Analysing textual dataset using both bibliographic and NLP tools

For this analysis, we have taken into account 343 scientific publications (papers, letters, news and other indexed electronic material) selected from WoS for the period January, 1th - March, 30th 2020, containing at least one of following keywords (no upper and lowercase distinction): COVID-19, COVID2019, COVID-2019, COVID 2019, COV19, 2019-nCoV, SARS-CoV-2, SARS-Cov2, coronavirus disease 2019, Novel Coronavirus, New Coron-

[2] for instance, the open research archive specifically created to face this emergency named CORD-19, made available by the Allen Institute for Artificial Intelligence [1], in March 2020 was counting more than 45.000 scientific publications and, among them, 33.000 full text papers about corona-virus.

avirus.
The obtained 343 publications firstly have been classified via the so called topic modelling [3] approach, a procedure widely adopted in NLP to clusterize a corpus of documents on the basis of latent topics (identified considering documents' words) they refer to. On the basis of results[3] obtained analysing title, abstracts and keywords, the publications have been subdivided in three sets. First 5 top papers of each group have then been analyzed to understand the macro-topic characterizing each set, and we named the tree groups as:

1. COVID-19 spread and social impact (100 publications)
2. relations between COVID-19 and already known viruses (109 publications)
3. COVID-19 diagnosis, transmission and clinical features (132 publications).

Then, the three publications' groups have been analyzed using visualization techniques provided by the `Bibliometrix` software.

A.2.3 Some results

As an example, results about the third cluster (related to COVID-19 diagnosis, transmission and clinical features) are reported in Figure A.12, where:

1. the Co-occurrence Network shows the graph of relations among keyword groups; in this chart it is possible to identify four keyword groups: two related to different COVID-19 names, one related to risk management in Wuhan city (where the virus was firstly identified) and the last one, not linked to other keywords group, related to diagnosis and clinical treatment.
2. The Collaboration Network represents the graph of relations among countries of publications' authors. In the Collaboration Network three country clusters are identified: the first one centered on China (but with an unexpected presence of Spain and Cyprus), the second one related to USA and Europe (but with the unexpected presence of Saudi Arabia), the third one isolated and including only two countries (Germany and Sweden).

[3] The topic model algorithm was not able to classify 2 papers with sufficient certainty.

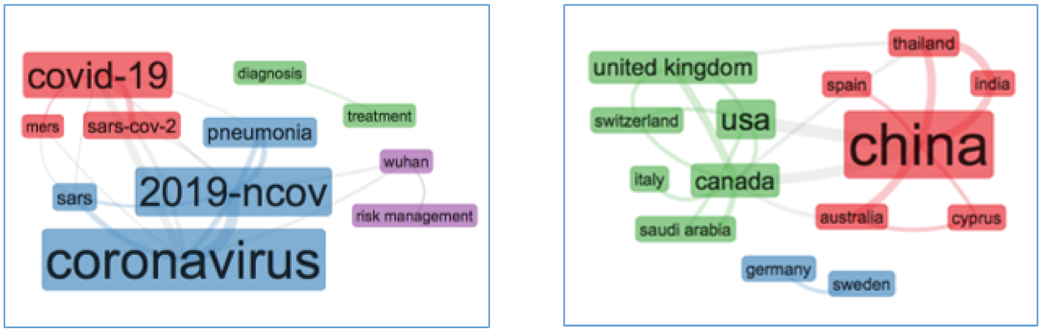


Figure A.12: Co-occurrence (on the left) and Collaboration (on the right) networks for the cluster of documents related to COVID-19 diagnosis, transmission and clinical features

Also the Three-Field Plot reported in Figure A.13 is about the 132 publications classified as related to diagnosis, transmission and clinical features of COVID-19 (WoS 30.03.2020), and it shows relationships between the more important keywords [4] (to the left), research institutions (in the middle) and reviews (to the right).

In addition to the generic keywords as "pneumonia" (characterizing only the 0.2% of publications), there are terms as "d-dimer" and "coagulation process" that could be of particular interest. Researchers, following with the mouse the colored rectangles of the chart, can highlight the number of related publications and, also in this case, filter the most relevant ones. The Three-Field plot can be easily modified by replacing the institution names with the main authors (typically the contact person of the research group), and in this way it is possible to relate keywords to journals through main authors.

In WoS, the number of scientific publications about COVID-19 is growing every day, and it is possible to visualize and navigate them using the statistical techniques presented in this Appendix.

Further possibilities are related to the analysis of other open research oriented databases: among others arXiv, bioRxiv and medRxiv [4]. A larger availability and a better quality of these archives, together with an improved usage of the analysis and visualization techniques for these data, can promote a faster spread of scientific knowledge, with an actually higher efficiency than in the past.

[4] To improve readability, in this chart the term COVID-19 and its synonyms have not been plotted.

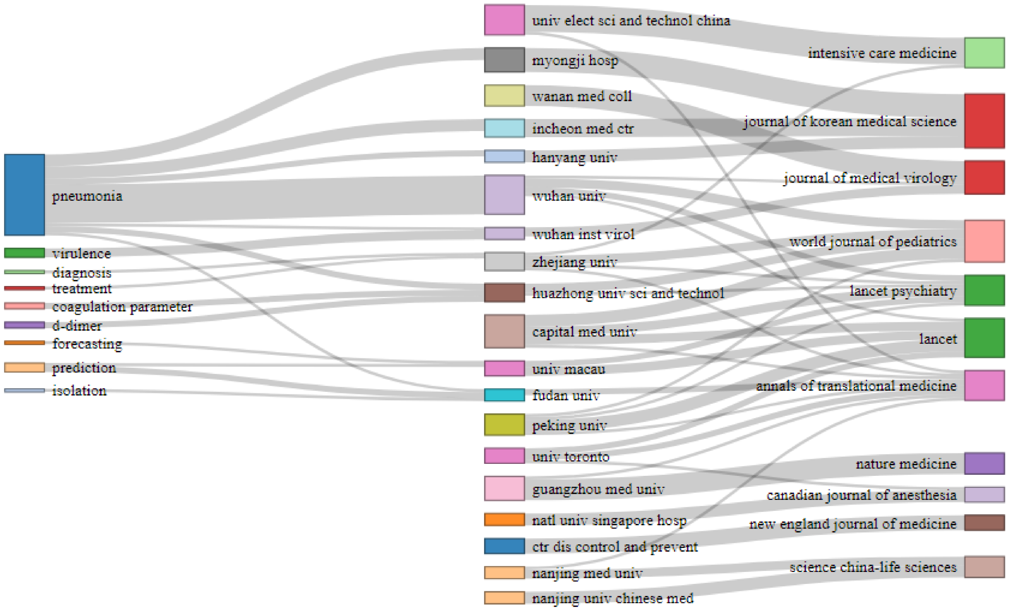


Figure A.13: Visualisation of relationships among keywords, research institutions and reviews for 132 publications about diagnosis, transmission and clinical features of COVID-19 (WoS 30.03.2020)

A.2.4 References

Databases and methodologies:

[1] pages.semanticscholar.org/coronavirus-research

[2] www.bibliometrix.org

[3] www.tidytextmining.com/topicmodeling.html

[4] www.ncbi.nlm.nih.gov/pubmed/; www.arxiv.org; www.biorxiv.org; www.medrxiv.org

A.3 h^*, an index to measure the credibility of preprints in the COVID-19 Open Research Dataset (CORD-19)

Remarks Some of the contents presented in this section have been presented in:
SIS2021, 50th Meeting of the Italian Statistical Society
University of Pisa,June 21th-25th 2021
in a short paper entitled:
Using Google Scholar to measure the credibility of preprints in the COVID-19 Open Research Dataset (CORD-19)
M. Migliorati, M. Carpita, E. Brentari

A.3.1 Introduction

The COVID-19 crisis highlighted, among others, the difficulty of finding reliable scientific information as soon as they are produced, selecting them in the huge amount of the available material [7]. The typical peer review process, adopted by scientific editors, normally takes several months before a paper is published, and this long period is not acceptable in crisis situations, when relevant information must be made accessible as soon as possible. From the other side, open-access preprints repository (as arXiv, bioRxiv and medRxiv) enable posting of scientific material in a very short time, but without offering any guarantee about the reliability of published contents. In this Appendix we propose a possible solution to this issue, by defining a credibility measure, named h^*-index, to be assigned to the preprints posted in repositories, driving users in a proper selection of materials.
The basic idea is to retrieve and summarize in a single measure some data (automatically retrieved accessing a free source as Google Scholar) about each author of a preprint (h-index, number of co-authors and length of scientific career), and assigning the higher value among authors to the preprint. First results, derived from applying the procedure to a sample of 100 preprints randomly selected from each of the three repositories, are really encouraging, showing how not only preprints, but also archives themselves can be evaluated in this credibility perspective.
Results show that Google Scholar is a wide-ranging bibliographic resource relatively simple to be used and providing several useful information. Nevertheless, it is not always precise in returning expected results, due both to unregistered authors (registration is on a voluntary basis) and some well-

known issues in authors' naming (consistency, homonymy, diacritics) [3]. This Appendix is structured as follows: section A.3.2 describes the dataset, section A.3.3 illustrates the proposed procedure, section A.3.4 summarizes the main findings and section A.3.5 presents some conclusions and future research directions.

A.3.2 The CORD-19 dataset

The CORD-19 dataset [9] is a "free resource of tens of thousands of scholarly articles about COVID-19, SARS-CoV-2, and related coronaviruses for use by the global research community". The Kaggle site [6], where it is possible to find this dataset, too, explains that "In response to the COVID-19 pandemic, the White House and a coalition of leading research groups have prepared the COVID-19 Open Research Dataset (CORD-19)".

This dataset contains COVID-19 and coronavirus-related research (e.g. SARS, MERS, etc.) from several sources (e.g. PubMed's PMC, Microsoft Academic, World Health Organization, arXiv, bioRxiv, medRxiv), and offers the possibility of downloading the *metadata.csv* file, a single repository where each contribution is normalized in terms of contents (offering, among the others, title, abstract, authors and source, i.e. all the features needed for our further analyses). We worked on a dataset dated July 2020, counting 192,509 references. For our purposes, the file was filtered on the basis of the source (we are interested only in preprints repositories, i.e. arXiv, bioRxiv, medRxiv) and furtherly restricted to preprints containing some COVID-19 related keywords or in the title either in the abstract[5]. In the end, we obtained a set of 8,186 preprints.

For each of the 3 archives we selected a random sample of 100 preprints, constituting the starting point of the analysis.

A.3.3 The procedure

A.3.3.1 The credibility h^*-index

The credibility h^*-index is based on the well-known h-index [4], modified to compare researchers that work in different scientific fields and with different lengths of scientific careers. A huge number of h-index variations has been proposed (see [1] for a review), and our approach integrates some of these variations.

[5] For that, we used the same query as CORD-19, an OR condition on terms *COVID-19*, *Coronavirus*, *Corona virus*, *2019-nCoV*, *SARS-CoV*,*MERS-CoV*, *Severe Acute Respiratory Syndrome*, *Middle East Respiratory Syndrome.*

A simple procedure to calculate the credibility of a scientific preprint based on co-authors' h-index is defined in the following way:

1. for each co-author three variables must be considered:
   h: the classical h-index
   t: the total number of co-authors of the h-papers, i.e. the papers considered in the h-index definition, including the author under investigation
   a: the length of the scientific career, defined as
   $year_{preprint} - year_{oldest-h-paper} + 1$

2. co-author h^*-index is calculated as follows:

$$h^* = \frac{h}{m \times a} = \frac{h^2}{t \times a} \tag{A.1}$$

   where

$$m = t/h \tag{A.2}$$

   is the average number of co-authors considering all the h-papers (the idea of dividing h by m can be found in [2], the idea of dividing h by a can be found in [4]). The h^*-index ranges from 0 (h-index=0 for the author) to h (the author wrote all its h-papers alone, in the same year of the preprint under investigation).

3. the credibility index of the preprint is the highest h^* among the h^* of all the n co-authors as in [5]:

$$h^*_{preprint} = max\{h^*_{co-auth_1}, .., h^*_{co-auth_n}\} \tag{A.3}$$

A.3.3.2 Using Google Scholar

An R procedure accessing Google Scholar to retrieve data for calculating the h^*-index for a preprint, starting from the title and the authors' list, has been developed. Google Scholar services are studied to be accessed only via browser (no ad hoc API are available), so the only feasible procedure is based on simulating an interactive access via a browser. To do that:

- a string representing a query to Google Scholar for the preprint title and authors' names is built, respecting the syntax established for browser interaction

- that query string is executed invoking the right verb (*GET*, in this case) to be used in the HTTP protocol driving the browser interaction

- An answer to the query string, constituted by HTML results' pages, is produced by Google Scholar

- HTML results' pages must be parsed to retrieve the data we are looking for.

Google Scholar policy suggests of avoiding BOT accesses: IP users addresses are banned if they do too many or too fast requests. Consequently, random delays between two and four minutes were inserted between two consecutive accesses, to comply with that access policy.
Apart from details, the algorithm is based on the following three main steps, repeated for all the preprints in the sample:

- starting from the preprint title, grab Google Scholar pages to find the authors' Google Scholar identifiers (they are strings of 12 characters, the key for querying the system)

- grab the Google Scholar pages using title and author's ID to retrieve all the data needed for calculating the h^*-index for the author. To do this step, the R package `scholar2021` has been integrated in the implementation.

- calculate the maximum h^* among authors, and assign it to the preprint.

A.3.4 Results

We applied the described procedure to a random sample of 100 preprints for each archive, obtaining the results in Table A.2.

Using h^*-index it is also possible to verify the level of credibility of a whole archive, as summarized in Figure A.14. Results show that arXiv seems containing "more credible" preprints with respect to the other two archives, showing an higher h^*-index average (0.52 VS 0.39 and 0.47 of bioRxiv and medRxiv, respectively), but a h^*-index distribution with higher variability and asymmetry too.

Table A.2: Some statistics for the preprint samples from the three archives in CORD-19.

Statistics	arXiv	bioRxiv	medRxiv
no. sample preprints	100	100	100
no. too-many-authors preprints	1	14	9
no. no-preprint	1	3	2
no. evaluable preprints	98	83	89
no. authors	388	723	621
no. unknown authors	158	445	422
% unknown authors	40.72	61.55	67.95
mean of authors per preprint	3.96	8.71	6.98
mean of unknown authors per preprint	1.61	5.36	4.74
no. preprints without h^*-index	9	4	22
no. preprints with h^*-index	89	79	67
total h^*-index	46.48	30.69	31.56
mean h^*-index	0.52	0.39	0.47
st dev h^*-index	0.28	0.18	0.24

Note. We didn't analyze preprints with more than 20 authors (too-many-authors in the table) for avoiding overstress Google Scholar. Moreover, due to the CORD-19 dataset under analysis, it can happen that a preprint is not more available (no-preprint in the table). Authors not found in Google Scholar (because not registered, or because of missed correspondence between CORD-19 and Google Scholar authors' names) are reported as unknown. Archive total h^*-index is the sum of h^*-index for archive sample preprints with a h^*-index. At last, the mean of authors per preprint is calculated as num authors / (num preprints with h^*-index).

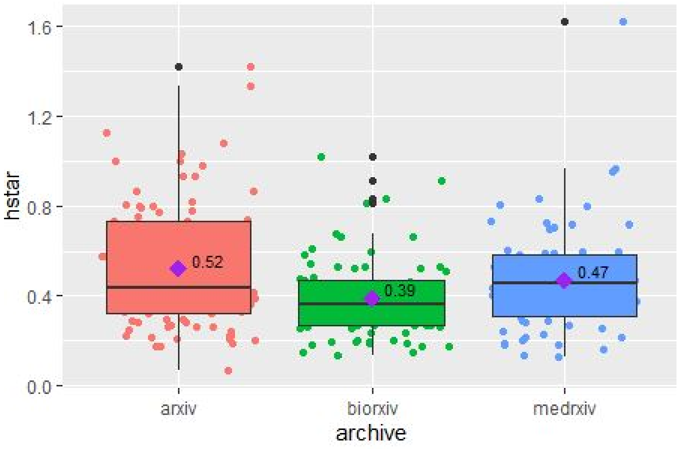


Figure A.14: The h^*-index box-plot for the preprint samples of the three archives in CORD-19.

The proposed approach is strongly based on the availability of data about the authors in Google Scholar. If an (important) author is not registered, the credibility index will suffer for this lackness, eventually producing a h^*-index lower than the real one.
Actually, in our analyses Google Scholar unknown authors effect is not so heavy on index calculation, apart from some relatively rare cases where important authors, with high number of citations, are not registered in that system.

A.3.5 Conclusions and futures directions

In this Appendix we described the procedure developed with the goal of attributing a "credibility" measure to preprints published in open access repositories as arXiv, bioRxiv, medRxiv,. This measure can be important in crisis situation as COVID-19, when it is necessary to select and access scientific papers as soon as possible, without delays due to peer review, but with a certain degree of credibility.
We described the h^*-index, based on the classical h-index and assigned to a preprint on the basis of the highest h^*-index value among co-authors. Results concerning the credibility of a sample of 100 preprints for each archive are reported, and it is shown how also archives can be compared with respect the credibility classification of their preprints. Results suggest that arXiv seems to be preferable with respect to other archives.
Future directions will address different definitions of the credibility index, taking into account definitions considering just a part of a scientific career (e.g. only the last five or ten years), studying its properties and comparing it with other indexes (e.g ten years h^*. Moreover, the dataset will be updated and the sample size increased. At last, the coverage offered by Google Scholar will be further investigated.

A.3.6 References

[1] Alonso, S., Cabrerizo, F.J., Herrera-Viedma, E., Herrera, F.:h-index: A Review Focused in its Variants, Computation and Standardization for Different Scientific Fields. Journal of Informetrics 3:4, 273-289, 2009

[2] Batista, P., Campiteli, M., Kinouchi O.: Is it possible to compare researchers with different scientific interests? Scientometrics, 68, 179-189, 2006

[3] Demetrescu, C., Ribichini, A., Schaerf, M.: Accuracy of author names

in bibliographic data sources: an Italian case study. Scientometrics 117, 1777–1791, 2018

[4] Hirsch, J.E.: An index to quantify an individual's scientific research output, Proceedings of the National Academy of Sciences 102(46):16569-16572, 2005

[5] Hirsch, J.E.: halpha: An index to quantify an individual's scientific leadership. Scientometrics 118, 673–686, 2019

[6] www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge, accessed 25/02/2021

[7] https://blogs.lse.ac.uk/impactofsocialsciences/2020/09/23/are-preprints-a-problem-5-ways-to-improve-the-quality-and-credibility-of-preprints/, accessed 25/02/2021

[8] Guangchuang, Y. and Keirstead, J.: scholar: analyze citation data from Google Scholar.
R package version 0.2.2, https://CRAN.R-project.org/package=scholar, 2021

[9] Wang, L.L., Lo, K., Chandrasekhar, Y., Reas, R., Yang, J., Eide, D., Funk, K., Kinney, R.M., Liu, Z., Merrill, W., Mooney, P., Murdick, D., Rishi, D., Sheehan, J., Shen, Z., Stilson, B.B., Wade, A.D., Wang, K., Wilhelm, C., Xie, B., Raymond, D.M., Weld, D.S., Etzioni, O., Kohlmeier, S.: CORD-19: The Covid-19 Open Research Dataset. arXiv, 2020