UNIVERSITY OF CALIFORNIA
Los Angeles

Goal-Oriented Forecasting: Predicting Soccer Match Outcomes with Deep Learning

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Applied Statistics

by

Sheng Chen

2024

ABSTRACT OF THE THESIS

Goal-Oriented Forecasting: Predicting Soccer Match Outcomes with Deep Learning

by

Sheng Chen
Master of Applied Statistics
University of California, Los Angeles, 2024
Professor Guido Montúfar, Chair

Soccer, often referred to as "football" in the heart of Europe, has a deep-rooted cultural significance that transcends national boundaries. The sport's appeal extends far beyond the pitch, encompassing a wide array of enthusiasts, from die-hard fans to data-driven strategists. In recent years, the fusion of deep learning models with the captivating world of football has taken center stage, revolutionizing our approach to predicting match outcomes. We delve into the fusion of state-of-the-art artificial intelligence and machine learning techniques with the intricacies of a sport that inspires fervent devotion. Our aim is straightforward: to unearth the potential of deep learning models in enriching our capacity to anticipate which team will emerge victorious on the hallowed turf.

We will delineate the primary objectives of this research essay, elaborate on the methodology employed, and elucidate our anticipated contributions to the existing body of knowledge in both the realms of deep learning and sports analytics. Finally, we will underscore the significance of precise football match outcome prediction as an evolving and multi-dimensional research domain that holds great promise for aficionados, professionals, and researchers throughout Europe and beyond.

The thesis of Sheng Chen is approved.

Yingnian Wu

Paik Schoenberg

Guido Montúfar, Committee Chair

University of California, Los Angeles

2024

To my grandpa ...

who dedicated years fostering

my mathematical education

and patience to overcome any difficulties

TABLE OF CONTENTS

List of Figures . . . . . **vii**

List of Tables . . . . . **viii**

1 Introduction **1**

2 Data Collection **2**

2.1 Web Hierarchy 2

2.2 Objective 2

2.3 Organizing Data 3

2.4 Exploratory Data Analysis 3

2.4.1 PCA 4

3 Methodology **6**

3.1 Data Augmentation 6

3.1.1 SMOTE and its variations(borderline, ADASYN) 6

3.1.2 Tabular GAN 7

3.2 Feature Selection 7

3.2.1 Exhaustive Search 7

3.2.2 Lasso 8

3.2.3 Mutual Information 8

3.3 Models 8

3.3.1 Linear Regression 8

4 Result **9**

5 Future Work **10**

LIST OF FIGURES

LIST OF TABLES

2.1 Outcome with skewed class proportion 4

2.2 Logistic Regression Accuracy after kernel PCA 5

CHAPTER 1

Introduction

In the world of soccer, predicting game outcomes and player performances has always been a challenging task. With the increasing availability of data, including statistics, team lineups, and player profiles, there is a growing interest in using data analysis and machine learning techniques to gain insights and make predictions. This document outlines a comprehensive approach to soccer game prediction, focusing on the Spanish and German leagues during the seasons 2017-2018 and 2021-2022. We aim to understand which team features play the most important role in goal difference, predict future game outcomes, and explore the possibility of predicting games with different starting lineups.

Modern methods are being used to analyze data. Foremost among these is the use of Expected Goals (xG), a statistical tool by Opta that carefully measures how likely a team is to score based on factors including the quality and number of scoring chances created. In the field of betting, where more advanced models have become popular, among which are Machine learning algorithms like Random Forests, Gradient Boosting, and deep neural networks. All these methods share the same idea: use past match data, player performance info, and even factors like weather and injuries to make accurate game predictions. In-play betting has made things more exciting by allowing changes to betting odds while a game is currently underway. Crucial for accurately predicting the outcome, specific measurements are taken to quantify how well both teams and individual players perform. This includes things like where players are on the field, how accurate their passes are, and factors related to their mindset. Furthermore, the industry closely monitors market performance and betting trends, enhancing their ability to comprehend the game's evolution over time.

CHAPTER 2

Data Collection

2.1 Web Hierarchy

The data utilized in this study was sourced from fbref.com and encompassed statistical information from the five major European football leagues. This dataset covers a span of five seasons, specifically from the 2017-2018 season to the 2021-2022 season. To compile this dataset, information was extracted from a total of 20 league tables. Notably, the dataset comprises detailed information on 380 (or 306 for Bundesliga) matches for each league per year. This comprehensive dataset serves as a fundamental component of the research conducted in this thesis. We employed BeautifulSoup (bs4) to automatically extract links from each league page, which allows us to access detailed information about the games.

2.2 Objective

Which (team) feature plays the most important role in goal difference? Can we predict the goal difference of future games? Can we predict the goal difference of future games given different starting lineups? All three questions can be considered as a self-supervised learning problem since we are using historical data on teams.

Idea for objective 1: This is a classic regression problem with an emphasis on the interpretation of the coefficients, which can give us information on each feature's impact on the output variable.

Idea for objective 2: we can calculate a weighted average of team performance over the past N games to substitute for each row of the model matrix and predict future results.

Idea for objective 3: As an extension of the previous question, we can calculate a weighted average of player performance over the past N games, and this data is used to assess the current form of players in the upcoming games so that based on game dates, we can pull out each player's past five performance and adjust the team data using some function (eg. sum) and adjust the model matrix accordingly. We maintain a "Player Name" dictionary to link players to their respective statistics. A player's level of performance relative to their potential level of performance is called form. Since soccer games are strongly influenced by the current form of the team. Hence, as we consider player stats as a vital aspect of our analysis, we include this factor of fluctuation in performances.

2.3 Organizing Data

We organize data into tabular format, with rows being each game and columns being the team features. To investigate the questions of interest, we need three different matrices. All three matrices should have the same dimensions. The first matrix we need is the team-performance matrix, with rows indicating each game, and columns that include all team features. The columns can also be separated by home and away team features. The second matrix we need is the past-team-performance matrix, with each row replaced by the weighted average of the past n performances from the two teams. N can be a hyperparameter to tune later on. The last matrix we need is the past-player-performance matrix, each row replaced by the aggregate (maybe sum, or other function f) of the weighted average of the past n performances of all players from the two teams. Ideally, the player matrix should give us the most information about the performance and thus predict more accurately, but it is computationally harder to form such a matrix due to data management.

2.4 Exploratory Data Analysis

We investigate the proportion of our output variable, the goal difference, which contains possible values from -9 to +9 excluding -8. If we look at the value counts, it is easy to

-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9
1	0	3	4	44	134	351	762	1481	2229	1815	1072	550	236	99	30	6	4	2

Table 2.1: Outcome with skewed class proportion

notice that there are few cases of +9, -9, and other classes of large absolute values, making it an extremely imbalance classification problem. We have several ways of dealing with such problems, including removing outliers, generating data, or using weighted loss

2.4.1 PCA

Principle Component Analysis utitlizes some of the basic facts in linear algebra to apply data compression. Here, we use PCA to observe the explained variance before we start the modeling. Assume the data consists of points $\{\mathbf{x^{(1)}}, \mathbf{x^{(2)}}, ..., \mathbf{x^{(m)}}\} \in \mathbb{R}^{\mathbf{n}}$. We need some encoding function $\mathbf{f}$ and decoding function $\mathbf{g}$, where $\mathbf{c^{(i)}} = \mathbf{f}(\mathbf{x^{(i)}})$ lies in the lower-dimensional space $\mathbb{R}^{\mathbf{l}}$. For basic PCA, we use the simplest matrix product for the decoder $\mathbf{g}(\mathbf{x^{(i)}}) = \mathbf{D}\mathbf{x^{(i)}}$ with $\mathbf{D} \in \mathbb{R}^{m \times l}$. The goal is to minimize the construction loss:

$$\mathbf{c}^* = \arg\min_{\mathbf{c}} \|\mathbf{x} - \mathbf{g}(\mathbf{c})\|_{\mathbf{2}}^{\mathbf{2}}$$

where we find the By solving the above, we get $\mathbf{c} = \mathbf{D^T x}$, then reconstruction loss becomes

$$\arg\min_{D} \|\mathbf{X} - \mathbf{D}\mathbf{D^T}\mathbf{X}\|_{Frobenius}$$

, where $\mathbf{X} = [\mathbf{x^{(1)}}\, \mathbf{x^{(2)}} \, ... \, \mathbf{x^{(m)}}]^{\mathbf{T}}$. This optimization problem is solved through eigendecomposition with the optimal D obtained through vertically combining the n eigenvectors of $\mathbf{X^T X}$ that corresponds to the n largest eigenvalue.

In kernel PCA, we replace $\mathbf{X}$ with function $\phi(\mathbf{X})$ that is able to operate in high dimensions, without ever computing in that space, but rather the dot product in a low dimension. If we use PCA with different kernel functions, we get the following logistic regression accuracy.

Kernel Function	Accuracy
linear	0.26096866
poly	0.25811966
rbf	0.25754986
sigmoid	0.26267806
cosine	0.25868945

Table 2.2: Logistic Regression Accuracy after kernel PCA

CHAPTER 3

Methodology

The first step when we get our training data is to make sure the imbalanced problem is addressed. There are typically two ways of dealing with class imbalance: one is through data augmentation, and the other is using loss function with re-weighting terms. Here, we choose to apply several different data augmentation techniques, and later we compare the results on modeling testing data with the method without augmentation but with a reweighting loss function.

3.1 Data Augmentation

To apply some of the techniques below, there is a minimum number of points required (theoretically we can do 2, but with a lower number of observations, newly generated points would all be near the line connecting the original two points). For our dataset, which contains some classes with numbers of observations less than 10, I decided to remove these extreme outliers in our classification modeling and left us with the following outcome distribution.

3.1.1 SMOTE and its variations(borderline, ADASYN)

The Synthetic Minority Over-sampling Technique(SMOTE) is a technique to generate synthetic tabular data. All variations of this oversampling technique create points by using a convex combination of points that belong to the selected minority classes, and by repeatedly applying this method to expand all classes to the same number of counts, which, in our case, is 2229.

Quickly running a linear regression after borderline SMOTE gives us fig. 4

3.1.2 Tabular GAN

For tabular GAN and Conditional GAN, both methods emulate the distribution of the training matrix and outcome vector. Again, we are attempting to generate data using TGAN so that each outcome class has an equal count. Therefore, we split the data one by one for each class and generated the required number for each class.

3.2 Feature Selection

In our case, feature selection is used as a preprocessing step in conjunction with machine learning models for regression/classification purposes. The purpose is to find the best feature subset with a determined number of features. Feature selection methods can be classified into wrapper, embedded, filter methods, or the hybrid of the previous three methods. Wrapper methods treat the classifier as a black box and select features based on the classification result using some metric (eg. forward, backward selection); embedded methods carry out selection through the classifier algorithm itself (e.g. regularization, tree-based algorithms); filter methods have the selection process independent of classifier algorithms (e.g. univariate ANOVA). [feature selection review] Typically, filter methods are the fastest, and wrapper the slowest.

3.2.1 Exhaustive Search

Assume we have fixed the number of features to select, denote it as k. We can implement a greedy search of $\binom{\mathbf{n}}{\mathbf{k}}$ number of possible subsets of size k to find the "optimal" subset based on the metrics we choose. In this case, we use logistic regression accuracy for the sake of computational efficiency, and set $k = 3$.

3.2.2 Lasso

Lasso, or L1 regression, is a common regularization technique based on linear regression. It adds a weighted L1 norm of the coefficient vector to the least squares function. It is not hard to notice, once the weight was set large enough, some elements in the coefficient vector will diminish to zero, and thus performing feature selection while doing regression(embedded feature selection).

3.2.3 Mutual Information

Mutual information (MI) is a measure of the amount of information that one random variable has about another variable, and in our case, it is used to quantify and compare the relevance of different feature subsets concerning the output variable. Since all of our dataset's features are continuous, we need nonparametric methods based on entropy estimation from k-nearest neighbors distances as described in Estimating Mutual Information and mi between discrete and continuous. The result is implemented by the mutual_info_classif function in scikitlearn.feature_selection. We apply univariate Mutual Information between each feature and output variable to perform this filter method.

3.3 Models

3.3.1 Linear Regression

CHAPTER 4

Result

CHAPTER 5

Future Work