UNIVERSITÀ
DEGLI STUDI
DI MILANO

Applying Statistical Learning Techniques to Understand Professional Women's Tennis

Samaher Brahem

A report submitted as part of a Statistical Learning project

for the degree of

Master of Science in Data Science for Economics

Examiner:

Prof. Silvia Salini

May 2024

Abstract

This project explores the application of both unsupervised and supervised learning techniques in analyzing Women's Tennis Association (WTA) data. The unsupervised aspect involves clustering the top 30 players, while the supervised part focuses on building predictive models for match outcomes. The dataset comprises a variety of player performance metrics, attributes, and match characteristics. In the unsupervised learning phase, methods such as principal component analysis (PCA), k-medoids, and hierarchical clustering are used to uncover player segments and patterns within the data. On the supervised learning front, logistic regression, classification tree, and random forest models are employed to predict match outcomes based on player and match attributes.

Keywords: Women Tennis Association (WTA), Hierarchical Clustering, K-means, PCA, Classification Tree, Random Forest, Logistic Regression.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This first chapter serves as the introduction to this report, providing a concise overview of the tennis game rules, context and objectives of this project, and understanding the dataset used in the analysis.

1.1 Tennis 101

Tennis is a racquet sport that can be played in singles (one vs. one) or doubles (two vs. two) formats. However, for the purpose of this project, the focus is specifically on exploring and modeling women's singles tennis. The sport follows a hierarchical structure: players compete for points to win games, which are then accumulated to secure sets, and ultimately sets are gathered to clinch the overall match. A point in tennis starts with the server delivering the ball to the opposing player, known as the receiver, to initiate a rally. The player who wins the rally earns the point, unless the server fails to make a valid serve within two attempts, in which case the point goes to the receiver by default. To win a game, a player must accumulate a minimum of four points with a lead of at least two over their opponent. Matches are decided as best of 3 in women's tennis. Throughout the year, tennis players participate in various tournaments, earning ranking points based on their performance. These points determine their official rankings and eligibility for future tournaments. The top tier of women's tennis is the WTA tour, featuring 58 tournaments

of varying prestige held annually over one to two weeks, often with multiple tournaments occurring simultaneously.

In the table below, definitions of key terms related to tennis are provided. These terms are commonly used in discussions and analyses of tennis matches and tournaments. Each term is accompanied by a brief explanation to aid in understanding its significance within the context of the sport.

Term	**Definition**
Serve	The action of a player hitting the ball to start a point.
Point	A unit of scoring in tennis, awarded to the winner of a rally.
Game	A collection of points. To win a game, a player must win at least four points with a lead of at least two.
Set	A collection of games. To win a set, a player must win at least six games with a lead of at least two.
Double Fault	A fault that occurs when a player fails to make a valid serve on both attempts, resulting in the loss of the point.
Break Point	A situation where the receiver has the opportunity to win the game of the server.
Ace	A serve that the receiver fails to touch with their racket, resulting in the server winning the point.
Grand Slam	The four major tennis tournaments: Australian Open, French Open, Wimbledon, and US Open.

Table 1.1: Definitions of Tennis Terms

1.2 Context & Objectives

This project is conducted within the framework of the Statistical Learning course led by Professor Silvia Salini at the University of Milan. Its core objectives entail the exploration and application of various statistical learning methodologies, encompassing both supervised and unsupervised learning paradigms.

1.3 The Dataset

1.3.1 Data Description

Data Source. The dataset used for this project is freely available under a non commercial licence[1] and has been sourced from the GitHub repository maintained by Jeff Sackmann[2]. This repository, established nine years ago, encompasses comprehensive data pertaining to Women's Tennis Association (WTA) matches, commencing from the year 1968. Notably, the repository received its latest update just last month.

Data Initial Condition. The dataset consists of approximately 2,800 women's singles WTA tour matches in the year 2023. It comprises 50 variables, each detailing various aspects of both the winning and losing sides' performance in a match. These variables encompass metrics such as the number of aces, break points faced and saved, serves in and won, along with player attributes like age, height, dominant hand, rank points, and WTA rank. Additionally, the dataset includes match-specific details such as match duration, tournament name and level, and match round. A comprehensive list of these variables is provided in the subsequent table 1.3.1.

[1] `https://creativecommons.org/licenses/by-nc-sa/4.0/`

[2] `https://github.com/JeffSackmann/tennis_wta`

Variable	Description
tourney_name	Name of the tournament.
surface	Surface type: Hard, Clay, or Grass.
tourney_level	Level of the tournament.
winner_name	Name of the winner.
winner_hand	Dominant hand of the winner.
winner_ht	Height of the winner.
winner_ioc	Country code of the winner.
winner_age	Age of the winner.
loser_name	Name of the loser.
loser_hand	Dominant hand of the loser.
loser_ht	Height of the loser.
loser_ioc	Country code of the loser.
loser_age	Age of the loser.
score	Match score.
round	Round of the match.
minutes	Match length in minutes.
w_ace	Number of aces by the winner.
w_df	Number of double faults by the winner.
w_svpt	Number of serve points by the winner.
w_1stIn	Number of first serves made by the winner.
w_1stWon	Number of first-serve points won by the winner.
w_2ndWon	Number of second-serve points won by the winner.
w_SvGms	Number of serve games won by the winner.
w_bpSaved	Number of break points saved by the winner.
w_bpFaced	Number of break points faced by the winner.
l_ace	Number of aces by the loser.
l_df	Number of double faults by the loser.
l_svpt	Number of serve points by the loser.
l_1stIn	Number of first serves made by the loser.
l_1stWon	Number of first-serve points won by the loser.
l_2ndWon	Number of second-serve points won by the loser.
l_SvGms	Number of serve games won by the loser.
l_bpSaved	Number of break points saved by the loser.
l_bpFaced	Number of break points faced by the loser.
winner_rank	WTA rank of the winner.
winner_rank_points	Number of ranking points of the winner.
loser_rank	WTA rank of the loser.
loser_rank_points	Number of ranking points of the loser.

Table 1.2: Explanation of the Main Variables in the Initial Dataset

1.3.2 Data Preprocessing

Null Values. In this step, I eliminated matches that are not complete either due to injury of one of the players, walkover, or any other reason.

Score Column Transformation. I splitted the score variable that contains the overall score in 1 column into different columns that describe the score set by set as illustrated by the table 1.3.2 below.

Before Modification	**After Modification**
score: 6-4 6-2	w_set1 l_set1 w_set2 l_set2 w_set3 l_set3 6 4 6 2 0 0
score: 6-3 6-7(4) 7-5	w_set1 l_set1 w_set2 l_set2 w_set3 l_set3 6 3 6 7 7 5

Table 1.3: Transformation of the score variable

Grand Slam Match Identification. To highlight the fact whether a match is a Grand Slam match or not, I created a dummy variable "grand_slam" that takes the value of 1 if the match is a grand slam match (i.e. it is part of one of the four major tennis tournament as explained in 1.1) and 0 if not.

Final Rounds Match Identification. To highlight the fact that a match is in final rounds (Quarter final, Semi final, or Final) as opposed to preliminary rounds, I created a dummy variable "final_rounds" that takes the value of 1 if the match is in final rounds and 0 if not.

Removing Unnecessary Variables. In order to refine the analysis and focus on the essential variables, several columns have been excluded from the dataset. These omitted columns include various identifiers such as "tourney_id", "draw_size", and "match_num",

along with temporal data like "tourney_date". Additionally, details regarding player seeding ("winner_seed" and "loser_seed"), entry types ("winner_entry" and "loser_entry"), and match format ("best_of")—which typically adheres to a best-of-three format in women's tennis—have been removed. Furthermore, tournament level ("tourney_level"), participant identifiers ("winner_id" and "loser_id"), match outcome ("score")—transformed into a set-by-set score format—and match round ("round")—replaced by a categorical variable describing this information—have also been excluded. This curation process ensures that the analysis focuses solely on the most relevant variables for the project objectives.

Creating a Balanced Dataset. Although the match outcome was discernible from the winner and loser variables, a formal dependent variable column was lacking. This dependent variable required records of both match outcomes (win and loss) to ensure predictive accuracy of the model later in the analysis. Consequently, it was imperative to introduce a column delineating the match outcome from each player's standpoint—not solely the winner's or loser's. To establish parity between players, all columns pertaining to the winner (containing "winner" or "w") were relabeled as ("player" or "p"). Similarly, columns referencing the loser ("loser" or "l") were renamed as ("opponent" or "o"). Subsequently, the column "win" was created to document the match outcome from the player's perspective. To capture the opposite outcome as well, we replicated the dataset's records while interchanging the player and opponent information. Then, I merged both perspectives to create 1 balanced data set where approximately 50% of the matches are won and the other 50% are lost from the player perspective.

Narrowing Data Scope to Matches Involving the Top 100 Players. To ensure consistency and focus in the analysis, I made the decision to include only data associated with matches played by players ranked within the top 100 of the WTA rankings. This approach allows me to concentrate the analysis on matches involving top-ranked players, providing a more coherent and insightful examination of the data.

Scaling Numerical Variables. To ensure consistency in the analysis of variables with differing units of measurement, standardization was employed. This process involves transforming each variable to have a mean of 0 and a standard deviation of 1, facilitating easier comparison across variables. The standardization formula used is as follows:

$$\text{Standardized Value} = \frac{\text{Original Value} - \text{Mean}}{\text{Standard Deviation}}$$

This process is also known as z-score normalization and it ensures that all numerical variables are on the same scale, enabling more accurate and meaningful analysis.

Outliers. After exploring the variables' distributions as shown in the figure 1.1 and the boxplot of scaled numerical variables as shown in the figure 1.2 I found some outliers in the data. But after looking at each variable separately and considering what I know about the game, I realized that most of these outliers were actually valid data points, not mistakes. For example, the outlier identified in opponent/player height is a correct value. She's the American player Lauren Davis with only 1.57 m as height. So, I decided to keep the outliers because they provide valuable insights into the full picture. Nevertheles, I corrected the ones I spotted, like in this example (spotted from the minutes variable distribution 1.3) where in the dataset we have the length of the match between Ons Jabeur and Camila Osorio was 316 minutes but when I checked in the WTA official website it was 122 minutes only.

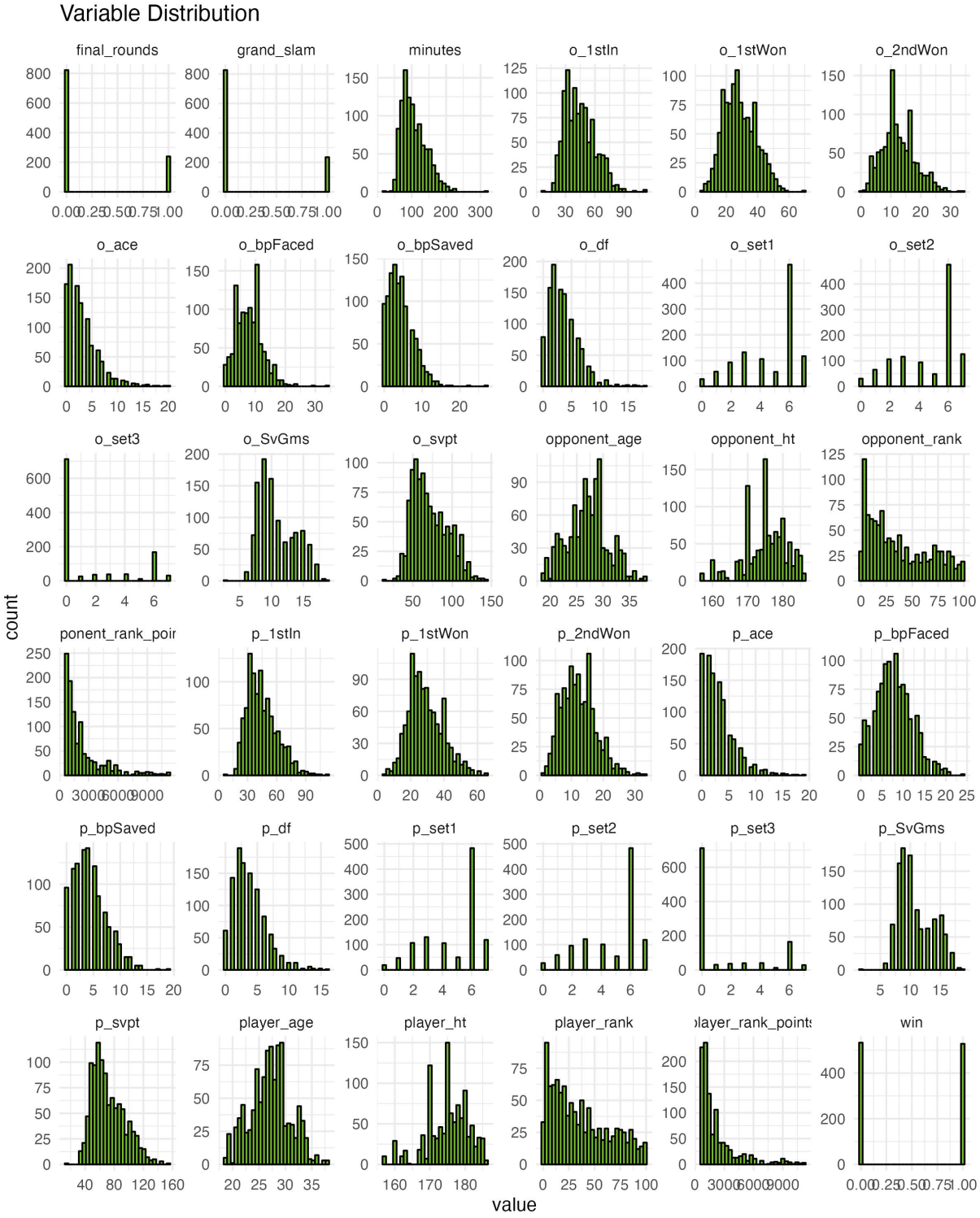

Figure 1.1: Variables Distribution

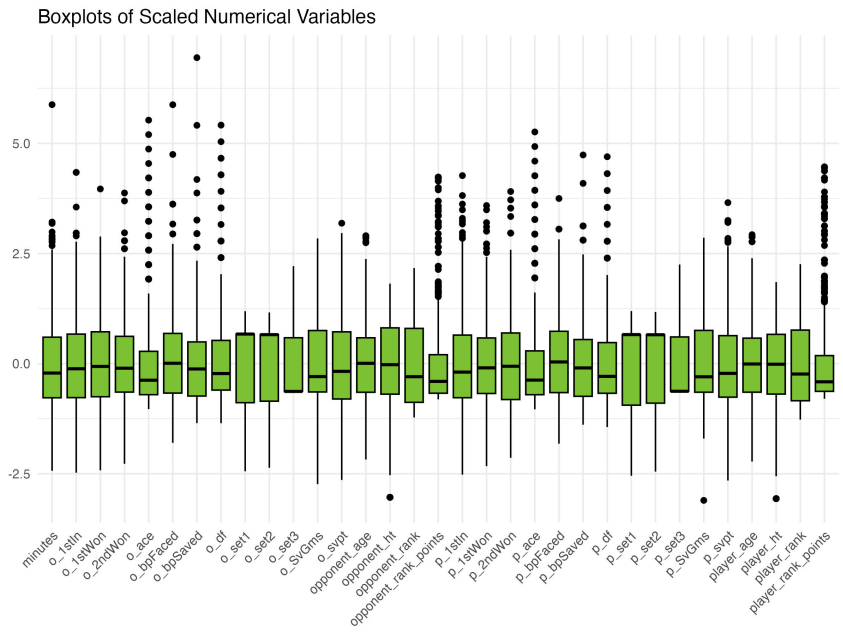

Figure 1.2: Boxplot of Scaled Numerical Variables

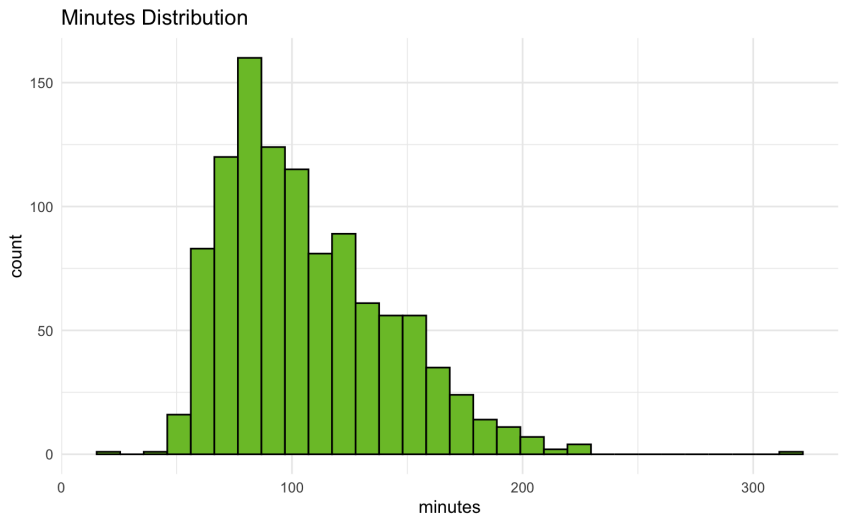

Figure 1.3: Minutes Variable Distribution

Chapter 2

Unsupervised Learning

This chapter discusses the unsupervised learning techniques used to explore this dataset. For this part of the analysis, I chose a subset of the dataset containing player-specific average match statistics, such as the average number of aces, double faults, serve points won, first serve points won, second serve points won, serve games played, average break points saved, and average break points faced. Additionally, I restricted the dataset to include only players ranked within the top 30. Here is a summary of the variables included in the subset:

- **player_name**: The name of the player.
- **avg_p_ace**: The average number of aces per match for the player.
- **avg_p_df**: The average number of double faults per match for the player.
- **avg_p_svpt**: The average number of serve points per match for the player.
- **avg_p_1stWon**: The average number of 1st serve points won per match for the player.
- **avg_p_2ndWon**: The average number of 2nd serve points won per match for the player.
- **avg_p_SvGms**: The average number of serve games played per match for the player.
- **avg_p_bpSaved**: The average number of break points saved per match for the player.
- **avg_p_bpFaced**: The average number of break points faced per match for the player.

2.1 Principal Component Analysis (PCA)

What is PCA? Principal Component Analysis (PCA) is a statistical technique used to reduce the dimensionality of a dataset while preserving as much of the variation in the data as possible. PCA achieves this by transforming the original variables into a new set of orthogonal variables called principal components.

The main idea behind PCA is to identify patterns in the data by finding the directions, or principal components, along which the data varies the most. These principal components are calculated in such a way that the first principal component accounts for the maximum amount of variation in the data, the second principal component (orthogonal to the first) accounts for the maximum remaining variation, and so on.

Let $\mathbf{X}$ be an $n \times p$ matrix representing the original data, where n is the number of observations and p is the number of variables. The first principal component, denoted as $\mathbf{z}_1$, is a linear combination of the original variables:

$$\mathbf{z}_1 = \mathbf{X}\mathbf{v}_1$$

where $\mathbf{v}_1$ is a unit vector representing the direction of maximum variance in the data, known as the first principal component loading vector. It is calculated by finding the eigenvector corresponding to the largest eigenvalue of the covariance matrix $\mathbf{S}$ of $\mathbf{X}$.

Subsequent principal components, $\mathbf{z}_2, \mathbf{z}_3, \ldots, \mathbf{z}_p$, are also linear combinations of the original variables, subject to the constraint that they are orthogonal to each other. Each successive principal component captures the maximum remaining variance in the data.

The transformation from the original variables to the principal components can be expressed as:

$$\mathbf{Z} = \mathbf{X}\mathbf{V}$$

where $\mathbf{Z}$ is the $n \times p$ matrix of principal components, and $\mathbf{V}$ is the $p \times p$ matrix of loading vectors, with columns $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_p$.

PCA can be used for various purposes, including dimensionality reduction, data visualization, noise reduction, and feature extraction. By retaining only a subset of the principal components that capture the most variance in the data, PCA can simplify complex datasets while preserving important information for further analysis.

Application After analyzing the output of the principal component analysis, we can see, as shown in the figure 2.1, that the percentage of variance explained by the first principal component and the second principal component combined reach approximately 76.4%.

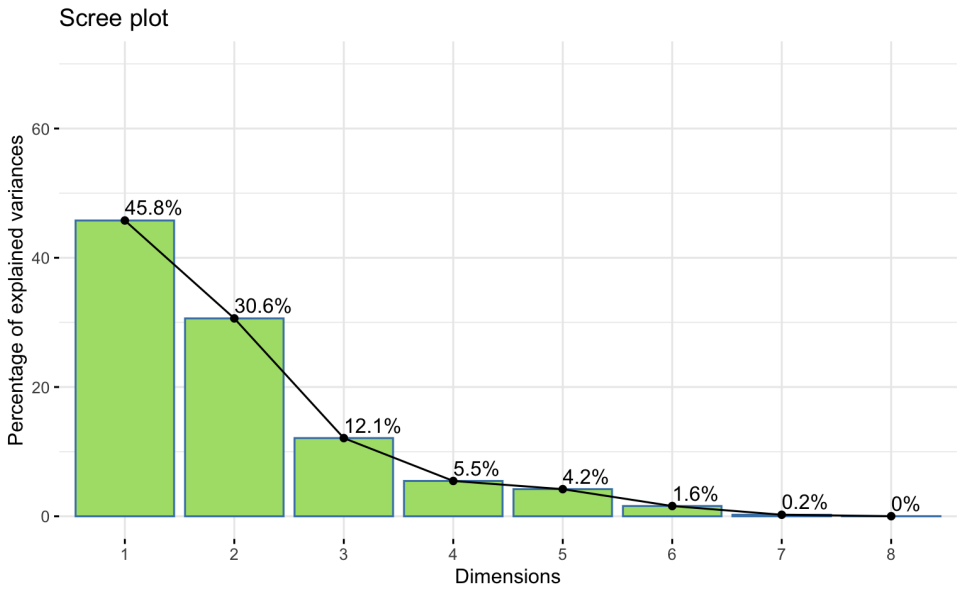

Figure 2.1: Variance Explained by Principal Components

This table 2.1 summarizes the loadings of the variables on the first two principal components, along with the standard deviations, proportions of variance, and cumulative proportions explained by each component.

Variable	**PC1**	**PC2**
avg_p_ace	0.026	-0.792
avg_p_df	-0.249	-0.328
avg_p_svpt	-0.469	-0.034
avg_p_1stWon	-0.279	-0.047
avg_p_2ndWon	-0.323	-0.343
avg_p_SvGms	-0.380	-0.072
avg_p_bpSaved	-0.431	0.202
avg_p_bpFaced	-0.453	0.313
Standard Deviation	1.5221	1.2450
Proportion of Variance	0.4577	0.3062
Cumulative Proportion	0.4577	0.7639

Table 2.1: Principal Component Analysis Results

Principal Component 1: PC1 demonstrates a strong negative influence on variables such as average serve points (**avg_p_svpt**), average break points faced (**avg_p_bpFaced**), average break points saved (**avg_p_bpSaved**), serve games played (**avg_p_SvGms**), and second serve points won (**avg_p_2ndWon**). This suggests that higher values of PC1 correspond to lower values of these variables. Overall, PC1 reflects players' performance in terms of match statistics, emphasizing aspects like serving effectiveness and resilience under pressure.

Principal Component 2: The most negatively influential variable for PC2 is the average number of aces (**avg_p_ace**), followed by the average number of double faults (**avg_p_df**). Conversely, PC2 shows a moderate positive association with break points faced (**avg_p_bpFaced**). This indicates that higher values of PC2 are associated with higher numbers of aces and double faults, while also suggesting that players exhibiting more aggressive serving strategies may face more double faults but fewer break points.

Figure 2.2: PCA Plot

Results In the PCA plot 2.2, each point represents a player, and their positions are determined by the values of PC1 and PC2 as shown in the table 2.1. The **light green** arrows indicate the first two principal component loading vectors. Let us interpret the color scheme based on the variables' squared cosine values (`cos2`).

- **Blue**: Players associated with variables in blue have a strong representation in the principal component space, indicating that their performance in terms of match statistics strongly influences the variability observed along PC1 and PC2. Specifically, players with higher values of PC1 tend to have lower values of variables such as average serve points, break points faced, break points saved, serve games played, and second serve points won.

- **Green**: Players associated with variables in green have a moderate representation in the principal component space. While their performance contributes to the variability captured by PC1 and PC2 to a lesser extent compared to players in blue, they still exhibit observable patterns in their match statistics.

- **Brown**: Players associated with variables in brown have a weak representation in the principal component space. Their performance in match statistics contributes minimally to the variability observed along PC1 and PC2, indicating that these players may not exhibit significant trends or patterns in their gameplay.

2.2 Clustering

2.2.1 K-medoids

What is K-medoids? K-medoids is a clustering algorithm that partitions a dataset into a predetermined number of clusters, where each cluster is represented by a medoid. Unlike k-means, which uses the mean of data points as cluster centers, the k-medoids algorithm uses actual data points as cluster representatives. This makes k-medoids more robust to outliers and noise in the data.

The algorithm iteratively assigns each data point to the nearest medoid, updates the medoids to minimize the total dissimilarity within each cluster, and repeats until convergence. The dissimilarity measure, often represented by a distance metric such as Euclidean distance, determines the similarity between data points and medoids.

The k-medoids algorithm can be formally defined as follows:
Given a dataset $X = \{x_1, x_2, \ldots, x_n\}$ consisting of n data points and a predetermined number of clusters k, the goal is to partition the data into k clusters $C = \{C_1, C_2, \ldots, C_k\}$, where each cluster is represented by a medoid $m_i \in C_i$ such that it minimizes the total dissimilarity within clusters:

$$\text{minimize} \sum_{i=1}^{k} \sum_{x_j \in C_i} d(x_j, m_i)$$

where $d(x_j, m_i)$ is the dissimilarity (distance) between the data point x_j and the medoid m_i. The algorithm iteratively updates the medoids and assigns data points to the nearest medoids until convergence, where the total dissimilarity within clusters is minimized.

How to Determine the Optimal K? Determining the optimal number of clusters, k, for a k-medoids algorithm is a crucial step in the clustering process. Various methods exist to identify the most suitable k value, each with its advantages and limitations. One approach involves using domain knowledge or prior experience to select a reasonable k value based on the context of the dataset and the problem at hand. However, in many cases, an objective and data-driven method is preferred. One common technique is the elbow method, which involves plotting the within-cluster sum of squares (WCSS) against different k values and selecting the k value at the "elbow" point, where the rate of decrease in WCSS slows down significantly. Another method is the silhouette score, which measures the compactness and separation of clusters for different k values. A higher silhouette score indicates better-defined clusters, and the k value corresponding to the highest silhouette score is chosen as the optimal k.

Application In order to determine the most suitable number of clusters (k) for categorizing the top 30 WTA players, I initially applied **elbow method** described in the previous paragraph. As shown in the figure 2.3, this method didn't give us a clear cutoff on which k is optimal. So, I proceeded to apply the **silhouette method**. Figure 2.4 illustrates that the maximal average silhouette width was obtained at k = 2. Therefore, the optimal number of clusters is 2.

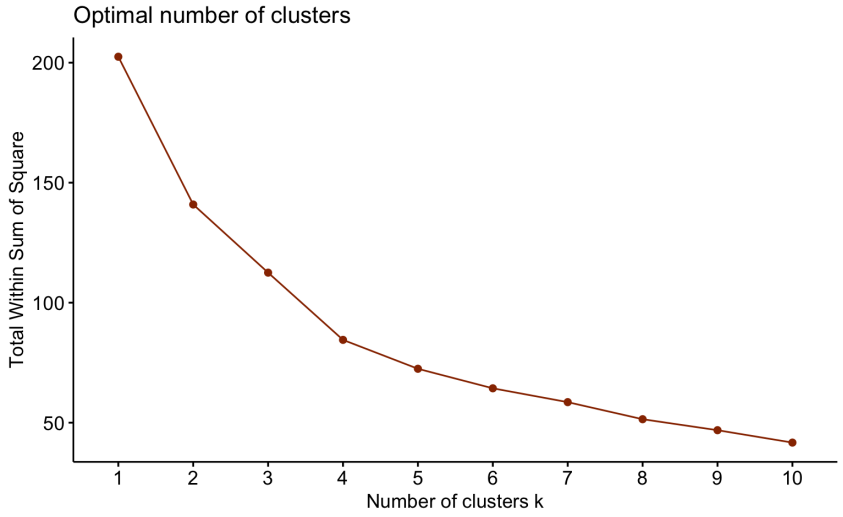

Figure 2.3: The Elbow Method

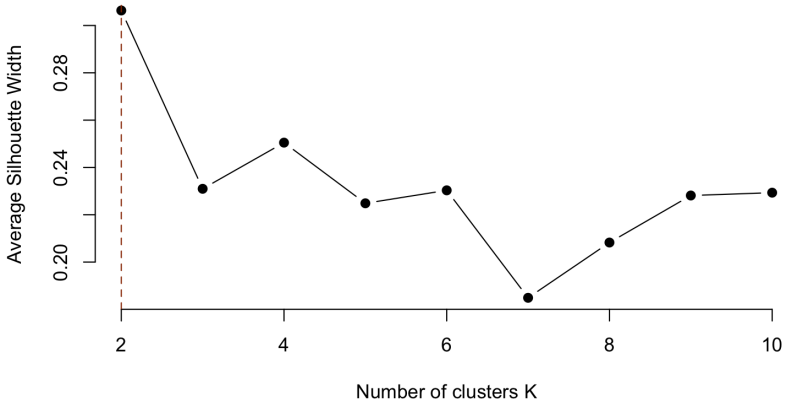

Figure 2.4: The silhouette Method

Results The results, as shown in figure 2.5 , reveal the presence of two distinct clusters. Upon closer examination, as demonstrated in Table 2.2, we can see that cluster 1 comprises over 93% of the top 10 players. This clustering aligns logically with the nature of tennis, as top players are expected to excel across various performance metrics discussed earlier.

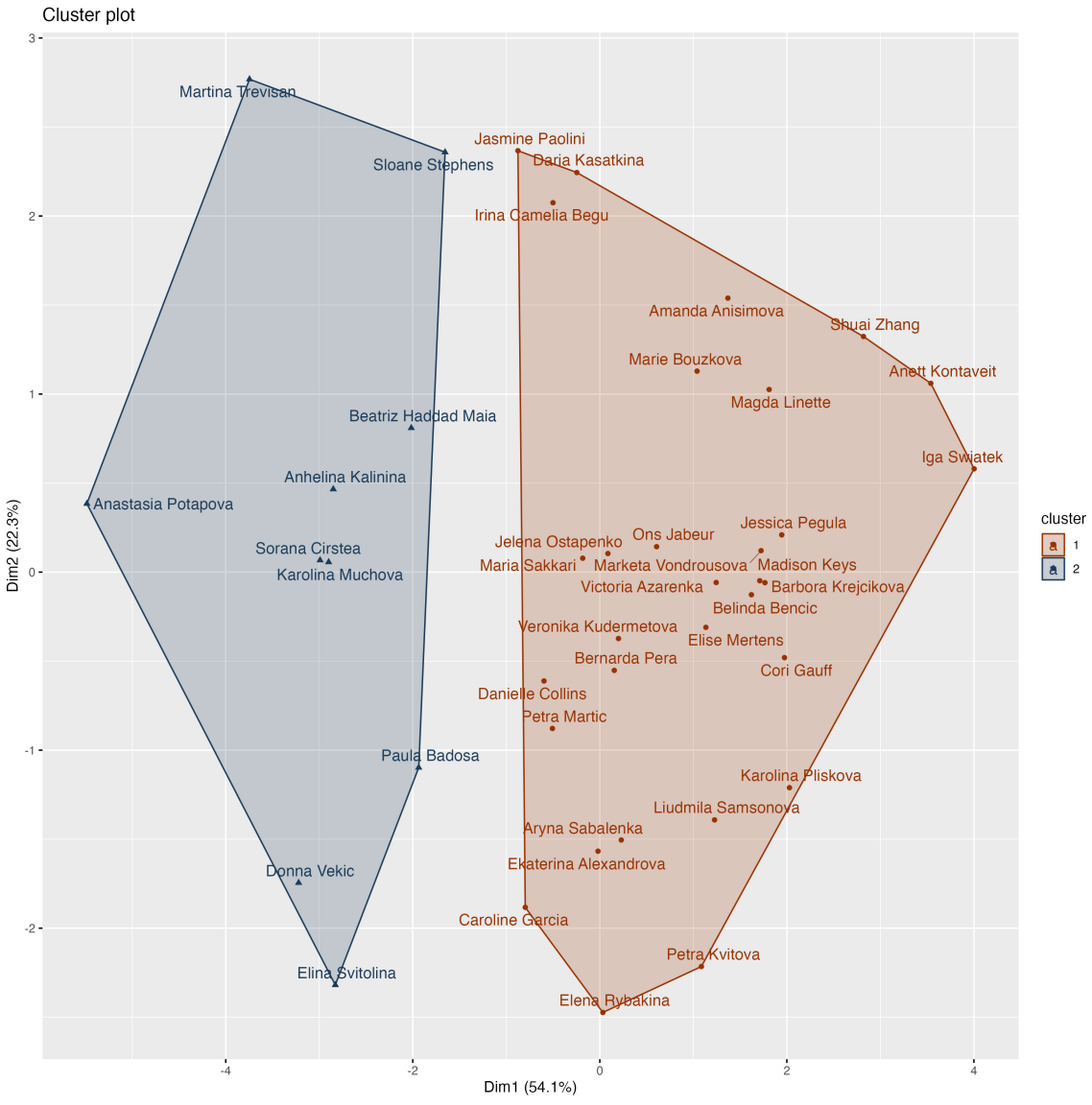

Figure 2.5: K-medoids Partitioning at k = 2

player_name	**top_10**	**cluster**
Aryna Sabalenka	1	1
Barbora Krejcikova	1	1
Belinda Bencic	1	1
Caroline Garcia	1	1
Cori Gauff	1	1
Daria Kasatkina	1	1
Elena Rybakina	1	1
Iga Swiatek	1	1
Jessica Pegula	1	1
Karolina Muchova	1	2
Maria Sakkari	1	1
Marketa Vondrousova	1	1
Ons Jabeur	1	1
Petra Kvitova	1	1
Veronika Kudermetova	1	1

Table 2.2: Clustering Results of Top 10 Players

2.2.2 Hierarchical Clustering

What is Hierarchical Clustering? Hierarchical clustering is a method used to group similar objects into clusters by iteratively merging or splitting clusters based on their similarities or dissimilarities. Unlike K-means or K-medoids, hierarchical clustering does not rely on a specific formula but rather employs proximity or dissimilarity measures between data points, such as Euclidean distance or correlation distance, to determine which clusters to merge or split at each step. One key distinction between hierarchical clustering and K-means/K-medoids is that hierarchical clustering generates a hierarchy of clusters, allowing for a more flexible approach to clustering without the need to pre-specify the number of clusters. Additionally, hierarchical clustering can handle non-convex clusters and clusters of different shapes, making it suitable for various types of data.

Application

i. Complete Linkage

Complete linkage is a method used in hierarchical clustering to measure the distance between clusters. In complete linkage, the distance between two clusters is defined as the **maximum distance** between any pair of points in the two clusters.

Mathematically, for two clusters C_k and C_l with data points x_i and x_j, the distance $d(C_k, C_l)$ is given by:

$$d(C_k, C_l) = \max_{x_i \in C_k, x_j \in C_l} d(x_i, x_j)$$

Here, $d(x_i, x_j)$ represents the distance between data points x_i and x_j, calculated according to the chosen distance metric, such as Euclidean, Manhattan, or cosine distance.

Complete linkage tends to produce clusters with more compact and spherical shapes compared to other linkage methods, as it focuses on the maximum dissimilarity between clusters. This can be advantageous in situations where compactness and separation of clusters are desired.

In this context, it would help identify clusters where players have significantly different

performance profiles. For example, it could group players who excel in certain aspects of the game while struggling in others, leading to distinctive clusters that highlight specific strengths and weaknesses.

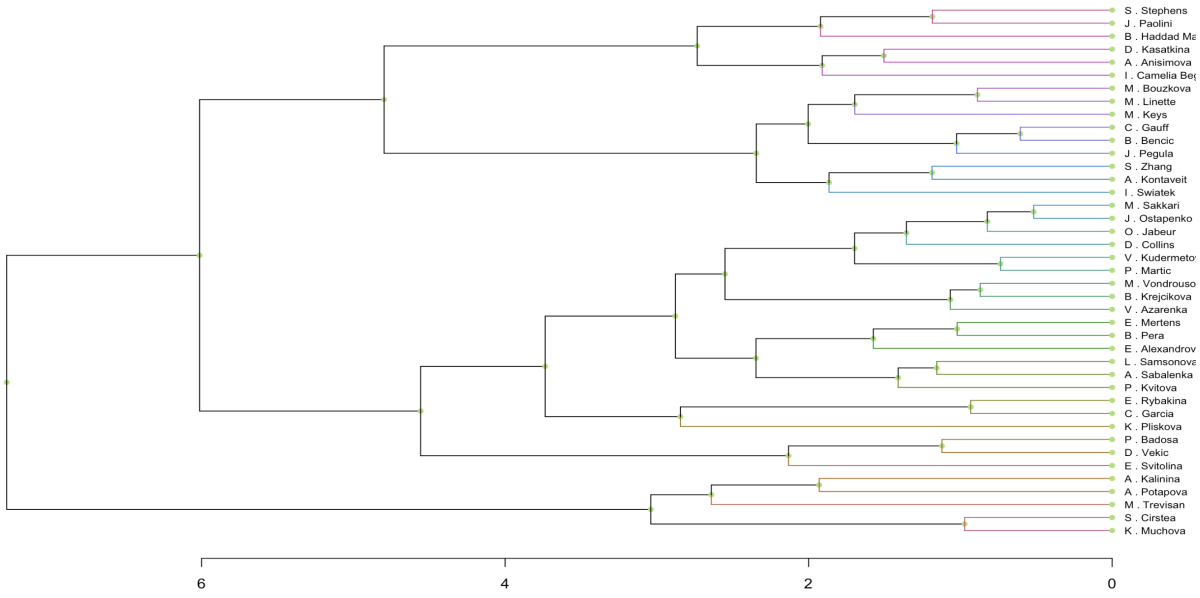

Figure 2.6: Hierarchical Clustering with Complete Method

ii. Average Linkage

Average linkage is another method employed in hierarchical clustering to determine the distance between clusters. Unlike complete linkage, which focuses on the maximum distance between any pair of points in the two clusters, average linkage calculates the distance as the **average** of all pairwise distances between points in the two clusters.

Mathematically, for two clusters C_k and C_l with data points x_i and x_j, the distance $d(C_k, C_l)$ is computed as:

$$d(C_k, C_l) = \frac{1}{|C_k| \times |C_l|} \sum_{x_i \in C_k} \sum_{x_j \in C_l} d(x_i, x_j)$$

Here, $|C_k|$ and $|C_l|$ represent the number of data points in clusters C_k and C_l, respectively, and $d(x_i, x_j)$ signifies the distance between data points x_i and x_j, using the specified distance metric.

Average linkage is known to produce more evenly sized clusters compared to complete linkage and can be advantageous when dealing with datasets where clusters have varying densities or shapes.

In this context, this method would help create more evenly sized clusters by considering the collective performance of players. It can accommodate variations in player skill levels and playing styles, resulting in clusters that represent a balanced mix of different player profiles.

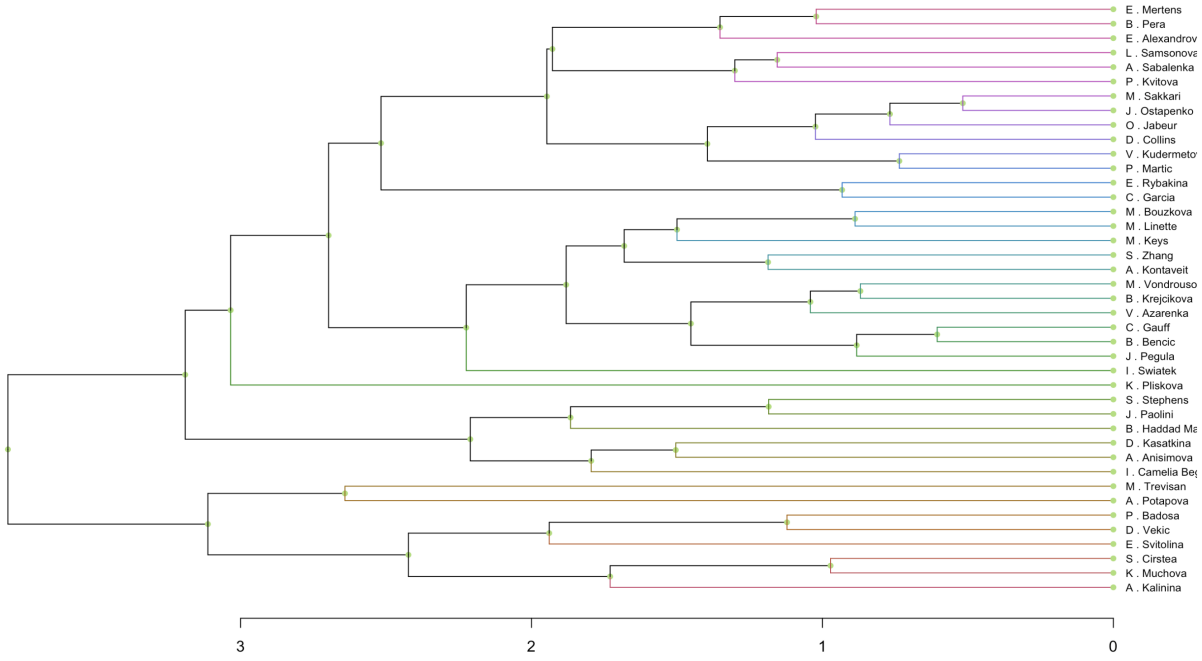

Figure 2.7: Hierarchical Clustering with Average Method

iii. Complete VS. Average

To gain a comprehensive understanding of the diversity in player performance and playing styles among the top 30 WTA players, I compared the outcomes of the complete and average linkage methods. As illustrated in Figure 2.8, while the inter-cluster distances exhibited variability, the clusters maintained consistent compositions. Notably, players like Kvitova, Samsonova, Sabalenka, and Rybakina, known for their powerful serves and

proficiency in scoring aces, consistently formed a cluster across both methods.

An additional noteworthy observation is the proximity between Iga Swiantek and Anett Kontaveit in the complete method, indicating similarities in their playing styles. However, in the average method, this proximity was diminished, potentially influenced by their respective rankings in the 2023 season—Swiantek held the top rank while Kontaveit ranked 17th, reflecting differences in overall performance.

Overall, the complete method highlighted distinctions in playing styles, while the average method moderated these differences, resulting in more balanced clusters that better represent the overall performance of players.

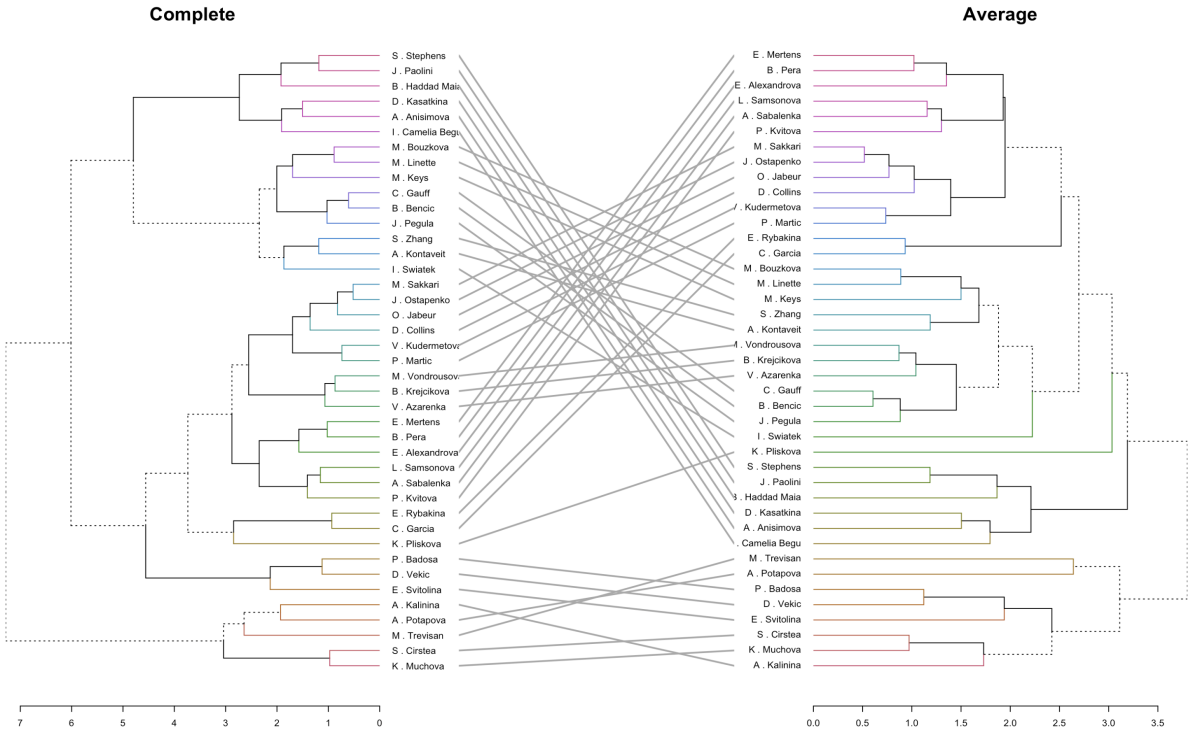

Figure 2.8: Comparison between Complete and Average Methods

Chapter 3

Supervised Learning

This chapter discusses the supervised learning techniques used to predict the outcome of a match, represented by the target variable "win" . I selected a subset of the dataset comprising player attributes, including age, height, dominant hand, and WTA rank. Additionally, I incorporated match-specific details such as match duration, grand slam indicator, final round indicator, court surface, and first-set victory. Below is a comprehensive list of the variables encompassed in the subset:

- **win**: Indicates whether the player won the match.
- **grand_slam**: Indicates if the match occurred in a grand slam tournament.
- **final_rounds**: Indicates if the match was in the final rounds of a tournament.
- **set1_p_win**: Indicates if the player won the first set of the match.
- **player_hand_R**: Indicates if the player is right-handed.
- **opponent_hand_R**: Indicates if the opponent is right-handed.
- **surface_grass**: Indicates if the playing surface is a grass court.
- **surface_clay**: Indicates if the playing surface is a clay court.
- **player_ht**: Player's height in centimeters.

- **player_age**: Player's age at the time of the match.
- **opponent_ht**: Opponent's height in centimeters.
- **opponent_age**: Opponent's age at the time of the match.
- **minutes**: Duration of the match in minutes.
- **player_rank**: Player's WTA ranking at the time of the match.
- **opponent_rank**: Opponent's WTA ranking at the time of the match.

3.1 Logistic Regression

What is Logistic Regression? Logistic regression is a statistical method used for modeling the probability of a binary outcome based on one or more predictor variables. It is commonly used when the dependent variable is categorical and has two possible outcomes, usually coded as 0 and 1.

In logistic regression, the logistic function, also known as the sigmoid function, is used to model the probability of the outcome. The logistic function is defined as:

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

where:

- $P(Y = 1|X)$ is the probability of the outcome being 1 given the predictor variable X.
- β_0 and β_1 are the coefficients of the logistic regression model.

Assumptions of Logistic Regression:

1. *Binary Outcome:* The dependent variable should be binary, meaning it can take only two possible values.

2. *Independence of Observations:* The observations in the dataset should be independent of each other.

3. *Linearity of Log Odds:* The relationship between the predictor variables and the log odds of the outcome should be linear.

4. *No Multicollinearity:* There should be little or no multicollinearity among the predictor variables.

Relevance in Predicting Tennis Match Outcomes Logistic regression is highly relevant and suitable for predicting tennis match outcomes based on previous literature. Tennis matches have a binary outcome, where a player either wins or loses the match. There are no ties. Logistic regression allows us to model the probability of a player winning a match based on various predictor variables such as player rankings, match statistics, surface type, and player characteristics. Previous studies have demonstrated the effectiveness of logistic regression in predicting tennis match outcomes, making it a valuable tool in tennis analytics and decision-making processes. Its ability to provide interpretable results and estimate probabilities makes it particularly useful for understanding the factors influencing match outcomes.

Application

Correlation Analysis To identify the key variables influencing match outcomes and assess potential multicollinearity, I conducted an exhaustive correlation analysis across all variables in the subset. This analysis, depicted in Figure 3.1, aims to uncover relationships between predictors and detect any redundancies or dependencies among them. Notably, the target variable "win" exhibits the highest correlation with "set1_p_win", followed by "player_rank" and "opponent_rank", and then by "opponent_age." Additionally, "player_age" and "player_ht" show some correlation with the target variable, albeit to a lesser extent.

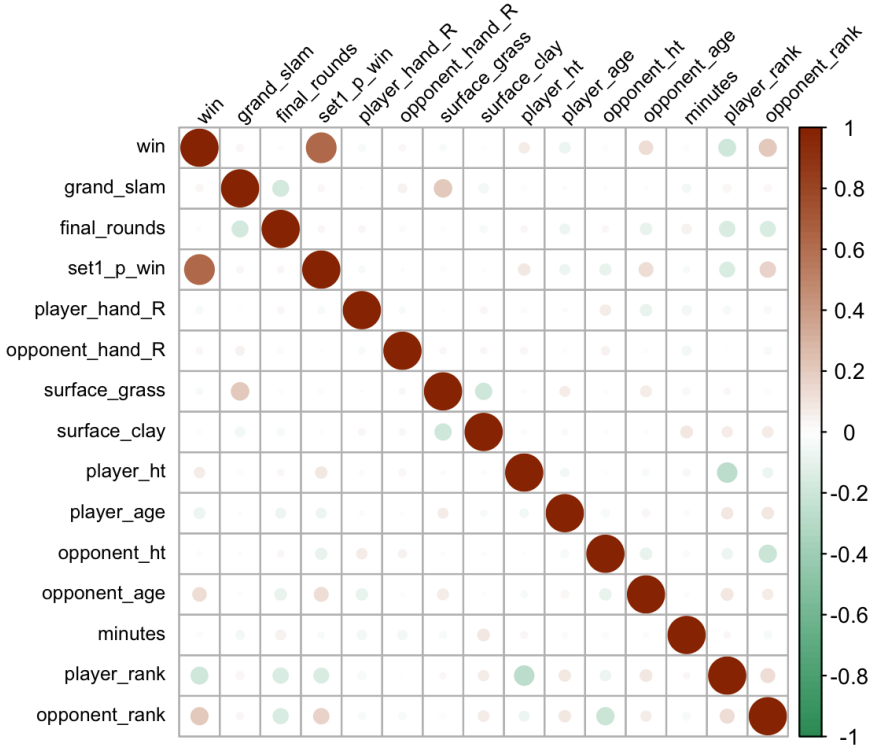

Figure 3.1: Correlation Analysis

Full Model Before starting the analysis, I partitioned the subset into an 80% training set and a 20% test set. In the initial phase, I executed the logistic regression model with all variables present in the subset. Among these, "set1_p_win", "player_rank", and "opponent_rank" emerged as statistically significant predictors, demonstrating significance at conventional levels. Notably, "opponent_ht" also exhibited statistical significance, albeit at a 10% significance level. These variables will serve as the predictors for the reduced model, elaborated upon in the subsequent section.

```
Call:
glm(formula = win ~ ., family = binomial(link = "logit"), data = scaled_train_data)

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.457728   0.415183  -3.511 0.000446 ***
grand_slam       0.201468   0.236430   0.852 0.394144
final_rounds     0.022492   0.227927   0.099 0.921391
set1_p_win       2.988244   0.190137  15.716  < 2e-16 ***
player_hand_R   -0.080270   0.305099  -0.263 0.792477
opponent_hand_R  0.100903   0.308016   0.328 0.743221
surface_grass   -0.425802   0.310442  -1.372 0.170189
surface_clay    -0.159534   0.238644  -0.669 0.503813
player_ht       -0.004203   0.093706  -0.045 0.964224
player_age      -0.123226   0.094676  -1.302 0.193066
opponent_ht      0.185792   0.096127   1.933 0.053264 .
opponent_age     0.138199   0.098005   1.410 0.158504
minutes          0.069369   0.094424   0.735 0.462549
player_rank     -0.453309   0.102160  -4.437 9.11e-06 ***
opponent_rank    0.447368   0.100410   4.455 8.37e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1176.34  on 848  degrees of freedom
Residual deviance:  747.98  on 834  degrees of freedom
AIC: 777.98

Number of Fisher Scoring iterations: 4
```

Figure 3.2: Logistic Regression - Full Model

Reduced Model Although I have already determined the predictor variables for the reduced model based on their statistical significance, I also employed the backward selection method as a secondary check on variable selection. As depicted in Figure 3.3, the results aligned with those previously identified, reaffirming the selection of the same variables.

```
Call:
glm(formula = win ~ set1_p_win + opponent_ht + player_rank +
    opponent_rank, family = binomial(link = "logit"), data = scaled_train_data)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   -1.48370    0.13338 -11.124  < 2e-16 ***
set1_p_win     3.02506    0.18787  16.102  < 2e-16 ***
opponent_ht    0.17902    0.09488   1.887   0.0592 .
player_rank   -0.44054    0.09687  -4.548 5.42e-06 ***
opponent_rank  0.42658    0.09788   4.358 1.31e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1176.34  on 848  degrees of freedom
Residual deviance:  754.71  on 844  degrees of freedom
AIC: 764.71

Number of Fisher Scoring iterations: 4
```

Figure 3.3: Logistic Regression - Reduced Model (Backward Selection)

Results To gain deeper insights into the outcomes derived from the reduced model (Figure 3.3), I exponentiated the coefficients to interpret them as odds ratios. I obtained the following results:

	(Intercept)	set1_p_win	opponent_ht	player_rank	opponent_rank
Odds Ratio	0.2267977	20.5952856	1.1960389	0.6436892	1.5320034

Table 3.1: Odds Ratios from the Reduced Model

From the output above, we derive the following interpretations:

- Players who win the 1st set have approximately 20.6 times higher odds of winning the match compared to those who lose the 1st set.
- With every 1 centimeter increase in the opponent's height, the odds of winning the match increase by a factor of approximately 1.2.
- Each unit increase in the player's ranking corresponds to a decrease in the odds of winning by a factor of approximately 0.64. (Note: The negative log odds for player ranking in the previous output is -0.44054)
- With every 1 unit increase in the opponent's ranking, the odds of winning increase by a factor of approximately 1.53.

Prediction Accuracy

Confusion Matrix From the confusion matrix (figure 3.4), the following observations can be made:

- The **overall accuracy** of the model is 0.7877, indicating that it correctly predicts 78.77% of the cases.
- The **confidence interval for the accuracy** provides a range within which the true accuracy is likely to fall. In this case, it ranges from 0.7265 to 0.8408.
- **Kappa** serves as a measure of agreement between the predicted and observed classifications. With a kappa value of 0.5696, there is moderate agreement between the model's predictions and the actual outcomes.
- **Sensitivity**, also known as the "True Positive Rate," measures the proportion of actual positives correctly identified by the model. In this context, it stands at 0.8000, indicating that the model correctly identifies 80% of the positive cases.

- **Specificity**, or the "True Negative Rate," measures the proportion of actual negatives correctly identified by the model. With a specificity of 0.7717, the model accurately identifies 77.17% of the negative cases.

- The **Balanced Accuracy** represents the average of sensitivity and specificity. Here, it is calculated as 0.7859, providing an overall measure of the model's performance.

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 96 21
         1 24 71

               Accuracy : 0.7877
                 95% CI : (0.7265, 0.8408)
    No Information Rate : 0.566
    P-Value [Acc > NIR] : 1.033e-11

                  Kappa : 0.5696

 Mcnemar's Test P-Value : 0.7656

            Sensitivity : 0.8000
            Specificity : 0.7717
         Pos Pred Value : 0.8205
         Neg Pred Value : 0.7474
             Prevalence : 0.5660
         Detection Rate : 0.4528
   Detection Prevalence : 0.5519
      Balanced Accuracy : 0.7859

       'Positive' Class : 0
```

Figure 3.4: Confusion Matrix of the Reduced Model

ROC Curve The results described above can also be confirmed through the Receiver Operating Characteristic (ROC) curve, which is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. In simpler terms, it shows the trade-off between the true positive rate (sensitivity) and the false positive rate (1 - specificity) as the discrimination threshold changes.

In the context of this model:

- The Area Under the ROC Curve (AUC) is a widely used metric to quantify the performance of a classification model. It represents the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

- An AUC of 1 indicates a perfect classifier, while an AUC of 0.5 suggests a classifier that performs no better than random chance.

In Figure 3.5, the AUC is calculated as 0.846. This means that there is an 84.6% chance that the model will be able to distinguish between positive and negative outcomes. In other words, the model has a good discriminatory ability, with a higher probability of correctly ranking a randomly chosen positive instance higher than a randomly chosen negative instance.

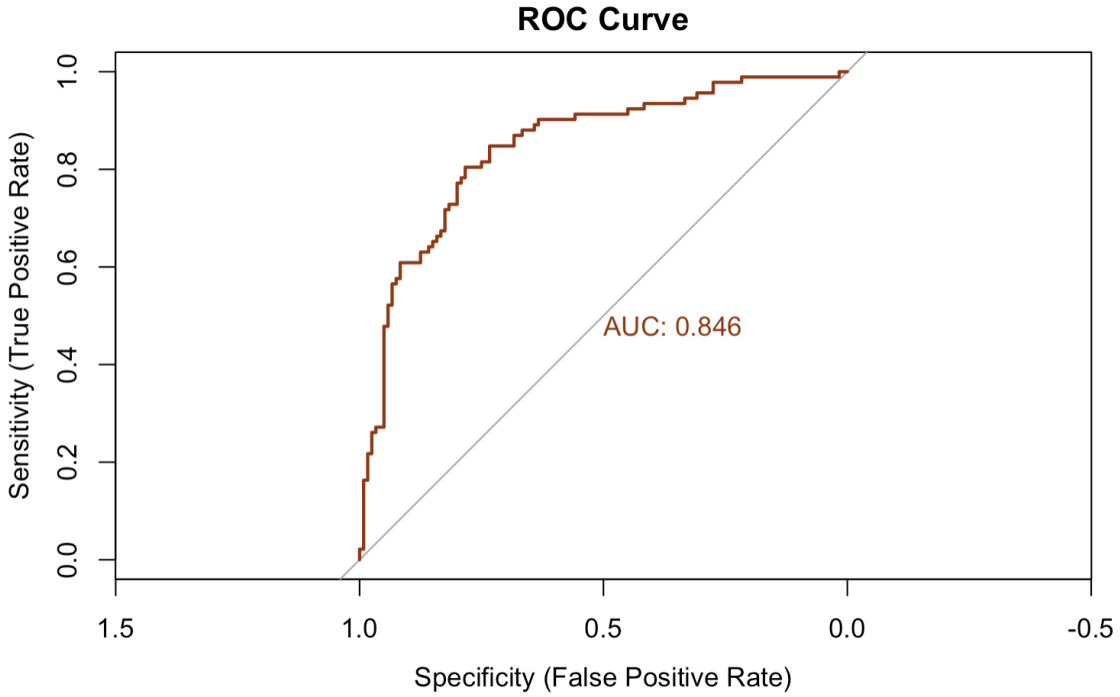

Figure 3.5: ROC Curve of the Reduced Model

3.2 Classification Tree

What is a Classification Tree? A classification tree is a type of decision tree used to predict a qualitative response, such as a categorical outcome, rather than a quantitative one. In a classification tree, each observation is predicted to belong to the most commonly occurring class of training observations within the region to which it belongs.

Similar to regression trees, classification trees use recursive binary splitting to grow the tree. However, in the classification setting, the residual sum of squares (RSS) cannot be used as a criterion for making binary splits. Instead, a common alternative is the classification error rate, which is the fraction of training observations in a region that do not belong to the most common class. However, in practice, two other measures are often preferred: the Gini index and the deviance.

The Gini index, defined as $G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$, where $\hat{p}_{mk}$ is the proportion of training observations in the mth region that are from the kth class, measures the total variance across the K classes. A small value of the Gini index indicates that a node contains predominantly observations from a single class, making it a measure of node purity. An alternative measure is cross-entropy, given by $D = -\sum_{k=1}^{K} \hat{p}_{mk} \log(\hat{p}_{mk})$. Both the Gini index and cross-entropy are numerically similar and are commonly used in practice.

Algorithm Steps The algorithm for constructing a classification tree can be simplified into the following steps:

1. Start with the root node containing all the training data.
2. For each feature, calculate impurity measures (e.g., Gini index or cross-entropy) for possible split points.
3. Select the feature and split point that minimize impurity.
4. Create child nodes by splitting the data based on the selected feature and split point.

5. Repeat steps 2-4 recursively for each child node until a stopping criterion is met (e.g., maximum tree depth or minimum node size).

These steps result in a binary tree where each internal node represents a decision based on a feature and each leaf node represents a class label.

Relevance in Predicting Tennis Match Outcomes Classification trees are highly relevant in predicting tennis match outcomes due to their ability to handle qualitative responses. By considering various features such as player rankings, match statistics, playing surface, and player characteristics, classification trees can effectively partition the feature space and identify decision rules that lead to accurate predictions. The interpretability of classification trees also makes them valuable in understanding the factors influencing match outcomes and guiding decision-making processes in tennis analytics.

Application After applying the classification tree model to the subset mentioned in the introductory section 3, I obtained this tree (figure 3.6) that can be described as follows:

- At the root node, we start with all 849 training observations.
- If the probability of winning the first set (`set1_p_win`) is less than 0.5, the majority class is predicted as 0 (loss).
- If the probability of winning the first set is greater than or equal to 0.5:
  - If the match duration (`minutes`) is greater than or equal to 116 minutes:
    - If the player's ranking (`player_rank`) is greater than or equal to 53, the majority class is predicted as 0 (loss).
    - If the player's ranking is less than 53, the majority class is predicted as 1 (win).
  - If the match duration is less than 116 minutes, the majority class is predicted as 1 (win).

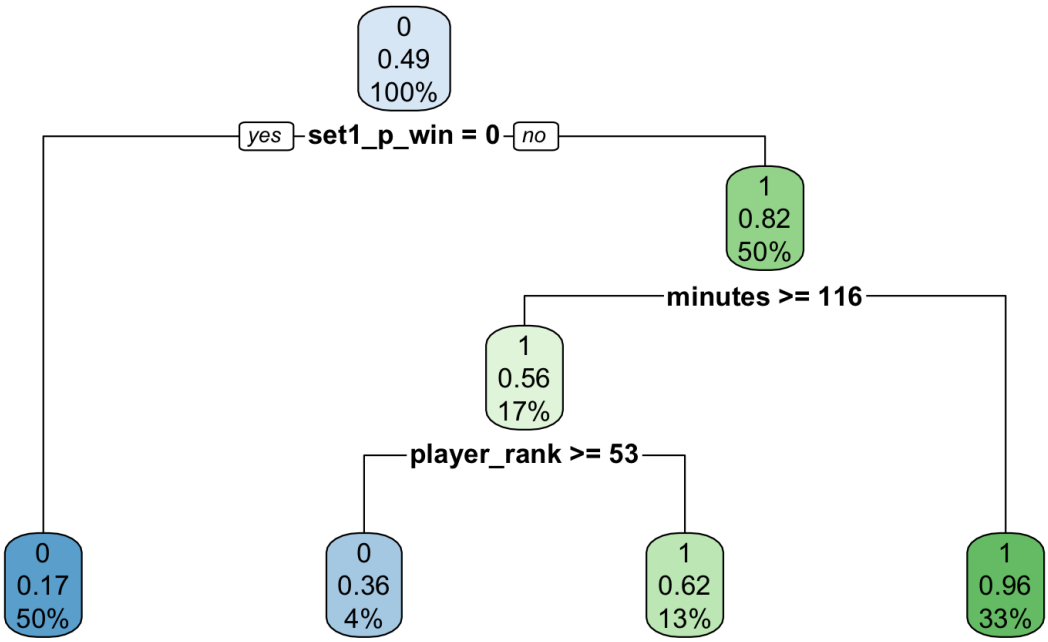

Figure 3.6: Classification Tree

Results The results of this classification tree are highly intuitive within the context of this project, and in tennis matches in general. The decision to predict a loss when the probability of winning the first set is less than 0.5 aligns with the common understanding that losing the first set often leads to a match loss. Conversely, when the probability of winning the first set is greater than or equal to 0.5, the model considers additional factors. If the match duration exceeds 116 minutes, which is more or less the average duration of WTA matches, indicating a potentially lengthy and competitive match, the player's ranking becomes pivotal. A higher ranked player (player_rank less than 53) tends to signify a more experienced or higher-seeded player, thus leading to a prediction of a win. Conversely, a lower ranked player suggests a less experienced or lower-seeded player, leading to a prediction of a loss. When the match duration is shorter than 116 minutes, indicating a potentially shorter and possibly one-sided match, the model predicts a win, reflecting the likelihood of the player with the advantage winning the match swiftly.

Prediction Accuracy The table 3.2 provides insights into the performance of a classification tree model through a confusion matrix and accuracy assessment. In the matrix, the rows signify the predicted outcomes, while the columns represent the actual outcomes. From the matrix, we observe that the model correctly predicts 86 instances as "Win" (True Positives), but misclassifies 29 instances as "Win" when they are actually "Loss" (False Positives). Additionally, 18 instances are erroneously predicted as "Loss" when they are "Win" (False Negatives), while 79 instances are correctly classified as "Loss" (True Negatives). Calculating the accuracy, which is the proportion of correct predictions out of the total instances, yields a value of 0.7783019, indicating that the model achieves an accuracy rate of approximately 77.83%.

	Actual Win	Actual Loss
Predicted Win	86	29
Predicted Loss	18	79
Accuracy	0.7783019	

Table 3.2: Confusion Matrix and Accuracy - Classification Tree

3.3 Random Forest

What is Random Forest? Random Forest is a powerful ensemble learning method that operates by constructing a multitude of decision trees during training and outputs the mode of the classes for classification problems or the mean prediction for regression problems. It combines the concept of bagging (bootstrap aggregating) and random feature selection to build a robust and accurate predictive model.

Random Forest creates a number of decision trees, each trained on a bootstrap sample of the data. During each split of the tree, a random subset of features is considered, ensuring diversity among the trees.

Relevance in Predicting Tennis Match Outcomes Random Forest is highly relevant in predicting tennis match outcomes due to its robustness in handling complex relationships between predictor variables and outcomes. By leveraging multiple decision trees and random feature selection, Random Forest effectively captures the nuances of tennis match dynamics, including player performance, match statistics, playing surface, and other influential factors. Moreover, its ability to handle large datasets and mitigate overfitting makes it suitable for analyzing vast amounts of tennis match data and generating accurate predictions.

Application After applying the random forest model to the subset mentioned in the introductory section 3, I obtained the results described in the table down below:

	Actual Win	Actual Loss
Predicted Win	81	22
Predicted Loss	23	86
Accuracy	0.7877358	

Table 3.3: Confusion Matrix and Accuracy - Random Forest

Results The results presented in Table 3.3 display the performance metrics of a Random Forest model in predicting tennis match outcomes. The confusion matrix illustrates the distribution of predicted outcomes compared to the actual outcomes. Specifically, out of 103 matches predicted as wins, 81 were accurately classified, while 22 were incorrectly classified as losses. Similarly, out of 109 matches predicted as losses, 86 were accurately classified, while 23 were incorrectly classified as wins. The overall accuracy of the Random Forest model is calculated to be approximately 78.77%.

Chapter 4

Challenges & Limitations

4.1 Unsupervised Learning:

- **Granularity:** One challenge lies in refining the granularity of clustering analysis to capture the diverse performance profiles of players. This includes considering factors such as player performance on different surfaces (e.g., grass, clay), in various tournament rounds (e.g., final rounds), and across different tournament types (e.g., grand slam tournaments). By incorporating these nuances, we can ensure that the clustering analysis accurately identifies meaningful player segments.

- **Temporal Relevance:** Another challenge is addressing the temporal relevance of historical player statistics in the clustering process. Traditional clustering methods may overlook the dynamic nature of player performance over time, leading to potentially outdated or less relevant cluster assignments. Implementing techniques to incorporate temporal trends and prioritize recent performance data could enhance the accuracy and relevance of the clustering results.

4.2 Supervised Learning:

- **Temporal Discounting:** In supervised learning for predicting match outcomes, a significant challenge is mitigating temporal discounting, where historical match data may become less relevant over time. Techniques such as time-weighted feature averaging and incorporating time decay functions can help address this challenge and ensure that the predictive models remain relevant and accurate over time.

- **Feature Relevance and Selection:** Another challenge lies in identifying and selecting relevant features for predicting match outcomes. While the provided features offer valuable insights into player characteristics and match conditions, there may be additional factors that significantly influence match outcomes (e.g., player form, recent performance trends, injuries). Exploring domain-specific knowledge could help identify and incorporate these additional features into the predictive models, thereby improving their predictive power and robustness.

By addressing these challenges in both unsupervised and supervised learning contexts, we can enhance the accuracy, relevance, and predictive power of the models, ultimately providing more valuable insights into player performance and match outcomes in the world of tennis.