**ATP Match Predictions Model Documentation**

**Overview:**

The purpose of this model is to predict outcomes (winner versus loser) of matches played by tennis professionals on the American Tennis Professional (ATP) tour in the years 2018 through 2022, and on an unseen test set consisting of 2023 matches.

**Data:**

Match data and player data were collected from January 2018 to December 2022. Data was sourced from Jeff Sackman, who collects comprehensive match data and provides it for free [on his website](https://www.tennisabstract.com/) and [GitHub](https://github.com/JeffSackmann).

There are 12,815 observations and 49 columns, including match data such as match location, surface, year, score, points won, seed, ranking, minutes played, etc.

**Exploratory Data Analysis Key Findings**:

An exploratory data analysis (EDA) was performed.

A correlation map was constructed with a select number of variables. Some correlations make sense, such as the best\_of variable (the number of sets required to be played) and the draw size. Larger draw tournaments are best of 5 sets format, while smaller tournaments are best of 3. Winner serve points and loser serve points are also highly correlated, which make sense given that the winner of a match will have more points won on serve than the loser.

**A graph with blue dots and red text

Description automatically generated**

The top twenty players who won the most matches in the last five years were isolated. As expected, this chart contains players that are consistently in the top 10-20 ranking. Novak Djokovic, arguably the best player of all time, had the most match wins in the last five years.

A graph of a number of people

Description automatically generated

In addition to the raw number of match wins, win percentage is another key statistic. Some players don’t enter many tournaments throughout the year, but are highly ranked because they win nearly all of the matches they play. This chart provides more helpful information. The “big three” – Novak Djokovic, Rafael Nadal, and Roger Federer have had the highest win percentages in the last five years. This makes complete sense as they have dominated the sport for nearly two decades.

A graph of red bars

Description automatically generated

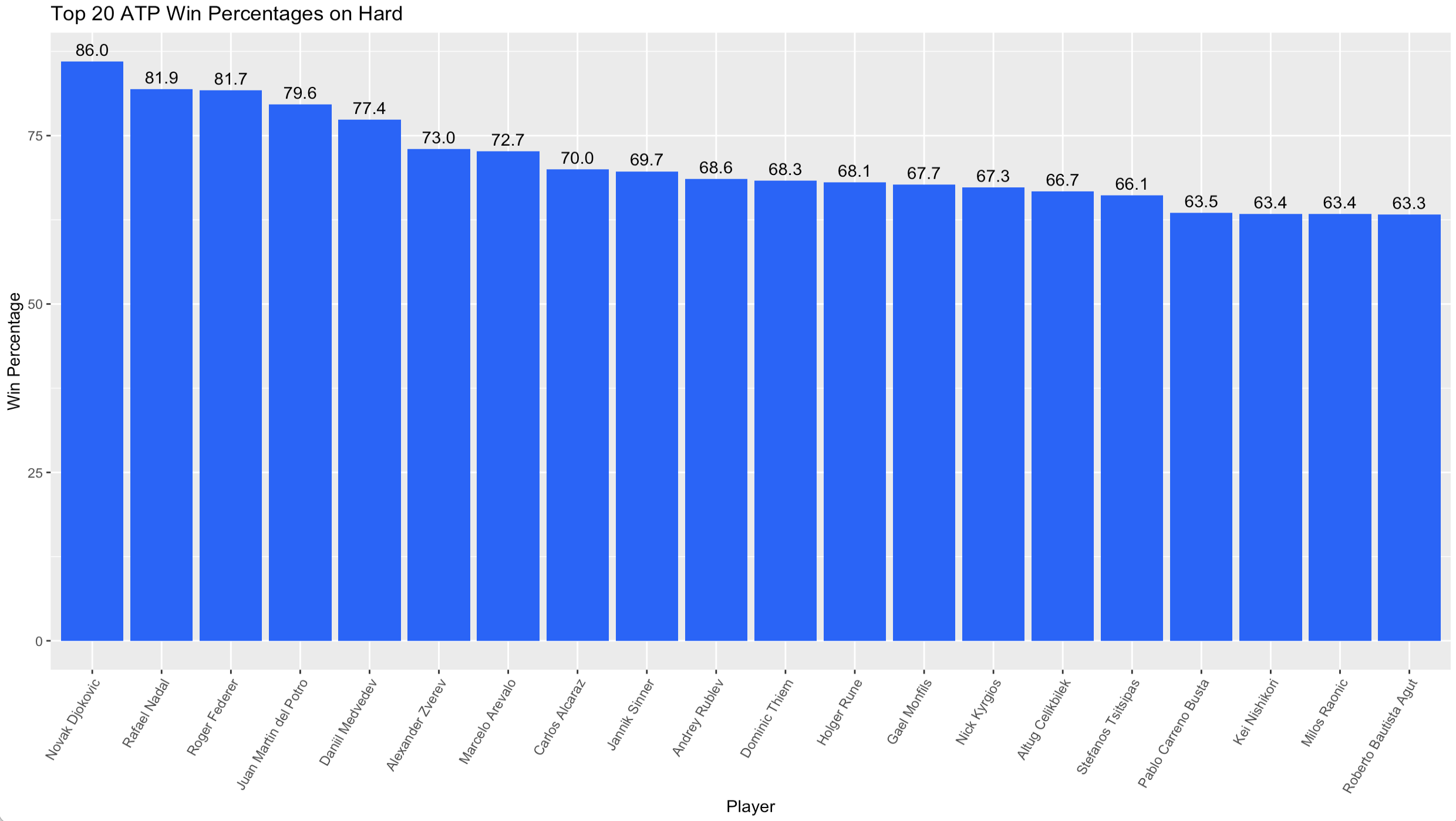
Tennis is unique in that it is played on multiple surfaces: hard court, grass, and clay. Certain players thrive on certain surfaces. What about win percentages according to surface?

**A graph of green bars

Description automatically generated**

**A graph of orange bars

Description automatically generated**

****

Player height is a well known factor for being successful in tennis, as tall players can have strong serves and win points off of their serves easily. What is the mean height in the ATP? It seems to be about 185cm, which is about 6’1’’. It’s not entirely normally distributed, as most players hover around 175cm and 195cm in height. The strong drop off is because short players are very disadvantaged by not having strong serves and pace, while very tall players are disadvantaged in that they have strong serves but are limited in on court agility and movement due to their size.

A graph of a height

Description automatically generated with medium confidence

**Preprocessing Steps:**

*Seed*

All seeds during preprocessing and modeling were set to 42.

*Removal of Certain Matches*

Matches from the NextGen finals were removed for a few reasons. These are exhibition matches, so ranking points are not awarded. Scoring is also different with sets being first to 4 rather than the traditional first to 6. The tournament is also restricted to players under 20 years of age, and the ATP frequently tests new match rules at this tournament, making it unlike other ATP matches.

Matches from the Laver Cup were also removed for several reasons. The players participating are selected by Roger Federer and the Laver Cup administration team, rather than based on ranking. The scoring also uses a ten-point tiebreaker instead of a third set, making it unlike other ATP matches. It is also an exhibition event; although it counts toward formal head-to-heads, players may behave differently in an exhibition event, especially when dead rubber matches are played.

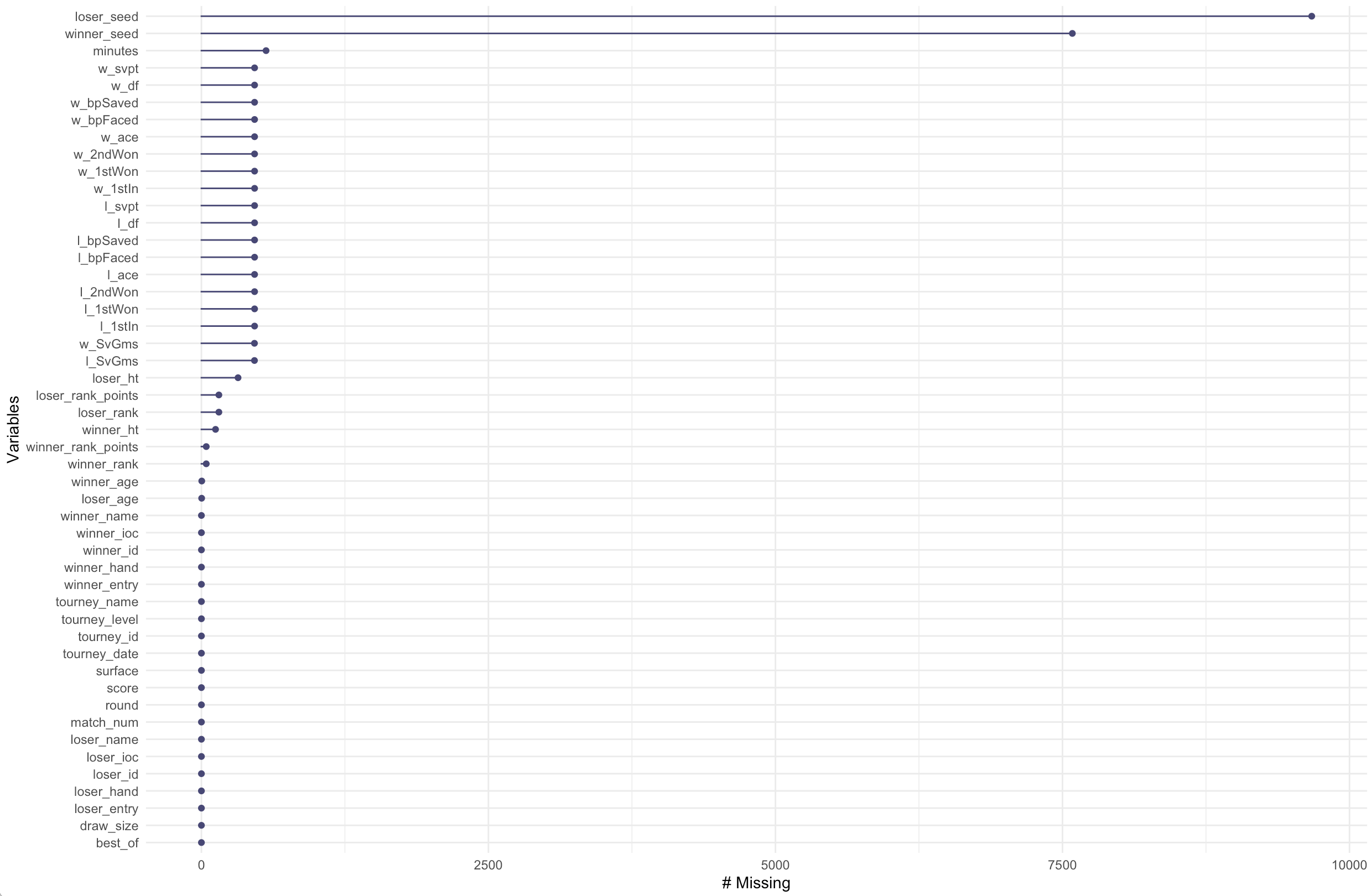
*Missing Data*

Data was inspected for missingness.

The winner seed and loser seed were the most common missing values because not everyone is seeded during a tournament (in the US Open, for example, there are 32 seeds out of a 128-player draw). Missing seed values were imputated with zero.

A close-up of a graph

Description automatically generated



Player heights were also missing for many players. Heights were searched on Google and imputated with the correct value, usually sourced from the ATP website.

The winner\_entry and loser\_entry columns are not helpful and were dropped from the dataset due to high percentage of missing values.

After imputation and dropping unhelpful columns, the data was filtered for complete cases. This resulted in removal of 362 incomplete cases. The dataset now contains 11,998 observations and 48 columns.

*Feature Extraction*

Granular score data was extracted from the variable score\_full, which reports the match score in its entirety. Variables were created for the score of each set as well as set tiebreaks. A variable indicating whether the match resulted in a retirement or walkover was also created.

Dummy variables were created indicating whether the winner of the match won each set in the match.

The following features were created for both the winner and loser of the match (creation is shown for winner only to avoid redundancy). Many of these features were derived from [Ultimate Tennis Statistics](https://www.ultimatetennisstatistics.com/glossary).

* **Elo rating system** – calculating relative skill levels of players in competitor-vs-competitor games
  + Elo can give us insight into a player's performance over time and relative to others
  + The only input necessary to make a prediction is the difference between two players’ ratings.
  + The formula is as follows:
    - 1 – (1 / (1 + (10^((difference) / 400))))
  + Example: If we wanted to forecast a rematch of the last match of the Davis Cup Finals, we would take the Elo ratings of Nadal and Denis Shapovalov (2203 and 1947), find the difference (256), and plug it into the formula, for a result of 81.4%, Nadal’s chance of winning. If we used the negative difference (-256), we’d get 18.6%, Shapovalov’s odds of scoring the upset.
  + [Code from this github repo was used to calculate the Elo rating function.](https://github.com/sleepomeno/tennis_atp/blob/master/examples/elo.R)
  + Odds for best of 3 and best of 5 matches were calculated.
* **First Serve Percentage (w\_1st\_made)** – total amount of first serves made successfully
  + w\_1stin/w\_svpt
  + First serves in / total serve points
* **Second Serve Percentage (w\_2nd\_made)** – total amount of second serves made successfully
  + First, calculate number of second serves in
  + w\_2ndIn = w\_svpt – w\_1stin – w\_df
  + Total serve points – first serves made – double faults
  + Next calculate second serve percentage
  + Second serves made / total serve points for second serve
  + w\_2ndIn/(w\_svpt – w\_1stIn)
* **First Serve Points Won Percentage** – the percentage of first service points won
  + w\_1st\_serve\_perc\_win = w\_1stWon/w\_svpt
  + This is a commons statistic in tennis and indicates the percentage of the points won on the first serve.
  + Generally, someone should be winning points on their first serve because it is stronger than the second serve (and players should hold their service games).
* **Second Serve Points Won Percentage** – the percentage of second service points won
  + Assuming a second serve doesn't count as an additional service point
  + w\_2nd\_serve\_perc\_win = w\_ 2ndWon/(w\_svpt - w\_1stIn)
  + Second serve points won/(total service points – first serves made)
* **First Serve Rating** – percentage of first serves made/win percentage on first serves
  + <https://www.atptour.com/en/news/infosys-atp-insights-first-serve-rating>
  + This is a new metric created by the ATP
  + >60 is great, <30 is horrible
  + w\_1st\_made\*100 / w\_1st\_serve\_perc\_win
* **Second Serve Rating** – percentage of second serves made/win percentage on second serves
  + w\_2nd\_made\*100 / w\_2nd\_serve\_perc\_win
* **First Serve Effectiveness** - first serve points won % divided by second serve points won %
  + w\_1st\_serve\_perc\_win/w\_2nd\_serve\_perc\_win
* **Win % on Return of Serve** – the percentage of points won when returning serve
  + For the winner: (Loser first serves in – loser first serves won) + (loser total service points – loser first serves in) – (loser second serves won – loser double faults) / loser total service points
  + The first expression is points won from the other player’s first serve
  + The second expression is the number of second serve attempts
  + The third expression is the other player’s number of second serve points won MINUS double faults
  + (l\_1stIn - l\_1stWon + l\_svpt - l\_1stIn - l\_2ndWon - l\_df)/l\_svpt
* **Points Dominance Ratio** - % of return points won divided by % of service points lost
  + Percentage of return points won is calculated by subtracting percentage of serve points won from 1
  + Calculate percentage of serve points won: w\_1stWon+ w\_2ndWon / w\_svpt
  + Subtract this value from 1 to get w\_returnwon\_perc\_total
  + w\_returnwon\_perc\_total/ l\_returnwon\_perc\_total
* **Win Percentage on Break Point** – break points saved divided by break points faced
  + w\_bpSaved / w\_bpFaced
* **Win Percentage on Break Point** – break points saved divided by break points faced
  + w\_bpSaved / w\_bpFaced
* **Break Point Converted Percentage** – successful number of break points converted (won) against opponent
  + For winner: (l\_bpFaced – l\_bpSaved)/l\_bpFaced
* **Break Point Ratio** - % of break points converted (won) divided by % of faced break points lost
  + For winner: w\_bp\_convert\_perc/l\_bp\_convert\_perc
* **Points to Sets Over-Performing Ratio** - % of sets won divided by % of total points won
  + First calculate percent of sets won: w\_set\_tot/(w\_set\_tot + l\_set\_tot)
  + Second, calculate percent of total points won w\_ptswon\_perc: (w\_1stWon + w\_2ndWon + l\_1stIn - l\_2ndWon + (l\_svpt - l\_1stIn) - l\_2ndWon)/( w\_svpt + l\_svpt)
  + Then, divide sets won by points won: w\_setwon\_perc/w\_ptswon\_perc
* **Games to Sets Over-Performing Ratio** – percent of sets won divided by percent of games won
  + First, separate the score into games per player
  + Second, add up the number of games per set won by each player
  + Third, calculate the percentage of games won over total number of games played
  + Finally, calculate the ratio: w\_setwon\_perc/w\_gameswon\_perc
* **Points to Games Over-Performing Ratio** - % of games won divided by % of total points won
  + Usually previously created features, calculated as the following: w\_gameswon\_perc/ w\_ptswon\_perc
* **Break Points Over-Performing Ratio** - % of break points won (saved + converted) divided by % of total points won
  + Calculate break points won percentage (l\_bpFaced - l\_bpSaved + w\_bpSaved)/(w\_bpFaced + l\_bpFaced)
  + w\_win\_bp\_perc / w\_ptswon\_perc
* **Break Points Saved Over-Performing Ratio** - % of break points saved divided by % of service points won
  + Using previously calculated features
  + w\_bp\_saved\_perc / w\_servewon\_perc\_total
* **Break Points Converted Over-Performing Ratio** - % of break points converted divided by % of return points won
  + w\_bp\_convert\_perc / w\_returnwon\_perc\_total
* **Ace Rate** – number of aces over total number of service points
  + w\_ace/w\_svpt
* **Double Fault Rate** – number of DFs over total number of service points
  + w\_df/w\_svpt
* **Rank difference between two players**
  + winner\_rank – loser\_rank
* **Upsets scored** - Matches won over higher-ranked players (according to ATP ranking)
  + A dummy variable with 1 indicating if the higher ranked player lost
* **Upsets against** - Matches lost from lower-ranked players (according to ATP ranking)
  + A dummy variable with 1 indicating if the lower ranked player lost

Additional feature extraction was performed by calculating rolling averages of the above statistics. This provides more information into a player’s performance over time. Furthermore, it is better input for a model. Features that were not rolling averages were then removed. Categorical data was then dummified to make all features numerical.

The data was then scaled to a mean of zero and standard deviation of one.

*Feature Selection*

Random forest and principal component analysis were performed separately on the data for feature selection. The top twenty features from each analysis were chosen for preliminary model testing.

Below is a plot of the random forest feature importance mean decrease accuracy and mean decrease gini values for the top 20 variables. These top 20 variables were selected as inputs for preliminary and final modeling.

A graph with a number of text

Description automatically generated with medium confidence

PCA was also performed using the prcomp function, which performs single value decomposition. Below are scree plots for the dataset. A scree plot displays the eigenvalues on the y-axis and the number of components on the x-axis. The eigenvalues represent the amount of variance captured by each factor or component. The point where the curve starts to flatten out is known as the "elbow." It indicates the optimal number of factors or components to keep. After this point, adding more factors contributes less to explaining the variance in the data. In the scree plot below, there is an elbow around 10 factors.

The cumulative scree plot shows the cumulative percentage of variance explained by each successive factor or component on the y-axis. The x-axis still represents the number of factors or components, and the plot indicates the cumulative variance explained with the addition of each principal component.

A graph of a graph

Description automatically generated

A graph with a dotted line

Description automatically generated

*Preliminary Modeling*

The following preliminary models were tested, with their test set area under the curve reported in the table below.

|  |  |  |
| --- | --- | --- |
| Feature Selection | Model | Validation Set AUC |
| Random forest | Logistic regression | 0.643 |
| Random forest | Support vector machine (linear kernel) | 0.643 |
| Random forest | Support vector machine (radial kernel) | 0.828 |
| Random forest | XGBoost | 0.822 |
| PCA | Logistic regression | 0 .679 |
| PCA | Support vector machine (linear kernel) | 0.53 |
| PCA | Support vector machine (radial kernel) | 0.809 |
| PCA | XGBoost | 0.731 |
| PCA | Random forest | 0.783 |
| None | Shallow neural network | 0.658 |

Based on the results of these preliminary models, random forest with SVM radial kernel and random forest with XGBoost were chosen for final model testing with hyperparameter tuning in MLflow.

**Modeling:**

The MLflow server displays the following.

1. Model parameters and hyperparameters
2. Metrics: f1 score, TN, TP, FN, FP, precision (winner), precision (loser), recall (winner), recall (loser), AUC, test set accuracy, train set accuracy
3. Model artifacts: the scripts used to generate the model, the model as an RDS file, a confusion matrix, a csv of the prediction distributions, a receiving operator characteristics (ROC) area under the curve (AUC) graph, a csv of the test predictions, and a csv of the training set distributions.
   1. Test prediction csv – listing the outcome label, the predicted label, the probability of the prediction, and the id for identification
   2. ROC AUC graph – calculates the trade off between sensitivity and specificity; a high AUC is desired as it indicates sensitivity and specificity are both maximized
   3. Confusion matrix – illustrates the number of correctly and incorrectly classified matches

*Model Types*

Two model types were tested.

1. Random Forest + Support Vector Machine (Radial Kernel)
2. Random Forest + XGBoost

Both models were run with a subset of the data: twenty of the top features selected via random forest.

*Train Test Split*

The data was split into 75% training and 25% validation. The stratify argument is set to “label” to ensure class balance in the training and validation datasets.

*Hyperparameters*

MLFlow was used to test various hyperparameters. The hyperparameters to tune and their respective range of tested values were chosen based on expert opinion. The number of hyperparameters to tune and their ranges were also chosen based on what could feasibly be tested in a reasonable amount of time. Hyperparameters that were not tuned were left as the default values.

The following hyperparameters and ranges were chosen for the selected models:

1. RF + SVM (Radial Kernel)
   1. C = 10, 100, 1000, 2000, 3000, 4000, 5000
   2. Gamma = 0.1, 0.01, 0.001, 0.0001
   3. Seed = 42
2. RF + XGBoost
   1. Eta = 0.01, 0.05, 0.1, 0.2, 0.3
   2. Max depth = 5, 10, 15, 20
   3. N rounds = 20, 30, 40, 50
   4. Booster = ‘gbtree’
   5. Objective = ‘binary:logistic’
   6. Seed = 42

**Performance Metrics:**

The following metrics were collected: F1-score, AUC, true positives, true negatives, false positives, false negatives, precision, recall, accuracy, and Brier loss.

F1-score and AUC were used to select the best-performing models. Our goal was also to minimize false negatives as identification of master frame class is imperative to our objectives.

**Results:**

The following results represent the optimal model, its hyperparameters, and its metrics for each run. F1 score, AUC, and test accuracy are reported.

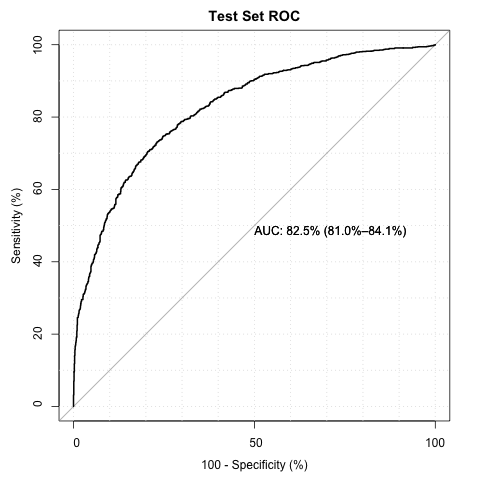
It should be noted that the top-performing model for XGBoost may not be the optimal model. This is because during hyperparameter tuning and testing, overfitting occurred with an increasing number of n\_rounds. The best-performing model had an n\_rounds value of 60, the highest among those tested. To avoid possible overfitting, the best performing model with the lowest n\_rounds value was also chosen (n\_rounds=20).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model Type | Hyperparameters | F1 | AUC | Test Accuracy |
| Random Forest + SVM Radial Kernel | C=10  gamma=0.1 | 0.738 | 0.825 | 0.75 |
| Random Forest + XGBoost | Eta=0.3  max\_depth = 15  n\_rounds=20 | 0.781 | 0.882 | 0.797 |
| Random Forest + XGBoost | Eta=0.2  max\_depth = 10  n\_rounds=60 | 0.817 | 0.907 | 0.828 |

Below is the confusion matrix and ROC curve for the RF + SVM Radial Kernel model.

A red and blue squares with black text

Description automatically generated



**Unseen Test Set Performance:**

An unseen test set containing matches from the year 2023 was obtained to test the top models. This was used on the top three chosen models to evaluate their performance.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model Type | TP | TN | FP | FN | F1 | Recall (1) | Precision (1) | Accuracy | AUC |
| RF + SVM Radial Kernel | 668 | 897 | 333 | 393 | 0.648 | 0.630 | 0.667 | 0.683 | 0.750 |
| RF + XGBoost (Top) | 737 | 941 | 289 | 324 | 0.706 | 0.694 | 0.718 | 0.732 | 0.805 |
| RF + XGBoost (Optimal) | 732 | 947 | 283 | 329 | 0.705 | 0.690 | 0.721 | 0.732 | 0.800 |

Below are the confusion matrix and ROC curve on the 2023 test set for the RF + SVM Radial kernel.

A screenshot of a graph

Description automatically generated



Below are the confusion matrix and ROC curve on the 2023 test set for the top performing RF + XGBoost.

A red and blue squares with black text

Description automatically generated

A graph showing a curve

Description automatically generated

Below are the confusion matrix and ROC curve on the 2023 test set for the optimal performing RF + XGBoost.

A screenshot of a graph

Description automatically generated

**A graph of a curve

Description automatically generated**

**Final Model Selection:**

**Model Limitations:**

Data drift is a concern regarding this model. Data drift leads to a reduction in the predictive power of the model over time. Newly collected data may change or shift, resulting in poorer model results and out-of-date model. An attempt to address this was made with the calculation of rolling averages and using recent data. There can be several causes of data drift, the most common being the change in data over time. This is likely in sports, during which new, strong-performing athletes emerge and older athletes age and retire from the sport with declining performance. Models may require re-training with new data, or more complex dynamic models such as neural networks may be employed.

**Model deployment:**

Model deployment is pending.

**Authors:**

Ascha, Mona