UNIVERSIDAD
COMPLUTENSE
MADRID

UNIVERSITÀ
DEGLI STUDI
FIRENZE

Machine learning models for sports prediction and betting

Fernando Díaz Porras

Antonio Barrena Lara

Santiago Larregla Apaolaza

Rubén Colmenar Moreno

Alicia Torres García

Directed by: Lorenzo Amato

July 1, 2021

Abstract

Getting a good betting strategy has always been the goal of mathematicians during many years. In this work, we will design some betting strategies with machine learning techniques and compare them to make as much profit as possible.
We will start with a strong pre-processing of data to obtain the values of each player from the data we have up to each date. Subsequently, three different models are made and a selection of variables is added. From here, five different strategies are proposed for betting, obtaining the performance of each one of them.

Keywords

Tennis, strategy, bookmakers, model, accuracy, ROI.

Contents

1 Introduction

Historical tennis data is widely available online, some websites provide access to information about players, the outcome of matches, and statistics related to player performance in particular matches. Some websites provide historical data in structured form (CSV or Excel files). On the website `GitHub`, ([1]) we can find the statistics of the games played in the last 20 years. On the site `tennisdata`, ([2]) we can find the odds of some bookmakers for the same games.

The amount of data that is available online is one of the advantages for trying to predict tennis matches. Another advantage is that there are no ties, which simplifies a lot of studies. The last reason tennis is chosen over other sports is the small number of variables used in our models, while soccer and other similar sports need the conditions of the pitch, the shape of players, location, etc.

Figure 1: Wimbledon match

We will try to answer this question: *can we improve the predictions made by bookmakers?* With all information discovered, we will design a model that predicts the result of the matches based on the characteristics of each match and each player.

On the other hand, the second question to answer is: *can we develop a profitable betting strategy?* With the help of the model, we will create 6 betting strategies. After that, we will compare them and we will choose the best strategy with the goal of earning as much money as possible. However, there are some nuances that have to be highlighted as the volatility. We do not want strategies with high volatility because the probability of losing money increases. We will discuss this point and much more in the following sections of this work.

To answer these questions, we will develop during this work three sections. Pre-processing techniques, where we explain the technical part of this work, the train machine learning models part, where we train our models on a dataset that we create, and the betting strategy section where we describe all strategies. Finally, we include a section for the conclusions of the results obtained.

2 Pre-processing techniques

The chosen programming language is Python due to the good handling of dataframes and datasets. As mentioned before, we will be using big datasets and choosing the appropriate language is crucial.

We have two databases, distinguished by year, one in CSV format and the other in XLSX format. On the one hand, we will use the years 2015, 2016, and 2017 and group them in one database that contains many characteristics of each game during these years in the main tennis tournaments. We will be focused on 2016 and 2017, but we need the information of 2015 to get a better base on the averages that are going to use to make the predictive model. On the other hand, the second database is only based on the odds of several bookmakers during the years 2016 and 2017.

The first problem that we faced is to merge both datasets. Each player can play every year several matches, so to solve that, we decided to take into consideration the rank points and the name of each player and join it with every match.

The second problem was to consider the appropriate variables in our dataset. First, we considered all variables in our files and we defined new variables with the purpose of explaining better the characteristics of the players. To avoid managing a lot of variables, we made a selection of the variables.

2.1 Summary of Features

In the table 1, we provide a summary of all extracted features. It should be noted that all variables in the table 1 will be doubled, by the winning and losing player, except for those referring to matches.

Most of the variables were imported from the dataset, but some of the variables were calculated based on the needs of the model by elemental operations. For example, $w\_complete = w\_total\_service\_p \cdot l\_total\_return\_p$ to obtain the number of long points (with more hits than serve and return).

At this point, some variables took the null value, so we had to take care in not doing some inappropriate operations.

Feature	Explanation
avg	Average of the odds of all bookmakers
B365	Betting odds in Bet365
EX	Betting odds in EX
LB	Betting odds in LB
sets	Number of sets make by each player
Max	Betting odds in Max
PS	Betting odds in PS
1stIn	Number of successful first services
1stWon	Number of points with the first serve
2ndWon	Number of points with the second serve
ace	Number of aces
bpFaced	Number of break points faced by the player
Saved	Number of break points saved by the player
svpt	Number of points with the serve
age	Age of the player
hand	Righthanded or lefthanded
id	Player id
name	Name
rank_points	Ranking points of the player
match_num	Number of the match
minutes	Average of minutes played
tourney_date	Date of the tournament
2ndIn	Number of successful second serves
1st_serve	Percentage of successful first serves
1st_serve_points_won_p	Percentage of first serve points won
2nd_serve_points_won_p	Percentage of second serve points won
1st_serve_return_points_won	Number of return points won with the fist serve
2nd_serve_return_points_won	Number of return points won with the second serve
1st_serve_return_points_won_p	Percentage of return points won with the fist serve
2nd_serve_return_points_won_p	Percentage of return points won with the second serve
break_points_won	Number of break points won
total_service_p	Percentage of winning the point with the serve
total_return_p	Percentage of winning a return point
adv_on_serve	Additional percentage with the serve
complete	Percentage of points with more than 2 hits

Table 1: Summary of Features

3 Train machine learning models

A training and validation study of a machine learning model will be carried out, using the previously explained data set. For the analysis, the standard procedure followed in the development of prediction models will be followed.

1. *Data cleaning*

It is the first step for the development of any model, it tries to eliminate or impute the values missing, outliers or null values, to carry out a study of the variables, to analyse the correlation between them, ...

2. Development of the model

   The *dataset* is divided into a training, validation and test set.

   - **Train**

     As a training set we chose the years **2015-2017**, with these set we train our model
   - **Validation**

     As a validation set we chose the year **2018**
   - **Test**

     As a test set we chose the year **2019-2020** where the odds obtained will also be compared with bookmakers' odds.

3. Validation of the model

   To evaluate the performance of the model, this last step, the validation of the model, will be carried out using the test set in which the model will be predicted and different metrics will be obtained to evaluate the validity of the model. The metrics we are going to use are the following:

   - Accuracy, model accuracy is defined as the number of classifications a model correctly predicts divided by the total number of predictions made. It's a way of assessing the performance of a model, but certainly not the only way.
   - ROI

Three supervised learning models will be developed: logistic regression, *Random Forest* y *XGBoost*.

3.1 Feature selection

Variable selection is the process of selecting a subset of all available traits for use in a model. The main motivation for applying this technique to tennis match prediction is the possibility of improving the prediction accuracy by removing irrelevant features. Prediction accuracy by removing irrelevant features. A model with fewer features has a lower variance, which avoids overfitting the training set. In addition, feature selection will allow us to know the relative importance of the different features in predicting match outcomes There are two main flavours of feature selection algorithms: forward selection and backward elimination.
In forward selection, the features which cause the greatest improvement in the evaluation metric are progressively added, until all features have been added or no improvement is gained by adding additional features. Conversely, backward elimination begins the full set of features and removes those whose elimination results in the greatest improvement in the evaluation metric. Algorithms 1 and 2 give the pseudo code for forward selection and backward elimination, respectively.

Algorithm 1 Forward Selection

1: Let M_0 be the model just with an intercept (thus, prediction = sample mean)
2: For k = 0, ..., p-1 ;

2.1 Consider all p-k models increasing the number of regressors of M_k with one additional regressor;

2.2 Choose the best model according to R2 among these models. This will be the model M_{k+1}

3: Choose M_k for k= 0, 1, ..., p using a model assessment indicator (Cp, BIC, R2, cross-validation)

Algorithm 2 Backward Elimination

1: Let M_0 be the model just with an intercept (thus, prediction = sample mean)
2: For k = p, p-1,..., 1;

2.1 Consider all k models decreasing the number of regressors of M_k with one additional regressor

2.2 Choose the best model according to R2 among these models. This will be model M_{k-1}:

3: Choose M_k for k= 0, 1, ..., p using a model assessment indicator (Cp, BIC, R2, cross-validation)

Each approach selects a different optimal number of features. For example, forward selection selected 14 of 66 features and backward elimination selected 17 of 66 features. In the table 2

Forward	**Backward**
j1_avg	j1_svpt
j2_avg	j1_1stIn
j2_PS	j1_1stWon
j2_adv_on_serve	j1_2ndWon
j2_total_service_p	j1_avg
j1_2nd_serve_points_won_p	j1_age
j2_total_return_p	j1_B365
j1_break_points_won	j1_PS
j1_complete	j1_2ndIn
j1_PS	j1_break_points_won
j2_age	j2_svpt
j1_B365	j2_1stWon
j1_age	j2_2ndWon
j1_bpSaved	j2_avg
	j2_age
	j2_PS
	j2_break_points_won

Table 2: Feature Selection

3.2 Logistic Regression Model

Logistic regression is a type of generalised regression used as a classification method, used to predict the outcome of a categorical variable as a function of the predictor variables. If the dependent variable has two categories, it is a binary model, while if it has more than two categories, it is a multinomial model.
The purpose is to predict the probability of occurrence of an event. This assignment of probability to an individual is based on the characteristics of the individuals to whom the event does or does not actually occur.
Therefore, a new variable is generated as a result of the prediction. In addition, it will deliver the weight of each variable according to the level of incidence in the increase or decrease of that probability.
The closer the actual values match the predicted values, the better the model fit. The overall fit is assessed by the likelihood statistic, which is distributed as a Chi-square.
To avoid the problems that a linear model would give, we need to model the probability $p(x)$ with a function that returns values between 0 and 1. There are many functions that do this, but in particular logistic regression uses a function called **logistics function**:

$$P(Y = 1/X) = \frac{e^{\beta_0+\beta_1 X_1+\ldots+\beta_n X_n}}{1 + e^{\beta_0+\beta_1 X_1+\ldots+\beta_n X_n}}$$

where $P(Y = 1/X)$ is the probability that Y takes the value 1, which indicates the presence of the characteristic given the values of the covariates $X = (x_1, \ldots, x_n)$, β_0 is the constant of the model and β_i the weights associated with each of the covariates x_i.
If we divide the above function by its complementary, we obtain the **Odd ratio**, which is used to compare the influence of the explanatory (or independent) variables on the dependent variable.

$$\frac{P(Y = 1/X)}{1 - P(Y = 1/X)} = \exp\left(\beta_0 + \sum_{i=1}^{n} \beta_i X_i\right)$$

If we apply the logarithm, we are left with a linear equation,

$$\log\left(\frac{P(Y = 1/X)}{1 - P(Y = 1/X)}\right) = \beta_0 + \sum_{i=1}^{n} \beta_i X_i \rightarrow \log\left(\frac{p_i}{1 - p_i}\right) = \beta_0 + \sum_{i=1}^{n} \beta_i X_i$$

For the estimation of the coefficients, maximum likelihood estimators are used, which are calculated using iterative methods, i.e., estimations that maximise the probability of obtaining values of the dependent variable.
Another important aspect to remember is that qualitative variables with more than one class must be transformed to as many dummy variables as categories minus 1 have.

Feature	Coefficient Feature	Coefficient	
β_0	-0.001526	j2_PS	-0.018691
j2_2nd_serve_return_points_won_p	0.493936	j2_avg	-0.022592
j1_2ndWon	0.347540	j2_1st_serve_return_points_won	-0.041257
j2_1stIn	0.145500	j2_svpt	-0.041288
j2_1stWon	0.115927	j2_ace	-0.059978
j1_hand_R	0.101569	j2_age	-0.097790
j2_Max	0.090288	j1_1stIn	-0.103563
j1_age	0.057628	j1_1stWon	-0.119948
j2_1st_serve_return_points_won_p	0.051911	j2_complete	-0.127918
j1_ace	0.046110	j1_Max	-0.129941
j1_rank_points	0.043467	j2_2ndWon	-0.268956
j2_1st_serve	0.041536	j1_2nd_serve_return_points_won_p	-0.399714
j2_break_points_won	0.039275		
j2_total_service_p	0.036979		
j2_total_return_p	0.034466		
j2_2nd_serve_return_points_won	0.029731		
j1_1st_serve_return_points_won	0.029725		
j2_2ndIn	0.028243		
j2_1st_serve_points_won_p	0.027791		
j1_1st_serve	0.027613		
j1_2ndIn	0.026489		
j1_bpSaved	0.025840		
j1_svpt	0.024241		
j1_break_points_won	0.023731		
j1_PS	0.021869		
j1_avg	0.016850		
j2_rank_points	0.016372		
j1_1st_serve_points_won_p	0.015728		
j1_adv_on_serve	0.012624		
j1_2nd_serve_points_won_p	0.011863		
j1_B365	0.010588		
j2_adv_on_serve	0.005415		
j1_total_return_p	0.003378		
j1_complete	0.000434		
j1_minutes	-0.000002		
j2_minutes	-0.000010		
j2_B365	-0.000212		
j1_2nd_serve_return_points_won	-0.003503		
j1_total_service_p	-0.005077		
j2_bpSaved	-0.006810		
j1_1st_serve_return_points_won_p	-0.008546		
j2_2nd_serve_points_won_p	-0.009296		

Table 3: Coefficients Logistic Regression

3.3 Random Forest

A tree is a graphical and analytical way of representing all events and occurrences that may arise from a decision taken at a certain point in time. They help us to make the 'best' decision, from a probabilistic point of view, from a range of possible decisions.
It allows to visually display the problem and to organise the calculation work to be done.
The above process can produce good predictions on the training set, but is likely to overfit the data, leading to poor performance on the test set. A smaller tree with fewer splits can lead to lower variance and lower bias. A better strategy is to grow a very large tree and then prune it to obtain a subtree, for this we use weaker link pruning, known as cost-complexity pruning.
The *random forest* is a combination of predictor trees, i.e., instead of fitting a single tree, many of them are fitted in parallel to form a 'forest'. In each new prediction, all trees that form the 'forest' participate by contributing their predictions.
The Random forest method is a modification of the bagging process. Recall that the bagging process is based on the fact that, by averaging a set of models, the variance is reduced.
Random forest makes a random selection of m predictors before evaluating each split. The best way to find the optimal value of m is to evaluate the Out-Of-Bag (OOB) error for different values of m.
The OOB is a way of validating the random forest model, it is the average error for each x_i calculated using predictions of the trees that do not contain x_i in their respective bootstrap sample.

In the table 4, we can see the 15 most important features

Feature	**Importance**
j2_PS	0.103121
j1_avg	0.096936
j1_PS	0.095840
j2_Max	0.092333
j1_Max	0.085813
j2_avg	0.081877
j1_B365	0.069550
j2_B365	0.067727
j2_total_service_p	0.016078
j2_adv_on_serve	0.015714
j2_rank_points	0.014872
j2_complete	0.013274
j1_rank_points	0.011511
j2_2nd_serve_points_won_p	0.011301
j2_1st_serve_points_won_p	0.010058

Table 4: 15 most important features in Random Forest model

3.4 XGBoost

XGBoost (Extreme Gradient Boosting) is a supervised predictive algorithm that uses the principle of boosting. The idea of boosting is to generate several weak prediction models, which in this case, are our decision trees, but the boosting results of these, due to the sequential processing with a loss or cost function, which minimises the error iteration after iteration, thus making it a stronger model, with better predictive power and greater stability in its results. To achieve a stronger model from these weak models, an optimisation algorithm is used, in this case Gradient Descent.

During the training phase, the necessary parameters for each of the weak models are iterative and adjusted in an attempt to find the minimum of an objective function, which can be the classification error ratio, AUC, the RMSE, ... Each model obtained is compared with the previous one, and if a model with better results is obtained, then this is taken as the basis for making the relevant modifications. If, on the other hand, worse results are obtained, we go back to the previous best model and modify it in a different way.
The previous process is repeated until we reach a point where the difference between models is negligible, which indicates that we have found the best possible model or when we reach a maximum number of iterations previously set by the user. XGBoost uses as its weak models decision trees of different types, which can be used for classification and regression tasks.

In the table 5, we can see the 15 most important features in XGBoost model

Feature	**Importance**
j1_PS	0.341048
j1_Max	0.186025
j1_avg	0.153304
j2_PS	0.088964
j2_Max	0.017621
j1_2nd_serve_points_won_p	0.016232
j1_2nd_serve_return_points_won_p	0.013618
j2_bpSaved	0.012760
j2_total_service_p	0.012519
j2_1st_serve_points_won_p	0.012429
j2_2nd_serve_points_won_p	0.012113
j1_B365	0.011669
j1_1st_serve_points_won_p	0.010872
j2_total_return_p	0.008723
j1_total_return_p	0.007599

Table 5: 15 most important features in XGBoost model

3.5 Results

Once all models have been made, we make a comparative table of the accuracy of all of them.

	LR	LR forward	LR forward	XGBoost	Random Forest
Accuracy train	0.70474	0.703460	0.70646268	0.7230483	0.73105
Accuracy validation	0.69102	0.68816	0.68938	0.704489	0.720816
Accuracy test	0.672993	0.6775956	0.67960	0.684785	0.68162

Table 6: Accuracy of all models

In the table above, we can see the efficiency of the models, highlighting the gradients boost and random forest, and we can also see how by making a selection of variables, the logistic regression model improves in the test sets.

We also calculate the accuracy of bookmakers by obtaining:

$$\text{Accuracy of B365} = 0.66281421$$

$$\text{Accuracy of PS} = 0.676105$$

4 Betting strategy

Ultimately, we can develop a betting strategy and evaluate our model by betting against a bookmaker. In this case, the main metric used to evaluate the models in the literature is the return on investment. We have developed 5 different betting strategies, and one sixth as a combination of two of the previous ones.

- **Strategy 1:** This is probably the most basic strategy, and it consists of placing a bet of capital fixed to the winner (according to our model) of each match.
- **Strategy 2:** In this other strategy, we place a bet in a match, only if the probability of the winner in our model is bigger than the probability of the bookmaker for that same player. The bet is always the same fixed quantity.
- **Strategy 3:** This strategy is the same as the previous one, but in this case the bet is proportional to the difference between our probability for the winner and the bookmakers' probability for that player.
- **Strategy 4:** This strategy is different to the previous ones since it is the first one in which we consider the possibility of placing a bet on the loser (according to our model). Of course, if we do this for a reason, the payment would be bigger. To apply this strategy, we fix a lower bound and a difference and we only make a bet of the capital fixed if the player has a probability higher than the lower bound, and the difference between the probability of our model and the bookmakers' probability is higher than the difference chosen.

- **Strategy 5:** This strategy is similar to the previous one, with the difference that, as in the third strategy, here, the bet on the loser will be proportional to the difference between the probability of our model and the bookmakers' probability. Of course, the bet is only placed if the two conditions mentioned in the previous strategy are satisfied.

- **Strategy 6:** This final strategy combines Strategy 3 and Strategy 4.

For the moment, we only have compared the accuracy of the different models, but we are more interested in obtaining the biggest possible profits and not just in predicting the results of each game. For this reason, we use six strategies in each model, and we see how much money we would obtain.

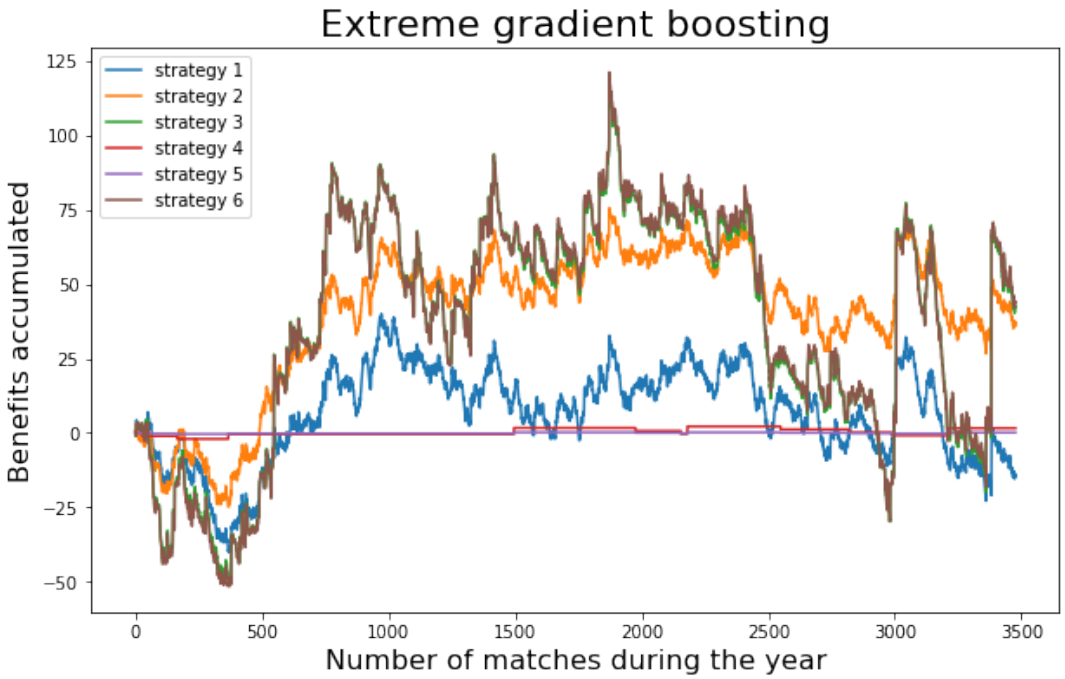

Figure 2: Benefits accumulated with Extreme Gradient Boosting during 2019-2020

In the picture, we can start thinking that some strategies are far better than the other ones. This could be misleading, because there are large differences between the the money we bet following some strategies or others. For example, on one side we have the first strategy, in which we place a bet of a fixed capital in each match, and on the other side we have the fourth strategy, in which we only place a bet if two demanding conditions were satisfied.

Because of this, we need to compute the profits as a percentage of the money invested in each strategy. Doing this for each model, we obtain the following table.

PROFITS PERCENTAGE						
Model	Strat1	Strat2	Strat3	Strat4	Strat5	Strat6
LR	-0.46%	2.510%	1.860%	53.33%	54.25%	3.020%
XGB	-0.39%	1.420%	1.650%	14.63%	9.140%	1.710%
RF	0.960%	3.530%	2.390%	58.74%	47.88%	2.480%
LRf	-1.06%	1.480%	1.760%	-17.5%	-19.7%	1.470%
LRb	-0.40%	2.280%	1.370%	35.14%	31.42%	2.010%

Figure 3: Table of returns of each model in 2019-2020

Now we are in a condition to select the best model, the Random Forest. This model is the second best model according to the accuracy of the test data, and the first one according to the validation data. In addition, following the Random Forest predictions, we would obtain higher profits (in percentage).

In the following two figures, we see the benefits accumulated and the profit percentage for the chosen model.

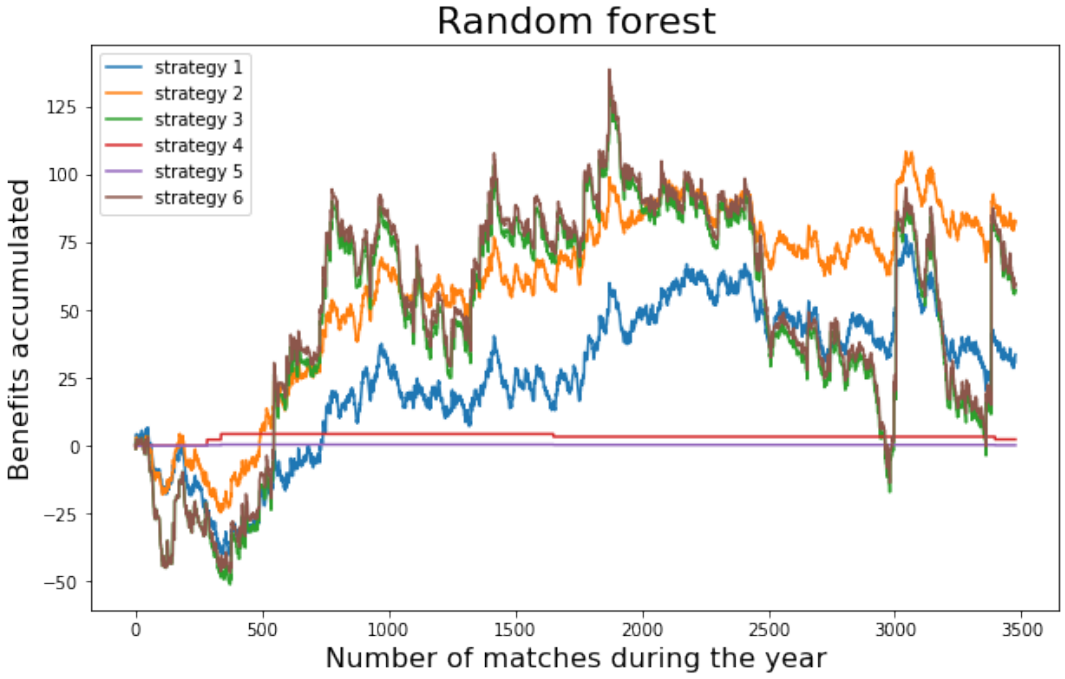

Figure 4: Benefits accumulated with Random Forest during 2019-2020

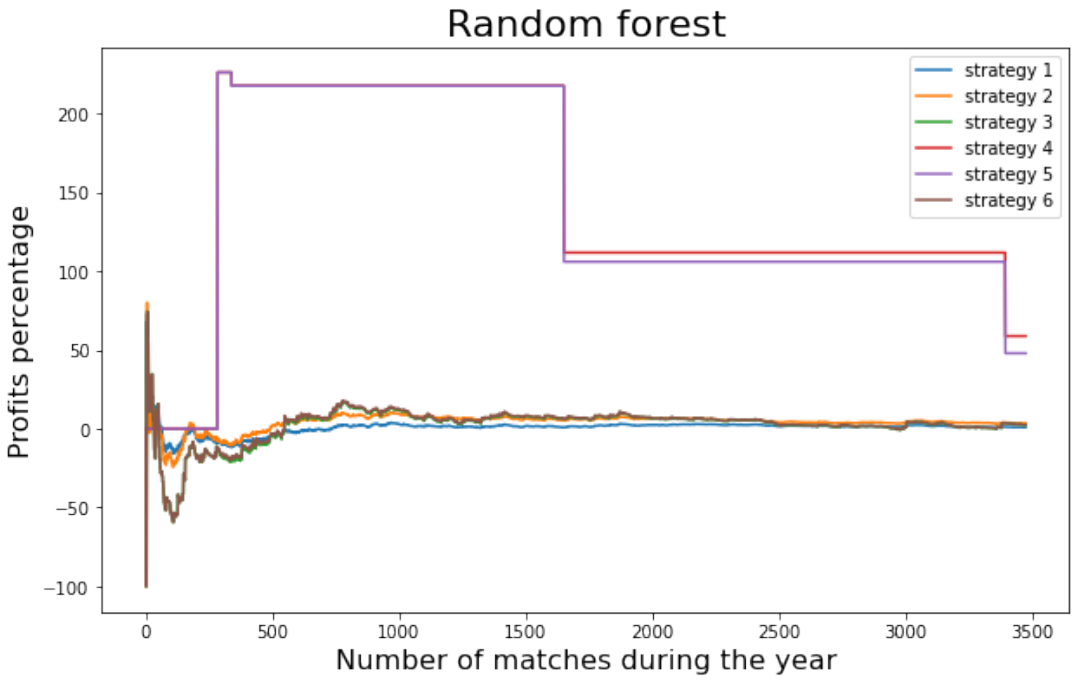

Figure 5: Profits percentage with Random Forest during 2019-2020

5 Conclusions

The development of optimal betting strategies is not an easy task. Throughout the work, three different models have been developed, obtaining that the best of all in terms of accuracy (in the test set) is XGBoost with an accuracy of 68.48% followed very closely by Random Forest with an accuracy of 68.16%, and in last place the logistic regression. If we consider the profit percentage for each model, we would choose the Random Forest model, because is the model that allows us to earn more money (as profits' percentage). Since the accuracy is just a little bit lower than the XGBoost, we will define the Random Forest model as the final model.

In reference to the strategies, the chosen one would be Strategy 6, because combining the Strategy 3, which is a consistent strategy that will practically ensure that we will earn some profit, and Strategy 4, that is a much more risky strategy but it could give us a higher profit percentage. This would be equivalent in some way to build a portfolio with the majority of funds in low-return, low-risk assets, and a small part of funds in assets with higher returns and risk. In this way and moving the capital that we have available with the different volatilities associated with the two strategies selected, we can maximize our profit.

Considering the precision of our models, we can obtain as a general conclusion, that access to bets through predefined strategies of machine learning study may be a viable option in the case that you have a surplus capital that you do not know where to invest. Should never be an option when you are going to invest a necessary or important money for you. Keep in mind that a bookmaker always starts with an advantage respect to the user, this means that in the vast majority of cases always earn money.

Carry out a long strategy to make a profit against a bookmaker cannot be compared to investing your funds in other types of financial assets. On the other hand, there is a danger of gambling behind betting at any bookmaker, which makes it more inappropriate to take this path as the main route to profit.

References

[1] https://github.com/JeffSackmann/tennis_atp

[2] http://www.tennisdata.co.uk/alldata.php

[3] Sipko, M., & Knottenbelt, W. (2015). Machine learning for the prediction of professional tennis matches. MEng computing final year project, Imperial College London.