Tennis Betting Strategies based on Neural Networks

Jeremy Flahault[a], Nicolas Le Roger[a], Marius Cristian Frunza[b,c]

[a]*Dauphine University, Place du Marechal de Lattre de Tassigny, 75016, Paris, France*
[b]*Ural Federal University(UrFU), 620002, 19 Mira street, Ekaterinburg, Russia*
[c]*Schwarzthal Tech,231b Business Design Centre, 52 Upper Street, Islington, London, United Kingdom, N1 0QH*

Abstract

The aim of this paper is to explore betting strategies for tennis matches using neural networks. We used public data and we implemented a neural network prediction model, with an accuracy of 85% on the validation and testing sets. Based on the predictive model we tested several investment strategies which incorporates investor's risk profile. The optimal strategy is to place bets on games where our model indicates a high likelihood of victory and an appropriate bookmaker's odds.

Keywords: Sport betting, Tennis, Machine Learning, Neural Networks, Statistical Models

1. Introduction

Gambling has been skyrocketing in the past few years. In the United States in 2018, the gambling market revenue reached 79.42 billion USD, while the worldwide revenue reached 449.3 billion USD. Moreover, the global online gambling market was valued at 53.7 billion USD in 2019 and is expected to grow at a fast pace in this new decade.

Each year between May and June, the Roland Garros and Wimbledon tennis tournaments are held back to back. Therefore, during this period tennis becomes the most dominant gambling sport and headlines of the sports section of every media outlet in the UK, the US, and France among other countries. It is one of the reasons why online gambling on tennis matches has grown to such an extent, that it has now overtaken betting on horse races and comes in second betting market after football. As the market grows, there is an increasing potential profit for online gamblers.

Email addresses: jeremy.flahault@gmail.com (Jeremy Flahault), lerogern@hotmail.fr (Nicolas Le Roger)

This article enriches the literature of betting models on tennis matches and explores whether betting strategies on tennis can be a profitable long term investments. The remainder of our paper is structured as follows:

Contents

2. Literature Review

2.1. Theoretical Models

The very first tennis models were based on the assumptions that the probability of winning a point is constant. Barnett and Clarke (2002) build the pioneer model using a Markov Chain based on that simple assumption and explained their idea very clearly. Let's take two players, A and B, with a constant probability p of winning a point. P(a,b) is the probability that A wins given the score (a,b), we have:

$$P(a,b) = pP(a+1,b) + (1-p)P(a,b+1) \tag{1}$$

Barnett et al. (2006), figured out that using a Markov chain model to predict the length and outcomes of tennis matches, where the probability of each player winning a point on serve is independent and identically distributed (i.i.d.), overestimates the number of games and sets played in a match. A revised Markov chain model is then formulated, followed by a revised model for games in a match that has an additive effect on the probability of the server winning a point.

The assumption of independence on the distribution of tennis points was investigated by Klaassen and Magnus (2001). They concluded that although points are not independent and identically distributed (i.i.d.), the deviations from i.i.d. are small and hence the i.i.d. assumption is justified in many applications, such as forecasting. Indeed, they have tested for independence of points from four years of Wimbledon point-by-point data and showed that winning the previous point has a positive effect on winning the current point, and at important points it is more difficult for the server to win the point than at less important points.

O'Malley (2008) introduced the closed-form "Tennis formula" by defining the probability of winning a game with only one parameter: the probability of winning a point.

$$\mathbb{P}(WinGame) = p^4\left(15 - 4p - \frac{10p^2}{1-2p(1-p)}\right) \tag{2}$$

Based on this formula, O'Malley (2008) derives the probability of a player to win a tiebreaker, set and match, assuming that the probability of winning a point follows a Bernoulli random variable. He succeed to set up a few equations based on 2 parameters (the probability of winning a point on serve or on return) such as the probability of

winning a best-of-three-sets or best-of-five-sets match.

The common-opponent stochastic model for predicting the outcome of professional tennis matches by introduced by Knottenbelt et al. (2012) is mainly based on Barnett et al. (2006) and O'Malley (2008) studies. This model is also relying on the strengths of the players as server or receiver and their probability to win a point in both situations.

The limit of these models, is that they are very global. Theoretically, we could model each match, but they lack accuracy and it is nearly impossible to forecast consistent results and beat the market with those models. Indeed, one can easily see that Barnett et al. (2006)'s model does not take into consideration the surface of the court, which is an important driver of the outcome. Therefore, using these model types involves a tremendous data-cleaning work in order to get an accurate probability given specific parameters such as the surface of the court and the year of the match.

Since tennis is an individual sport, a punter on tennis matches should have a different approach from a punter on a collective sport. For instance, with football gambling, the right approach would be to model the strength of the strikers of team A and the strength of the defenders of team B (and vice versa) to know which team is likely to score more goals, and thus to win (Dixon and Coles (1997)). However with tennis, the complexity does not come from the players, but from their performance with respect to multiple game parameters: the surface of the court, the location of the tournament, the opponents, the prize, their rankings etc. Therefore, a comprehensive model for predicting the results of tennis matches should encompass all these aspects.

2.2. Econometric Models

This section reviews the advances in tennis matches modeling, with a focus on econometric based methods and machine learning techniques, and their applications to betting.

Klaassen and Magnus (2003) provided a model of the probability of a given player winning a tennis match, with the prediction updated on a point-by-point basis. Easton and Uylangco (2010) did a **point-by-point comparison** of that model with the probability of a given player winning the match, as implied by betting odds. The predictions implied by the betting odds match the model predictions closely, with an extremely

high correlation being found between the model and the betting market. The results for both men's and women's matches also suggest that there is a high level of efficiency in the betting market, demonstrating that betting markets are a good predictor of the outcomes of tennis matches. The significance of service breaks and service being held is anticipated up to four points prior to the end of the game. However, the tendency of players to lose more points than would be expected after conceding a break of service is not captured instantaneously in betting odds. Easton and Uylangco (2010) found no evidence of a biased reaction to a player winning a game on service.

More recently Ingram (2019) presented a **point-based Bayesian hierarchical model** for predicting the outcome of tennis matches. The model predicts the probability of winning a point on serve given the surface, the tournament and match date. Each player is given a serve and return skill which is assumed to follow a Gaussian random walk over time. Each player's skills depend on the surface and other parameters. When evaluated on the ATP's 2014 season, the model outperforms other point-based models, predicting match outcomes with greater accuracy (68.8%) and lower log loss. Results provided by Ingram (2019) are competitive with approaches modeling the match outcome directly, demonstrating the forecasting potential of the point-based modeling approach.

Lisi and Zanella (2013) proposed a **logistic regression** to forecast winners regarding 2012 tournaments. Based on an Akaike's Information Criterion they found out that five explanatory variables are needed to best explain the probability of the favorite player to win: ATP points, age, surface, home factor, odds of the bookmakers that reflect external information such as physical conditions, a particular sequence of wins or defeats, etc. Their out of sample test is made on the four grand slam matches in 2013. The the model correctly guesses the final outcome 77.2% of times against the 78% for bookmakers.

Sim and Choi (2019) suggested a **stochastic discrete-time Markov chain model** trained on a point-by-point dataset of men's single matches played in the ATP tour from 2011 to 2015. Their statistical test results show that the identicality of point winning probabilities is not a valid assumption. The server's point winning probability from scores 40:0, 30:15, 15:30, and 0:40 are significantly different. On the other hand, the independence is proven to be a generally valid assumption except for 40:15 where who won the previous point influences the point winning probability. Game winning

probabilities and the importance of each point in winning a game are analyzed using the Markov Chain model by court surfaces and player groups of the different levels of serve effectiveness.

Gu and Saaty (2019) structured an **Analytic Network Process model** which is a multi-criteria regression model. They used 44 situational and performance data factors for each match. 8 factors are indicators of the basic information of a match, and 36 factors are descriptive and performance parameters for both players (18 for each player). Half of the 18 factors are descriptive parameters and the other half are performance metrics. The accuracy of their model is estimated at 85.1%.

Del Corral and Prieto-Rodriguez (2010) tested whether the differences in rankings between individual players are good predictors for Grand Slam tennis outcomes. This study estimates separate **probit**[1] **models** for men and women using Grand Slam data from 2005 to 2008. The explanatory variables are divided into three groups: a player's past performance, a player's physical characteristics, and a match's characteristics. Del Corral and Prieto-Rodriguez (2010) explore three alternative probit models. In the first model, all of the explanatory variables are included, whereas in the other two models, either the player's physical characteristics or the player's past performances are not considered. The accuracies of the different models are evaluated both in-sample and out-of-sample by computing Brier scores and comparing the predicted probabilities with the actual outcomes from the Grand Slam tennis matches from 2005 to 2008 and from the 2009 Australian Open.

2.2.1. Bradley-Terry type model

McHale and Morton (2011) used a Bradley-Terry type model (Bradley and Terry (1952)) which was mostly used for handling data on paired comparisons such as citation patterns in statistical journals. Applied to tennis, this model was first used by Glickman (1999) who followed Dixon and Coles (1997)'s approach in predicting football outcomes and showed that it was possible to update the players strength by using the maximum likelihood, but he did not try to forecast any match with it. His model also gave the number of games won by each player during a match. Later, McHale and Morton (2011) used it and they were able to compute the probability of victory of player i over

[1]In statistics, a probit model is a type of regression where the dependent variable can take only two values. *Wikipedia*

player j the following way:

$$\frac{\alpha_i}{\alpha_i + \alpha_j}$$

where α_i and α_j are positive-valued parameters representing each player's ability.

The weights of the historical results were set using an exponential decay function and the input parameters of their model were the surface and the control of the half life of the decay function.

However, McHale and Morton (2011) noted that their model had no allowance for any dependence between consecutive games hence that it dd not account for the intrinsic structure of tennis, where players are far more likely to win games when serving than when receiving.

Their betting strategy was adjusted on a season by season basis and to select their best strategy they considered investing on matches generating the highest profit (best odds). The history of matches they used for this strategy started between 2001 and 2003 and increased with time. This strategy still had a yearly return above 10% (except in 2005), but the number of bets won was well below 50%.

2.3. Machine Learning approaches

With more and more data easily accessible, machine learning is growing at a fast pace. It can be useful to review past researches with new approaches.

Cornman et al. (2017) investigated many different machine learning fields for the professional tennis matches prediction. Using 5000 matches and different models such as **Logistic Regression or Support Vector Machine**, they managed to create a strategy that earns an average of 3.3% per match on test set data with a random forest.

The last well-known machine learning model that is extremely useful with information hard to synthesize is the **Neural Network model**. Based on a Neural Network with Multi Layer Perceptrons (MLP), Somboonphokkaphan et al. (2009) used statistical and environmental data with approximately 10 features and trained a three-layer feed-forward Artificial Neural Network (ANN). Their best trained model had a predicting accuracy of 75% for Grand Slam tournaments between 2007 and 2008.

A more recent machine learning model used to predict professional tennis matches was proposed by Sipko (2015). He considered a supervised machine learning algorithm and used players' historical performance by taking a wide variety of statistics to predict match outcomes. He defined a novel method of extracting 22 features from raw

historical data, including abstract features, such as player fatigue and injury. Using the resulting dataset, he developed and optimised models based on two machine learning algorithms: **logistic regression and artificial neural networks**. When evaluated on a test set of 6,315 ATP matches played in the years 2013-2014, Sipko (2015)'s models outperformed Knottenbelt et al. (2012)'s common-opponent model, the contemporary state-of-the-art stochastic model. His neural network generated a return on investment of 4.35% against the betting market, an improvement of about 75%.

Wilkens (2020) extended previous research by applying a **wide range of machine learning techniques**[2], covering ten years of male and female professional singles matches. Wilkens (2020) analyzed whether a variety of machine learning techniques outperforms simpler techniques such as logistic regressions, with regard to predicting the outcome of matches. The paper found that the odds of bookmakers encompass most of the available information to predict the outcomes of matches. Nevertheless, long term betting strategies applied to multiple prediction models and using various money management strategies are mainly negative (unless one has access to the most favorable market quotes).

3. Model Selection

Previous models described in the literature had not used Machine Learning algorithms thoroughly as an investment strategy. Somboonphokkaphan et al. (2009) used neural networks to predict the result over a short period (1 or 2 years) with more or less success (3.3 % earnings per match, between 75 % and 80 % of accuracy). Therefore, this paper explores an approach using neural networks as a mean of prediction for investing.

ANNs can detect complex relationships between the various features of the match. However, they have a "black box" nature, meaning that the trained network gives us no additional understanding of the system, it is too difficult to interpret. Furthermore, ANNs are prone to overfitting and therefore necessitate a large amount of training data. Also, ANN model development is highly empirical, and the selection of the hyperparameters of the model often requires a trial and error approach. However, due

[2]Logistic regression, neural networks, Support Vector Machine, random forest, gradient boosting machine.

to its success in the above-mentioned experiment, this approach is clearly deserves further investigation.

There are a lot of features that are not taken into consideration with basic strategies. For instance, for some players, the surface of the court is very important. For example, Federer never lost against Wawrinka in 15 confrontations when the surface was 'Hard', but only won 5 times out of 8, when the surface was 'Clay'. If we continue further, we can even notice than Wawrinka won 2 times out of 2 against Federer, when they faced each-other during the 'Monte Carlo Masters'. Knowing that, would you bet on Federer if he faces Wawrinka, and that they are playing on 'Clay' for the 'Monte Carlo Masters'? For this specific match, it is very likely that Federer still has a lower odd than Wawrinka, but knowing these facts, would you bet on Federer?

In addition, some players have more endurance than others, so playing on 'the best of 5 sets' could be more favourable than 'the best of three sets'. As we can see, there are many features which can be used to predict the outcome of a match. However, it is very complex to understand their link in order to get the best prediction. This is why, we believe it is worth creating a Machine Learning algorithm that could learn the hidden links between previous results and features (name of the tournament, surface of the court, number of set to win, name of the opponent...) and increase our profit in order to avoid the non-profitable "lowest odd" strategy.

Neural networks are deep learning algorithms that can be optimized in many different ways. First, one must see a neural network as a polynomial function and the hidden layers correspond to the order of the function. Hidden layers correspond to hidden layers of neurons which are stacked between the input layer and the output layer. As we increase the number of hidden layers, the order of the function increases as well, the function becomes more powerful and can fit more complex data trends. This is why neural networks require a lot of data. The bigger the amount the data and the features, the harder it is for a machine to understand the links between features. However, if the data is easy to understand for the machine or that we have few features, the algorithm can overfit the data and it will decrease the prediction capacity of the model. In such case, one should decrease the number of hidden layers of the model. Similarly, the model can underfit the data when the data is too hard to understand for the machine. In this case, one must increase the number of hidden layers. This is one way of improving a neural network, but there are many more.

For instance, in each hidden layer there is a specific number of neurons. The model is sensitive to this number, meaning that changing this number has an impact on the learning and prediction capacity of the model.

In addition, in our paper, we use supervised feed forward neural networks. These type of neural networks can be viewed as stacks of neurons aggregated through layers. In particular, this type of deep learning algorithm, learns in a specific direction, from left to right as shown in the picture below.

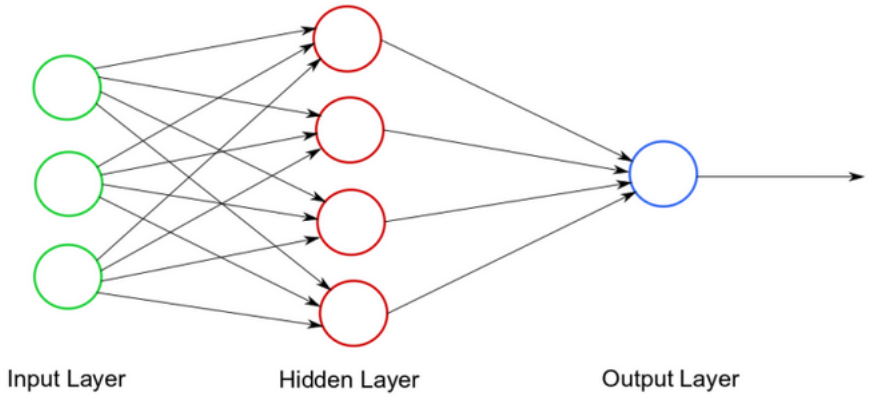

Figure 1: Feed Forward Neural Network

The learning procedure actually consists in setting weights to neurons and the final weight is the output of the model. This is why we obtain values between 0 and 1. However it is possible to get a classified prediction. The prediction would then be an array of 0 and 1. In our case we want a value between 0 and 1 hence we will say that we are doing a regression. In order to update these weights, a back-propagation procedure computes the error term of the final value with respect to the true value and updates the initial weight in order to reduce the error in the next iteration. This procedure goes backwards, starting by the predicted value and going back to the first layer of neurons through the hidden layers.

As one can observe, there are iterations. Indeed, the model can compute a first prediction by setting all the weights randomly but at the end, the weights will not be updated and the model will have not learned anything from the data. By increasing the number of epochs, the model increases its number of iterations. If it is set to two for example, the model will only update once the weights of the model. If the number of epochs is three, the model will update twice the weights of the model etc. Hence, to

increase the learning of the model, one can increase the number of epochs.

Finally, the weights could be set randomly but we are setting the weights with the '*He*' initializer corresponding packages available in Keras, but one could have look at '*Xavier*' or '*He*' initializer to get more information. More importantly, there are activation functions such as "*relu*", and "*sigmoid*" which are mainly used due to better properties than their peers. Activation functions are used in the back-propagation procedure and the "relu" (Rectified Linear Unit) function is quite efficient in many cases as it greatly accelerates gradient descent algorithm convergence, and it needs less computation than "sigmoid" or "Tanh". This is why we will use it for our input and hidden layers, while we will use the sigmoid function for the output layer to have better accuracy. We will not discuss further the other parameters because it is not the purpose of this paper, still one will find the complete model in the python notebook with all the parameters used.

4. Dataset presentation

4.1. Gathering Information

Compared to the previous articles, currently it is much easier to access the data now, therefore we gathered much larger database than what is presented in the academic literature.

First of all, in order to be transparent with every reader, we deliberately state that we downloaded the database from tennis-data.co.uk. This database is one of the most complete, free of access, and comprises many features. Some of them can be very useful to enable us understand why a player has won this particular game against another specific player. The data is as-ambled year by year, so the original data source , from 2001 to 2020, are parsed and merged into a data lake. Our data encompasses the matches of men players and some of the most helpful features are: the year, the tournament, the location, the players involved, the ranking etc. However, we had to delete some useless features such as the ATP columns that did not provide good information. Table 1 shows the features of the initial dataset kept for modeling purposes.

With all these features, our database is composed of 51,945 records representing in average more than 2,700 matches by year with 16 features.

Name of the feature	Example	Description
Date	01/01/2001	Date of the match
Tournament	French Open	Official name of the tournament
Series	Grand Slam	To which category belongs the tournament
Court	Outdoor	Conditions of the match
Surface	Clay	Surface of the court
Round	Final	At which stage the two players competed
Best of	5	Maximum number of set potentially played
Winner	Nadal R.	Name of the winner
Loser	Djokovic N.	Name of the loser
WRank	2	World ranking of the winner*
LRank	1	World ranking of the loser*
Wsets	3	Number of sets won by the winner
Lsets	1	Number of sets won by the loser
Comment	Completed	Comment on the outcome of the match
B365W	1.42	Odd of the winner before the match by Bet365
B365L	1.67	Odd of the loser before the match by Bet365

*At the time of the match

Table 1: Features of the initial dataset kept for modeling purposes

4.2. Dataset cleaning

Our very broad objective is to build a strategy to outperform the market of sport betting. Therefore, thinking in terms of P&L leads us to delete all the rows where odds were missing (less than 5% of our database) since we cannot compute any strategy with missing odds.

Our first major unreal world hypothesis is that we choose to only work with matches that have been completed. Indeed, even if players shared insights on their current physical form, we believe that no bookmaker or algorithm can predict the injury of a player during a match. Therefore, we consider that any retirement once the match began is an outlier and must be deleted (less than 1%). On the other hand, we wanted to highlight the fact than any future strategy will be 'over estimated' by, at maximum, 1%.

Then, we believe that we must take the progression of a player into consideration, so we can't have a single evaluation over his entire career. We decided to update the statistics every year, so we transform the current date format to keep nothing but the year.

Our biggest problem regarding the cleaning of the database was to have only numerical values. Indeed, all the player's names were qualitative values. Even if some encoding function already exists, it would not completely solve our issue; we had to create our own function to have the same "code number" for a player, no matter if he belongs to the 'Winner' or 'Loser' column (Ex: Nadal R. is no longer defined with letters, but has the number 795, either if his name appears in the 'Winner' or 'Loser' column.).

Another problem we faced is that the category of tournaments changed over time. You might be aware that tournaments' importance is defined based on the points offered to the winner. Usually, the bigger the number of points offered, the more important the tournament. Nowadays, the smallest tournament rewards 250 points to the winner (called 'ATP250'), but before 2009 it was named an 'International' tournament. Same for the 'International Gold' that became 'ATP500'. The third category is 'Master 1000', formerly named 'Masters'. Finally, the 'Grand Slam' and 'Masters Cup' tournaments did not change.

4.3. Features selection

As our main objective is to predict the output of a given match, we cannot keep features that result from the outcome of this specific match (i.e. 'Wsets', 'Lsets' and all the games won by the players during the match can not be keep because they can not help us to predict the winner since we only have those values after the match.). In addition, we choose to drop the features that have no intrinsic value like 'Round' since we suppose that no matter the stage of the tournament, each player has the same wish to win, or those that can be described by other labels such as 'Tournament'. Indeed, the 'French Open' (Roland Garros) can be described with the features 'Series', 'Court', 'Surface' and 'Best of' since it is a 'Grand Slam' played 'Outdoor' on 'Clay' with a maximum of '5' sets.

Furthermore, the main idea when you want to analyse data is to have the fewest

features with the greatest information embedded. This is why we chose to adjust a lot our database to extract all the value it has. Indeed, we could have encoded our thirteen variables with the 'One Hot Encoding' method and its available function in python. However, we would have ended up with thousands of features that are made of 0 and 1, so we would have lost an outrageous quality of data. Owing to this fact, and here is the main idea of our data analysis, we have chosen to increase the number of features and to associate it with the percentage of victory of each player given the features.

With no feature engineering and a basic encoding, our database looks like in Table 2.

Winner	**Looser**	Date	Indoor
414	**649**	2018	1
649	**212**	2018	1
414	**721**	2019	1

Table 2: Example of a basic encoding with indoor matches played in 2018 and 2019

One way to understand why the player '414' won against the player '649' in 2019 on an indoor court is by computing the first player's yearly percentage of victory on this type of court. This is why we added the two columns : 'W%W Indoor' and 'L%W Indoor'. 'W%W Indoor' corresponds to the yearly percentage of victory of the winner on an indoor court, and 'L%W Indoor' is the same for the loser.

Winner	**Loser**	Date	Indoor	W%W Indoor	L%W Indoor
414	**649**	2018	1	0.83	0.74
649	**212**	2018	1	0.74	0.75
414	**721**	2019	1	0.90	0.69

Table 3: Extract of our database with indoor matches played in 2018 and 2019 and its statistics.

Now we can understand a little bit more why certain players beat others. Indeed, we can suppose that the way players play is more favourable for some matches with specific conditions. Indeed, it is granted that Rafael Nadal is extremely efficient on 'Clay', but the way he plays is less suitable for 'Indoor' matches on a 'Hard' surface.

One can notice that these statistics are computed yearly. For instance, the player 649 won 74% of his indoor matches in 2018, independently from the match he just

played because it was already taken into consideration for the computation (i.e. no matter if he belongs to the 'Winner' or 'Loser' column). On the other hand, it is understandable that the player 414 did not have the same percentage of victory in 2018 and 2019. Obviously, we apply the same mechanism for each initial label to have a complete database.

Furthermore, notice that the method with which we computed the rates is not the most accurate one. Indeed, we could say we are 'back-testing' everything because we compute these rates at the end of the year, while in reality some matches would not have been played. For instance, if a tournament occurs in January 2019, a player can display, before the event, a 90% rate of winning outdoor matches (this rate comes from all his outdoor matches in 2019), while in reality, at the end of January 2019 he would have played fewer matches and could have had a very different ratio of victory. But as we know all the results of 2019, we decided to give him his yearly percentage of victory and not the percentage 'at the time of the tournament'. Moreover, if we want to apply our overall sports betting strategy in real time, we will need to download the latest data after each tournament, and update all the features after each results, in order to have the most up-to-date database.

Once applying this pattern on all the labels, we obtain a bigger database that initially planed. Recall that we must encode our most relevant features (i.e. we are only displaying a 1 into the corresponding columns related to the information of the match.) and for each initial label (from Table 1.1) we compute the yearly percentage of victory associated with this initial label for the winner and the loser. Let's have a look to an explicit example. If we encode the variable 'Surface', we will create four columns ('Hard', 'Carpet', 'Clay', 'Grass') instead of one ('Surface'). Then for each of these four columns, we will display a '1' if the match is played on one of these surfaces, and 0 otherwise. Finally, for each of these four new columns, we will create two associated columns to compute the percentage of winning match on the specific surface for the two players. If the match is played on 'Hard', this column will display a '1' and the two associated new columns we will be named 'W%W Hard' (for 'Winner player, % win on Hard surface') and 'L%W Hard' (for 'Loser player, % win on Hard surface'). Thereby, our dataset includes 48,326 records and 62 features.

5. Model Calibration

We implemented the neural network model calibration on Python using the "scikit-learn" package, but since we have a large dataset we can use deep learning techniques as it is quite powerful.

First, before optimizing our parameters, we split our dataset into a training and validation set and a test set. The training and validation set gathers observations from 2001 to 2015, and the test set gathers observations from 2016 to 2019. The goal will be to predict the outcome of the games played between 2016 and 2019. A validation set will be defined as a random pool of the training and validation set and will be used to improve the learning and prediction capacity of the model. In our case we choose the validation set to represent 25% of the training plus validation set; hence the training set corresponds to 75% of the global training and validation set.

In order to select the best model, one must find the optimal parameters and hyper-parameters of the model. There are multiple ways to find them, but we will only elucidate three of them. First, one can change the parameters of the model by hand and plot the history of the learning curves on the training set and the validation set. These values shown below give us an indication if the model is overfitting or underfitting.

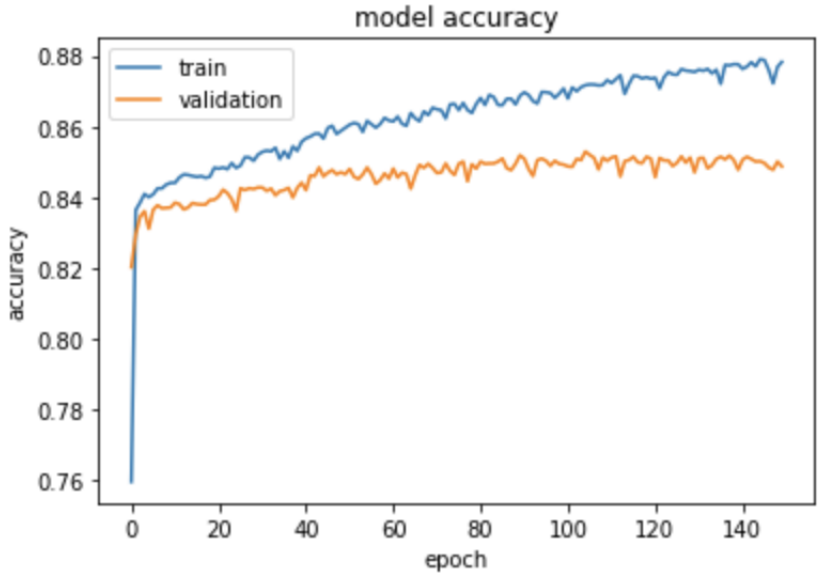

Figure 2: Accuracy of Our Final Model

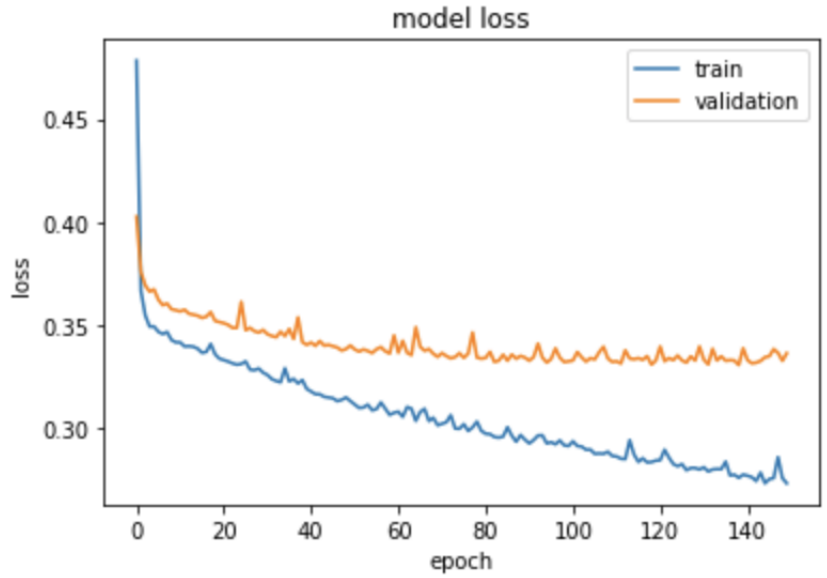

Figure 3: Loss of Our Final Model

If the learning curves of the validation set are diverging from the learning curves of the training set, the parameters used in the model can be improved. In the graphs above, we can a see that the learning curves of the validation set (for the accuracy as well as for the loss) converge to a certain value with a decreasing loss function. It means that our model is quite good, even if it starts to show some signs of overfitting. However, if we had taken more hidden layers for example, we would have observed the accuracy of the validation set decrease and be much more volatile than the one we see here. Similarly, for the loss of the validation set, we would have observed an increase and a higher volatility than what we observe now. As our main objective is not to find the perfect neural network architecture (and this is why we will not talk about the learning rate), we will acknowledge that this model is suitable for our usage.

As we can see on the first graph, the accuracy of our model is close to 85%, meaning that in 85% of the matches played in the validation set, the neural network predicted correctly the outcome. Recall that the validation set corresponds to 25% of the complete training and validation set. The latter is composed of 38'247 values, which means that the algorithm was right 8'127 times out of a total of 9'561 games played.

Still, there are at least two other ways to optimize a neural network's parameters. The second way is by using "grid search" which is also a tool available in "scikit-learn". The way it works is that we must define a grid of parameters and feed it to the model

that will test all the combinations of parameters possible and select the best estimator. The third approach is the bayesian method. This method is more efficient than "grid search" as it computes the probability of a parameter to be optimal and converges towards that optimal parameter. In other words, it does not test all the parameters fed as it founds the optimal one before (if the array given is large enough) testing them all. If one reader desires to reproduce our study with an improved neural network architecture, it could be a good point to start from the bayesian method.

Now that we have a good and reliable model, we will use the test set to predict the outcomes of the matches played between 2016 and 2019. Using the true outcome of each game played during that period of time, we note that the model is correct in 83% of the cases. Using the fact that the model is not always right (17% of the cases), we will elaborate strategies in order to maximize the return on investment of an investor.

5.1. Portfolio investment simulation

There is not point in wanting to predict the outcome of every match. This implies that the gambler seeks for the highest ROI possible. Our model enables an investor to reach a high ROI of 300%-400% every year. However, as we have realized through this analysis, one must not bet on every game. Indeed, there is always a probability of being wrong when judging a tennis match. Here we established a model which enables the gambler to bet on an almost sure game. The prediction capacity of our model is of 97% on average, meaning that when the gambler bets on a game, he is almost certain of being right. This is at the core of the proposed strategy.

5.1.1. Refresher on Betting

In order to understand the following study, we will refresh the very basics of sport betting. From now to the end of this study, we will talk about profit and not about gain. Indeed, we recall that,

$$\text{profit} = \text{gain} - \text{initial bet}$$

so the result of a 1 euro winning bet with an odd of 1.55 is equivalent to a profit of 0.55 euros (1.55-1). Then, odds are related to the probability of winning. Indeed the smaller the odd, the higher the chance of winning. Therefore, we have:

$$Odd_A = \frac{1}{\text{probability that A wins}}$$

Similarly, the implicit probability of winning from a bookmaker odd at 1.55 is computed as follows:

$$\text{probability that A wins} = \frac{1}{1.55} = 64.5\%$$

Owing to this fact, we now fully understand that the player with the lowest odd is the favorite player. Finally, in this second chapter, we will always make a 1 euro bet, on every single match first, and then on a selection of matches.

5.1.2. Basic Sports Betting Strategies

- **'Maximum Earning Strategy': our Upper-Bound Benchmark**

  Our first approach is not to build a real strategy, but to understand the environment in which we will play. How much profit can we make? How difficult is it? How much accuracy do we need to make profit in tennis sport betting? To estimate our potential maximum profit, we 'just' need to bet on the winner for each match. This will lead us to 100% of accuracy, but obviously this is really not a feasible strategy since it cannot be achieved in real life. It will only be done to delimit our upper bound benchmark. After establishing real strategies, we will be able to compare them to the maximum potential profit, that is to say to the 'Maximum earning strategy'.

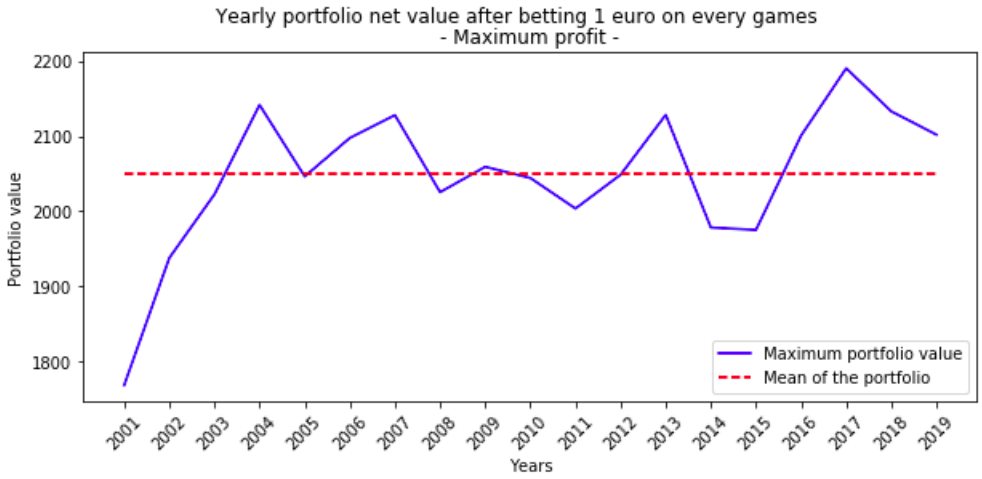

Figure 4: Yearly P&L of the 'Maximum Earning Strategy'

Here is the maximum profit, for each year, that can be won by an investor with 100% of accuracy. Our aim is now to define the most basic strategy and to see if this latter is profitable or not.

- **Is the 'Lowest Odd Strategy' Suitable for Investors?**

  Before building any complex strategies, we need to understand the most basic one. For each game, what happens if we bet on the best ranked player? One could believe that this is the most basic strategy because, theoretically, the better the player is ranked, the better he is. However, we decided that the "favorite player" is not always the best ranked player. Tennis lovers will understand this example. Let's suppose that Djokovic N. (currently World #1) is going to play against Nadal R. (currently World #2) at Roland Garros. Knowing that Nadal already won 12 times this tournament (93 wins for 2 losses) and only one title for Djokovic, would you say that the best ranked player (i.e. Djokovic) is going to be the favorite player, and thus have the lowest odd?
  Therefore, to find our basic strategy, we ask ourselves "What if we bet on the lowest odds?". This strategy is nothing else than following the bookmaker's opinion. Indeed, doing so will enable us to catch unobserved patterns: if a player said that he felt unwell before the game, this information will be priced and incorporated within the odds before the start; unlike with the 'best ranked player' bet.
  Our first step is now to find out how good is the 'lowest odd strategy'. Obviously, it must not be profitable because if it was, every gambler would just bet on the lowest odd of the bookmaker and make money against him.

  Here we build a portfolio where the only guideline is to bet on every match and to gamble the same amount of money, 1 euro, on the lowest odd of each match. The result of such a strategy from early 2001 to the end of 2019 ends up with 68.93% of accuracy! Not bad, right? Unfortunately, we ended up with a loss of roughly 3,400 euros. Indeed, we computed the yearly profit (or loss in this case) as well as the accuracy of this 'Lowest odd strategy' below.

  From Figures 5 and 6, one can easily see that accuracy is quite high every year. Globally, the higher the accuracy the lower the losses registered. However, even with 72% of accuracy, we are not making a profit... Therefore, we need to build our own strategy!

- **'No Bet' Strategy**

  Following this idea, we could try to implement a new category called: "No bet".

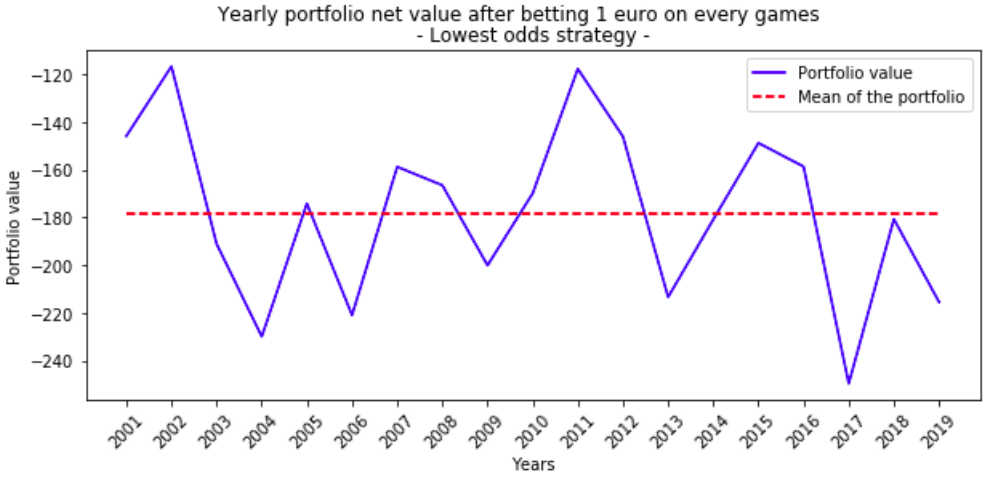

Figure 5: Yearly P&L of the 'Lowest odd strategy'

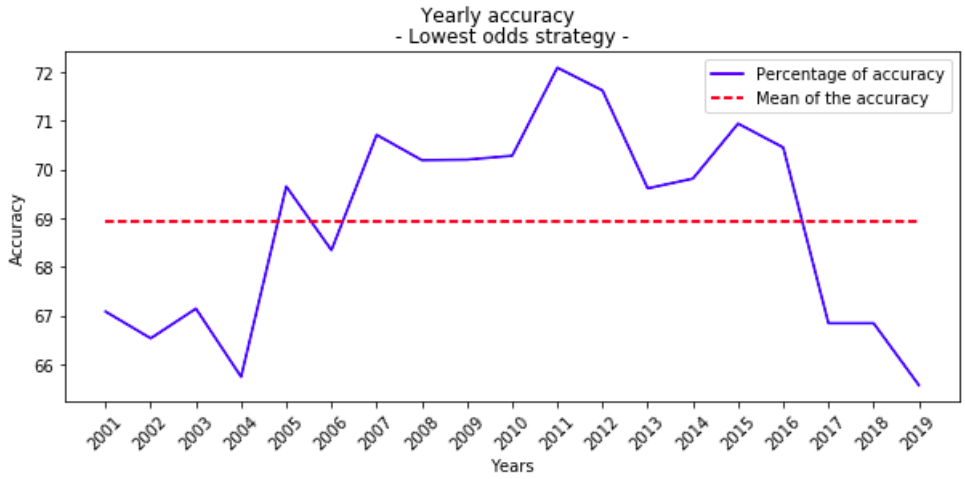

Figure 6: Yearly accuracy of the 'Lowest odd strategy'

This new output of our neural network prediction should display, neither player A nor B, but "No bet" if the algorithm is less confident than a given threshold (let's say 80%) that a given player will win. As we can get the probabilities returned by the 'predict()' function, we can introduce the 'No bet' output when the probability is between a lower-bound and an upper-bound. After some computations, we find out that the greatest accuracy (92.5%) is achieved when the lower threshold is 0.3 and the upper threshold is 0.7.

In other words, when the prediction returned by the neural network is less than

0.3 we state that the player A is going to win. If the prediction is bigger than 0.7 it means that the neural network predicts that the player B is going to win, and between this range we associated a 'No Bet' value.

This strategy leads us to take 6,820 bets between 2016 and 2019, on which we won 6,269 of them. However, we can't compare the portfolio value of this strategy with the previous one since we are not investing the same initial amount. Therefore, from now to the end of this study, we are going to talk in term of 'Return On Investment' (ROI) (Figure 7).

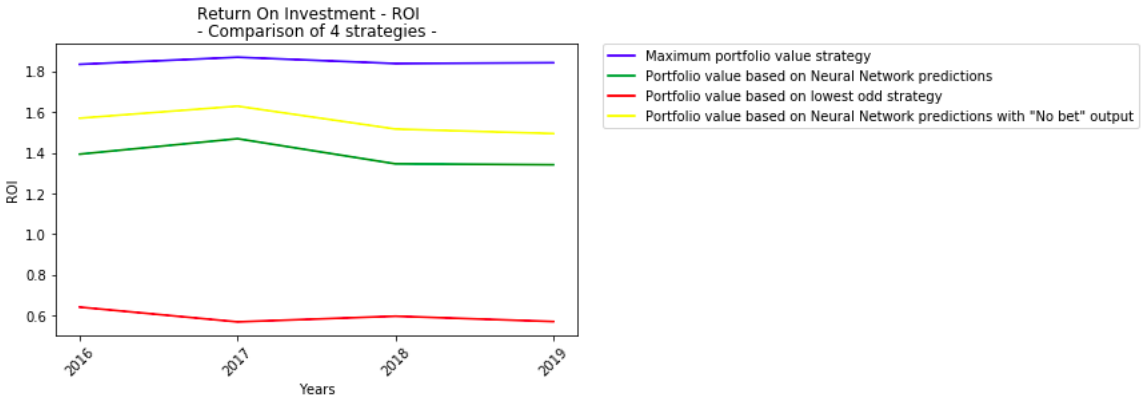

Figure 7: Return over Investment: Comparison of the 4 strategies

The striking result from the figure above, is that the 'No bet' strategy is better than the basics strategies as well as our first prediction based on neural network (in term of ROI). One can conclude that in a portfolio optimisation approach, we do not care about the total number of matches correctly predicted. We rather have good accuracy and this is why games selection is very important in sport betting : if you don't know, don't bet.

Looking at the figure enables us to make hypotheses. The first one is that matches which are the easiest to predict are those for which we predict the result with a high probability of accuracy (i.e. the favorite player and the outsider are clearly well separated). The problem is that the more 'favorite' a player is, the lower his odd of winning. Therefore, the trade-off between accuracy and risk is now our main problem. For instance, would you rather bet on a 1.05 odd with 95%

of accuracy, or on a 1.4 odd with 60% of accuracy? According to you, which strategy will be the more profitable one ?

5.1.3. Trade-Off Between Accuracy and Return

The biggest problem of our neural network is that it sometimes have confidence in a wrong result. We do not think this can be fixed since we acknowledge that surprising results definitely happen. Indeed, it is the beauty of the sport, when the outsider beats the favorite player! Sometimes, our prediction is not very strong, meaning that the neural network is somewhat undecided. For example, the very last predicted value of 2019 gives 0.59, when the result should be closer to 1 since player B won. In this case, its worth taking the risk since the odds were 1.80 and 2.00. On the other hand, the neural network can return predictions between 0.20 and 0.40 for odds between 1.1 and 1.4. In that case, we are not highly sure that player A is going to win and the odd is not very attractive...

In a nutshell, the legitimate question is: 'Should we always bet within this range of prediction and unattractive odd?'. Indeed, each time we bet on an attractive odd (let's say 1.4 or more) with a low confidence, we can significantly increase our profit but we will definitely decrease our accuracy and thus have more losing bets than before which can on the long term, reduce our profit as well.

Even if we supposed the opposite, it could be an obvious answer for many of us. The first hypothesis is that we definitely should not bet on very low odds since the risk is to big for the reward. But how to define the 'low-odd' threshold under which we are not gambling? To answer this question, we will focus on the trade-off between accuracy and return, or more precisely, on trying to optimise a portfolio when having non persistent prediction rates or non attractive odds. To do so, we will create three strategies that sport gamblers can be familiar with.

1. **First strategy: Usual Gambler Strategy** The first strategy represents a gambler that relies on the neural network prediction with the 'No bet' output if the returned probability is below 0.4 or above 0.6. This gambler always bets when he is confident, or quite confident if he is facing an appealing odd. In a neural network approach, it would be defined as:
    - I always bet when the prediction of the neural network is below 0.15 or above 0.85, no matter the odd.

- When the prediction is between 0.15 and 0.4 or between 0.6 and 0.85, I bet only if the odd is above 1.4 (*because 1.4 was our average odd following the "No bet" strategy*).
- I do not bet when the prediction is between 0.4 and 0.6.

After implementing this strategy, we ended up with 7,633 matches well predicted out of 8,788 (87% accuracy). Depending on the year, our ROI is between 1.42 and 1.55 while our accuracy varies between 85% and 88%.

2. **Second Strategy: Usual Gambler Strategy** The next strategy represents a gambler that always want to be rewarded for taking a risk. Even if he is confident, we wants to be rewarded. This gambler will definitely avoid betting on a 1.05 odd even with 95% of probability and will ask for at least 1.2. In a neural network approach, it would be defined as:
   - I bet when the prediction of the neural network is below 0.15 or above 0.85, only if the odd is above 1.2.
   - I bet when the prediction is between 0.15 and 0.4 or between 0.6 and 0.85, only if the odd is above 1.4.
   - I do not bet when the prediction is between 0.4 and 0.6.

   After implementing this strategy, we ended up with 5,811 matches well predicted out of 6,918 (84% accuracy). Depending on the year, our ROI is between 1.50 and 1.70 while our accuracy varies between 82% and 85%.
3. **Third Strategy: Usual Gambler Strategy** The last strategy represents a gambler that seeks reward no matter the risk. This gambler needs high reward, nearly regardless of the prediction of the neural network. He will always seek for the highest odds even if the neural network confidence interval is low. In a neural network approach, it would be defined as:
   - I bet when the prediction of the neural network is below 0.4 or above 0.6, only if the odd is above 1.4.
   - I do not bet when the prediction is between 0.4 and 0.6.

   After implementing this strategy, we ended up with 4,066 matches well predicted out of 5,101 (80% accuracy). Depending on the year, our ROI is between 1.61 and 1.90 while our accuracy varies between 79% and 81%.

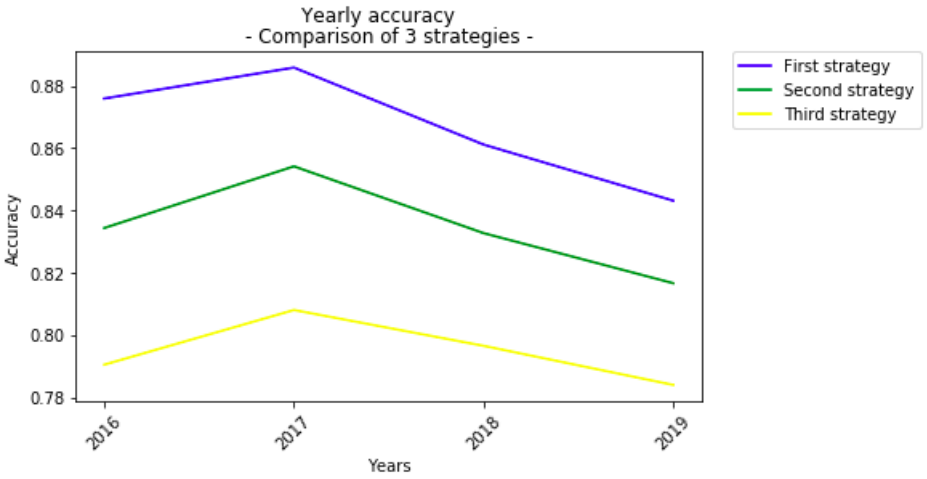

Figure 8: Model accuracy: Comparison of the 3 strategies

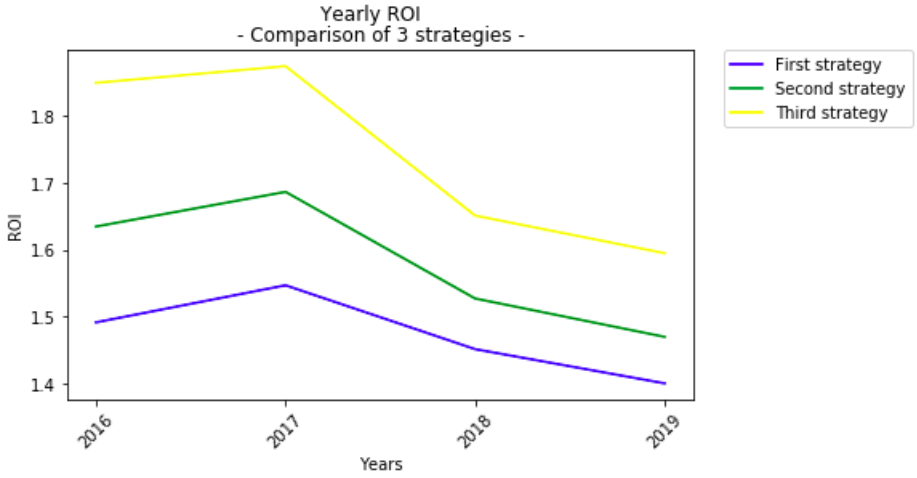

Figure 9: Return over Investment: Comparison of the 3 strategies

In Figures 8 and 9 , it is easy to see the trade-off between accuracy and return. Indeed, with our data and these three specific strategies, we obtained the greatest ROI with the strategy having the lowest accuracy. This result corresponds to the third strategy which is the one where the gambler bets only if the odd is attractive, even if the prediction is uncertain. (One must note that we are not betting randomly when the odds are appealing since we follow the neural network's predictions which are, on average, pretty accurate.)

Therefore, one conclusion that can be made is that we do not care about having 100% of accuracy to optimise a portfolio. Indeed, we reached the greatest ROI with

an average of 81% of accuracy even if other strategies were way more accurate. This is why we believe that it is not worth betting on very low odds (let's say 1.05) even if we have a really high probability of success. Instead, we rather bet on attractive odds (let's say 1.4 or more) with lower accuracy. In other words, we accept to lose nearly 8% of accuracy to gain over 20% of ROI (according to our dataset). Since we know that our neural network is accurate, we will still end up with a decent accuracy that enables us to make more profit.

5.1.4. Finding the Optimal Strategy: ROI Maximisation

For now, we compared the return on investment of betting on the lowest odd on each game, as well as three gambler strategies, and we finished by stating the existence of an inverse relation between accuracy and ROI. However, one investor is not willing to replicate a pre-determined gambler strategy. He will rather look for the optimal strategy, given its database and its features engineering work.

There is numerous and complex libraries available in Python to find the best function's parameters. Some of them are as easy as testing all the parameters and keeping the best result. Here, we will adopt a similar approach; running an optimization function regarding the thresholds that delimited the confidence of the neural network's output, with a 0.01 precision, and the corresponding odds to know when betting. The results are crystal clear, since after hundreds of simulations the four threshold are: [0, 0, 0.95, 1] and the corresponding odds are [1.0, 1.4]. Here, what a reader should understand is just the interpretation. The first striking result is that the first range, that corresponds to the interval in which we are allowed to bet for the win of player A, is nonexistent. Indeed, its lower bound is zero, as well as it's upper bound. It means that no matter the odd, an investor should not bet within this range.

Doing the same reasoning with the upper range (for player B) enable us to understand the pattern of the optimal strategy:

- I bet when the prediction of the neural network is above 0.95, only if the odd is above 1.4.
- I do not bet when the prediction is below 0.95.

Therefore, this optimal strategy can be seen as a strategy that detects misprices on player B's chances to win from the bookmakers. Indeed, no bet should have an odd

of 1.4 or more, if we associated it a 95% confidence of winning. After implementing this strategy, we ended up with 940 matches well predicted out of 969 (97% accuracy). Depending on the year, our ROI is between 3.55 and 4.14 while our accuracy varies between 96% and 97%.

One can easily see that we hugely increase our accuracy as well as our ROI. But the best way to show it, is to compare our results with the ones of our previous strategies.

As we said, the striking result is the drastic increase of the accuracy, compared to our gambler strategies. If one want to go further into the details, he can see that we managed to keep the same accuracy in 2018 and 2019, while all the gambler strategies showed a decrease in their accuracy for the same period (Figure 10).

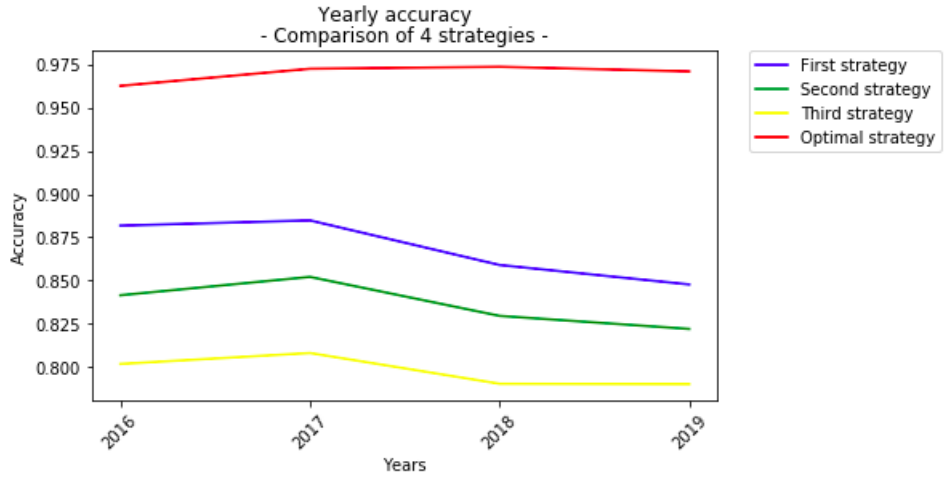

Figure 10: Evolution of the model accuracy for the studied strategies

Finally, if there is one thing to retains about sport strategies is that we should always think in term of comparable strategies. Indeed, we could believe that the more we bet, the more we will increase our portfolio value. While this statement can be true, the reality is biased. As a matter of fact, after some thoughts, one will easily understand that, in addition to investing more money in betting, the investor will face more losses as well.

The time has come to think in terms of return on investment to fully understand the optimal strategy's potential. As the figure below shows, the optimal strategy is clearly better than the other strategies regarding its ROI. This strategy implies that an investor invests on 969 matches over 4 years, *(i.e. nearly 1 bet on each trading days)*, and all of the bets must have an odd over 1.4 as well as a neural network prediction

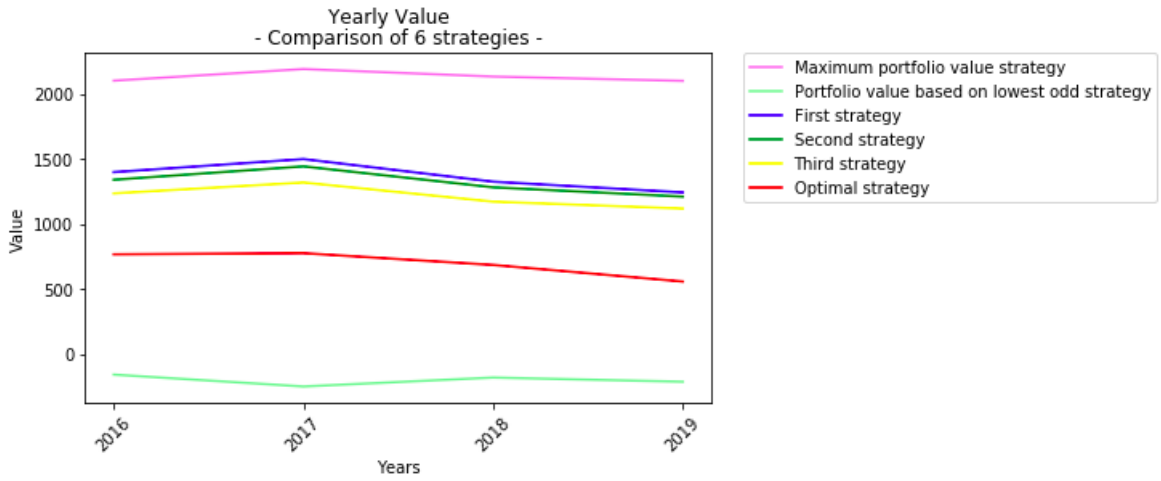

Figure 11: Evolution of the portfolio value for the studied strategies

over 0.95. This clearly shows us that a potential investor could detect some kind of mispricing from the bookmakers since following this strategy will lead to an average winning rate of 97% and an overall return on investment of 3.84 over 4 years (Figure 12).

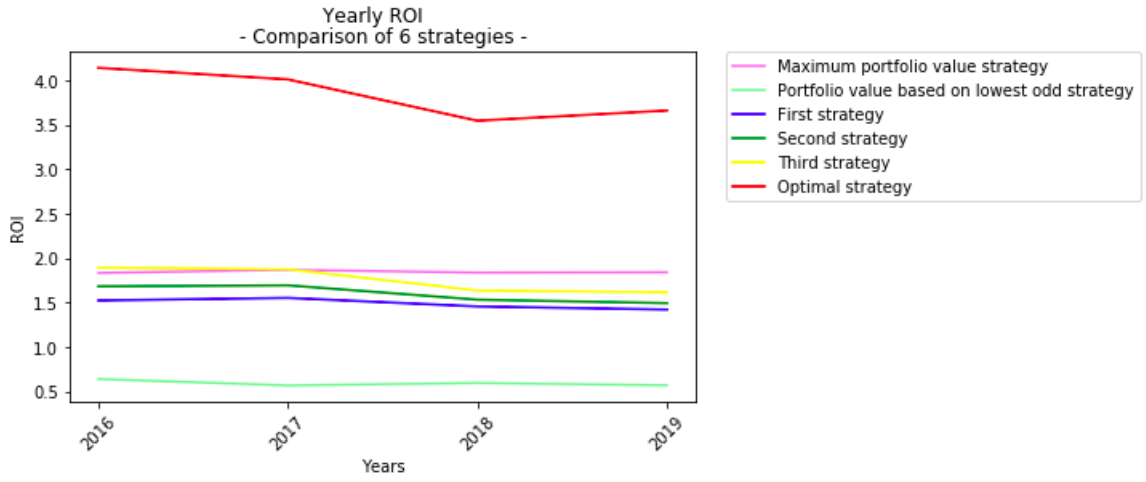

Figure 12: Evolution of the ROI for the studied strategies

6. Can Tennis Betting Break Into the Financial World?

In this section we will analyze the return on investment of the optimal strategy with a simple investment in a benchmark index such as the CAC40 between the 4^{th}

of January 2016 and the 30^{th} of December 2019. Using the results obtained in the previous section, the investor invested 1 euro on each of the 969 matches chosen by the algorithm on a horizon of 4 years. Of course, since there is a 4 years gap, there are discounting and accounting matters to discuss, making a 969 euros investment more worthy than the final profit made at the end of the 4 years. However, we will not take this into consideration since it will bring more complexity than clarity to the strategy we are trying to put in place. Indeed, we invest on a daily basis hence the accounting has to be made daily on 4 years. This means there are 969 calculations to do in order to get the accounting of each bet right. At the end, if we take a risk free rate as the reference accounting and discounting rate, the profit should be approximately the same with and without accounting or discounting since the rate is close to 0.

6.1. Is the Optimal Strategy Better Than an Investment in the CAC40?

After getting the data from Wahoo! finance, we computed the daily and yearly returns of the CAC40 index depicted in Figure 13 and Figure 14.

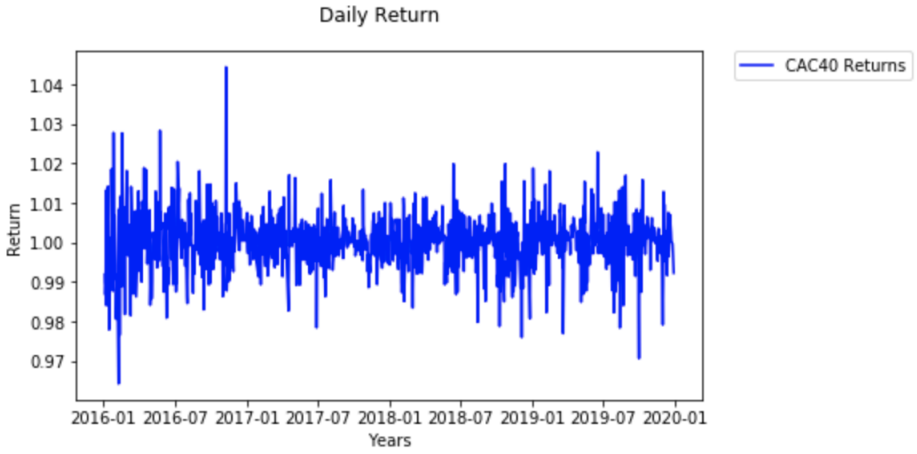

Figure 13: Daily Returns of the CAC40 Between January 4^{th}, 2016 and December 30^{th}, 2019

In Figure 14, we clearly see that the yearly return on investment never exceeded 130% over the 4 years time horizon. At first glance we would say that this simple investment in the CAC40 is much less profitable than an investment in the optimal strategy previously obtained. By taking the return on investment of the previous optimal strategy and comparing it to the simple CAC40 investment, we get the results depicted in Figure 15.

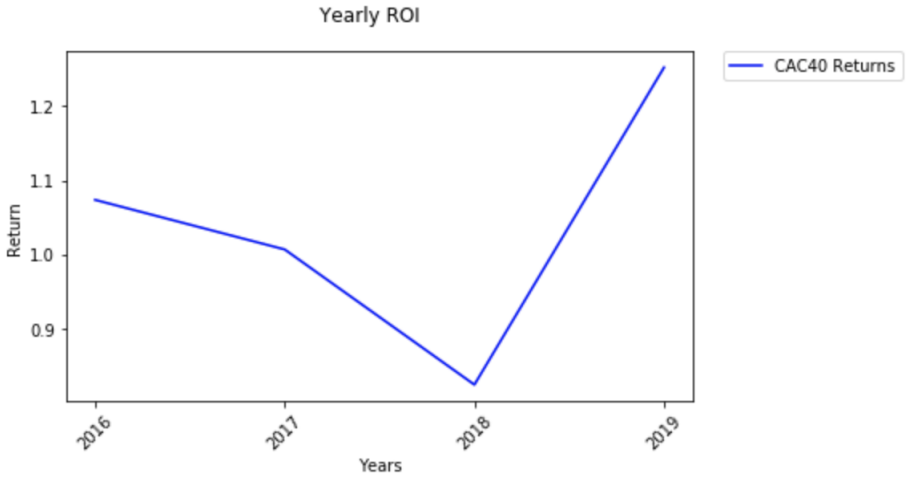

Figure 14: Yearly Return on Investment of the CAC40 Between 2016 and 2019

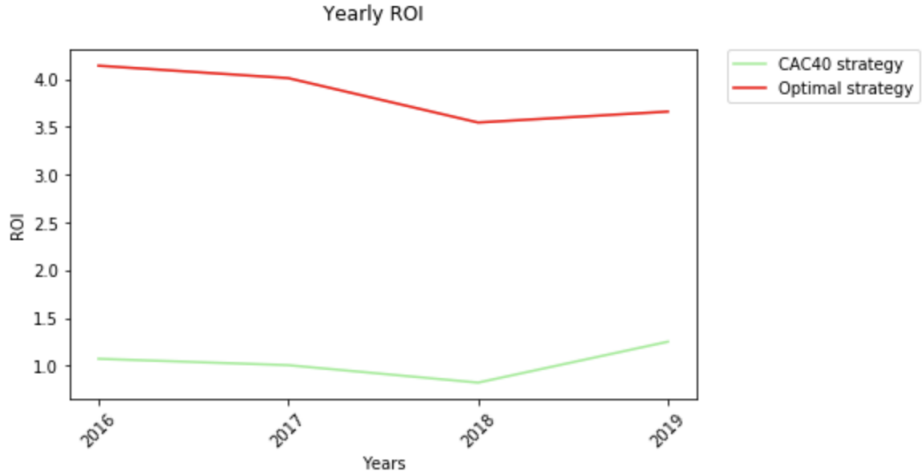

Figure 15: Yearly Return on Investment of the CAC40 and the Optimal Strategy Between 2016 and 2019

Now it is even clearer that the optimal strategy is much more profitable than the simple investment in the CAC40. Indeed, every year between 2016 and 2019, the return on investment of the optimal strategy is between 2.5 to 4 times that of the simple CAC40 investment.

Recall that until now we computed the yearly return on investment of our strategies, meaning that each year we started a new investment. Now we will consider both investment strategies from the beginning of 2016 to the end of 2019; we will think in terms of a portfolio investment. To compare them, we will first compute the value of

each investment strategy over the complete 4 years when the original investment is 969 euros. Then, we will compute the return on investment of each investment strategy over the complete 4 years as is it more relevant.

Despite the decrease of the CAC40 portfolio value in 2018, there is a clear difference between the values of both portfolios which can be observed in Figure 16.

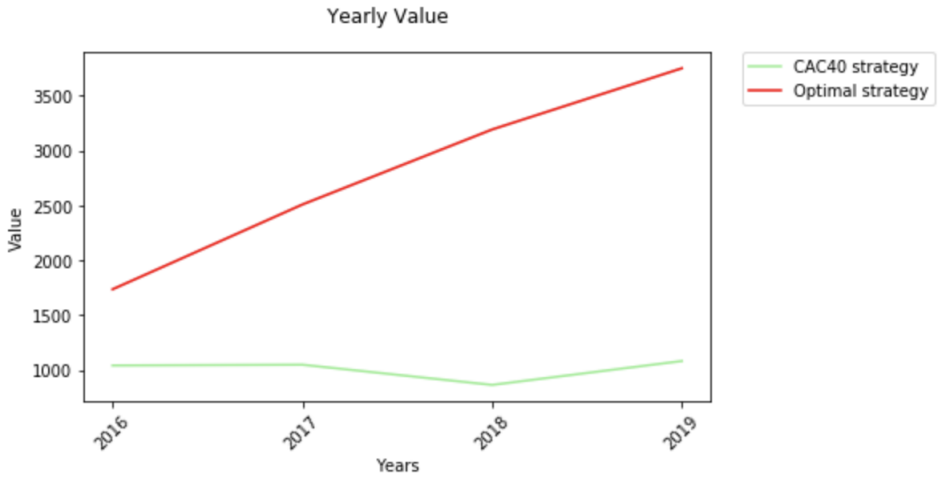

Figure 16: Yearly Value of the Optimal Strategy Portfolio and the Simple CAC40 Investment Portfolio Between 2016 and 2019

As one can see in Figure 16, the value of the portfolio using the optimal strategy is strictly increasing through time. The portfolio finished 2016 at a value of 1,750 euros and ended 2019 with a value of 3'750 euros. On the other side, the simple CAC40 investment finished at a value just above 1,050 euros, after experiencing ups and downs. Hence, the first portfolio has around 2'700 euros of added valued in comparison to the second portfolio at the end of 2019.

This optimal strategy shows once again that it is much better than a simple investment in the CAC40. The return on investments of both portfolios are also visible and easy to compute. In order to see the difference between both portfolios, we computed the value of the optimal strategy portfolio in terms of the simple CAC40 investment portfolio (SCIP). We get in 2016 a portfolio worth 167% of the SCIP, in 2017 a portfolio worth 239% of the SCIP, in 2018 a portfolio worth 370% of the SCIP and in 2019 a portfolio worth 347% of the SCIP. The value of the optimal strategy portfolio keeps increasing, but between 2018 and 2019 the SCIP increased too making the value of the high value portfolio seem smaller in 2019 than the previous year.

Finally, even if it is easy to observe from some of the graphs we have plotted in this paper, we will compute the volatility of both portfolios in order to have an idea of their riskiness. As expected, the optimal strategy portfolio has a daily volatility of 0.06% or a yearly volatility of 1%, whereas the CAC40 has a daily volatility of 0.74% or a yearly volatility of 11%.

To conclude, the volatility of our optimal strategy is 10 times smaller than that of a simple investment in a benchmark index, and the return on investment of our optimal strategy is 3.47 times higher than that of the simple investment in a benchmark index.

7. Conclusion

In this study, we understood that following the odds of bookmakers unables us to make profit. In our case, betting each match on the lowest odd from Bet365 during 10 years will create a net loss of 3.400 euros. Adding to this the sum of the implied probabilities of winning greater than 1, and the existence of a juice is confirmed.

To make profit with a long term investment, we first believed that the more accuracy we can have, the bigger the P&L. However, we realized that we shouldn't care much about accuracy but rather about ROI. Indeed, when recreating gambler strategies, we found out that the most accurate ones give the lowest ROI. This is clearly showing the importance of the trade-off between accuracy and return; or in others words; between risk and return.

Before discussing more of the strategies, we must say that the neural network could probably by optimized even more and the results could be even more relevant. Indeed, as we saw in part 4, the accuracy of our neural network is taken into consideration for the investment strategies of the investor. As a result the investor uses the accuracy to his own benefit. Still, in part 4, we were able to extract the highest performing strategy that maximizes the return on investment of an investor. We noticed that an investor must choose to invest only when the predicted value is between 0.95 and 1, and when the odd is higher than 1.4. We also did a backtest of this strategy and realized that its yearly return on investment is higher than 350% which is highly profitable. In the last section we decided to compare this strategy with a simple investment in a benchmark index such as the CAC40 on a 4 years time horizon between 2016 and 2019. We noticed that the CAC40 investment is not as profitable as the optimal strategy we found. Actually, the optimal strategy is 3.47 times more profitable than the simple

CAC40 investment, and is 10 times less risky.

To go further, even if this strategy seems very attractive, we believe there is room to some improvements. For instance, we deleted matches where one player retired due to an injury, saying that we can't predict it. In reality, predicting injuries is feasible but requires a more complex database. With our current database, the only thing that we could have done is to track a player's performance over the last months, supposing that the more matches he plays, the more he is willing to get injured. However, several players can play hundreds of matches without being injured, so we would have needed individual information that our database does not contain.

Then, we deleted the "Round" features saying that no matter the stage of the tournament, each player has the same wish to win. Even if it is true, a future work can introduce a behavioral component, saying that after a specific round (i.e. Quarter-Final, Semi-Final or Final) a player can feel what Ivan Hoffman called "The fear of winning", which will reduce his chances of winning.

We could also have taken the progression of a player into consideration more precisely. We updated the characteristics (% of winning matches, ranking, etc.) each year, but we could have done it after each match to be closer to what happens in reality. As we said, it was easy to compute statistics on a yearly basis when dealing with past data, but the way we computed statistics is not feasible in reality.

Finally, our very last thought would be to implement this strategy and compare the victory probabilities obtained through our strategy and through that of the bookmakers, and bet if we believe that a player should have a higher odd!

References

Barnett, T., Brown, A., Clarke, S.R., 2006. Developing a model that reflects outcomes of tennis matches, in: Proceedings of the 8th Australasian Conference on Mathematics and Computers in Sport, Coolangatta, Queensland, pp. 3–5.

Barnett, T., Clarke, S.R., 2002. Using microsoft excel to model a tennis match.

Bradley, R.A., Terry, M.E., 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. Biometrika 39, 324–345.

Cornman, A., Spellman, G., Wright, D., 2017. Machine learning for professional tennis match prediction and betting. Technical Report. Working Paper, Stanford University, December.

Del Corral, J., Prieto-Rodriguez, J., 2010. Are differences in ranks good predictors for grand slam tennis matches? International Journal of Forecasting 26, 551–563.

Dixon, M.J., Coles, S.G., 1997. Modelling association football scores and inefficiencies in the football betting market. Journal of the Royal Statistical Society: Series C (Applied Statistics) 46, 265–280.

Easton, S., Uylangco, K., 2010. Forecasting outcomes in tennis matches using within-match betting markets. International Journal of Forecasting 26, 564–575.

Glickman, M.E., 1999. Parameter estimation in large dynamic paired comparison experiments. Appl. Statist. 48, 377–394.

Gu, W., Saaty, T.L., 2019. Predicting the outcome of a tennis tournament: Based on both data and judgments. Journal of Systems Science and Systems Engineering 28, 317–343.

Ingram, M., 2019. A point-based bayesian hierarchical model to predict the outcome of tennis matches. Journal of Quantitative Analysis in Sports 15, 313–325.

Klaassen, F.J., Magnus, J.R., 2001. Are points in tennis independent and identically distributed? evidence from a dynamic binary panel data model. Journal of the American Statistical Association 96, 500–509. doi:10.1198/016214501753168217.

Klaassen, F.J., Magnus, J.R., 2003. Forecasting the winner of a tennis match. European Journal of Operational Research 148, 257–267.

Knottenbelt, W., Spanias, D., Madurska, A., 2012. A common opponent stochastic model for predicting the outcome of professional tennis matches. Computers and Mathematics with Applications 64, 3820–3827. doi:10.1016/j.camwa.2012.03.005.

Lisi, F., Zanella, G., 2013. Tennis betting: Can statistics beat bookmakers? Electronic Journal of Applied Statistical Analysis 00, 1–35. doi:10.1285/i20705948v10n3p790.

McHale, I., Morton, A., 2011. A bradley-terry type model for forecasting tennis match results. International Journal of Forecasting 27, 619–630.

O'Malley, J., 2008. Probability formulas and statistical analysis in tennis. Journal of Quantitative Analysis in Sports 4, 15–15. doi:10.2202/1559-0410.1100.

Sim, M.K., Choi, D.G., 2019. The winning probability of a game and the importance of points in tennis matches. Research Quarterly for Exercise and Sport , 1–12.

Sipko, M., 2015. Machine Learning for the Prediction of Professional Tennis Matches. Ph.D. thesis. Imperial College London.

Somboonphokkaphan, A., Phimoltares, S., Lursinsap, C., 2009. Tennis winner prediction based on time-series history with neural modeling 1, 18–20.

Wilkens, S., 2020. Sports prediction and betting models in the machine learning age: The case of tennis. Available at SSRN 3506302 .