University of Liège

Big Data Project

Review 4 - Improvement of the model

Master 1 in Data Science & Engineering

Authors :
Tom Crasset
Maxime Lamborelle
Antoine Louis

Professors :
G. Louppe
P. Geurts
B. Cornelusse

Academic year 2018-2019

1 Reminder on the predicting process

To predict the winner of the French Open, the idea is to use a match-by-match model. We would train a supervised machine learning model that will, for a particular match between two players, predict which player will have the most chance to win the match. Then, we would use this model to predict the winner of all possible matches between the 32 best seeds according to the current ATP ranking, giving a total of 496 matches to predict. Once the result of all these matches are known, the next step would consist in generating a lot of possible draws with respect to the well-known distribution of the seeds in a Gran Slam tournament. Predicting the winner of a given draw would then come down to progressively predicting each winner of each match by progressing in the draw from the first round to the final. Doing that for a large number of possible draws would finally allow us to output percentage values of seeds winning the tournament.

The supervised machine learning algorithm requires a set of labelled examples for training. In the context of tennis prediction, each training example corresponds to a single historical tennis match played between two players, and is composed of two elements :

— A vector of input features $\mathbf{x}$, composed of the characteristics of the match and the players, and also some statistical data about the players and their performances.

— An output value y, corresponding to the outcome of the match. Considering a match between two players, labelled as Player 1 and Player 2, the output value will be equal to 1 if Player 1 won, and 0 if Player 1 lost.

Data about previous matches were found on the website `www.tennis-data.co.uk`, that provides historical data on tennis matches of ATP tournaments from 2000 to 2018 in a structured form for a total of 50000 matches. Concerning the statistical data about the players, no explicit data set was found online, there rather were a multitude of websites that computed the statistics from the many matches. Using a small Python script, these data were scraped for approximately 1200 players from the website `www.ultimatetennisstatistics.com`. Table 1 shows some features of the collected data.

Player details	Match details	Statistics of player
Name	Tournament type	Percentage of first serve
Age	Location	Percentage of aces
Country of birth	Surface	Percentage of double faults
Current ATP rank	Result	Games per match
...	Date	Total break points won
	...	Total points won
		Fastest serve
		Average first serve speed
		...

Table 1 – Features considered to train the model

After some cleaning of our data set, we decided to train it with the Random Forest algorithm because it was a flexible and easy-to-use machine learning algorithm widely used for classification problems. Moreover, it is quick, simple and it reduces overfitting thanks to the bagging method. With some cross-validation, we computed the ideal maximal depth for the decision trees and got an an accuracy score on the testing set of 67%. We also got an idea of the most important features according to the algorithm, which were the percentage of matches won by the players, the game dominance, the percentage of matches played outdoors and the percentage of matches played against a TOP 100 player. Figure 1 shows the 100 more important features in the Random Forest algorithm.

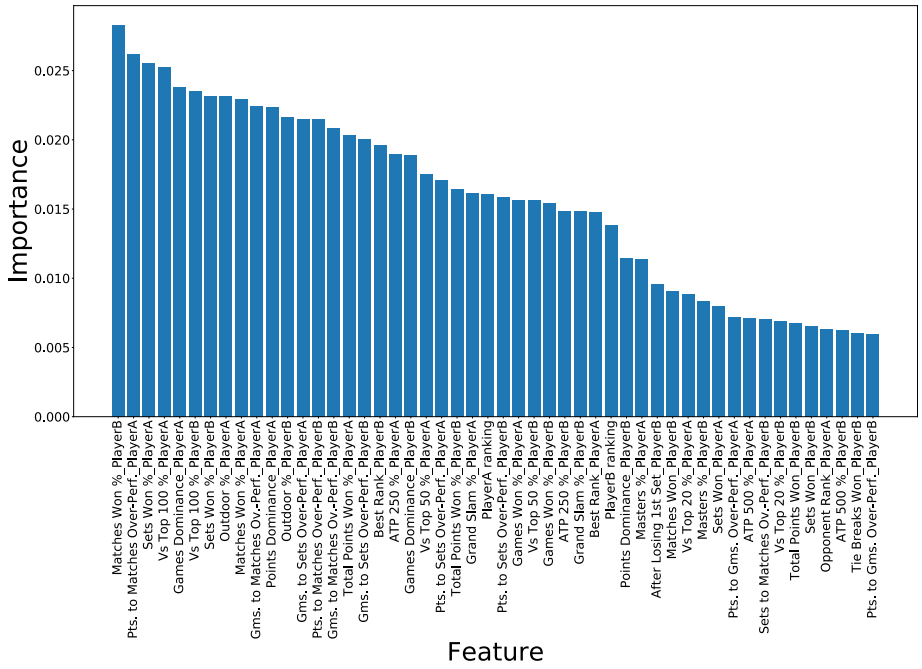

FIGURE 1 – Features importance in the Random Forest algorithm

However, our model was not conclusive at all. Indeed, the matrix of probabilities that we got for each player to win the tournament reveals that the Serbian Novak Djokovic would win the tournament with a probability of 100%. By analysing the predictions, we observed that the model predicted Novak Djokovic to win all its possible matches, leading then obviously to a probability of 100% to win the tournament. In practice, this scenario is not possible and some improvements are necessary.

2 Improvement of the feature representation

Previously, our prediction model considered the characteristics of both players independently of each other. As a consequence, we had two different values of the same variable for each statistical feature. The problem with this representation is that the model may assign, due to noise in the data, more importance to a feature of Player 1 than to the same feature for Player 2. This is problematic in the sense that this difference of importance for the same features could produce a different outcome prediction when the labels of the players are swapped (i.e., if Player 1 becomes Player 2 and vice-versa). Obviously, this is not what we expect. We would like our model to be symmetric in its outcome predictions.

A solution to this problem consists in building a new feature resulting in the difference of the two values representing the same variable. For example, considering the features $Rank_1$ and $Rank_2$, we build a new feature $Rank = Rank_1 - Rank_2$ that will be used to train our model. Moreover, using variable differences halves the number of features, which reduces the variance of the model and therefore helps to prevent overfitting.

3 Improvement of the model

3.1 Random Forest

Previously, we approached the tennis prediction problem as a binary classification problem : the output predictions of the Random Forest algorithm were either 1 if Player 1 won, or 0 if Player 1 lost. To get a more precise output, we computed the probability of each class using our new data set of difference variables and a Random Forest algorithm. The accuracy of this model turned out to be 66% based on a test on 20% of the data. Because this accuracy is quite small, we tried another model : a logistic regression.

3.2 Logistic Regression

The logistic regression model is a classification algorithm that also gives some measure of the certainty of an instance belonging to a class, which can be used as the match-winning probability. It is indeed commonly used in the literature for sport betting and is attractive in the case of tennis prediction for its speed of training, resistance to overfitting, and for directly returning a match-winning probability.

To reduce overfitting, we can pre-process the data by making a feature selection in order to only keep the most important ones. The feature selection can be done by recursive feature elimination (RFE), using a function provided by the `sklearn` library. This function takes the final number of features we want to keep as hyperparameter and returns the most important ones. To estimate this parameter, a randomized search, also provide by `sklearn`, can be used. The RFE will first fit the logistic regression on all the features. Then, at each iteration, the feature that is the least significant (the one with the smallest coefficient) is eliminated. Finally, we got an accuracy of 67.7% by keeping the best features which are :

Feature	Importance
Matches Won % Difference	10,6 %
Vs Top 100 % Difference	10,4 %
Pts. to Matches Over-Perf. Difference	8,9 %
Gms. to Matches Ov.-Perf. Difference	7,2 %
Outdoor % Difference	7,0 %
Sets Won % Difference	6,6 %
Total Points Won % Difference	6,2 %
Gms. to Sets Over-Perf. Difference	6,0 %
Games Dominance Difference	5,8 %
Points Dominance Difference	5,1%
ATP 250 % Difference	5,1%
Games Won % Difference	4,7 %
Vs Top 50 % Difference	4,5 %
Pts. to Sets Over-Perf. Difference	4,5 %
Best Rank Difference	3.4 %
Grand Slam % Difference	3,2 %

4 Predicting the winner

By analysing the probabilities that each seed has to win a match against another seed, we noticed that they are all very close to 50%. This can be explained by the fact that we only make predictions considering the best 32 players, for whom the characteristics are probably very close to each other. Our model, that was trained on more than 1100 players ranked from 400 to 1, will then have no difficulties to predict the winner between a player in the TOP 5 and a player ranked outside the TOP 200, but won't make a precise prediction on a match between two TOP 5 players. Even in practice, the result of such a match is more uncertain.

As a consequence in our Monte Carlo process, the probability distribution is close to uniform. Indeed, after a huge number of draws, each seed wins the tournament almost the same number of times resulting in a probability of winning for each one of them of 3,125%.

Finally, the predicted winner of the tournament is Cecchinato with 4.65% of chance of winning, followed by Basilashvili with 3.99%. The player that has the smallest probability of winning among the 32 seeds is Cilic with 2.2%.

5 Conclusion

Again, our model lacks precision and some further improvements need to be done. We need to find a way to better distinguish the best players. We could keep the logistic regression but use a kernel to model a non-linear relation in the input space. Another approach that can be taken is the popular field of Deep Learning. By reading some literature, we found that Somboonphokkaphan[1] trained a three-layer feed-forward ANN for match prediction with the backpropagation algorithm. Several different networks with different sets of input features were trained and compared. The best-performing network had 27 input nodes, representing features of both players and the match, and had an average accuracy of about 75% in predicting the outcomes of matches in the 2007 and 2008 Grand Slam tournaments. Therefore, this approach clearly deserves further investigation.

Références

[1] A. Somboonphokkaphan, S. Phimoltares, and C. Lursinsap. *Tennis Winner Prediction based on Time-Series History with Neural Modeling.* IMECS 2009 : International Multi-Conference of Engineers and Computer Scientists, Vols I and II, I :127–132, 2009.