University of Liège

Big Data Project

Review 3 - Data analysis

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 Choosing a technology

In order to make our data processing, some tools are needed. All data processing components were implemented in the `Python` programming language, as a result of its many packages for scientific computing, such as `NumPy` and `Pandas`, which have made the implementation brief and efficient.

Given the size of our training dataset and the chosen model, there was no need to use a large-scale distributed system such as AWS or Google Cloud to train our model. So the data processing was made on a local machine with an Intel 8 core i5 and 8 GB of DDR4 RAM.

Making an analysis reproducible is crucial to get consistent results in practice. There are many causes that can lead to non-determinism, such as shuffling of the dataset, changes in ML frameworks or randomness in the machine learning model itself. To combat these, we chose to fix the size of our dataset and sort it before beginning to train, so if our model uses a fixed range of the dataset, the contents of this set will be consistent across runs. Moreover, we chose to use the `scikit-learn v0.20.1` and not deviate from this version. Finally, to prevent (most of the) randomness of our model, we fixed the most important hyperparameters such as *n_estimators*, *max_depth* and *random_state*.

To make a faster prediction, our fitted estimator can be saved in a `.pkl` format so that it can easily be loaded and reapplied to our data files.

2 Processing steps

Predicting the 2019 French Open's winner according to our match-by-match model requires some progressive steps. The first one is, given our dataset, to train a chosen model and test its accuracy. If the model is adequate, the next step consists in using it to predict the result of all possible matches between the 32 best seeds according to the ATP ranking. Once all these results are known, some possible actual draws can easily be generated with respect to the well-known distribution of the seeds. Then, predicting a winner in a draw comes down to progressively predicting each winner of each match by progressing in the draw until the final. Doing that for a given number of possible draws allows us to output some percentages for the potential winners.

2.1 Training the data

The first step was to choose a machine learning algorithm to train our data. The Random Forest algorithm was the one we decided to go with because it is a flexible and easy to use machine learning algorithm that is widely used for classification problems and often produces a good result. Moreover, it is quick, simple and it reduces overfitting thanks to the bagging method.

With some cross-validation, we could compute the ideal maximal depth for the decision trees in order to avoid overfitting. In Figure 1, one can see the evolution of the AUC scores for the training and the testing data, depending on the maximal depths. The AUC allows to see the degree of separability of the two classes, it tells how much the model is capable of distinguishing between classes.The higher the AUC, the better the model is at predicting

0's as 0's and 1's as 1's. As can be seen in the graph, the greater the maximal depth, the more the training data overfits. It seems that a maximal depth of 3 is ideal to avoid that phenomenon.

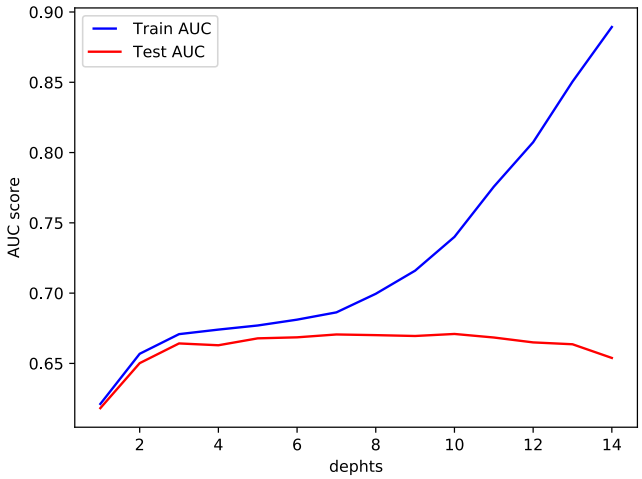

FIGURE 1 – AUC scores depending on maximal depths

Furthermore, Figure 2 shows the 100 more important features in the Random Forest algorithm. Among the 10 most important features stand 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 players. These features have an importance varying between 2% and 2,5%. Then, the next 20 features have an importance between 1,5% and 2% and finally, the rest of the 100 most important features have an importance of less than 1%.

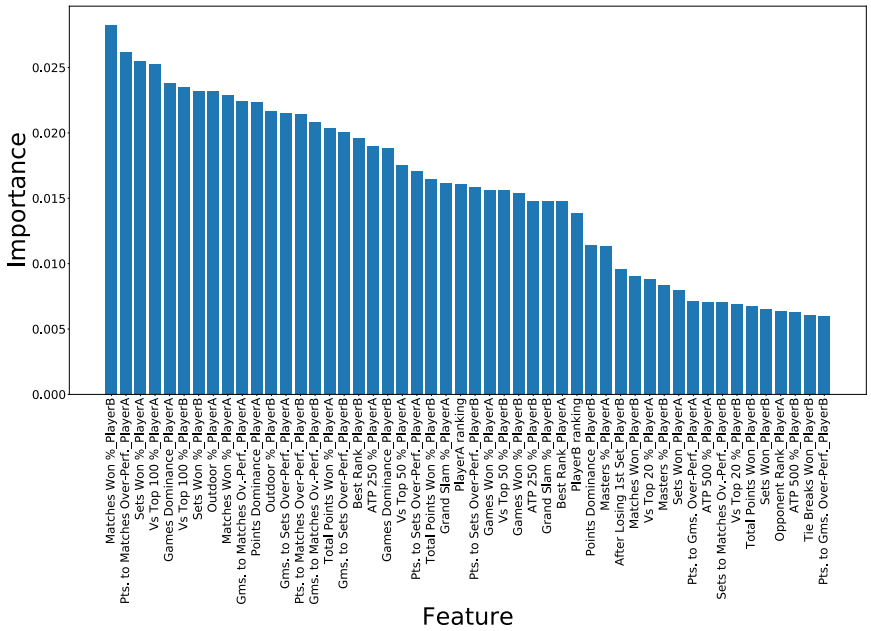

FIGURE 2 – Features importance in the Random Forest algorithm

2.2 Predicting matches results

For the prediction of matches, we decided to start from the third round, making the hypothesis that the 32 seeds will all arrive to this round, which is most likely. Moreover, if we had started our match predictions from the first round, where all 128 players were considered, it would have been very difficult to predict which players ranked outside the TOP 104 have passed the qualifications.

Then, the problem comes down to predicting all possible matches of the third round considering the 32 seeds. The number of matches results to predict is simply a combination without repetition of 2 players among 32, that is

$$C_{32}^{2} = \frac{32!}{2!30!} = \frac{32.31}{2} = 496 \text{ matches}$$

The predicted results of these matches were placed in a .csv file of three columns : id of player A, id of player B and id of the winner, where the id corresponds to the current ranking of the player.

2.3 Predicting a winner

Once the results of all possible matches among the 32 seeds are known, we use a Monte Carlo algorithm to predict the probabilities for each of the 32 players to win the tournament.

First we need to generate draws. The draws are generated by recursion with respect to the distribution of the seeds in the third round, that is the seeds 1 to 8 face one of the seeds 25 to 32, drawn by lot, and the seeds 9 to 16 face one of the seeds 17 to 24, drawn by lot. Moreover, the seeds 1 and 2 can only meet in final and they theoretically face the seeds 3 or 4 (according to the draw) in the semifinals.

Once the draws are generated, a recursive algorithm, such as the merge sort algorithm, is used to predict the winner of each part of the sub-draw and outputs the winner of the total draw. Then, we repeat these operations a large amount of times (10000 times) and count the number of times each player won the tournament. Monte Carlo theory tells us that if we normalised these results, we would obtain the probabilities for each player to win the tournament. The winner is the player with the highest probability.

3 Analysing the results

Once the model has been computed, we eventually get a matrix of probabilities for the players to win the tournament. The results were surprising because only one player has a non-zero value in the matrix, **Novak Djokovic**, with a probability to win the tournament of 100%.

We first thought that it was an error in the code of tournament prediction but when we analysed the results of the machine learning model, we observed that the model predicted the player with the ID 1 (Novak Djokovic) to win all its possible matches. So relating to that, it is not surprising that he wins the tournament in all cases.

3.1 Testing model on previous tournament

To test our model, we decided to try to predict the winner of previous tournaments. Unfortunately, none of our predictions were correct. The model always returns only one winner with a probability of 100%. Here are the results :

— Winner of 2016 : **Kei Nishikori**. He actually lost in the round 16th.
— Winner of 2017 : We were missing the 32th seed in our data (Mischa Zverev) so we were unable to predict the winner of the tournament.
— Winner of 2018 : **David Goffin**. He actually lost in the round 16th.

4 Further improvements

As seen in the previous section, our model is not optimal because no player can win the tournament with a probability of 100%. Therefore, some improvements could be made in the future.

4.1 Machine Learning model

The first way to improve the model is to change the machine learning model. Here, we used a random forest which is a Classifier. The model predicts only if the player wins or loses the game. But some nuance can be brought in. If a logistic regression model is used, the model could predict the probability of a player to win a game. This probability could be used in the prediction of the tournament winner by taking randomly the winner of each game according to this probability.

4.2 Data

We already use a large data set, so collecting more data is not our priority. But in this model, the data from 4 years ago have the same weight as the data from the current year although the former is much more relevant to predicting the winner. Some other features could then be weighed in the same way.

4.3 Draws

In the current model, we only take the 32 seeds of the tournament. In further versions of the model, more players could be taken into account, and the draws could start at the beginning of the tournament. However, the difficulty to generate the draws of deeper round and predict which player will come out of the qualifications makes this task very difficult.

5 Conclusion

Here is presented the first version of our model. Lots of improvements have to be done to raise the accuracy of this model but the general approach can give good results.