Web Intelligence Project Documentation

Ioan VLad Luca

January 2020

1 Introduction

The project was modeled to complete the following task: predict the winner of Australian Open 2020. In lack of data in regards with the Australian Open 2020 match schema the 2019 schema was used.

1.1 Components

The project has 3 major components:

- tennis folder: the 2019.xlxs suffered some changes;
- main.ipynb: containing the prediction models and simulation;
- utils.py: contains some functions used for the data processing part

1.2 General Idea

The general idea is to used data from all the tennis competitions that took place from 2014 to 2019(until the Australian Open) and to fit them in 3 models:

- LogisticalRegression;
- RandomForestClassifier;
- k-NN

and chose the one with the best score to simulate the matches for Australian Open 2019.

1.3 Initial preparation

Before the first run of the main file the BASE_PATH should be set to the path where the tennis folder is located

2 Analysis of the code

In this section I will discuss the steps I followed for the creation of the program.

2.1 Data Handling

The first step is to prepare the data for the models by choosing the X and Y parts of the model. Initially the data set was organised under the winner/loser format, but I considered the P1/P2 format and an additional column(P1_won) that can signal if P1 or P2 won. The P1_won is going to be used as the Y for the models.

The X of the models will consists of some initial features found initial in the data set plus two custom features. The already existing features used are:

- Tournament: the match tournament;
- Court: the match place(outdoor/indoor);
- Surface: the court surface;
- Round: the hierarchical round;
- Best of: number of played sets;
- Series: the series of the match;
- P1Rank: the rank of P1 at the time of the match;
- P2Rank: the rank of P2 at the time of the match;
- P1Pts: the number of pints for P1 at the time of the match;
- P2Pts: the number of pints for P2 at the time of the match;
- AvgP1 betting score for P1;
- AvgP2 betting score for P2.

The custom features added are:

- P1_Experience;
- P2_Experience;
- P2_W/L;
- P2_W/L;

The experience is represented by the number of matches played by the player until the current match. The W/L ratio is represented by: $\frac{Nr\_Loses}{Nr\_Wins} * 100$ where Nr_Loses is the number of loses the players has until the match point and Nr_Wins the total number of wins until the match date.

Implementation: All the functions that deals with the organization of the data are found in utils.py. The main idea is that the Winner / Loser columns were changed in P1/P2 columns. After this the P1 column contained all the time the winner, so A a P1_won column was created with pseudo random values chosen of 1 and 0. The data frame was parsed and every time the P1_won was 0 the P1 and P2 were switched, so the data was keep correct. NaN values were replace with the mean.

Factorization: Each non-numerical feature was factorized using the pandas function: factorize().

2.2 Models

For the prediction of data 3 models were implemented using the sklearn lib and compared :

- LogisticalRegression;
- RandomForestClassifier;
- k-NN.

2.2.1 LogisticalRegression

In regards with this model only one parameter was set custom: max_iter, because the model was reaching the max limit on a data frame of this dimension.

2.2.2 RandomForestClassifier

For the Random Forest a comparison was did in regards with the criterion. The gini and entropy(Informational gain) criterion were compared and the one which gave the best score was used.

2.2.3 k-NN

In this case the only parameter that needed choosing was the number of k. For this task k was assigned $\sqrt{n}$ where n is the number of total elements in the training set.

2.2.4 Training/Test set

The training and test set were chosed with the skllearn function: $train\_test\_split()$ and the test set is set at 0.25 of the training test.

2.2.5 Comparison

The RandomForestClassifier seems to be the one with the highest score of 0.82 (approximately), the LogisticalRegression is around: 0.65 on the test set and the k-NN at: 0.64.

2.3 The score of the models on the 2019 data

The 2019 Austral Open data was also used as an experiment as a training data, although this is consider the production data too. I did this to see the differences between a randomly chosen test set and a test set that is represented as future events. I observed that the Random Forest and the k-NN performed similar on the test data and on the Australian Open 2019 data. The Logical regression had a boost in score of: 0.05 (0.65 on test, 0.70 on Australian Open).

2.4 The simulation of the matches

The simulation of the matches was done in the following way:

- the best classifier was chose
- a result list was created with the 1st round schema as the first entry
- for each round simulate the result and merge lines 2 by 2
- append the new list in the result

The hierarchical scheme of the matches can be seen by changing the index of the results list (where 0 is 1st round and 6th round is the final)

3 Conclusion

To put it in a nutshell the Random Forest has a better performance of a data set this big. The Random Forest score can be improved by adding new features and by analysing the importance of the curent ones.