Tennis Prediction Project Documentation

Darius-George Oanea

October 2019 - 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 2 major components:

- tennis folder: the 2019.xlxs suffered some changes;
- Project.ipynb: containing the prediction models and simulation;

1.2 General Idea

The general idea is to used data from all the tennis competitions that took place from 2015 to 2019(until the Australian Open) and to fit them in 6 models:

- Linear Regression;
- Random Forest Classifier;
- k-NN
- Decision Tree
- Ada Boost
- Naive Bayes

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

If we define the outcome of a tennis match as the "Winner's name", the problem would be a multi-classification task with an output taking labels belonging to the names of all players. This approach is not realistic since for each match we have just two players and the winner is among them. One way to explore that is to consider two additional features 'player1' and 'player2' to which we randomly assign the match winner and loser, the target variable would belong to the couple (0,1) and the problem would be binary classification.

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;
- 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 points for P1 at the time of the match;
- P2Pts: the number of points for P2 at the time of the match;
- p1vsp2: head-to-head wins variable : count of all times when player1 beated player2 minus the count of all times when player2 beated player1 before the year of the match.
- wp1 vs wp2: the victories are also weighted with the number of sets played
- RankDiff: P1Rank - P2Rank
- PtsDiff: P1Points - P2Points
- WeightedRndDiff: rounds victories rate per player and per round type
- RoundDiff: rounds victories rate per player and per round type taking in consideration importance of the round

The custom features added are:

- p1vsp2;
- wp1 vs wp2;
- PtsDiff;
- WeightedRndDiff;
- RoundDiff;

Player1 Vs Player2 Head-to-head wins variable : count of all times when player1 beated player2 minus the count of all times when player2 beated player1 before the year of the match.

Rank and Points differences (Player1 - Player2) The two features we're going to introduce are simply the difference between players performances (Points and Ranks).

Rounds Difference

Weights to be attributed to victories rate per round The weights have ascending values (importance of rounds)

WeightedRndDiff Depends only on the the two players RoundDiff Depends on the two players + the round of the match

Implementation: All the functions that deals with the organization of the data are found at the beginning of the notebook.

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

2.2 Models

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

- Linear Regression;
- Random Forest Classifier;
- k-NN;
- Decision Tree;
- Ada Boost;
- Naive Bayes;

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.

2.3 AdaBoost

Ada boost score was 0.55 (the lowest). It was used with 50 estimators and a learning rate 1.

2.4 Decision Tree

Decision tree score was 0.63. It was used with max depth of 10. The gini criterion was used.

2.5 Naive Bayes

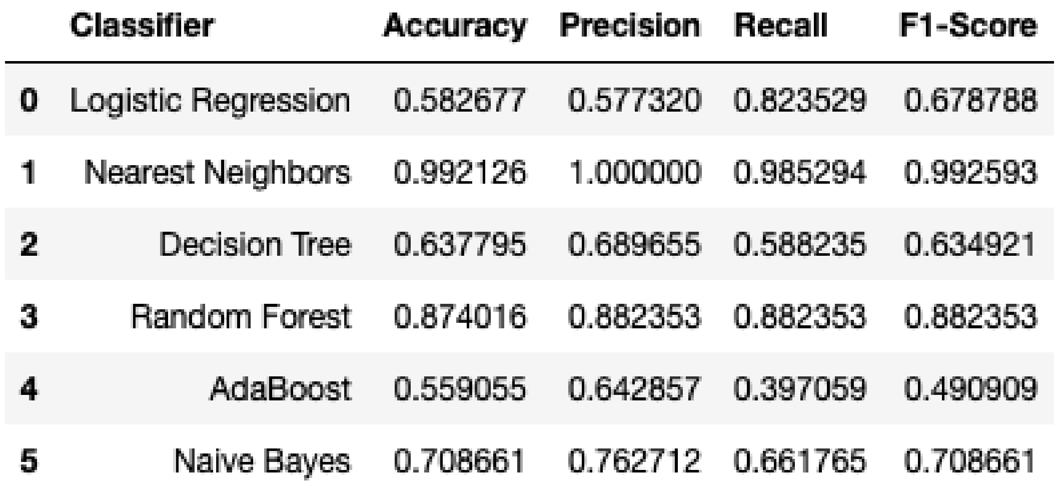
Its score was 0.70. Naive Bayes used by me was Gaussian Naive Bayes.

2.5.1 Comparison

The k-nn Algorithm seems to be the one with the highest score of 0.95 (approximately), the LogisticalRegression is around: 0.58, the DecisionTree: 0,63, the RF: 0.87, AdaBoost: 0.55, NaiveBayes: 0.70

3 Accuracy Measures

For the accuracy measures I have used Precision, Recall and F Score.



	Classifier	Accuracy	Precision	Recall	F1-Score
0	Logistic Regression	0.582677	0.577320	0.823529	0.678788
1	Nearest Neighbors	0.992126	1.000000	0.985294	0.992593
2	Decision Tree	0.637795	0.689655	0.588235	0.634921
3	Random Forest	0.874016	0.882353	0.882353	0.882353
4	AdaBoost	0.559055	0.642857	0.397059	0.490909
5	Naive Bayes	0.708661	0.762712	0.661765	0.708661

3.1 RF Recursive Elimination

Also, for the RF, I've used feature ranking with recursive feature elimination. This assigns weights to features. The goal of recursive feature elimination is to select features by recursively considering smaller and smaller sets of features.

I have implemented also a recursive feature elimination with cross-validation.

4 Conclusion

The best accuracy is around 95 percent. From my point of view is pretty decent. The nature of the problem is not an ordinary ML task since it includes the Time factor that I did not explore using time series techniques. I have not used betting data because it's the output of other people models, and it does not make sense to use it and pretend having built a realistic ML Model.