Machine Learning and Data Mining project: Predicting the outcome of NBA games using home statistics

Bruno Bonaiuto Bolivar, Carlo De Nardin, Fatma Moustafa Sayed Mahmoud Sayed

Course of AA 2021-2022 - DSSC

Abstract

Basketball is one of the most popular sports in the US and in the world. Many sponsors invest their money in teams and players. With the enormous amount of data that exists, the task of predicting the winning team of an NBA game based on different statistics have interested many researchers in the field of machine learning. Also many fans have high hopes when their team plays at home and would even bet that their team will win given the "land" factor. In the course of the last few years it was found that different home statistics had an important impact on why a home team won with a 60 % probability.

1 Problem statement

The purpose of this project is to predict the winning team of an NBA game based on the match statistics of the home team. The home team is the team with the land factor since it plays in the home stadium with many fans. Instead the away team often comes from afar and is supported by few fans during the game. The input variables are samples composed by different game statistics: ranking, field goal percentage (FGP), field goal three point percentage (FG3P), free throws made (FTM), rebounds (REB), assists (AST), steals (STL) and the blocks (BLK). The output is a binary result {YES, NO} that specifies if the home team wins or not.

2 Assessment and performance indexes

Once we build our models using different techniques, it is crucial to evaluate their performance and check if these prediction were good or bad. The task of selecting the appropriate performance metrics to evaluate the models we built should suit the problem and the data. We are solving a **binary classification** problem and our dataset is slightly balanced with {YES = 3845, NO = 2715}. The indexes should take into consideration that the cost of false positives and false negatives are not different. In our problem, a wrong prediction in any way means, for example, losing betting money or change in sponsorship deals which are bot costly. The metrics that could be considered to evaluate our algorithms are:

capitalize the first letter

- **confusion matrix**: the positive class definition, in our analysis, is: home team not winning, since it is the class with the least values
- **accuracy**: intuitive performance metric and very useful one when the dataset is balanced or slightly balanced
- **AUC**: measure of classifier performance which does not bias on size of test

we should define any abbreviation>> Area Under ROC

We did not consider the sensitivity or specificity because for our problem ~~the cost of~~ a false prediction is costly whether it is a false positive or false negative.

3 Proposed solution

Since it is a binary classification problem, different models have been proposed:

capitalize first letter

solutions

- **trees**: Trees are one of the most common solution in a binary classification problem. It is fast to learn, readable and also intuitive.

capitalize all the letters because it is an abbreviation

- **svm**: Support Vector Machines offer a high flexibility since it uses the kernel functions to separate the learning observations.
- **knn**: K-Nearest Neighbor captures similarity to predict new data points based on the distance. It is useful when it is more important to obtain a high accuracy model instead a human-readable model.

4 Experimental evaluation

4.1 Data

In this project, 5 NBA seasons, taken from a Kaggle dataset [1], have been considered (2014 - 2018) since from 2019 to 2021 the home team winning probability decreases due to the covid-19 pandemic. The dataset used is composed from a selection combining 3 different datasets: games, games details and rankings. This allowed for a more broad overview of the impact of different statistics on our problem. Only the home team statistics were considered for the prediction of home team winning to avoid any correlation between the variables. The dataset was split in two parts randomly by setting a seed to perform reproducible splits: train set (80 %) and test set (20 %).

of variables

4.2 Procedure

Different models have been used to solve the problem, as described in the proposed solution.

4.2.1 Single tree and trees aggregation

The first model that was considered was a simple unpruned tree with the highest complexity possible. The the best comlexity parameter was taken to build a pruned tree. ~~This technique was chosen~~ to find the best single tree.
Since big trees have a high complexity and they do not generalize the data very well,

and

Two different aggregation techniques were applied: bagging and random forest. In both techniques the number of the trees were analyzed to choose the best model. Furthermore, for the random forest, the parameter of variables taken for each tree was chosen by default ($m = \sqrt{nofvariables}$). number of variables

4.2.2 Support Vector Machines

Support Vector Machines might be a less interpretable technique than trees; however, they are highly flexible. The train set, with all the variables included, was fitted with the default parameters: C-classification, radial kernel and cost = 1.

4.2.3 KNN

The data were normalized to compute the K-Nearest N neighbor. To obtain the best K, a sequence was done from from a range of the Euclidean formula.

4.3 Results and discussion

The models' results were obtained by training classifiers with the training data. For the prediction phase, the test data were used.

4.3.1 Single tree and trees aggregation

The unpruned single tree resulted in a tree with high complexity. By pruning the tree a better result was obtained. The comparison of the two trees was done with the accuracy given by the confusion matrix (unpruned tree = 0.72, pruned tree = 0.77).

we already have these numbers in the table so they r redundant

The results obtained with the bagging approach and the random forest approach were better than a single tree. The most important part was selecting the best number of trees to use to get the optimal classifier. By training the trees several times, through the use of test error and oob error, an approximate number of 80 to 100 trees were selected. The comparison of the bagging and random forest methods was done with the accuracy (bagging = 0.800, random forest = 0.808).

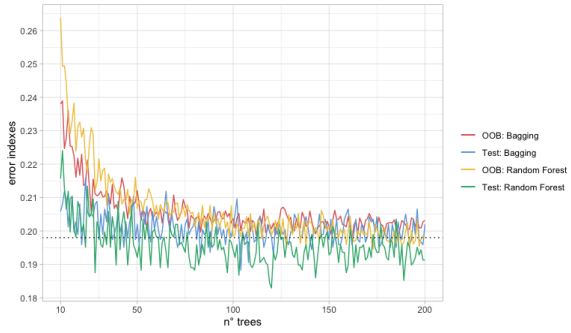

Figure 1: number of trees against test error and OOB error

y versus x not the opposite >> test error and OOB error for different number of trees

4.3.2 Support Vector Machines

Cross validation with 10 folds was done for different SVM kernels: linear, polynomial, radial and sigmoid and then the CV error was plotted against the cost. The CV error for the radial kernel decreases dramatically initially but tends to increase after a few iterations. The other kernels show a non-uniform pattern. This results maybe is given due the numbers of support vector which is 2433. High number of support vectors means that we have high tolerance, low variance and high bias.

add this line because it explains why having a high number of support vectors is not good

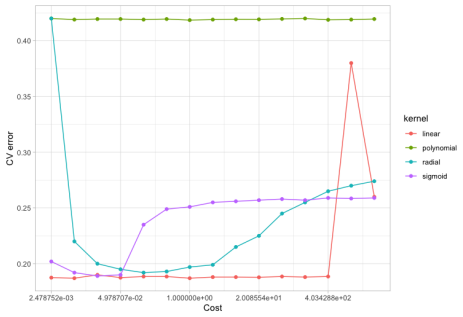

Figure 2: cost against CV error

CV error against different values of cost for the different kernels

4.3.3 K-Nearest Neighbor

The results obtained with the K-Nearest Neighbor methods by computing the best k across the test set was k equals to 65, which is a local estimate respecting the number of observation.

4.3.4 Results and conclusion

The performance indexes for the different techniques are shown in the below table

For the different indexes discussed in the assessment the results are shown below.

I think it would be better to remove sensitivity and specificity and not have anything bold

Index	Single tree	Pruned tree	Bagging	RF	SVM	KNN
Accuracy	0.719	0.779	0.8	0.808	**0.817**	0.811
Sensitivity	0.682	0.705	0.725	0.741	**0.744**	0.732
Specificity	0.744	0.828	0.85	0.854	**0.865**	0.864
AUC	0.79	0.767	0.788	0.797	**0.805**	0.801

In general, the results show that although we get a better accuracy and a higher AUC for the SVM it remains computationally complex. We have 2433 support vectors out of the 5248 observations. Thus, it is not the best solution for this problem with this dataset. The results obtained with the tree models reflect the general theory of these models. For the tree models, the random forest has turned out to be one of the best techniques. Also the bagging technique has shown results similar to the random forest, so it means that the variables chosen in the exploratory phase were strongly independents. The results obtained from K-Nearest Neighborhood was good although the computational effort was high with a large dataset.