**Using decision tree to predict the winner of NBA**

*Author: Yang Liu*

**Introduction:**

I will predict the winning team in the NBA by using Decision Tree. If you have watched the NBA, you may know that the two teams scored in the game. The points are very close, and it is hard to tell the winner. Sometimes the last minute can only be determined, so it is difficult to predict the winner. Many other sport games have similar situation, the expected big winner may be defeated by another team.

Many previous studies on the prediction of sports events have shown that the accuracy rate varies with different sports games, and the upper limit is between 70% and 80%.

**Instructions**

It is very simple to run this program. You can open Colab on google, upload the CSV dataset and let it read the .ipynb file.\_

**Documented program listings**

* leagues\_NBA\_2019\_games\_games.csv
* leagues\_NBA\_2018\_standings\_expanded-standings.csv
* Project2­­\_Yang\_Liu.ipynb

**Programming**

Firstly, I import the data by using pandas. The dataset can be got from <https://www.basketball-reference.com/leagues/NBA_2019_standings.html>.

图形用户界面

中度可信度描述已自动生成After I print that, I find there are several problems.

* Date is in string format, not date object
* No data in the first row
* Incomplete or incorrect header

图形用户界面, 文本, 应用程序

描述已自动生成Using .read\_csv and .column to clean it. Get the result:

To calculate the accuracy, I use 1 to mean the home team wins and 0 to mean the visitor team wins. Although the data set does not clearly give the wins and losses of each team, you can get it with a little compare.

文本, 应用程序

描述已自动生成

Then, create two features that can help us make predictions. They are the wins and losses of the two teams in the last game. Because winning the last game can roughly indicate that the team has a higher level.

图片包含 图形用户界面

描述已自动生成

The next step is to use Decision tree to predict the winner and use cross-validation to test.

文本

描述已自动生成

After we use Decision Tree, we could get a better results. Plus, we can still increase the accuracy by adding different features and using different algorithms.

The next step is feature engineering. To decide which feature we should use, we can think about those two questions.

* What kind of team has a higher performance?
* Who is the winner when two team matched in the last time?

For the first feature above, I want to create a feature named” whether home team is better compared to the visitor team”, and we also can use the data in 2018. We measure the ability by the rank in 2018.

文本

描述已自动生成

For the code above, firstly, we import the data. Secondly, we read the each row. By avoiding the situation that some team change the name, we need to replace the name and upload the rank and feature. Finally, we could use the decision tree and cross-validation to get the better result-60.2% accuracy. The result is good, but we can do better.

For the second feature, Although the team ranking helps predict (The top-ranked ones have a better chance of winning), sometimes the bottom-ranked teams can beat the top-ranked teams. There are many reasons. For example, some styles of the lower-ranked teams happen to hit the weak underbelly of the strong.

For the code below, we create a dictionary to save the data of last match winner. Then let it read each rows and sort it by Alphabet order for using the same key. Thirdly, to find the last winner and upload it for the reference when they match again. Finally, using Decision tree and the cross-validation to get the result-60.5% accuracy.

文本

描述已自动生成

To check whether the Decision tree can get the good model in a big data situation. I will add new team for the training.

Scikit-learn need to handle those feature firstly, so I will use LabelEncoder to convert the string to the int.

图示

中度可信度描述已自动生成

Exact all teams’ name and combine those together to build a matrix

文本

描述已自动生成

To avoid normalization problem, like algorithm thinks team1 and team2 are similar, team4 and team10 are different, but the fact it each team are different, I will use normalization to figure out this problems, such as the OneHotEncode method. Then we use Decision Tree and cross-validation to get the result-60.1%

图形用户界面, 文本, 应用程序

描述已自动生成

Accuracy is higher than the first accuracy, but is not good to last accuracy. This is may because after adding new feature, decision tree mishandles it. Hence, I decide to use another algorithm. I think this is data mining because we need to try different algorithms and features.

Now I will use random forest to predict the result. Random Forest is a good wat to avoid overfitting problem by creating more decision tree. About the bias, we use “mean” to reduce the variance for the high accuracy. The code shown below.

文本

描述已自动生成

The accuracy is 61.5%. Just by using the different algorithm, we could get a better accuracy.

If we use different subset selection, we may get a more efficient result.

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

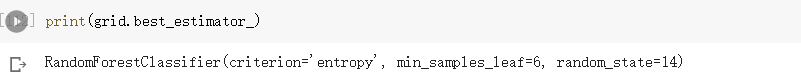
The accuracy is 62.9%

Then we can use the GridSearchCV to find the best parameters.

文本

描述已自动生成

We print the best parameter to see what value we used.



**Conclusion:**

We could find when we choose different parameter, the result is different. Sometimes the accuracy increases but sometimes it decrease. Hence, the main point is to decide which parameter we should use. Here is the one of the important problems for data mining.