CISC 372 Assignment 1 Report

Professor: Steven Ding

Name: Chengxu Li

Student Number: 20073324

Last Edited: 22th Feb

Thought process:

Firstly, I tried to understand the given script. For the first half part of the script, things went smoothly. They were basic python grammar and I learned them from CISC 121. Even though I was not familiar with pandas and numpy, but with the help of internet, I found related documentation source. For the latter part, I got a little confused by the Pipeline method that was implemented in the script. But as I read along, I figured that it's just a part of preprocessing which allows us to choose our own strategy to deal with the missing values.

The grid search surprised me. Although I knew that python was convenient, I didn't think it could be so convenient…☺. I can basically just write in numbers and it would test those numbers for me without any for loops.

After understanding the code, I have a simple idea of how to improve the performance. I was not satisfied with the feature selection. I have a passion of traveling myself, actually, I'm writing this report at Toronto with my parents from China now. I am living at an Airbnb house. Therefore, I knew at the first moment that the selected features were not complete at all. Important features like "reviews for location" and "number of bathrooms" are neglected. So I added some features that I consider important using common sense to the numerical feature list and categorical list. In total, there were 13 features. There are attributes that should never be included such as "Host_ID" and "Host_name", they would cause serious overfitting, unless there's a scientific study suggesting people tend to pay more for someone called "John".... I ran the code and I did see an improvement in accuracy and the extra wait time was fine, not too long. The accuracy was below 70% and

improvement was considered "tiny".

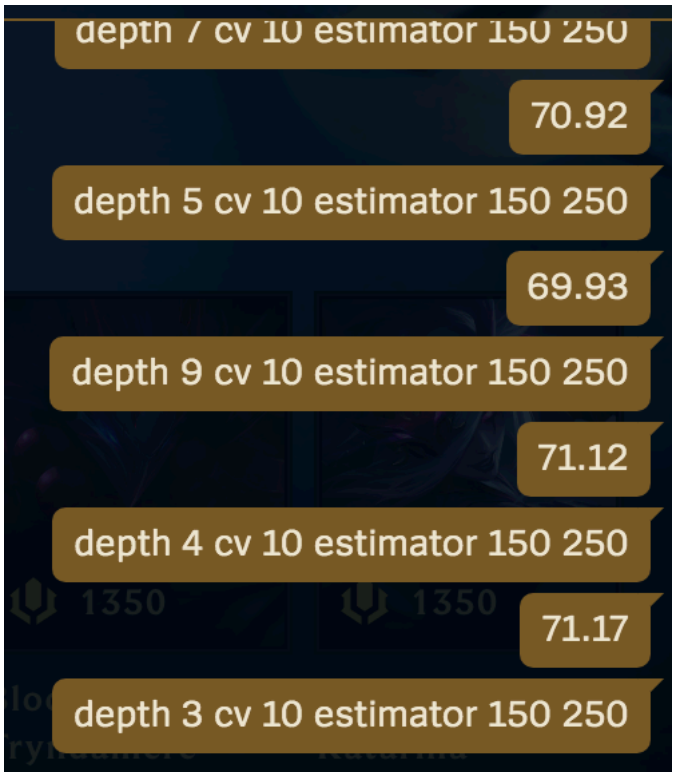
I then started looking at the model itself. XGBoosting seems like a decent model for this problem since the data has got quite a number of attributes and it's easy to implement with python. With the model being confirmed as a good model, I checked the configuration of the model, which contains "number of trees" and "tree depth". For the number of trees, it definitely should be increased. The original values were 50 and 100, which are relatively low even without considering the newly added features. Therefore, I turned it to a list containing following values: 100, 150, 200, 250, 300. As the trees are not computationally expensive because it's XGBoosting. Then I ran it again, turns out the accuracy had a bigger improvement in accuracy which was over 70% while the running time was significantly increased. I think it's OK for it to take a few minutes, we are training a model that can be used repeatedly in the future, so consuming a few extra minutes is acceptable.

After that, I decided to test the max_depth variable. I feel like 10-20 tree depth is not necessary since we only have 13 features in total. It might cause overfitting as well. With that being considered, I decided to put it as 5, 7, 9, 11, 15. This could also help us ignore some noises in dataset. After all, it's still considered as a small dataset. Unfortunately, we can't learn the best value for practical use so we can use that value for following tests to save time. My guess for the best depth value should be one of 5 or 7.

I ran the code again and this time, I waited for HALF AN HOUR. It still hurts me when I think of it.. Then I simplified the value from the list. For example for the number of trees, I only kept 150, 250 and 350. For the max depth value, I only kept 5, 7 and 9.

I see it as a "one-time effort" which can benefit us for a long-term use, I think it's OK

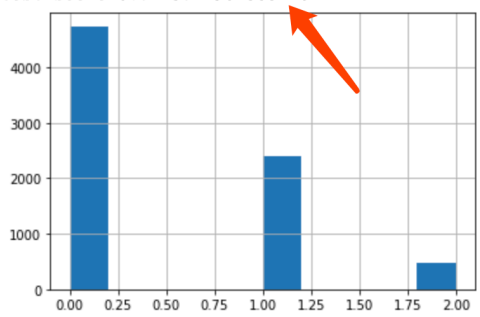
for it to take around 10 minutes to run. However, since this is an assignment, I didn't want professor to spend all day waiting for my code to run. I decided to find the most appropriate value for those variables using control variate method. I recorded those parameters and outcomes and I found that the best value for number of estimators is around 225 and maximum number of trees is 3!!!!! That was surprising right? Allowing a higher number of tree depth actually causes the accuracy to go down!!



There were also parts where I tried different things just out of curiosity, I decided to not put them into the report for the sake of simplicity.

The best accuracy I got in Colab was around 71.2 percent.

```
traning 7631
testing 7632
Fitting 10 folds for each of 1 candidates, totalling 10 fits
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  10 out of  10 | elapsed:   28.4s finished
/usr/local/lib/python3.6/dist-packages/sklearn/preprocessing/_label.py:235: DataConversionWarning: A
  y = column_or_1d(y, warn=True)
/usr/local/lib/python3.6/dist-packages/sklearn/preprocessing/_label.py:268: DataConversionWarning: A
  y = column_or_1d(y, warn=True)
best score 0.7115721902383126
```



Questions:

1. If we do not limit the number of trials, a student can try 100 times a day and use the best one while the code might not be actually that good. Good results might come out by odds.
2. Because you don't want us to spend too much time on the private leaderboard and forget about the report. Focusing on the leaderboard would lead us to focus on the result not the process. What you want is for us to concentrate on perfecting the process.
3. I used XGBoosting model. I made it more flexible by reducing the maximum tree depth. I made it less flexible by changing Cross-validation from 5 to 10.