A. The data I used is for flight delays from the Kaggle dataset for the time periods 2019 and 2020.

- 2019 : '/kaggle/input/flight-delay-prediction/Jan_2019_ontime.csv'
- 2020 : '/kaggle/input/flight-delay-prediction/Jan_2020_ontime.csv'

B. The variables I used for this particular problem are:

- DAY_OF_MONTH
- DAY_OF_WEEK
- DEP_DEL15
- DEP_TIME_BLK
- CANCELED
- DIVERTED
- DISTANCE_cat
- ARR_TIME_BLOCK

C. To solve the problem, I used the XGBoost algorithm.

D. I used resources from GCP in order to avoid long latency to run the model.

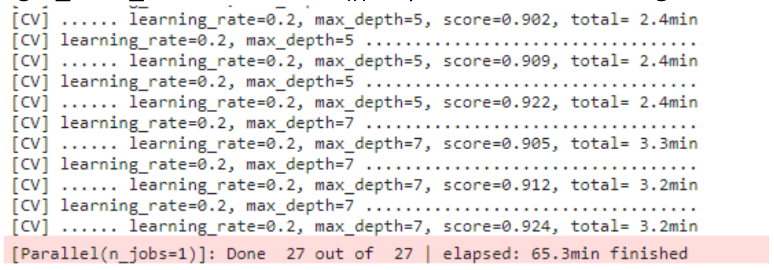
E. The results obtained are that:

We have used an XGBClassifier with a binary logistic argument
and parameters(max_depth and learning_rate).

I then defined a variable grid_search , which is defined by the GridSearchCV function containing arguments such as

- verbose=3, GridSearchCV should print updates during execution
- n_jobs= -1 (which means it will use all available processor cores) or (to see if the problem is due to multi-threading) n_jobs=1.

  We then use grid_search.fit(cat_vars_ohe_2019_final, target_2019_final.values.ravel()) to produce the following results:



```
[CV] ...... learning_rate=0.2, max_depth=5, score=0.902, total= 2.4min
[CV] learning_rate=0.2, max_depth=5 .................................
[CV] ...... learning_rate=0.2, max_depth=5, score=0.909, total= 2.4min
[CV] learning_rate=0.2, max_depth=5 .................................
[CV] ...... learning_rate=0.2, max_depth=5, score=0.922, total= 2.4min
[CV] learning_rate=0.2, max_depth=7 .................................
[CV] ...... learning_rate=0.2, max_depth=7, score=0.905, total= 3.3min
[CV] learning_rate=0.2, max_depth=7 .................................
[CV] ...... learning_rate=0.2, max_depth=7, score=0.912, total= 3.2min
[CV] learning_rate=0.2, max_depth=7 .................................
[CV] ...... learning_rate=0.2, max_depth=7, score=0.924, total= 3.2min
[Parallel(n_jobs=1)]: Done  27 out of  27 | elapsed: 65.3min finished
```

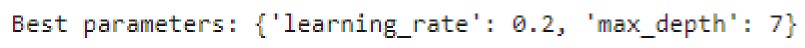
I still use the variables:

- **pred = grid_search.predict(cat_vars_ohe_2020_final)**
- **pred_prob = grid_search.predict_proba(cat_vars_ohe_2020_final)**

To be able to predict 2020 variable values.

Finally, we print the values of the parameters with the

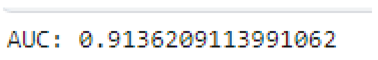
print(f'Best parameters: {grid_search.best_params_}')



```
Best parameters: {'learning_rate': 0.2, 'max_depth': 7}
```

As, and the resulting AUC value:

print(f'AUC: {roc_auc_score(target_2020_final,pred_prob[:,1])}')



```
AUC: 0.9136209113991062
```

From the above result we can conclude that the XGBoostClassifier algorithm is more efficient than the usual Classifier algorithm, because the first gives us an AUC = 0.913 and the second AUC = 0.88.

Of course, in this case we could include more parameters to make our model more efficient, e.g

params = {
'max_depth': [3, 5, 7],
'learning_rate': [0.01, 0.1, 0.2],
'n_estimators': [100, 200, 300],
'gamma': [0, 0.1, 0.2],
'min_child_weight': [1, 5, 10],
'subsample': [0.5, 0.7, 1.0],
'colsample_bytree': [0.6, 0.8, 1.0]}

AUC ROC is a very common and useful metric for evaluating the performance of classification models, particularly in binary classification problems.

The AUC ROC index ranges from 0 to 1. A perfect classification model will have an AUC ROC equal to 1, while a random model will have an AUC ROC of about 0.5. Therefore, the value of 0.913 is quite good and indicates that your model performs very well in predicting the classes of your data, compared to a random model.

However, it is important to note that although the AUC ROC is a useful metric, it should not be used alone to evaluate a model's performance. Other metrics such as precision, recall, F1 score, and predictive value may also be important depending on the problem we are trying to solve.

F. https://www.kaggle.com/code/dimitrisvrasidis/predicting-the-delay-of-flights-auc-0-88

Additionally, we could try other methods such as RandomizedSearchCV which is faster but less exhaustive than GridSearchCV. Another good practice would be to normalize or standardize their continuous characteristics. This could improve the performance of some models.