Warehouse Storage Optimization - Report 8

Dhruvil Dave
School Of Engineering And Applied Sciences
Ahmedabad University
Enrollment No.: AU1841003
email: dhruvil.d@ahduni.edu.in

Dhatri Kapuriya
School Of Engineering And Applied Sciences
Ahmedabad University
Enrollment No.: AU1841129
email: dhatri.k@ahduni.edu.in

Harvish Jariwala
School Of Engineering And Applied Sciences
Ahmedabad University
Enrollment No.: AU1841050
email: harvish.j@ahduni.edu.in

Nisarg Thoriya
School Of Engineering And Applied Sciences
Ahmedabad University
Enrollment No.: AU1841142
email: nisarg.t@ahduni.edu.in

***Abstract*—This is the eighth progress report of our group *Gopher - Group 5* for Machine Learning (CSE523) course project.**

***Index Terms*—Time series forecasting, classification, hyper-parameter tuning**

I. Introduction

For our project, we decided to use the Amazon Bin Images Dataset.

This is originally a Computer Vision dataset. The Amazon Bin Image Dataset contains over 530,000 images and metadata from bins of a pod in an operating Amazon Fulfillment Center.

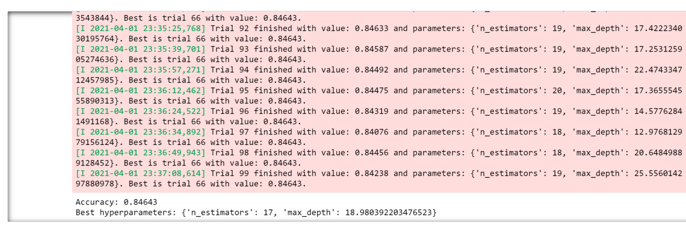
II. Brief insight of previous work

In previous week, we focused on exploring what hyper-parameter tuning actually is. We explored the some models in detail and explored hyper-parameter tuning libraries like Optuna, scikit-optimise and Ray-tune. Then we decided to moved forward with Optuna.

III. Task Performed And Outcomes

This week, we tried exploring optuna in detail and tried to optimise our models in optuna. Firstly, we tried to optimise the Random Forest Classifier using this library. The main code of that is as given below:

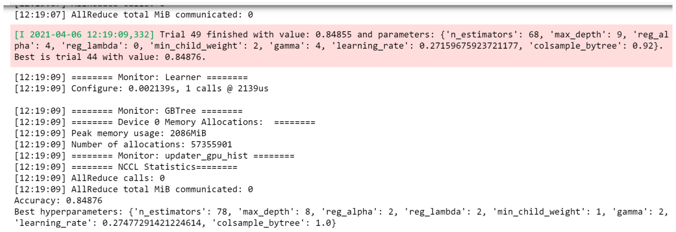
```
import optuna

def objective(trial):

    n_estimators = trial.suggest_int('n_estimators', 2, 20)
    max_depth = int(trial.suggest_float('max_depth', 1, 32, log=True))

    clf = sklearn.ensemble.RandomForestClassifier(n_estimators=n_estimators, max_depth=max_depth, n_jobs=-1)

    clf.fit(x_train.drop(columns='3'), y_train)

    return clf.score(x_test, y_test)

study = optuna.create_study(direction='maximize')
study.optimize(objective, n_trials=100)

trial = study.best_trial

print('Accuracy: {}'.format(trial.value))
print("Best hyperparameters: {}".format(trial.params))
```



We then tried to optimise the parameters for XGBoost Classifier using optuna. We used the following code snippet for that.

```
def objective(trial):
    param = {
                "n_estimators" : trial.suggest_int('n_estimators', 0, 100),
                'max_depth':trial.suggest_int('max_depth', 2, 20),
                'reg_alpha':trial.suggest_int('reg_alpha', 0, 5),
                'reg_lambda':trial.suggest_int('reg_lambda', 0, 5),
                'min_child_weight':trial.suggest_int('min_child_weight', 0, 5),
                'gamma':trial.suggest_int('gamma', 0, 5),
                'learning_rate':trial.suggest_loguniform('learning_rate',0.005,0.5),
                'colsample_bytree':trial.suggest_discrete_uniform('colsample_bytree',0.1,1,0.01),
                'nthread' : -1,
                'verbosity' : 3,
                'tree_method': 'gpu_hist'
            }
```

```python
    model = XGBClassifier(**param)

    model.fit(x_train.drop(columns='3'), y_train)

    return model.score(x_test, y_test)

study = optuna.create_study(direction='maximize')
study.optimize(objective, n_trials=50)

trial = study.best_trial

print('Accuracy: {}'.format(trial.value))
print("Best hyperparameters: {}".format(trial.params
     ))
```

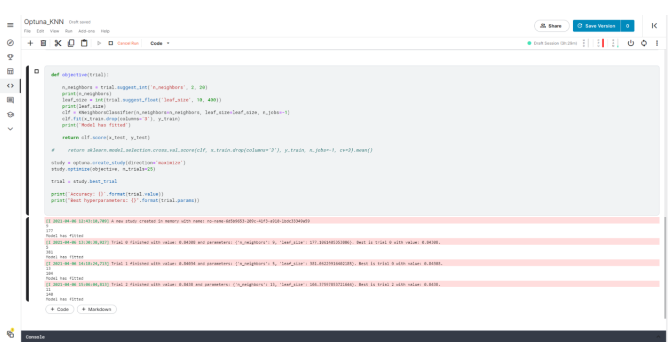


```
[12:19:07] AllReduce total MiB communicated: 0
[I 2021-04-06 12:19:09,332] Trial 49 finished with value: 0.84855 and parameters: {'n_estimators': 68, 'max_depth': 9, 'reg_alpha': 4, 'reg_lambda': 0, 'min_child_weight': 2, 'gamma': 4, 'learning_rate': 0.27159675923721177, 'colsample_bytree': 0.92}. Best is trial 44 with value: 0.84876.
[12:19:09] ======== Monitor: Learner ========
[12:19:09] Configure: 0.002139s, 1 calls @ 2139us

[12:19:09] ======== Monitor: GBTree ========
[12:19:09] ======== Device 0 Memory Allocations:  ========
[12:19:09] Peak memory usage: 2086MiB
[12:19:09] Number of allocations: 57355901
[12:19:09] ======== Monitor: updater_gpu_hist ========
[12:19:09] ======== NCCL Statistics========
[12:19:09] AllReduce calls: 0
[12:19:09] AllReduce total MiB communicated: 0
Accuracy: 0.84876
Best hyperparameters: {'n_estimators': 78, 'max_depth': 8, 'reg_alpha': 2, 'reg_lambda': 2, 'min_child_weight': 1, 'gamma': 2, 'learning_rate': 0.27477291421224614, 'colsample_bytree': 1.0}
```

As you can see, we tried to optimise both of those classifiers using optuna. For the Random Forest Classifier, we tried to set n_ estimators between 2 and 20 and max depth between 1 and 32. Doing this for about 100 trials, we obtained a maximum accuracy of 84.643% with n_estimators=17 and max depth=18.9

For the XGBoost classifier, we tried to use the direct modelling, but that method was too slow and took considerable amount of time. So we trained the model on a GPU to increase the speed. Even after that, modelling the dataset was slow. Hence, we decided to reduce the n_estimators from 1000 to 100. While also changing other parameters that can be seen from the code above. Using this method, we obtained an accuracy of 84.876%

Lastly we tried implementing this in KNN model. But it was drastically slower on our local machine.



So it requires a fairly faster machine to which we did not have an access.

IV. Tasks for Upcoming Week

The main tasks to be performed in the upcoming week are:

1) Explore the libraries and models in details and apply hyperparameter tuning
2) Explore autoML for hyper parameter tuning