ASSIGNMENT 3.4 (a) and (b)

on

Model Deployment and Maintance

Submitted by:

Haseebullah Shaikh (2303.KHI.DEG.015)

and

Faiza Gulzar Ahmed (2303.khi.deg.001)

Dated: 5th May 2023

All files related to Mlos and train.py are uploaded on my github repository.

Assignment 3.4 (a):

Task 01:

Write a component that will log metadata of your Classification model that you trained on the day dedicated to Supervised Learning. Remember to include all metadata that are important to track for this problem.

```
def track_with_mlflow(model, X_test, Y_test, mlflow, model_metadata):
    mlflow.log_params(model_metadata)
    mlflow.log_metric("accuracy", model.score(X_test, Y_test))
    mlflow.sklearn.log_model(model, "rfc", registered_model_name="sklearn_rfc")
```

Assignment 3.4 (b):

Task 01:

Run your Classification model that you trained on the day dedicated to Supervised Learning in MLFlow.

mlops > train.py > main

```python
import fire
import mlflow
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn import datasets
from sklearn.pipeline import make_pipeline
# from sklearn.metrics import f1_score
from sklearn.model_selection import train_test_split


```

mlops > train.py > main

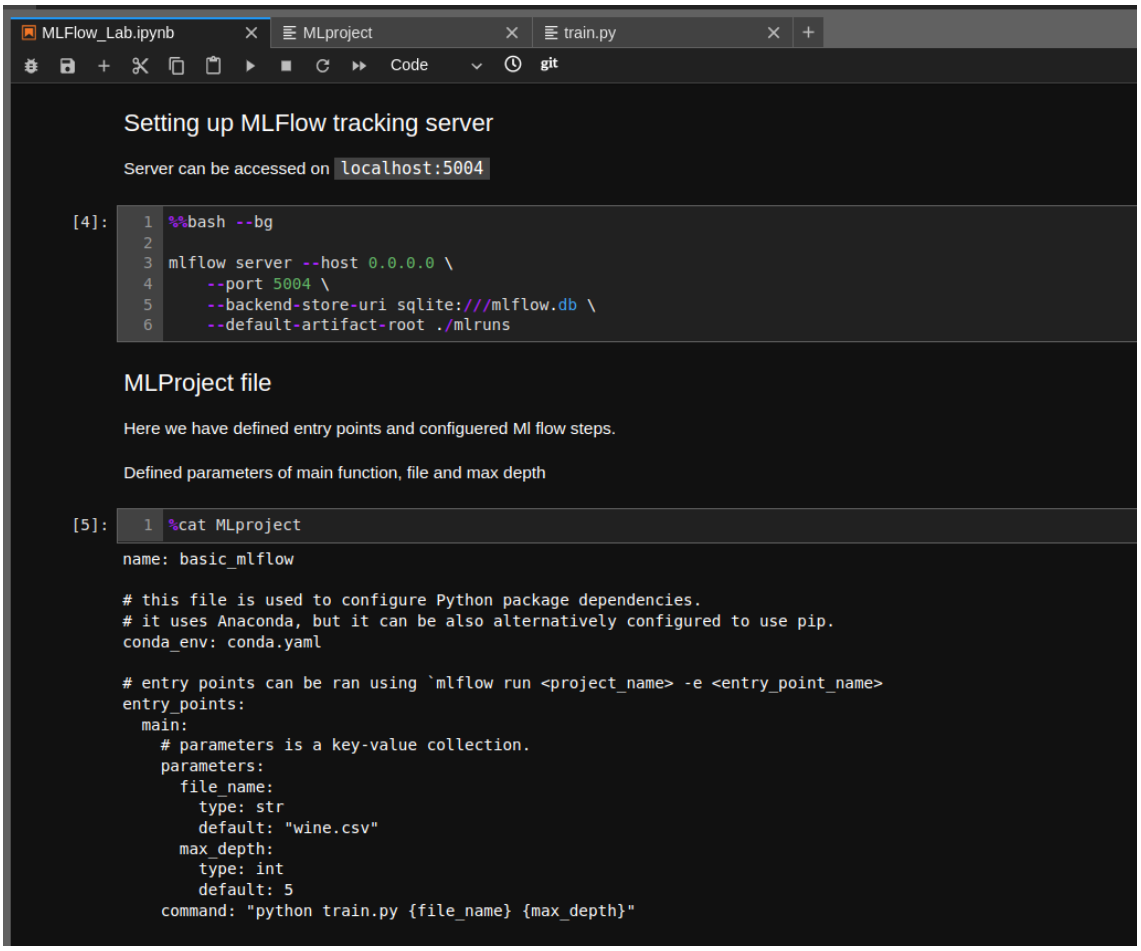
```python
def split_data(wine_data):

    wine_x = wine_data.iloc[:,:6]
    wine_y = wine_data['quality']
    x_train, x_test, y_train, y_test = train_test_split(wine_x, wine_y, test_si

    return x_train, x_test, y_train, y_test


def setup_rfc_pipeline(max_depth):
    rfc_model = RandomForestClassifier(max_depth)
    rfc_pipe = make_pipeline(StandardScaler(), rfc_model)
    return rfc_pipe


```

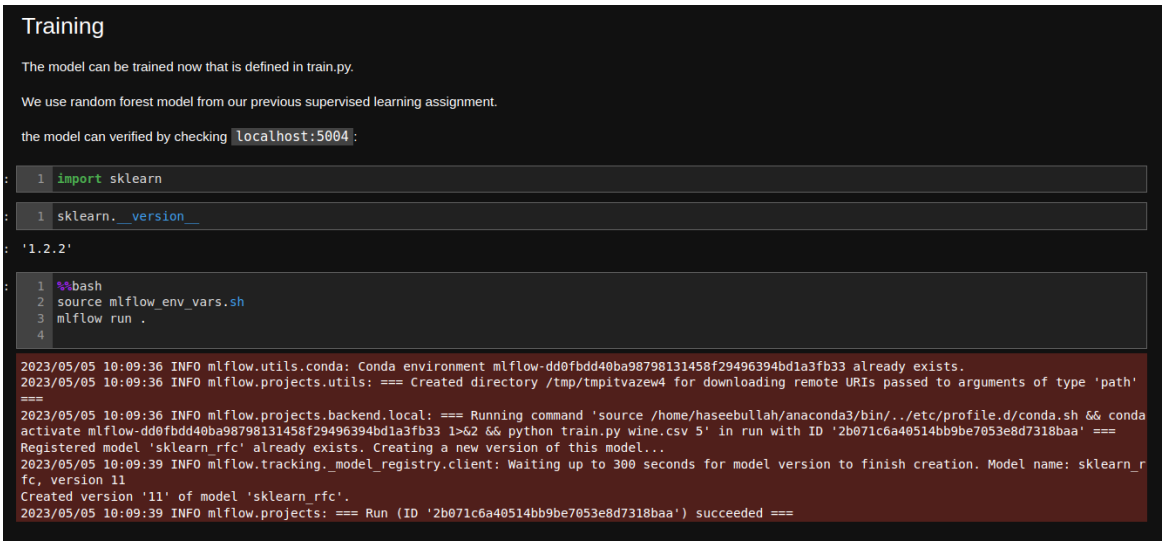
```python
def track_with_mlflow(model, X_test, Y_test, mlflow, model_metadata):
    mlflow.log_params(model_metadata)
    mlflow.log_metric("accuracy", model.score(X_test, Y_test))
    mlflow.sklearn.log_model(model, "rfc", registered_model_name="sklearn_rfc")
```

mlops > train.py > ...

```python
def main(file_name: str, max_depth: int):

    wine_data = pd.read_csv(file_name)
    x_train, x_test, y_train, y_test = split_data(wine_data)

    with mlflow.start_run():
        rfc_pipe = setup_rfc_pipeline(max_depth)
        rfc_pipe.fit(x_train, y_train)
        model_metadata = {"max_depth":max_depth}
        track_with_mlflow(rfc_pipe,x_test,y_test,mlflow,model_metadata)


if __name__ == "__main__":
    fire.Fire(main)

```

MLFlow lab file





Inspecting stored models

The trained models are stored in `mlruns/0`.

These directories contain artifacts and config that is needed to serve them.

The path is same that was defined in class example

[9]:
```
%%bash
last_model_path=$(ls -tr mlruns/0/ | tail -1)
cat mlruns/0/$last_model_path/artifacts/knn/MLmodel
```

```
artifact_path: knn
flavors:
  python_function:
    env:
      conda: conda.yaml
      virtualenv: python_env.yaml
    loader_module: mlflow.sklearn
    model_path: model.pkl
    predict_fn: predict
    python_version: 3.10.6
  sklearn:
    code: null
    pickled_model: model.pkl
    serialization_format: cloudpickle
    sklearn_version: 1.2.2
mlflow_version: 2.3.1
model_uuid: ea83736a0f804e099ea9bc17ea1f735f
run_id: 0659857acaa94fe0ad0a80bfbbe483cf
utc_time_created: '2023-05-03 11:09:26.836996'
```

utc_time_created: '2023-05-03 11:09:26.836996'

[10]:
```
import mlflow
```

[11]:
```
mlflow.__version__
```

[11]: '2.3.1'

Serving model

The model has been traine, which can seen on `localhost:5004`.

The latest version 10 is moved to production stage.

The following cell will serve the model at localhost on port 5005.

[12]:
```
%%bash --bg
source mlflow_env_vars.sh
mlflow --version
mlflow models serve -m models:/sklearn_rfc/Production -p 5005 --env-manager=conda

```

Prediction

We have load data that we can feed into prediction server.

[13]:
```
df = pd.read_csv("wine.csv")
df.head()
```

[13]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

Now predicting the quality of wine

warning: this might fail at first because the prediction server didn't spin up; in this case wait a minute

Now predicting the quality of wine

warning: this might fail at first because the prediction server didn't spin up; in this case wait a minute

[16]:
```
%%bash
data='[[7.4,0.7,0,1.9,0.076,11], [7.4,0.7,0,1.9,0.077,10]]'
echo $data

curl -d "{\"inputs\": $data}" -H 'Content-Type: application/json' 127.0.0.1:5005/invocations
```

```
[[7.4,0.7,0,1.9,0.076,11], [7.4,0.7,0,1.9,0.077,10]]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    87  100    23  100    64   7504  20880 --:--:-- --:--:-- --:--:-- 29000
{"predictions": [5, 5]}
```

[17]:
```
%%bash
data='[[7.4,0.7,0,1.9,0.076,11], [0.0,0.1,1,1.9,0.077,10]]'
echo $data

curl -d "{\"instances\": $data}" -H 'Content-Type: application/json' 127.0.0.1:5005/invocations
```

```
[[7.4,0.7,0,1.9,0.076,11], [0.0,0.1,1,1.9,0.077,10]]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100    90  100    23  100    67  10962  31935 --:--:-- --:--:-- --:--:-- 45000
{"predictions": [5, 6]}
```

[18]:
```
%%bash
data='[[0.4,0.7,1,0.5,0.076,11], [0.0,0.1,0,1.9,0.000,13]]'
columns='["fixed acidity", "volatile acidity", "citric acid", "residual sugar","chlorides", "free sulfur dioxide"]'
echo $data

curl -d "{\"dataframe_split\":{\"columns\":[\"fixed acidity\", \"volatile acidity\", \"citric acid\", \"residual sugar\",\"chlorides\", \"free su
```

```
[[0.4,0.7,1,0.5,0.076,11], [0.0,0.1,0,1.9,0.000,13]]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   221  100    23  100   198   6104  52547 --:--:-- --:--:-- --:--:-- 73666
{"predictions": [4, 7]}
```

Hurrah!, Now we can see the prediction of our model :-).

[ ]:

Updated ML project file

Help

MLFlow_Lab.ipynb | MLproject | train.py

```
name: basic_mlflow

# this file is used to configure Python package dependencies.
# it uses Anaconda, but it can be also alternatively configured to use pip.
conda_env: conda.yaml

# entry points can be ran using `mlflow run <project_name> -e <entry_point_name>
entry_points:
  main:
    # parameters is a key-value collection.
    parameters:
      file_name:
        type: str
        default: "wine.csv"
      max_depth:
        type: int
        default: 5
    command: "python train.py {file_name} {max_depth}"
```

Local host 5004: Mlflow