

ASSIGNMENT 3.4 (a) and (b)
on
Model Deployment and Maintenance

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
1  import fire
2  import mlflow
3  import pandas as pd
4  from sklearn.ensemble import RandomForestClassifier
5  from sklearn.preprocessing import StandardScaler
6  from sklearn import datasets
7  from sklearn.pipeline import make_pipeline
8  # from sklearn.metrics import f1_score
9  from sklearn.model_selection import train_test_split
10
11
```

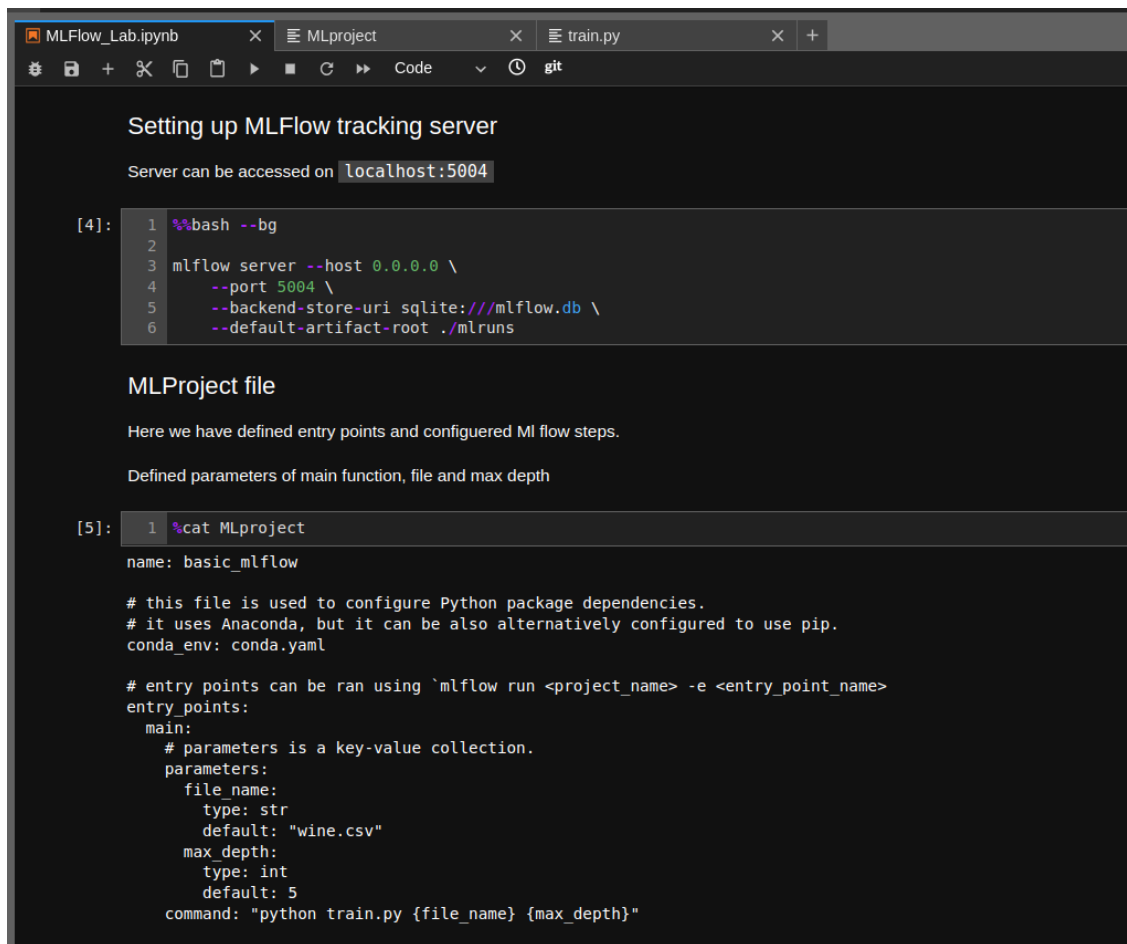
```
mlops > train.py > main
12  def split_data(wine_data):
13
14      wine_x = wine_data.iloc[:, :6]
15      wine_y = wine_data['quality']
16      x_train, x_test, y_train, y_test = train_test_split(wine_x, wine_y, test_si
17
18      return x_train, x_test, y_train, y_test
19
20
21  def setup_rfc_pipeline(max_depth):
22      rfc_model = RandomForestClassifier(max_depth)
23      rfc_pipe = make_pipeline(StandardScaler(), rfc_model)
24      return rfc_pipe
25
26
```

```
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 > ...

```
32 def main(file_name: str, max_depth: int):  
33  
34     wine_data = pd.read_csv(file_name)  
35     x_train, x_test, y_train, y_test = split_data(wine_data)  
36  
37     with mlflow.start_run():  
38         rfc_pipe = setup_rfc_pipeline(max_depth)  
39         rfc_pipe.fit(x_train, y_train)  
40         model_metadata = {"max_depth": max_depth}  
41         track_with_mlflow(rfc_pipe, x_test, y_test, mlflow, model_metadata)  
42  
43  
44 if __name__ == "__main__":  
45     fire.Fire(main)  
46  
47
```

MLFlow lab file



The screenshot shows a JupyterLab window with three tabs: 'MLFlow_Lab.ipynb', 'MLproject', and 'train.py'. The 'MLFlow_Lab.ipynb' tab is active, displaying a notebook with two cells. The first cell, titled 'Setting up MLFlow tracking server', contains a terminal prompt [4]: and a code block for running the 'mlflow server' command. The second cell, titled 'MLProject file', contains a terminal prompt [5]: and a code block for running 'cat MLproject', which displays the contents of the MLProject file. The MLProject file defines a project named 'basic_mlflow' with entry points for 'main' and 'parameters'.

```
Setting up MLFlow tracking server

Server can be accessed on localhost:5004

[4]: 1 %%bash --bg
      2
      3 mlflow server --host 0.0.0.0 \
      4       --port 5004 \
      5       --backend-store-uri sqlite:///mlflow.db \
      6       --default-artifact-root ./mlruns

MLProject file

Here we have defined entry points and configured ML flow steps.

Defined parameters of main function, file and max depth

[5]: 1 %cat MLproject

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

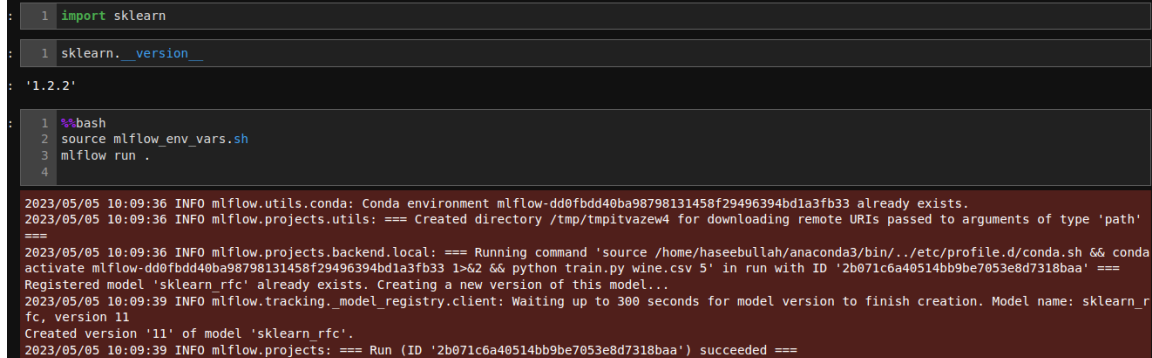
# entry points can be ran using `mlflow run <project_name> -e <entry_point_name>`
entry_points:
  main:
    # parameters is a key-value collection.
    parameters:
      file_name:
        type: str
        default: "wine.csv"
      max_depth:
        type: int
        default: 5
    command: "python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

Training

The model can be trained now that is defined in train.py.

We use random forest model from our previous supervised learning assignment.

the model can be verified by checking localhost:5004 :



The screenshot shows a JupyterLab window with a single tab 'train.py'. The notebook contains a code block for importing sklearn and checking its version, followed by a terminal prompt [4]: and a code block for running 'source mlflow_env_vars.sh' and 'mlflow run .' in a bash shell. Below the code blocks, the output of the training process is displayed, showing logs from MLFlow and the successful completion of the training run.

```
1 import sklearn

1 sklearn.__version__

'1.2.2'

1 %%bash
2 source mlflow_env_vars.sh
3 mlflow run .
4

2023/05/05 10:09:36 INFO mlflow.utils.conda: Conda environment mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33 already exists.
2023/05/05 10:09:36 INFO mlflow.projects.utils: === Created directory /tmp/tmpitvazew4 for downloading remote URIs passed to arguments of type 'path' ===
2023/05/05 10:09:36 INFO mlflow.projects.backend.local: === Running command 'source /home/haseebullah/anaconda3/bin/./etc/profile.d/conda.sh && conda activate mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33 l>62 && python train.py wine.csv 5' in run with ID '2b071c6a40514bb9be7053e8d7318baa' ===
Registered model 'sklearn_rfc' already exists. Creating a new version of this model...
2023/05/05 10:09:39 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rfc, version 11
Created version '11' of model 'sklearn_rfc'.
2023/05/05 10:09:39 INFO mlflow.projects: === Run (ID '2b071c6a40514bb9be7053e8d7318baa') succeeded ===
```

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]: 1 %bash
2 last_model_path=$(ls -tr mlruns/0/ | tail -1)
3 cat mlruns/0/$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: ea83736a0f804e099ea9bc17ealf735f
run_id: 0659857acaa94fe0ad0a80bfbe483cf
utc_time_created: '2023-05-03 11:09:26.836996'
```

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

```
[10]: 1 import mlflow
```

```
[11]: 1 mlflow.__version__
```

```
[11]: '2.3.1'
```

Serving model

The model has been trained, which can be 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]: 1 %bash --bg
2 source mlflow_env_vars.sh
3 mlflow --version
4 mlflow models serve -m models:/sklearn_rfc/Production -p 5005 --env-manager=conda
5
```

Prediction

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

```
[13]: 1 df = pd.read_csv("wine.csv")
2 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]: 1 %bash
      2 data='[[7.4,0.7,0,1.9,0.076,11], [7.4,0.7,0,1.9,0.077,10]]'
      3 echo $data
      4
      5 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]: 1 %bash
      2 data='[[7.4,0.7,0,1.9,0.076,11], [0.0,0.1,1,1.9,0.077,10]]'
      3 echo $data
      4
      5 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]: 1 %bash
      2 data='[[0.4,0.7,1,0.5,0.076,11], [0.0,0.1,0,1.9,0.000,13]]'
      3 columns=["fixed acidity", "volatile acidity", "citric acid", "residual sugar", "chlorides", "free sulfur dioxide"]
      4 echo $data
      5
      6 curl -d "{\"dataframe_split\":{\"columns\":[${columns}], \"data\": $data}" -H 'Content-Type: application/json' 127.0.0.1:5005/invocations

[[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 :-).

[ ]: 1
```

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

mlflow2.3.1

ExperimentsModels

GitHubDocs

Registered Models

Share and manage machine learning models. [Learn more](#)

Create Model

Q Search by model names or tags

Search

Clear

Name	Latest Version	Staging	Production	Last Modified	Tags
sklearn_knn	Version 18	—	Version 9	2023-05-03 16:09:27	—
sklearn_rfc	Version 11	—	Version 9	2023-05-05 10:09:39	—

1

10 / page

Created Time: 2023-05-04 09:51:13

Last Modified: 2023-05-05 10:09:39

> Description [Edit](#)

> Tags

> Versions

AllActive 1Compare

<input type="checkbox"/>	Version	Registered at	Created by	Stage
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 11	2023-05-05 10:09:39		None
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 10	2023-05-04 11:03:26		None
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 9	2023-05-04 10:41:31		Production
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 8	2023-05-04 10:38:18		Archived
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 7	2023-05-04 10:31:09		Archived
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 6	2023-05-04 10:24:19		Archived
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 5	2023-05-04 10:15:25		Archived
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 4	2023-05-04 10:04:36		Archived
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 3	2023-05-04 10:02:10		None
<input type="checkbox"/>	<input checked="" type="checkbox"/> Version 2	2023-05-04 09:59:10		None