

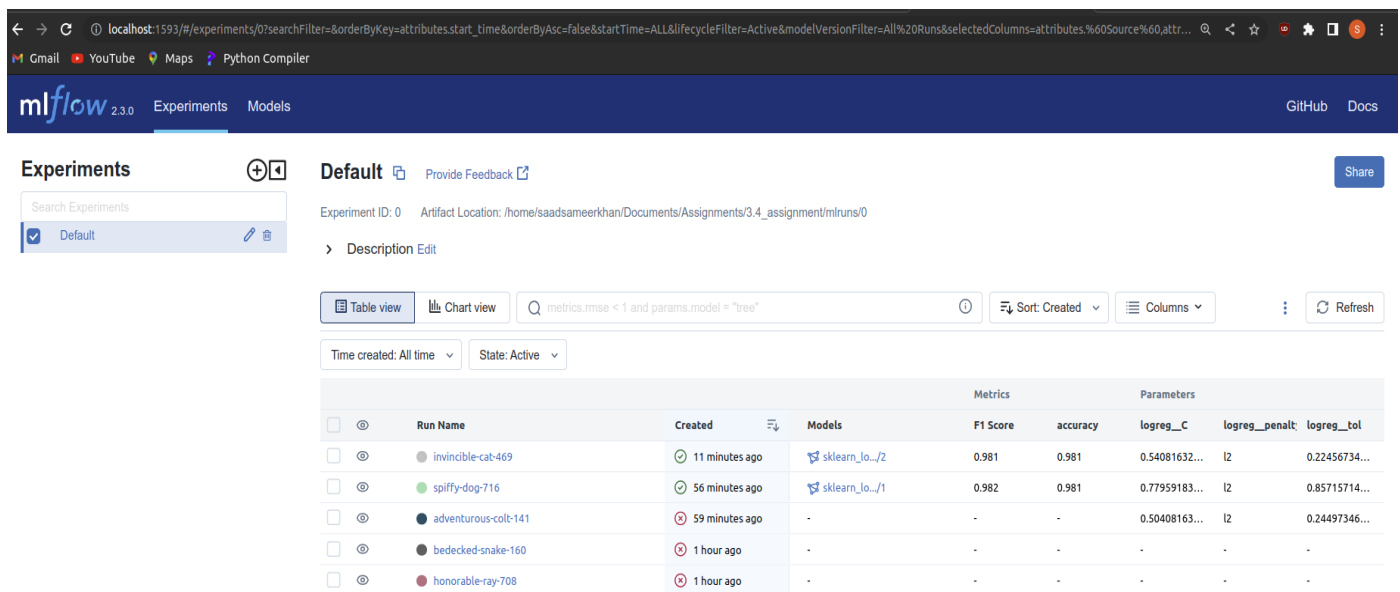
Graded Assignment 3.4(a) & 3.4(b)

Name: Hamza Mohammad Asim

Employee#: 2303.KHI.DEG.014

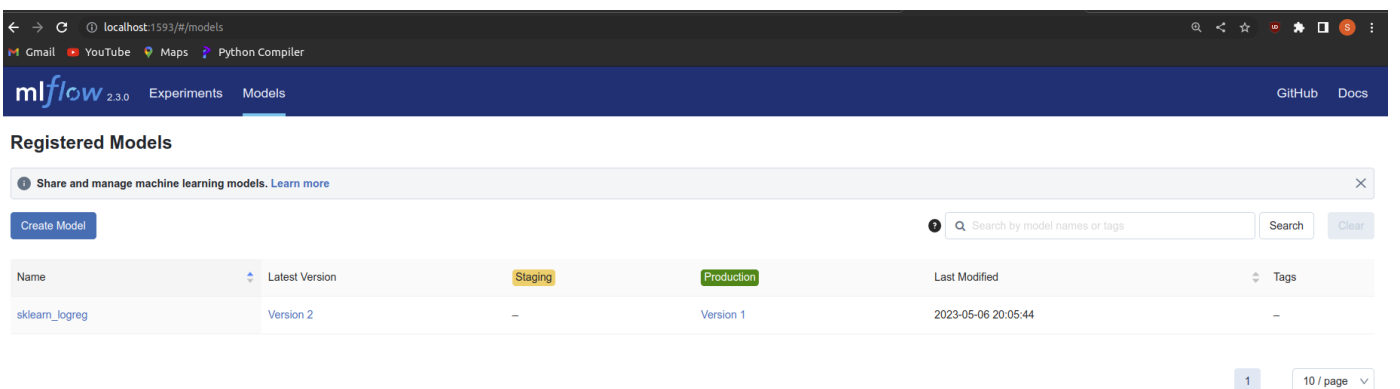
Collaborated with: Saad Sameer Khan (2303.KHI.DEG.034)

Mlflow server (running at port 1593):



The screenshot shows the Mlflow Experiments page. The top navigation bar includes the Mlflow logo, version 2.3.0, and links to Experiments and Models. The left sidebar shows the 'Experiments' tab selected. The main content area displays the 'Default' experiment. Below the search bar, there are tabs for 'Table view' and 'Chart view'. A search filter is applied: 'metrics.rmse < 1 and params.model = "tree"'. The table shows a list of runs with columns for Run Name, Created, Models, F1 Score, accuracy, logreg_C, logreg_penalty, and logreg_tol. The runs are sorted by 'Created' time.

Run Name	Created	Models	F1 Score	accuracy	logreg_C	logreg_penalty	logreg_tol
invincible-cat-469	11 minutes ago	sklearn_lo.../2	0.981	0.981	0.54081632...	l2	0.22456734...
spiffy-dog-716	56 minutes ago	sklearn_lo.../1	0.982	0.981	0.77959183...	l2	0.85715714...
adventurous-colt-141	59 minutes ago	-	-	-	0.50408163...	l2	0.24497346...
bedecked-snake-160	1 hour ago	-	-	-	-	-	-
honorale-ray-708	1 hour ago	-	-	-	-	-	-



The screenshot shows the Mlflow Models page. The top navigation bar includes the Mlflow logo, version 2.3.0, and links to Experiments and Models. The left sidebar shows the 'Models' tab selected. The main content area displays the 'Registered Models' section. Below the search bar, there are tabs for 'Create Model' and 'Search by model names or tags'. The table shows a list of models with columns for Name, Latest Version, Staging, Production, Last Modified, and Tags. The models are sorted by 'Last Modified' time.

Name	Latest Version	Staging	Production	Last Modified	Tags
sklearn_logreg	Version 2	-	Version 1	2023-05-06 20:05:44	-

train.py file:

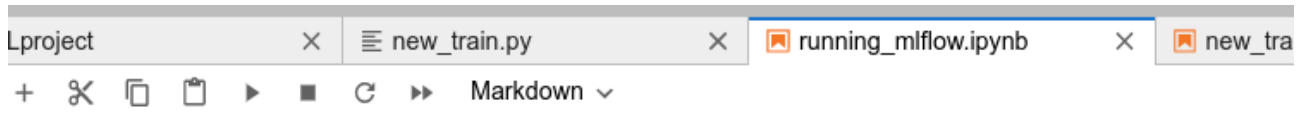
```
MLproject x new_train.py x running_mlflow.ipynb x new_train.ipynb
1 import fire
2 import mlflow
3 import pandas as pd
4 import numpy as np
5 from sklearn import datasets
6 from sklearn.linear_model import LogisticRegression
7 from sklearn.model_selection import RandomizedSearchCV
8 from sklearn.pipeline import Pipeline
9 from sklearn.preprocessing import StandardScaler
10 from sklearn.model_selection import train_test_split
11 from sklearn.metrics import accuracy_score, f1_score
12
13 #splitting our data
14 def split_data(x, y):
15     X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=1522)
16     y_train = y_train.values.ravel()
17     return X_train, X_test, y_train, y_test
18
19
20 # setting a pipeline that scales and instantiates our model
21 def scale_pipeline(scaler_name, scaler, model_name, model):
22     steps = [(scaler_name, scaler), (model_name, model)]
23     pipeline_model = Pipeline(steps)
24     return pipeline_model
25
26 # randomized_search looks for parameters that performs the best within a specified range
27 def randomized_search(pipe, x_train, y_train):
28     #specifying the ranges with param_grid
29     param_grid = {
30         "logreg_penalty" : ["l2"],
31         "logreg_tol" : np.linspace(0.0001, 1, 50),
32         "logreg_C" : np.linspace(0.1, 1, 50)
33     }
34     randomized_search_cv = RandomizedSearchCV(pipe, param_grid, cv=10, n_iter=10)
35     randomized_search_cv.fit(x_train, y_train)
36
37     #selecting the best optimal parameters
38     best_params = randomized_search_cv.best_params_
39
40     #selecting the pipeline that performed the best with the most optimal parameters
41     best_pipeline = randomized_search_cv.best_estimator_
42     return best_pipeline, best_params
43
44
45 def track_with_mlflow(model, X_test, Y_test, mlflow_model_metadata):
```

```

45 def track_with_mlflow(model, X_test, Y_test, mlflow, model_metadata):
46     #logging the model parameters
47     mlflow.log_params(model_metadata)
48
49     y_pred = model.predict(X_test)
50
51     #performing metrics
52     accuracy = accuracy_score(Y_test, y_pred)
53     f1 = f1_score(Y_test, y_pred, average="weighted")
54
55     #logging the metrics
56     mlflow.log_metric("accuracy", accuracy)
57     mlflow.log_metric("F1 Score", f1)
58
59     #logging our model
60     mlflow.sklearn.log_model(model, "logreg", registered_model_name="sklearn_logreg")
61
62
63 def main(logreg_type: str, solver_name: str):
64     wine = datasets.load_wine()
65     X = pd.DataFrame(wine.data, columns = wine.feature_names) # dataframe with all feature columns
66     y = pd.DataFrame(wine.target, columns = ['encoded_class']) # dataframe with target column
67
68     #removing the last 2 columns so that we can make predictions on them later **THROUGH MLFLOW**
69     X = X.iloc[:-2]
70     y = y.iloc[:-2]
71
72     X_train, X_test, y_train, y_test = split_data(X, y)
73
74     with mlflow.start_run():
75         #initializing the LogisticRegression model
76         log_model = LogisticRegression(multi_class=logreg_type, solver=solver_name, random_state=1522)
77
78         #setting up the pipeline
79         pipeline = scale_pipeline("scaler", StandardScaler(), "logreg", log_model)
80
81         #getting the best pipeline and the best parameters
82         best_pipe, model_metadata = randomized_search(pipeline, X_train, y_train)
83
84         #fitting data to the best pipeline
85         best_pipe.fit(X_train, y_train)
86
87         #tracking with mlflow
88         track_with_mlflow(best_pipe, X_test, y_test, mlflow, model_metadata)
89

```

running_mlflow.ipynb file



```
[44]: 1 !python -c "import sys; print(sys.executable)"
      /home/saadsameerkhan/anaconda3/bin/python
```

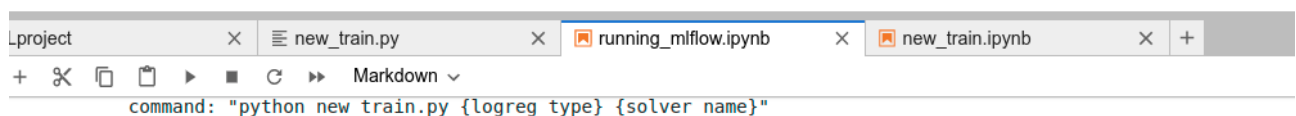
Setting up mlflow server at port 1593

```
[45]: 1 %%bash --bg
      2
      3 mlflow server --host 0.0.0.0 \
      4     --port 1593 \
      5     --backend-store-uri sqlite:///mlflow.db \
      6     --default-artifact-root ./mlruns
```

```
[46]: 1 %cat MLproject
      name: basic_mlflow

      conda_env: conda.yaml

      entry_points:
        main:
          # parameters is a key-value collection.
          parameters:
            solver_name:
              type: str
              default: "lbfgs"
            logreg_type:
              type: str
              default: "multinomial"
          command: "python new_train.py {logreg_type} {solver_name}"
```



Running MLproject file

```
[47]: 1 %%bash
      2 source mlflow_env_vars.sh
      3 mlflow run .

2023/05/06 20:05:41 INFO mlflow.utils.conda: Conda environment mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33
2023/05/06 20:05:41 INFO mlflow.projects.utils: === Created directory /tmp/tmpdnadj8kn for downloading remote
type 'path' ===
2023/05/06 20:05:41 INFO mlflow.projects.backend.local: === Running command 'source /home/saadsameerkhan/anacc
nda.sh && conda activate mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33 1>&2 && python new_train.py multinomi
d43e51bfbf4d3e87821c1c2fcbce7e' ===
/home/saadsameerkhan/anaconda3/envs/mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packag
py:33: UserWarning: Setuptools is replacing distutils.
  warnings.warn("Setuptools is replacing distutils.")
Registered model 'sklearn_logreg' already exists. Creating a new version of this model...
2023/05/06 20:05:44 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version t
e: sklearn_logreg, version 2
Created version '2' of model 'sklearn_logreg'.
2023/05/06 20:05:44 INFO mlflow.projects: === Run (ID 'a0d43e51bfbf4d3e87821c1c2fcbce7e') succeeded ===
```

Checking MLmodel file of last model run

It gives info about the model, the input output data as well as some metadata

```
[48]: 1 %%bash
      2 last_model_path=$(ls -tr mlruns/0/ | tail -1)
      3 cat mlruns/0/$last_model_path/artifacts/logreg/MLmodel
```

```
artifact_path: logreg
flavors:
  python_function:
    env:
      conda: conda.yaml
      virtualenv: python_env.yaml
    loader_module: mlflow.sklearn
    model_path: model.pkl
    predict_fn: predict
    python_version: 3.11.3
  sklearn:
    code: null
    pickled_model: model.pkl
    serialization_format: cloudpickle
    sklearn_version: 1.2.2
mlflow_version: 2.3.1
model_uuid: 5d2f9556dcea4f46aa6caf6365acbfa3
run_id: a0d43e51bfbf4d3e87821c1c2fcbce7e
utc_time_created: '2023-05-06 15:05:43.389914'
```

Serving model (at port 1594)

Serving the model that is in production

so that we can give it our data

and it give us its predictions

```
[49]: 1 %%bash --bg
      2 source mlflow_env_vars.sh
      3 mlflow --version
      4 mlflow models serve -m models:/sklearn_logreg/Production -p 1594 --env-manager=conda
```

Selecting the last 2 rows of the dataset

These rows were unseen by the model during training

```
[66]: 1 from sklearn import datasets
      2 wine = datasets.load_wine()
      3 X = pd.DataFrame(wine.data, columns = wine.feature_names) # dataframe with all feature columns
      4 y = pd.DataFrame(wine.target, columns = ['encoded_class']) # dataframe with target column
      5 X['target'] = y
      6 test_df = X.iloc[-2:]
      7 test_df
```

```
[66]:  alcohol  malic_acid  ash  alkalinity_of_ash  magnesium  total_phenols  flavanoids  nonflavonoid_phenols  proanthocyanins  color_
176    13.17         2.59  2.37             20.0         120.0          1.65          0.68              0.53              1.46
177    14.13         4.10  2.74             24.5          96.0          2.05          0.76              0.56              1.35
```

Making prediction

```
[68]: 1 sample2 = test_df.iloc[1].tolist()
      2 sample2 = sample2[:-1]
      3 sample2
```

```
[68]: [14.13, 4.1, 2.74, 24.5, 96.0, 2.05, 0.76, 0.56, 1.35, 9.2, 0.61, 1.6, 560.0]
```

Giving these 2 samples to the model

```
[69]: 1
      2 [[13.27,4.28,2.26,20.0,120.0,1.59,0.69,0.43,1.35,10.2,0.59,1.56,835.0],[13.17,2.59,2.37,20.0,120.0,1.65,0.68,0.53,1.46,9.3,0.6,1.62,840.0]]
      3 data
      4
      5 {"inputs": $data} -H 'Content-Type: application/json' 127.0.0.1:1594/invocations
```

```
[[13.27,4.28,2.26,20.0,120.0,1.59,0.69,0.43,1.35,10.2,0.59,1.56,835.0],[13.17,2.59,2.37,20.0,120.0,1.65,0.68,0.53,1.46,9.3,0.6,1.62,840.0]]
```

% Total	% Received	% Xferd	Average Dload	Average Upload	Speed	Time Total	Time Spent	Time Left	Current Speed
100	174	100	23	100	151	19557	125k	--:--:--	169k

```
{"predictions": [2, 2]}
```

```
[70]: 1 %bash
      2 data='[[13.27,4.28,2.26,20.0,120.0,1.59,0.69,0.43,1.35,10.2,0.59,1.56,835.0],[13.17,2.59,2.37,20.0,120.0,1.65,0.68,0.53,1.46,9.3,0.6,1.62,840.0]]'
      3 echo $data
      4
      5 curl -d {"instances": $data} -H 'Content-Type: application/json' 127.0.0.1:1594/invocations
```

```
[[13.27,4.28,2.26,20.0,120.0,1.59,0.69,0.43,1.35,10.2,0.59,1.56,835.0],[13.17,2.59,2.37,20.0,120.0,1.65,0.68,0.53,1.46,9.3,0.6,1.62,840.0]]
```

% Total	% Received	% Xferd	Average Dload	Average Upload	Speed	Time Total	Time Spent	Time Left	Current Speed
100	177	100	23	100	154	9770	65420	--:--:--	88500

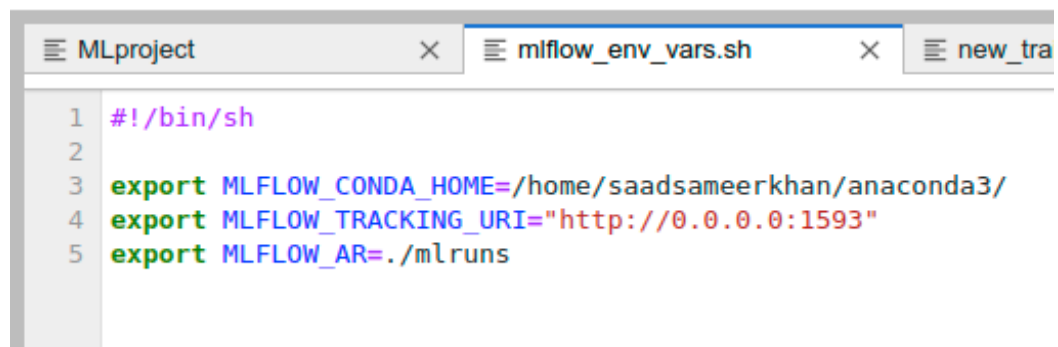
```
{"predictions": [2, 2]}
```

▼ The model has made correct predictions!

MLProject File:

```
MLproject × new_train.py × running_mlflow.ipynb
1 name: basic_mlflow
2
3 conda_env: conda.yaml
4
5 entry_points:
6   main:
7     # here we specify the parameters that the our main function takes
8     parameters:
9       solver_name:
10         type: str
11         default: "lbfgs"
12       logreg_type:
13         type: str
14         default: "multinomial"
15     command: "python new_train.py {logreg_type} {solver_name}"
```

mlflow_env_vars.sh file



The image shows a code editor window with three tabs: 'MLproject', 'mlflow_env_vars.sh', and 'new_tra'. The 'mlflow_env_vars.sh' tab is active and contains a shell script with five lines of code. The first line is a comment. The next three lines are 'export' statements for MLFLOW_CONDA_HOME, MLFLOW_TRACKING_URI, and MLFLOW_AR. The code is color-coded: comments are purple, 'export' is green, and variable names are blue.

```
1 #!/bin/sh
2
3 export MLFLOW_CONDA_HOME=/home/saadsameerkhan/anaconda3/
4 export MLFLOW_TRACKING_URI="http://0.0.0.0:1593"
5 export MLFLOW_AR=./mlruns
```