MLFlow Project Report on Titanic Dataset

Steps:

1. Install MLFlow and required libraries.

   Run the following commands in terminal:
   ```
   pip install mlflow
   pip install scikit-learn
   ```

2. Load, preprocess the data and perform EDA.

   ## EDA Insights :

   - Only **38%** of the passengers **survived**
   - Did any **gender** have a better chance of survival?
     - **Females** had a survival rate of **74%**
     - **Males** had a survival rate of **19%**
   - Did belonging a particular **socio-economic class** have an impact chances of survival?
     - **Upper class** had a survival rate of **63%**
     - **Middle class** had a survival rate of **47%**
     - **Lower class** had a survival rate of **41%**
   - Did people with **family** have a better chance of surviving?
     - People **with Siblings/spouses** had rougly **50%** probability of surviving
     - People **without Siblings/spouses** had a **35%** probability of surviving
     - People with Siblings/spouses had a better chance of surviving probably because they might have been given preference.
     - Similar trend can be seen for people with parents/children v/s those without.

   Preprocessing and EDA have been documented in the following file:
   Titantic_preprocessing.ipynb

   (Link : https://github.com/h-mehta/SE-for-DS/blob/main/Data%20Engineering/MLFlow/titanic/Titantic_preprocessing.ipynb)

3. Save the preprocessed data so it can be directly read-in by the .py file that contains code to develop model on this pre-processed data.

   Data after pre-processing: titanic_processed_data.csv

   (Link: https://github.com/h-mehta/SE-for-DS/blob/main/Data%20Engineering/MLFlow/titanic/titanic_processed_data.csv)

4. Train the classification model and log the parameters and metrics after each run using MLFlow.

   Code in file: titanic_code.py

   (Link: https://github.com/h-mehta/SE-for-DS/blob/main/Data%20Engineering/MLFlow/titanic/titanic_code.py)

5. Can create custom logs also apart from those generated by MLFlow:

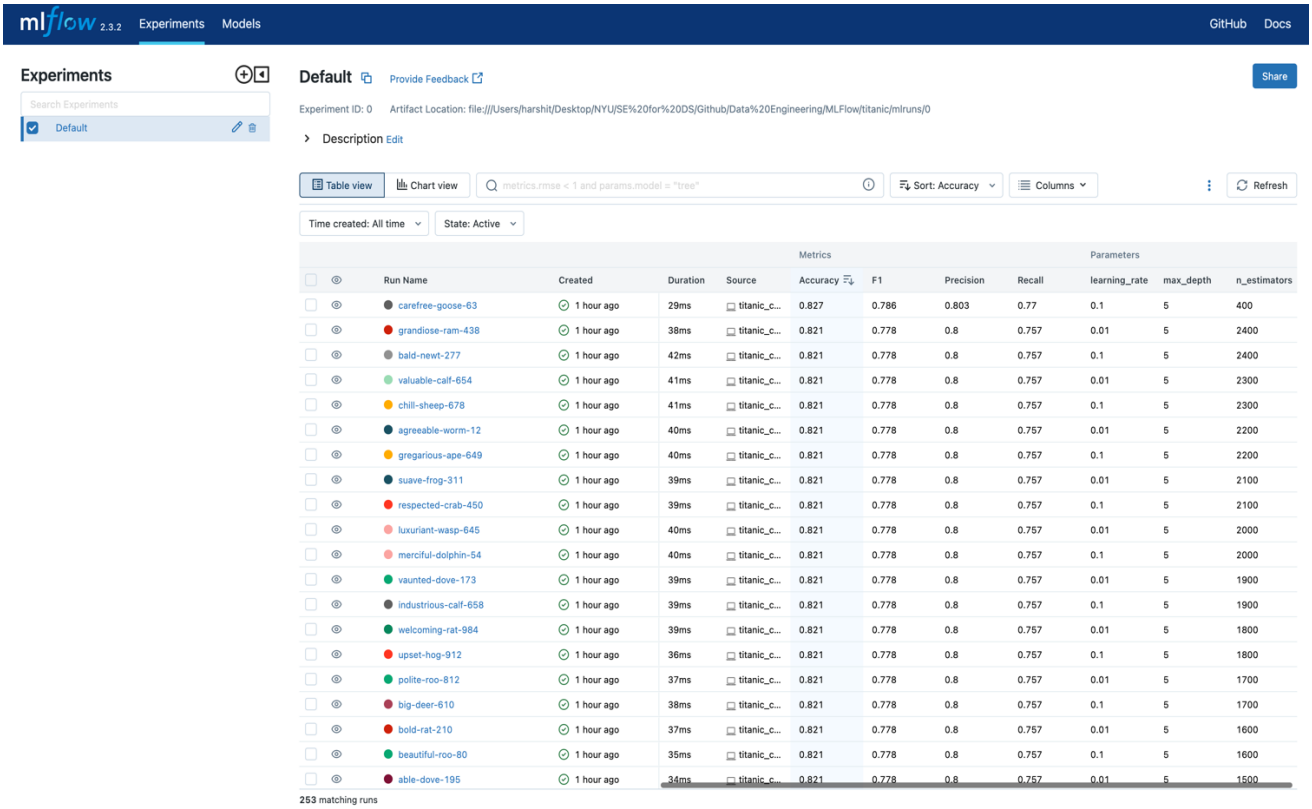
```
2023-06-05 16:57:58,158 - __main__ - INFO - ----------------------------------------
2023-06-05 16:57:59,938 - __main__ - WARNING -  Start of new run ...
2023-06-05 16:58:01,439 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:01,439 - __main__ - INFO -  PARAMETERS : ---
2023-06-05 16:58:01,439 - __main__ - INFO - n_estimators : 2400
2023-06-05 16:58:01,439 - __main__ - INFO - learning_rate : 0.05
2023-06-05 16:58:01,439 - __main__ - INFO - max_depth : 7
2023-06-05 16:58:01,439 - __main__ - INFO -  METRICS : ---
2023-06-05 16:58:01,439 - __main__ - INFO - Accuracy : 0.7877094972067039
2023-06-05 16:58:01,439 - __main__ - INFO - Precision : 0.7647058823529411
2023-06-05 16:58:01,439 - __main__ - INFO - Recall : 0.7027027027027027
2023-06-05 16:58:01,439 - __main__ - INFO - F1 : 0.7323943661971832
2023-06-05 16:58:01,440 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:03,216 - __main__ - WARNING -  Start of new run ...
2023-06-05 16:58:04,888 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:04,889 - __main__ - INFO -  PARAMETERS : ---
2023-06-05 16:58:04,889 - __main__ - INFO - n_estimators : 2400
2023-06-05 16:58:04,889 - __main__ - INFO - learning_rate : 0.01
2023-06-05 16:58:04,889 - __main__ - INFO - max_depth : 5
2023-06-05 16:58:04,889 - __main__ - INFO -  METRICS : ---
2023-06-05 16:58:04,889 - __main__ - INFO - Accuracy : 0.8212290502793296
2023-06-05 16:58:04,889 - __main__ - INFO - Precision : 0.8
2023-06-05 16:58:04,889 - __main__ - INFO - Recall : 0.7567567567567568
2023-06-05 16:58:04,889 - __main__ - INFO - F1 : 0.7777777777777778
2023-06-05 16:58:04,889 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:06,670 - __main__ - WARNING -  Start of new run ...
2023-06-05 16:58:08,458 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:08,458 - __main__ - INFO -  PARAMETERS : ---
2023-06-05 16:58:08,458 - __main__ - INFO - n_estimators : 2400
2023-06-05 16:58:08,458 - __main__ - INFO - learning_rate : 0.01
2023-06-05 16:58:08,458 - __main__ - INFO - max_depth : 6
2023-06-05 16:58:08,458 - __main__ - INFO -  METRICS : ---
2023-06-05 16:58:08,459 - __main__ - INFO - Accuracy : 0.8044692737430168
2023-06-05 16:58:08,459 - __main__ - INFO - Precision : 0.782608695652174
2023-06-05 16:58:08,459 - __main__ - INFO - Recall : 0.7297297297297297
2023-06-05 16:58:08,459 - __main__ - INFO - F1 : 0.7552447552447553
2023-06-05 16:58:08,459 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:10,195 - __main__ - WARNING -  Start of new run ...
2023-06-05 16:58:12,220 - __main__ - INFO - ----------------------------------------
2023-06-05 16:58:12,221 - __main__ - INFO -  PARAMETERS : ---
2023-06-05 16:58:12,221 - __main__ - INFO - n_estimators : 2400
2023-06-05 16:58:12,221 - __main__ - INFO - learning_rate : 0.01
2023-06-05 16:58:12,221 - __main__ - INFO - max_depth : 7
2023-06-05 16:58:12,221 - __main__ - INFO -  METRICS : ---
2023-06-05 16:58:12,221 - __main__ - INFO - Accuracy : 0.776536312849162
2023-06-05 16:58:12,221 - __main__ - INFO - Precision : 0.75
2023-06-05 16:58:12,221 - __main__ - INFO - Recall : 0.6891891891891891
2023-06-05 16:58:12,221 - __main__ - INFO - F1 : 0.7183098591549296
2023-06-05 16:58:12,221 - __main__ - INFO - ----------------------------------------
(base) harshit@Harshits-MacBook-Pro titanic %
```

The custom logs are recoding parameters (n_estimators, learning rate, max depth) and metrics (Accuracy, Precision, recall, F1) successfully.

6. Write bash script to run the model for various combinations of hyperparamters:

   (Refer: https://github.com/h-mehta/SE-for-DS/blob/main/Data%20Engineering/MLFlow/titanic/grid_search.sh)

7. Run the model for different values of hyperparameters – MLFlow keeps a log of hyperparameters and metrics for each run – and then run MLFlow UI:



After running the for 250+ combinations of hyperparameters, we can see that the best accuracy is achieved with the following hyperparameters:

Learning rate: 0.1
Max depth: 5
Estimators: 400

Accuracy: 82.7%
F1: 0.786
Precision: 0.803
Recall: 0.77