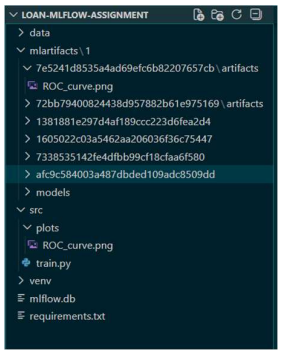
Loan Prediction MLflow Assignment 24040208005-Ritick Raj Gupta

Using three different models like **Logistic Regression**, **Decision Tree**, **Random forest** with **GridSearchCV** to automatically find the hyper parameter, to get trained on loan_dataset. And visualize the best model based on different metrics like accuracy.

1. Project folder view



MLflow tracking URI

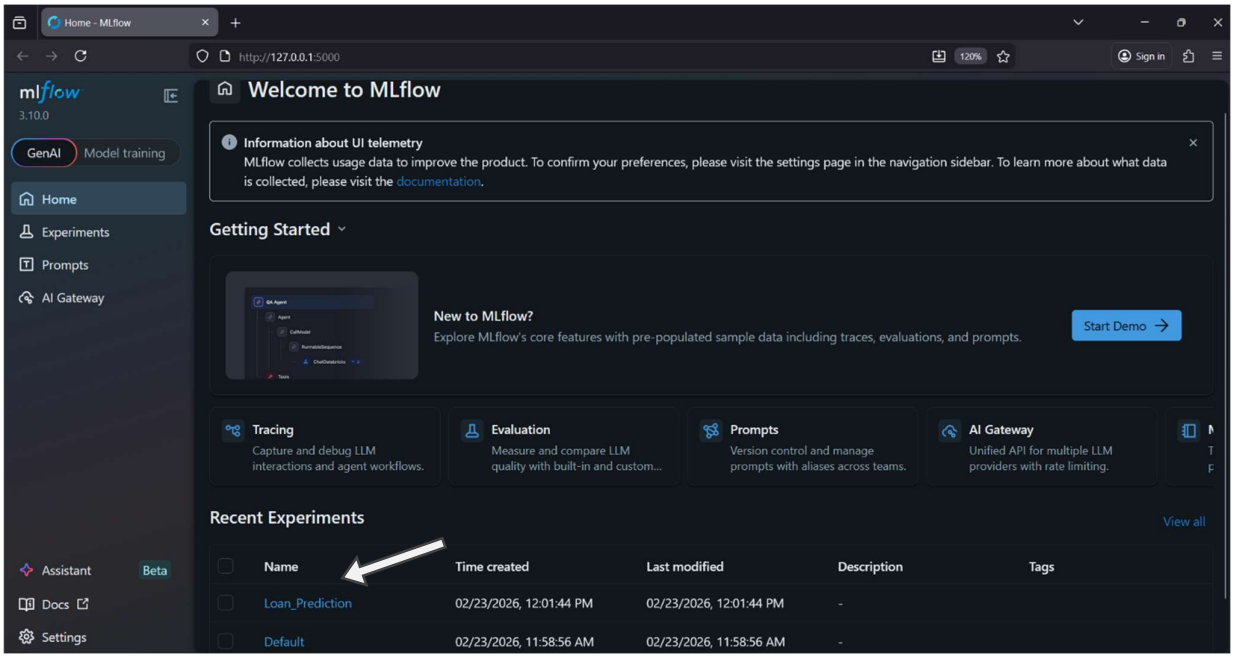
```python
#Setting MLflow tracking URI
mlflow.set_tracking_uri("http://localhost:5000")

mlflow.set_experiment("Loan_Prediction")

os.makedirs("plots", exist_ok=True)
```

2. Commanding mlfow ui, to start ML flow user interface @ http://localhost:5000.

```
(venv) PS E:\MSC 4\MLOps\loan-MLflow-assignment> mlflow ui
Backend store URI not provided. Using sqlite:///mlflow.db
Registry store URI not provided. Using backend store URI.
[MLflow] Security middleware enabled with default settings (localhost-only). To allow connections from other hosts, use --host 0.0.0.0 and configure --allo
wed-hosts and --cors-allowed-origins.
2026/02/24 11:15:17 WARNING mlflow.server: MLflow job execution requirements not met (MLflow job backend does not support Windows system.). Server will sta
rt without job execution support. Errors will be surfaced at job invocation time.
INFO:     Uvicorn running on http://127.0.0.1:5000 (Press CTRL+C to quit)
INFO:     Started parent process [7224]
INFO:     Started server process [19208]
INFO:     Waiting for application startup.
INFO:     Started server process [19848]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Started server process [18584]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Application startup complete.
INFO:     Started server process [11712]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     127.0.0.1:51123 - "GET / HTTP/1.1" 304 Not Modified
INFO:     127.0.0.1:51764 - "GET /ajax-api/3.0/mlflow/server-info HTTP/1.1" 200 OK
INFO:     127.0.0.1:51764 - "GET /ajax-api/3.0/mlflow/assistant/config HTTP/1.1" 200 OK
INFO:     127.0.0.1:51765 - "GET /ajax-api/3.0/mlflow/ui-telemetry HTTP/1.1" 200 OK
INFO:     127.0.0.1:51771 - "GET /ajax-api/2.0/mlflow/experiments/search?max_results=5&order_by=last_update_time+DESC HTTP/1.1" 200 OK
```



3. model evaluation function **mlflow logs** in train.py

```python
#Model Evaluation Function
def model_metrics(actual, pred):
    accuracy = metrics.accuracy_score(actual, pred)
    f1 = metrics.f1_score(actual, pred, pos_label = 1)

    fpr, tpr, _ = metrics.roc_curve(actual, pred)

    auc = metrics.auc(fpr, tpr)

    plt.figure(figsize=(6, 6))
    plt.plot(fpr, tpr, label=f"AUC = {auc:.2f}")
    plt.plot([0, 1], [0, 1], 'r--')
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title("ROC Curve")
    plt.legend()
    plt.savefig("plots/ROC_curve.png")
    plt.close()

    return accuracy, f1, auc
```

```python
def mlflow_logs(model, X, y, name):
    with mlflow.start_run(run_name=name) as run:
        run_id = run.info.run_id
        mlflow.set_tag("run_id", run_id)
        mlflow.set_tag("model_name", name)

        # Predictions
        pred = model.predict(X)

        # Metrics
        accuracy, f1, auc = model_metrics(y, pred)

        # Log Parameters (best params from GridSearch)
        mlflow.log_params(model.best_params_)

        # Log Metrics
        mlflow.log_metric("Mean CV Score", model.best_score_)
        mlflow.log_metric("Accuracy", accuracy)
        mlflow.log_metric("F1-Score", f1)
        mlflow.log_metric("AUC", auc)

        # Log Artifact (ROC Curve)
        mlflow.log_artifact("plots/ROC_curve.png")

        # Log Model
        mlflow.sklearn.log_model(model, name)
```

4. Used three different models: Logistic Regression, Decision Tree and Random forest models

```python
#Logistic Regression
lr = LogisticRegression(random_state=SEED)
lr_param_grid = {
    'C' : [100, 10, 1.0, 0.1],
    'penalty' : ['l1', 'l2'],
    'solver' : ['liblinear']
}

lr_gs = GridSearchCV(
    estimator=lr,
    param_grid=lr_param_grid,
    cv=5,
    n_jobs=-1,
    scoring='accuracy'
)
lr_model = lr_gs.fit(X_train, Y_train)
```

```python
#Decision Tree
dt = DecisionTreeClassifier(random_state = SEED)
dt_params_grid = {
    'max_depth': [3,5,7,9],
    'criterion': ['gini','entropy']
}

dt_gs = GridSearchCV(
    estimator=dt,
    param_grid=dt_params_grid,
    cv=5,
    n_jobs=-1,
    scoring='accuracy'
)
dt_model = dt_gs.fit(X_train, Y_train)
```

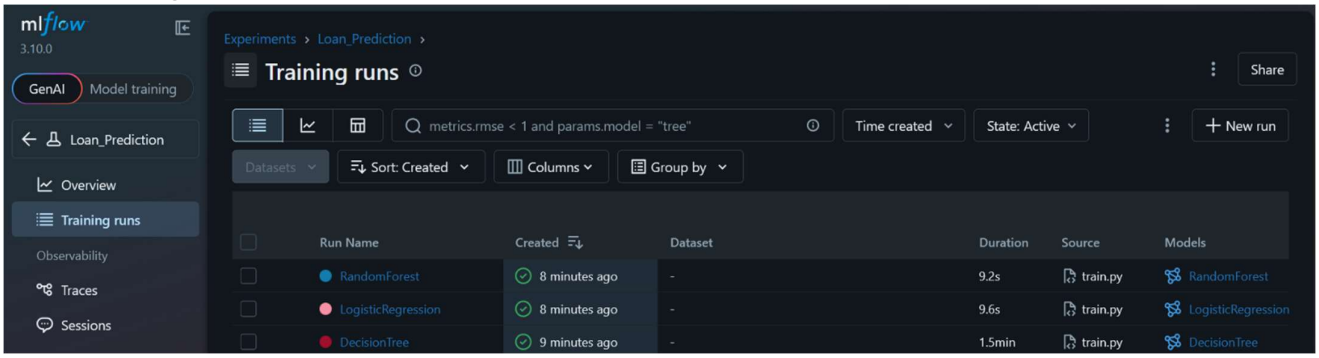
```python
#Random Forest
rf = RandomForestClassifier(random_state=SEED)
rf_params_grid = {
    'n_estimators':[100,200],
    'max_depth':[10,20],
    'criterion':['gini', 'entropy']
}

rf_gs = GridSearchCV(
    estimator=rf,
    param_grid=dt_params_grid,
    cv=5,
    n_jobs=-1,
    scoring='accuracy'
)
rf_model = rf_gs.fit(X_train, Y_train)
```

5. Commanding train.py to run the models and put them on MLflow UI:

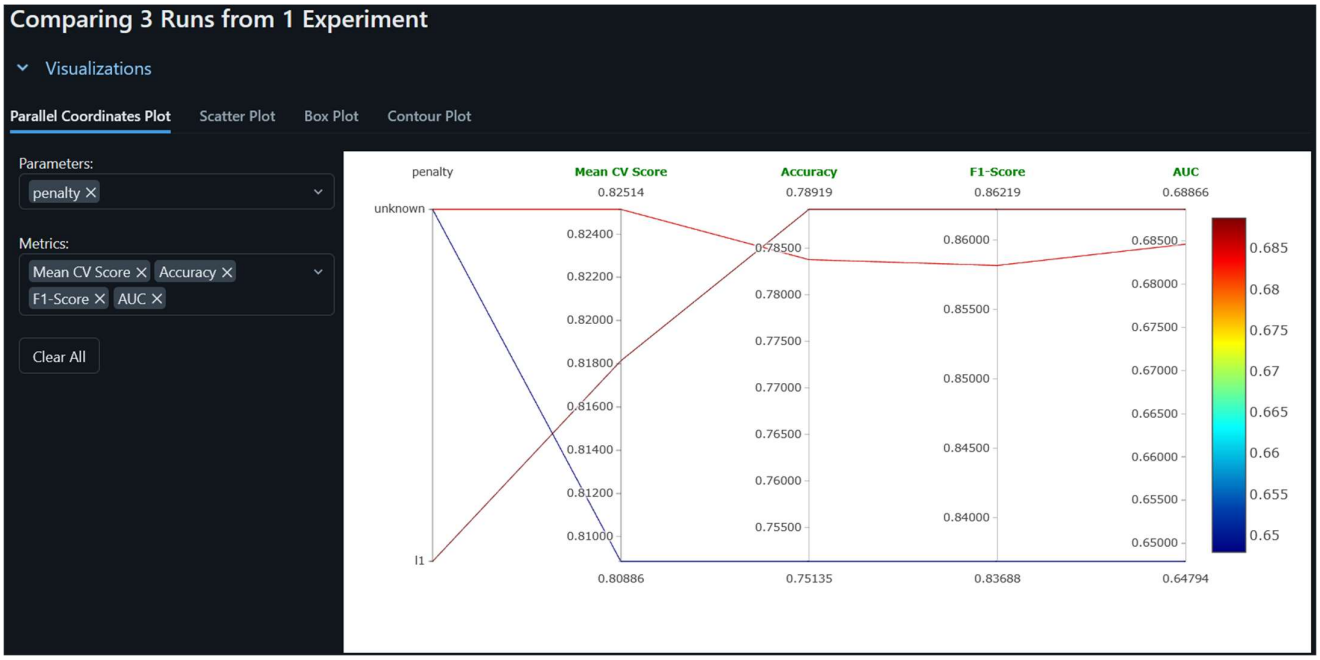
```
(venv) PS E:\MSC 4\MLOps\loan-MLflow-assignment\src> py train.py
 safe alternative is the 'skops' format. For more information, see: https://scikit-learn.org/stable/model_persistence.html
🏃 View run DecisionTree at: http://localhost:5000/#/experiments/1/runs/94a83266863e460c9881c19d100ec635
🧪 View experiment at: http://localhost:5000/#/experiments/1
2026/02/24 11:33:16 WARNING mlflow.models.model: `artifact_path` is deprecated. Please use `name` instead.
2026/02/24 11:33:17 WARNING mlflow.sklearn: Saving scikit-learn models in the pickle or cloudpickle format requires exercising caution beca
use these formats rely on Python's object serialization mechanism, which can execute arbitrary code during deserialization. The recommended
 safe alternative is the 'skops' format. For more information, see: https://scikit-learn.org/stable/model_persistence.html
🏃 View run LogisticRegression at: http://localhost:5000/#/experiments/1/runs/40e23d708c93497f8c94c0a4fc1a62f3
🧪 View experiment at: http://localhost:5000/#/experiments/1
2026/02/24 11:33:25 WARNING mlflow.models.model: `artifact_path` is deprecated. Please use `name` instead.
2026/02/24 11:33:26 WARNING mlflow.sklearn: Saving scikit-learn models in the pickle or cloudpickle format requires exercising caution beca
 safe alternative is the 'skops' format. For more information, see: https://scikit-learn.org/stable/model_persistence.html
🏃 View run LogisticRegression at: http://localhost:5000/#/experiments/1/runs/40e23d708c93497f8c94c0a4fc1a62f3
🧪 View experiment at: http://localhost:5000/#/experiments/1
2026/02/24 11:33:25 WARNING mlflow.models.model: `artifact_path` is deprecated. Please use `name` instead.
2026/02/24 11:33:26 WARNING mlflow.sklearn: Saving scikit-learn models in the pickle or cloudpickle format requires exercising caution beca
🧪 View experiment at: http://localhost:5000/#/experiments/1
2026/02/24 11:33:25 WARNING mlflow.models.model: `artifact_path` is deprecated. Please use `name` instead.
2026/02/24 11:33:26 WARNING mlflow.sklearn: Saving scikit-learn models in the pickle or cloudpickle format requires exercising caution beca
2026/02/24 11:33:25 WARNING mlflow.models.model: `artifact_path` is deprecated. Please use `name` instead.
2026/02/24 11:33:26 WARNING mlflow.sklearn: Saving scikit-learn models in the pickle or cloudpickle format requires exercising caution beca
2026/02/24 11:33:26 WARNING mlflow.sklearn: Saving scikit-learn models in the pickle or cloudpickle format requires exercising caution beca
use these formats rely on Python's object serialization mechanism, which can execute arbitrary code during deserialization. The recommended
 safe alternative is the 'skops' format. For more information, see: https://scikit-learn.org/stable/model_persistence.html
🏃 View run RandomForest at: http://localhost:5000/#/experiments/1/runs/79fbefb657834f2caf071ad1abb7e225
🧪 View experiment at: http://localhost:5000/#/experiments/1
Training complete. Check MLflow UI!
(venv) PS E:\MSC 4\MLOps\loan-MLflow-assignment\src>
```

6. Training Runs: All the models



7. Comparing 3 Runs in one Experiment:

7.1. Visualization: comparing MeanCV Score, Accuracy, F1-Score and AUC with hyperparameter – penalty



7.2. Run details and Metrics:

Run details

Run ID:	79fbefb657834f2caf071ad1...	94a83266863e460c9881c19...	40e23d708c93497f8c94c0a...
Run Name:	RandomForest	DecisionTree	LogisticRegression
Start Time:	02/24/2026, 11:33:24 AM	02/24/2026, 11:31:46 AM	02/24/2026, 11:33:14 AM
End Time:	02/24/2026, 11:33:33 AM	02/24/2026, 11:33:14 AM	02/24/2026, 11:33:24 AM
Duration:	9.2s	1.5min	9.6s

Parameters

Metrics

Show diff only

AUC	0.685	0.648	0.689
Accuracy	0.784	0.751	0.789
F1-Score	0.858	0.837	0.862
Mean CV Score	0.825	0.809	0.818

7.3. ROC Curve

Random Forest

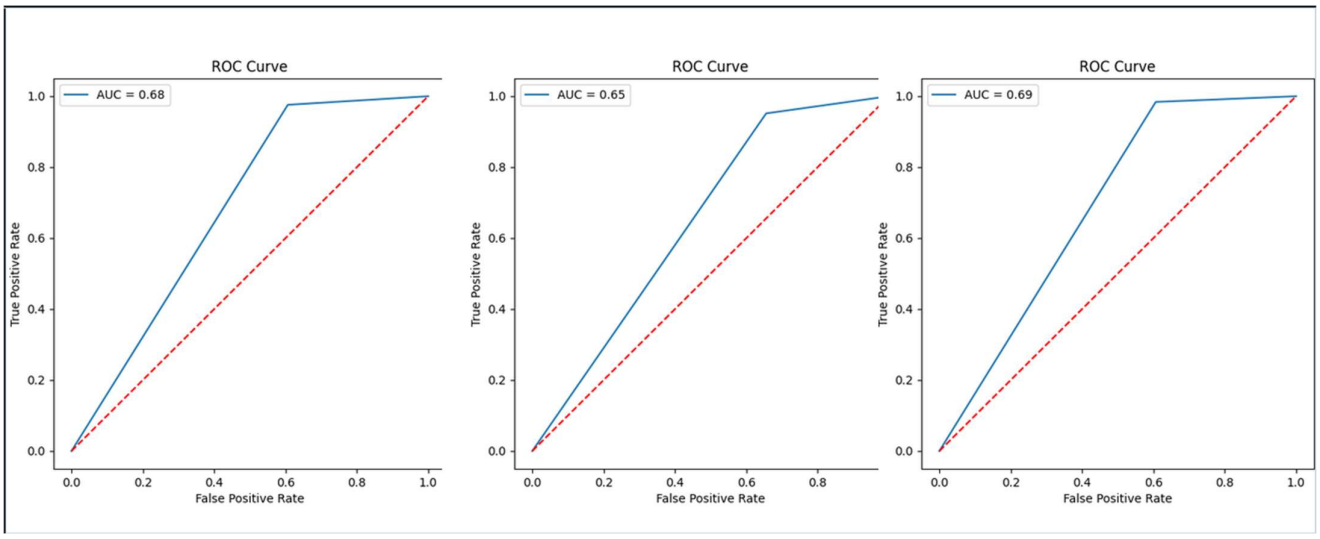
AUC = 0.68

Decision Tree

AUC = 0.65

Logistic Regression

AUC = 0.69



** From the above comparison, we come to a conclusion that **LogisticRegression** has best performance.

Hence, that is your final production model.