**Assignment 4**

**Name:** Snehal Laxmikant Yelwande  
**Class:** TY A CSE(AI)  
**Roll No:** 63  
**Batch:** P-2  
**PRN:** 22311760

**Problem Statement:** Automate preprocessing and training using **Apache Airflow** or **Prefect**.

**Theory**

**Need of Workflow Orchestration**

Machine learning projects involve multiple steps such as data collection, preprocessing, model training, evaluation, and deployment. Running these steps manually is error-prone and makes automation, monitoring, and reproducibility difficult.

**Workflow orchestration tools** like **Airflow** and **Prefect** help by:

* Automating pipelines (ETL, ML training).
* Scheduling jobs at specific intervals.
* Handling dependencies between tasks.
* Monitoring workflow execution.

**Apache Airflow**

* DAG-based (Directed Acyclic Graph).
* Widely used for data engineering workflows.
* Requires Linux/WSL2 or containers (not fully supported on Windows).

**Prefect**

* Python-first orchestration tool.
* Simpler to use, works cross-platform (Windows-friendly).
* Provides **Prefect Orion dashboard** for monitoring flows.

**Workflow with Prefect**

1. Define tasks (preprocessing, training, evaluation).
2. Combine tasks into a flow.
3. Run flow locally or schedule it.
4. Monitor execution in **Prefect Orion UI**.

**Executions**

**Step 1 – Install Prefect**

pip install prefect

**Step 2 – Define Prefect Flow**

**File: iris\_prefect\_flow.py**

from prefect import task, flow

from sklearn.datasets import load\_iris

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score

def load\_data():

iris = load\_iris(as\_frame=True)

return iris.data, iris.target

def split\_data(X, y):

return train\_test\_split(X, y, test\_size=0.2, random\_state=42)

def train\_model(X\_train, y\_train):

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

return model

def evaluate\_model(model, X\_test, y\_test):

preds = model.predict(X\_test)

acc = accuracy\_score(y\_test, preds)

print(f" Model Accuracy: {acc:.4f}")

return acc

def iris\_pipeline():

X, y = load\_data()

X\_train, X\_test, y\_train, y\_test = split\_data(X, y)

model = train\_model(X\_train, y\_train)

evaluate\_model(model, X\_test, y\_test)

if \_\_name\_\_ == "\_\_main\_\_":

iris\_pipeline()

**Step 3 – Run Flow**

python iris\_prefect\_flow.py

Output:

Model Accuracy: 1.0000

**Step 4 – Start Prefect Orion Dashboard**

prefect orion start

* Opens **Prefect UI** at http://127.0.0.1:4200.
* You can monitor task execution, logs, and schedules here.

**Output**

* Flow executed successfully with tasks logged.
* Accuracy printed in console.
* Prefect dashboard showed task dependencies and logs.

**Conclusion**

Automation of ML pipelines is critical for reproducibility and scalability.

* With **Prefect**, tasks (data loading, splitting, training, evaluation) were automated into a single workflow.
* The pipeline was monitored through the **Prefect Orion UI**, ensuring transparency and easier debugging.
* Prefect is lightweight, Python-friendly, and works well on Windows, making it a strong alternative to Airflow.

**Link to GitHub Repository:** [**MLOPS\_Assignments**](https://github.com/SnehalLY/MLOPS_Assignments)