

MLOPS Lab 2

Name : Ashiq Firoz
Roll : 2022BCD0013

Github Repository Link : <https://github.com/2022bcd0013-ashiq-firoz/lab2>

Job summary with metrics for all experiments

The screenshot shows the GitHub Actions interface for the repository '2022bcd0013-ashiq-firoz / lab2'. The 'All workflows' tab is selected, displaying a list of workflow runs. The left sidebar shows the 'Actions' menu with options like 'All workflows', 'Automated Training and Metric Report...', 'Management', 'Caches', 'Attestations', 'Runners', 'Usage metrics', and 'Performance metrics'. The main area shows four workflow runs, each with a green status icon, a title, a description, a branch, and a completion time.

Workflow Run	Status	Branch	Actor	Event	Status	Branch	Actor
Model-RandomForest, 100 trees, depth 15, testsplit-0.2, pre-process...	Completed	main	ashiq-firoz	Automated Training and Metric Reporting Using GitHub Actions #4: Commit a6bc63a pushed by 2022bcd0013-	16 minutes ago		
Model-RandomForest, 50 trees, depth 10, testsplit-0.2, pre-processin...	Completed	main	ashiq-firoz	Automated Training and Metric Reporting Using GitHub Actions #3: Commit e947969 pushed by 2022bcd0013-ashiq-firoz	18 minutes ago		
Model-Linear-lasso, regularize, testsplit-0.2, pre-processing - stand...	Completed	main	ashiq-firoz	Automated Training and Metric Reporting Using GitHub Actions #2: Commit 0ecd320 pushed by 2022bcd0013-ashiq-firoz	22 minutes ago		
Model-LinearReg, Default, testsplit-0.2, pre-processing - none	Completed	main	ashiq-firoz	Automated Training and Metric Reporting Using GitHub Actions #1: Commit ce35c65 pushed by 2022bcd0013-	32 minutes ago		

The screenshot shows the 'Summary' page for a workflow run. The left sidebar shows the 'Summary' tab selected, with options for 'All jobs', 'train (3.11)', 'train (3.12)', 'Run details', 'Usage', and 'Workflow file'. The main area displays the 'train (3.12) summary' and a 'Model Evaluation Summary (Python 3.12)'. The summary includes the name 'Ashiq Firoz', roll number '2022BCD0013', model 'Linear Regression', Mean Squared Error (MSE) of 0.5467, and R² Score of 0.2598. Below the summary, there is a table of artifacts produced during runtime.

Name	Size	Digest
training-artifacts-py3.11	1.29 KB	sha256:21fc88e2b89689dd43bc5c8c39be1b6f4351ab9c0b5b9dd19a...
training-artifacts-py3.12	1.29 KB	sha256:1d682b7a7d92df055ff2f2d1a1623d72ccb39ea9abfc3b85e2...

1. How did GitHub Actions improve experiment reproducibility?

GitHub Actions enforces a fixed execution environment, dependency versions, and scripted training steps. Every run is automatically triggered and executed identically on clean machines. This eliminates environment drift common in local or notebook-based experiments.

2. How easy was it to compare results across runs?

Comparison was straightforward since metrics were logged in a consistent JSON format and summarized per run. Each workflow execution stored artifacts and evaluation summaries. Python version matrices further enabled systematic cross-environment comparison.

3. What role does Git commit history play in experiment tracking?

Each experiment is tied to a specific commit hash, ensuring code–result traceability. Changes in performance can be directly mapped to code modifications. This creates an implicit, version-controlled experiment log.

4. What were the benefits of this approach compared to Lab 1?

Unlike Lab 1's manual notebook runs, this approach automated training, evaluation, and logging. It supported multiple models and environments without manual intervention. Results were reproducible, auditable, and centrally stored.

5. What limitations does this approach have?

GitHub Actions has limited compute resources and execution time. It is less interactive than notebooks for exploratory analysis. Advanced experiment management (e.g., hyperparameter sweeps) requires additional tooling.