Lab 2: Automated Training and Metric Reporting Using GitHub Actions

Objective

This lab introduces students to CI-driven machine learning workflows using GitHub Actions. Students will manually modify model code for each experiment, push changes to GitHub, and use GitHub Actions to:

- Automatically train the model
- Compute evaluation metrics
- Display metrics in the **GitHub Actions Job Summary**
- Store trained models and results as **workflow artifacts**

This lab demonstrates how **automation improves reproducibility and traceability**, even when experiment configuration is still manual.

Prerequisites

- Completion of **Lab 1**
- GitHub account
- Basic knowledge of:
  - Git/GitHub
  - Python
  - Regression model

Dataset: Use the **same Wine Quality dataset** from Lab 1.

Source: https://archive.ics.uci.edu/dataset/186/wine+quality

Evaluation Metrics

All runs must compute and report:

1. **Mean Squared Error (MSE)**
2. **R^2 Score**

These metrics must:

- Be printed by the training script
- Appear in the **GitHub Actions Job Summary**
- Be saved to a results file

Task 1: Create a GitHub repository

You are requested create a new GitHub account with username in the format *<roll_no>_<name>* (preferably using institute email id). Ex: 2022bcs0123_benthomas

Create a new public repository “lab2” and organize your project folder as follows

- Dataset directory
- Training script
- Requirements file
- Output directory for:
    - Trained model
    - Evaluation results (JSON)
- GitHub Actions workflow directory

Task 2: Training Script

The training script must:

1. Load the dataset
2. Apply pre-processing and feature selection
3. Train the selected model
4. Evaluate the model using:
    - MSE
    - R^2 Score
5. Save:
    - The trained model to a file
    - Evaluation metrics to a **JSON file**
6. Print metrics to standard output

Task 3: GitHub Actions Workflow

You must create a GitHub Actions workflow that:

1. Triggers on:
    - Push to the main branch
    - Pull request on main branch
2. Sets up:
    - Python environment
    - Required dependencies
3. Executes the training script
4. Captures output metrics
5. Writes evaluation results to the **Job Summary** (The summary should also contain your name and roll number)
6. Uploads artifacts

Note: Hyperparameters, pre-processing steps, and feature selection **MUST NOT** be controlled from the workflow file.

Task 4: Artifact Storage

Each workflow run must upload **artifacts**, including:

Required Artifacts

1. **Trained model file**
2. **Results JSON file** containing:
    - MSE
    - R^2 score

Artifacts must be downloadable from the GitHub Actions run page.

Task 5: Run Multiple Experiments

You must perform **all experiments as done in Lab 1** , where each experiment involves:

- Editing the training script (e.g., model type, hyperparameters, preprocessing)
- Committing the changes with a meaningful commit message (Eg. “Model- Lasso, alpha-0.1, testsplit-0.2, pre-processing – scalar”)
- Pushing to GitHub to trigger a workflow run

Each run represents one experiment.

Task 6: Analysis

Students must answer:

1. How did GitHub Actions improve experiment reproducibility?
2. How easy was it to compare results across runs?
3. What role does Git commit history play in experiment tracking?
4. What were the benefits of this approach compared to Lab 1.
5. What limitations does this approach have?

Deliverables

Students must submit:

1. GitHub repository link
2. Screenshot(s) showing:
    - Job summary with metrics for all experiments (should include your name and roll number)
    - Downloadable artifacts for all experiments
3. Answers to Analysis questions