**MLOps Assignment 1**

**Task M1 Summary**

**Description of the Work Completed**

This project involved the implementation of **CI/CD** pipelines using **GitHub Actions** to automate the processes of code validation, model building, and model deployment for a machine learning application.

The project focused on developing two key workflows: a Continuous Integration **(CI)** pipeline and a Continuous Deployment **(CD)** pipeline, each designed to enhance the development and deployment lifecycle of the machine learning model.

These strategic choices in the CI/CD pipelines align with best practices in software development and MLOps, promoting efficiency, reliability, and scalability of machine learning deployments.

|  |  |
| --- | --- |
| CI Pipeline: | The CI pipeline was set up to trigger on pull requests to the main branch. The workflow involved the following steps:   * **Code Linting:** Utilized **flake8** to ensure Python code quality by detecting syntax errors and enforcing coding standards. * **Model Building:** Automated the training of a machine learning model using the **RandomForestRegressor** and logged the best hyperparameters. * **Unit Testing:** Verified that the model's hyperparameters fell within an acceptable range, ensuring model consistency and reliability. |
| CD Pipeline: | The CD pipeline was designed to be triggered manually via **workflow\_dispatch** and involved:   * **Branch Validation:** Ensured that deployments only occur from the **main** branch. * **Model Training:** Re-trained the model to ensure that the latest code changes are included in the model version to be deployed. * **Docker Image Creation and Deployment:** Built and pushed Docker images of the trained model to **Docker Hub** for deployment, leveraging Docker's containerization capabilities. |

**Justification for the Choices Made**

|  |  |
| --- | --- |
| GitHub Actions for CI/CD: | * **Integration:** GitHub Actions provided seamless integration with our code repository, allowing for automated workflows triggered by pull requests and direct manual execution. * **Flexibility:** It allowed for customized workflows that could cater to the specific needs of our CI/CD processes, such as conditional branching and artifact management. * **Community and Support:** With extensive documentation and community support, GitHub Actions was a reliable choice for implementing our CI/CD workflows. |
| Model Validation and Docker Deployment: | * **Model Validation:** Validating model parameters ensures robustness and adherence to expected standards, thereby maintaining the quality of the deployed models. * **Docker:** Using Docker for deploying models encapsulates the application and its dependencies in a container, providing consistency across different environments and simplifying deployment and scaling processes. * **Docker Hub:** Serving as a centralized repository for our Docker images, Docker Hub facilitated easy sharing and deployment of model containers across various platforms. |