MLOps Assignment1 Group132

Team Members-

- ARJUNWADKAR AJAY NAVEEN SHAILAJA → 2023aa05074
- CHOPRA CHIRAG SURESH SANGEETA → 2023aa05520
- JAREENA BEGUM SHAIK → 2022ac05562
- NIKHIL SHARMA → 2023aa05539

Summary of work done:

We started with M1: MLOps Foundations, where we began with simple steps using a basic model from scikit-learn for RandomForestClassifier. We set up the conda environment for the project, extracted the environment config and requirements.txt. Here, we added a few steps like linting, testing, and sample deployment in the GitHub workflow. In the linting stage, we used flake8 to ensure the format and standards were followed while coding. We added a train.py in the scripts folder for training the RandomForestClassifier model and saved the model as a pkl file. We also added a test_train.py in the tests folder to check the saved model, which was used in the test stage of the GitHub CI/CD workflow.

For M2: Process and Tooling, we installed MLflow and DVC for experiment tracking along with data versioning. We used the RandomForest model and trained it multiple times, understood the MLflow concepts, and explored them on the MLflow UI. We also added a new row manually and started versioning the data using DVC. We used DVC commands like init, add, etc., to check how the checksum varies and how the data is tracked. We also added the logs for dataset versions in MLflow. Additionally, we experimented using nested loops for multiple cases with various hyperparameters.

For M3: Model Experimentation and Packaging, we used Scikit-learn's GridSearchCV to provide multiple hyperparameter sets, searching for and identifying the best combinations for RandomForestClassifier. We logged this in MLflow experiments, found the best model, and documented it. Afterward, we started packaging the model, executing it inside a Flask app, and creating a Docker build for running locally.

Finally, we updated the CI/CD workflow in GitHub Actions to include all the necessary steps. In the last stage, we started the app as part of the workflow, which required manual termination since the stages would not end automatically and would keep running.

Through this journey, we gained hands-on experience with MLOps concepts and tools like GitHub Actions, MLflow, DVC, and Docker, enhancing our understanding of continuous integration and deployment in machine learning projects. We deepened our knowledge of scikit-learn, explored hyperparameter tuning, and implemented best practices like linting, testing, and experiment tracking. This project also gave us insights into packaging and deploying ML models as services, highlighting the importance of automation and version control in maintaining reproducibility and efficiency in real-world machine learning workflows.