Javad Razi
j.razi@outlook.com

Homayoon Sadeghi
homayoon.9171@gmail.com

1. Overview

In this project, you will apply the concepts and techniques you have learned in the course to deploy a machine learning model into a production environment. You will use common MLOps best practices and automation tools and pipelines, including Flask or FastAPI for model deployment, Docker for containerization, and MLflow for experiment tracking and model management.
While we specify requirements in the project, you have the freedom to choose alternative methods or tools as long as your choice makes sense and you can justify it. This project is designed to give you hands-on experience in deploying a machine learning model into a production environment. We encourage you to be creative and make informed decisions based on sound reasoning.

2. Requirements

2.1. Model Deployment

You will be required to deploy your machine learning model as a web service using tools such as Flask or FastAPI. You should also use one of the common model deployment patterns and strategies covered in the course. You should justify your choice of deployment pattern and strategy based on the requirements and constraints of your specific project. In addition, you should consider factors such as scalability, maintainability, and cost when making your decision.

Bonus

- Implement end-to-end tests for your deployed machine learning application to ensure its functionality and performance.
- Implement version synchronization and model version metadata management for your deployed machine learning application to keep track of different versions of your model and data.

2.2. Automation

You will be required to implement some form of automation for your machine learning model deployment process. This could include steps such as containerization, continuous integration and continuous deployment (CI/CD), and automation of the ML workflow. Some popular tools for automating the deployment process include Docker, GitHub Actions, and MLflow.

1. **Containerization:** You will be required to containerize your machine learning application using Docker. This will make it easier to deploy and run your application in different environments.
2. **MLflow:** You will be required to use MLflow to track experiments and manage models for your project. This includes keeping track of different versions of your model, data, and code, making it easier to reproduce and share your work with others.

Bonus

- Use more features of MLflow for your project. This could include setting up a model registry or serving infrastructure, using a tracking server to centrally store and manage experiment data and model artifacts, or packaging code into reproducible runs and sharing models with others.
- Setup a continuous integration and continuous deployment (CI/CD) pipeline for your machine learning application using tools such as GitHub Actions. This will help you automate the process of building, testing, and deploying your application.

2.3. Monitoring

You will be required to use MLflow to monitor your machine learning application. This includes tracking performance metrics such as accuracy, precision, recall, and F1 score, as well as monitoring data quality. To track these metrics using MLflow, you will need to:

- Define the metrics that you want to track in your code.
- Log the metric values during training and evaluation.
- Use the MLflow ”Tracking UI” to visualize and compare the metric values across different runs.

Bonus

- Automate the process of retraining your deployed machine learning model. Utilize relevant performance metrics to identify when your model's performance has degraded. As soon as a targeted metric falls below a predefined threshold, trigger your model training pipeline automatically. This will enable you to rebuild your model using the latest training data, evaluate its performance to ensure improved results, and deploy the updated model to production. With this automation, you can ensure that your model is continually optimized to provide accurate and reliable predictions.
- Set up an alerting procedure for your machine learning application. This could include sending alerts when performance metrics fall below a certain threshold or when data quality issues are detected. Some popular tools for setting up monitoring and alerts for your deployed application include Prometheus, Grafana, and DataDog.
- Integrate MLflow with other tools such as TensorBoard for more advanced visualizations of your model's performance.

3. Deliverables

Note: Please remember that **it is a requirement** for this project that you implement your code in GitHub so that we can have access to your pipelines, actions, CI/CD procedures, codes, and commits.

3.1. Deployed Machine Learning Application

You will be required to submit a link to your deployed machine learning application.

3.2. API Documentation

You will be required to provide the API documentation of your application that outlines how to utilize your machine learning model in a production environment. To streamline this process, we suggest utilizing tools like Swagger or Postman to automatically generate API documentation.

3.3. GitHub Access

You will be required to add the Github account of the course (@SharifMLSD) as collaborator to your GitHub project so that we can have access to your pipelines, actions, CI/CD procedures, codes, and commits.

3.4. Latest Commit Hash

You will be required to submit the hash of the git commit that corresponds to your deployed application.

3.5. Optional Demo Video

As an optional deliverable, you may choose to upload a short video (max 10 minutes) demonstrating your deployed machine learning application. This video can serve as a backup in case there are any issues during the live presentation of your application. By providing a recorded demonstration of your project, you can ensure that we have an alternative way to assess part of your work if things don't go as planned during the live presentation.

3.6. Project Report

You will be required to submit a project report that includes the following sections:

1. **Deployment:** A description of how you deployed your machine learning model, including your chosen deployment pattern and strategy, and justification for these choices. This section should also include a discussion of any trade-offs or challenges you faced when making your deployment decisions.
2. **Automation:** A description of the automatic deployment pipeline you set up for your machine learning model, including details on containerization, CI/CD, and automation of the ML workflow. This section should also include a discussion of any tools or technologies you used to automate the deployment process.
3. **Monitoring:** A description of how you set up monitoring for your deployed machine learning application, including details on monitored metrics and feedback loops.
4. **Challenges and Trade-offs:** A discussion of any challenges or trade-offs you faced during the deployment phase of the project, including any dead-ends encountered and how you overcame them.

4. Final Submission

For your final submission, please submit the specified files as a zip file with the format ”Phase3-StdNo1-StdNo2-StdNo3.zip”. Please make sure that the zip file is named correctly and contains all these required files:

- A report file containing your project report.
- The files comprising your API documentation. If the API documentation is available online, please provide a link to it.
- A text file that includes the following information:
  - On the first line, include the link to your web-application.
  - On the second line, include the link to your project’s GitHub repository.
  - On the third line, include the Git commit hash that corresponds to the version of your application deployed in the production environment.
- An optional demo video (max 10 minutes) of your project.

5. Tips

- Start with a simple deployment pipeline and gradually add complexity as needed. This will help you avoid over-engineering your solution and make it easier to debug issues when they arise.
- Deployment can be a complex process with many moving parts. To make it more manageable, try breaking it down into smaller, more manageable steps. This will help you stay organized and focused, and make it easier to track your progress.
- There are many sample MLOps projects available on GitHub that can provide valuable insights and inspiration for your own project. Some repositories that contain sample MLOps projects for the deployment section include MLOps Platform Skeleton, mlflow-example and awesome-mlops.

6. Extra Reading Material

Here are some additional resources that may help you complete this project:

- Continuous Delivery for Machine Learning
- MLOps: Continuous delivery and automation pipelines in machine learning
- Monitoring Machine Learning Models in Production
- A/B Testing for Machine Learning Models