

MLOPS Project report

DVC Pipeline



i200699 Hammad

i200820 Usman

I202440 WALEED

Data Acquisition

* We generated a dataset from the given script.
* The data was stored in a csv and maintained by DVC.
* The DVC created a hash of contents of file and stored it in the google drive for remote access.
* It also helped version control the data.
* The credentials of google drive were stored in the google service cloud and a json file was used to access them.

Data Pre-processing

* We normalized and scaled features by converting timestamps into numeric values and dropping irrelevant rows.
* The data was split into training and validation sets.

Model Training

* We chose random forest for training the model and Grid Search CV for tuning the model.
* The DVC pipeline was made in dvc.yaml where all steps of the pipeline were defined and used to train this model on the prepared dataset and extract results.
* Best prams were used to identify the best hyperparameters for optimal performance.
* These parameters and metrics are then tracked using MLflow library and connecting MLflow client with the MLflow server.
* The best model is searched among all experiments keeping in mind the metrics calculated of the model. The best model is then registered for the production stage.

Deployment

* The best model is then packaged into a docker container.
* A dockerfile is used to define the commands for docker to execute on the relevant folders and files and a docker.yaml is used to define the workflow.
* The docker image is pushed to Docker Hub for easy accessibility.
* The credentials of Docker Hub were stored in GitHub secrets.
* A flask app is used to upload a csv file. In this way, the model is tested on the live data and serves predictions using the deployed model.

Concept Drift Monitoring

* A concept drift monitoring system can be implemented to observe the model’s performance over time.
* In a monitoring system, a model’s accuracy over time can be observed after each retraining on the appended dataset.
* In this way it can be observed whether the model training is going in the right direction or not by tracking the relevant drift metrics.
* Automated retraining triggers based on observed drift can be easily set up using an if-statement and defining thresholds for the metrics.