MLOps Zoomcamp

**Module 1:**

**Session 1.3: Training a ride duration prediction model**

16/05/2023

* **DictVectorizer()**

Transforms lists of feature-value mappings to vectors.

This transformer turns lists of mappings (dict-like objects) of feature names to feature values into Numpy arrays or scipy.sparse matrices for use with scikit-learn estimators.

* Imp commands:

!pip install pyarrow

from sklearn.feature\_extraction import DictVectorizer

read parquet data files do feature extraction and train linear regression and lasso regression

**Session 1.4: Course overview**

22/05/2023

* As we get again in the notebook we don’t know which model was better, hence experiment tracking and model registry is very important for model tracking and for that we use MLflow.
* We created function for reading files, but if we want to change the data we have to go back again into the directory. We will learn machine learning pipeline.
* Steps will be :

1. Load and prepare the data
2. Vectorise (convert the data into feature matrix
3. Train the model

So we can convert all these steps into ML pipeline

1. The output of the pipeline is ML model; we use the model before putting it into a Machine learning service (Web service), but there are other ways to deploy the model.
2. Model monitoring, if model is not performing properly we will get an alert. So we go back to our pipeline and execute it with new data. So this pipeline will produce a new model we will use into the service. Highest automation level. Automatic training and model deployment we need to develop such systems.
3. We will also learn processes involved.

**Session 1.5 : MLOps Maturity Model**

23/05/2023

* There is a level from0 to 4 for MLOps, where 0 means no MLOps and 4 means full MLOps automation
* **Level 0: No MLOps automation at all:**
* A jupyter notebook, no proper pipelining, no experiment tracking, no automation
* **Level 1: DevOps but no MLOps:**
* In this level there is already automations, so there are experienced developers helping the data scientist
* So in this case it is possible to deploy the model
* Releases are automated
* Unit & integration tests, CI/CD and other process but they are not ML aware.
* It is good for engineering background, but we don’t have experiment tracking, no reproducibility, DS are separated from engineers.
* **Level 2: Automated training**
* There is training pipeline or we can say a python script which you can run
* Experiment tracking, and we know which models are in the production so we have kind of model registry
* Training is automated but deployment is not automated but kept simple
* Low frequency deployment
* DS work with engineers
* **Level 3: Automated deployment**
* Easy to deploy model
* Here we mostly have ML platform where we have to make the API call, so model will be available on the location.
* Data preparation- model training- deploy the model
* A/B tests
* Model monitoring
* **Level 4: Full MLOps automation**
* So observe the model, we make adjustements ,retrain the model
* Deploy the model, automatically deploy
* A/B tests
* Highest level of automation

**When do we need to be at each level?**

* if we are working on a proof of concept(POC) we do not require to be on level 4

so no automation required for level 0, code can be written in jupyter (POC requirement can be satisfied with level 0)

* not all of the services need to be at level 4

most of the models require level 2

**Module 2:**

**Session 2.1: Experiment Tracking Intro**

24/05/2023

**Important concepts:**

ML experiment: the process of building an ML model

Experiment run: each trial in an ML experiment

Run artefact: any file associated with an ML run

Experiment metadata

**What’s experiment tracking?**

* Experiment tracking is the process of keeping track of all the relevant information from an ML experiment:
* Source code
* Environment
* Data
* Model
* Hyper parameters
* Metrics

**Why Experiment tracking is so important?**

* Reproducibility
* Organization
* Optimization

**MLflow:**

* It is an open source platform for the machine learning lifecycle
* It’s just a python package that can be installed with pip, and it contains four main modules:
* Tracking
* Models
* Model registry
* Projects

**Tracking experiment with MLflow:**

The MLflow module allows you to organize your experiments into runs, and to keep a track of:

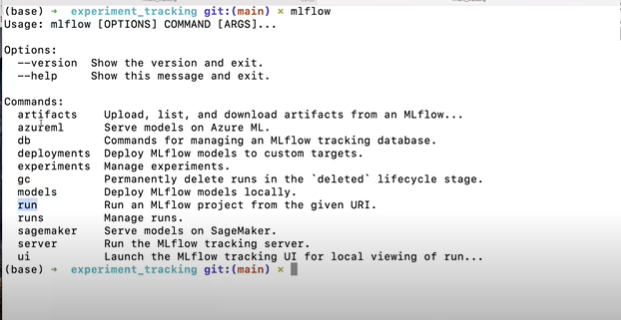
* Parameters (e.g.: hyper parameters, path to the training data)
* Metrics (any metric)
* Metadata (any information related to experiment, facts)
* Artifacts (any file)
* Models

Along with this information, MLflow automatically logs extra information about the run:

* Source code
* Version of the code (git commit)
* Start and end time
* Author

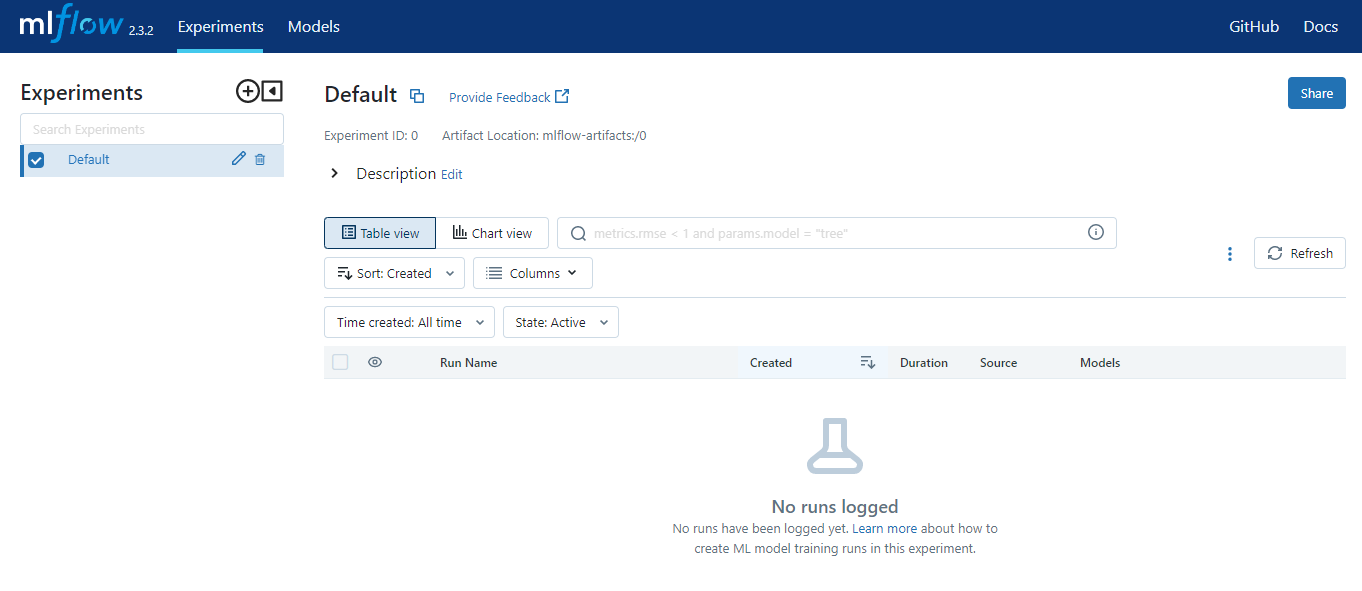
(need to install mlflow)

!pip install mlflow

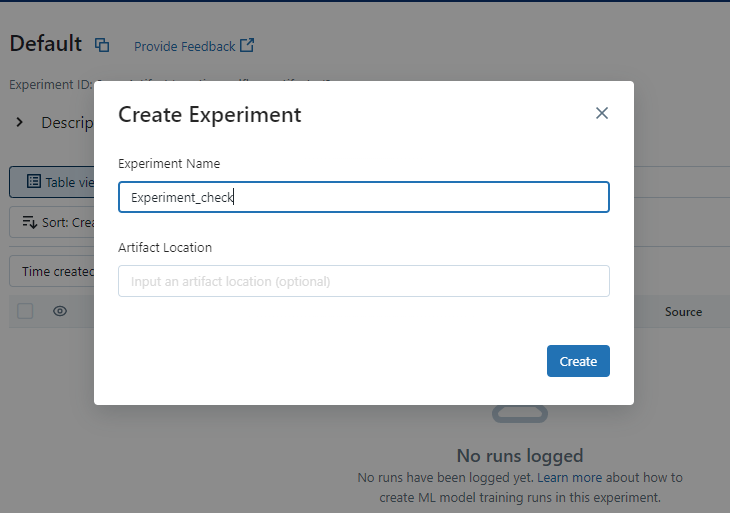


* If run the command **mlflow ui**

We get serving on launching url, and using that we get following mlflow ui

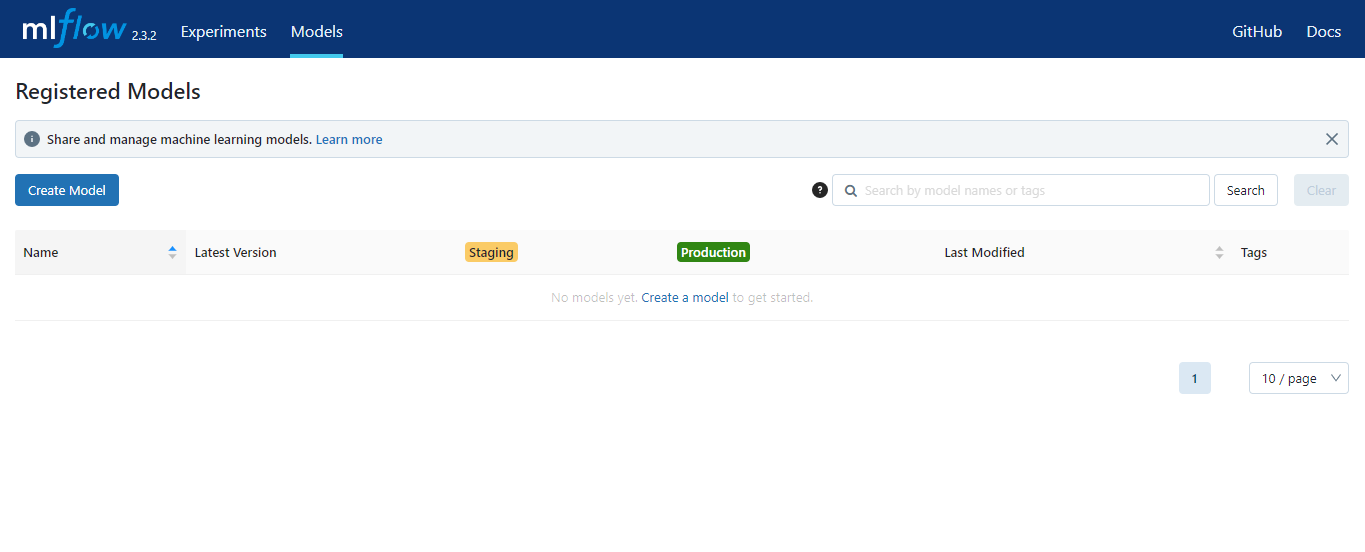


While creating an experiment



Artefact location could be any locations where your files are, it could be a local folder or s3 buckets

There is another tab dedicated to models which is known as model registry



**Session 2.2: Getting started with MLflow**

24/05/2023