MLOps - NLP - Case Study

Q1 System design: Based on the above information, describe the KPI that the business should track.

KPIs are quantified measurements of factors affecting a company's objectives. KPIs, basically, embody a focus on the strategic and operational improvement of a business objective or target.

Currently, if you need to extract diseases and treatments from free text, a trained person with clinical knowledge must manually look at the clinical notes and then pull this information. A data entry team would manually look at the clinical text and extract information about diseases and their treatment data. A data validation team would validate the extracted information. This process is prone to errors, and as the data increases, the extracted information. This process is prone to errors, and as the data increases, the data-entry team's size would need to be scaled up. Automating this process would result in reduced man-hours. The data-entry team would not be required. The data validation team would still need to perform validation on extracted data. It would also significantly reduce manual errors.

Based on the above information, business should track the KPI as **Reduction of manual hours & manual errors.**

Q2. System Design: Your company has decided to build an MLOps system. What advantages would you get by opting to build an MLOps system?

MLOps system increases the productivity of data scientists and machine learning engineers. There are many repetitive tasks in ML modelling. MLOps stand for automating the entire workflow of the ML model. This saves time and avoids human-induced errors.

Following are the benefits of applying MLops system-

1. **Code, artifact and experiment tracking**- As a data scientist we can Bridge the gap between model building and model deployment tasks. We can build a Repeatable process. Can focus on collecting, organising & tracking model training information.

2. **Continuous integration and deployment**- we can Provide end-to-end traceability so that we can achieve Improvement in time to reach the market.

3. **Continuous training and model monitoring**-Continuous training is an aspect of machine learning operations that automatically and continuously retrains machine learning models to adapt to changes in the data before it is redeployed. Model monitoring enables your AI team to identify and eliminate a variety of issues, including bad quality predictions and poor technical performance. As a result, your machine learning models deliver the best performance. Monitoring models effectively is very important for making your machine learning service successful. Continuous delivery and monitoring Improve time to market.

4. **Significant amount of data** is generated through the BEHEALTHY APP.

5. **Reduced lag in development & deployment stage**-It helps in acquiring, cleaning setting up tracking & versioning for experiments, we can set the deployment.

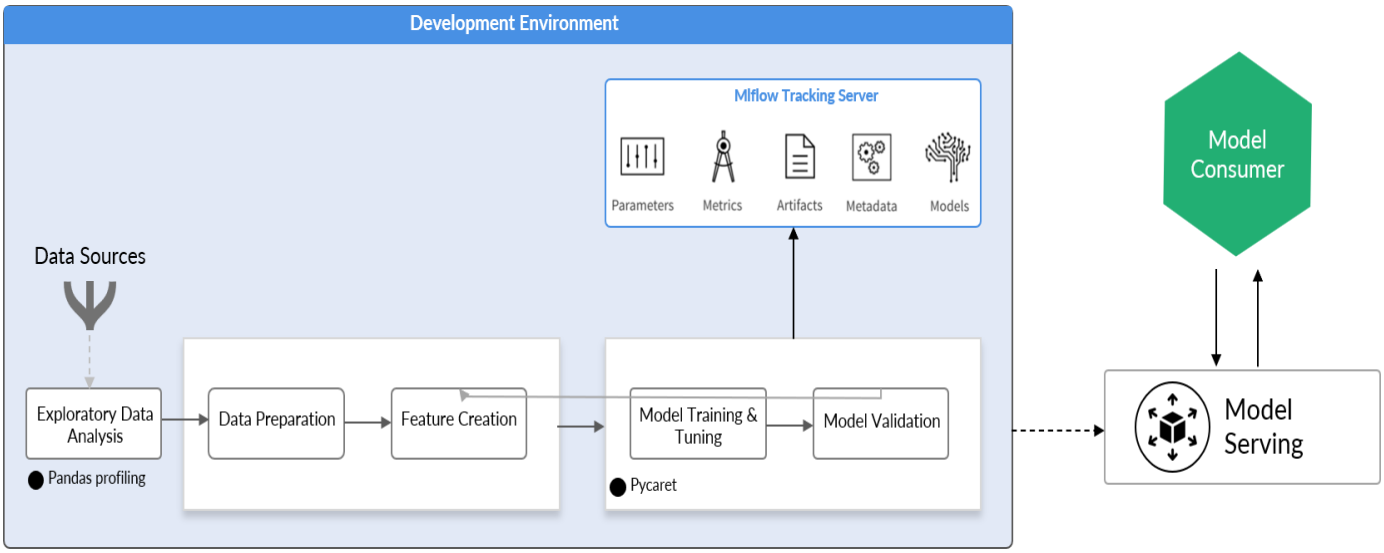
6.Real time analytics.

Q3. System design: You must create an ML system that has the features of a complete production stack, from experiment tracking to automated model deployment and monitoring. For the given problem, create an ML system design (diagram)

The design process of an ML system and the different phases of the MLOps life cycle is shown below. Each stage is critical as it helps us create a reliable, scalable, maintainable and adaptable ML system. The ML system depends upon the use case you are working with and thus varies along the business needs and infrastructure. Therefore, while designing an MLOps-based solution for a problem, you must understand the root problem and design its solution iteratively.

The system architecture is divided into two environments-

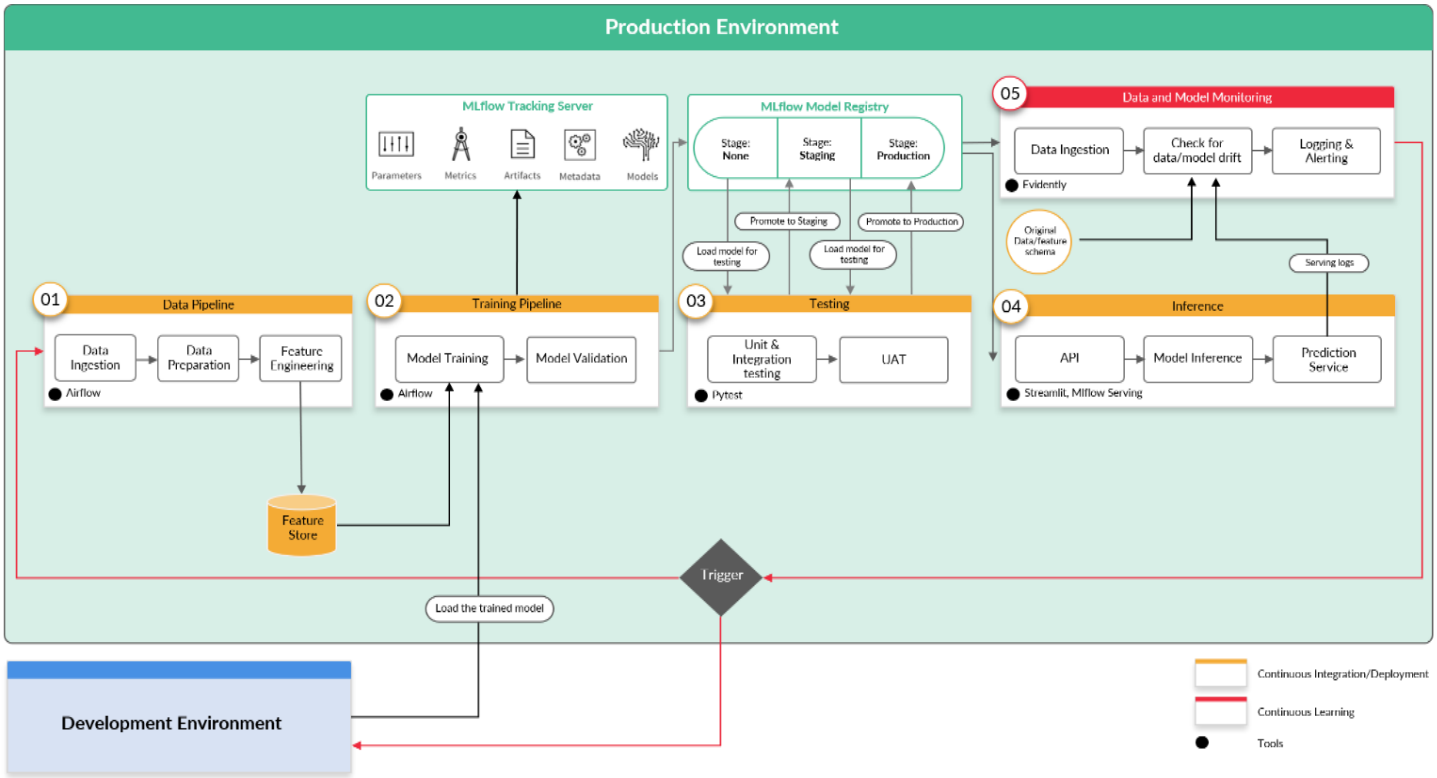
A. Development environment: This environment is structured to help you understand which model will perform the best for the given problem statement and data. This is an environment where you can perform rapid experimentation around data and models.



Model development in the development environment is an interactive process and broadly consists of the following steps:

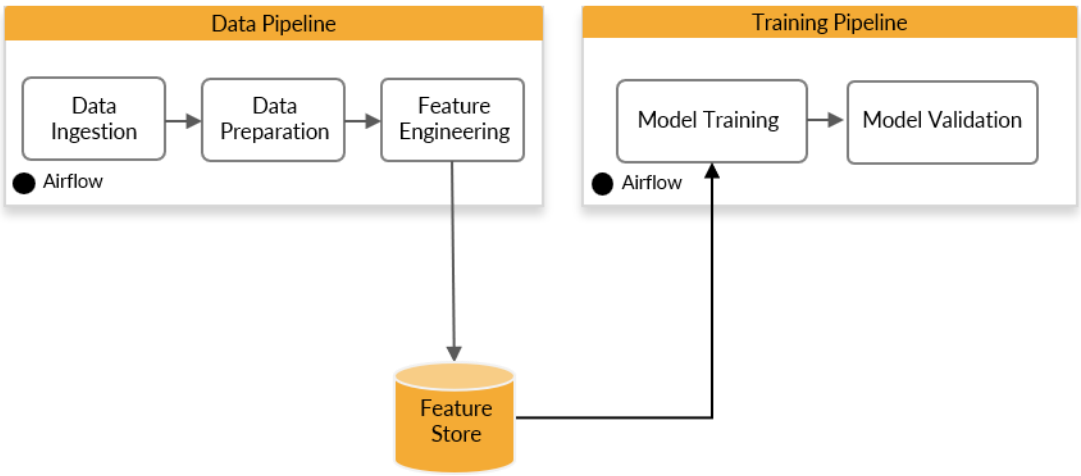
1.Exploratory data analysis
2.Data preparation
3.Model training and tuning- Rapid experimentation is essential to selecting the baseline model in this stage. Here, you can experiment with multiple models/algorithms; however, it is essential to strategise which models you want to select. For your use case, you will majorly work with classical ML models owing to their simplicity and explainability.
4.Model validation
5.Model serving

B. Production environment: This environment is where you operationalise the best model identified from the development environment/stage.

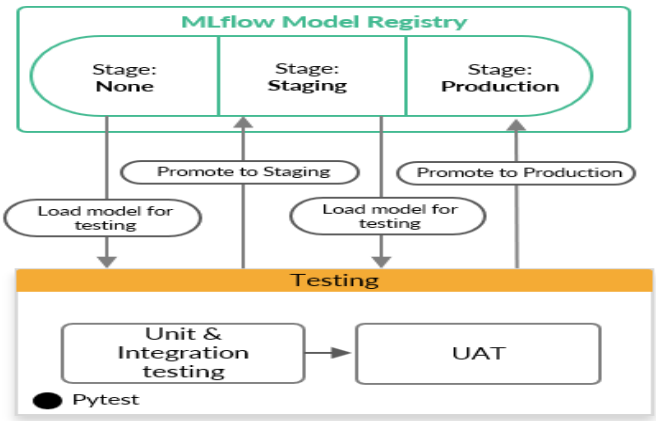


The components covered in the production environment are as follows:

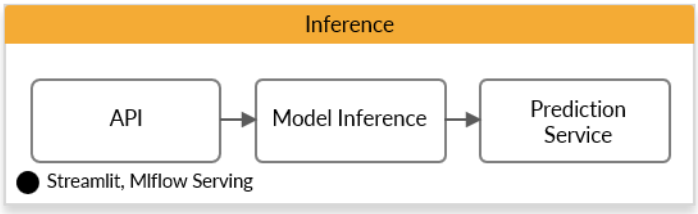
Data and training pipeline-



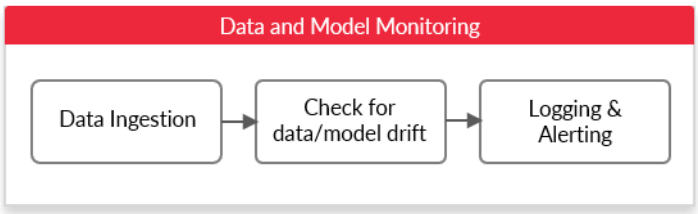
Testing-



Inference-



Data and model monitoring-



Q4. System design: After creating the architecture, please specify your reasons for choosing the specific tools you chose for the use case.

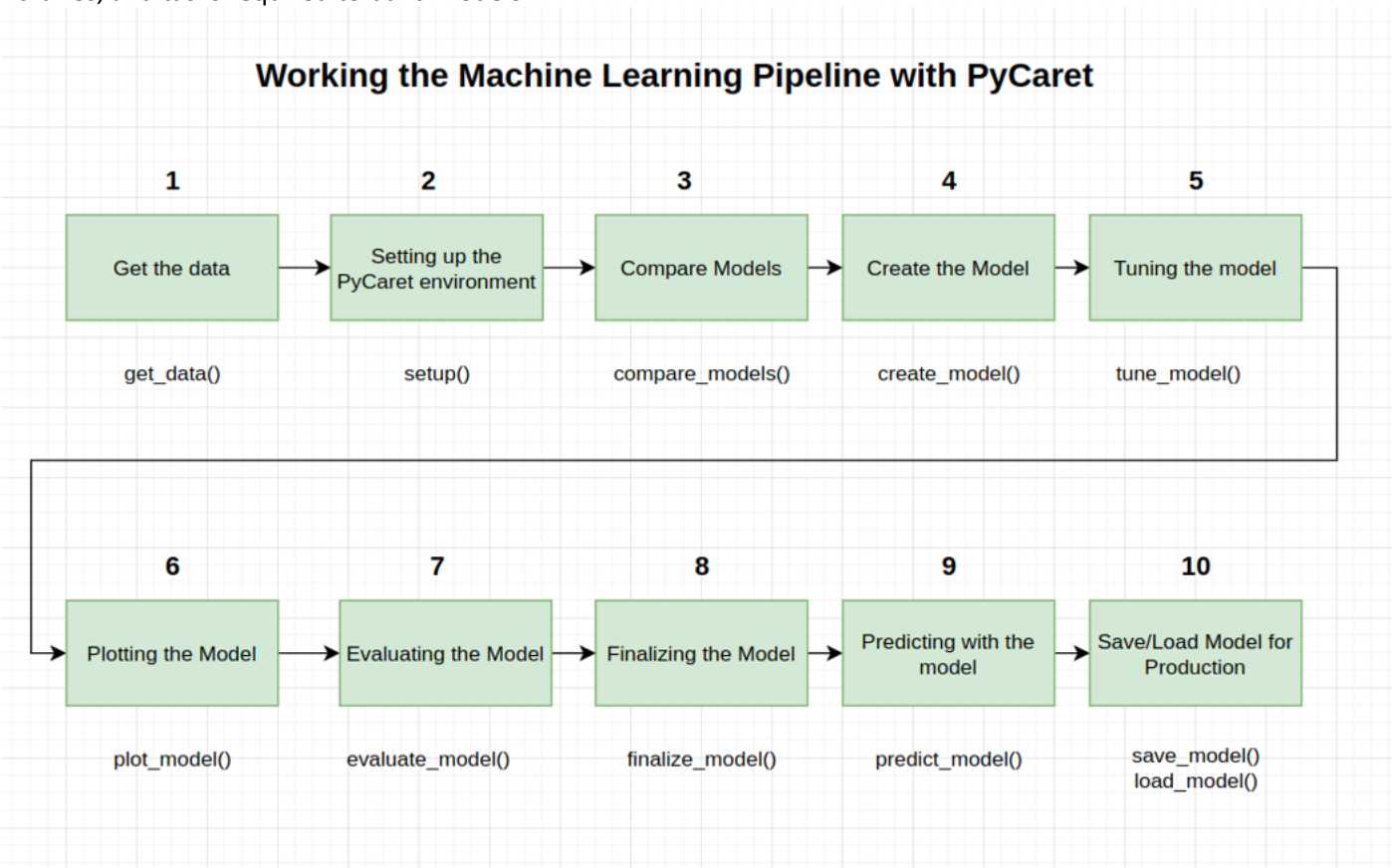
We must consider following parameters before choosing the specific tools,

Cost
Optimal model
Less complex
Robust model

We are using open-source platform to manage the ML lifecycle for our case study.

1. Automating Model Development using Pycaret-

The PyCaret library can be used as an approachable alternative, thereby reducing the number of lines of code, libraries, and tasks required to build models.



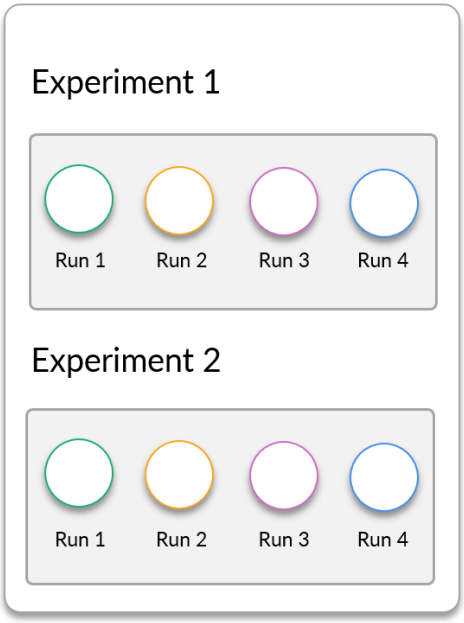
PyCaret is ideal for: -

Experienced Data Scientists who want to increase productivity.
Citizen Data Scientists who prefer a low code machine learning solution.
Data Science Professionals who want to build rapid prototypes.
Data Science and Machine Learning students and enthusiasts.
In the case study, you will use PyCaret for efficient model building and tuning as part of rapid experimentation in the development environment.

2.Run and Track Modelling Experiments using Mlflow-

MLflow is an open-source platform to manage the ML lifecycle, including experimentation, reproducibility, deployment, and a central model registry. Therefore, it is an excellent solution for tracking all your model development experiments. Broadly, these are the core advantages of using MLflow for your machine learning workflow: Tracking all your experiments: MLflow can track all the details of runs and experiments while working with your notebooks/scripts. You don't have to keep a manual log of all the metrics, hyperparameters, models, etc.

Managing all models in a Centralized Repository to: A data science team creates multiple models every day and without a central repository it is difficult to manage models stages: from development to staging, and finally, to archiving or production. MLflow helps in storing all the runs that you do in a centralized repository.



3.**For automating the workflow**- In the Behealthy case study inference process, we will be using Apache Airflow, an open-source orchestration tool widely adopted by the industry. We are using Apache Airflow as a tool for building complex pipelines in the form of directed acyclic graphs. Apache Airflow is a versatile tool using which you can,

Create manage pipeline
Manage pipeline
Monitor pipeline.
4. Evidently: Evidently AI is another open-source tool, which helps in evaluating and monitoring models in production

Q5. **Workflow of the solution**:

Data and model experimentation- (Pandas profiling, Pycaret, Mlflow)

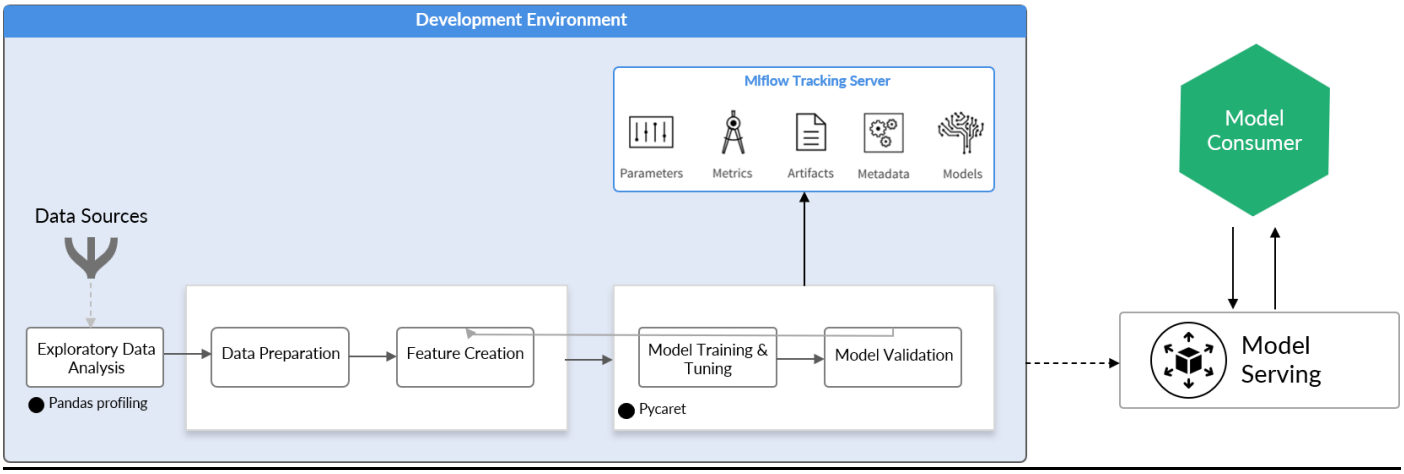
Automation of data pipeline —(Airflow)

Automation of the training pipeline –(Airflow)

Automation of inference pipeline –(Airflow)

Continuous monitoring pipeline- (Streamlit, Evidently)

<u>Data and model experimentation- (Pandas profiling, Pycaret, Mlflow)</u>



1)We have cleaned the raw data and performed EDA using pandas profiling, now we will move on to creating a model using pycaret.

2) There are many functions available in the library

setup ()

This function initializes the environment in PyCaret and creates the transformation pipeline to prepare the data for modelling and deployment. setup () must be called before executing any other function in PyCaret.

compare models()

Once the PyCaret setup has transformed the data, you can use this function 'compare_models()' to train and evaluate the performance of all the models/ estimators (using cross-validation) available in the model library of PyCaret

create_model()

The function compare_models() trains multiple models using cross-validation; however, you can use this function if you want to train one model

tune_model()

Any model that you create using the create_model() function is trained using the default hyperparameters. If you want to customize the hyperparameter tuning, you can use the tune_model() function.

3)MLflow is an open-source platform to manage the ML lifecycle, including experimentation, reproducibility, deployment, and a central model registry.

Install MLflow with Python's pip package manager

You can log metrics, hyperparameters, and models through MLflow Tracking API, using the following commands:

log_metric('metric_name')

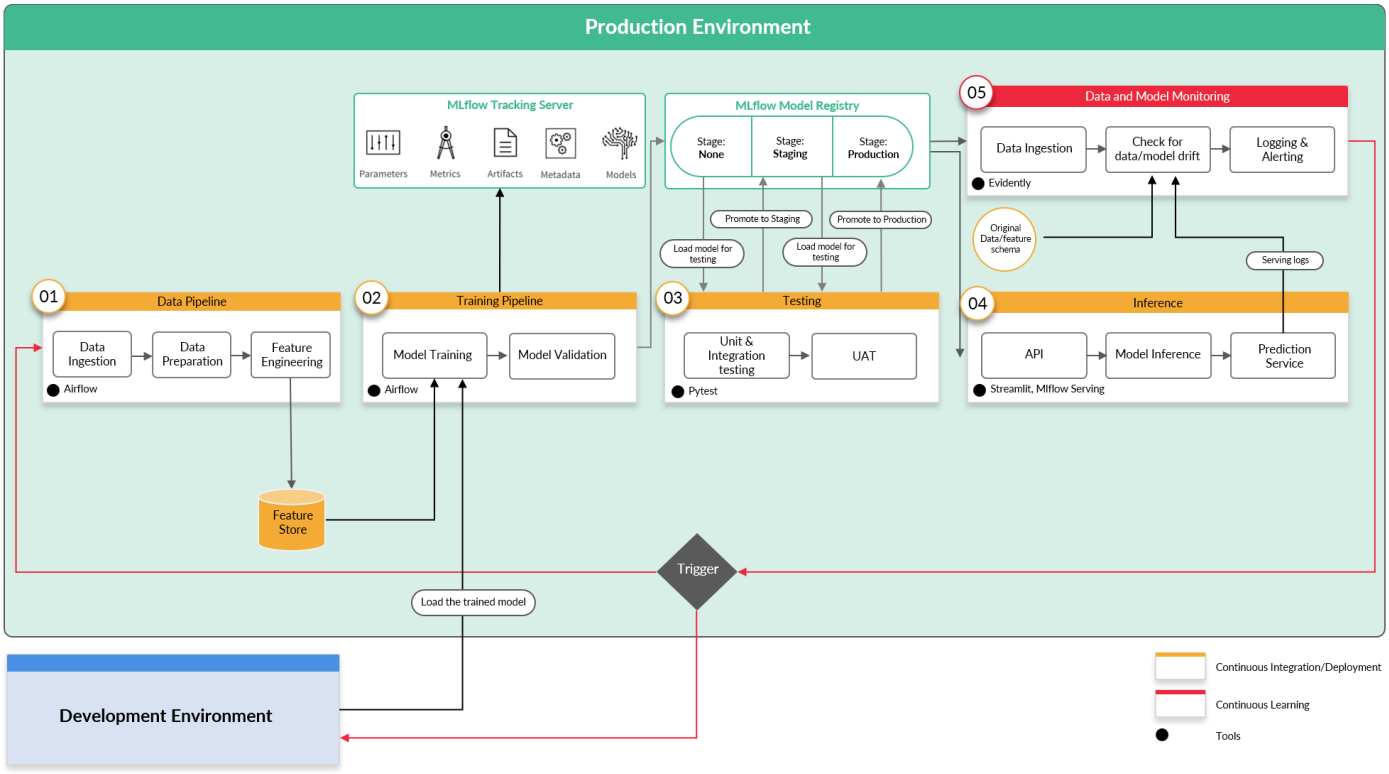
log_param('hyperparameters')

mlflow.sklearn.log_model('model_name')

4)Once logged, they will be automatically saved into a local ./mlruns directory.

5)The contents of runs and experiments can also be visualised using MLflow UI through the following command: The UI will automatically open in the local host at default port(5000): http://localhost:5000. Try exploring the different functionalities present in the UI.

Automation of data pipeline —(Airflow)



Pre steps for creating automated data pipeline-

1)First we will break down all the data cleaning tasks into functions in the utils.py file you have been provided with functions level breakdown of all the tasks.

2)Once you have converted the code into the respective functions. You will modify the function to read their input data and write their output to the database present in file 'utils_output.db'. 'utils_output.db' will be created once you execute the build_dbs function

3) create a test notebook and import utils file in it and run all the functions in the proper order. This will help you debug your code and create the 'utils_output.db' database

4) store all the constant values in constants.py

5)create a folder named 'unit_test'

6)Now go to the 'unit_test' folder and open the 'test_with_pytest.py' file

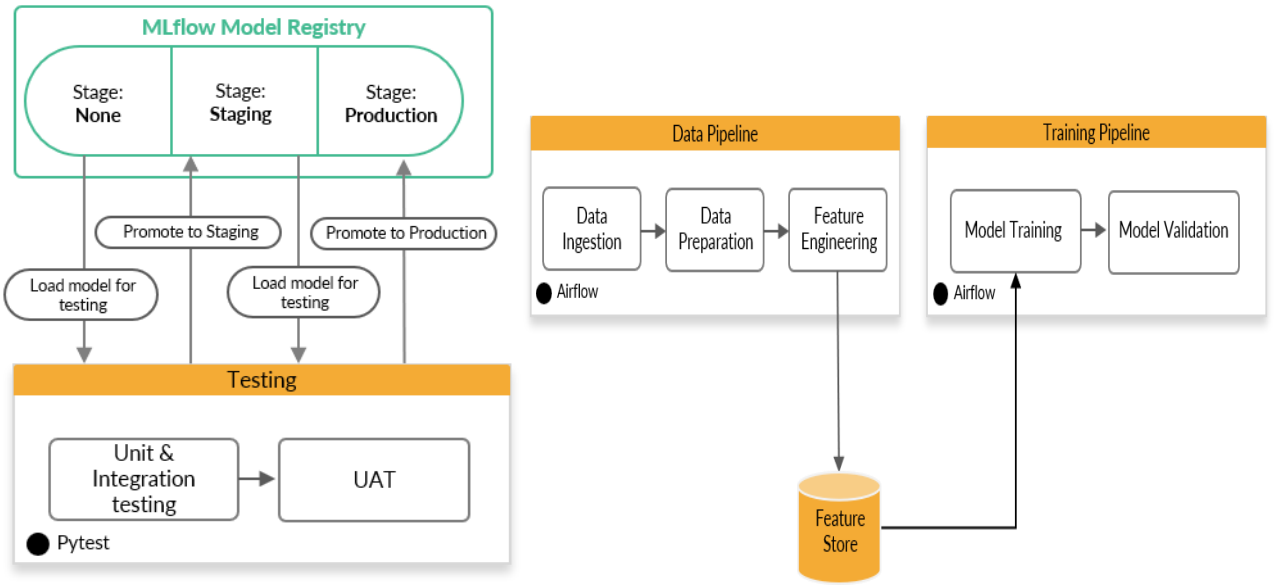
7)Save all the mapping files namely " city_tier_mapping.py", interaction_mapping.csv, significant_categorical_level.py in the 'mapping' folde

8)Now open 'data_validation_check.py'

9)Arrange all the files in proper order

Create automated data pipeline by connecting to server of Airflow through terminal. With this you should be able to run the data pipeline.

Automation of the training pipeline –(Airflow)



1)First we will break down all the training tasks into functions in the utils.py file

The function will be executed in the following order so make sure to read the input for the functions appropriately encode_features -> get_trained_model Set the mlflow experiment name.

2)Once you have modified the functions, create a test notebook and import utils file in it and run all the functions in the proper order. This will help you debug your code

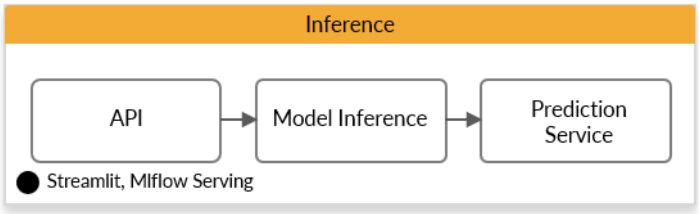
3)Once you have ensured that the functions in utils.py file are working as intended, store all the constant values in constants.py

4)The following file should be present in training pipeline

training_pipeline.py,constants.py ,utils.py

5)Create a training pipeline & run it.

Automation of inference pipeline –(Airflow)



1) First we will break down all the model inference tasks into functions in the utils.py file

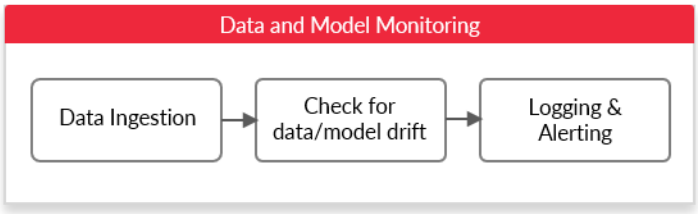
2)Once you have written the code into the respective functions. You will modify the function to read their input data and write their output to the database present in file

3)The function will be executed in the following order so make sure to read the input for the functions appropriately encode_data -> input_features_check -> load_model -> prediction_col_check

4)Once you have ensured that the functions in utils.py file are working as intended, store all the constant values in constants.py

6)Create a inference pipeline & run it

Continuous monitoring pipeline-(Streamlit,Evidently)



1)Evidently, known as an open-source Python library for data scientists and ML engineers, helps evaluate, test and monitor the performance of ML models and the status of data from validation to production.
2)Install the Evidently package and enable it to run properly.
3)Read the data
4)Generate the data drift report

Actions to be taken under the following conditions After deploying the model, we noticed that there was a sudden increase in the drift due to a shift in data

1. What component/pipeline will be triggered if there is any drift detected? What if the drift detected is beyond an acceptable threshold?

2. What component/pipeline will be triggered if you have additional annotated data?

3. How will you ensure the new data you are getting is in the correct format that the inference pipeline takes?

A typical continuous monitoring process consists of the following steps: -

-In the inference stage, a sample of the request-response (user-model) interaction during prediction serving is captured as serving logs.

-A comparison is made between the original data/ feature schema or distribution and the generated schema or distribution from the serving logs.

-If there is any deviation between the two schemas/ distribution, an alert is raised to the respective stakeholders to detect data drift.

The next action can be taken based on the deviation detected. This is configured in the component called ‘trigger’. To understand this, let’s imagine three different scenarios of measuring data drift by comparing the distribution of the Gender column in the new and the old data.

- 1-2% deviation: No action is needed as the deviation is not significant.

- 5-8% deviation: The data and model training pipelines are triggered for retraining the model with the new data.

- > 10% deviation: This can occur when the user behaviour changes drastically. In such scenarios, you must go back to the development environment to perform data and model experimentation.