

Optimal MLOps pipeline on GCP for loan Eligibility Prediction

Project Overview

Business Overview

Data Science has become a significant part of solving real-world problems in all domains. Many of the industries are investing in leveraging business value with the help of data science. Continuous integration of data, and Continuous deployment of models also known as CI-CD, assists the continuous operation of data science models in production.

While there are various ways to deploy the ML pipeline in production, industrialists still struggle to take advantage of such services from the wide variety of services available. In this project, we aim to deploy a machine learning model developed in the project [Loan Eligibility Prediction using Gradient Boosting Classifier](#) on the Google cloud platform(GCP) with a minimal number of services.

Aim

To build optimal MLOps pipeline on Google cloud platform to deploy loan eligibility prediction model in production

Tech Stack

- Language: Python
- Libraries: Flask, gunicorn, scipy, xgboost, joblib, seaborn, fancyimpute, scikit_learn
- Services: Flask, Docker, GCP, Gunicorn

Prerequisites

It is advisable to have a basic knowledge of the following services to better understand the project.

- Flask
- Docker
- Cloud Build
- Cloud Run
- Cloud Source Repository

Approach

Step 1:

1. Clone the repository
2. Create a Flask App (app.py)
3. Build a Dockerfile

Once the files are created, create a new repository and commit the changes. From here on, this will be your source repository. Proceed with the below steps

Step 2: Cloud Build Trigger

1. In your GCP console, create a new cloud build trigger
2. Point the trigger to your source repository

Step 3: Cloud Run

1. In Cloud Run, point the CI/CD server towards your cloud build trigger out
2. The output from cloud build will be in Artifacts Registry, which holds a docker image
3. Cloud run will provide an endpoint, a HTTPS URL that will serve the flask app created
4. Add the permission "allUsers" with roles as "Cloud Run Invoker" and save the changes
5. Once changes the change reflects, the HTTPS URL will be accessible

Project Takeaways

1. MLOps Architecture overview
2. Cloud architecture
3. Different cloud providers
4. Various deployment engines
5. Why Flask?
6. Building Flask API
7. What is Docker?
8. Building Docker image
9. Cloud Source Repository
10. How to create a Source repository and commit changes to it?
11. How to set up the git repository and commit changes?
12. Cloud Build
13. How to trigger Cloud build?
14. Cloud Run
15. Deployment using Cloud Run