# Explain what Machine Learning Engineering Is

Machine learning engineering, at its core, is the practice of putting machine learning applications into production environments and making them scalable. Similar to how DevOps engineers ensure that software remains stable in production environments and that new environments can be easily created, machine learning engineers ensure that machine learning applications and models always produce the correct results and can be quickly deployed to various environments.

Machine learning engineering draws its roots from DevOps. Many best practices of DevOps can be found within the realm of machine learning engineering. For example, using microservice and setting up CD / CI are best practices for both machine learning engineering and DevOps.

Because machine learning engineering shares many of its best practices with DevOps, it’s natural that another practice emerged, MLOps. MLOps is a combination of DevOps methodologies and machine learning. In essence, MLOps is closely related to DevOps in that one of its focuses is to productionalize code. In a word where more and more machine learning models are trained and applied, it is important to build in rail guards to ensure that models not only remain correct, but are also applied correctly and on the right data. For example, one problem of machine learning is data shift. Data shift used to not be a problem when the rate at which we collected data was slow. However, in today’s world, where data is being collected on everything at every second, data might gradually change after a machine learning model is applied in production. How to deal with constantly evolving data is one of the challenges that machine learning engineers and MLOps can help resolve.

# Create a simple docker file that runs a flask app. Explain how it works

I’m new to docker so I replicated the hello world docker example in Chapter 11 of the book. I was able to build the docker image and run it. I’ll explain how it works below.

The screenshot above is the app that the docker file will run. It prints the hello world message.

A screenshot of a computer

Description automatically generated with medium confidence

The screenshot below is the docker file. The docker file tells docker how to build a docker image. There are a few things worth noting about this docker file.

1. FROM python:3.7.3-alpine  
   This tells docker to use a prebuilt docker image for Python 3.7.3 based on the Alpine distribution
2. WORKDIR /app  
   This specifies the working directory of the docker image
3. COPY  
   The two COPY statements copy files in the folder to the docker image being built
4. RUN pip install -r requirements.txt  
   Installs all required packages in the docker image
5. ENTYPOINT, CMD  
   Tells docker which application to run and which file to run inside the application

Text

Description automatically generated

After building the docker image, I ran it as a container using this command:

docker run --rm -d -v pwd:/app -p 5000:5000 docker

Afterwards, I checked if the container is running properly, so I used curl to grab content in the url. The screenshot below shows the message I was expecting, so the container ran successfully.

