

Practical-6

Aim: Deployment of ML project in docker using Flask

Task 1: Ensure that the required libraries are installed:

- Install the Docker Command Line Interface Tool from:
<https://docs.docker.com/desktop/>
- Install Flask library (<https://flask.palletsprojects.com/en/2.3.x/installation/>)
- Install gunicorn library (<https://docs.gunicorn.org/en/latest/install.html>)

pip install Flask

pip install gunicorn

Task 2: Create the docker file using the steps described in theory material.

```
Practical 6 > Dockerfile > ...
1  # Use an official Python runtime as a parent image
2  FROM python:3.8-slim
3
4  # Set the working directory to /app
5  WORKDIR /app
6
7  # Copy the current directory contents into the container at /app
8  COPY . /app
9
10 # Install any needed packages specified in requirements.txt
11 RUN pip install --no-cache-dir -r requirements.txt
12
13 # Make port 80 available to the world outside this container
14 EXPOSE 80
15
16 # Run app.py when the container launches
17 CMD ["python", "app.py"]
18
```

Task 3: Create the docker image using docker build command.

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 6> docker build -t pr6 .
[+] Building 26.6s (18/10) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 507B
=> [internal] load metadata for docker.io/library/python:3.8-slim
=> [auth] library/python:pull token for registry-1.docker.io
=> [1/4] FROM docker.io/library/python:3.8-slim@sha256:3cb3eae0dfa00f89921c9e780618c515a7cbb5f0e0c531dc9b657cf9f155f3a66
=> [internal] load build context
=> => transferring context: 3.87kB
=> CACHED [2/4] WORKDIR /app
=> [3/4] COPY . /app
=> [4/4] RUN pip install --no-cache-dir -r requirements.txt
=> exporting to image
=> => exporting layers
=> => writing image sha256:229eeb5573892a648250973d55f8ec1e0c446ebf1634896caa6d2959a790dc2a
=> => naming to docker.io/library/pr6
What's Next?
View a summary of image vulnerabilities and recommendations -> docker scout quickview
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
pr6	latest	229eeb557389	3 minutes ago	372MB

Task 4: Run the docker container to execute the docker image and host the machine learning model using gunicorn wsgi server.

```
PS C:\Users\Nakul\Downloads\MLOPs\Practicals\Practical 6> docker run -p 4000:80 pr6
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 340-094-520
```

Task 5: Compare the performance of the model in docker container and flask script deployment.

The performance of Docker compared to local hosting can vary depending on several factors, and it's not always straightforward to determine whether Docker is faster or slower.

Docker might be faster due to optimized resource allocation, efficient caching, and consistent environments, especially if the Docker container is well-tailored for the application.

Docker might be slower if the overhead introduced by containerization is significant, or if there are misconfigurations impacting resource utilization.

Conclusion:

The decision on whether Docker hosting is faster or slower depends on the specific characteristics of your application and how well it is configured for Docker. It's recommended to perform detailed benchmarking and resource monitoring to draw accurate conclusions for your particular use case.