Step by step guide to learn and productionizing
ML Application using
Docker, MySQL

Open Terminal/Windows PowerShell - 1:

- Navigate(cd) to the **DevProRedCode** folder.

  cd <>/DevProRedCode

- Navigate(cd) to the **App** subfolder.

  cd App

- Check whether mysql image is locally available or not.

  docker images

- If mysql image is not listed, then pull mysql image from the docker hub.

  docker pull mysql

- Recheck for the images.

  docker images
  ls
  cd AppMySQL

- View the Dockerfile

- Build app_mysql image from Dockerfile

  docker build -t <name:tag> <dockerfile location>
  docker build -t app_mysql .

  docker images

- Run: Create and Start the container

  Docker run

-d : Run container in background and print container ID
-i : Interactive
-p : Map port in the container to port on the Docker host.
e.g. -p <Docker host port>:<Container port >
-t : Allocate a pseudo-TTY
--mount : Attach a filesystem mount to the container
E.g. --mount <Docker host path>:<Container path>
Usually Container path will be container's WORKDIR
Note: Mount dependent on host machine directory structure

--name : Assign a name to the container

docker run -p 3306:3306 --mount type=bind,source=/home/jeevan/Desktop/DevProRedCode/App/AppMySQL/,target=/AppMySQL --name App_MySQL -d app_mysql

(or)

docker run -p 3306:3306 -v /home/jeevan/Desktop/DevProRedCode/App/AppMySQL/:/AppMySQL --name App_MySQL -d app_mysql

- List running containers

  docker ps

- Find the container IPAddress using **Inspect**.

  docker inspect App_MySQL

- Runs a new command in a running container.

  -i : interactive
  -t : Allocate a pseudo-TTY
  docker exec -it App_MySQL bash

- Check whether cust_data.dump is there in the current folder or not.

  ls

  Note: If the file is not present, problem is with the volume mapping/mount

- Connect to mysql

  mysql -u <username> -p<password>

  mysql -u root -pinsofe

- Show databases

  ```
  show databases;
  ```

- Create cust_db database if doesn't exist

  ```
  create database cust_db;
  show databases;
  ```

- Change database to cust_db

  ```
  use cust_db;
  show tables;
  ```

  exit -> This is to come out of MySQL

- Create bank table and populate the data using cust_data.dump file

  ```
  mysql -u root -pinsofe cust_db < cust_data.dump
  ```

- Connect to mysql

  ```
  mysql -u root -pinsofe
  ```

- Execute following commands

  ```
  use cust_db;
  show tables;
  select * from bank limit 5;
  select count(*) as NumRec from bank;
  ```

  exit -> This is to come out of MySQL

  exit -> This is to come out of the App_MySQL container

- Change directory to AppPython

  ```
  cd ../AppPython
  ```

- Build app_python image from Dockerfile

  ```
  docker build -t app_python .
  ```

- List the Docker images

docker images

- Create and Run the Docker container

docker run -p 1234:1234 --mount type=bind,source=/home/jeevan/Desktop/DevProRedCode/App/AppPython/,target=/AppPython --name App_Python -it app_python bash

(or)

docker run -p 1234:1234 -v /home/jeevan/Desktop/DevProRedCode/App/AppPython/:/AppPython --name App_Python -it app_python bash

- Check whether the notebooks folder is there in the current folder or not.

ls

Note: If the file is not present, problem is with the volume mapping/mount

Open Terminal/Windows PowerShell - 2:

- List running containers

docker ps

- Inspect and identify the MLApp_Python container IP address

docker inspect App_Python

Observation: "IPAddress": "172.17.0.3"

Go to Terminal/Windows PowerShell - 1:

- Run jupyter notebook.

jupyter notebook --no-browser --ip=0.0.0.0 --port=1234 --allow-root

E.g. Open the browser and past following URL

http://0.0.0.0:1234/?token=b4e8dd99627d1b072da61ce27cd95c3f891407e02e81393b

(or)

http://172.17.0.3:1234/?token=b4e8dd99627d1b072da61ce27cd95c3f891407e02e81393b

(or)

http://127.0.0.1:1234/?token=b4e8dd99627d1b072da61ce27cd95c3f891407e02e81393b

Note: For Toolbox Installation type, only virtual box IPaddress has to be used

http://**192.168.99.100**:1234/?token=b4e8dd99627d1b072da61ce27cd95c3f891407e02e81393b

- Got to notebook directory and run 01_Python_MySQL.ipynb

  Press Ctrl+C and y

  exit -> To come out of Container

Open Terminal/Windows PowerShell - 2:

- Check whether required containers are running or not

  docker ps
  docker ps -a

- Navigate to AppMySQL folder

  cd AppMySQL

  - Runs a new command in a running container.

    docker exec -it App_MySQL bash

- Create bank table and populate the data using cust_data.dump file

  mysql -u root -pinsofe cust_db < cust_data.dump

- Connect to mysql

  mysql -u root -pinsofe

  use cust_db;

  select count(*) as NumRec from bank;

  exit -> To exit MySQL

  exit -> To exit container

- Inspect App_MySQL to find its ip address

  docker inspect App_MySQL

  Note: Change the ip address in the notebook accordingly

- Inspect App_Python to find its ip address

  docker inspect App_Python

- Open Notebook in the browser
- To delete the docker containers – Use the commands very cautiously

  docker rm <container-id>

  To delete the docker containers

  docker rmi <image1,image2,image3>

********** BREAK **********

Go to Terminal/Windows PowerShell - 1:

- Navigate (cd) to MLApp subfolder in DevProRedCode folder

  cd <>/DevProRedCode/MLApp/

- Navigate to AppMySQL subfolder

  cd AppMySQL

- List available images

  docker images

- If app_mysql docker is not available, build it using the Dockerfile

  docker build -t app_mysql .
  docker images

- Run: Create and Start the container

  docker run -p 3306:3306 --mount type=bind,source=/home/jeevan/Desktop/DevProRedCode/MLApp/AppMySQL/,target=/AppMySQL --name MLApp_MySQL -d app_mysql

(or)

docker run -p 3306:3306 -v /home/jeevan/Desktop/DevProRedCode/MLApp/AppMySQL/:/AppMySQL --name MLApp_MySQL -d app_mysql

- List running containers

  docker ps

- Inspect App_MySQL container to find its IPAddress

  docker inspect MLApp_MySQL

  Observation: "IPAddress": "172.17.0.2"

- Runs a new command in a running container.

  docker exec -it MLApp_MySQL bash

- Connect to mysql

  mysql -u root -pinsofe

- Show databases

  show databases;

- Change database to cust_db

  use cust_db;
  show tables;

  exit -> This is to come out of MySQL

- Check whether cust_data.dump is there in current folder

  ls

- Create bank table and populate the data using cust_data.dump file

  mysql -u root -pinsofe cust_db < cust_data.dump

- Connect to mysql

  mysql -u root -pinsofe

- Execute following commands

```
use cust_db;
show tables;
select * from bank limit 5;
select count(*) as NumRec from bank;
```

```
exit   -> This is to come out of MySQL
```

```
exit   -> This is to come out of the MLApp_MySQL container
```

- Just confirm whether MLApp_MySQL container is still running or not.

```
docker ps
```

- Navigate to AppPython folder

```
cd ../AppPython
```

- List available docker images.

```
docker images
```

- If app_python image is not listed, then build app_python image from Dockerfile

```
docker build -t app_python .
```

- List the Docker images

```
docker images
```

- Create and Run the Docker container

```
docker run -p 1234:1234 --mount type=bind,source=/home/jeevan/Desktop/DevProRedCode/MLApp/AppPython/,target=/AppPython --name MLApp_Python -it app_python bash
```

(or)

```
docker run -p 1234:1234 -v /home/jeevan/Desktop/DevProRedCode/MLApp/AppPython/:/AppPython --name MLApp_Python -it app_python bash
```

Go to Terminal/Windows PowerShell - 2:

- List running containers

```
docker ps
```

- Inspect and identify the MLApp_Python container's IP address

  docker inspect MLApp_Python

  Observation: "IPAddress": "172.17.0.3"

- Inspect and identify the MLApp_MySQL container's IP address

  docker inspect MLApp_MySQL

  Observation: "IPAddress": "172.17.0.2"

Go to Terminal/Windows PowerShell - 1:

- Check whether reading the data MySQL and some pre-process functions are working as expected

  python Python_MySQL.py

- Read the data from MySQL, Pre-process, build the model and save everything as Pipeline

  python build.py

- Make Predict on test data in .csv file

  python predict.py

- To delete the docker containers – Use the commands very cautiously

  docker rm <container-id>

  To delete the docker containers

  docker rmi <image1,image2,image3>