

ML model deployment in Docker:

1. Creating Virtual Environment (venv)

```
E:\MSC 4\MLOps\Iris-docker-assignment>python -m venv venv  
E:\MSC 4\MLOps\Iris-docker-assignment>venv\Scripts\activate
```

2. Installing all the dependencies in requirement.txt

```
(venv) E:\MSC 4\MLOps\Iris-docker-assignment>pip install -r requirements.txt  
Collecting fastapi (from -r requirements.txt (line 1))  
  Downloading fastapi-0.129.0-py3-none-any.whl.metadata (30 kB)  
Collecting uvicorn (from -r requirements.txt (line 2))  
  Downloading uvicorn-0.40.0-py3-none-any.whl.metadata (6.7 kB)  
Collecting scikit-learn (from -r requirements.txt (line 3))  
  Using cached scikit_learn-1.8.0-cp312-cp312-win_amd64.whl.metadata (11 kB)  
Collecting pandas (from -r requirements.txt (line 4))  
  Using cached pandas-3.0.0-cp312-cp312-win_amd64.whl.metadata (19 kB)  
Collecting numpy (from -r requirements.txt (line 5))  
  Using cached numpy-2.4.2-cp312-cp312-win_amd64.whl.metadata (6.6 kB)  
Collecting joblib (from -r requirements.txt (line 6))  
  Using cached joblib-1.5.3-py3-none-any.whl.metadata (5.5 kB)  
Collecting starlette<1.0.0,>=0.40.0 (from fastapi->-r requirements.txt (line 1))  
  Downloading starlette-0.52.1-py3-none-any.whl.metadata (6.3 kB)  
Collecting pydantic>=2.7.0 (from fastapi->-r requirements.txt (line 1))  
  Downloading pydantic-2.12.5-py3-none-any.whl.metadata (90 kB)  
Collecting typing_extensions>=4.8.0 (from fastapi->-r requirements.txt (line 1))  
  Using cached typing_extensions-4.15.0-py3-none-any.whl.metadata (3.3 kB)  
Collecting typing_inspection>=0.4.2 (from fastapi->-r requirements.txt (line 1))  
  Downloading typing_inspection-0.4.2-py3-none-any.whl.metadata (2.6 kB)  
Collecting annotated_doc>=0.0.2 (from fastapi->-r requirements.txt (line 1))  
  Downloading annotated_doc-0.0.4-py3-none-any.whl.metadata (6.6 kB)  
Collecting click>=7.0 (from uvicorn->-r requirements.txt (line 2))  
  Downloading click-8.3.1-py3-none-any.whl.metadata (2.6 kB)  
Collecting h11>=0.8 (from uvicorn->-r requirements.txt (line 2))  
  Using cached h11-0.16.0-py3-none-any.whl.metadata (8.3 kB)  
Collecting scipy>=1.10.0 (from scikit-learn->-r requirements.txt (line 3))  
  Using cached scipy-1.17.0-cp312-cp312-win_amd64.whl.metadata (60 kB)  
Collecting threadpoolctl>=3.2.0 (from scikit-learn->-r requirements.txt (line 3))  
  Using cached threadpoolctl-3.6.0-py3-none-any.whl.metadata (13 kB)  
Collecting python_dateutil>=2.8.2 (from pandas->-r requirements.txt (line 4))  
  Using cached python_dateutil-2.9.0.post0-py2.py3-none-any.whl.metadata (8.4 kB)  
Collecting tzdata (from pandas->-r requirements.txt (line 4))
```

3. creating **train.py** from terminal and writing the code for the model to be trained on iris - data set.

```
(venv) E:\MSC 4\MLOps\Iris-docker-assignment>cd app  
(venv) E:\MSC 4\MLOps\Iris-docker-assignment\app>type nul > train.py  
(venv) E:\MSC 4\MLOps\Iris-docker-assignment\app>code train.py
```

4. Running train.py , saving the model and printing model's accuracy.

```
● (venv) PS E:\MSC 4\MLOps\Iris-docker-assignment\app> py train.py  
Model trained successfully!  
Accuracy: 1.0000
```

5. commanding docker build , which creates image from docker file and a “build context”

```
(venv) E:\MSC 4\MLOps\Iris-docker-assignment>docker build -t iris-docker-assignment .
[+] Building 7.9s (11/11) FINISHED
=> [internal] load build definition from dockerfile
=> => transferring dockerfile: 388B
=> [internal] load metadata for docker.io/library/python:3.10-slim
=> [auth] library/python:pull token for registry-1.docker.io
=> [internal] load .dockerrignore
=> => transferring context: 2B
=> [1/5] FROM docker.io/library/python:3.10-slim@sha256:e508a34e5491225a76fbb9e0f43ebde1f691c6a689d096d7510cf7fb
=> => resolve docker.io/library/python:3.10-slim@sha256:e508a34e5491225a76fbb9e0f43ebde1f691c6a689d096d7510cf7fb
=> [internal] load build context
=> => transferring context: 407B
=> CACHED [2/5] WORKDIR /code
=> CACHED [3/5] COPY requirements.txt .
=> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt
=> [5/5] COPY ./app ./app
=> exporting to image
=> => exporting layers
=> => exporting manifest sha256:a25aaaacc1c43ef768313af3de980e25e02accde0b7fa668c00b3bcef8591608f
=> => exporting config sha256:7727d7f37bb8c55ec906128620702d9de31cdc512e860a579e574d91097d01db
=> => exporting attestation manifest sha256:b667cf75159be9e9968769913ced5af66aa54d2f1d7d32e1a9048a8ee49e6c6e
=> => exporting manifest list sha256:b4d0fbalc5b51614bef8e10737369ea041443ebalba5693a9efc0f0a491f513
=> => naming to docker.io/library/iris-docker-assignment:latest
=> => unpacking to docker.io/library/iris-docker-assignment:latest

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/p7wzslqc6k0buprbbp0f5klq6
```

6. Docker file view in Docker



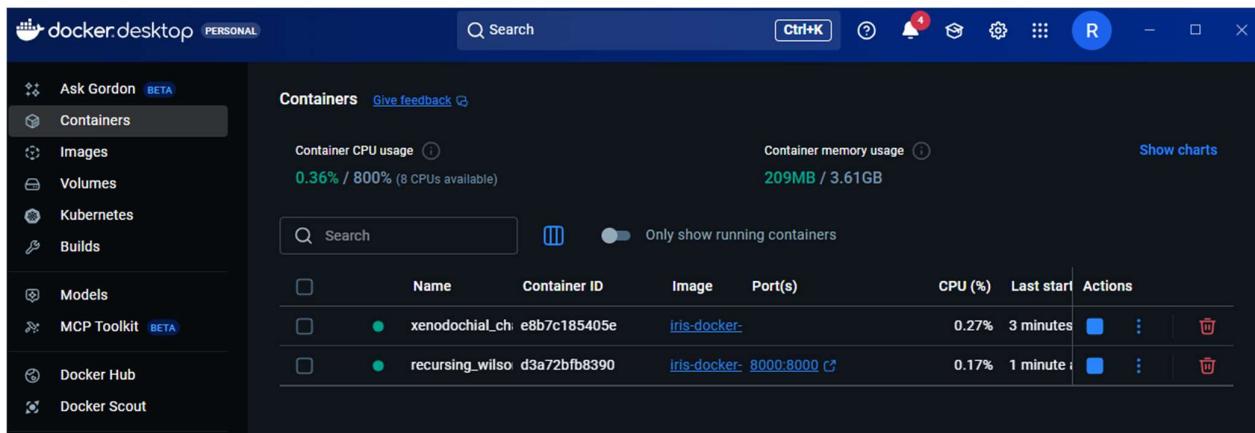
7. Commanding docker run: to create and start new container from a Docker Image

```
PS E:\MSC 4\MLops\Iris-docker-assignment> docker run -p 8000:8000 iris-docker-assignment
/usr/local/lib/python3.10/site-packages/sklearn/base.py:442: InconsistentVersionWarning: Trying to unpickle estimator DecisionTreeClassifier from version 1.8.0 when using version 1.7.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
    warnings.warn(
/usr/local/lib/python3.10/site-packages/sklearn/base.py:442: InconsistentVersionWarning: Trying to unpickle estimator RandomForestClassifier from version 1.8.0 when using version 1.7.2. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to:
https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
    warnings.warn(
    warnings.warn(
INFO:     Started server process [1]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO:     172.17.0.1:33054 - "GET / HTTP/1.1" 200 OK
INFO:     172.17.0.1:33064 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO:     172.17.0.1:42632 - "GET /docs HTTP/1.1" 404 Not Found
INFO:     172.17.0.1:48724 - "GET /favicon.ico HTTP/1.1" 404 Not Found
INFO:     172.17.0.1:48738 - "GET /docs HTTP/1.1" 200 OK
INFO:     172.17.0.1:48738 - "GET /openapi.json HTTP/1.1" 200 OK
INFO:     172.17.0.1:46192 - "POST /predict?sepal_length=2.3&sepal_width=1.3&petal_length=4.6&petal_width=0.2 HTTP/1.1" 200 OK
INFO:     172.17.0.1:47334 - "GET / HTTP/1.1" 200 OK
INFO:     172.17.0.1:47334 - "GET / HTTP/1.1" 200 OK
```

8. Browser view of the deployed Docker Image at port:8000

```
message: "Iris Classification API Running"
model_accuracy: 1.0 [35:1]
```

9. Containers in the Docker desktop, current running docker container



10. Accessing <https://192.168.34.1:8090/httpclient.html>, the default browser view, to get the prediction from the trained model, while providing the parameters.

FastAPI 0.1.0 OAS 3.1

default

POST /predict Predict

Parameters

Name	Description
sepal_length * required	number (query)
sepal_width * required	number (query)
petal_length * required	number (query)
petal_width * required	number (query)

11. Response results:

Responses

Curl

```
curl -X 'POST' \
-H 'accept: application/json' \
-d '' \
http://127.0.0.1:8000/predict?sepal_length=2.3&sepal_width=1.3&petal_length=4.6&petal_width=0.2
```

Request URL

```
http://127.0.0.1:8000/predict?sepal_length=2.3&sepal_width=1.3&petal_length=4.6&petal_width=0.2
```

Server response

Code Details

200

Response body

```
{ "prediction": 0,
  "model_accuracy": 1 }
```

Download

Response headers

```
content-length: 37
content-type: application/json
date: Mon, 20 Feb 2020 04:32:51 GMT
server: uvicorn
```

Responses

Code Description

200 Successful Response

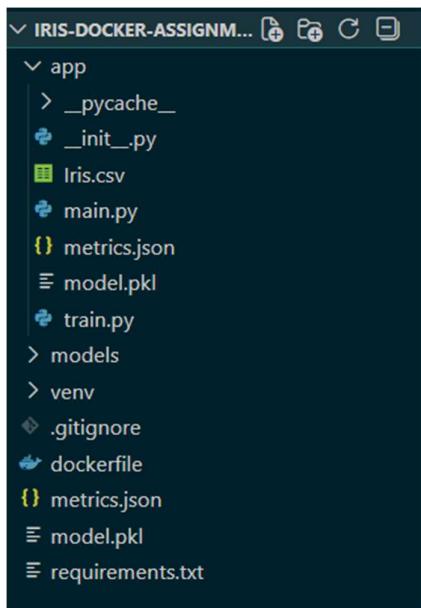
Media type

application/json

Controls Accept header.

Links

No links



App files:

- 1. Train.py
 - 2. Main.py
 - 3. Dockerfile

```
app > ⚡ main.py > ...
  1  from fastapi import FastAPI
  2  import joblib
  3  import numpy as np
  4  import json
  5
  6  app = FastAPI()
  7  model = joblib.load("app/model.pkl")
  8  with open("app/metrics.json", "r") as f:
  9      metrics = json.load(f)
 10
 11 @app.get("/")
 12 def home():
 13     return {
 14         "message": "Iris Classification API Running",
 15         "model_accuracy": metrics["accuracy"]
 16     }
 17
 18 @app.get("/accuracy")
 19 def get_accuracy():
 20     return metrics
 21
 22 @app.post("/predict")
 23 def predict(
 24     sepal_length: float,
 25     sepal_width: float,
 26     petal_length: float,
 27     petal_width: float
 28 ):
 29     data = np.array([[sepal_length, sepal_width, petal_length, petal_width]])
 30     prediction = model.predict(data)
 31
 32     return {
 33         "prediction": int(prediction[0]),
 34         "model_accuracy": metrics["accuracy"]
 35     }
 36
```

```
train.py  X  main.py  dockerfile  .gitignore

app > train.py > ...
1 import os
2 import json
3 import joblib
4 from sklearn.datasets import load_iris
5 from sklearn.model_selection import train_test_split
6 from sklearn.ensemble import RandomForestClassifier
7 from sklearn.metrics import accuracy_score
8
9 os.makedirs("models", exist_ok=True)
10
11 iris = load_iris()
12 X = iris.data
13 y = iris.target
14
15 X_train, X_test, y_train, y_test = train_test_split(
16     X, y, test_size=0.2, random_state=42
17 )
18
19 model = RandomForestClassifier(random_state=42)
20 model.fit(X_train, y_train)
21 y_pred = model.predict(X_test)
22 accuracy = accuracy_score(y_test, y_pred)
23 joblib.dump(model, "model.pkl")
24 metrics = {"accuracy": accuracy}
25 with open("metrics.json", "w") as f:
26     json.dump(metrics, f)
27
28 print("Model trained successfully!")
29 print(f"Accuracy: {accuracy:.4f}")

30
```