**Step 1: Structure Your Project**

A computer screen shot of a program

AI-generated content may be incorrect.

**Step 2: Create the requirements.txt File**

Only list needed package used in train.py

**Step 3: Create train.py File**

It is better to use several functions which is easier for following unit test.

In addition, must activate the environment and use python train.py to make sure it works. This is debugging with project related specific file, we will not discuss here. But one important thing is created a new folder as C:\hadoop\bin and use <https://downgit.github.io/#/home> download the folder: <https://github.com/cdarlint/winutils/tree/master/hadoop-3.3.6/bin>

Also add variable “HARDOOP\_HOME” as C:\hadoop and add path %HARDOOP\_HOME%\bin to path.

**“classWeight” and “label” must not be included in “features”!!!!!**

This is important for python train.py to work locally.

**Step 4: Create app\main.py under folder src and tests\test\_app\test\_main.py under main project folder.**

Now the folder looks like this:

A screenshot of a computer program

AI-generated content may be incorrect.

**Step 5: Run python -m pytest under the project root folder while under environment env**

Problem 1:

from src.app.main import app  
E ModuleNotFoundError: No module named 'src'

Solution: add \_\_init\_\_.py in folder src and src\app

Create file conftest.py in folder tests and the content is like this:

import sys

import os

project\_root = os.path.abspath(os.path.join(os.path.dirname(\_\_file\_\_), '..'))

# Add the project root directory to the sys.path

# This allows imports like 'from src.app.main import app'

sys.path.insert(0, project\_root)

That means we need to add the project root to the system path and make src and src\app as packages.

In any code file in the folder, we can ask python to import.

In summary, if two .py files are in the same folder, we can use import directly, if not we must

1. **make them as packages**
2. **add project root folder to system path in the notebook.**

Problem 2: We also need to add the following code to conftext.py to avoid error: **CreateProcess error=2**  
os.environ['PYSPARK\_PYTHON'] = sys.executable

os.environ['PYSPARK\_DRIVER\_PYTHON'] = sys.executable

Problem 3: we also need to add the following code to conftext.py to avoid any other errors related to spark running on windows:

os.environ['HADOOP\_HOME'] = "C:\\hadoop"

os.environ['PATH'] = f"{os.environ['HADOOP\_HOME']}\\bin;{os.environ['PATH']}"

**Because any JAVA errors is related to hardoop.dll and winutils.exe.**

Sometimes we need to run pytest --collect-only -s to make sure pytest collected all files, which can help identify problems

Be careful about the column preprocessing logic and lazy evaluation of pyspark.

-------------------------UNTIL NOW, ALL TRAINING WORK ARE DONE!!!!--------------------------------------

------------------------------The following step 6-8 are for docker trainer setup------------------------------

**It is like run python src\train.py but in the docker container.**

**Step 6: Create Dockerfile.trainer File**

For spark I need to find an image, I choose apache/spark:4.0.0-python3, so the first line of my Dockerfile.trainer is FROM apache/spark:4.0.0-python3

**Step 7: CD to project root, Run docker build -t hospital-readmission-trainer . in command prompt (No need to activate python env because this is using docker)**

1. Encountered Problem 1:

Defaulting to user installation because normal site-packages is not writeable

ERROR: Could not install packages due to an OSError: [Errno 13] Permission denied: '/nonexistent'

Solving:

Add USER root before RUN pip install --no-cache-dir -r requirements.txt in **Dockerfile File.**

1. Encountered Problem 2:

/opt/entrypoint.sh: line 128: /app/bin/spark-submit: No such file or directory

Solving:

Change CMD ["spark-submit", "src/train.py"] to ENTRYPOINT ["spark-submit", "src/train.py"] in **Dockerfile File.**

Everytime change Dockerfile File must rebuild and rerun.

**Step 8: At project root, Run docker run --rm -v "%cd%\output":/app/output hospital-readmission-trainer (No need to activate python env because this is using docker)**

Explain: Use "%cd%\outputs":/app/output

When you run this command, the **train.py** script will save the model to **/app/output** inside the container, and because of the **-v** "bridge", the model files will simultaneously appear in **Hospital\_Readmission\_Project/outputs** ("%cd%\outputs) folder on PC.

Explain: hospital-readmission-trainer is the image just build in previous step.

Encountered problem: pyspark.errors.exceptions.captured.IllegalArgumentException: java.net.URISyntaxException: Relative path in absolute URI:

Solution, the file load path must use relative path not absolute path because in docker container it does not understand the absolute path on your laptop.

------------------------------The following step 9-12 are for docker server setup------------------------------

**Step 9: Create Dockerfile.server, we need to install python, and java in it and also the packages in requirements.text**

**Step 10: Run docker build -f Dockerfile.server -t hospital-readmission-server .**

**Step 11: Run docker run -d -p 8000:80 --name my-app-container hospital-readmission-server**

**Step 12: Open a tab use address:** [**http://localhost:8000/docs**](http://localhost:8000/docs)

**Try it, if it works, run docker stop my-app-container and docker rm my-app-container**

**(the created image is not impacted)**

------------------------------The following step 13- are for Github CI/CD setup------------------------------

Step 13: Create ci.yaml in folder .github\workflows and push it to github.