PROJECT SPECIFICATION

Operationalize a Machine Learning Microservice API

Files Submitted

CRITERIA	MEETS SPECIFICATIONS
All files are submitted	The submitted repository includes a .circleci folder, a README.md file, a Dockerfile and Makefile, as well as an app.py file, a prediction script, and the necessary scripts to **run** and **upload** a microservice on Docker and Kubernetes. There should also be two output text files: `docker_out.txt` and `kubernetes_out.txt` that include the log output after a prediction is made, given some sample input data. *NOTE: Before submitting a link to your complete, project repository, make sure you have included all required and complete files (including run_kubernetes.sh, run_docker.sh, docker_out.txt, kubernetes_out.txt, and a .circleci build directory).*
`.circleci` folder is included	A `.circleci` folder is included in the Github repository. The directory holds a `config.yml` that checks the project code for errors. Your project should pass, as indicated by a CircleCI status badge in the repository README.

Code Quality & Enhancement

CRITERIA	MEETS SPECIFICATIONS
Extend app.py to log a prediction value	Add an additional logging statement to `app.py` that prints as "info" the output prediction for some given input data.
The project shows the proper use of documentation	The README file includes a summary of the project, how to run the Python scripts and web app, and an explanation of the files in the repository.
The project passes linting via a Makefile	Both the Dockerfile and the python file pass linting using pylint and hadolint. This may involve selectively customizing lint overrides in both tools. The lint should be run for both tools via the command `make lint`. Circleci build server validates step.

Docker Configuration

CRITERIA	MEETS SPECIFICATIONS
Dockerfile is complete	The Dockerfile should create a working directory, install the necessary dependencies, expose port 80, and specify that `app.py` run at container launch.
Dockerfile passes linting via a Makefile	The Dockerfile should pass `make lint` without errors. Circleci build server validates step.
Log output is saved in `docker_out.txt`	While running the docker container, call the prediction script, `make_predictions.sh`; the log output, which includes the output prediction value, should be included in your submission as a text file, `docker_out.txt`.
`run_docker.sh` is complete	Build, list, and run steps are completed in `run_docker.sh`.
Docker image is uploaded to docker via `upload_docker.sh`	The built docker image is uploaded to your own personal Docker ID, as indicated by a complete `upload_docker.sh`.

Kubernetes Configuration

CRITERIA	MEETS SPECIFICATIONS
`run_kubernetes.sh` is complete	This script runs a docker image with kubernetes, lists the kubernetes pod(s), and forwards the container port to a host, using `kubectl` appropriately.
An output prediction is saved in `kubernetes_out.txt`	While running on kubernetes, call `make_predictions.sh`; the terminal output should be included in your submission as a text file, `kubernetes_out.txt`.

Suggestions to Make Your Project Stand Out!

1. Extend the microservice to deliver additional functionality, say an additional prediction.
2. Make the kubernetes deployment work on multiple cloud platforms: i.e. GCP, AWS, and Azure.
3. Record a demo video that shows the scale up and scale down characteristics of the kubernetes application.
4. Create a simple web application front-end to accept user input data and produce a prediction.