Machine learning in production I: embedding a model in a backend

Machine learning in production

Machine learning in production

There are several ways of putting a model into production:

- - Via batch processing
- - Via web services
- - Via streaming systems

Each brings its advantages and disadvantages.

Batch processing is simple to set up but does not provide real-time predictions. Web services provide real-time predictions, but can suffer under heavy load and with an inelastic infrastructure.

The goal of the chapter is deploy a model in a cloud setting using web services. Then it will be to automate everything.
But first let's start with something simpler : using a model with streamilt

Using a ML model in streamlit

In this first exercise, we'll learn how to use a model in streamlit.

1. Download the files **train_model.py** and **houses.csv** accessible via this link

2. Run the file "train_model.py" which generates the model. Check that a regression.joblib file appears

3. Create a model_app.py file and import streamlit and joblib

4. In model_app.py, using the joblib library, load the model from the "regression.joblib "file

5. Using the st.number_input function, create three form fields for size, number of bedrooms and whether a house has a garden

6. Retrieve the information and pass it to the model via the predict function. Display the result in the streamlit

Use st.write to display the prediction result

A machine learning model in a web service mini project

The goal of this mini project is to deploy a web service running a machine learning model with docker.

In case you need to work on the basics of docker, go make the docker sessions on nowledgeable.

If you are not confortable with web developpement you can do the session web backend basics

Level 0 (on your local machine)

1. Preamble: Create a web server with FastAPI that accepts a GET request /predict which returns a json with {"y_pred": 2}.

2. Test your server with the browser, wget and a GUI http client (hugo, postman, insomnia) and the python library named requests

3. Modify the server so that it accepts a POST request

4. Update the /predict endpoint which so that it use the house prediction model.

5. Test the model with your favorite HTTP client

Level 0.5 (on a remote machine)

1. 1. Connect via SSH to the provided machine (ubuntu@20.54.250.141) password: Supermotdepasse!42.
2. 2. Create a folder with your initial (td for Tom dupont for isntance)
3. 3. Using scp or git clone, deploy your code on a remote machine (provided) in your folder
4. 4. Launch your web service on the remote machine
5. 5. Verify that you and a colleague can access your web service and get predictions

Level 1 (on your local machine)

You want now to run the backend inside a docker container. For that create a Dockerfile to create your custom Docker image

1. Create a dockerfile that will package your webservices.

2. Build the image and run the container with docker build and docker run commands. Beware you will need the -p flag for docker run

3. Test that you can use your web service.

Level 2 - Deploy the container in a cloud virtual machine

1. We have setup a Virtual machine for you, you need to deploy your web service with docker.

1. 2. Create a folder with your initials in the /home/ubuntu folder
2. Make sure to deploy your container on the virtual machine and ask a friend to use your model

Level 3 - **Automatically build images with a CI/CD**

1. Learn how to use github actions by doing the following exercises :

https://github.com/lcetinsoy/revision-git/blob/master/exercice5.md

2. Automatically build and deploy the container using a CD pipeline using gitlab or github action

When you git push on the main branch an update, a pipeline should be is triggered and the docker image should be build automatically and send to the docker hub image registry

3. Make sure that when a git push is made, the automatically built image is also deployed on the provided VM and the model is updated on production automatically

Level 4 - adapt the process to a more interesting model - bonus if you are a GOAT

Adapt the previous work to deploy in production a more interesting model :

- it can be any model you trained in another course (image classification, cell segmentation, etc.)
- A T5 or a Bert model from hugging face

Github url mini project

Send the github link of the mini project