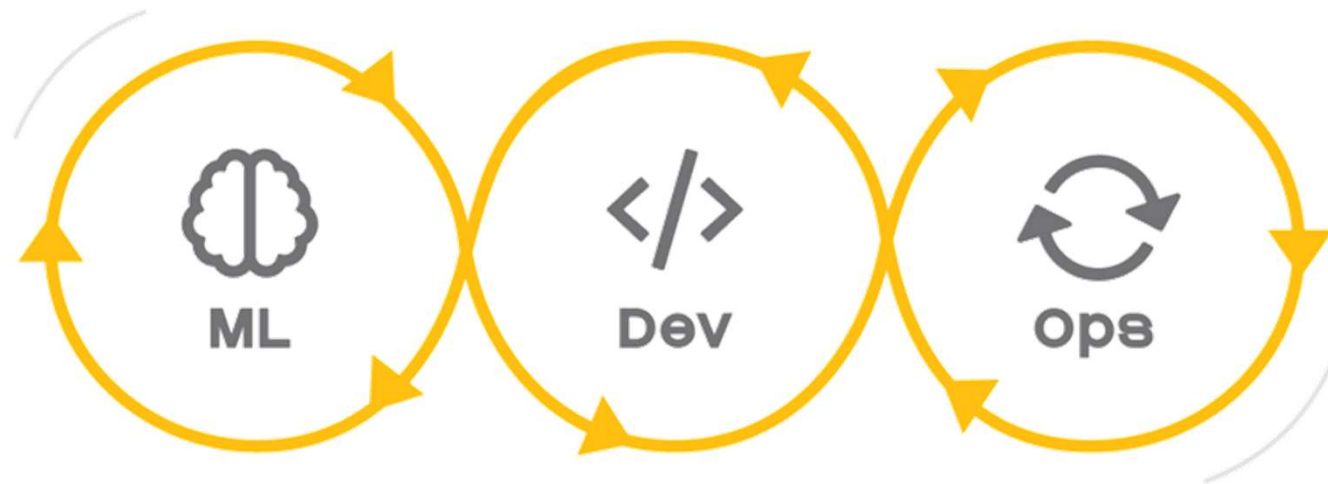# Workshop MLOps

Office E204

Plan du cours

- Introduction

- MLOps vs. DevOps

- Life cycle

- Strategy

- Configuration

  - ✓ Dockerfile

  - ✓ Pipeline de Jenkins

Introduction

The purpose of this workshop is to provide an example of how we can use DevOps tools such as Docker and Jenkins to create a machine learning pipeline.

ML + Dev + Ops = MLOps

MLOps vs. DevOps

The MLOps process is inspired by DevOps and is based on collaboration with DevOps teams for model deployment services.

- o In DevOps, we learned that it's all about simplifying software development, then deploying and monitoring it.

- o In MLOps, we focus on Machine Learning operations. operations (Machine Learning).

→ It's a useful approach to creating the best machine learning solutions for the end user.

MLOps vs. DevOps

|	**DevOps**	**MLOps**
Code	• Building a standard application • Standard libraries for specific use cases	• Building a model for inferences • Wide range of tools, languages and libraries
Artifact	Executable JAR	Serialized file
Validation	Unit tests	Model performance (error rate)
Roles	• Software Engineers • DevOps Engineers	• Data scientists • Machine Learning Engineers

Life cycle

The standard lifecycle for developing Machine Learning solutions is as follows:

o   Identify business needs and objectives

o   Collect the necessary data

o   Prepare data

o   Create model

o   Set model hyperparameters

o   Train and evaluate the model

o   Test and deploy model

o   Retraining, if necessary
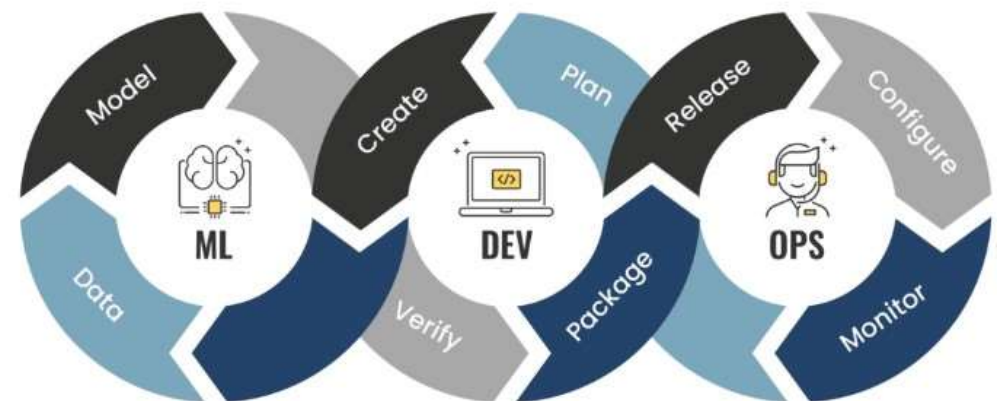
Strategy

The first step is to understand the inputs and outputs of our Python scripts, in order to understand the workflow of this project.
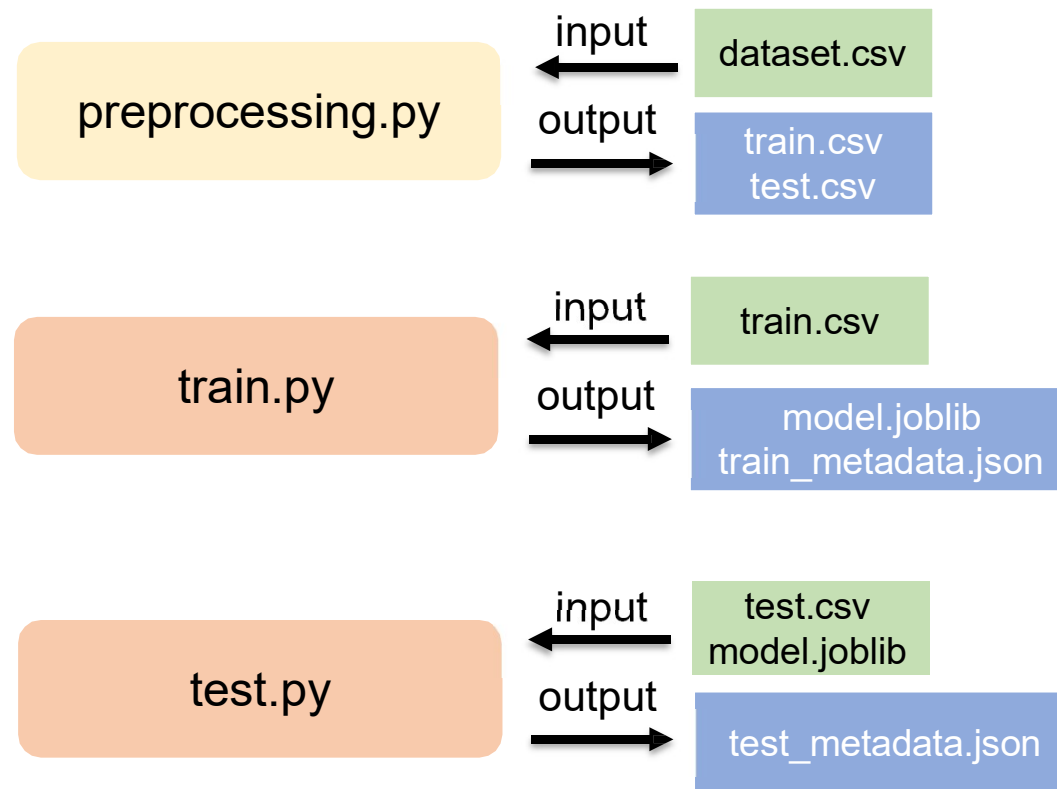
Configuration –Docker File

To define the docker file, we need to :

1. Specify where we want to run our pipeline.

   ➔For our pipeline, we'll simply use a jupyter image called "jupyter/scipy-notebook".

2. Install packages. To do this, we use the **RUN** command.

3. Create folders (model, raw_data, processed_data and results).

4. Specify the order in which the scripts and raw data in our directory are copied. These will be copied into our container once we've created it.

Configuration – Docker File

```dockerfile
FROM jupyter/scipy-notebook

RUN pip install joblib          ←——— To serialize and deserialize the model.

USER root
RUN apt-get update && apt-get install -y jq    ←——— To access values in json files.

RUN mkdir model raw_data processed_data results

ENV RAW_DATA_DIR=/home/jovyan/raw_data
ENV PROCESSED_DATA_DIR=/home/jovyan//*toDo*/

ENV MODEL_DIR=/home/jovyan/model
ENV RESULTS_DIR=/home/jovyan/results
ENV RAW_DATA_FILE=heart.csv

COPY /*toDo/* ./raw_data/*toDo*/
COPY preprocessing.py ./preprocessing.py
COPY train.py .//*toDo*/
COPY test.py ./test.py
```
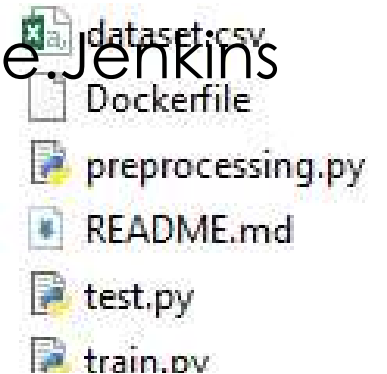
Configuration – Pipeline de Jenkins

Our project contains 3 scripts, dataset and a docker file. Jenkins

pipeline steps:

✓ Retrieving the project from Git

```
steps {
    echo 'Project is downloading...'
    git branch:'master', url:'https://github.com/RafrafiNawress/MLOps-Docker-Jenkins.git'

}
```

✓ Build and launch the image

```
steps {
    bat 'docker build -t mlops-model .'
    bat 'docker run -d --name model mlops-model'
}
```

Configuration – Pipeline de Jenkins

✓ Running commands inside the container

Note: The scripts must be run in order, then display the results.

```
steps {
        bat 'docker container exec model python3 preprocessing.py'
                                    To complete …
}
```

✓ Display validation accuracy and test precision

```
steps {
     bat 'docker container exec model python3 train.py'
     bat 'docker container exec model python3 test.py'
     bat 'docker container exec model cat /home/jovyan/results/train_metadata.json /home/jovyan/results/test_metadata.json'
     bat 'docker rm -f model'
}
```

Workshop MLOps

If you have any questions, please do not hesitate to contact us:

IT Department - UP ASI
E204

© 2023-2024 – ESPRIT – DevOps Module – MLOps