Data Science in production

Alaa BAKHTI

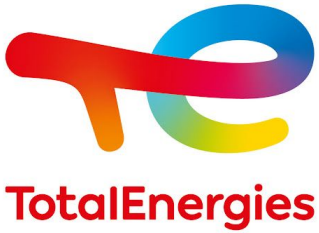
Who am I?

EPITA 2018 : did the piscine 🙀🥴🤕🙈

Teacher and advisor @ EPITA
- Advisor for AIS & DSA
- Teaches the courses:
  - Recommender Systems
  - Time Series
  - Data Science in production

Machine Learning Engineer @ OCTO Technology

Interested in Data Science, Software craftsmanship => AI in production



What about you?

- Your profile
- Any knowledge or experience with ML powered applications?
- Your expectations for the course

The course

- 8 sessions
- Grading:
  - Project - 60%
  - 2 practical works - 30%
  - Participation - 10%

Syllabus

1. Code versioning with Git and environment management in Python
2. Introduction
3. Coding best practices 1
4. Coding best practices 2
5. Versioning in a data science project
6. Quality validation for model building and integration
7. Model serving and deployment strategies
8. Model monitoring and retraining
9. Machine learning delivery best practices (bonus)

Some rules

- No showing up late to the courses !
- No computer during theory part
- No cheating in the practical work & project: 2 students last year were caught
  => disciplinary board

Data Science in production

Lecture 0: Code versioning with & environment management in Python

Alaa BAKHTI

Any questions on Git or miniconda environments setup?

Code versioning with Git

Code versioning with Git

Best practices

- Commit with each new change (working versions)
- Choose comprehensive commit messages
- Review the changes you've made before commiting
    - Visualize your changes and repository state with Git GUI tools like Sourcetree

Code versioning with Git and environment management in Python

Refer to the practical work here

Some resources for code versioning with Git

- Documentation for the different Git commands
- Visualizing Git Concepts with D3
- Learn Git Branching
- Adding locally hosted code to GitHub
- GIT PURR! Git Commands Explained with Cats!
- Pro Git book

Environment management in Python

Package

A package is defined by its name and its version e.g. *numpy-1.19.4*

The package version generally follows the following format: **MAJOR.MINOR.PATCH**

- The **MAJOR** version is incremented when an **incompatible API changes** are made
- The **MINOR** version is incremented when a **new functionality** is added in a **backwards compatible** manner
- The **PATCH** version is incremented when a **backwards compatible bug fixes** are made

Source: Semantic Versioning

PIP: package installer for Python

- Python package manager
- Used to install packages (e.g. pandas) along with their dependencies (e.g. numpy is a dependency for pandas)
- Comes built in Python

Dependency management

- Why?
  - Manage projects with different dependencies (*tensorflow 1.15 and tensorflow 2.8*)
  - Avoid dependency issues by organizing packages in isolated environments
  - Reproduction of environments
- How?
  - Create a virtual environment for your project: *a virtual environment is an isolated Python environment where a project's dependencies are installed in a different directory from those installed in the system's default Python path and other virtual environments*
  - Anaconda & miniconda come with their own package manager *conda*
  - *pip* can also be used in a conda environment to install packages that are not available in the Anaconda package repository
  - Some environment managers: miniconda, virtualenv, virtualenvwrapper, Pipenv, Poetry

requirements.txt file

- requirements.txt, environment.yaml
- Format ==========>

- Requirement specifiers for pip

- To install all the packages specified in the file
  - *pip install -r requirements.txt*

- Never

```
$ pip freeze > requirements.txt
```

```
└─ cat requirements.txt
jupyter==1.0.0
numpy==1.19.4
pandas==1.1.5
flake8-3.9.0
scikit-learn==0.23.2
matplotlib==3.4.1
seaborn==0.11.0
mlflow==1.12.1
```

IDE - Integrated development environment

- Why?
- Tools: PyCharm, Visual Studio code
- You can get the PyCharm Professional version licence with your EPITA email

Assignments

- Apply what you have learned on Git and conda (instructions): submission in 1 week
- Predict house prices (instructions): submission in 2.5 week