

MLOps for the rest of us

A poor man's guide to putting models in production

PyData Global 2022

Duarte O.Carmo

duarteocarmo.com - [@duarteocarmo](https://twitter.com/duarteocarmo)

Who even are you?

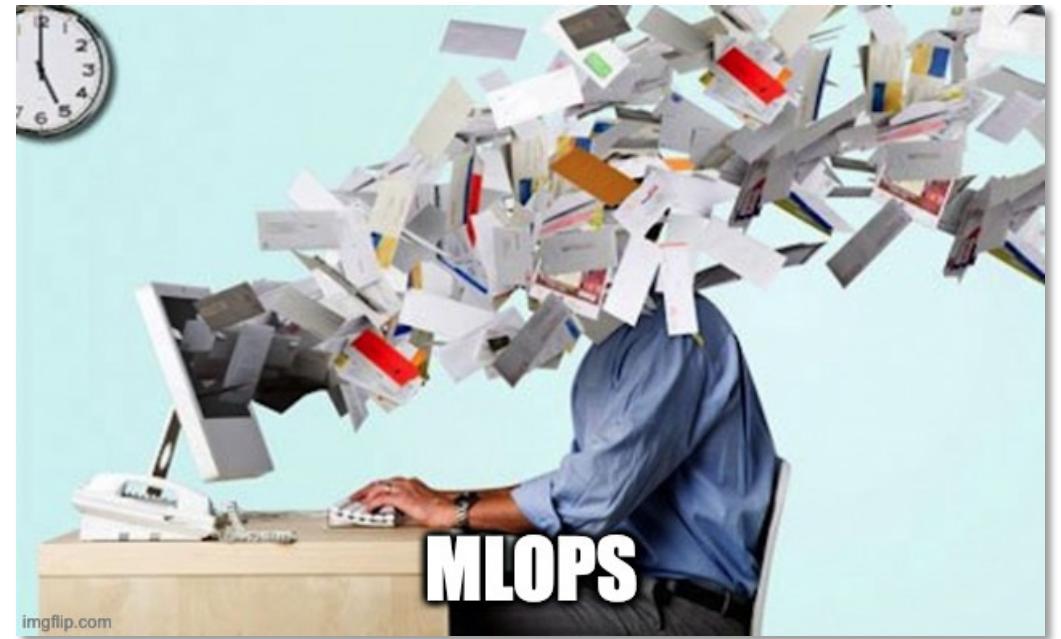


- */du-art/*
- ML/Software Engineer - Contractor
- From Lisbon, based in Copenhagen (PyData CPH ❤️)
- *Past:* Strategy, Product Management, New Ventures, Management Consulting
- I write code and solve problems end-to-end
- I like running a lot



This is a talk about helping small teams cut through the sh*t

- LinkedIn (you're not them)
- Cutting through the MLOps hype
- How to deliver models to production
- Unsexy ML (e.g., BoringML)
- *Opinions*, so many opinions
- Grains of salt



1 | LinkedIn

Story time

A large, diverse audience is seated in a dark theater, filling the frame from the foreground to the back. The audience members are looking towards the right side of the image, presumably towards a stage. The theater has multiple tiers of seating, and the walls are dark with rows of small, warm-toned lights. The overall atmosphere is dim and focused.

You're not Amazon

You're not Apple

You're not LinkedIn

You're not Facebook

(maybe some of you are, but you get my point)

Them

600+ ML Engineers

Dedicated teams

Own the Cloud

Research team

ML Platform team

You (or at least me)

4 people?

Humm.. You?

Keep costs down

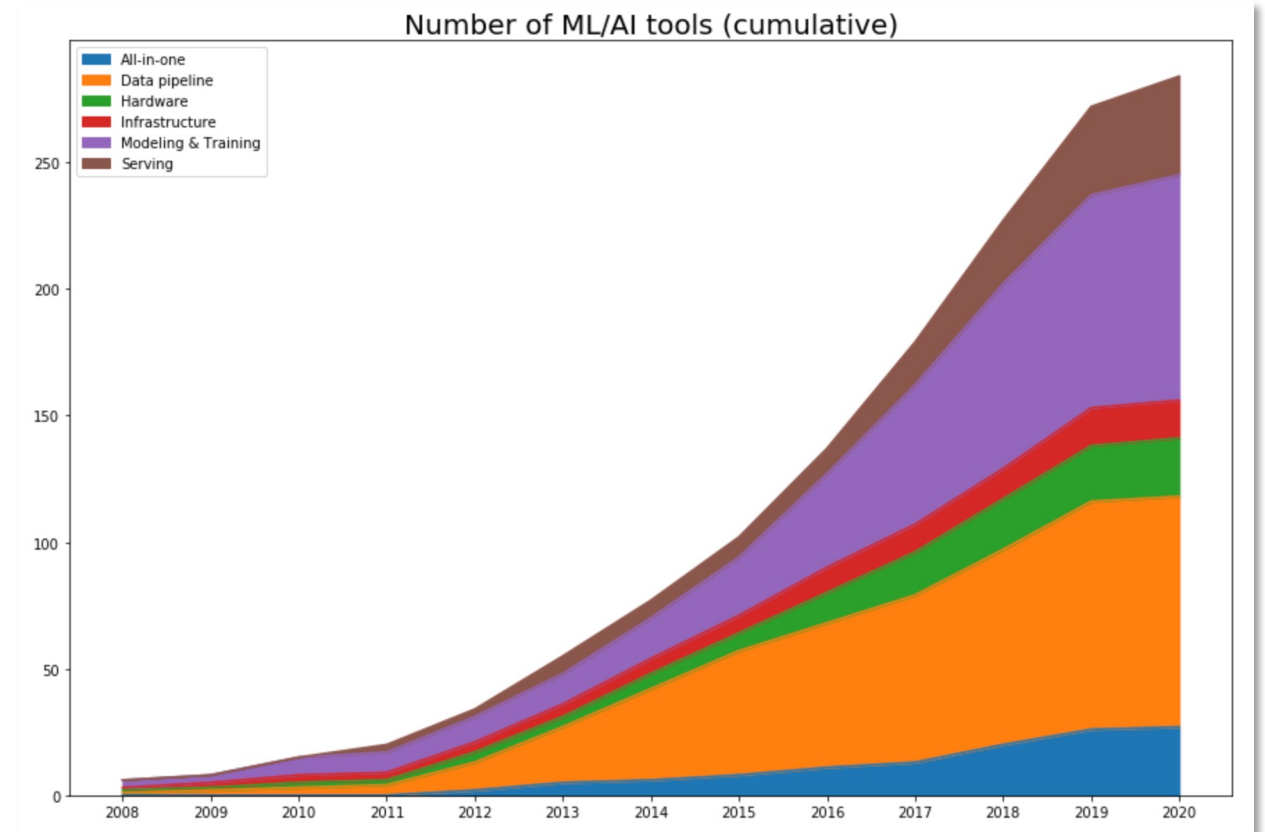
Deliver VALUE, FAST

LOL

2 | MLOps

MLOps is not about adopting tools, it's about delivering value

- Gold Rush Age
- FOMO
- Spam emails
- Focus on tools
- 22% have put a model in production
- The real problem: Providing value.

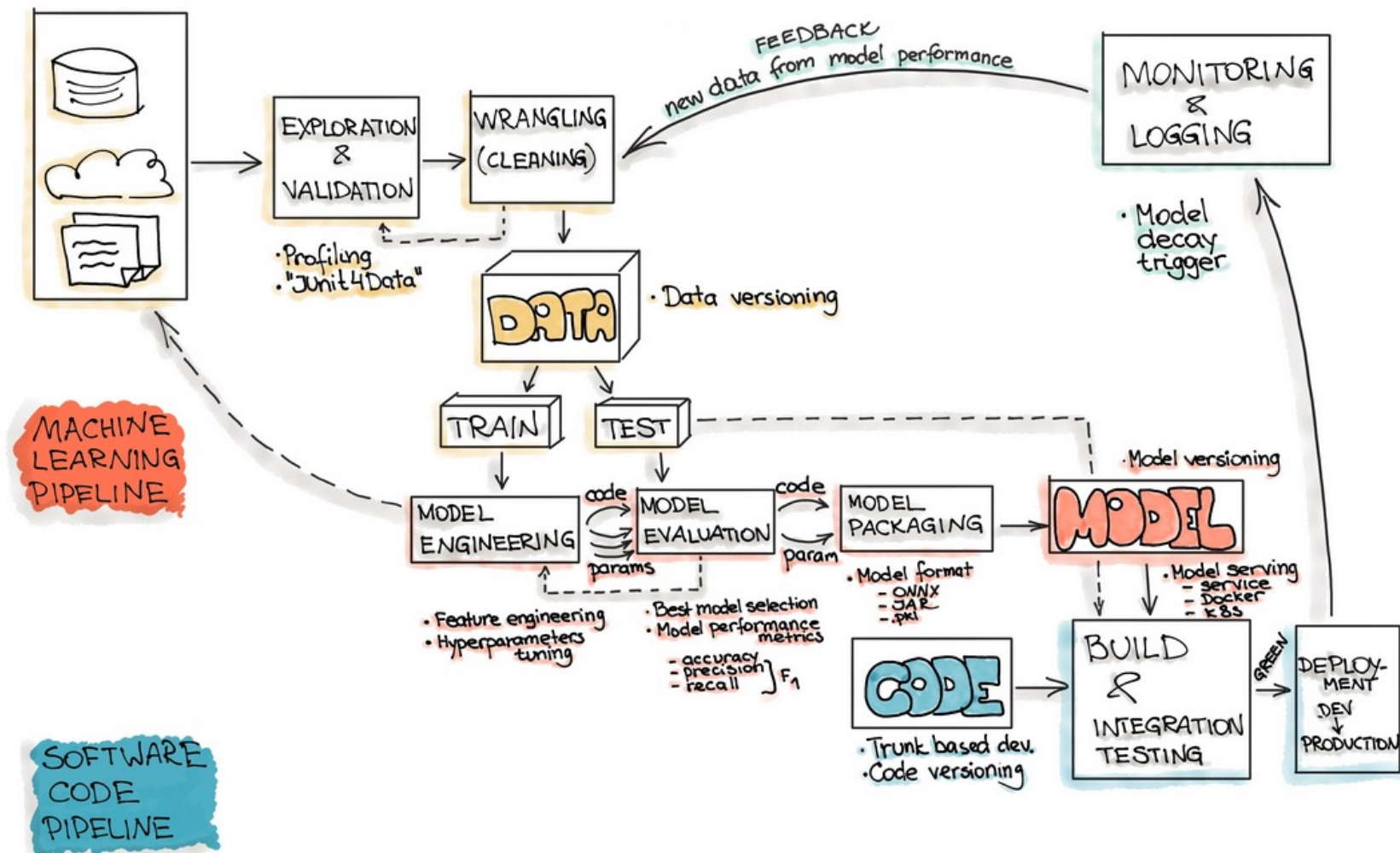


Credits: huyenchip.com

“Can you guys make a model in a couple of weeks?”

DATA PIPELINE

MACHINE LEARNING ENGINEERING



Credits: ml-ops.org

duarteocarmo.com - @duarteocarmo

3 | Solving (small scale) ML Problems

3.1 | People

3.2 | Data

3.3 | Start small

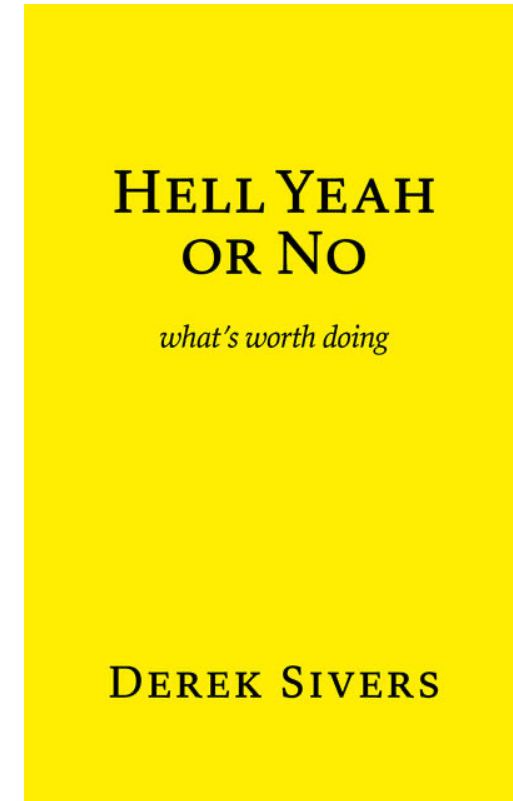
3.4 | Production

3.5 | Monitoring

3.6 | ML Platform

MLOps does not start with tools, it starts with people

- “We need an ML Model”
- “We need faster horses”
- The specialist (i.e. you)
- Be skeptical (if you can)
- Solution vs. Partnership



A dark, dimly lit basement with exposed pipes and a small table with chairs in the center.

Don't build in the basement

3.1 | People

3.2 | Data

3.3 | Start small

3.4 | Production

3.5 | Monitoring

3.6 | ML Platform

Data is though...

- You probably don't have it
- You got to label it
- Continuous learning
- Small number of labels

Data is though...

- You probably don't have it
- You got to label it
- Continuous learning
- Small number of labels

But don't despair

- Scrape, annotate
- Embeddings, PCA, similarity*
- Capture **feedback**
- Zero shot, embeddings, Fine-tune

* - github.com/koaning/bulk

duarteocarmo.com - @duarteocarmo

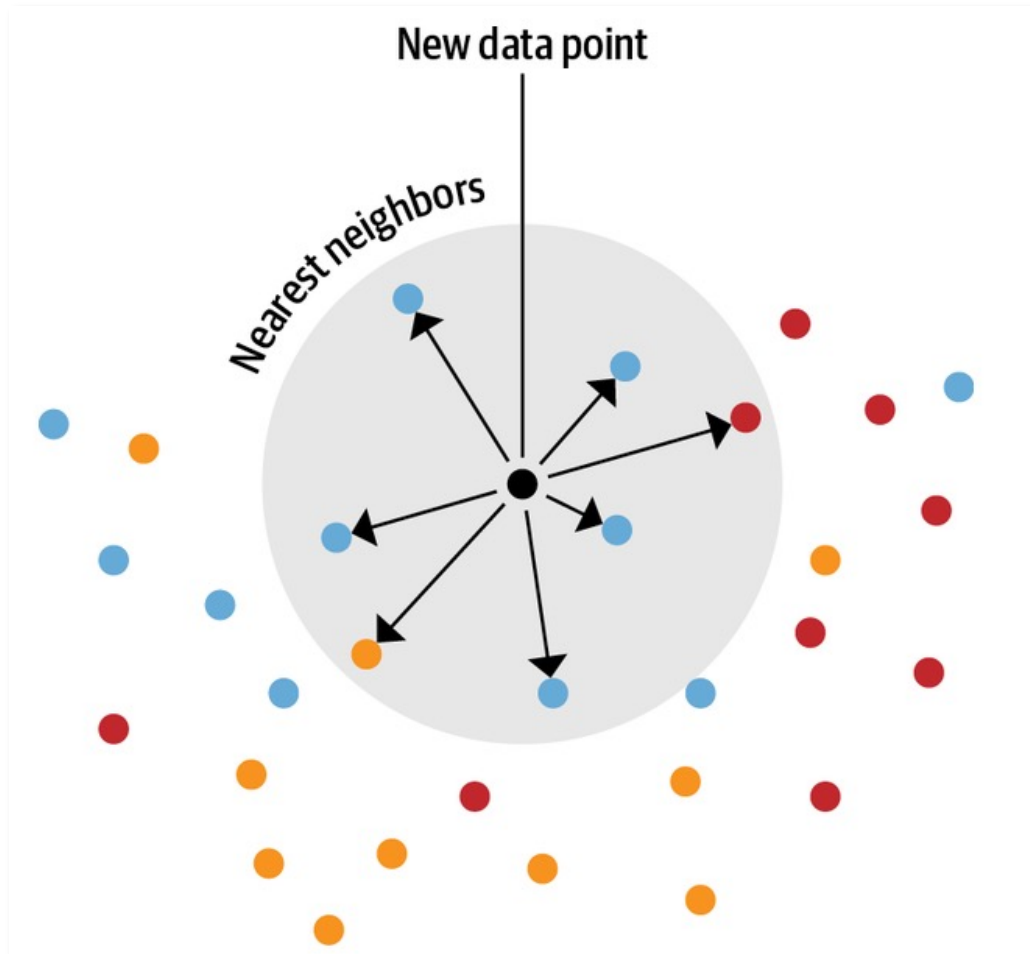


Figure 1: Making a lot with a Little
Credits: Lewis Tunstall, NLP with Transformers O'Reilly

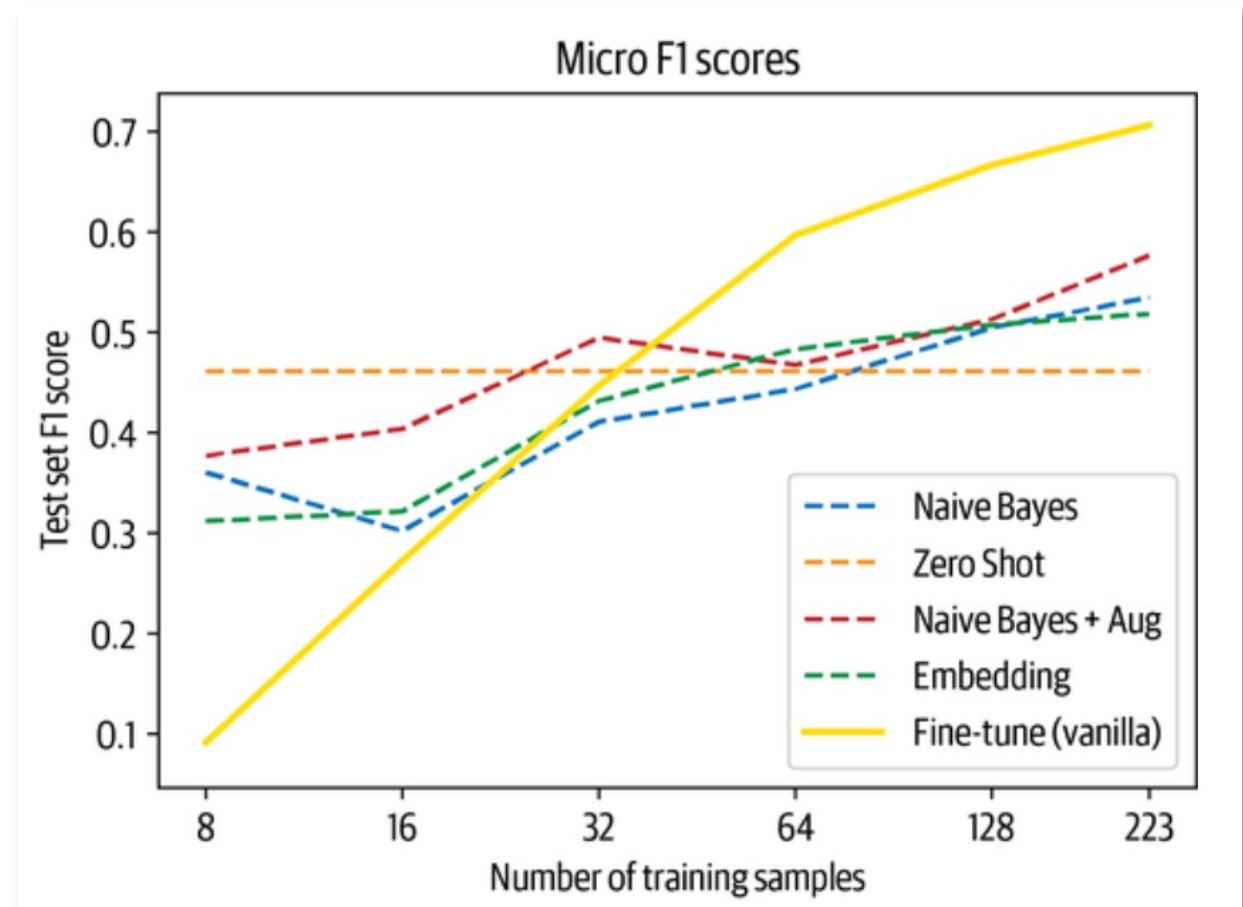


Figure 2: Nearest neighbour lookup
Credits: Lewis Tunstall, NLP with Transformers O'Reilly

- 3.1 | People
- 3.2 | Data
- 3.3 | Start small**
- 3.4 | Production
- 3.5 | Monitoring
- 3.6 | ML Platform

Packaging with pyproject and pip-tools

Using pyproject.toml (PEP 621):

```
[build-system]
requires = ["hatchling"]
build-backend = "hatchling.build"
```

```
[project]
name = "dreambox"
version = "11.22"
readme = "README.md"
requires-python = "≥ 3.10"
dependencies = [
    "grpcio=1.48.1",
    "pandas-gbq ≥ 0.17.8",
    "openai ≥ 0.25.0",
]
```

```
[project.optional-dependencies]
dev = [
    "black ≥ 22.10.0",
    "pip-tools ≥ 6.10.0",
    "pytest ≥ 7.2.0",
]
```

Generating requirements:

```
## Building dependencies
build:
    pip-compile -o=requirements.txt pyproject.toml
    pip-compile -e=dev -o=requirements-dev.txt pyproject.toml
```

Test enough to increase confidence and ensure production does not break

```
import pytest
import dreambox

class TestClassifier:
    @pytest.mark.parametrize(
        "text, expected_class",
        [
            ("I hate this", "pissed"),
            ("This is cool", "chilled out"),
            (" ", "chilled out"),
            (None, None),
            (1, None),
        ],
    )
    def test_basic_classifier(self, text, expected_class):
        assert dreambox.classify(text) == expected_class
```

- **pytest** is fine
- parametrize
- The obvious
- Edge cases – empty strings
- Validate user inputs

Make makes things simpler – pun intended

```
## Install for production
install:
    @echo ">> Installing dependencies"
    python -m pip install --upgrade pip
    python -m pip install -e .
    python3 -m pip install -r requirements.txt

## Install for development
install-dev: install
    python3 -m pip install -r requirements-dev.txt

## Build dependencies
build:
    pip-compile -o requirements.txt pyproject.toml

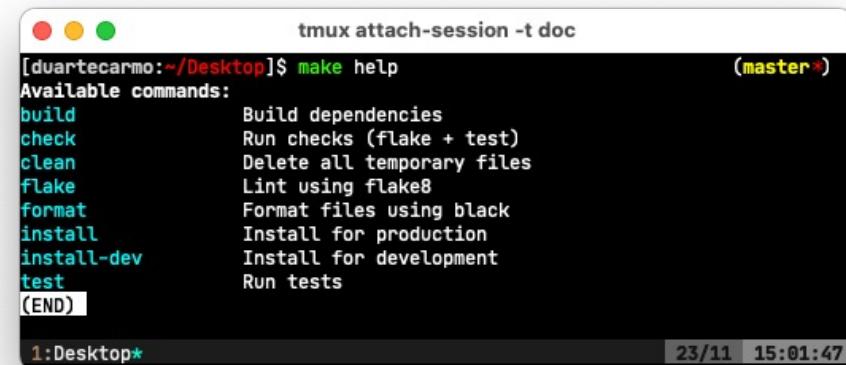
## Delete all temporary files
clean:
    rm -rf .ipynb_checkpoints
    rm -rf **/.ipynb_checkpoints
    rm -rf .pytest_cache
    rm -rf build
    rm -rf dist
```

```
## Lint using flake8
flake:
    flake8 src tests

## Format files using black
format:
    isort src/ tests/
    black -l 79 src/ tests/

## Run tests
test:
    pytest tests --log-level=WARNING

## Run checks (flake + test)
check:
    flake8 --ignore=E501,W503,E203 src
    black --check -l 79 src/ tests/
```



The screenshot shows a terminal window titled 'tmux attach-session -t doc'. The user is in a directory '~/Desktop' and has run the command 'make help'. The output lists the available Make commands and their descriptions:

```
[duarteocarmo:~/Desktop]$ make help (master)
Available commands:
build      Build dependencies
check      Run checks (flake + test)
clean      Delete all temporary files
flake      Lint using flake8
format     Format files using black
install    Install for production
install-dev Install for development
test       Run tests
(END)
```

The terminal status bar at the bottom shows '1: Desktop*', '23/11', and '15:01:47'.

tinyurl.com/makehelp

duarteocarmo.com - @duarteocarmo

CI/CD increases confidence, and enforces standards

```
check:
  runs-on: ubuntu-latest
  steps:
    - uses: actions/checkout@v2

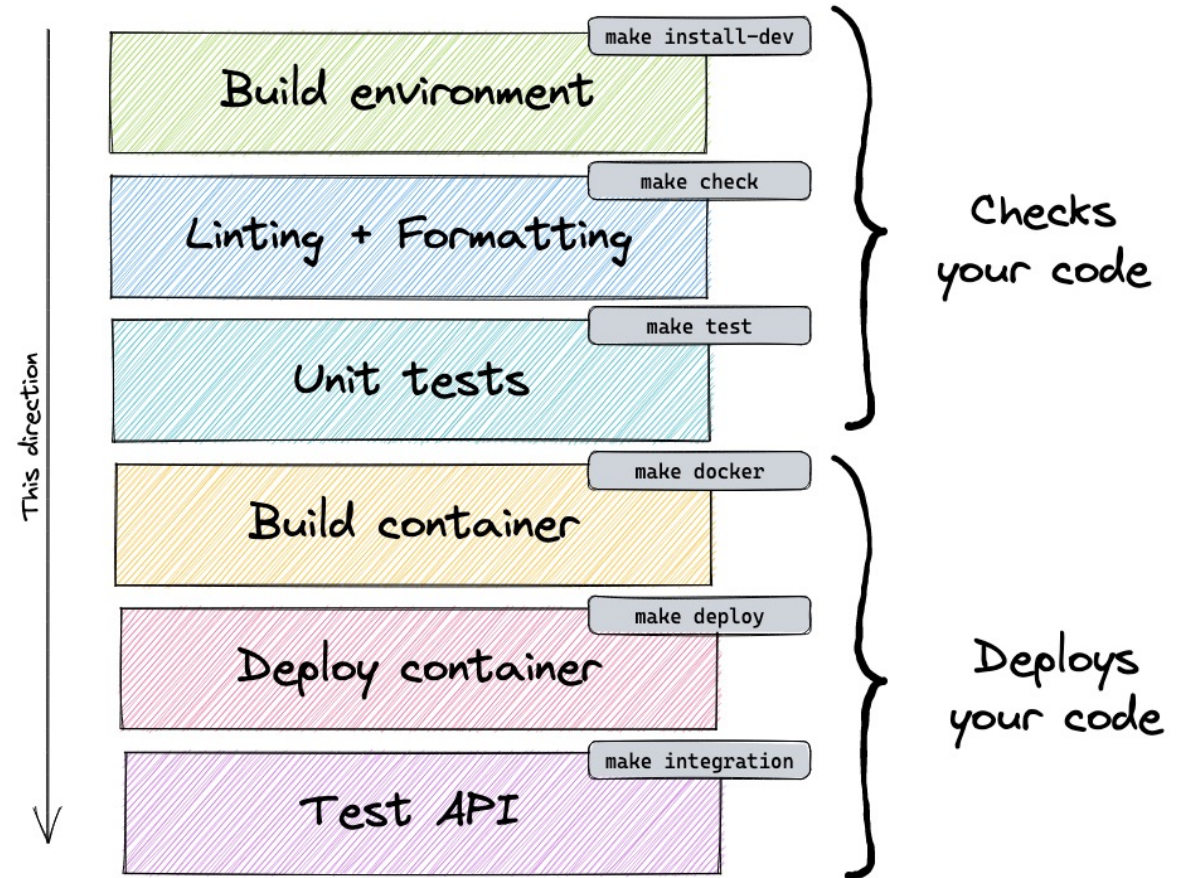
    - uses: actions/setup-python@v2
      with:
        python-version: "3.12"

    - name: Install requirements
      run: make install-dev

    - name: Run checks
      run: make check

    - name: Run tests
      run: make test
```

A simple ML deployment pipeline



- 3.1 | People
- 3.2 | Data
- 3.3 | Start small
- 3.4 | Production**
- 3.5 | Monitoring
- 3.6 | ML Platform

Delivering containerized models leveraging the Cloud

- FastAPI and PyDantic
- Documentation + Validation
- Dockerize all the things
- Choose the right Cloud

```
# We'll take this in:  
class Features(BaseModel):  
    sepal_length: confloat(ge=0.0, le=1.0)  
    sepal_width: confloat(ge=0.0, le=1.0)  
    petal_length: confloat(ge=0.0, le=1.0)  
    petal_width: confloat(ge=0.0, le=1.0)
```

```
FROM python:3.11
```

```
COPY requirements.txt /tmp/
```

```
RUN pip install --upgrade pip
```

```
RUN pip install torch --extra-index-url https://.../cpu
```

```
RUN pip install -r /tmp/requirements.txt
```

```
RUN mkdir -p /src
```

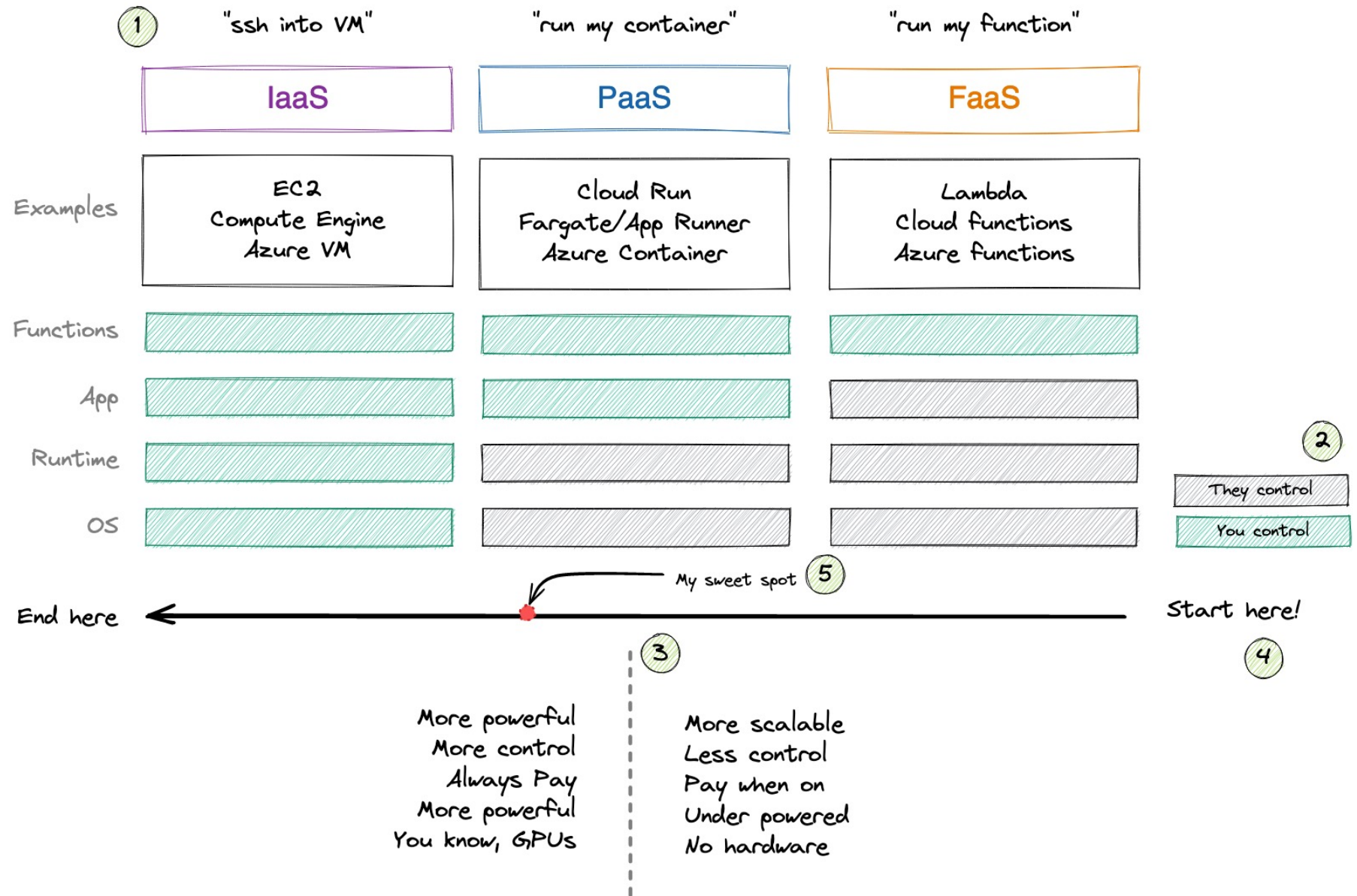
```
COPY src/ /src/
```

```
RUN pip install -e /src
```

```
EXPOSE 80
```

```
CMD ["make", "production"]
```

Choosing the right Cloud service matters

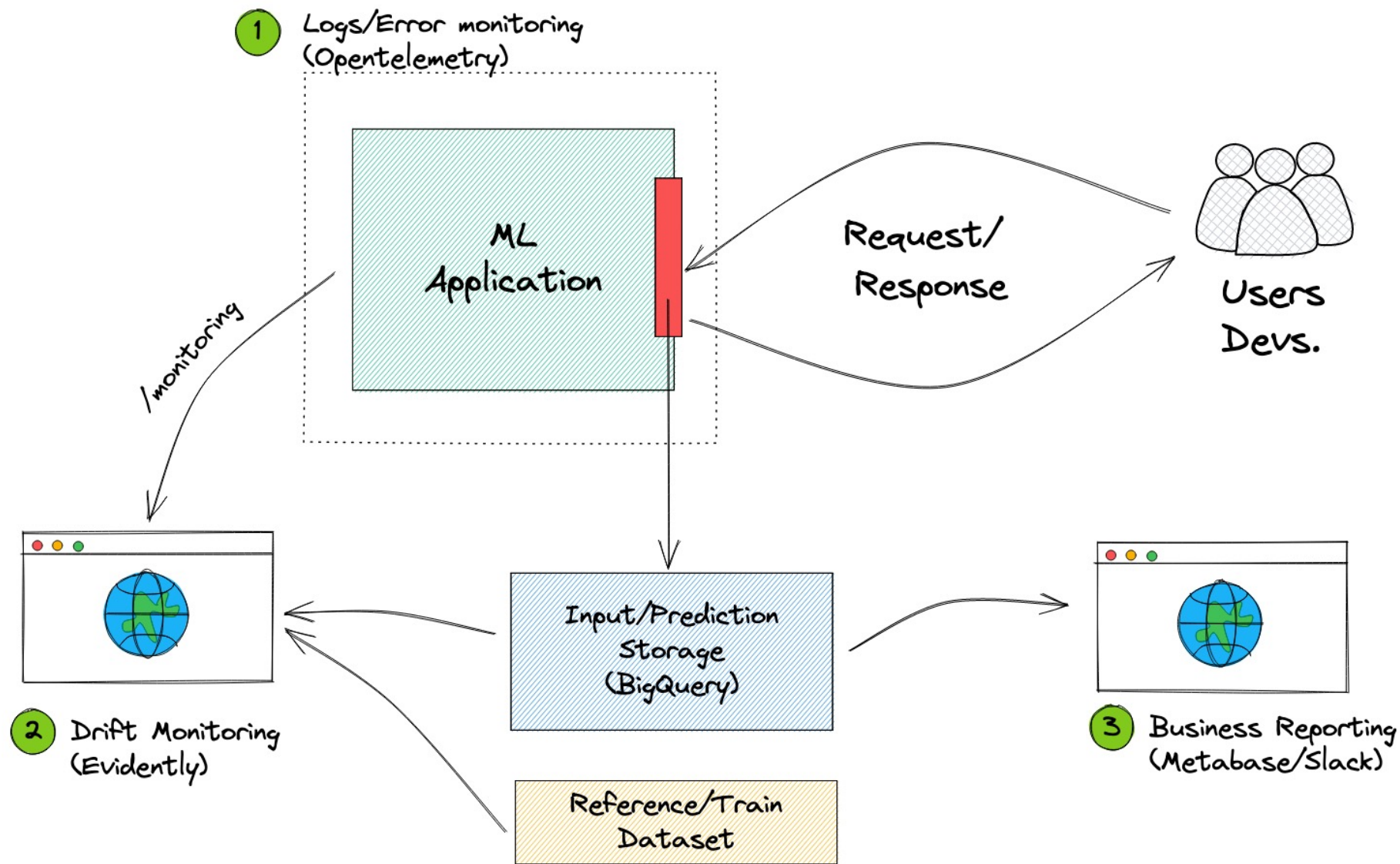


Clouds are the same: comparecloud.in

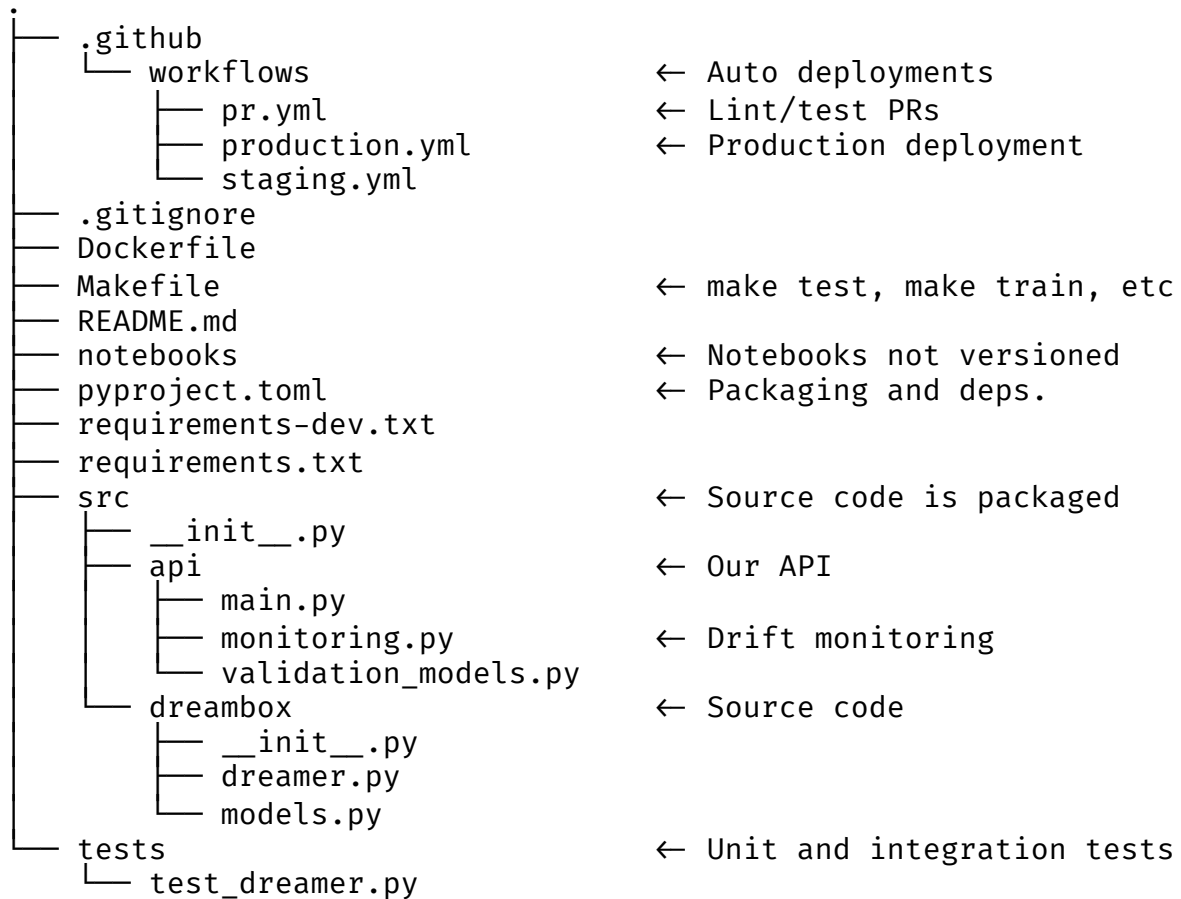
duarteocarmo.com - @duarteocarmo

- 3.1 | People
- 3.2 | Data
- 3.3 | Start small
- 3.4 | Production
- 3.5 | Monitoring**
- 3.6 | ML Platform

There are essentially 3 different types of monitoring



- 3.1 | People
- 3.2 | Data
- 3.3 | Start small
- 3.4 | Production
- 3.5 | Monitoring
- 3.6 | ML Platform**



It's a cookiecutter template.

github.com/cookiecutter/cookiecutter

duarteocarmo.com - @duarteocarmo

What's the point in the end?

1. You're not LinkedIn? **Embrace** it!
2. **Boring** technology is **good** technology
3. MLOps is about **delivering value**
4. **Squeeze all the juice** from the orange

Thank you, questions?