¿Cómo montar un sistema de experimentación con mlflow?

PyConES 2023

¡GRACIAS!

¿Quien soy?

Maialen Berrondo (Mai)

Trabajo en Auth0 | Okta

Machine Learning Engineer

Contacto

Linkedin: Maialen Berrondo

Twitter: @MaialenBerrondo

Github : 13Mai13/pycones23 -> Todo el material subido

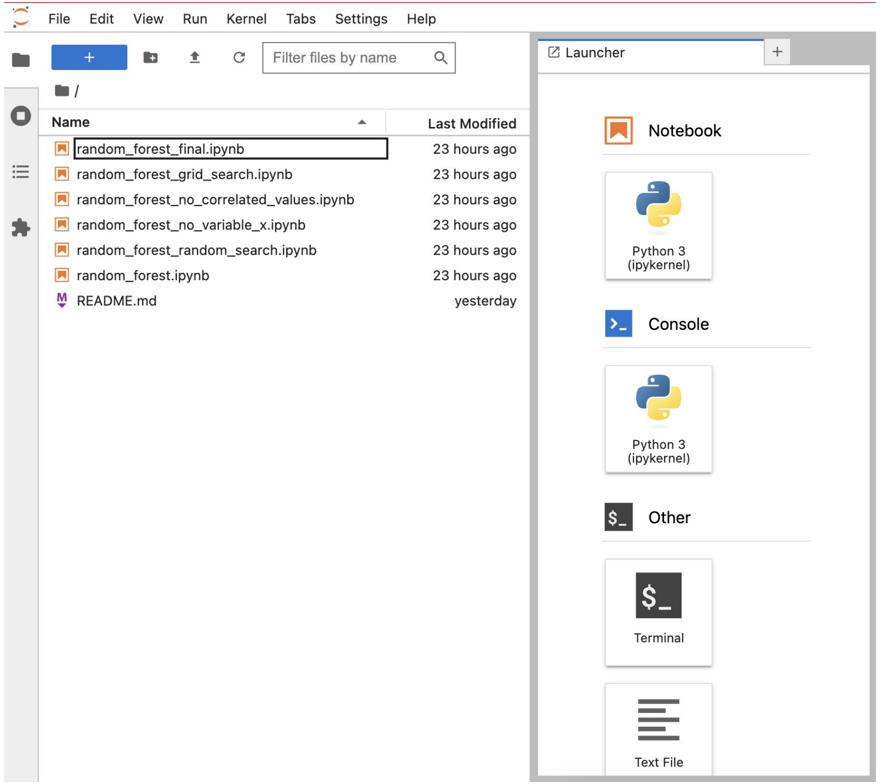
Introducción

¿Qué es mlflow?

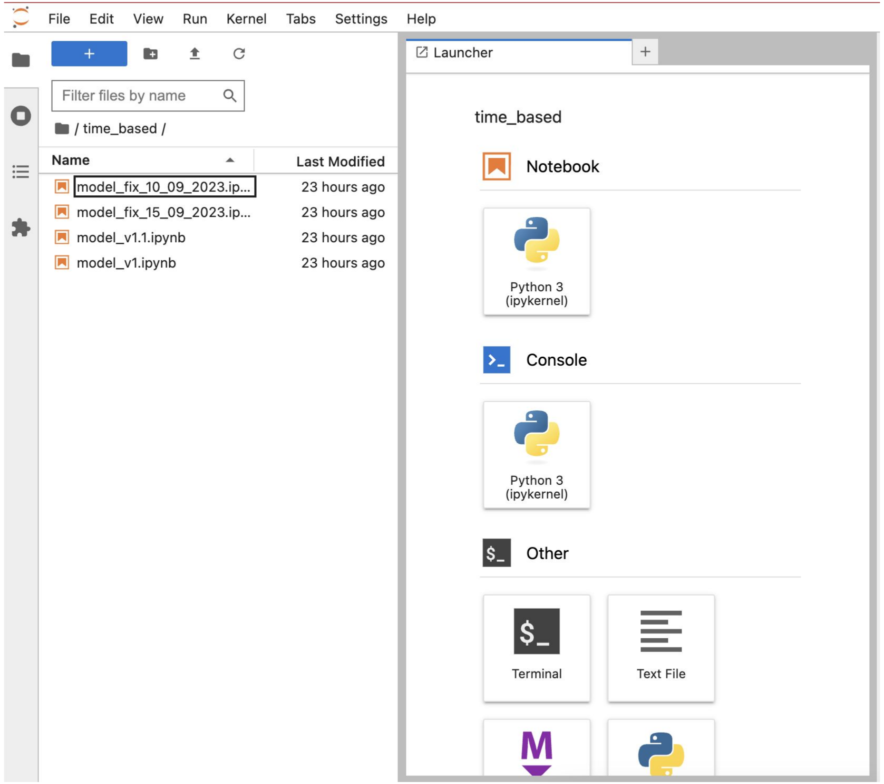
- Open Source
- Gestión de experimentos
- Reproducibilidad de resultados

Problema de la gestión de experimentos

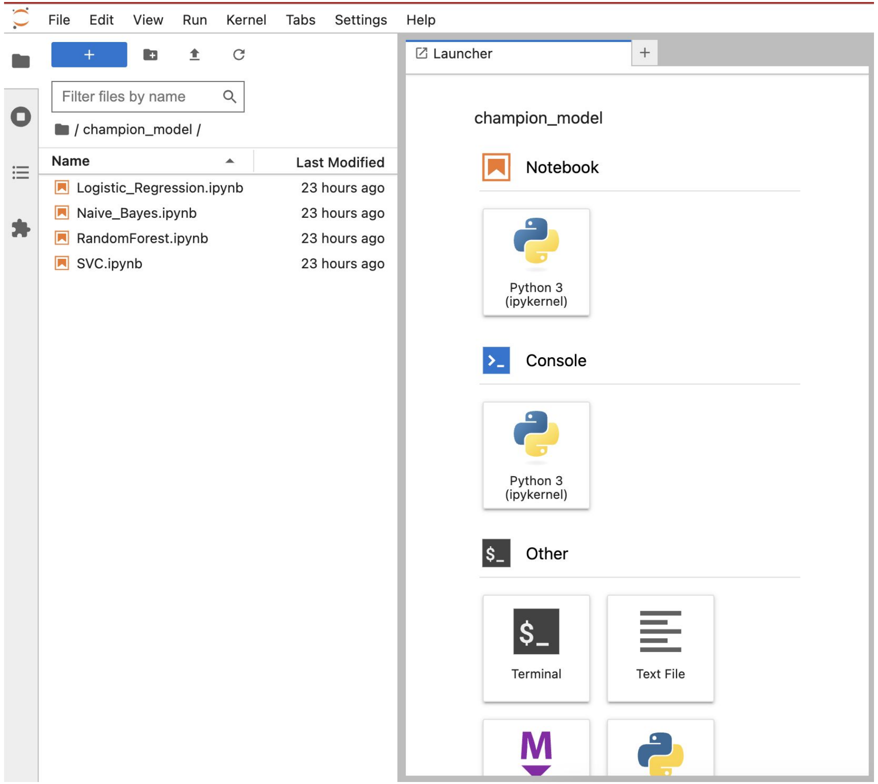
¿Cuál es el problema de la gestión de experimentos?



¿Cuál es el problema de la gestión de experimentos?



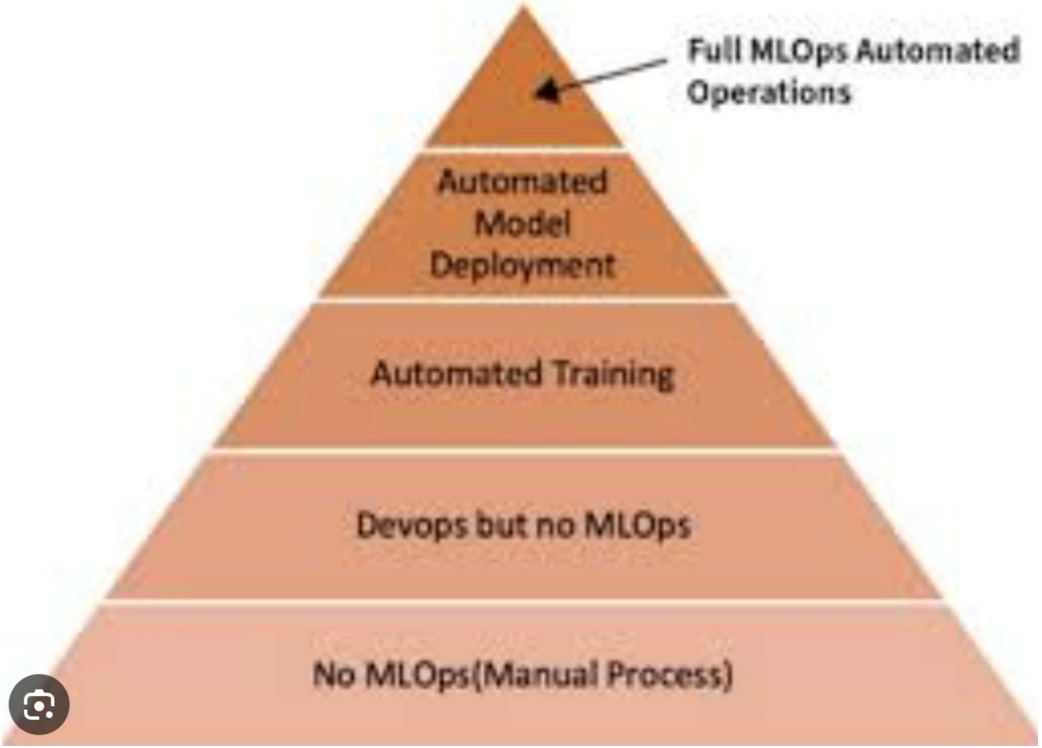
¿Cuál es el problema de la gestión de experimentos?



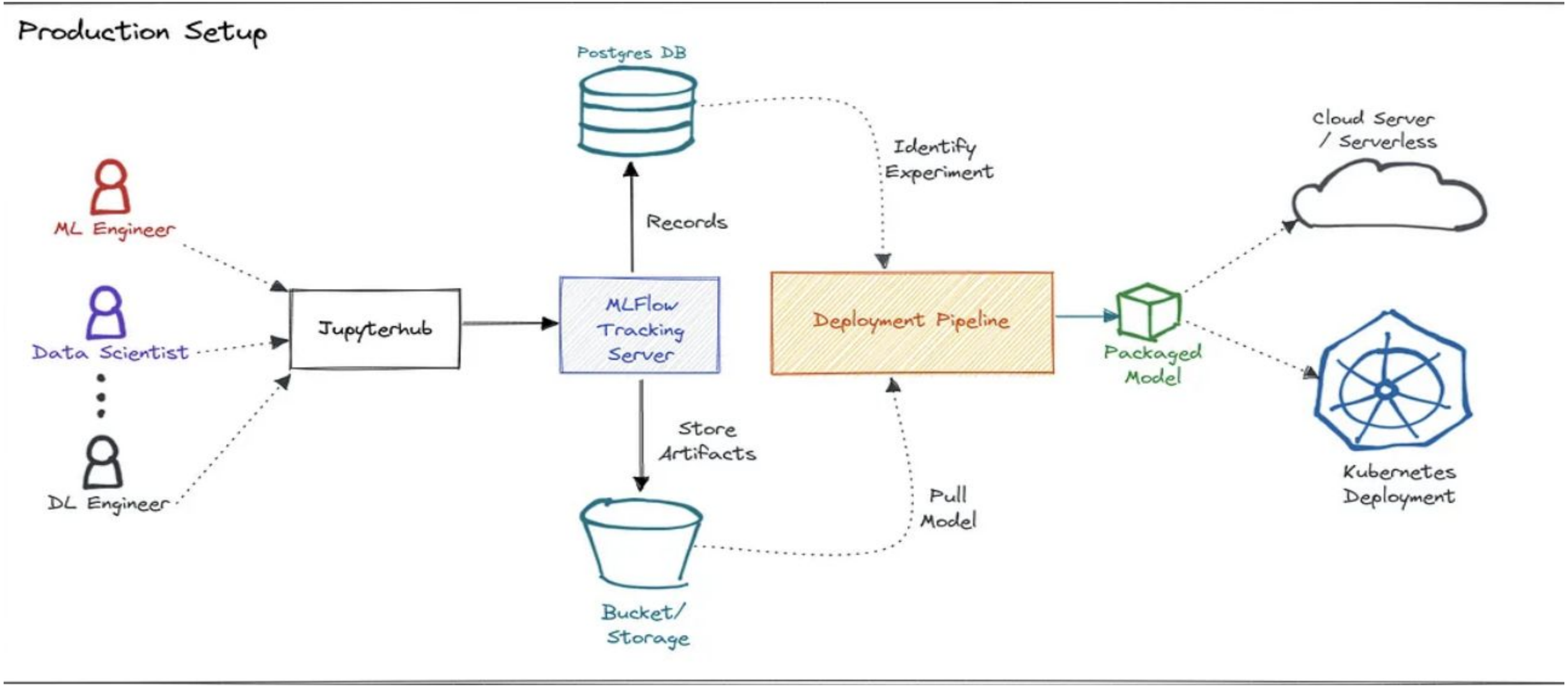
Problemas comunes

- Automatización del proceso de generación de modelos
- Gestión del ciclo de vida de modelos
- Colaboración entre equipos

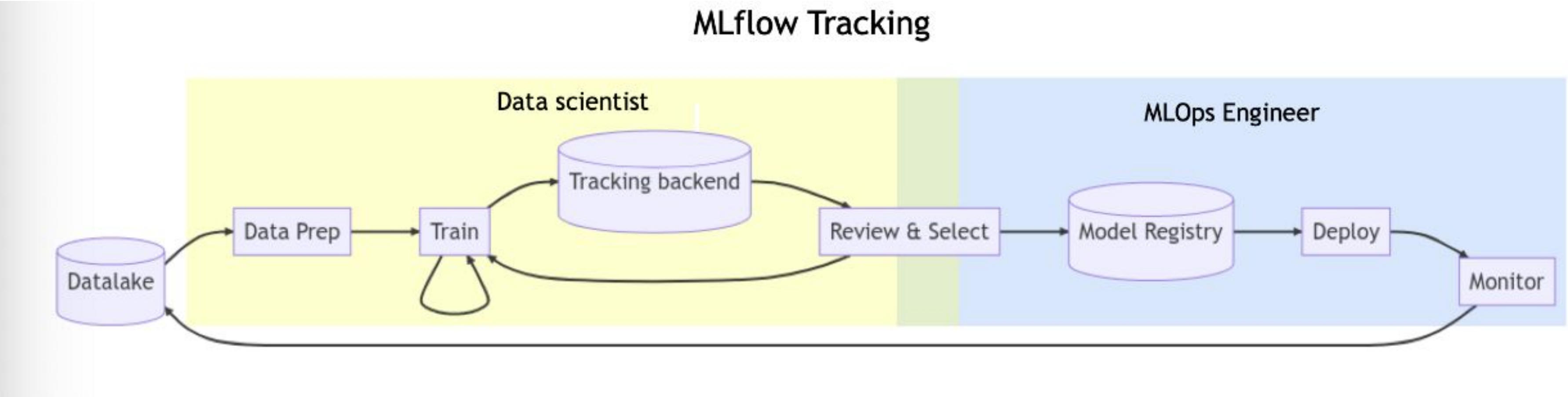
Maturity Model



Reproducibilidad y Colaboración



Reproducibilidad y Colaboración

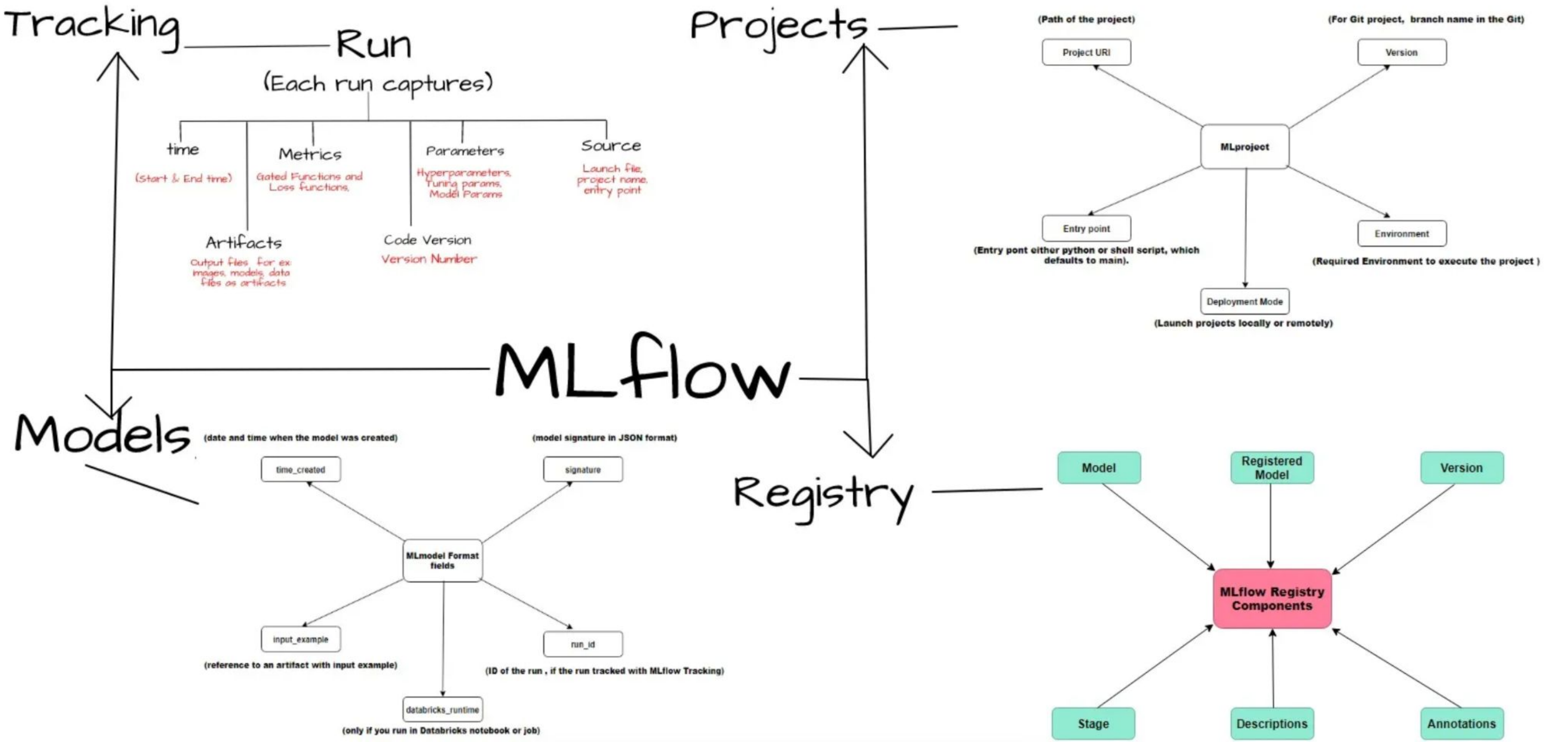


MLflow

¿ Qué ofrece MLflow?

- UI
- API para python
- Agnóstico en cuanto a: Cloud y base de datos
- Compatibilidad con notebooks

¿ Qué ofrece MLflow?



¿ Qué ofrece MLflow?

mlflow TRACKING	mlflow PROJECTS	mlflow MODEL REGISTRY	mlflow MODELS
Record and query experiments: code, data, config, and results.	Package data science code in a format that enables reproducible runs on many platforms	Store, annotate, and manage models in a central repository	Deploy machine learning models in diverse serving environments

Pasos para implementar MLflow

Instalación de mlflow

```
Untitled-1

conda install mlflow
```

```
Untitled-1

pip install mlflow
```

Mlflow en local

```
mlflow ui
```

Mlflow docker

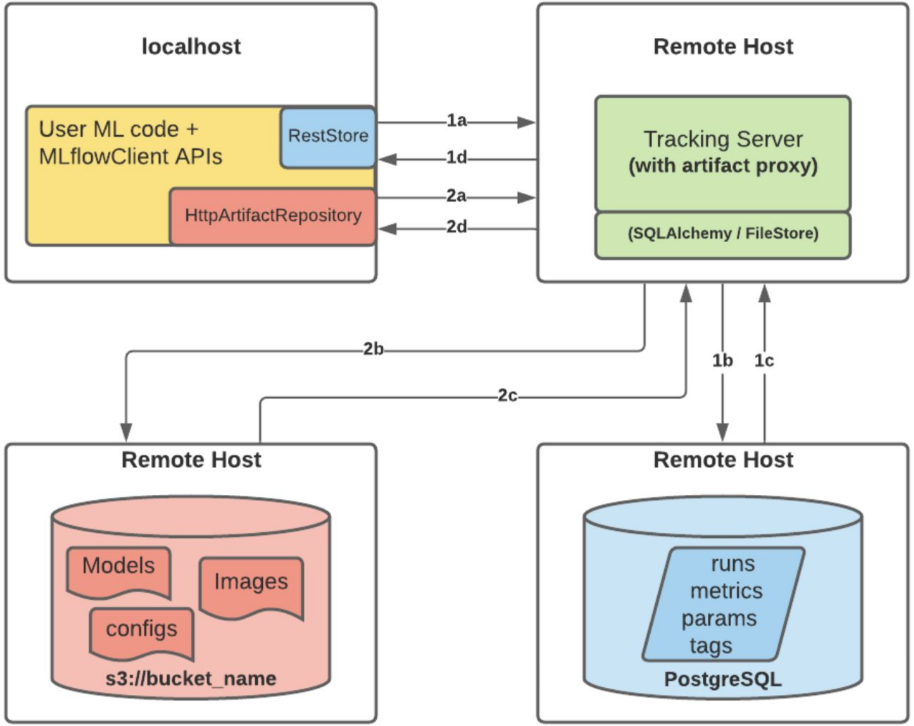
dockerfile

```
FROM python:3.11.0rc2-slim-bullseye

WORKDIR /home/mlflow

COPY entrypoint.sh .
COPY requirements.txt .

RUN apt update && apt upgrade -y

RUN apt -y install postgresql postgresql-contrib nginx

RUN pip install --upgrade pip && pip install -r requirements.txt

COPY nginx.conf /etc/nginx/nginx.conf

EXPOSE 8080

ENTRYPOINT ["/home/mlflow/entrypoint.sh"]
```

entrypoint.sh

```
#!/bin/bash

service nginx start

mlflow server \
  --host localhost \
  --port 5001 \
  --static-prefix /mlflow \
  --backend-store-uri
postgresql://${POSTGRES_USERNAME}:${POSTGRES_PASSWORD}@${POSTGRES_HOST}/mlflow \
  --default-artifact-root s3://mlflow-artifacts-eu-central-1/
```

Integración con sistemas de almacenamiento



Creación de un experimento en MLflow

experiment

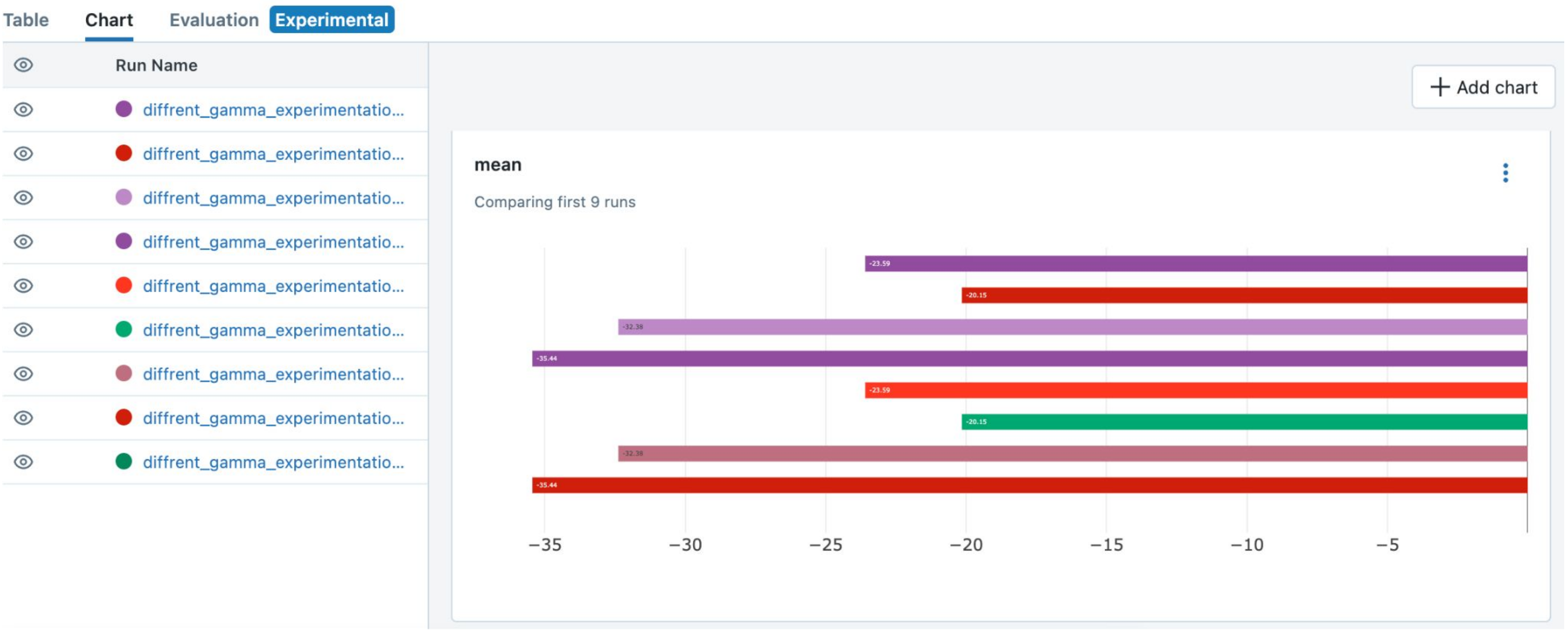
```
MLFLOW_TRACKING_URI = "http://localhost:5000/"

client = MlflowClient(tracking_uri=MLFLOW_TRACKING_URI)
mlflow.set_tracking_uri(MLFLOW_TRACKING_URI)
mlflow.set_experiment("v4")
```

Registro de parámetros, métricas y artefactos en un experimento

experiment

```
for gamma in [0.001, 0.01, 0.1, 0.5]:

    with mlflow.start_run(run_name=f"diffrent_gamma_experimentation_{gamma}") as
run:
        mlflow.log_input(mlflow.data.from_pandas(data), context="training")
        params = {'kernel':'rbf', 'C': 1e3, 'gamma': gamma}
        sk_learn_svr = SVR(**params)
        scores = cross_val_score(sk_learn_svr, x_scaled, y, cv=KFold(n_splits=10),
scoring='neg_mean_squared_error')

        mlflow.log_param("parameters", params)

        mlflow.log_metrics({'mean': scores.mean(), 'std': scores.std()})

        mlflow.sklearn.log_model(
            sk_model=sk_learn_svr,
            artifact_path="sklearn-model",
            registered_model_name="sk-learn-random-forest-reg-model"
        )
```

Seguimiento del rendimiento y comparación de experimentos



MLProjects

mlflow_projects

```
my_mlflow_project/
├── MLproject
├── conda.yaml
├── code/
│   ├── train.py
│   └── predict.py
└── data/
    └── dataset.csv
```

MLProjects

MLproject.yaml

```yaml
name: my_mlflow_project
entry_points:
  train:
    command: "python code/train.py"
    parameters:
      data_path: {type: str, description: "Path to the training data"}
      max_depth: {type: int, default: 5, description: "Maximum tree depth"}
  predict:
    command: "python code/predict.py"
    parameters:
      model_uri: {type: str, description: "URI of the trained model"}
      input_data: {type: str, description: "Input data for prediction"}
dependencies:
  conda:
    channels:
      - defaults
    dependencies:
      - scikit-learn=0.24.1
      - pandas=1.2.3
```

MLProjects

run_project

```
mlflow run my_mlflow_project/ -P data_path=data/dataset.csv -P max_depth=10
```

Versionado de modelos

mlflow.<model_flavor>.log_model()

```
mlflow.set_experiment("Model Registry")
m1 = 0.7
m2 = 0.8
with mlflow.start_run(run_name="first model run") as run:
    params = {"parameter1": 1, "parameter2": 2}
    sk_learn_model = """<Model of choice>(**params)"""

    # Log parameters and metrics using the MLflow APIs
    mlflow.log_params(params)
    mlflow.log_param("param_1", randint(0, 100))
    mlflow.log_metrics({"metric_1": m1, "metric_2": m2})

    # Log the sklearn model and register as version 1
    mlflow.sklearn.log_model(sk_model=sk_learn_model,
                             artifact_path="model",
                             registered_model_name="ModelOfChoice")
```

Versionado de modelos

mlflow.register_model()

```
result = mlflow.register_model("runs:/<run_id>/<path>", "model")
```

Versionado de modelos y etiquetado de versiones

create_registered_model()

```
from mlflow.tracking import MlflowClient
client = MlflowClient()

"""Creates an empty registered model with no version associated"""
client.create_registered_model("model-of-choice")
"""create a new version of the model"""
result = client.create_model_version(name="model",
                                     source="mlruns/0/<path>",
                                     run_id="<run_id>")
"""Renaming a Registered model"""
client.rename_registered_model(name="<old_model>",
                               new_name="<new_model>")
"""Promoting a model to Production stage"""
client.transition_model_version_stage(name="<model>",
                                      version=2,
                                      stage="Production")
```

Despliegue de modelos en diferentes entornos

```
mlflow.pyfunc.load_model()
```

```
mlflow models build-docker -m runs:/<RUN_ID>/model -n my_model
docker run -p 8080:8080 my_model
```

Despliegue de modelos en diferentes entornos

Untitled-1

```
import mlflow.sagemaker

mlflow.sagemaker.deploy(
    model_uri="runs:/<RUN_ID>/model",
    role="arn:aws:iam::123456789:role/service-role/MySageMakerRole",
    endpoint_name="my-endpoint"
)
```

Despliegue de modelos en diferentes entornos

Untitled-1

```python
import mlflow.azureml

mlflow.azureml.register(model_uri="runs:/<RUN_ID>/model",
                        model_name="my-model",
                        workspace_name="my-workspace",
                        subscription_id="your-subscription-id",
                        resource_group="your-resource-group")
```

Despliegue de modelos en diferentes entornos

```python
import mlflow.gcp

mlflow.gcp.deploy(
    model_uri="runs:/<RUN_ID>/model",
    model_name="my-model",
    project_id="your-project-id",
    region="us-central1",
    runtime_version="2.3",
)
```

Mlflow serving of model

serving models - mlflow serve

```
mlflow models serve -m /Users/mlflow/mlflow-
prototype/mlruns/0/7c1a0d5c42844dcdb8f5191146925174/artifacts/model -p 1234


curl -X POST -H "Content-Type:application/json; format=pandas-split" --data
'{"columns":["alcohol", "chlorides", "citric acid", "density", "fixed acidity",
"free sulfur dioxide", "pH", "residual sugar", "sulphates", "total sulfur dioxide",
"volatile acidity"],"data":[[12.8, 0.029, 0.48, 0.98, 6.2, 29, 3.33, 1.2, 0.39, 75,
0.66]]}' http://127.0.0.1:1234/invocations
```

Deep Dive

Autologging

PyTorch

▾ Metrics

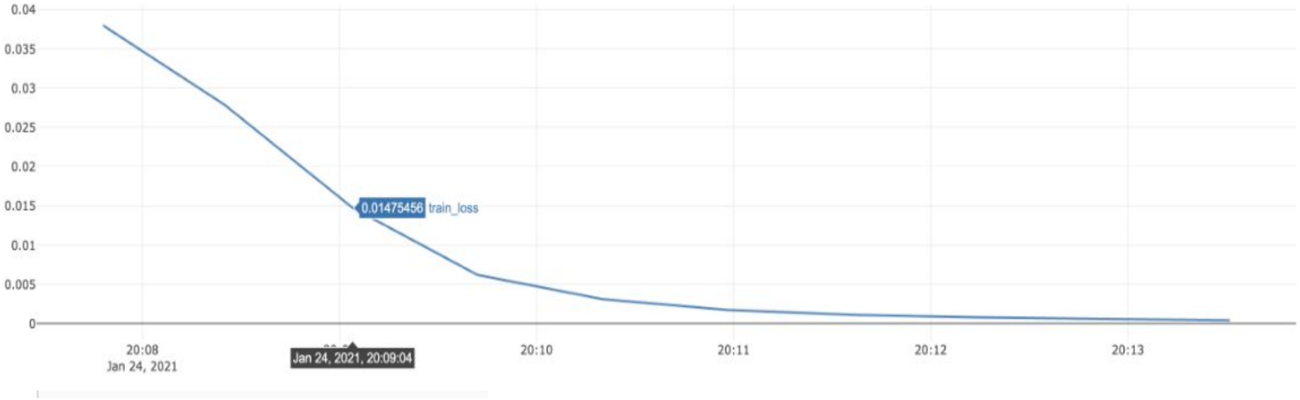
Name	Value
acc	0
acc_epoch	0.003
acc_step	0
train_loss	0.028
train_loss_epoch	0.089
train_loss_step	0.028

PyTorch

https://mlflow.org/docs/latest/python_api/mlflow.pytorch.html

https://bytepawn.com/automatic-mlflow-logging-for-pytorch.html

PyTorch



0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0.01475456 train_loss
20:08
Jan 24, 2021
Jan 24, 2021, 20:09:04
20:10
20:11
20:12
20:13

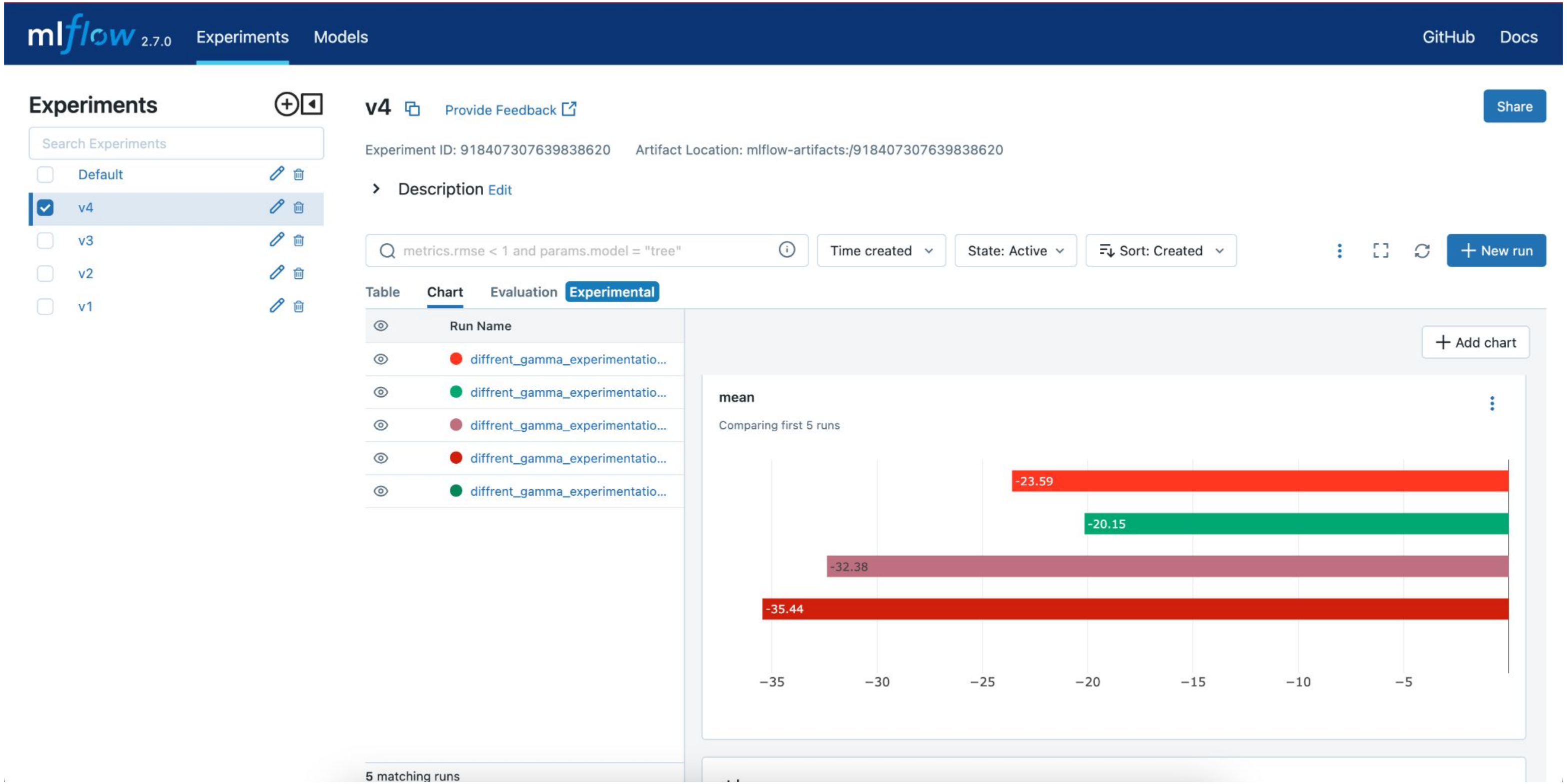
Sklearn & TensorFlow

https://mlflow.org/docs/latest/python_api/mlflow.sklearn.html

https://mlflow.org/docs/latest/python_api/mlflow.tensorflow.html

Monitoreo y visualización de experimentos

Interfaz web de MLflow



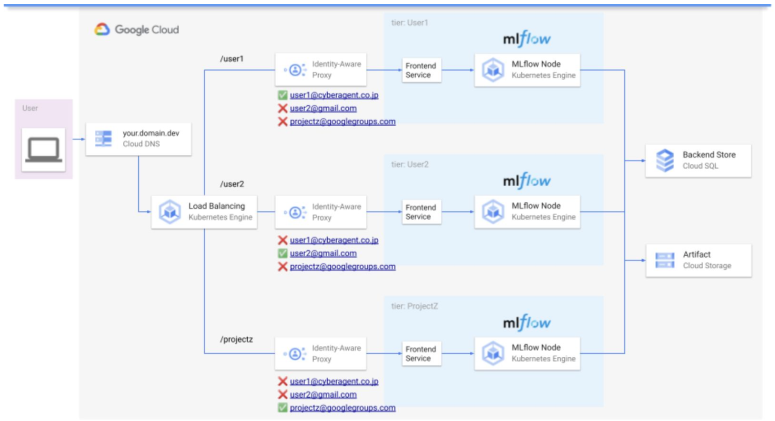
Escalabilidad y producción

Escalabilidad horizontal de mlflow

- Clusters -> Integración con multiples plataformas

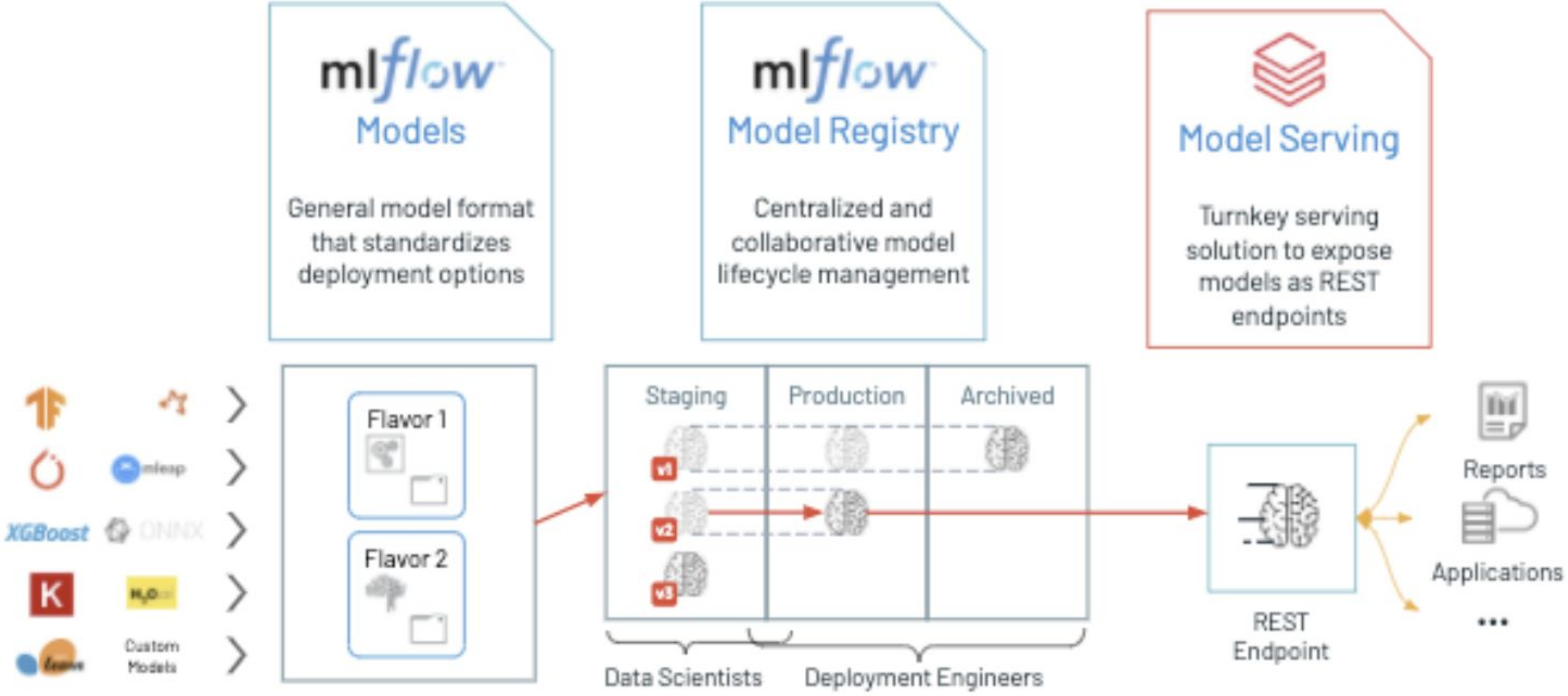
- Paralelismo

- Almacenamiento y monitoreo distribuido

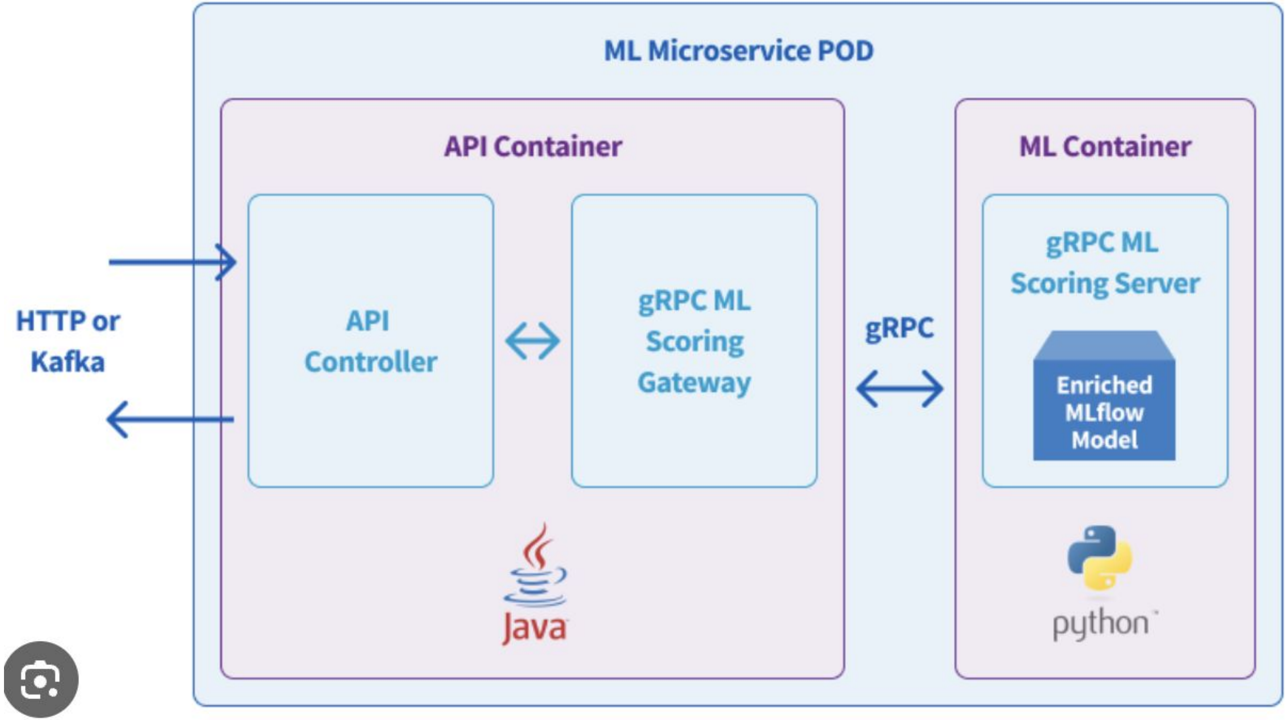


MLFlow + k8s:
https://medium.com/artefact-engineering-and-data-science/serving-ml-models-at-scale-using-mlflow-on-kubernetes-a83390718a92

Implementación en entornos de producción

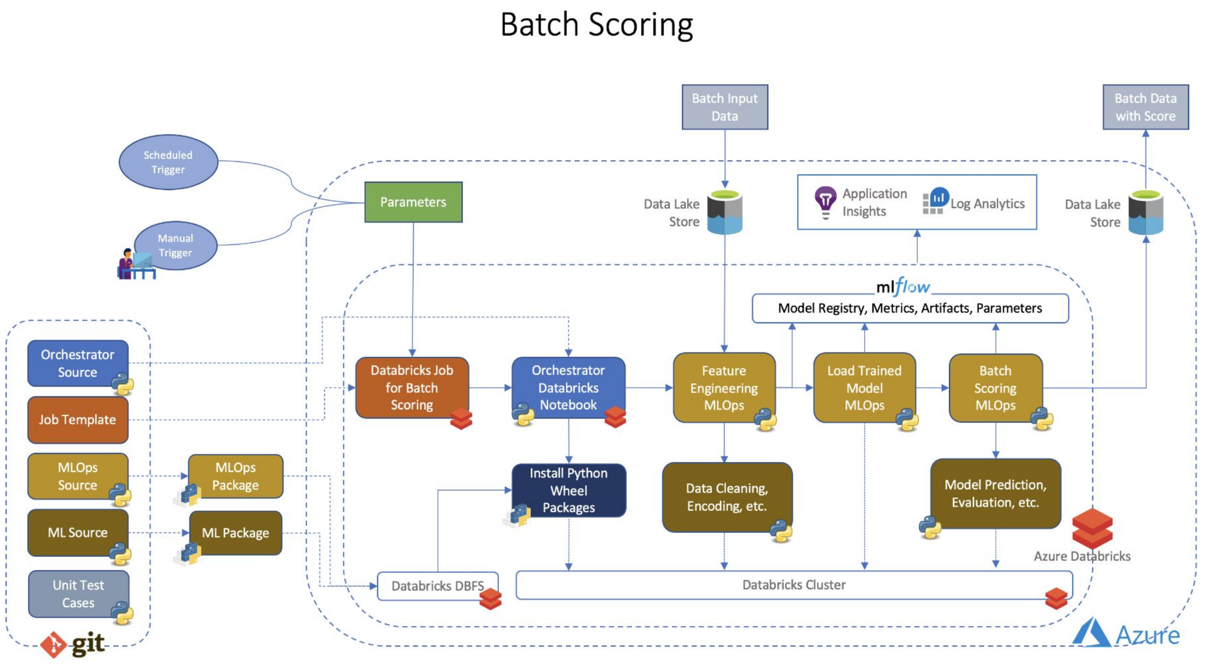


Implementación en entornos de producción



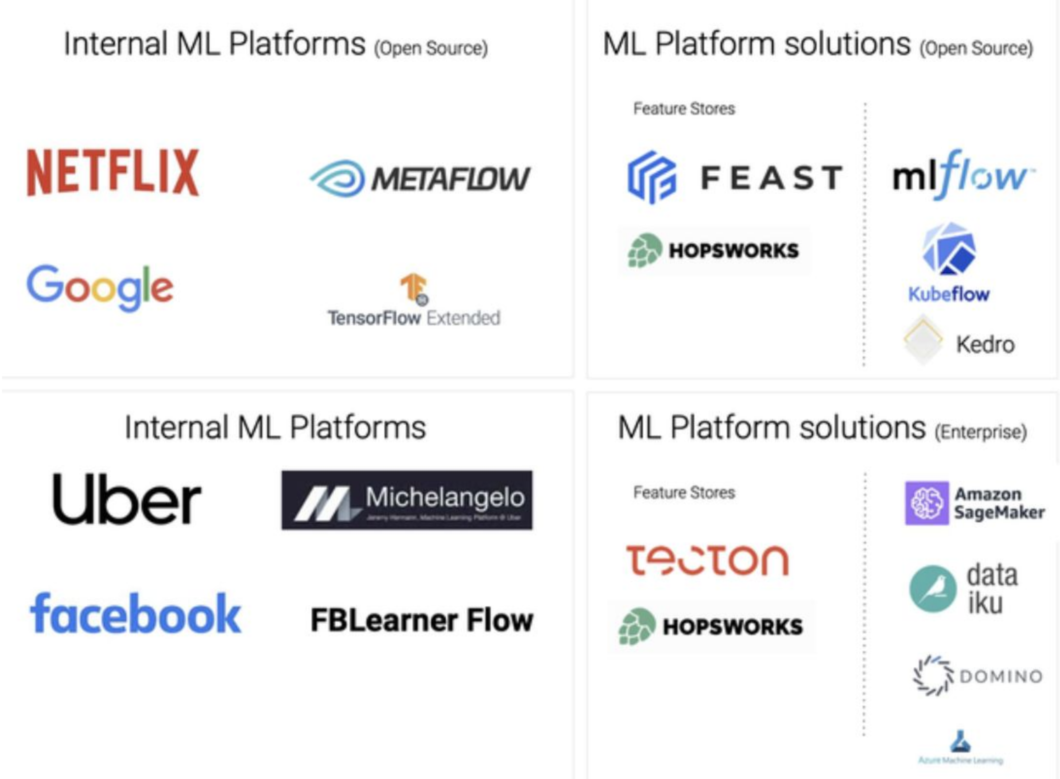
Implementación en entornos de producción

Batch Scoring



Otras soluciones

Diferentes industrias



Kubeflow

Kubeflow vs MLflow

Kubeflow vs MLflow

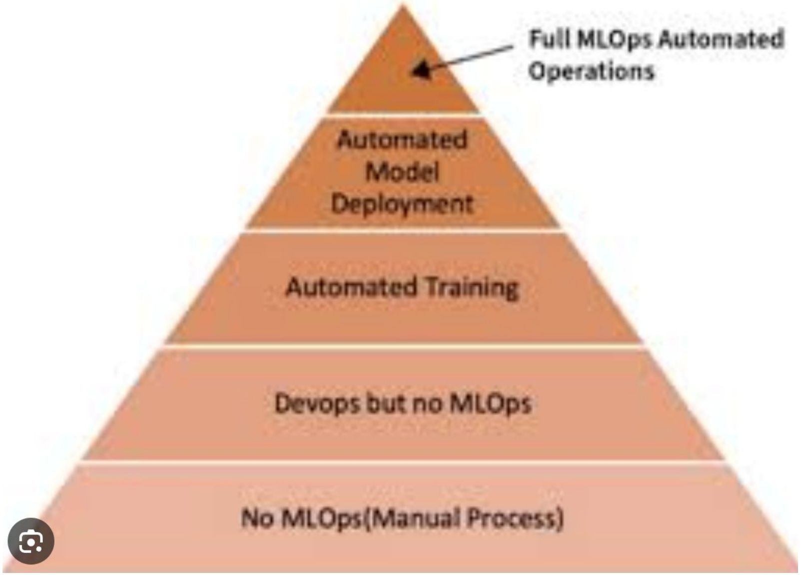
Kubeflow

mlflow™

Conclusiones y Recomendaciones

Recomendaciones

- Separación por experimentos
- Automatización de pipelines
- Versionado de modelos
- Separar experimentación de producción



¿Preguntas?

Recursos de interés

- Documentación oficial: https://mlflow.org/docs/latest/index.html

- Github: https://github.com/13Mai13/pycones23/

- Registro de modelos: https://medium.com/walmartglobaltech/model-and-data-versioning-an-introduction-to-mlflow-and-dvc-260347cd0f6e

- Python

¡GRACIAS DE NUEVO!

Contacto

Linkedin: Maialen Berrondo

Twitter: @MaialenBerrondo

Github : 13Mai13/pycones23 -> Todo el material subido