

# 3rd -MLOps

Speaker: 류창훈



# AI명예학회

SKHU

# 목차

- **Model Development**

- Model 구축 (Iris dataset)
- Pipeline
- Data 불러오기 (feat. DB)

- **Model Registry**

- MLFlow + MINIO flow



## - Model Development

### 라이브러리 설치

VS Code 터미널에

`pip install pandas`

`pip install scikit-learn`

`pip install joblib`

설치 되어 있을 수도 있는데  
혹시 모르니 한번 더 명령어 실행시켜서 확인해봐요~

## - Model Development

### 대표적 모델 저장 확장자

**.pkl or .pickle**: 보통 머신러닝 모델 저장할 때.

**.joblib**: scikit-learn 모델 저장할 때.

**.h5**: 딥러닝 모델 저장할 때.

**.pt**: Pytorch

**.pb**: TensorFlow

**.tflite**

등등

## - Model Development

# Pipeline

파이프라인?

쇠 파이프를 생각해보자.

보통 이것만 그냥 쓰지 않고, 무언가에 붙이고, 서로 이어서 쓰고, 그러지 않는가?

이처럼, 어떤 여러 단계들을 순차적으로, 막 끼 넣어서 한번에 쓸 수 있게 하는 도구를  
우리는 '**Pipeline**'이라고 한다.

ex) transform + predict

## - Model Development

# Query문

DB에 어떤 작업을 요청하기 위한 명령어 혹은 문장.

```
SELECT * FROM users;
```

```
SELECT * FROM iris_data ORDER BY id DESC LIMIT 100;
```

```
SELECT name, email FROM users;
```

```
SELECT * FROM orders WHERE amount > 100;
```

```
SELECT * FROM orders ORDER BY created_at DESC;
```

## - Model Development

근데 이걸 왜 쓸까?

정의는

‘DBMS에서 데이터를 효율적으로 검색, 조작, 관리 하기 위함’

이라고 한다.

Pandas 와 같은 라이브러리로 쿼리문과 동일하게 작업 가능.

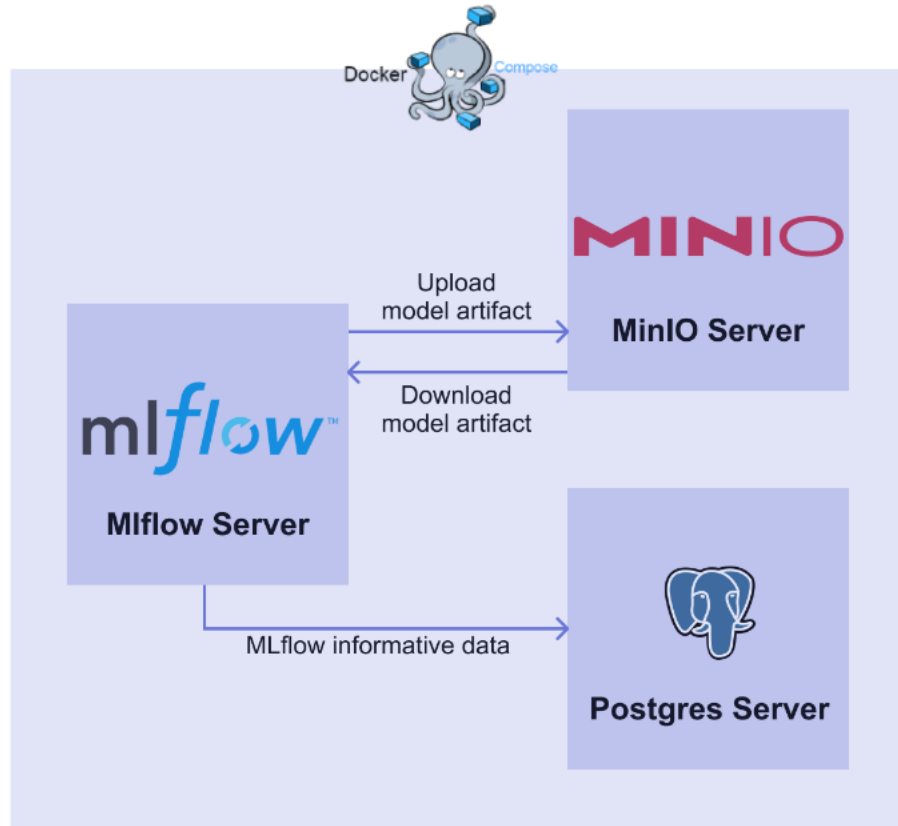
하지만, 이건 메모리에 데이터 **전체가 로드**되어야 가능한 것.

메모리에 부담을 주지 않게, DB로 한번 거치고, 간소화 해서 우리가 컴퓨터 다운 안되게 작업할 수 있다.

**코드 실습**



## - Model Registry

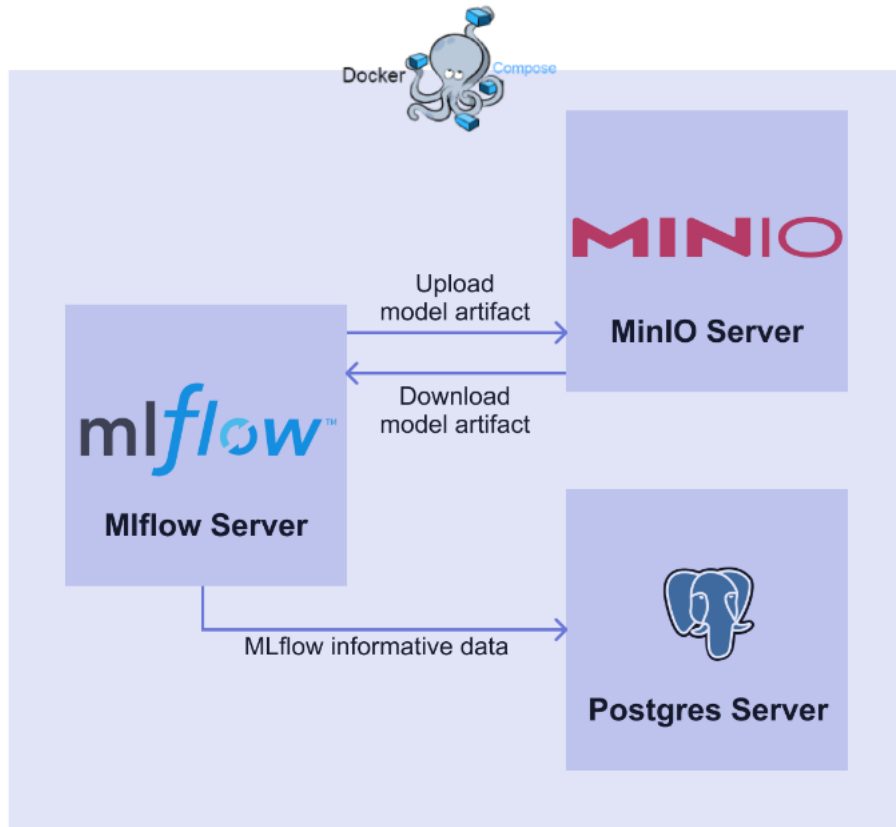


PostgreSQL: 그냥 DB, 데이터 저장소.

MinIO: 대용량 데이터 스토리지.

MLflow: 머신러닝 실험, 모델 관리.

## - Model Registry



전체적인 흐름을 봐보자.

어떤 모델, 출력결과를 MLflow에서 관리.

이걸 MINIO에 저장. 원한다면 여기서 또 데이터 끌어다가 쓰고,

MLflow의 어떤 무수한 정보들을 PostgreSQL에 저장하는 흐름.

## - Model Registry

사실 이런것도 대기업 Cloud 서비스를 이용하면 참 편하다.

(GCP, AWS, Azure 등등...)

근데 우리는 돈이 많지 않다....

무료인 이런것들도 한번 맛 봐보자.

## - Model Registry

깃허브 > study\_Resources\_MLOps 폴더 들어가기.

MLE\_Ch3 전체 다운.

# - Model Registry

```
SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
postgres 사용자의 암호:
psql (16.1)
도움말을 보려면 "help"를 입력하십시오.

postgres=# \d
릴레이션 목록
스키마 | 이름 | 형태 | 소유주
-----+-----+-----+-----
public | iris_data | 테이블 | postgres
public | iris_data_id_seq | 시퀀스 | postgres
(2개 행)

postgres=# |
```

```
Dockerfile > ...
1 FROM python:3.9-slim
2
3 RUN apt-get update && apt-get install -y \
4     git \
5     wget \
6     && rm -rf /var/lib/apt/lists/*
7
8 RUN pip install -U pip && \
9     pip install boto3==1.26.8 mlflow==1.30.0 psycpg2-binary
10
11 RUN cd /tmp && \
12     wget https://dl.min.io/client/mc/release/linux-amd64/mc && \
13     chmod +x mc && \
14     mv mc /usr/bin/mc
```

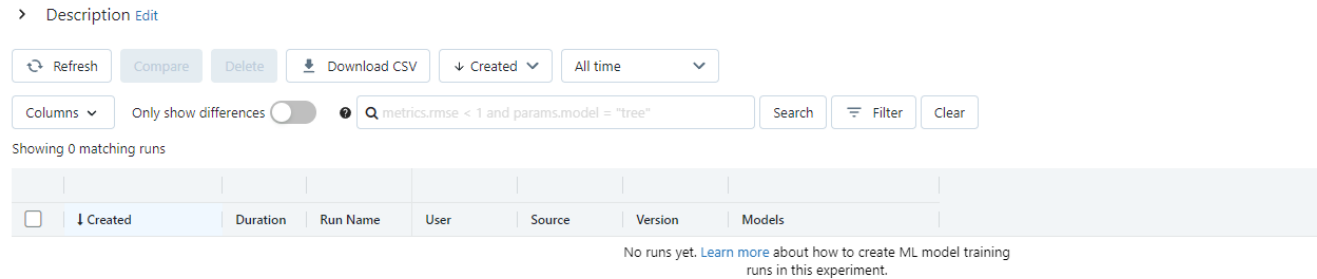
```
docker-compose.yml
1 version: "3"
2
3 services:
4   mlflow-backend-store:
5     image: postgres:14.0
6     container_name: mlflow-backend-store
7     environment:
8       POSTGRES_USER: mlflowuser
9       POSTGRES_PASSWORD: mlflowpassword
10      POSTGRES_DB: mlflowdatabase
11
12   mlflow-artifact-store:
13     image: minio/minio
14     container_name: mlflow-artifact-store
15     ports:
16       - 9000:9000
17       - 9001:9001
18     environment:
19       MINIO_ROOT_USER: minio
20       MINIO_ROOT_PASSWORD: miniostorage
21     command: server /data/minio --console-address :9001
22
23   mlflow-server:
24     build:
25       context: .
26       dockerfile: Dockerfile
27     container_name: mlflow-server
28     depends_on:
29       mlflow-backend-store:
30         condition: service_started
31       mlflow-artifact-store:
32         condition: service_started
33     ports:
34       - 5001:5000
35     environment:
36       AWS_ACCESS_KEY_ID: minio
37       AWS_SECRET_ACCESS_KEY: miniostorage
38       MLFLOW_S3_ENDPOINT_URL: http://mlflow-artifact-store:9000
39     command:
40       - /bin/sh
41       - -c
42       - |
43         mc config host add mlflowminio http://mlflow-artifact-store:9000 minio miniostorage &&
44         mc mb -ignore-existing mlflowminio/mlflow
45         mlflow server \
46           nd-store/mlflowdatabase \
```

5.38 GB / 5.38 GB in use 9 images Last refresh: 13 hours ago

<input type="checkbox"/>	Name	Tag	Status	Created	Size	Actions
<input type="checkbox"/>	<a href="#">ryuchanghoon/mc</a> d46448048aea	latest	In use	3 days ago	1.83 GB	<a href="#">▶</a> <a href="#">⋮</a>
<input type="checkbox"/>	<a href="#">ryuchanghoon/test-rest</a> 78c1a45db263	latest	In use	4 days ago	1.83 GB	<a href="#">▶</a> <a href="#">⋮</a>
<input type="checkbox"/>	<a href="#">ryuchanghoon/data-gene</a> 007124fbb9a4	latest	In use	6 days ago	591.91 MB	<a href="#">▶</a> <a href="#">⋮</a>
<input type="checkbox"/>	<a href="#">flask-re-server</a> c6e2816feaaf	latest	In use	8 days ago	1.83 GB	<a href="#">▶</a> <a href="#">⋮</a>
<input type="checkbox"/>	<a href="#">ryuchanghoon/flask-re-server</a> c6e2816feaaf	latest	In use	8 days ago	1.83 GB	<a href="#">▶</a> <a href="#">⋮</a>
<input type="checkbox"/>	<a href="#">postgres</a> 317a302c7480	14.0	In use	2 years ago	374.08 MB	<a href="#">▶</a> <a href="#">⋮</a>

## - Model Registry

localhost:5001 # mlflow 접속 확인.

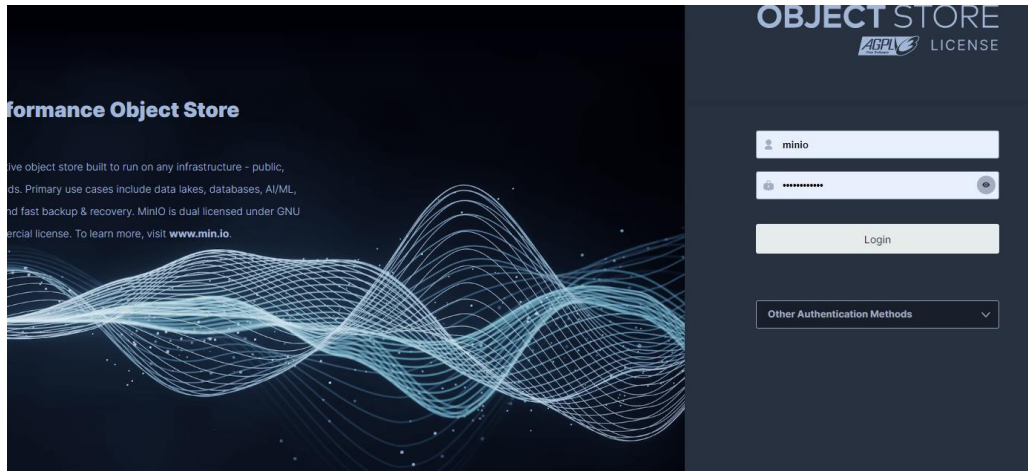


터미널에

`docker-compose up -d`

```
PS C:\Users\jch\Desktop\테스트\테스트\mlflow_test_code\mlflow_test> docker-compose up -d
[+] Running 3/3
✓ Container mlflow-artifact-store Started 0.0s
✓ Container mlflow-backend-store Started 0.0s
✓ Container mlflow-server Started 0.0s
```

localhost:9001 # MinIO 접속 확인



Name	Objects	Size	Access
mlflow	6	5.6 KiB	R/W

## - Model Registry

```
save_model_to_registry.py > ...
1  import os
2  from argparse import ArgumentParser
3
4  import mlflow
5  import pandas as pd
6  import psycopg2
7  from sklearn.metrics import accuracy_score
8  from sklearn.model_selection import train_test_split
9  from sklearn.pipeline import Pipeline
10 from sklearn.preprocessing import StandardScaler
11 from sklearn.svm import SVC
12
13
14
15 os.environ["MLFLOW_S3_ENDPOINT_URL"] = "http://localhost:9000"
16 os.environ["MLFLOW_TRACKING_URI"] = "http://localhost:5001"
17 os.environ["AWS_ACCESS_KEY_ID"] = "minio"
18 os.environ["AWS_SECRET_ACCESS_KEY"] = "miniostorage"
19
20
21 db_connect = psycopg2.connect(
22     user="postgres",
23     password= '...',
24     host="localhost",
25     port=5432,
26     database="postgres",
27 )
28 df = pd.read_sql("SELECT * FROM iris_data ORDER BY id DESC LIMIT 100", db_connect)
29
30 X = df.drop(["id", "timestamp", "target"], axis="columns")
31 y = df["target"]
32 X_train, X_valid, y_train, y_valid = train_test_split(X, y, train_size=0.8, random_state=2022)
33
34
35 model_pipeline = Pipeline([("scaler", StandardScaler()), ("svc", SVC())])
36 model_pipeline.fit(X_train, y_train)
37
38 train_pred = model_pipeline.predict(X_train)
39 valid_pred = model_pipeline.predict(X_valid)
```

# - Model Registry

터미널 >

```
python save_model_to_registry.py --model-name "sk_model"
```

Experiments

☐ Default

☒ new-exp

+

new-exp

Track machine learning training runs in experiments. [Learn more](#)

Experiment ID: 1

> Description Edit

Refresh

Compare

Delete

Download CSV

Created

All time

Columns

Only show differences

Search

Filter

Clear

Showing 1 matching run

	Created	Duration	Run Name	User	Source	Version	Models	train_acc	valid_acc
<input type="checkbox"/>	12 hours ago	5.7s	puzzled-dee...	rch	save_mo...	-	sklearn	0.963	1

Load more

sk\_model

MLmodel

conda.yaml

input\_example.json

model.pkl

python\_env.yaml

requirements.txt

Full Path:s3://mlflow/1/

MLflow Model

The code snippets bel

Model schema

Input and output scherr

Name

Inputs (4)



# - Model Registry

```
load_model_from_registry.py > ...
1  import os
2  from argparse import ArgumentParser
3
4  import mlflow
5  import pandas as pd
6  from sklearn.metrics import accuracy_score
7  from sklearn.model_selection import train_test_split
8
9
10 os.environ["MLFLOW_S3_ENDPOINT_URL"] = "http://localhost:9000"
11 os.environ["MLFLOW_TRACKING_URI"] = "http://localhost:5001"
12 os.environ["AWS_ACCESS_KEY_ID"] = "minio"
13 os.environ["AWS_SECRET_ACCESS_KEY"] = "miniostorage"
14
15
16 parser = ArgumentParser()
17 parser.add_argument("--model-name", dest="model_name", type=str, default="sk_model")
18 parser.add_argument("--run-id", dest="run_id", type=str)
19 args = parser.parse_args()
20
21 model_pipeline = mlflow.sklearn.load_model(f"runs://{args.run_id}/{args.model_name}")
22
23
24 df = pd.read_csv("data.csv")
25
26 X = df.drop(["id", "timestamp", "target"], axis="columns")
27 y = df["target"]
28 X_train, X_valid, y_train, y_valid = train_test_split(X, y, train_size=0.8, random_state=2022)
29
30
31 train_pred = model_pipeline.predict(X_train)
32 valid_pred = model_pipeline.predict(X_valid)
33
34 train_acc = accuracy_score(y_true=y_train, y_pred=train_pred)
35 valid_acc = accuracy_score(y_true=y_valid, y_pred=valid_pred)
36
37 print("Train Accuracy :", train_acc)
38 print("Valid Accuracy :", valid_acc)
```

# - Model Registry

new-exp > puzzled-deer-605

## puzzled-deer-605

Run ID: 325821539d0a41d3b9729aee625e5d2

Date: 2024-01-13 22:16:55

User: rch

Duration: 5.7s

Lifecycle Stage: active

> Description [Edit](#)

> Parameters

> Metrics (2)

> Tags

▼ Artifacts

▼ sk\_model

- MLmodel
- conda.yaml
- input\_example.json
- model.pkl
- python\_env.yaml
- requirements.txt

Full Path: s3://mlflow/1/325821539d0a41d3b9729aee625e5d2/artifacts/sk\_model

### MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. You can also [register it to the model registry](#)

#### Model schema

Input and output schema for your model. [Learn more](#)

Name	Type
------	------

#### Inputs (4)

canal length	double
--------------	--------

#### Make Predictions

Predict on a Spark DataFrame:

```
import mlflow
from pyspark.sql.functions import struct
logged_model = 'runs:/325821539d0a41d3b9729aee625e5d2/artifacts/sk_model'

# Load model as a Spark UDF. Override
loaded_model = mlflow.pyfunc.spark_udf(spark, logged_model)
```



## - Model Registry

터미널 >

```
python load_model_from_registry.py --model-name "sk_model" --run-id "RUN_ID"
```

> Parameters

▼ Metrics (2)

Name	Value
train_acc 	0.963
valid_acc 	1

> Tags