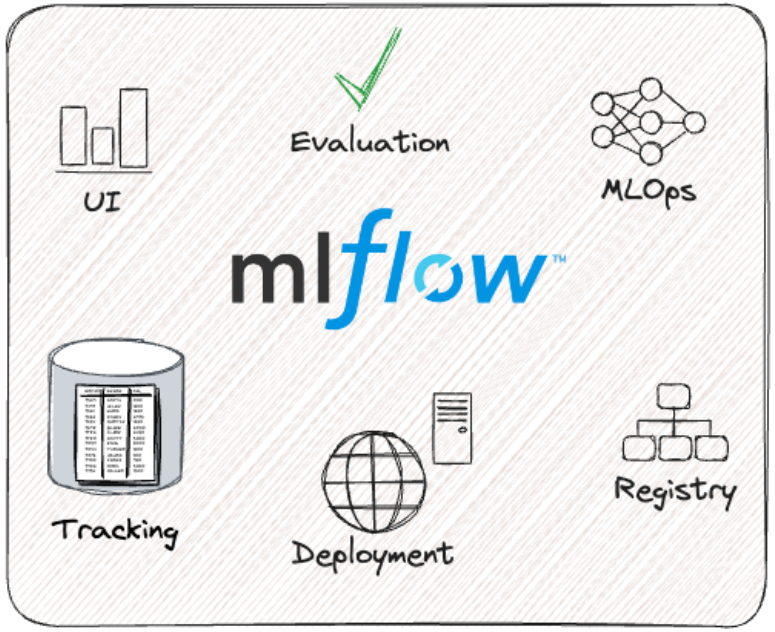
# MLFLOW

## A TOOL FOR MANAGING THE MACHINE LEARNING LIFECYCLE

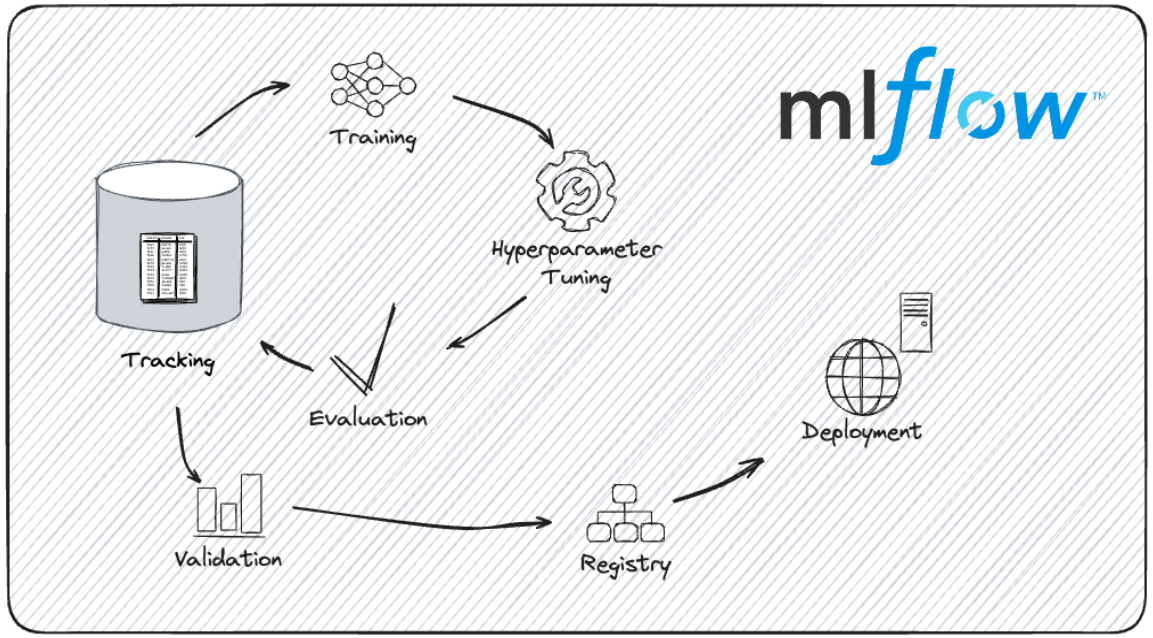
By: Novelya Putri Ramadhani

# APA ITU MLFLOW?

MLflow adalah platform open-source yang dirancang khusus untuk membantu praktisi dan tim machine learning dalam menangani kompleksitas proses machine learning. MLflow berfokus pada seluruh lifecycle proyek machine learning, memastikan bahwa setiap tahap dapat dikelola, dilacak, dan direproduksi.




The Model Development Lifecycle with MLflow



# FITUR UTAMA

## TRACKING

MLflow Tracking menyediakan pencatatan eksperimen yang komprehensif, pelacakan parameter, visualisasi metrik, dan manajemen artefak.

Manfaat Utama:

- Organisasi Eksperimen: Melacak dan membandingkan berbagai eksperimen model
- Visualisasi Metrik: Plot dan grafik bawaan untuk performa model
- Penyimpanan Artefak: Menyimpan model, plot, dan file lainnya untuk setiap run
- Kolaborasi: Berbagi eksperimen dan hasil antar tim

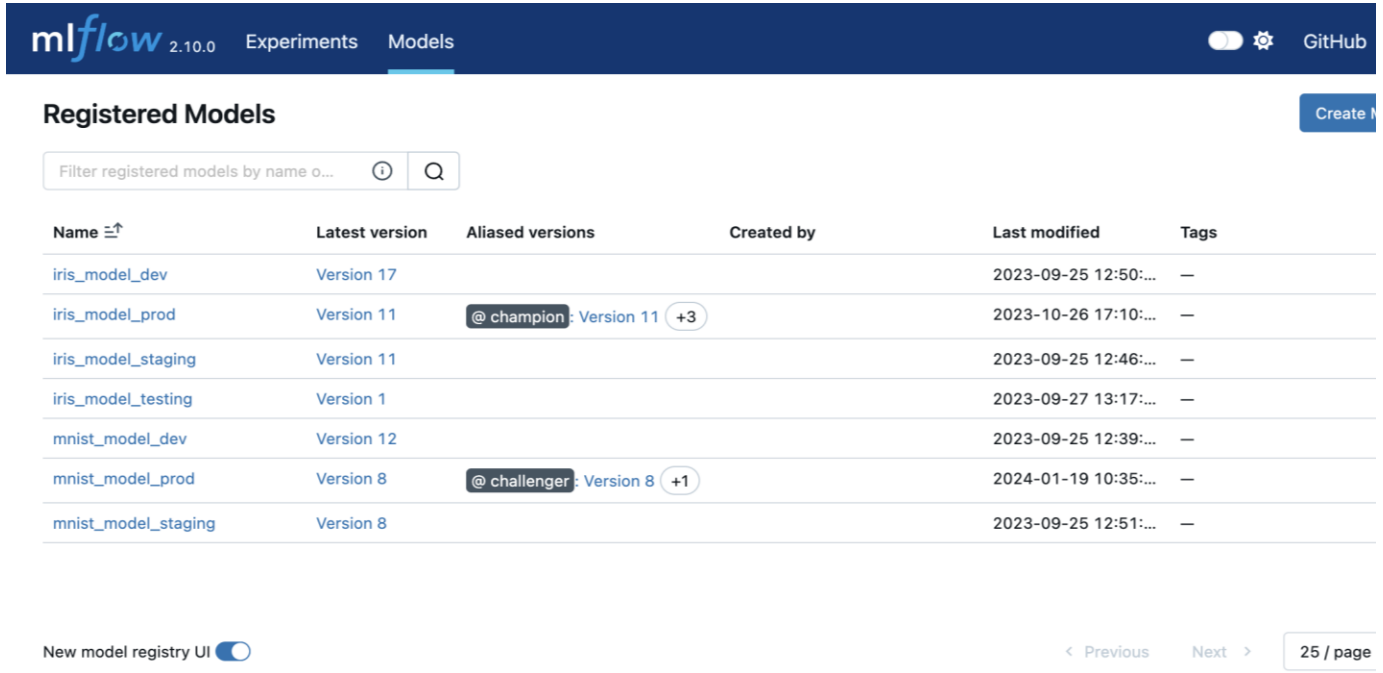
# FITUR UTAMA

## MODEL REGISTRY

MLflow Model Registry menyediakan versi model terpusat, manajemen tahap (stage), dan pelacakan asal-usul (lineage) model.

Manfaat Utama:

- Kontrol Versi: Melacak versi model dengan lineage otomatis
- Manajemen Tahap: Mempromosikan model melalui tahap staging, production, dan archived
- Kolaborasi: Alur kerja peninjauan dan persetujuan model berbasis tim
- Penemuan Model: Mencari dan menemukan model di seluruh organisasi



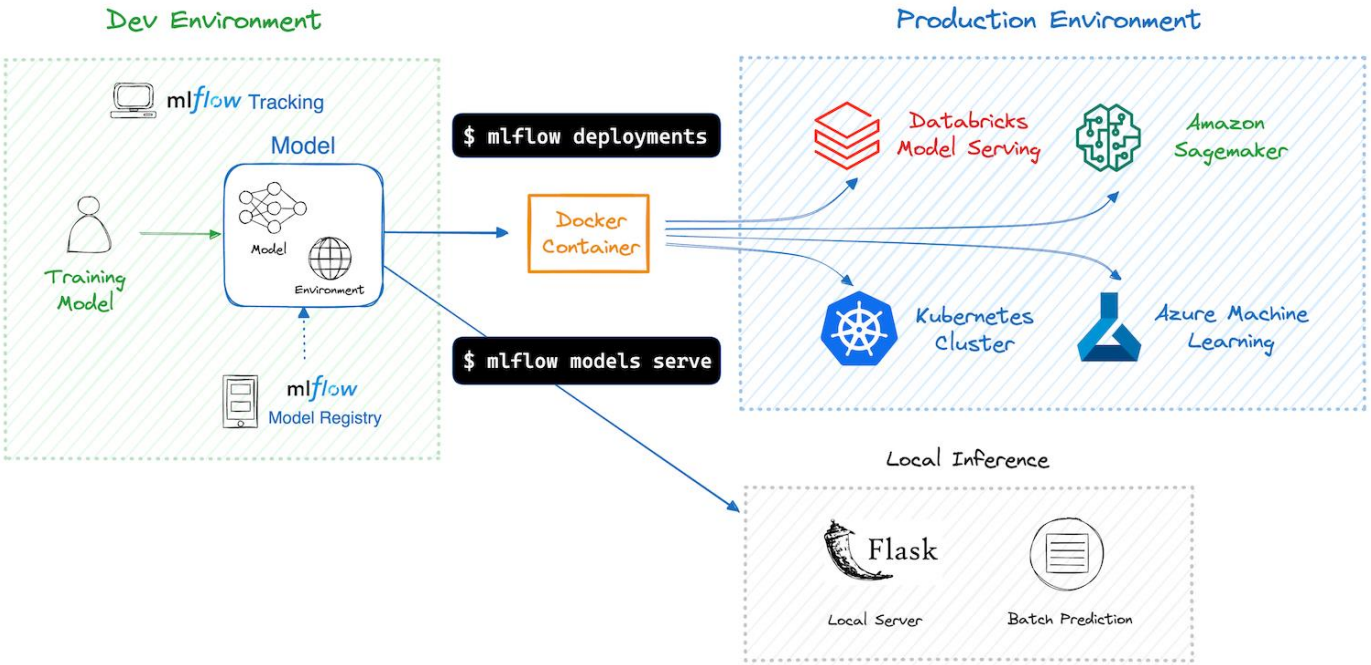
# FITUR UTAMA

## DEPLOYMENT

**MLflow Deployment** mendukung berbagai target deployment termasuk REST API, platform cloud, dan perangkat edge.

Manfaat Utama:

- **Beragam Target:** Deploy ke server lokal, platform cloud, atau lingkungan yang tercontainerisasi
- **Model Serving:** REST API bawaan dengan validasi input otomatis
- **Inference Batch:** Mendukung batch scoring dan prediksi offline
- **Siap Produksi:** Opsi deployment yang skalabel untuk kebutuhan enterprise



# FITUR UTAMA

## ML LIBRARY INTEGRATIONS

Scikit-learn

XGBoost

TensorFlow

PyTorch

Keras

Spark MLlib

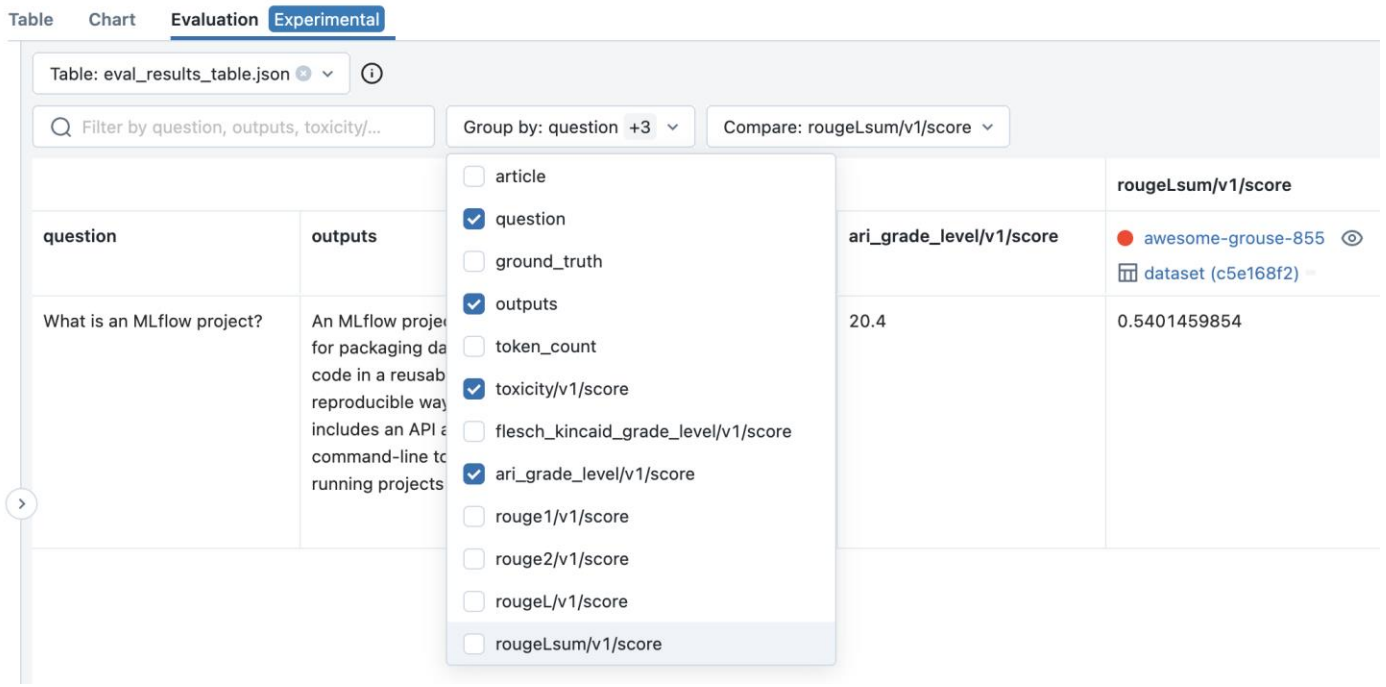
# FITUR UTAMA

## MODEL EVALUATION

MLflow Evaluation menyediakan alat validasi model yang komprehensif, perhitungan metrik otomatis, dan kemampuan untuk membandingkan model.

Manfaat Utama:

- Metrik Otomatis: Metrik evaluasi bawaan untuk classification, regression, dan lainnya
- Evaluator Kustom: Membuat fungsi evaluasi kustom untuk metrik domain-specific
- Perbandingan Model: Membandingkan berbagai model dan versinya side-by-side
- Dataset Validasi: Melacak dataset evaluasi dan memastikan hasil yang dapat direproduksi



# TUTORIAL

Kode lengkap dapat dilihat pada repositori GitHub berikut:
https://github.com/novelxv/MLflow-exploration

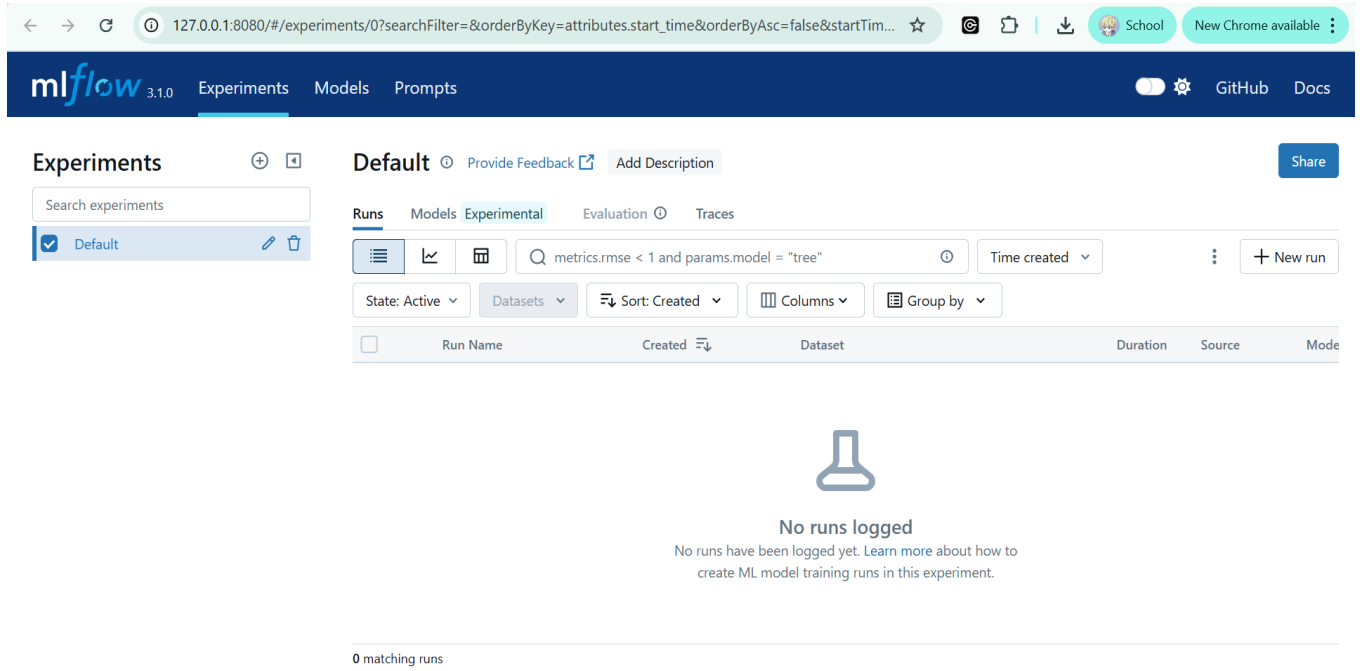
# SETUP MLFLOW

1. Install MLflow

```
pip install mlflow
```

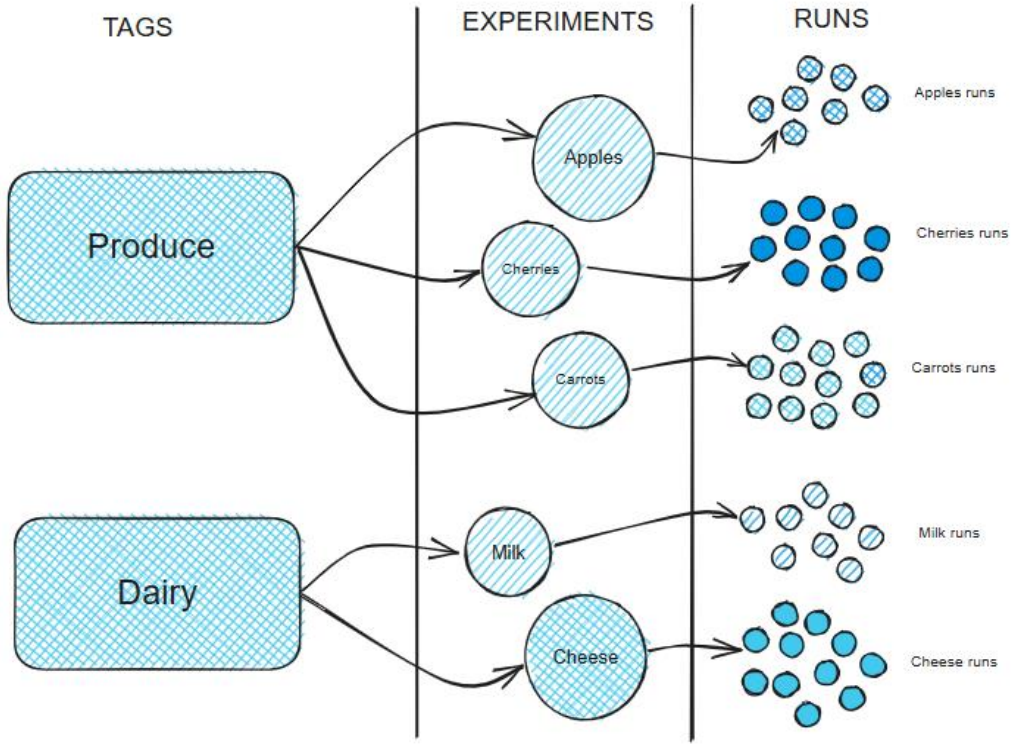
2. Launch Tracking Server

```
mlflow server --host 127.0.0.1 --port 8080
```



# EXPERIMENT

Dalam MLflow, experiment adalah wadah logis untuk mengelompokkan dan menyimpan berbagai run eksperimen machine learning yang memiliki tujuan atau konteks yang sama, seperti pengujian model untuk satu project atau dataset tertentu.




Effective grouping of modeling runs for a large project

# CREATE EXPERIMENT

1. Cantumkan deskripsi dan tag

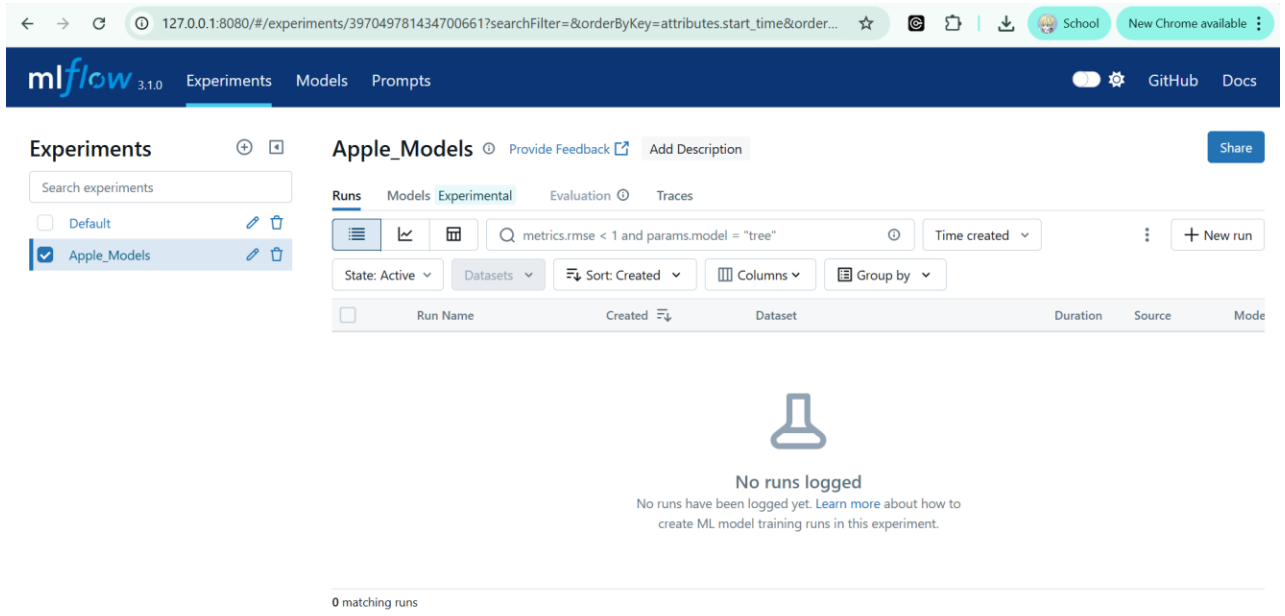
```
experiment_description = (
    "This is the grocery forecasting project. "
    "This experiment contains the produce models for apples."
)
```

```
experiment_tags = {
    "project_name": "grocery-forecasting",
    "store_dept": "produce",
    "team": "stores-ml",
    "project_quarter": "Q3-2023",
    "mlflow.note.content": experiment_description,
}
```

# CREATE EXPERIMENT

2. Create Experiment

```
produce_apples_experiment =
client.create_experiment(
    name="Apple_Models", tags=experiment_tags
)
```



# RUN EXPERIMENT

1. Definisikan Run

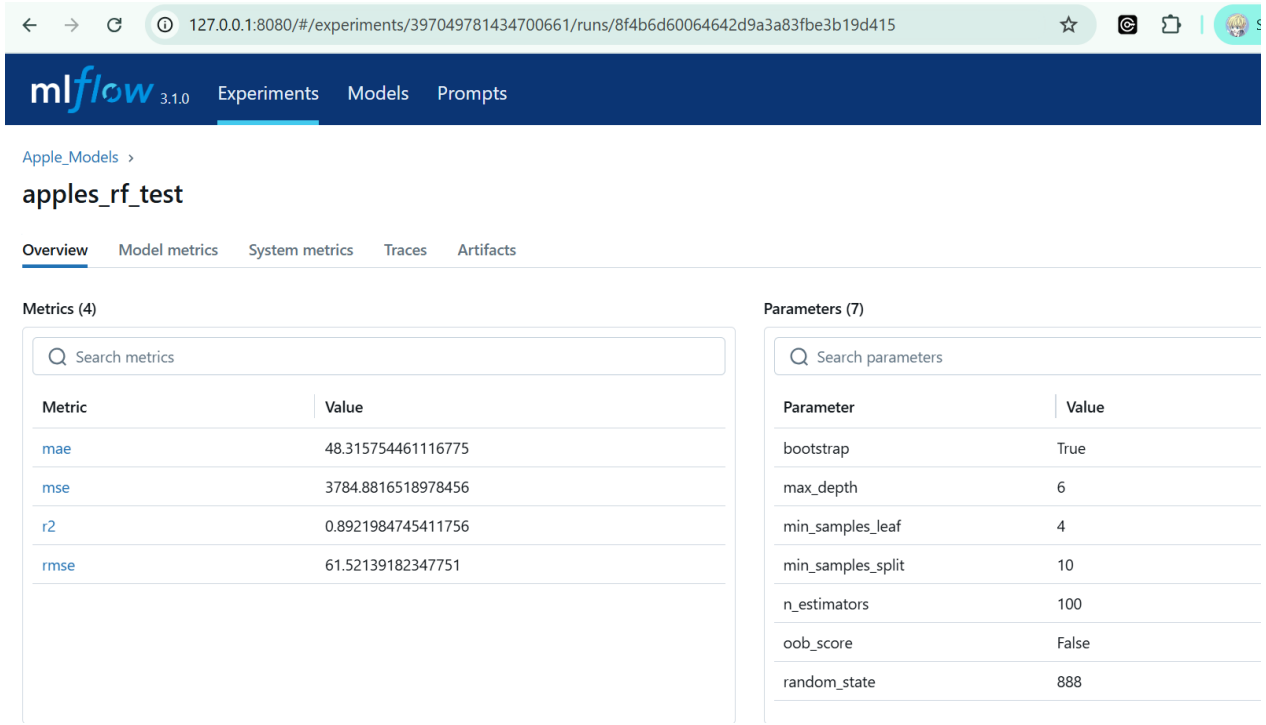
```
import mlflow

mlflow.set_tracking_uri("http://127.0.0.1:8080")

apple_experiment =
mlflow.set_experiment("Apple_Models")

run_name = "apples_rf_test"

artifact_path = "rf_apples"
```

2. Training Model

# RUN EXPERIMENT

3. Inisiasi Run dan Logging

```
with mlflow.start_run(run_name=run_name) as run:
    mlflow.log_params(params)
    mlflow.log_metrics(metrics)
    mlflow.sklearn.log_model(sk_model=rf, input_example=X_val, name=artifact_path)
    run_id = run.info.run_id
    print(">>>>> Run ID:", run_id)
```



```python
# Split the data into features and target and drop irrelevant date field and target field
X = data.drop(columns=["date", "demand"])
y = data["demand"]

# Split the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)

params = {
    "n_estimators": 100,
    "max_depth": 6,
    "min_samples_split": 10,
    "min_samples_leaf": 4,
    "bootstrap": True,
    "oob_score": False,
    "random_state": 888,
}

# Train the RandomForestRegressor
rf = RandomForestRegressor(**params)

# Fit the model on the training data
rf.fit(X_train, y_train)

# Predict on the validation set
y_pred = rf.predict(X_val)

# Calculate error metrics
mae = mean_absolute_error(y_val, y_pred)
mse = mean_squared_error(y_val, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_val, y_pred)

# Assemble the metrics we're going to write into a collection
metrics = {"mae": mae, "mse": mse, "rmse": rmse, "r2": r2}

# Initiate the MLflow run context
with mlflow.start_run(run_name=run_name) as run:

    # Log the parameters used for the model fit
    mlflow.log_params(params)

    # Log the error metrics that were calculated during validation
    mlflow.log_metrics(metrics)

    # Log an instance of the trained model for later use
    mlflow.sklearn.log_model(sk_model=rf, input_example=X_val, artifact_path=artifact_path)
```

We define parameters for training

We generate predictions

We log our trained model

We calculate error metrics based on predictions

We construct a collection of our metrics

We log the parameters used to train the model

We create an MLflow run

We log our metrics

# DEPLOY KE FLASK

Kode lengkap dapat dilihat pada repositori GitHub berikut:
https://github.com/novelxv/MLflow-exploration

# DEPLOY KE FLASK

1. Catat "run_id"
2. Install Flask

```
pip install flask
```

3. Buat app.py
4. Definisikan RUN_ID dan ARTIFACT_PATH

```
RUN_ID = "790e4965409d4ae588f296033d856875"
ARTIFACT_PATH = "rf_apples"
```

# DEPLOY KE FLASK

5. Load Model

```
model_uri = f"runs:/{RUN_ID}/{ARTIFACT_PATH}"
model = mlflow.sklearn.load_model(model_uri)
```

6. Buat Endpoint API

```
app = Flask(__name__)
@app.route("/predict", methods=["POST"])
def predict():
    data = request.get_json()
    df = pd.DataFrame(data)
    preds = model.predict(df)
    return jsonify({"predictions": preds.tolist()})
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)
```

# DEPLOY KE FLASK

7. Jalankan Aplikasi

```
python app.py
```

8. Testing

```
curl -X POST http://localhost:5000/predict \
  -H "Content-Type: application/json" \
  -d '{
    "average_temperature": [25.5, 30.2],
    "rainfall": [2.1, 0.5],
    "weekend": [1, 0],
    "holiday": [0, 0],
    "price_per_kg": [1.5, 2.0],
    "promo": [1, 0],
    "previous_days_demand": [1200, 1100]
  }'
```

Status Code: 200
Response: {'predictions': [1496.0293853175344, 947.1538219929763, 1473.6694622026 23]}

# REFERENSI

https://mlflow.org/docs/latest/ml/

# TERIMA KASIH