

MLFLOW

A TOOL FOR MANAGING THE MACHINE LEARNING LIFECYCLE

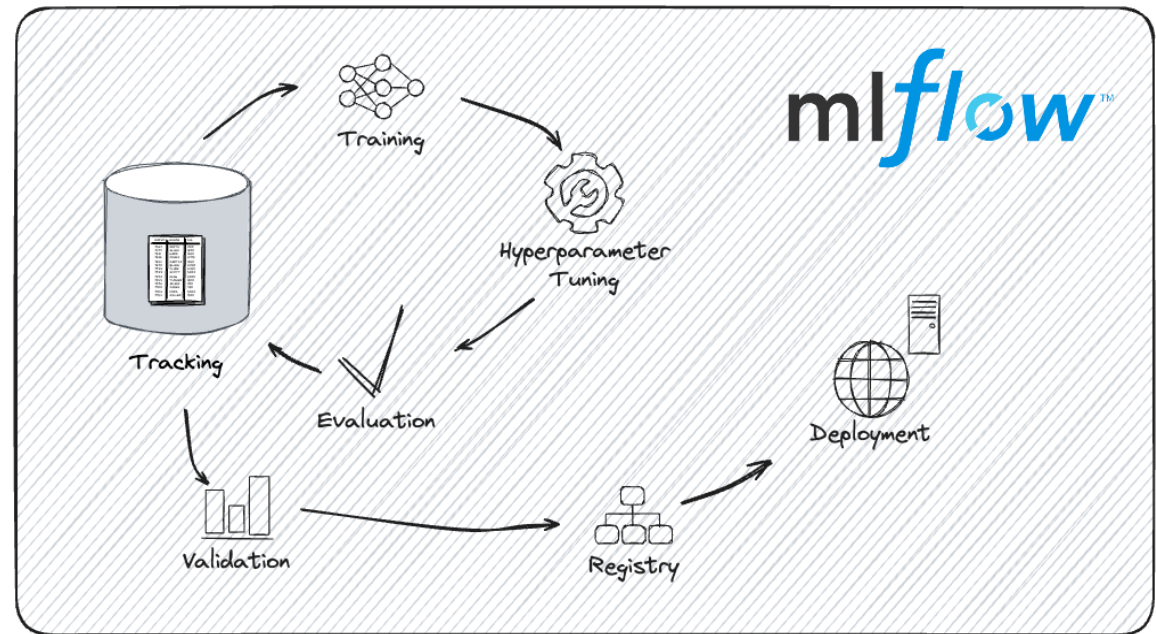
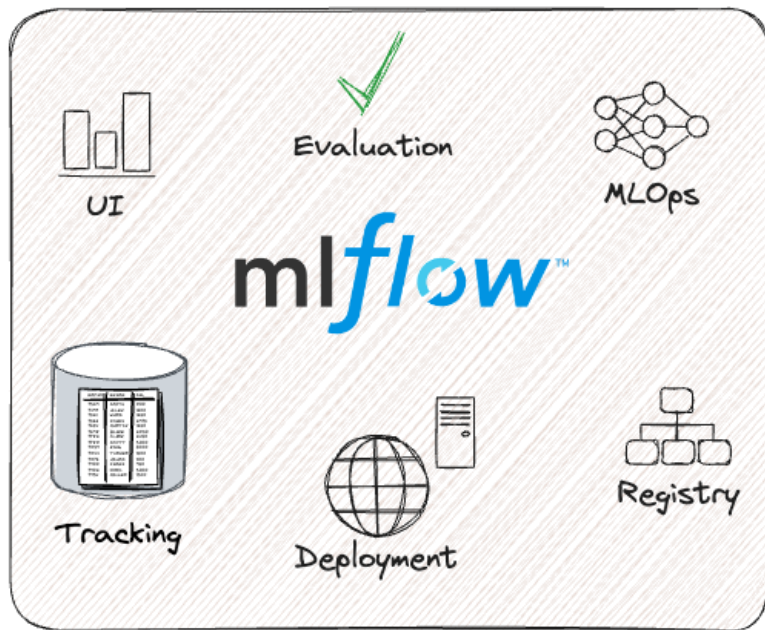
By: Novelya Putri Ramadhani

A decorative horizontal bar at the bottom of the slide with a gradient from red on the left to purple on the right.

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



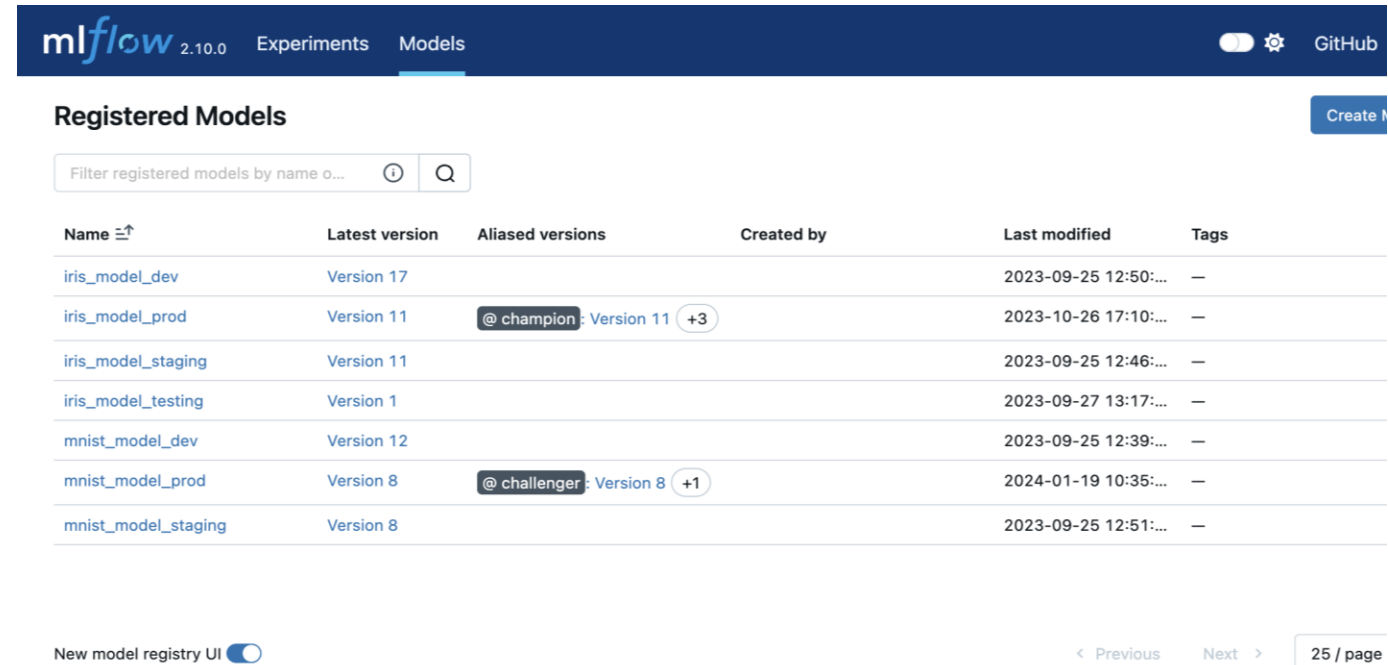
FITUR UTAMA

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



The screenshot displays the MLflow Model Registry interface. At the top, there's a navigation bar with 'mlflow 2.10.0', 'Experiments', and 'Models' tabs. Below this, the 'Registered Models' section is visible, featuring a search bar and a table of models. The table has columns for Name, Latest version, Aliased versions, Created by, Last modified, and Tags. Two models are highlighted with alias badges: '@ champion' for 'iris_model_prod' and '@ challenger' for 'mnist_model_prod'. At the bottom, there's a toggle for 'New model registry UI' and pagination controls showing '25 / page'.

Name	Latest version	Aliased versions	Created by	Last modified	Tags
iris_model_dev	Version 17			2023-09-25 12:50:...	—
iris_model_prod	Version 11	@ champion : Version 11 +3		2023-10-26 17:10:...	—
iris_model_staging	Version 11			2023-09-25 12:46:...	—
iris_model_testing	Version 1			2023-09-27 13:17:...	—
mnist_model_dev	Version 12			2023-09-25 12:39:...	—
mnist_model_prod	Version 8	@ challenger : Version 8 +1		2024-01-19 10:35:...	—
mnist_model_staging	Version 8			2023-09-25 12:51:...	—

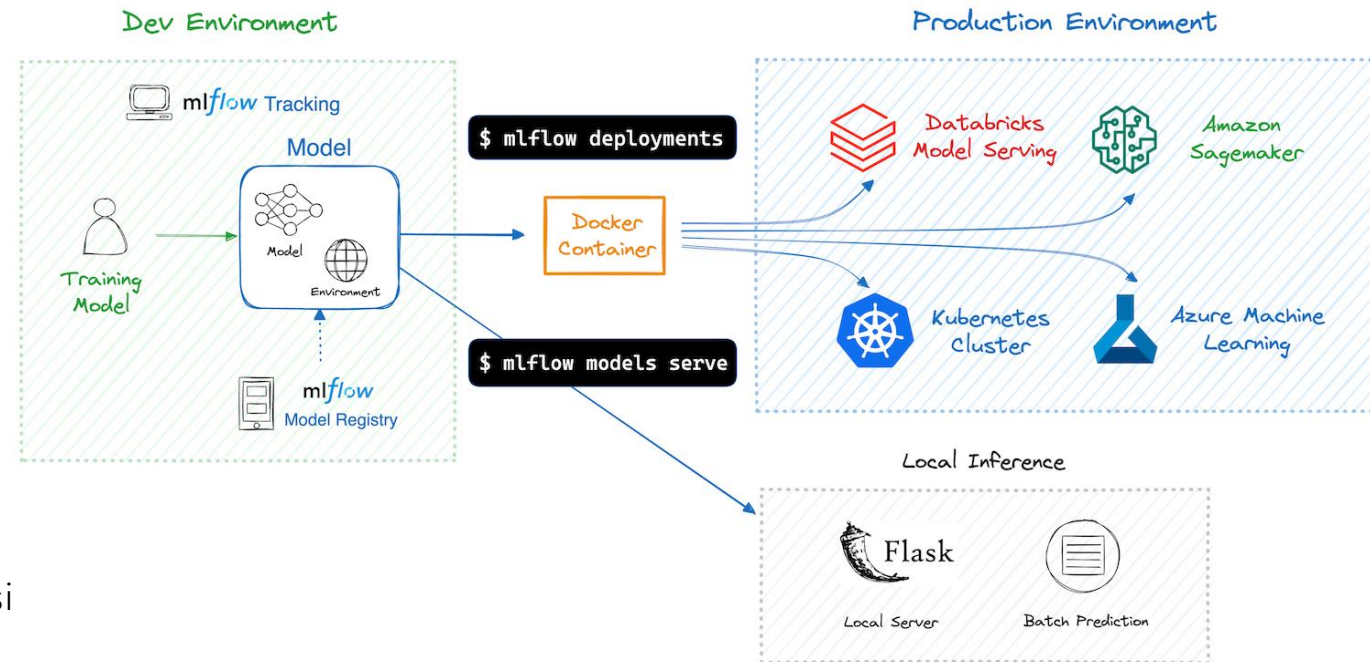
FITUR UTAMA

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

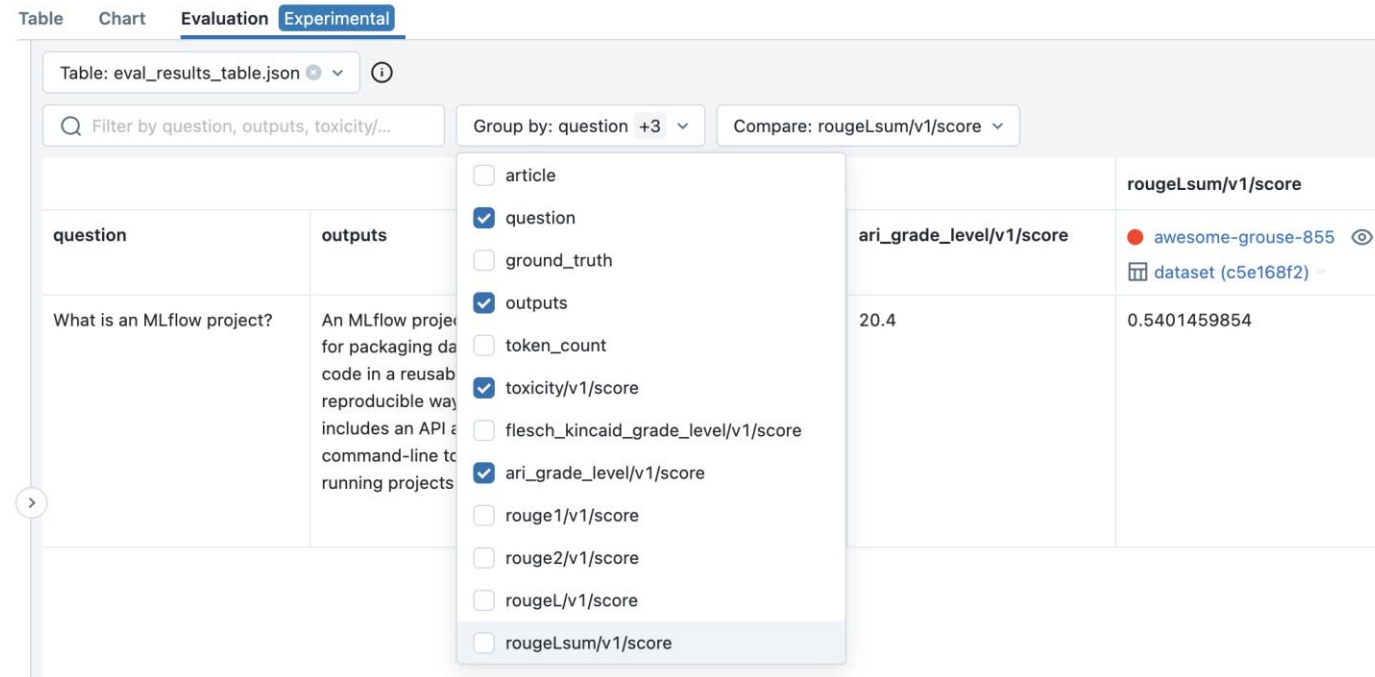
FITUR UTAMA

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



The screenshot shows the MLflow Evaluation Experimental tab. At the top, there are tabs for 'Table', 'Chart', 'Evaluation', and 'Experimental'. Below these, there's a table titled 'eval_results_table.json'. The table has columns for 'question', 'outputs', and 'rougeLsum/v1/score'. A dropdown menu is open, showing a list of metrics to be evaluated: 'article', 'question', 'ground_truth', 'outputs', 'token_count', 'toxicity/v1/score', 'flesch_kincaid_grade_level/v1/score', 'ari_grade_level/v1/score', 'rouge1/v1/score', 'rouge2/v1/score', 'rougeL/v1/score', and 'rougeLsum/v1/score'. The 'question', 'outputs', 'toxicity/v1/score', and 'ari_grade_level/v1/score' metrics are selected. The table shows a single row of results for the question 'What is an MLflow project?' with an output of 'An MLflow project for packaging data in a reusable, reproducible way, includes an API and command-line tools for running projects.' The 'ari_grade_level/v1/score' is 20.4, and the 'rougeLsum/v1/score' is 0.5401459854. The table is part of a dataset named 'awesome-grouse-855'.

question	outputs	rougeLsum/v1/score
What is an MLflow project?	An MLflow project for packaging data in a reusable, reproducible way, includes an API and command-line tools for running projects.	0.5401459854

TUTORIAL

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

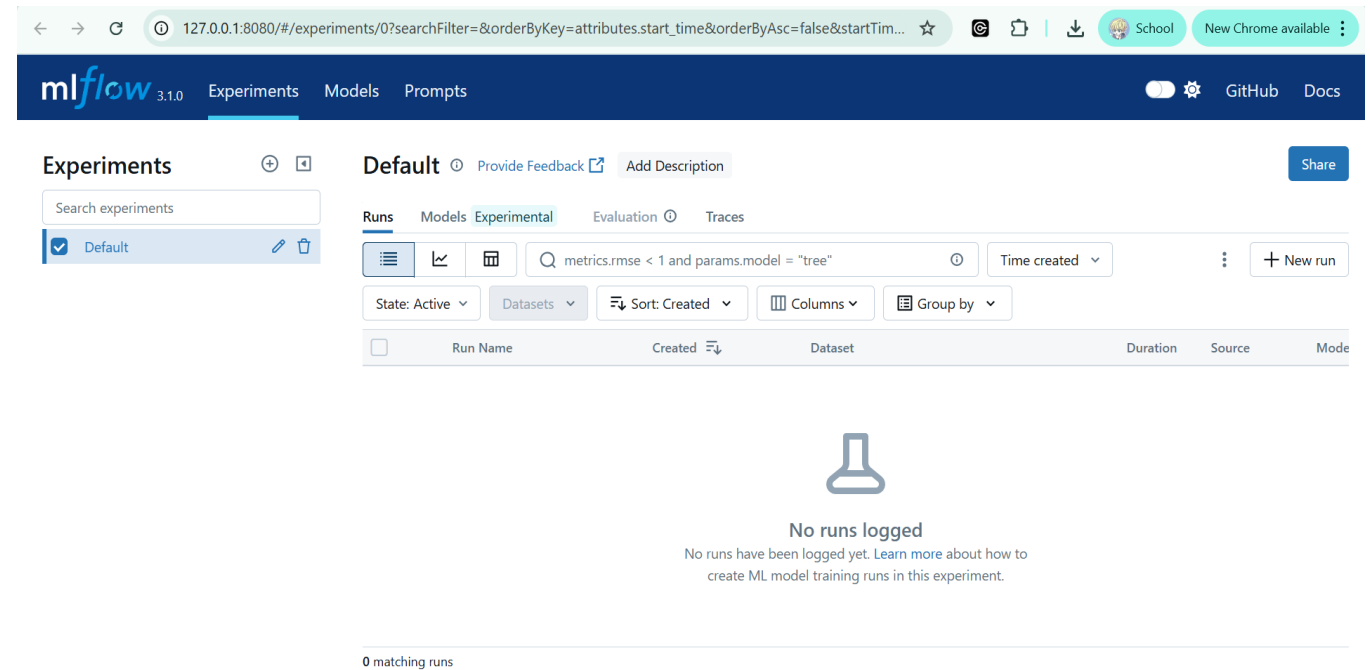
SETUP MLFLOW

1. Install MLflow

```
pip install mlflow
```

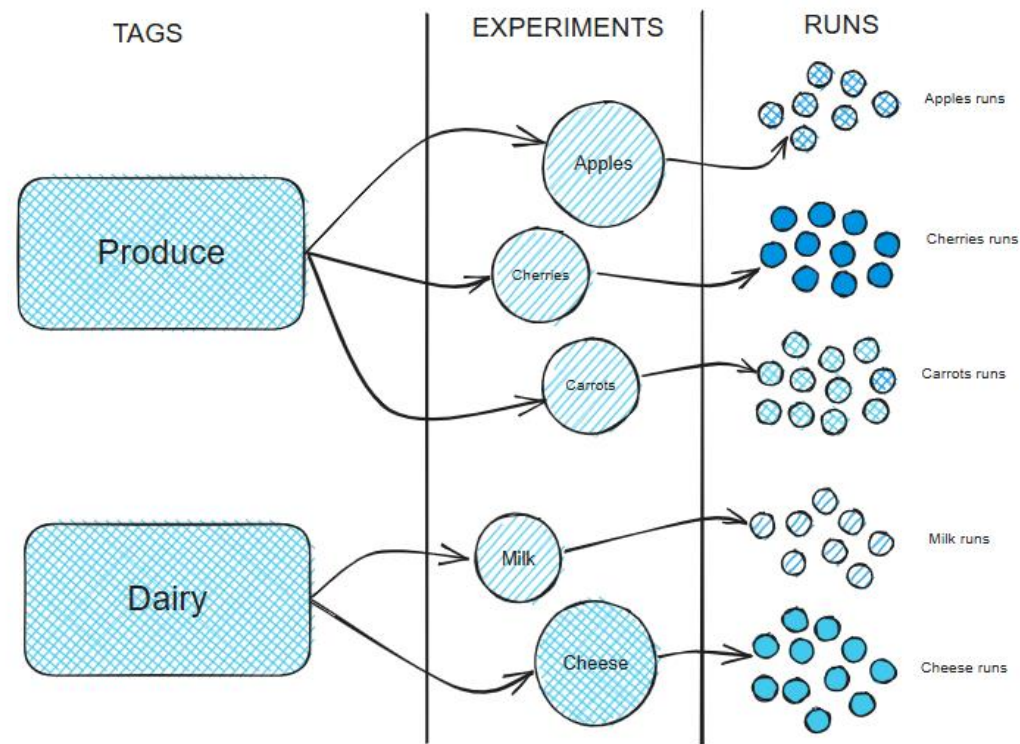
2. Launch Tracking Server

```
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

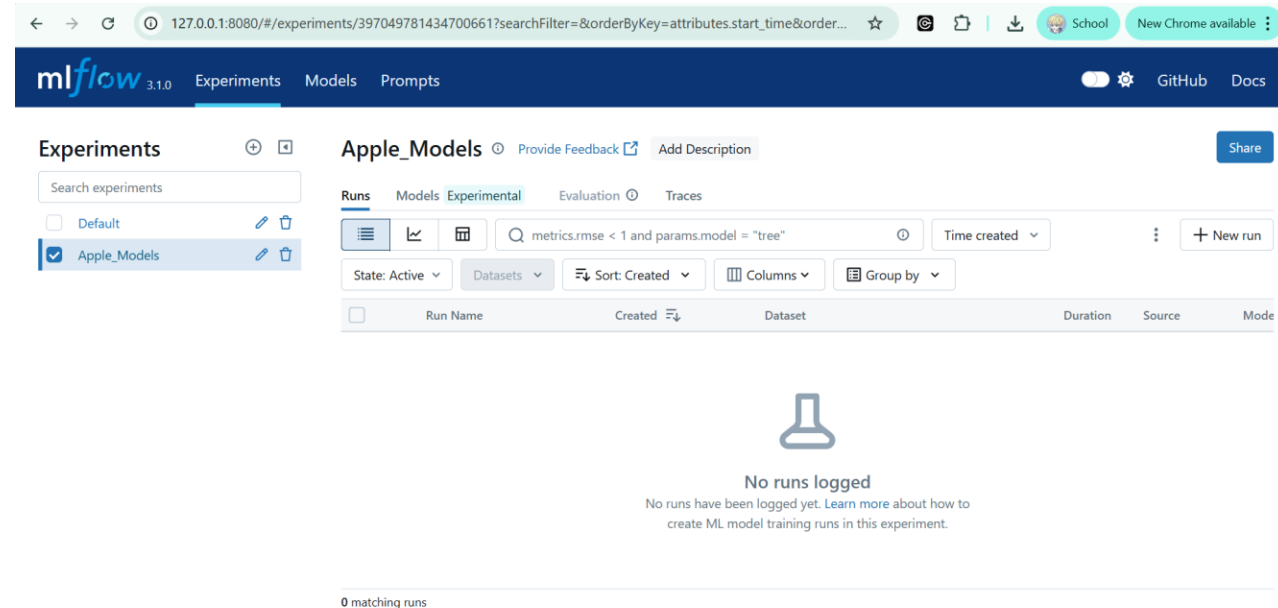
```
experiment_description = (  
    "This is the grocery forecasting project. "  
    "This experiment contains the produce models for apples."  
)
```

```
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  
)
```



The screenshot displays the MLflow web interface for an experiment named 'Apple_Models'. The interface includes a sidebar with a search bar and a list of experiments, where 'Apple_Models' is selected. The main panel shows the experiment details, including tabs for 'Runs', 'Models', 'Experimental', 'Evaluation', and 'Traces'. A search filter is applied: 'metrics.rmse < 1 and params.model = "tree"'. The 'Runs' tab is active, but it shows 'No runs logged' with a message: 'No runs have been logged yet. Learn more about how to create ML model training runs in this experiment.' The bottom of the page indicates '0 matching runs'.

RUN EXPERIMENT

1. Definiskan 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"

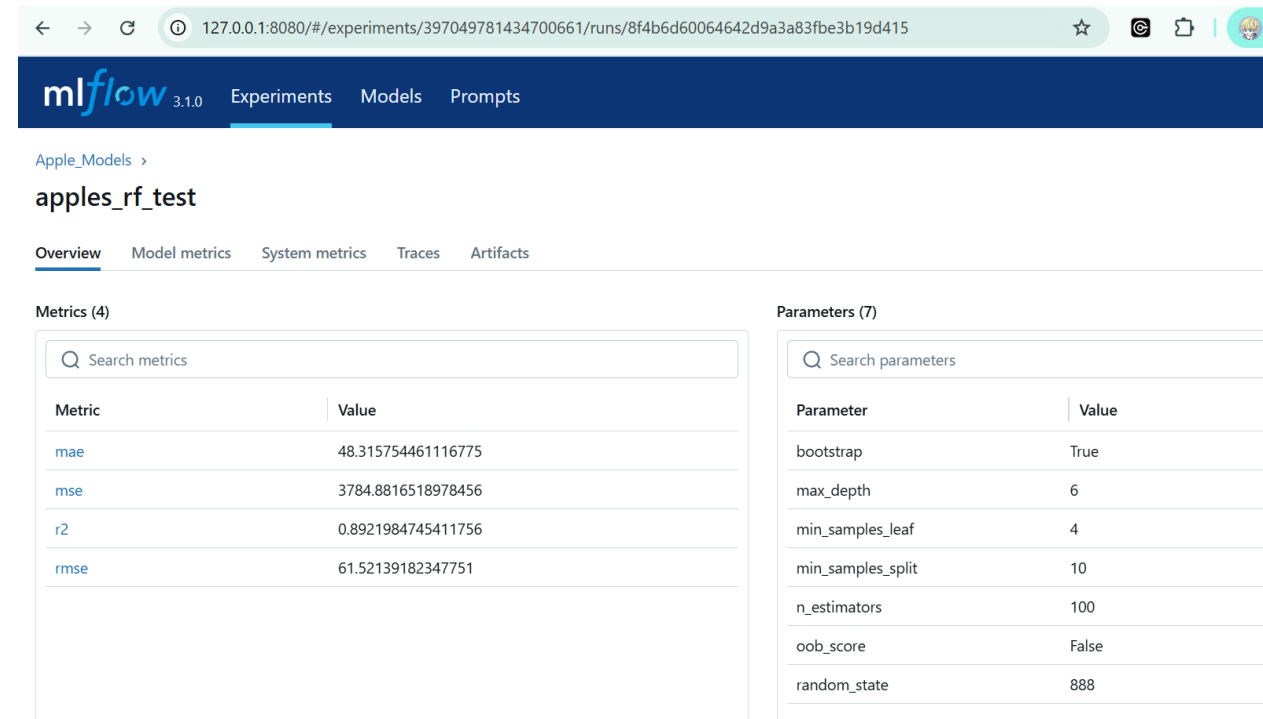
artifact_path = "rf_apples"
```

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)
```



The screenshot displays the MLflow web interface for an experiment named 'apples_rf_test'. The interface is divided into two main sections: 'Metrics (4)' and 'Parameters (7)'. The 'Metrics' section contains a table with four rows, each representing a different metric and its value. The 'Parameters' section contains a table with seven rows, each representing a parameter and its value. The interface also includes a search bar for metrics and parameters, and a navigation bar at the top with links to 'Experiments', 'Models', and 'Prompts'.

Metric	Value
mae	48.315754461116775
mse	3784.8816518978456
r2	0.8921984745411756
rmse	61.52139182347751

Parameter	Value
bootstrap	True
max_depth	6
min_samples_leaf	4
min_samples_split	10
n_estimators	100
oob_score	False
random_state	888

```
# 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,
}
```

We define parameters for training

```
# 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)
```

We generate predictions

We log our trained model

```
# 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)
```

We calculate error metrics based on predictions

We construct a collection of our metrics

```
# 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:
```

We create an MLflow run

We log our metrics

```
# 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 log the parameters used to train the model

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, 1473.669462202623]}
```

REFERENSI

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

TERIMA KASIH