Nama: Ahmad Rizki
NIM: 18221071

Laporan Experiment Tracking

Eksperimen tracking dilakukan dengan menggunakan tiga tools utama yaitu MLFlow, TensorBoard, dan DVC. Model yang digunakan dalam eksperimen ini adalah Random Forest (RF) dan Neural Network (NN).

Informasi Dataset

Nama Dataset: Adult
Link: https://archive.ics.uci.edu/dataset/2/adult

- **Donated on:** 4/30/1996
- **Task:** Predict whether annual income of an individual exceeds $50K/yr based on census data. Also known as the "Census Income" dataset.
- **Dataset Characteristics:** Multivariate
- **Subject Area:** Social Science
- **Associated Tasks:** Classification
- **Feature Type:** Categorical, Integer
- **Instances:** 48,842
- **Features:** 14
- **Missing Values:** Yes

Kolom-kolom dataset:

Variable Name	Role	Type	Description
age	Feature	Integer	Age
workclass	Feature	Categorical	Income type
fnlwgt	Feature	Integer	Final weight
education	Feature	Categorical	Education level
education-num	Feature	Integer	Numeric education level
marital-status	Feature	Categorical	Marital status
occupation	Feature	Categorical	Occupation
relationship	Feature	Categorical	Relationship status
race	Feature	Categorical	Race
sex	Feature	Binary	Gender

capital-gain	Feature	Integer	Capital gain
capital-loss	Feature	Integer	Capital loss
hours-per-week	Feature	Integer	Hours worked per week
native-country	Feature	Categorical	Country of origin
income	Target	Binary	Annual income (>50K, <=50K)

Komponen/Arsitektur Sistem

1. MLFlow:
   - MLFlow Tracking Server
   - MLFlow Client (untuk logging metrics, parameters, model artifacts)
   - Web UI untuk menampilkan eksperimen
2. TensorBoard:
   - File Log TensorFlow (events)
   - TensorBoard Web Server untuk visualisasi real-time
3. DVC:
   - DVC Pipeline (dvc.yaml)
   - Data versioning (file .dvc)
   - Metadata metrics dan plots dalam format CSV dan JSON

Cara Setup Instrumentasi/Pencatatan Eksperimen

1. MLFlow:
   - Instalasi menggunakan pip: `pip install mlflow`
   - Tracking dengan menambahkan kode `mlflow.log_param`, `mlflow.log_metric`, `mlflow.log_artifact`, dan menjalankan server UI dengan `mlflow ui`
2. TensorBoard:
   - Instalasi dengan pip: `pip install tensorboard`
   - Logging metrics TensorFlow menggunakan callback (`tf.keras.callbacks.TensorBoard`)
   - Visualisasi dengan menjalankan `tensorboard --logdir=logs`
3. DVC:
   - Instalasi dengan pip: `pip install dvc`
   - Setup pipeline dalam file `dvc.yaml`
   - Menjalankan seluruh pipeline menggunakan `dvc repro`
   - Menampilkan metrics dengan `dvc metrics show`

Metadata Eksperimen yang Dikumpulkan

1. MLFlow:

    - Parameter model (e.g., learning rate, epochs, batch size, n_estimators, max_depth)
    - Metric evaluasi (accuracy, precision, recall, F1-score, train loss)
    - Model artifact (NN model file `.h5`, RF model file `.pkl`)
2. TensorBoard:
    - Accuracy dan Loss per epoch (NN model)
    - Informasi waktu training per epoch
3. DVC:
    - Metrics akhir eksperimen dalam format JSON dan CSV
    - Detail pipeline eksekusi (stage dependencies, outputs)

Visualisasi/Laporan/Dashboard yang Disediakan

1. MLFlow:
    - Web UI interaktif untuk perbandingan antar eksperimen (tabel parameter dan metrics)
    - Grafik sederhana seperti bar chart untuk membandingkan metrics antar eksperimen
2. TensorBoard:
    - Dashboard web real-time untuk visualisasi detail metrics per epoch
    - Grafik tren akurasi dan loss yang interaktif
3. DVC:
    - Plot berbentuk grafik bar yang menunjukkan perbandingan akurasi antar model (RF dan NN)
    - Tabel metrics akhir dalam format tabel sederhana

Evaluasi Umum Perbandingan Ketiga Tool

Kriteria	MLFlow	TensorBoard	DVC
Kemudahan Instalasi	Mudah	Mudah	Mudah
Kemampuan Tracking	Param, Metric, Model Artifact	Metric (loss, accuracy) per epoch	Pipeline stage, final metric
Integrasi dengan code	Mudah via library Python	Mudah via TensorFlow callbacks	Harus setup manual pipeline
Visualisasi	Web UI yang interaktif	Dashboard interaktif, real-time	Plot statis, visualisasi sederhana
Penyimpanan Artefak	Lengkap (model, param, metric)	Terbatas (hanya metrics dan logs)	Terbatas (metric, pipeline info)
Cocok untuk	Eksperimen skala kecil - besar	Fokus pada eksperimen deep learning	Eksperimen berbasis pipeline data

Kesimpulan

- **MLFlow** adalah solusi yang komprehensif untuk tracking semua aspek eksperimen termasuk model artifact.
- **TensorBoard** sangat efektif khususnya untuk visualisasi real-time eksperimen deep learning.
- **DVC** unggul dalam manajemen pipeline, reproduksibilitas eksperimen, dan cocok untuk proyek data yang membutuhkan versioning pipeline yang jelas.

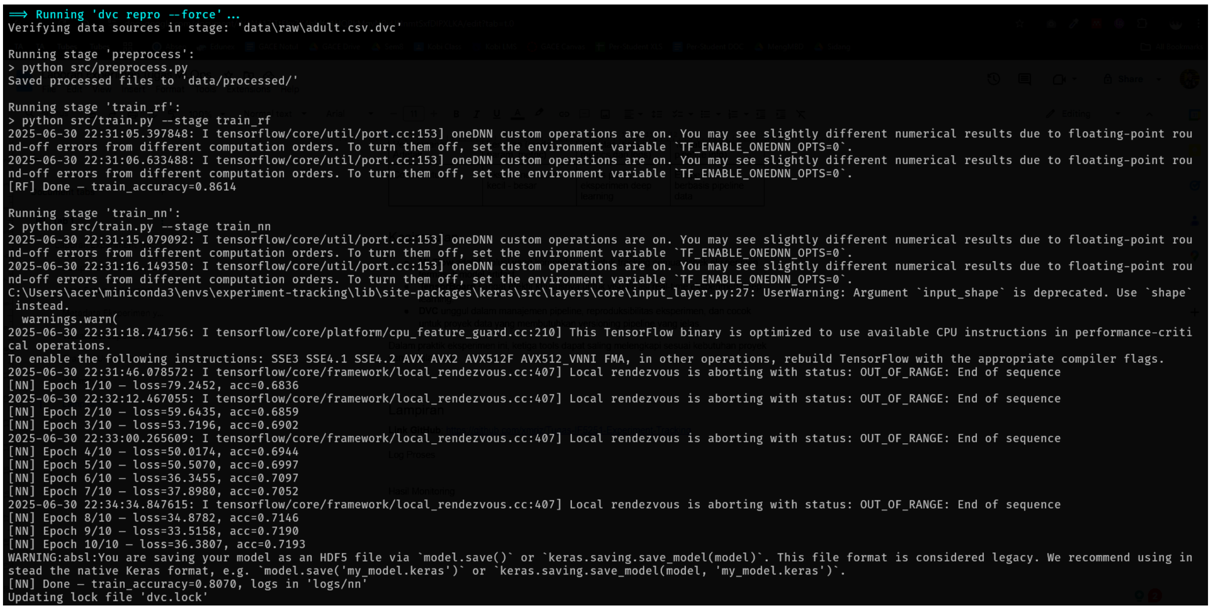
Dalam praktik eksperimen ini, ketiga tools dapat saling melengkapi sesuai kebutuhan proyek yang dijalankan.

Lampiran

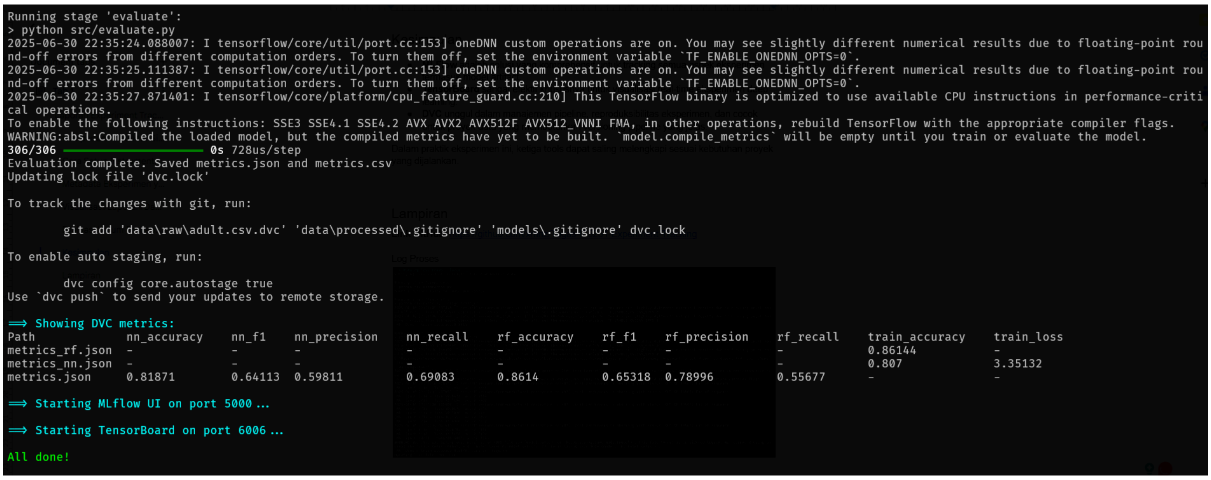
Link GitHub:

https://github.com/xmriz/Tugas-IF5251-Experiment-Tracking

Log Proses:



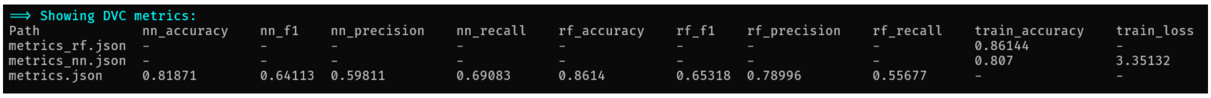
```
==> Running 'dvc repro --force' ...
Verifying data sources in stage: 'data\raw\adult.csv.dvc'

Running stage 'preprocess':
> python src/preprocess.py
Saved processed files to 'data/processed/'

Running stage 'train_rf':
> python src/train.py --stage train_rf
2025-06-30 22:31:05.397848: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point rou
nd-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-06-30 22:31:06.633488: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point rou
nd-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
[RF] Done — train_accuracy=0.8614

Running stage 'train_nn':
> python src/train.py --stage train_nn
2025-06-30 22:31:15.079092: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point rou
nd-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-06-30 22:31:16.149350: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point rou
nd-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
C:\Users\acer\miniconda3\envs\experiment-tracking\lib\site-packages\keras\src\layers\core\input_layer.py:27: UserWarning: Argument `input_shape` is deprecated. Use `shape`
 instead.
  warnings.warn(
2025-06-30 22:31:18.741756: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-criti
cal operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX512F AVX512_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2025-06-30 22:31:46.078572: I tensorflow/core/framework/local_rendezvous.cc:407] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
[NN] Epoch 1/10 — loss=79.2452, acc=0.6836
2025-06-30 22:32:12.467055: I tensorflow/core/framework/local_rendezvous.cc:407] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
[NN] Epoch 2/10 — loss=59.6435, acc=0.6859
[NN] Epoch 3/10 — loss=53.7196, acc=0.6902
2025-06-30 22:33:00.265609: I tensorflow/core/framework/local_rendezvous.cc:407] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
[NN] Epoch 4/10 — loss=50.0174, acc=0.6944
[NN] Epoch 5/10 — loss=50.5070, acc=0.6997
[NN] Epoch 6/10 — loss=36.3455, acc=0.7097
[NN] Epoch 7/10 — loss=37.8980, acc=0.7052
2025-06-30 22:34:34.847615: I tensorflow/core/framework/local_rendezvous.cc:407] Local rendezvous is aborting with status: OUT_OF_RANGE: End of sequence
[NN] Epoch 8/10 — loss=34.8782, acc=0.7146
[NN] Epoch 9/10 — loss=33.5158, acc=0.7190
[NN] Epoch 10/10 — loss=36.3807, acc=0.7193
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using in
stead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.
[NN] Done — train_accuracy=0.8070, logs in 'logs/nn'
Updating lock file 'dvc.lock'
```



```
Running stage 'evaluate':
> python src/evaluate.py
2025-06-30 22:35:24.088007: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point rou
nd-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-06-30 22:35:25.111387: I tensorflow/core/util/port.cc:153] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point rou
nd-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2025-06-30 22:35:27.871401: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance-criti
cal operations.
To enable the following instructions: SSE3 SSE4.1 SSE4.2 AVX AVX2 AVX512F AVX512_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you train or evaluate the model.
306/306 ━━━━━━━━━━━━━━━━━━━━ 0s 728us/step
Evaluation complete. Saved metrics.json and metrics.csv
Updating lock file 'dvc.lock'

To track the changes with git, run:

        git add 'data\raw\adult.csv.dvc' 'data\processed\.gitignore' 'models\.gitignore' dvc.lock

To enable auto staging, run:

        dvc config core.autostage true
Use `dvc push` to send your updates to remote storage.

==> Showing DVC metrics:
Path             nn_accuracy    nn_f1    nn_precision    nn_recall    rf_accuracy    rf_f1    rf_precision    rf_recall    train_accuracy    train_loss
metrics_rf.json  -              -        -               -            -              -        -               -            0.86144           -
metrics_nn.json  -              -        -               -            -              -        -               -            0.807             3.35132
metrics.json     0.81871        0.64113  0.59811         0.69083      0.8614         0.65318  0.78996         0.55677      -                 -

==> Starting MLflow UI on port 5000 ...

==> Starting TensorBoard on port 6006 ...

All done!
```

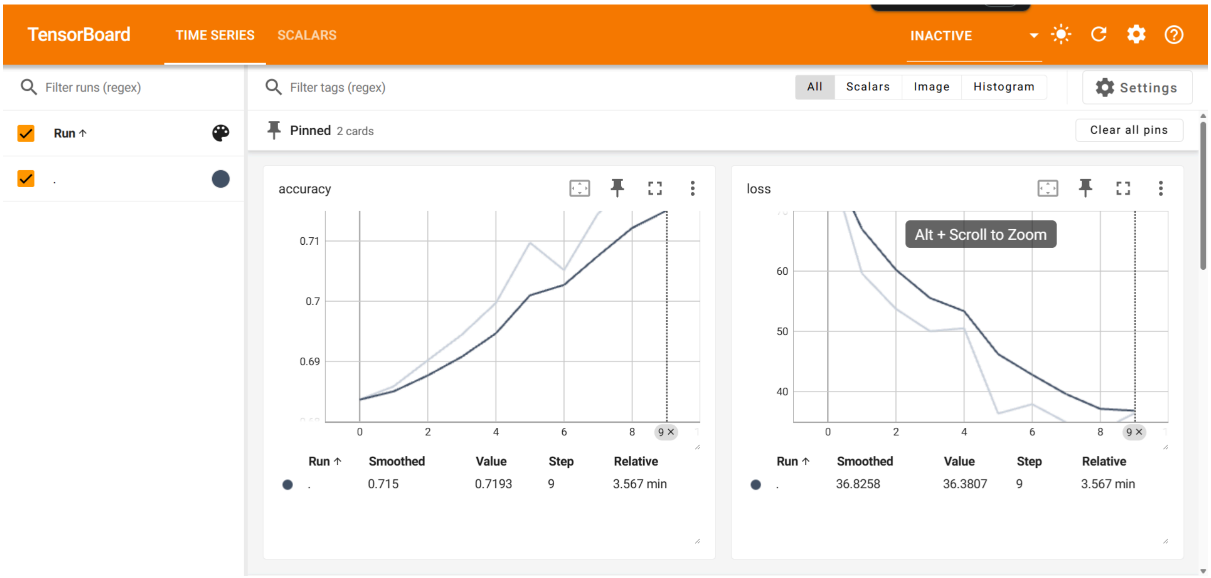
Hasil Monitoring:

DVC



```
⟹ Showing DVC metrics:
Path               nn_accuracy    nn_f1    nn_precision    nn_recall    rf_accuracy    rf_f1    rf_precision    rf_recall    train_accuracy    train_loss
metrics_rf.json    -              -        -               -            -              -        -               -            0.86144           -
metrics_nn.json    -              -        -               -            -              -        -               -            0.807             3.35132
metrics.json       0.81871        0.64113  0.59811         0.69083      0.8614         0.65318  0.78996         0.55677      -                 -
```

TensorBoard



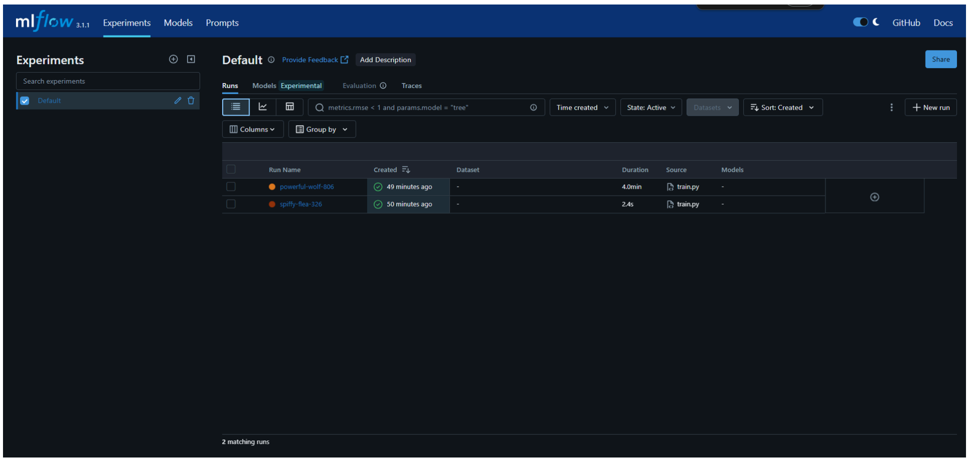
accuracy

Run ↑	Smoothed	Value	Step	Relative
● .	0.715	0.7193	9	3.567 min

loss

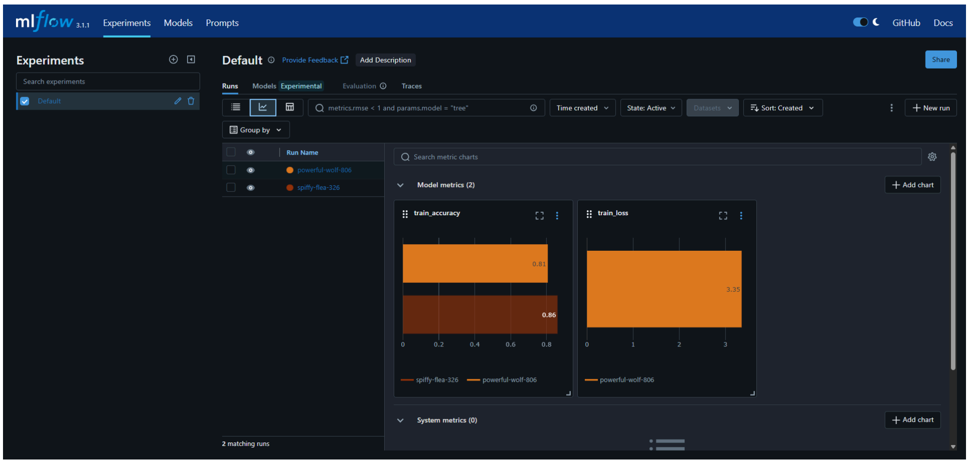
Run ↑	Smoothed	Value	Step	Relative
● .	36.8258	36.3807	9	3.567 min

MLFlow



	Run Name	Created	Dataset	Duration	Source	Models
□	● powerful-wolf-806	✓ 49 minutes ago	-	4.0min	train.py	-
□	● spiffy-flea-326	✓ 50 minutes ago	-	2.4s	train.py	-

2 matching runs


mlflow 3.1.1
Experiments
Models
Prompts
GitHub
Docs
Experiments
Search experiments
Default
Default
Provide Feedback
Add Description
Share
Runs
Models Experimental
Evaluation
Traces
metrics.rmse < 1 and params.model = "tree"
Time created
State: Active
Datasets
Sort: Created
New run
Group by
Run Name
powerful-wolf-806
spiffy-flea-326
Search metric charts
Model metrics (2)
Add chart
train_accuracy
0.81
0.86
0
0.2
0.4
0.6
0.8
spiffy-flea-326
powerful-wolf-806
train_loss
3.35
0
1
2
3
powerful-wolf-806
System metrics (0)
Add chart
2 matching runs