Assignment 02: Experiment Tracking and Model Lifecycle Management with MLFlow

AI545 - W26

Ahsanul Kabir

February 17, 2026

1 Part A: Experiment Design and Setup

Problem Definition: The modeling task is a binary classification to detect fraudulent credit card transactions. The goal is to identify the minority class (Fraud = 1) within a highly imbalanced dataset containing European cardholder transactions from September 2013.

Dataset: We are using the Kaggle Credit Card Fraud Detection dataset. It contains 284,807 transactions, where only 492 (0.172%) are fraudulent. The features include 'Time', 'Amount', and 28 principal components.

Hyperparameters: To optimize the Random Forest model, we will tune four specific hyperparameters:

- $n\_estimators$: The number of trees in the forest (controls variance).
- max_depth: The maximum depth of the tree (controls overfitting).
- $min\_samples\_split$: The minimum samples required to split an internal node.
- $min\_samples\_leaf$: The minimum samples required to be at a leaf node (smooths the model).

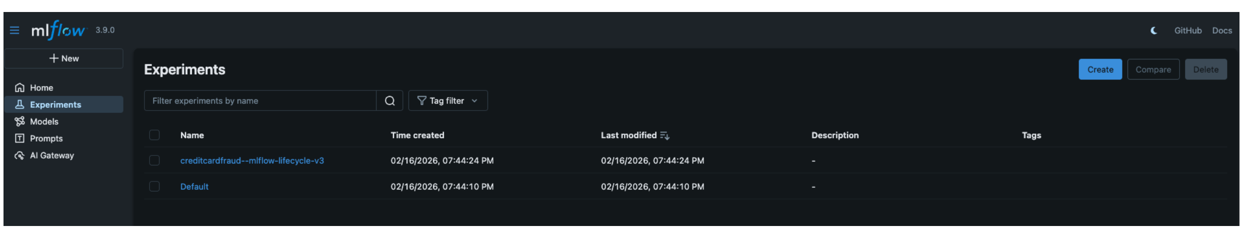

Figure 1: MLFlow Experiments

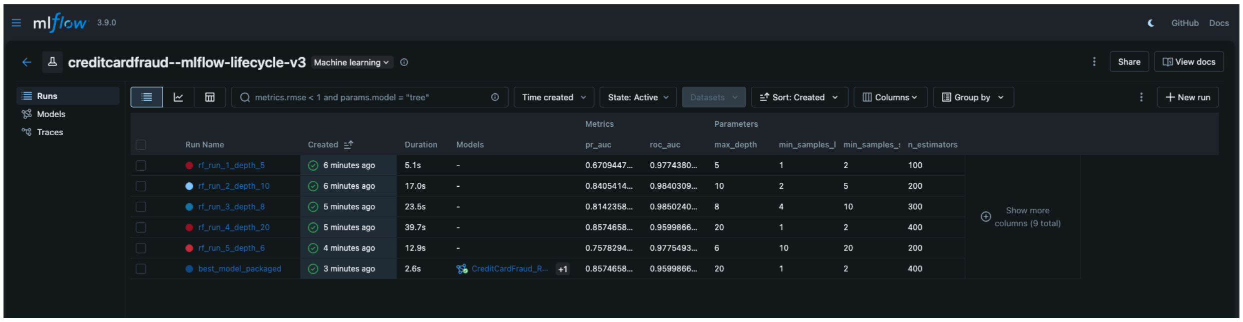

Figure 2: Details of an Experiment

2 Part B: MLflow Tracking

Artifacts: Confusion Matrix and Precision-Recall Curve.

Tags: run_purpose, model_family, and problem_type.

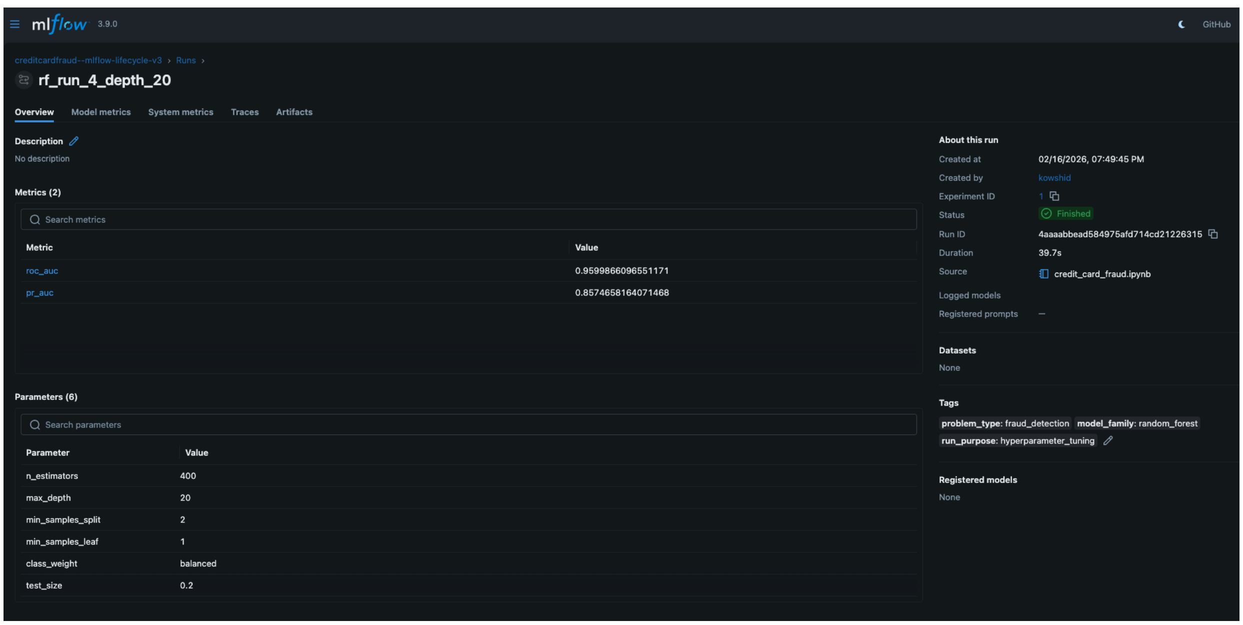

Figure 3: A Run Information

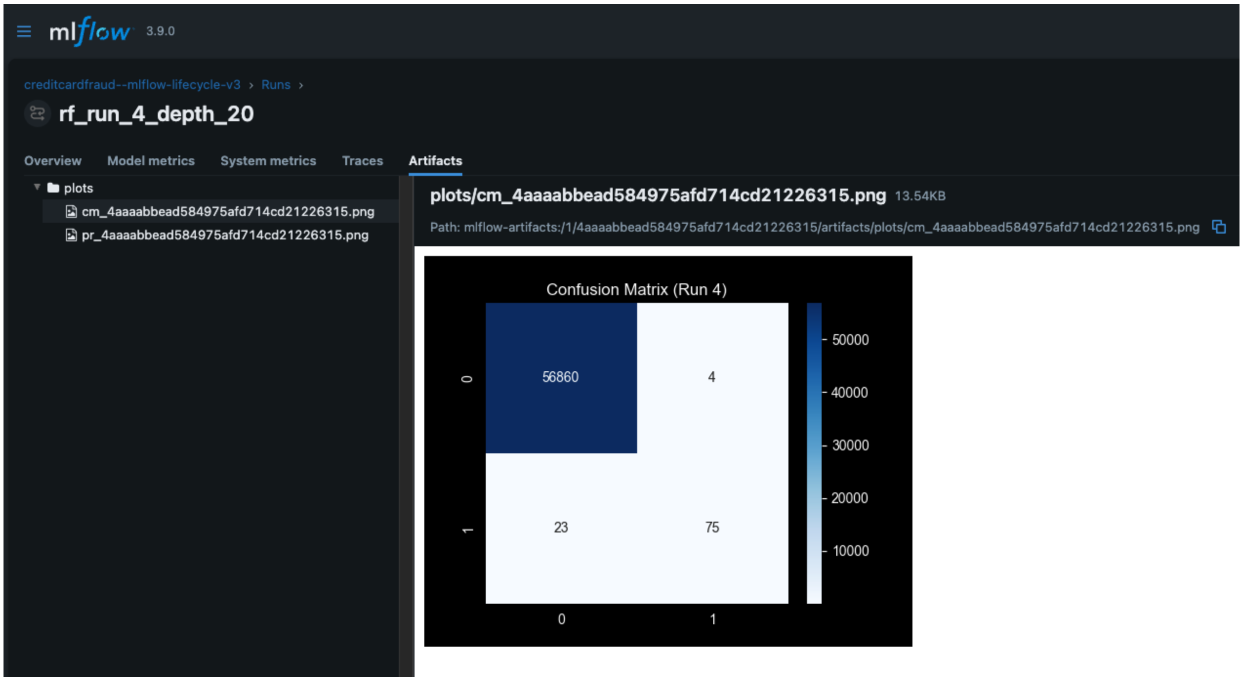

Figure 4: Confusion Matrix of a Run

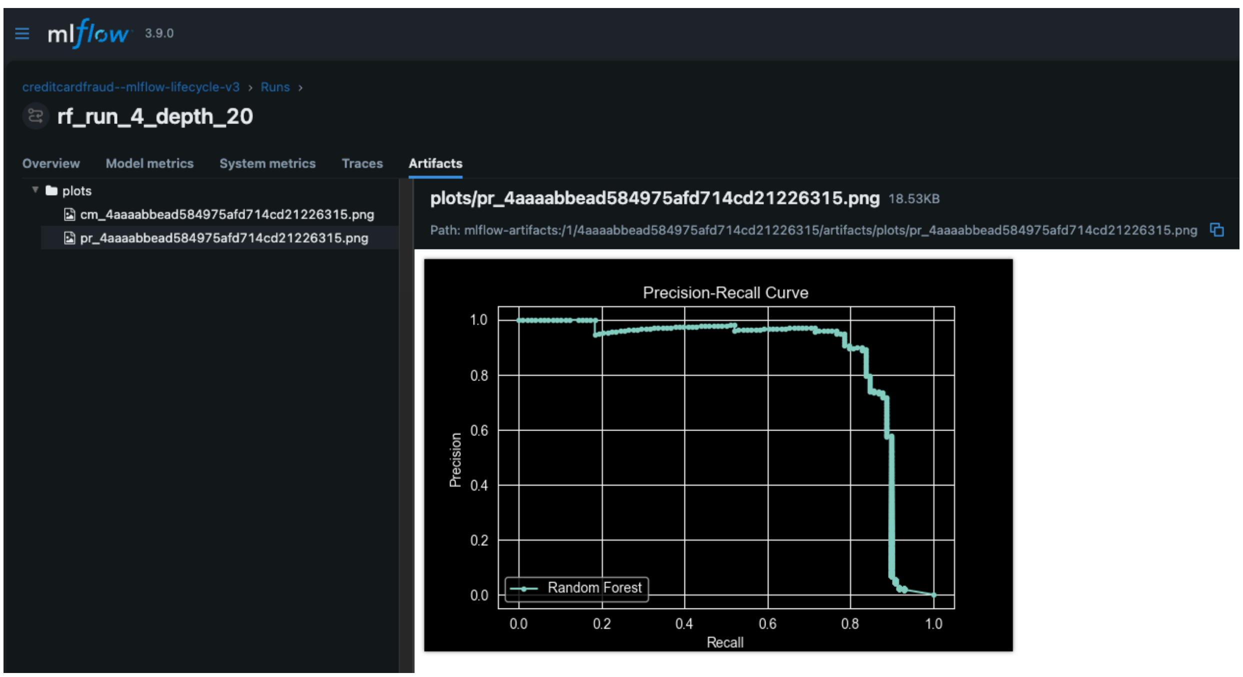

Figure 5: Precision-Recall Curve of a Run

3 Part C: Experiment Comparison and Analysis

Best Model: The best performing model was the run 4 with $n\_estimators = 400$, $max\_depth = 20$, $min\_samples\_leaf = 1$, and $min\_samples\_split = 2$. This is achieved by sorting by pr_auc (Average Precision).

This model achieved the highest Precision-Recall, indicating it maintains high Precision (few false alarms) while capturing a significant portion of fraud cases (high Recall).

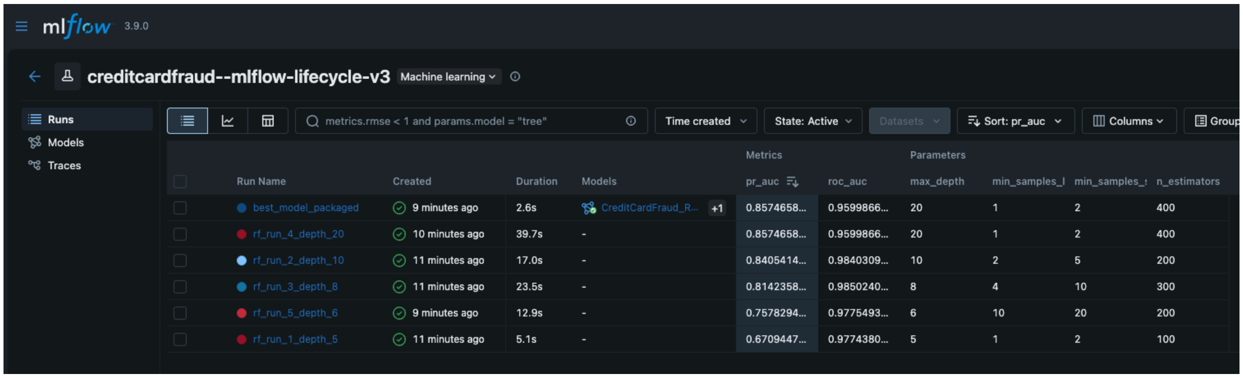

Figure 6: Runs Sorted by Precision-Recall

4 Part D: Model Packaging and Signatures

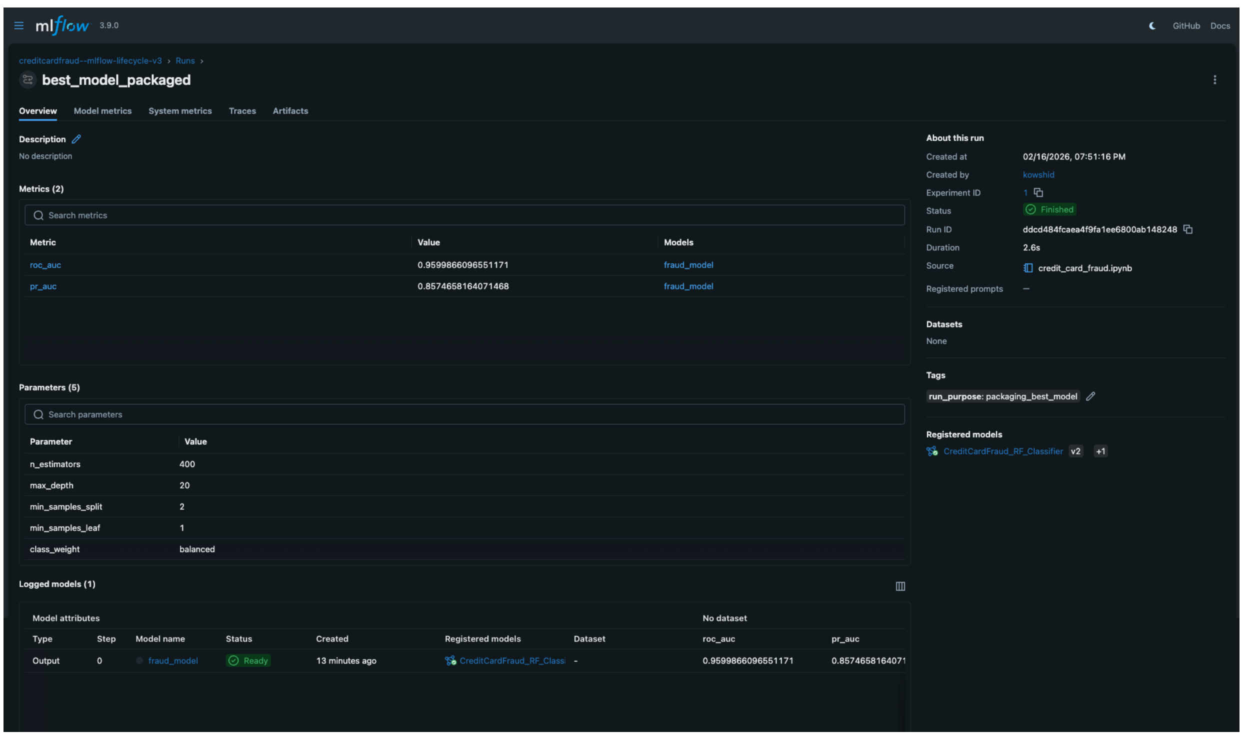

Figure 7: Overview of Best Model

4.1 Model Signature


```yaml
artifact_path: mlflow-artifacts:/1/models/m-c85e74fe975545f29d26173583ce1183/artifacts
flavors:
  python_function:
    env:
      conda: conda.yaml
      virtualenv: python_env.yaml
    loader_module: mlflow.sklearn
    model_path: model.pkl
    predict_fn: predict
    python_version: 3.11.14
  sklearn:
    code: null
    pickled_model: model.pkl
    serialization_format: cloudpickle
    sklearn_version: 1.8.0
    skops_trusted_types: null
is_signature_from_type_hint: false
mlflow_version: 3.9.0
model_id: m-c85e74fe975545f29d26173583ce1183
model_size_bytes: 12958248
model_uuid: m-c85e74fe975545f29d26173583ce1183
prompts: null
run_id: ddcd484fcaea4f9fa1ee6800ab148248
saved_input_example_info:
  artifact_path: input_example.json
  pandas_orient: split
  serving_input_path: serving_input_example.json
  type: dataframe
signature:
  inputs: '[{"type": "double", "name": "Time", "required": true}, {"type": "double",
    "name": "V1", "required": true}, {"type": "double", "name": "V2", "required":
    true}, {"type": "double", "name": "V3", "required": true}, {"type": "double",
    "name": "V4", "required": true}, {"type": "double", "name": "V5", "required":
    true}, {"type": "double", "name": "V6", "required": true}, {"type": "double",
    "name": "V7", "required": true}, {"type": "double", "name": "V8", "required":
    true}, {"type": "double", "name": "V9", "required": true}, {"type": "double",
    "name": "V10", "required": true}, {"type": "double", "name": "V11", "required":
    true}, {"type": "double", "name": "V12", "required": true}, {"type": "double",
    "name": "V13", "required": true}, {"type": "double", "name": "V14", "required":
    true}, {"type": "double", "name": "V15", "required": true}, {"type": "double",
    "name": "V16", "required": true}, {"type": "double", "name": "V17", "required":
    true}, {"type": "double", "name": "V18", "required": true}, {"type": "double",
    "name": "V19", "required": true}, {"type": "double", "name": "V20", "required":
    true}, {"type": "double", "name": "V21", "required": true}, {"type": "double",
    "name": "V22", "required": true}, {"type": "double", "name": "V23", "required":
    true}, {"type": "double", "name": "V24", "required": true}, {"type": "double",
    "name": "V25", "required": true}, {"type": "double", "name": "V26", "required":
    true}, {"type": "double", "name": "V27", "required": true}, {"type": "double",
    "name": "V28", "required": true}, {"type": "double", "name": "Amount", "required":
    true}]'
  outputs: '[{"type": "tensor", "tensor-spec": {"dtype": "float64", "shape": [-1]}}]'
  params: null
type_hint_from_example: false
```

```
utc_time_created: '2026-02-17 00:51:16.334083'
```

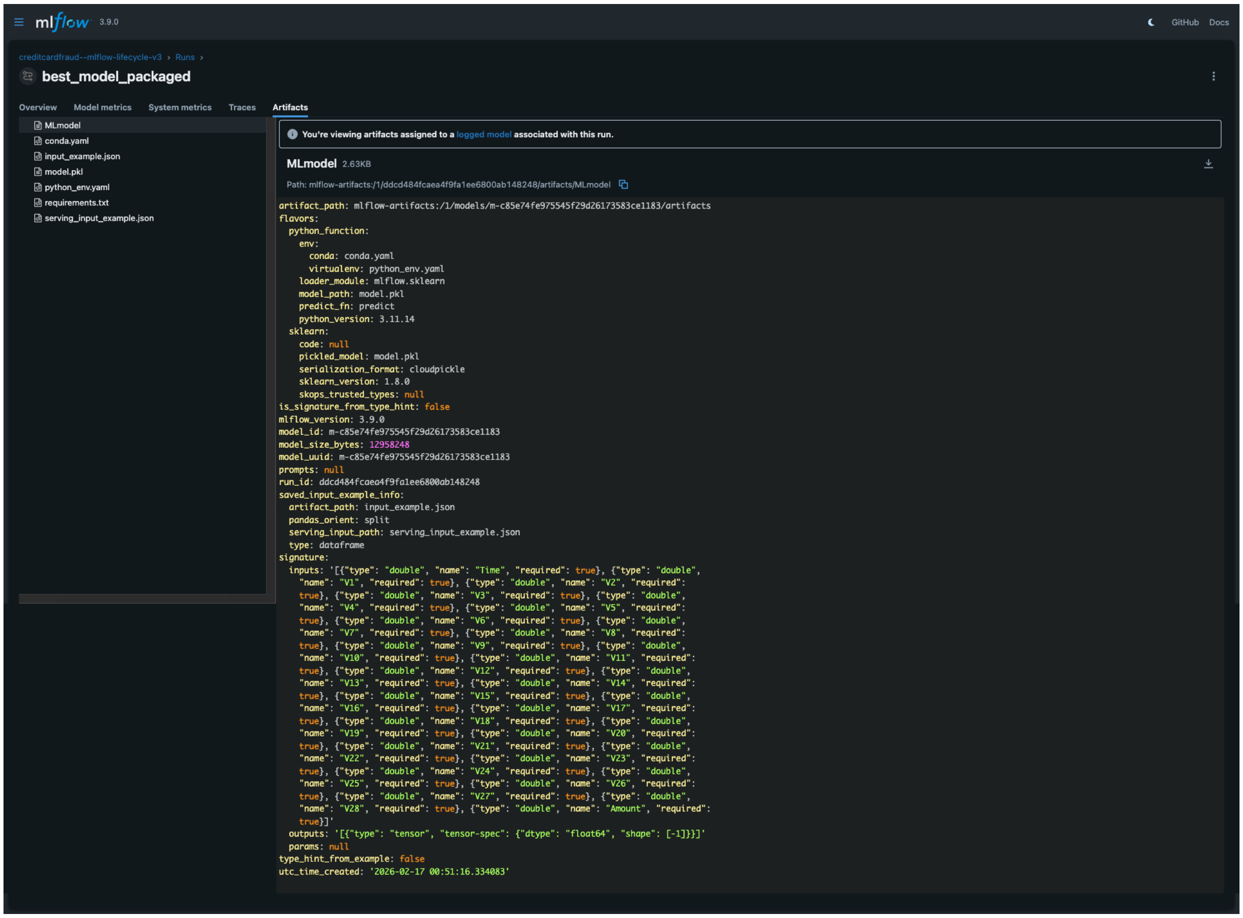

Figure 8: MLmodel Details

4.2 Input Example

```
{
  "columns": [
    "Time",
    "V1",
    "V2",
    "V3",
    "V4",
    "V5",
    "V6",
    "V7",
    "V8",
    "V9",
    "V10",
    "V11",
    "V12",
    "V13",
    "V14",
    "V15",
```

```
    "V16",
    "V17",
    "V18",
    "V19",
    "V20",
    "V21",
    "V22",
    "V23",
    "V24",
    "V25",
    "V26",
    "V27",
    "V28",
    "Amount"
  ],
  "data": [
    [
      161919,
      1.94674666728168,
      -0.752525821492348,
      -1.35512953289131,
      -0.661629913230695,
      1.50282190770174,
      4.02493282673061,
      -1.47966148383909,
      1.13988021689572,
      1.40681871374813,
      -0.157402912660682,
      -0.113729190329599,
      0.510277474082635,
      0.0612577807353701,
      -0.0665551394957695,
      1.32870182050036,
      0.352513902823239,
      -0.76566982664618,
      0.141937536859382,
      -0.451364651514137,
      -0.134434695487417,
      0.0761965101986691,
      0.297536541043559,
      0.307914569497895,
      0.690980284177926,
      -0.350316231256655,
      -0.388907264903717,
      0.0776408382302655,
      -0.0322477642202041,
      7.32
    ],
    [
      124477,
      2.03514918623719,
      -0.0488803143003422,
      -3.0586934537639,
      0.247945033712133,
      2.94348685227235,
```

  3.29869731157312,
  -0.0021924639358902,
  0.674781823977897,
  0.0458257088345251,
  0.28486427695311,
  -0.254903244909153,
  0.325559776048587,
  -0.405326546442527,
  0.7210684077145,
  -0.14844514500485,
  -0.754029381288908,
  -0.270842357681763,
  -0.695697841779672,
  -0.274411134595354,
  -0.227279116186558,
  0.0386282061449729,
  0.22819701963924,
  0.0355424132400545,
  0.707089987721862,
  0.512884596281977,
  -0.471197624846292,
  0.002519885391441,
  -0.0690016835527196,
  2.99
],
[
  41191,
  -0.991919644382192,
  0.603192643431104,
  0.711976017945661,
  -0.992424654946317,
  -0.825837705264936,
  1.95626134784203,
  -2.21260332725614,
  -5.03752305017971,
  0.0007717251978341,
  -2.00956091808815,
  -0.386845117309237,
  1.82016085004487,
  0.74777695874234,
  0.122746169585306,
  -1.72328460371944,
  1.12334415094773,
  -0.72461599457959,
  0.147254864861436,
  0.0046311776655328,
  1.28085631377318,
  -2.79835228768859,
  0.109525862489931,
  -0.436530155014404,
  -0.932803039560918,
  0.826684291046232,
  0.913773193731371,
  0.0380486856951838,
  0.18534031963132,

175.1
],
[
132624,
2.28571782188968,
-1.50023927705557,
-0.747565164139995,
-1.66811860629904,
-1.39414259363466,
-0.350338761903453,
-1.42798360644237,
0.010009593384969,
-1.11844672075498,
1.75612106388,
0.0931364463331114,
-0.72244987869778,
-0.468757242262352,
-0.195287757066107,
-0.63868321568293,
-0.0658875748672835,
0.0727807169728085,
0.7682372951547,
0.257424115538122,
-0.490641541573627,
-0.139669919903137,
0.0770132158391883,
0.208310169754412,
-0.538236349525744,
-0.278032030240713,
-0.162068187578527,
0.0180452709860015,
-0.0630052317493315,
6.1
],
[
59359,
-0.44874695349004,
-1.01144001580821,
0.115902670055464,
-3.45485413841999,
0.715770533817782,
-0.147490005243008,
0.504347404927186,
-0.113816843117209,
-0.0447824514547183,
-0.558955013780937,
-0.251076432441015,
-0.0547083000860354,
-0.782698209036227,
0.134659166035162,
-0.483007431899087,
-2.09609949871472,
-0.399525224990159,
1.59758878430156,
-0.082745581611311,

```
        -0.275296971138216,
        -0.243245217795912,
        -0.173298449581609,
        -0.0066924613429996,
        -1.3623830528357,
        -0.292234275970279,
        -0.144621719580633,
        -0.032580422581582,
        -0.0641935650630091,
        86.1
      ]
    ]
}
```


5 Part E: Model Registry and Lifecycle Management

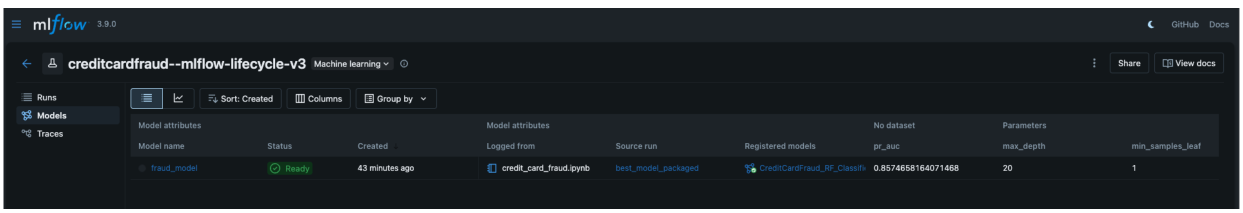

Figure 9: Models from Run

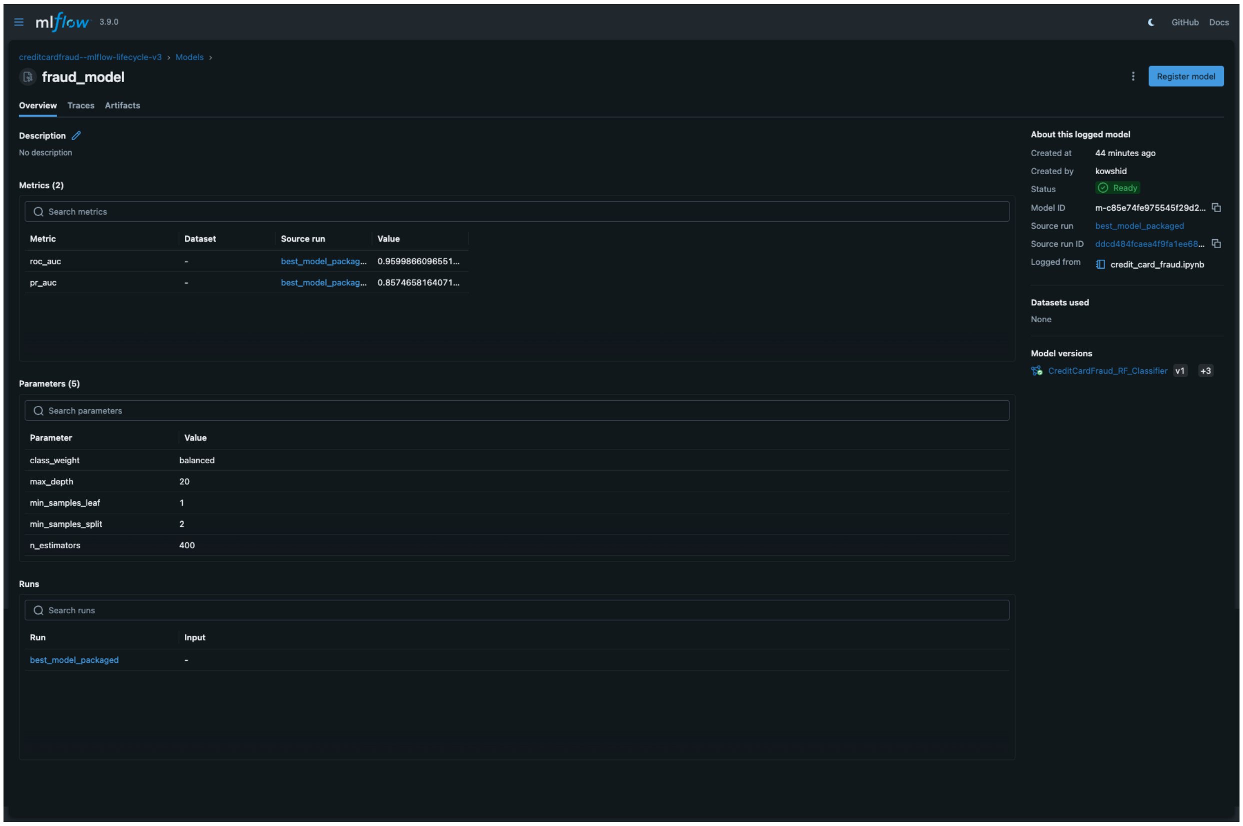

Figure 10: Overview of Model

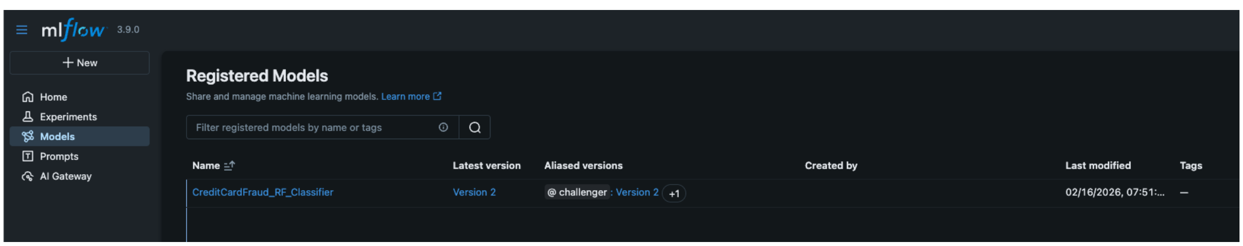

Figure 11: Registered Model

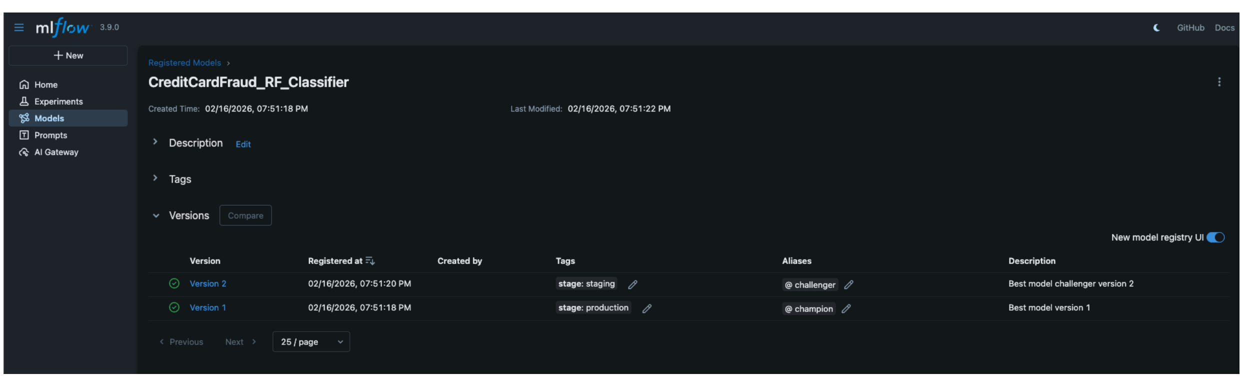

Figure 12: Registered Model Details

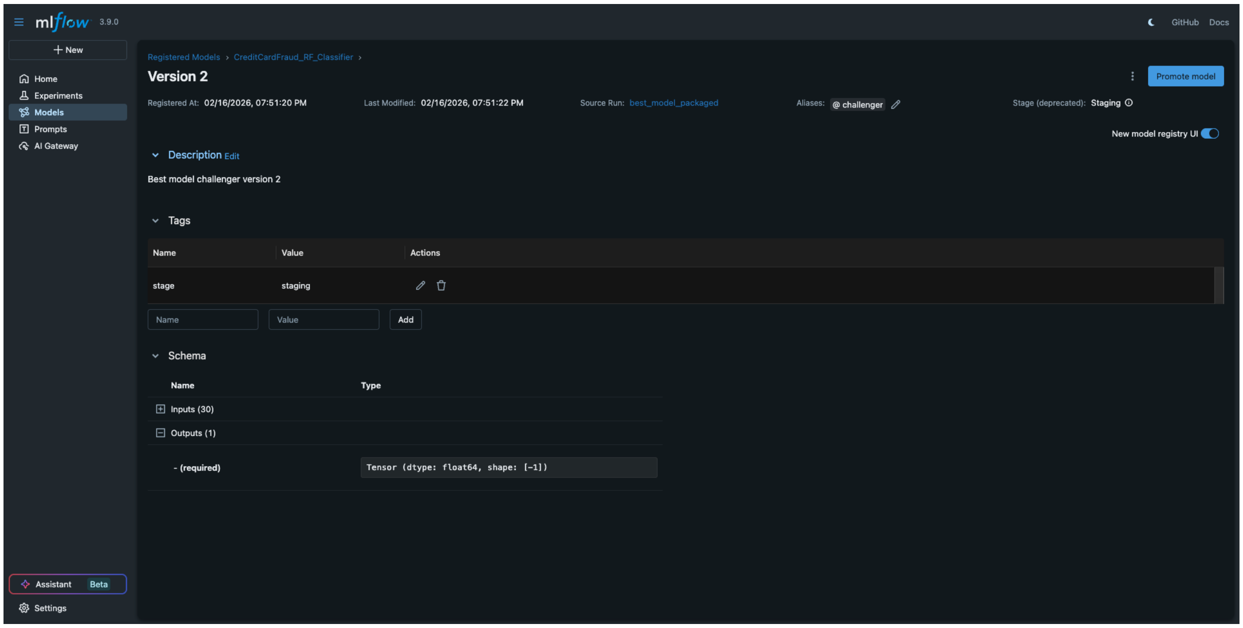

Figure 13: Registered Model Version Details

6 Part F: Reflection

MLflow solves practical problems in real ML systems by providing a single system of record for experiment runs, hyperparameters, metrics, artifacts, and packaged models, which improves reproducibility and team collaboration. It also supports governance through the Model Registry, where versions can be staged and promoted with documented rationale.

However, mistakes can still happen: data leakage, incorrect train/test splits, mislabeled data can occur even if everything is logged. Teams can also log incorrect/incomplete metadata and still lose reproducibility.

MLflow integrates well with DVC by using DVC for data version control while MLflow tracks run metadata and artifacts. We can log the DVC commit hash as a tag in MLflow, linking the exact data version to the experiment results. In a CI/CD pipeline, a change request could trigger a script that trains a model, logs it to MLflow, and compares its metrics against the current *Production* model. If the new model's metrics are better, the CI pipeline can automatically register it and move it to *Staging*. This makes promotion decisions auditable and automatic.