National College of Ireland

Configuration Manual

MSc Data Analytics
Research Practicum Part 2

Pranav Dinesh Ahire
Student ID: X23302372

School of Computing
National College of Ireland

Supervisor: Qurrat Ui Ain

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pranav Dinesh Ahire
Student ID:	X23302372
Programme:	Msc Data Analytics **Year:** 2024-25
Module:	Research Practicum Part 2
Lecturer:	Qurrat Ui Ain
Submission Due Date:	11 August 2025
Project Title:	Real-Time AI-Powered Anomaly Detection in Financial Transactions Using Microsoft Fabric and Power BI
Word Count:	**.....943.....Page Count:** .....18.....

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: .....

Date: .....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Pranav Dinesh Ahire
Student ID: x23302372

1 Requirements

1.1 Hardware Requirements

1. **Processor (CPU): Intel Core i7 or higher**
2. **Ram (Memory): 16GB**
3. **Storage: SSD with at least 256GB for faster I/O**
4. **Graphics (GPU): NVIDIA/AMD**

1.2 Software Requirements

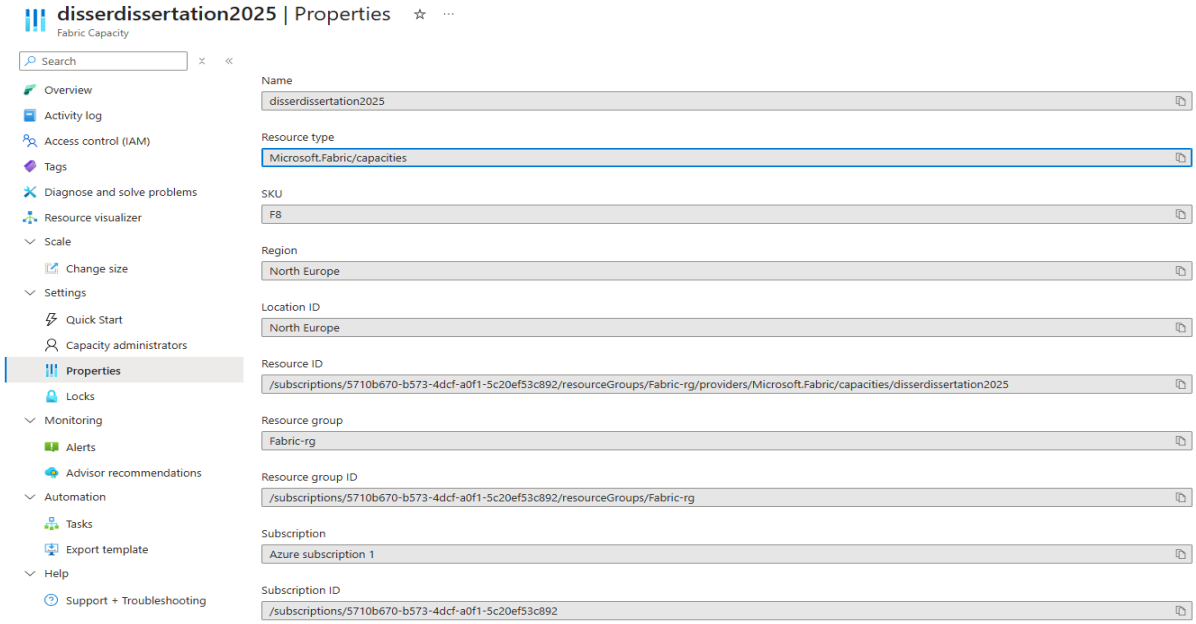
1. **Windows 11 (64-bit)**
2. **Azure Cloud Services (Create Resource for Micrcosoft Fabric)**
3. **Microsoft Fabric (F8)**
4. **Power BI (Services)**

2 Environment Setup

2.1 Azure and Microsoft Fabric Access

Accessed Microsoft Fabric https://portal.azure.com/#home

1. Create a Resource for Microsoft Fabric Capacity and enabled F8 Capacity from Admin portal under Capacity setting and enabled premium capacity.



2. Named it "**disserdissertation2025**"

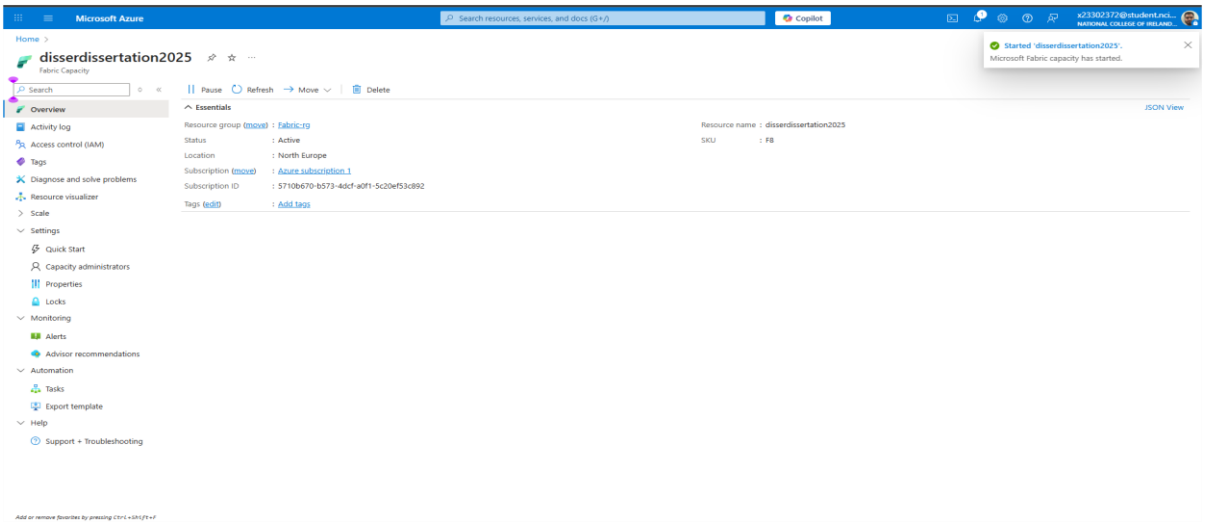


Figure 1AZURE Final Setup

2.2 Workspace Configuration in Microsoft Fabric

1. Open Microsoft Fabric in Browser login with same Id and configured with Azure resource (disserdissertation2025).
2. Create workspace in Microsoft Fabric and named **"Dissertation-FraudDetection".**

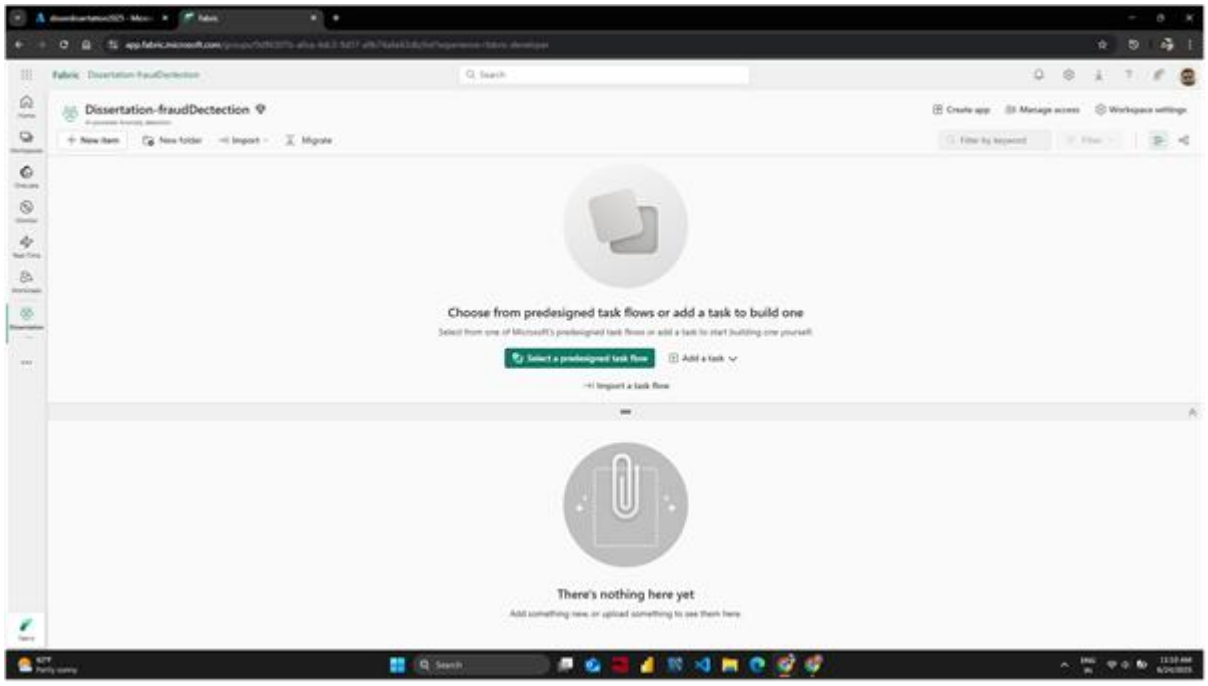


Figure 2. Workspace Created in Microsoft Fabric

3. Start Working on Project in the workspace .

3 Data Uploading and Lake house Configuration

3.1 Creating lakehouse

1. Click on **“New item”** and select Lakehouse and named it **“FraudDetectionlakehouse”.**

2 Uploaded all three **.csv files (transaction_data, users_data, cards_data)** in the lakehouse.

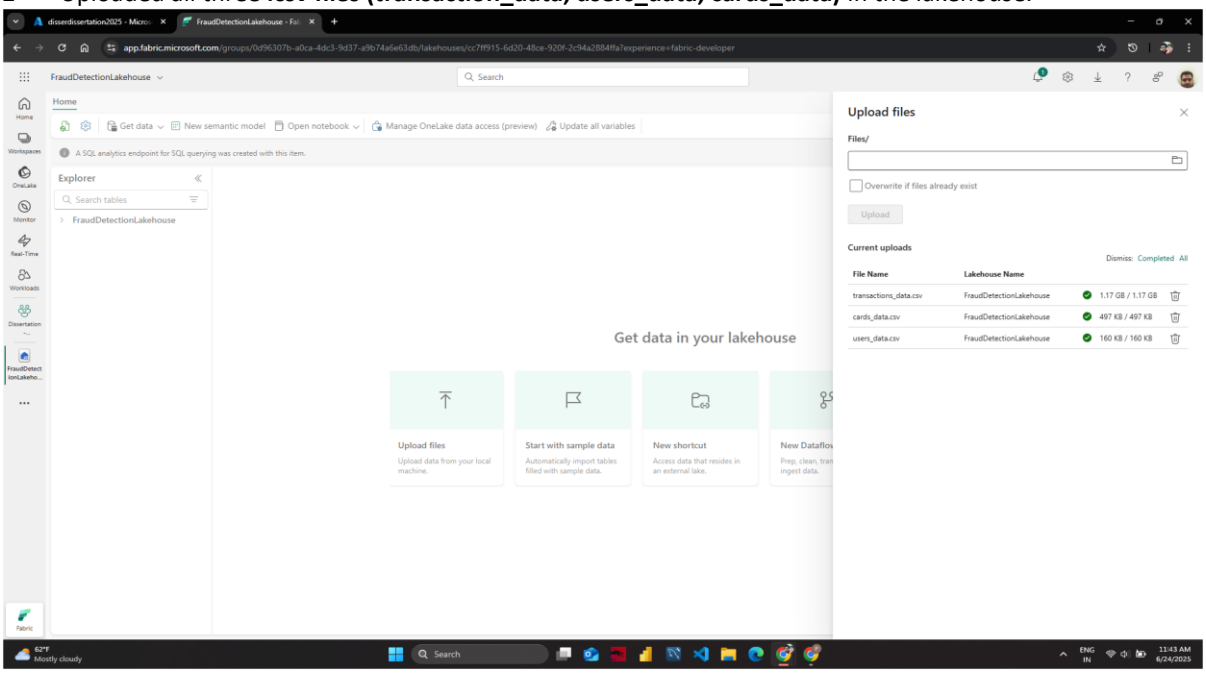


Figure 3. Dataset are all three datasets are uploaded in in lakehouse

3.2 Explore all three datasets.

1. Click on **“New Item”** add notebooks and named **“Dataset_Quick_Review”.**

2. Explore all three dataset. Start with card.csv data

2.1. Read the data from Microsoft Fabric Lakehouse using Pyspark and display first 20 rows.

```
1
2  df1 = spark.read.table("FraudDetectionLakehouse.cards")
3  df1.show()
4
```

[15] ✓ - Command executed in 2 sec 917 ms by Pranav Dinesh Ahire on 12:32:26 PM, 6/24/25

Figure 4.Code cell to read the data.

2.2. Knew the schema of each columns.

```
1  print("Schema for: cards")
2  df1.printSchema()
3
```

[16] ✓ - Command executed in 253 ms by Pranav Dinesh Ahire on 12:32:43 PM, 6/24/25

Figure 5.Code cell to explore the schemas.

2.3. Repeat step 2.1 and 2.2 for transaction.csv and users.csv dataset.

4 Feature Engineering in Dataflow Gen2

4.1 Dataflow Gen2 Creation

1. Launched **dataflow Gen2** with source FraudDetection Lakehouse and named **"FraudDetectionEngineering".**

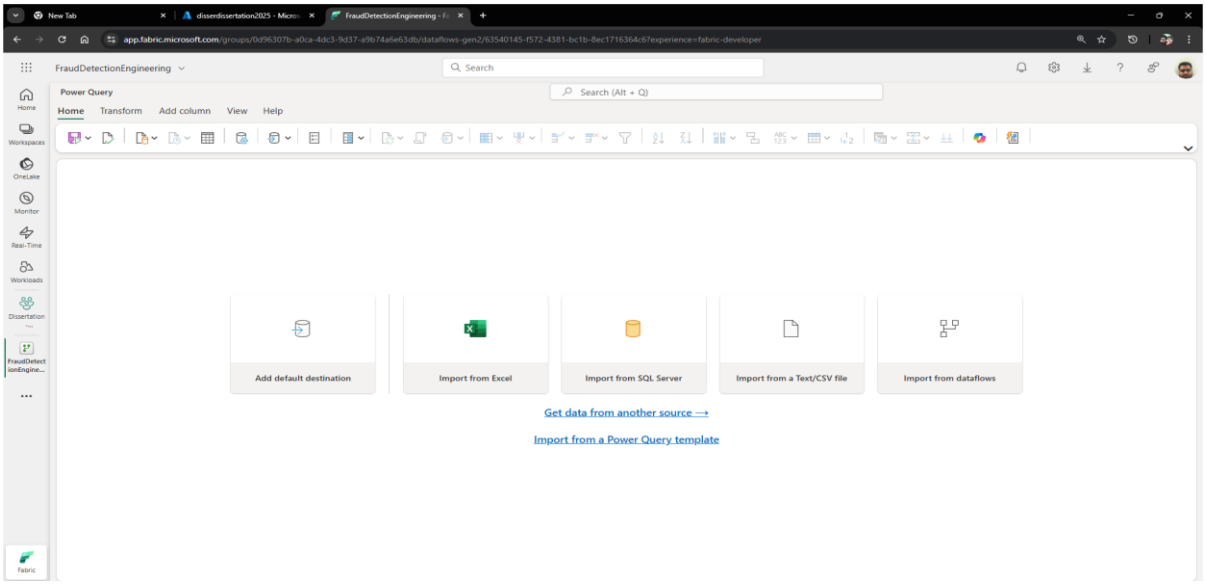


Figure 6.Dataflow Gen2 Successfully created.

2. Reviewed all columns and their values also checked their datatype.

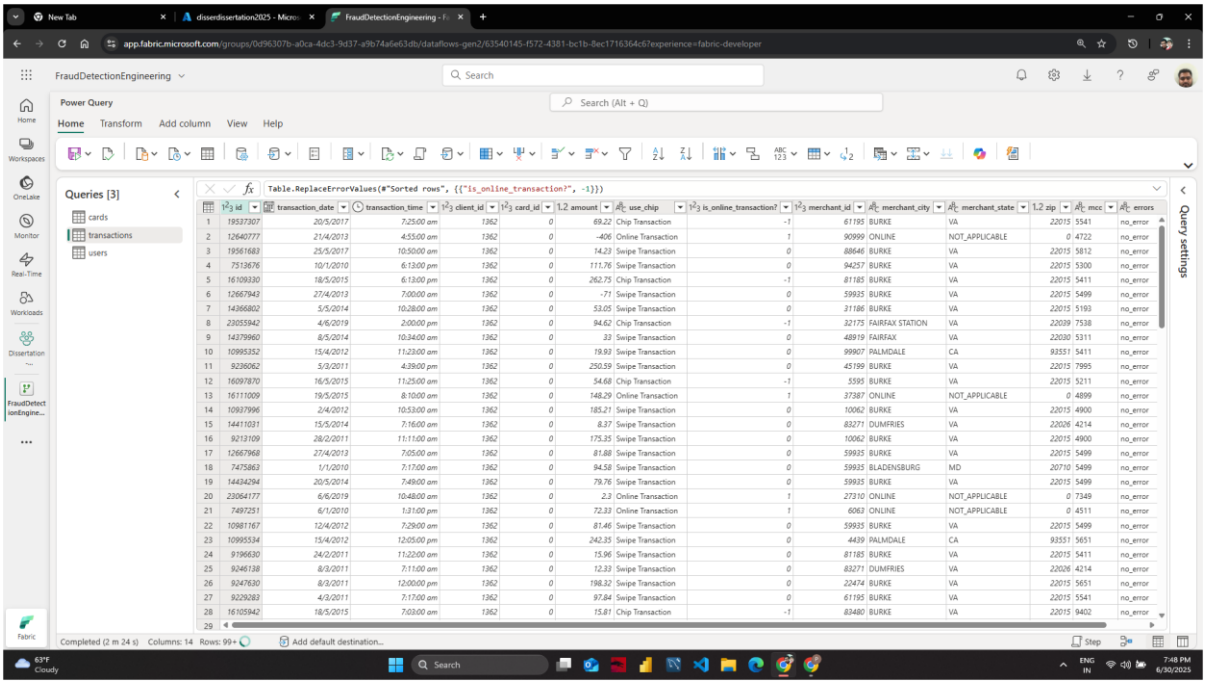


Figure 7. All 3 dataset in Dataflow Gen2.

3. After reviewing all datasets merged card data with transaction data using left outer joined and named **“transaction_card.csv”.**

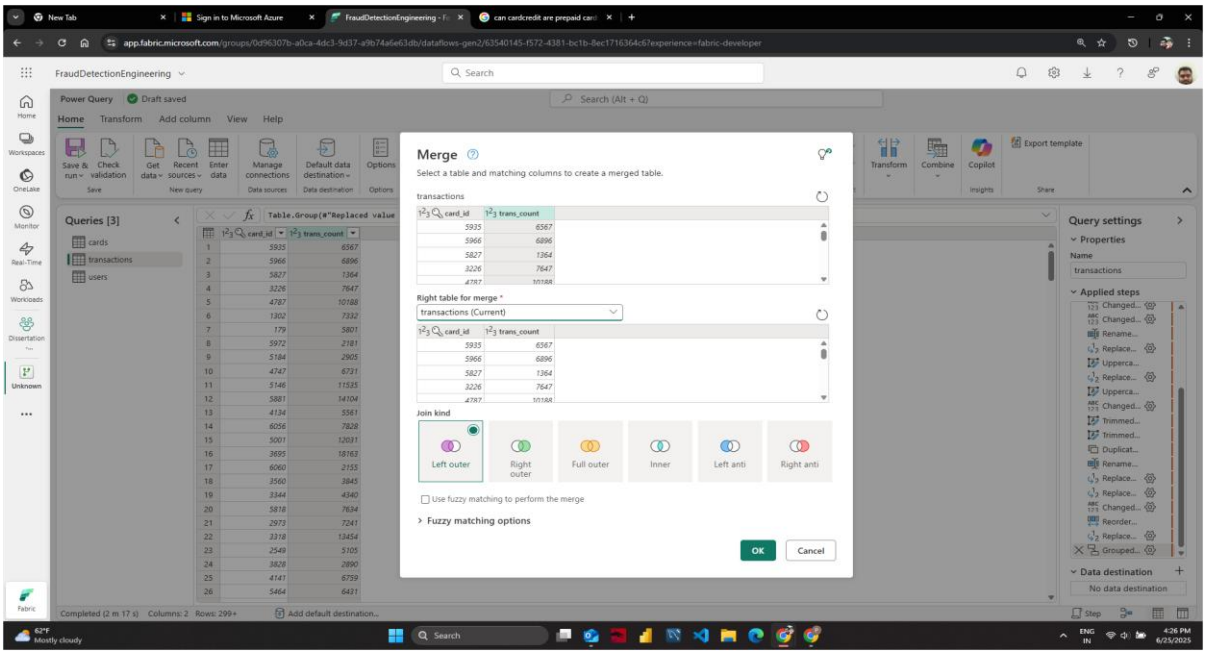


Figure 8. Merged card.csv with transaction.csv using left outer joint

4. Again merged transaction_cards.csv dataset with user.csv dataset using Left outer join and named **“ transaction_cards_users.csv”.**

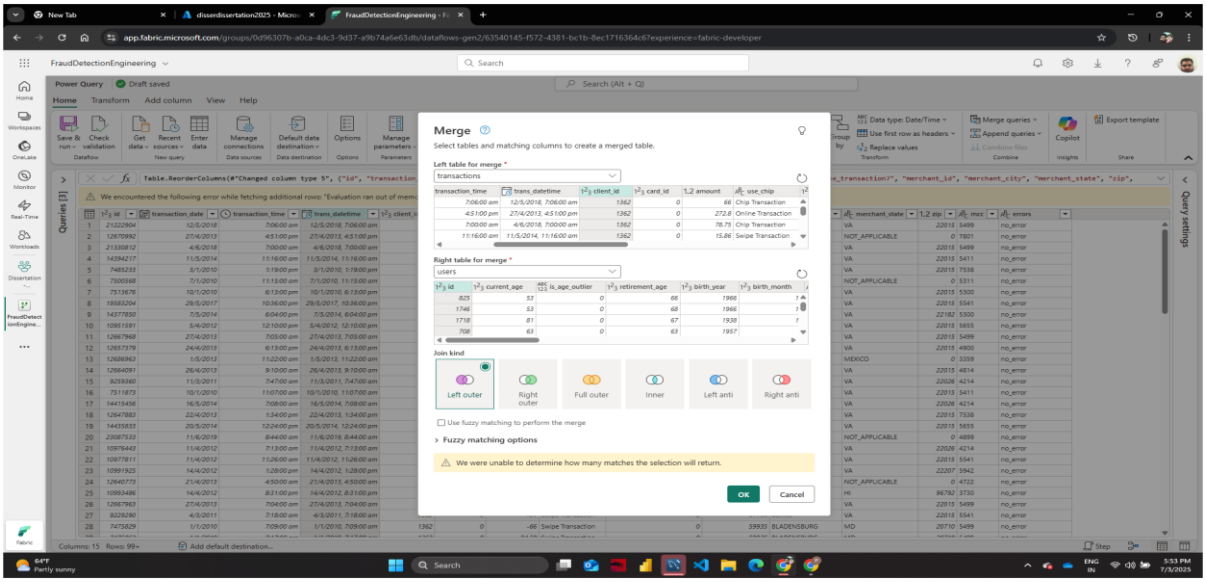


Figure 9. Merged users.csv with transaction_cards_users.csv using left outer join.

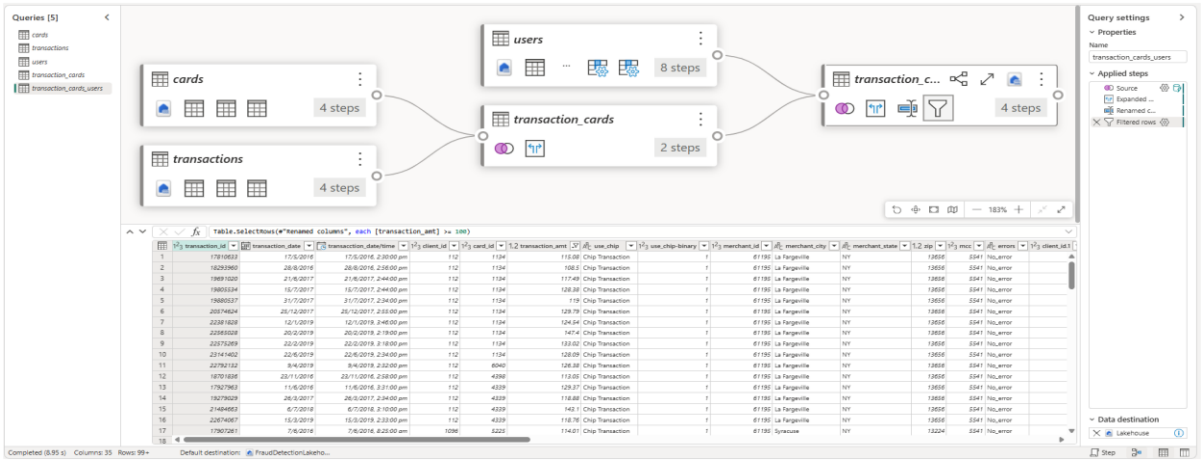


Figure 10. results after all joining.

4.2 Derived Features

1. **"transaction_cards_users.csv"** opened in Power Query Editor in Create custom columns using DAX formula in **Power Query Editor.**

2. Go to **"Add column"** and click on **"conditional column".**

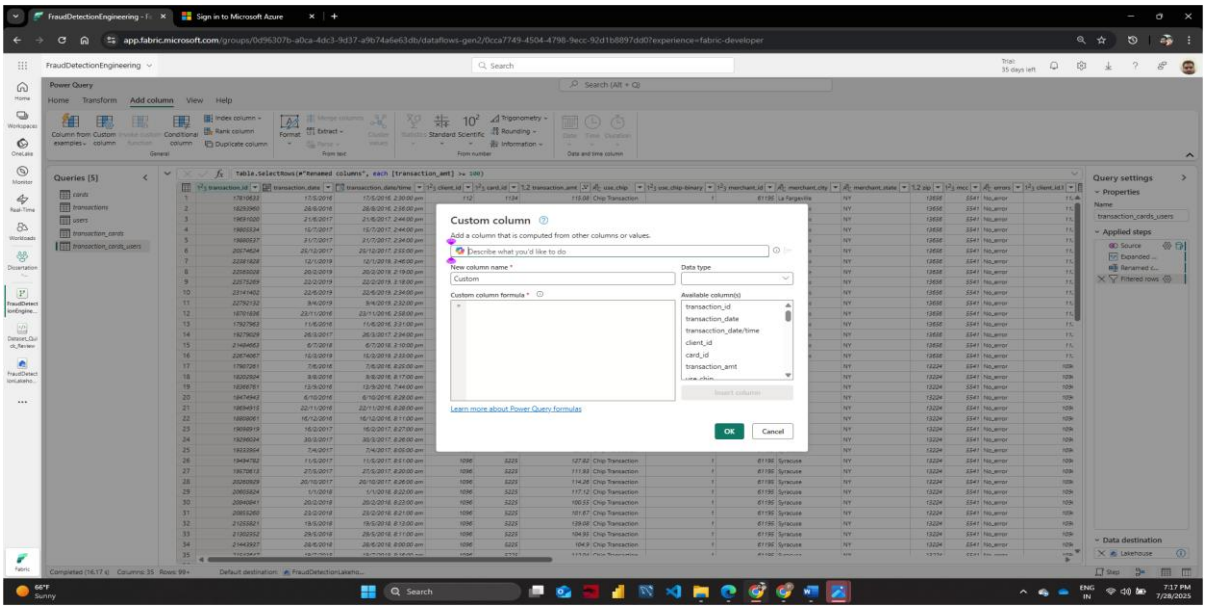


Figure 11. add custom column

3. For custom column **is_low_income_user?**

Formula = Table.AddColumn(#"PreviousStep", "is_low_income_user?", each if [user_income] < 25000 then 1 else 0, Int64.Type)

4. For **is_card_expired (Compare card_expiry_date with trans_date.)**

Formula = Table.AddColumn(#"PreviousStep", "is_card_expired", each if [card_expiry_date] < [trans_date] then 1 else 0, Int64.Type)

5. For **is_card_on_dark_web (binary column compare with card_on_Darkweb)**

Formula = Table.AddColumn(#"PreviousStep", "is_card_on_dark_web", each if Text.Lower([card_on_darkweb]) = "yes" then 1 else 0, Int64.Type)

5 Modeling using Synapse Notebook

5.1 Create Synapse Notebook

1. Created Synapse notebook and named **“FruadDetection_ML”**
2. Trained ML Models as per the scenarios

5.2 Apply ML model as per the scenarios

1. **Import Specific required Python Libraries.**

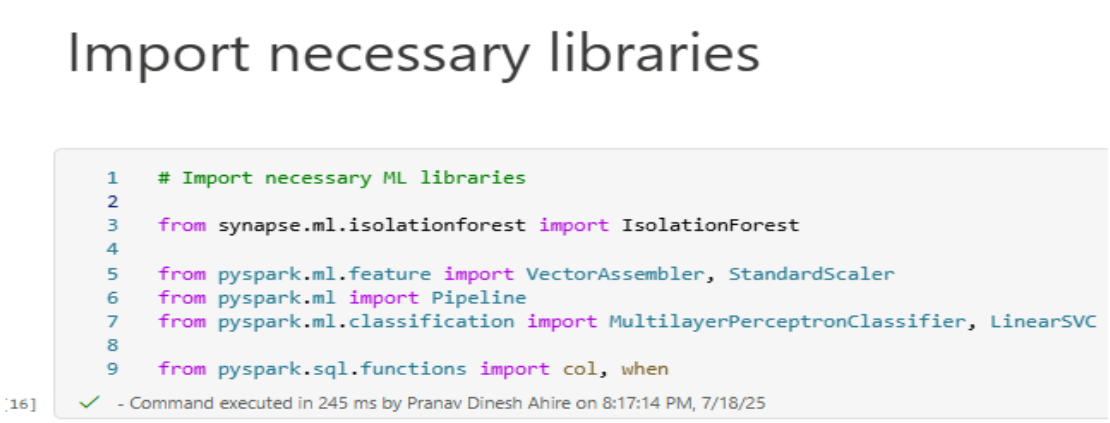
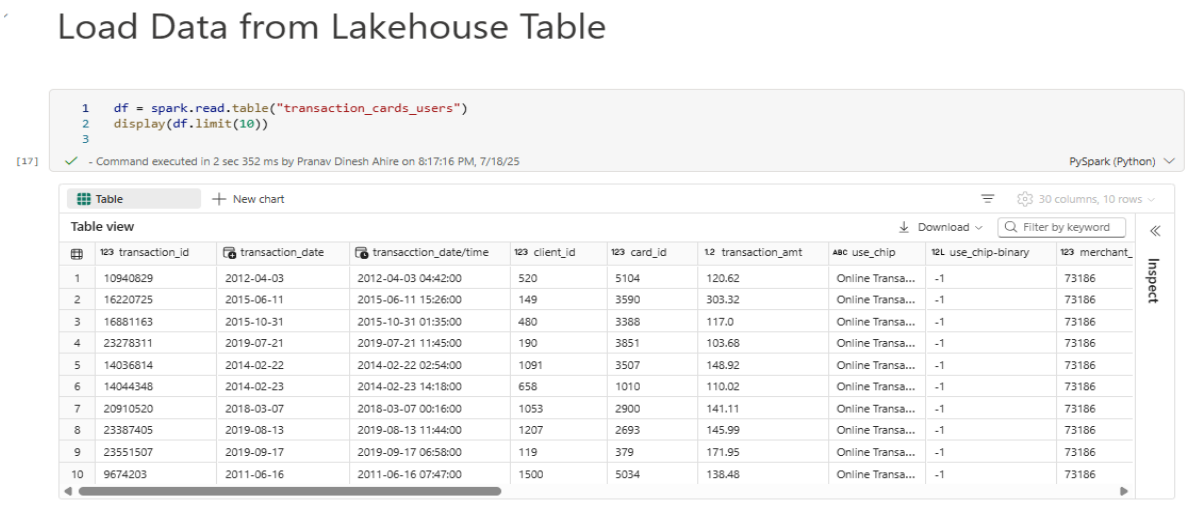


Figure 12. Screenshot of Import Library.

2. Load the preprocessed and feature-engineered table **(transaction_cards_users)** from the Microsoft Fabric Lakehouse. And displayed first 10 rows.



	transaction_id	transaction_date	transacction_date/time	client_id	card_id	transaction_amt	use_chip	use_chip-binary	merchant_
1	10940829	2012-04-03	2012-04-03 04:42:00	520	5104	120.62	Online Transa...	-1	73186
2	16220725	2015-06-11	2015-06-11 15:26:00	149	3590	303.32	Online Transa...	-1	73186
3	16881163	2015-10-31	2015-10-31 01:35:00	480	3388	117.0	Online Transa...	-1	73186
4	23278311	2019-07-21	2019-07-21 11:45:00	190	3851	103.68	Online Transa...	-1	73186
5	14036814	2014-02-22	2014-02-22 02:54:00	1091	3507	148.92	Online Transa...	-1	73186
6	14044348	2014-02-23	2014-02-23 14:18:00	658	1010	110.02	Online Transa...	-1	73186
7	20910520	2018-03-07	2018-03-07 00:16:00	1053	2900	141.11	Online Transa...	-1	73186
8	23387405	2019-08-13	2019-08-13 11:44:00	1207	2693	145.99	Online Transa...	-1	73186
9	23551507	2019-09-17	2019-09-17 06:58:00	119	379	171.95	Online Transa...	-1	73186
10	9674203	2011-06-16	2011-06-16 07:47:00	1500	5034	138.48	Online Transa...	-1	73186

Figure 13. Screenshot Load from Lakehouse

3. List of available Columns.

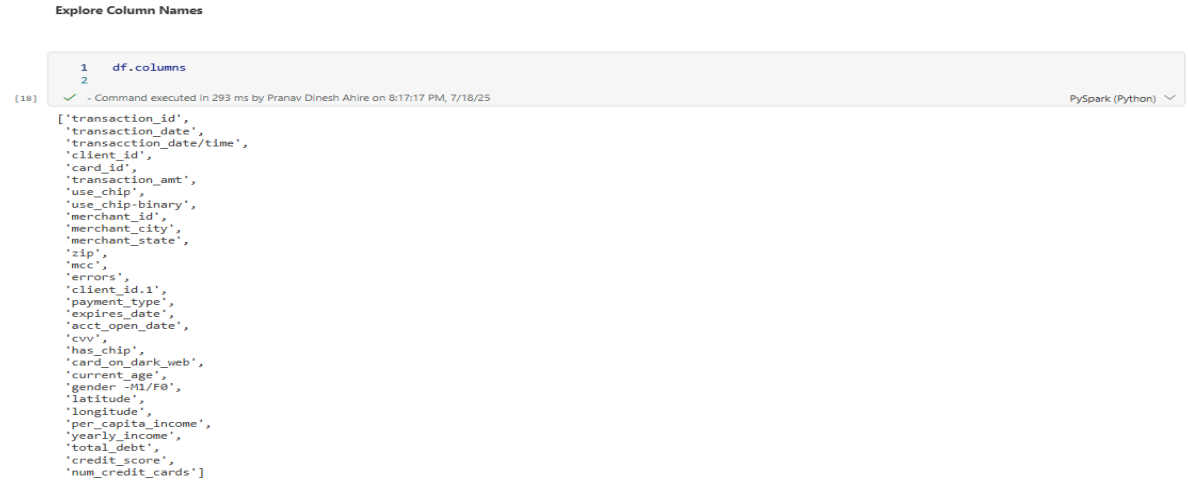


Figure 14.Explore Columns

4. Initial null fill to prevent model failure dure to nulls.

Figure 15.Cleaning to Prevent Model Failure

5. Extended null value treatment before feature assembly.

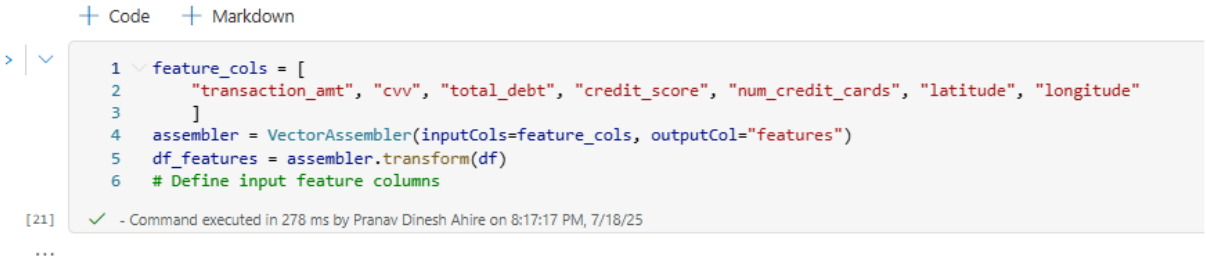


Figure 16.Feature assembling

6. Assembled selected numeric columns into single vector features, which required for all spark ML models.

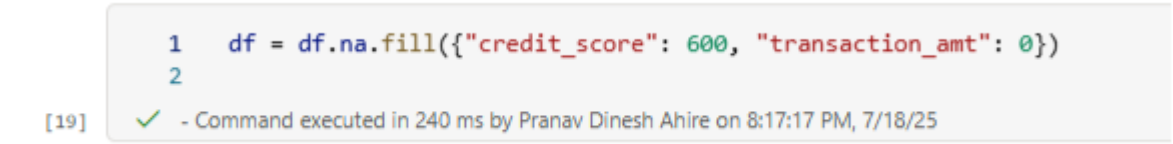


Figure 17. Column Normalization

7. Standardizes the features using z-score normalization, improving model convergence and performance.

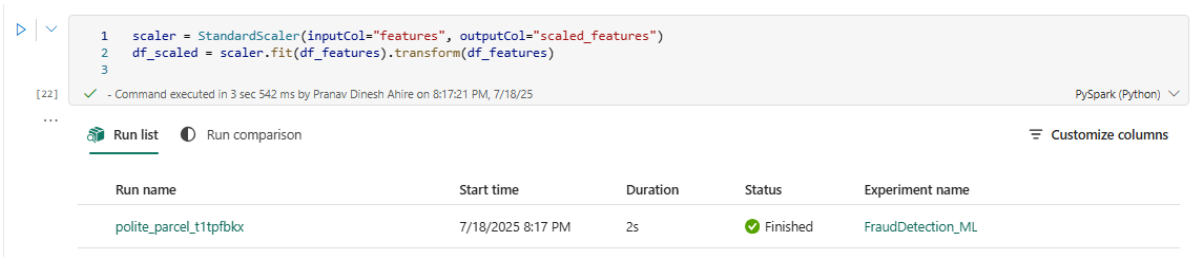


Figure 18. Scaling

8. Train **Isolation Forest** for **scenario 1 (High-value transactions by low-income users)**

To detect anomalies using features. it adds **outlierScore** and **predictedlabel** to the results.

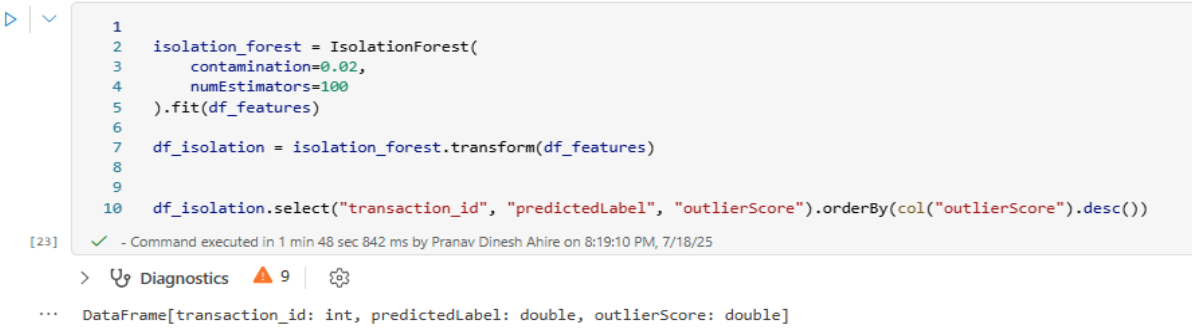


Figure 19. Apply Isolation Forest

9. Labeled **(heuristic Labeling)** Creation from Outlier Score it used to trained supervised models

Figure 20.Labeling from Outlier Score

10. Trained **ANN (Multilayer Perception Classifier)** for **Scenario 2 (Transactions via Expired/ Dark Web Cards)** and return **ann_prediction.**

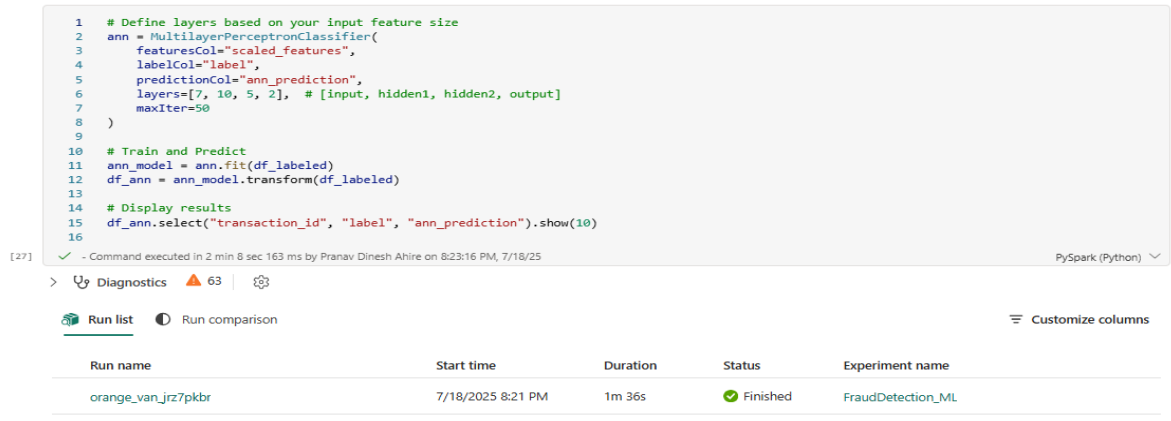


Figure 21. Apply ANN Model

11. Trained **SVM** for **Scenario 3(Users High Debt, Low Credi Score)** and returned **svm_predition.**

Figure 22. Apply SVM Model

12. Merge all three model predictions and displayed first 10

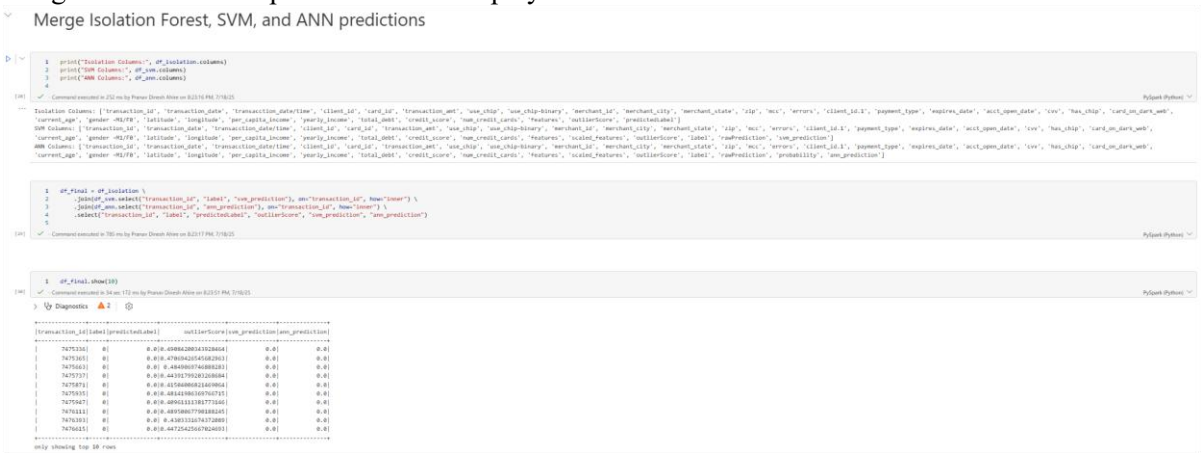


Figure 23. Merge All three Model results

13. **Evaluation of isolation Forest, ANN, and SVM**. Evaluate **Accuracy, Precision, Recall and F1 score** for each model using **multiClassifierEvaluator** also show confusion matrix of an each model.

Figure 24. Evaluation Isolation Forest.

Evaluation for ANN

```
# Initialize evaluator for ANN
evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="ann_prediction")

# Accuracy
ann_accuracy = evaluator.setMetricName("accuracy").evaluate(df_final)
# F1 Score
ann_f1 = evaluator.setMetricName("f1").evaluate(df_final)
# Precision
ann_precision = evaluator.setMetricName("precisionByLabel").evaluate(df_final)
# Recall
ann_recall = evaluator.setMetricName("recallByLabel").evaluate(df_final)

print("ANN Evaluation:")
print(f"Accuracy: {ann_accuracy:.4f}")
print(f"Precision: {ann_precision:.4f}")
print(f"Recall: {ann_recall:.4f}")
print(f"F1 Score: {ann_f1:.4f}")

# Confusion Matrix
print("Confusion Matrix (ANN):")
df_final.groupBy("label", "ann_prediction").count().show()
```

[33] ✓ - Command executed in 2 min 22 sec 759 ms by Pranav Dinesh Ahire on 8:33:17 PM, 7/18/25

> Diagnostics ⚠ 4

```
ANN Evaluation:
Accuracy: 0.9933
Precision: 0.9949
Recall: 0.9983
F1 Score: 0.9927
Confusion Matrix (ANN):
+-----+--------------+-------+
|label|ann_prediction|  count|
+-----+--------------+-------+
|    1|           0.0|   7884|
|    0|           0.0|1539385|
|    1|           1.0|   9702|
|    0|           1.0|   2641|
+-----+--------------+-------+
```

Figure 25. Evaluation ANN Model.

Evaluation for SVM

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

# Initialize evaluator for SVM
evaluator = MulticlassClassificationEvaluator(labelCol="label", predictionCol="svm_prediction")

# Accuracy
svm_accuracy = evaluator.setMetricName("accuracy").evaluate(df_final)
# F1 Score
svm_f1 = evaluator.setMetricName("f1").evaluate(df_final)
# Precision
svm_precision = evaluator.setMetricName("precisionByLabel").evaluate(df_final)
# Recall
svm_recall = evaluator.setMetricName("recallByLabel").evaluate(df_final)

print("SVM Evaluation:")
print(f"Accuracy: {svm_accuracy:.4f}")
print(f"Precision: {svm_precision:.4f}")
print(f"Recall: {svm_recall:.4f}")
print(f"F1 Score: {svm_f1:.4f}")

# Confusion Matrix
print("Confusion Matrix (SVM):")
df_final.groupBy("label", "svm_prediction").count().show()
```

[32] ✓ - Command executed in 2 min 35 sec 719 ms by Pranav Dinesh Ahire on 8:30:54 PM, 7/18/25

> Diagnostics ⚠ 2

```
SVM Evaluation:
Accuracy: 0.9901
Precision: 0.9903
Recall: 0.9998
F1 Score: 0.9865
Confusion Matrix (SVM):
+-----+--------------+-------+
|label|svm_prediction|  count|
+-----+--------------+-------+
|    1|           0.0|  15132|
|    0|           0.0|1541643|
|    1|           1.0|   2454|
|    0|           1.0|    383|
+-----+--------------+-------+
```

Figure 26.Evaluation SVM Model

14. Merge Evaluation results of all three models and displayed

```python
# Create score rows manually dictionary-based joins
from pyspark.sql import Row

# Convert metric values into Spark DataFrame for each model
score_rows = [
    Row(model="IsolationForest", accuracy=iso_accuracy, precision=iso_precision, recall=iso_recall, f1=iso_f1),
    Row(model="SVM", accuracy=svm_accuracy, precision=svm_precision, recall=svm_recall, f1=svm_f1),
    Row(model="ANN", accuracy=ann_accuracy, precision=ann_precision, recall=ann_recall, f1=ann_f1),
]

# Create Spark DataFrame of evaluation results
df_metrics = spark.createDataFrame(score_rows)
df_metrics.show()

```

[34] ✓ - Command executed in 784 ms by Pranav Dinesh Ahire on 8:33:18 PM, 7/18/25

```
+---------------+------------------+------------------+------------------+------------------+
|          model|          accuracy|         precision|            recall|                f1|
+---------------+------------------+------------------+------------------+------------------+
|IsolationForest|0.9912619292490696|               1.0|0.9911622761224519|0.9924626442412607|
|            SVM|0.9900520129365509|0.9902799055740232|0.9997516254589741|0.9864835884270508|
|            ANN|0.9932515266617594| 0.994904570569177|0.9982873181126648|0.9926661502117234|
+---------------+------------------+------------------+------------------+------------------+
```

Figure 27. Final Evaluation table.

15. Load both **(prediction and Evaluation)** tabled into lakehouse. For visualizations.

```python
df_final.write.mode("overwrite").saveAsTable("fraud_prediction_results")
# Save final prediction table
#df_final.write.mode("overwrite").option("header", True).csv("Files/final_predictions")

# Save evaluation metrics
df_metrics.write.mode("overwrite").saveAsTable("model_evaluation_score")
```

[35] ✓ - Command executed in 52 sec 70 ms by Pranav Dinesh Ahire on 8:34:10 PM, 7/18/25

Diagnostics ⚠ 1

Figure 28. Load Both Tables into Lakehouse.

16. Both tables from Lakehouse are loaded into Power BI Services for visualizations.

6 Visualizations

6.1 Connected Microsoft Fabric and Power BI Services.

1. **Login Power BI services** with the same email id and configured with **DissertationFraudDetection** Workspace.

2. Power BI services Connected to FraudDetectionLakehouse.

(Click on “Get Data” → “ Microsoft Fabric”)

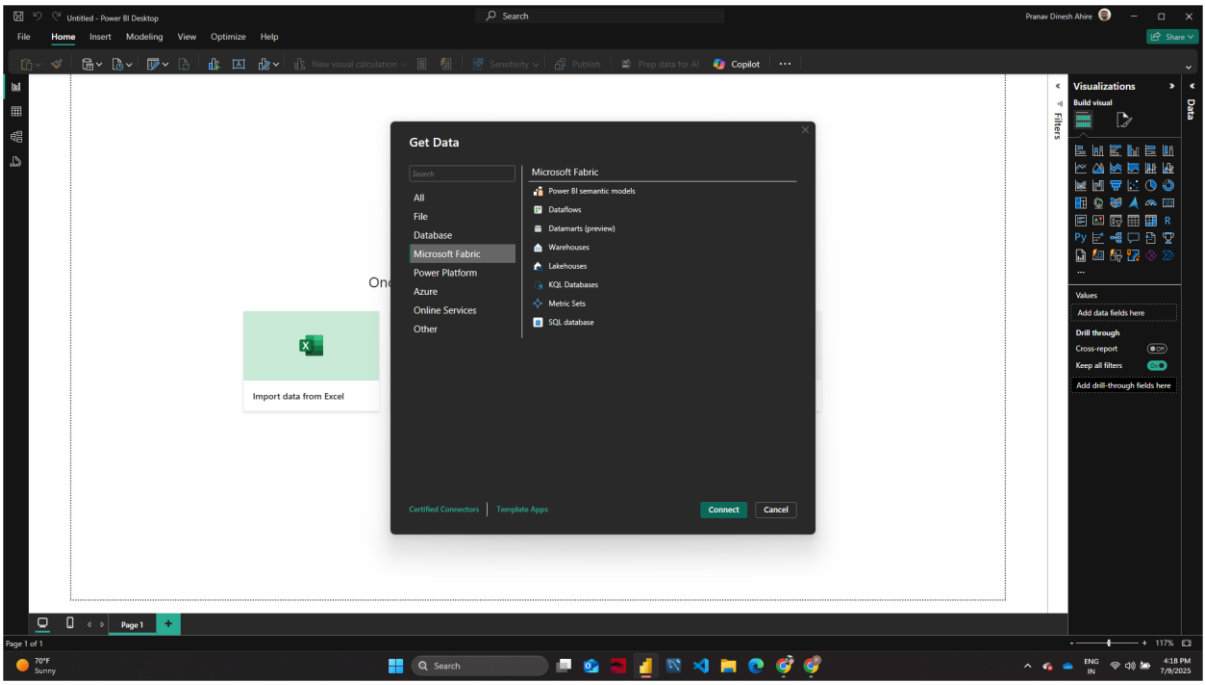


Figure 29. Connect Microsoft Fabric to Power BI Services.

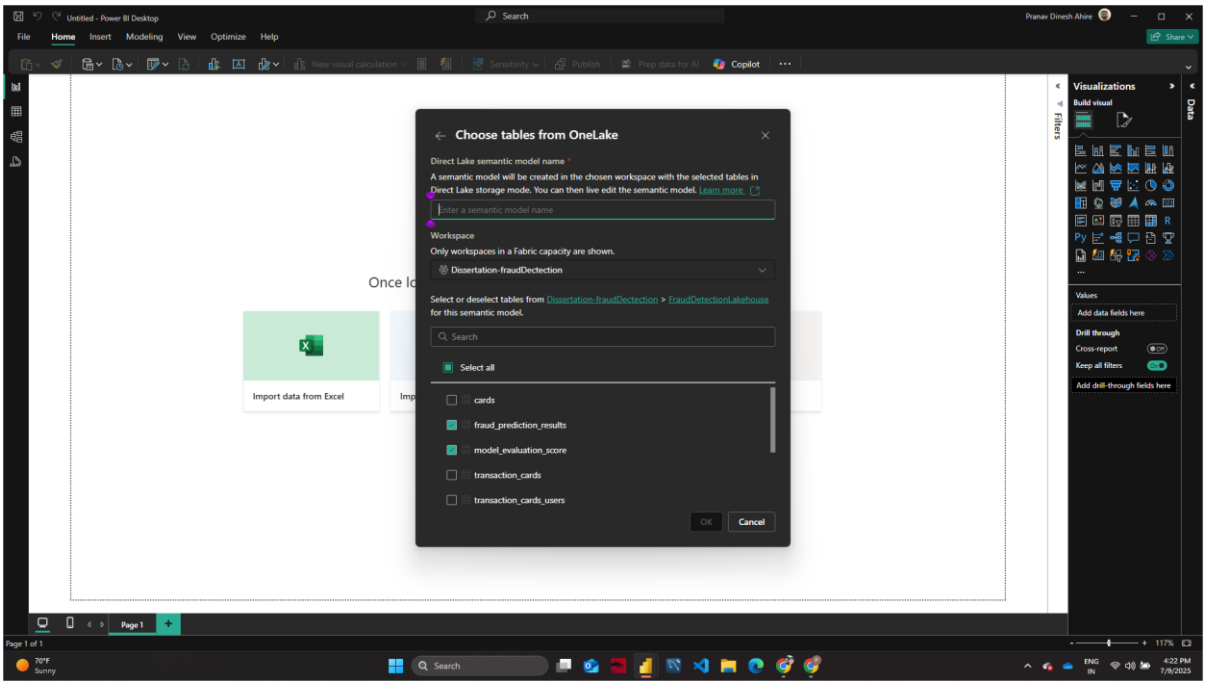


Figure 30. Load Both Tables from Lakehouse

6.2 Dashboard Developed

After Power BI Services connected with Microsoft Fabrics Lakehouse Two Dashboards (Created Fraud Detection Overview and Model Evaluation Summary)

5.2.1 Dashboard 1: Fraud Detection Overview

Visuals:

1. **Cards visuals: Total transaction, Legitimate count, Fraud count.**
2. **Pie chart: Legitimate vs. Fraud Count.**
3. **Bar chart: Prediction vs. Actual Prediction for ANN and SVM**
4. **Slicers: Filter by Actual label (Fraud or Legit)**

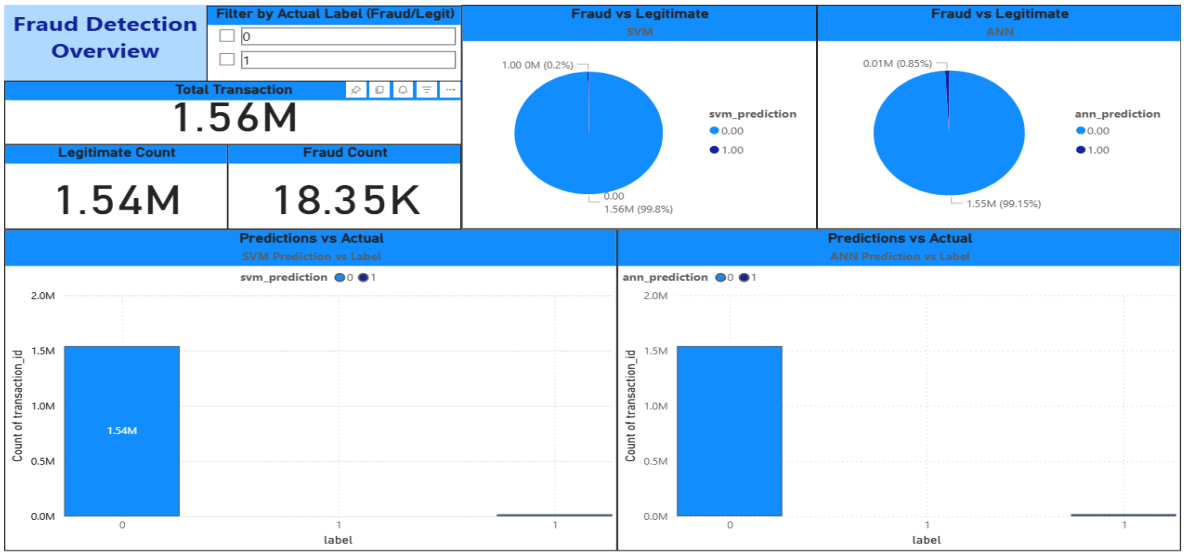


Figure 31. Dashboard1: Fraud Detection Overview.

5.2.2 Dashboard 2: Model Evaluation Summary

Visuals:

1. **Card Visual: F1 Score value and Accuracy for all models (Ascending ordered).**
2. **Score Table: Table format shows exact value of accuracy, Precision, Recall, F1 Score.**
3. **Bard Chart: Visuals Comparison of metrics across all models.**

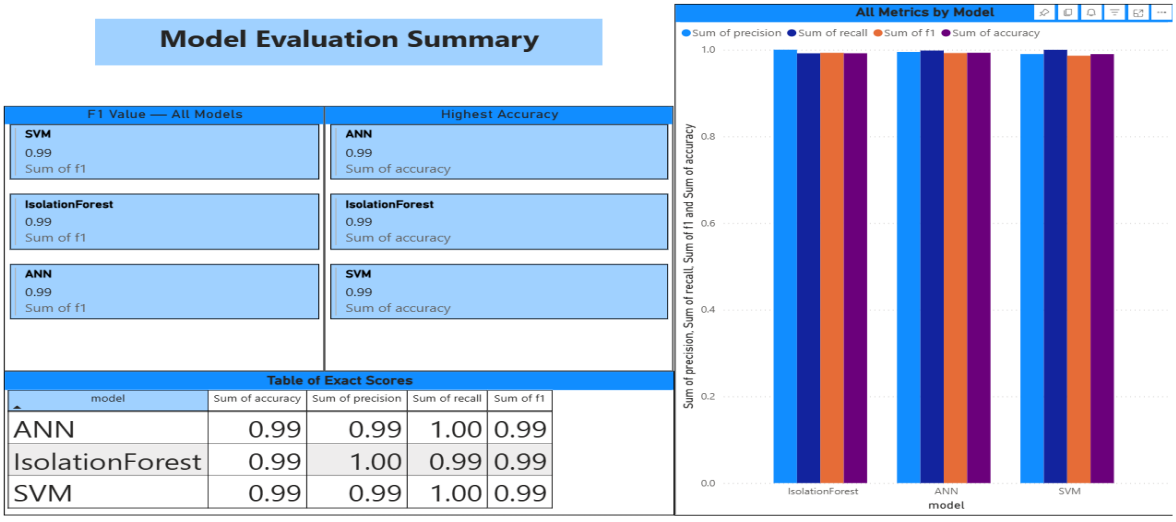


Figure 32. Dashboard2: Model Evaluation Summary.

7 Pipeline Deployment

7.1 Create pipeline Notebook

1. Click on **“New item”** add **“Data Pipeline”.**
2. **Connected to Workspace “Dissertation-FraudDetection”.**
3. **Add Dataflow1, Notebook1 and Semantic model refresh.**
4. **Connect Dataflow1 to Notebook1 and Notebook1 to Semantic model Refresh**

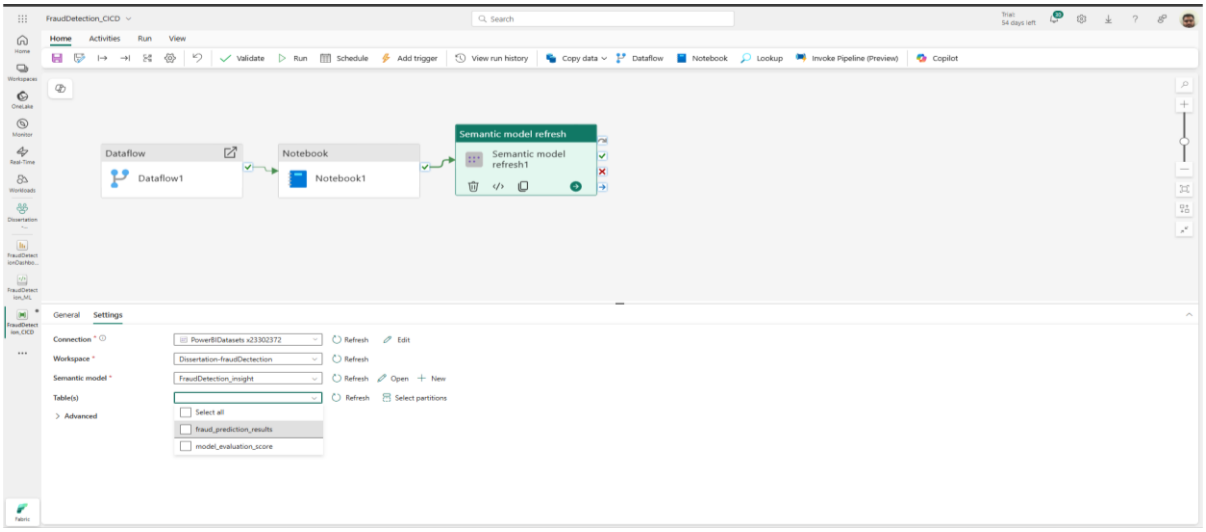


Figure 33. Pipeline Deployment

5. **Validate the Pipeline and Run.**

6. **The Pipeline Deployed Successfully.**

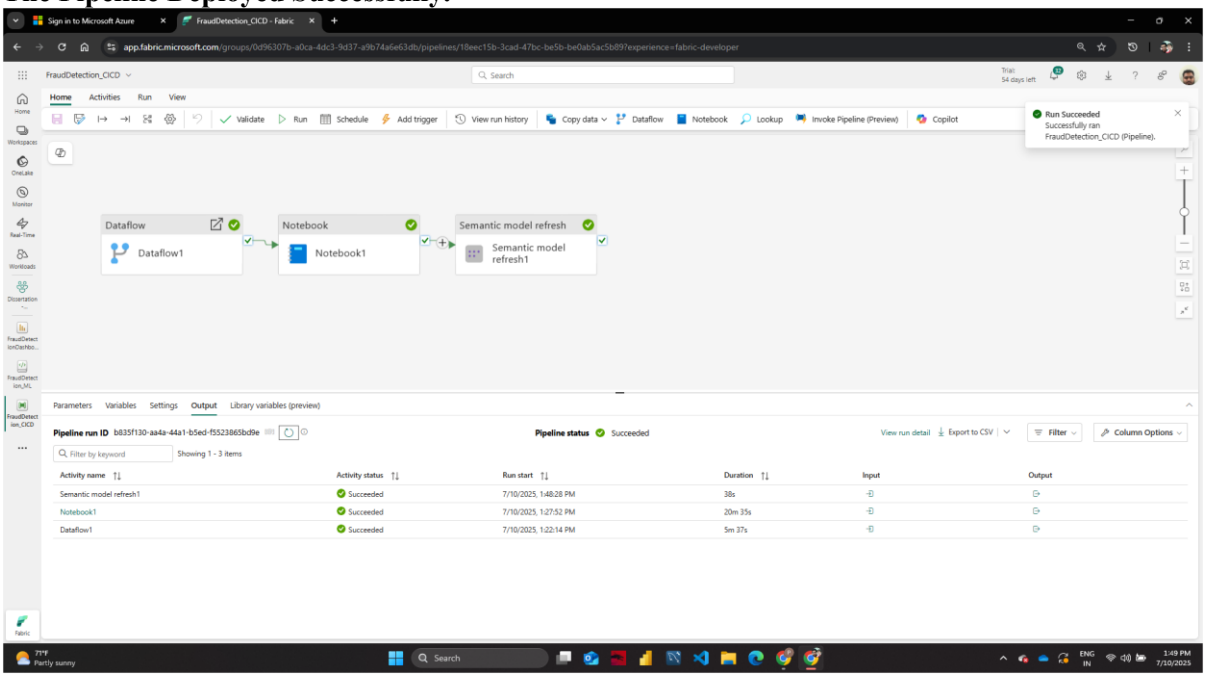


Figure 34. Pipeline Deployed Successfully.

7. End -to-end connections in Microsoft Fabric Workspace after Pipeline Deployment.

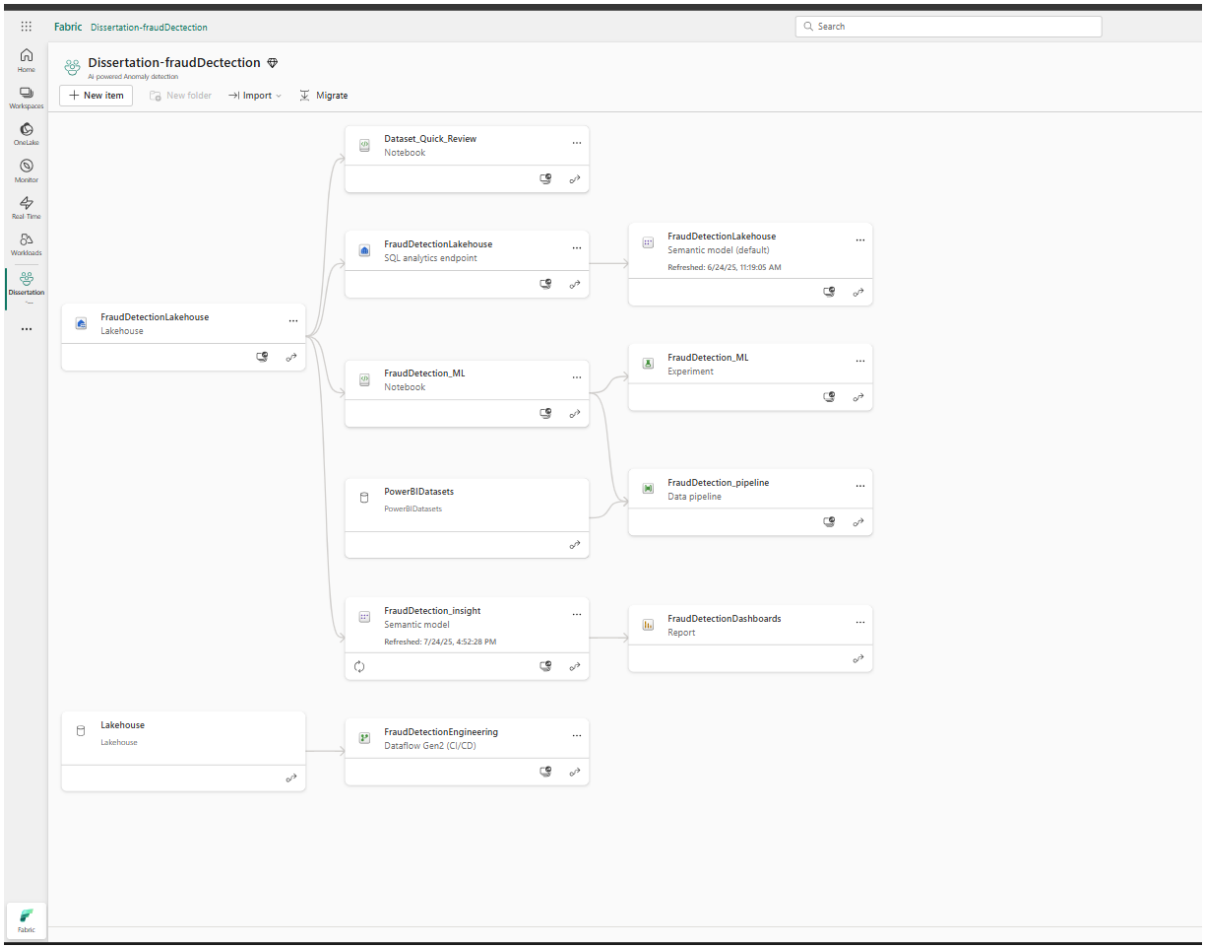


Figure 34. End-to-end Pipeline

References

1. Bradlay, A. and Schacht, B. (2024). Learn Microsoft Fabric. Packt Publishing, p.338.
2. Borra, P. (2024). Microsoft Fabric Review: Exploring Microsoft's New Data Analytics Platform. Zenodo. [online] doi:https://doi.org/10.5281/zenodo.11502585.
3. Ghosh, D. (2024). Microsoft Fabric: inside and out. Apress eBooks, pp.319–350. doi:https://doi.org/10.1007/979-8-8688-0131-0_10.
4. Plašić, J., Gaborović, A. and Stefanović, N. (2025). Unified Business Intelligence in The Microsoft Fabric Environment.Zornik Radova, [online] pp.407–415. doi:https://doi.org/10.46793/ebm24.407p.