**Big Data Application, Fall 2024**

**Mini Project - Jeet Rakesh Patel (**[**patejeet@iu.edu**](mailto:patejeet@iu.edu)**)**

**Introduction:**

In today’s digital economy, financial transactions generate vast amounts of data that organizations can leverage to uncover insights, forecast trends, and make data-driven decisions. This project focuses on analyzing transaction-level data to extract meaningful patterns, perform aggregations, and build predictive models using AWS services. By integrating big data processing tools like PySpark and machine learning capabilities of AWS SageMaker Autopilot, the project aims to showcase an end-to-end data pipeline for handling large-scale financial datasets.

The dataset, *preprocessed\_transactions*.*csv*, contains detailed customer transaction records. It includes **Transaction Amount, Customer Demographics, Transaction Date and Time, and Customer Account Balance**.

**Dataset Overview**

The dataset contains the following key attributes:

* TransactionID: Unique identifier for each transaction.
* CustomerID: Unique identifier for each customer.
* CustomerDOB: Date of birth of the customer.
* CustGender: Gender of the customer (e.g., Male, Female).
* CustLocation: Region or location where the transaction took place.
* CustAccountBalance: Account balance of the customer before the transaction.
* TransactionDate: Date on which the transaction occurred.
* TransactionTime: Time at which the transaction took place.
* TransactionAmount (INR): Amount of money involved in the transaction.
* Year: Year extracted from the transaction date.
* Month: Month extracted from the transaction date.

The dataset has been preprocessed to include additional attributes like Year and Month derived from the transaction date, enabling temporal analysis and trend identification.

The primary goal of this project is to process, analyze, and predict key insights from the transaction data. The project demonstrates using big data tools and cloud services to handle large datasets efficiently and gain actionable insights. The findings can be applied to understand customer spending behavior, forecast revenue trends, and identify high-value customers, ultimately supporting improved financial decision-making.

**Methodology :**

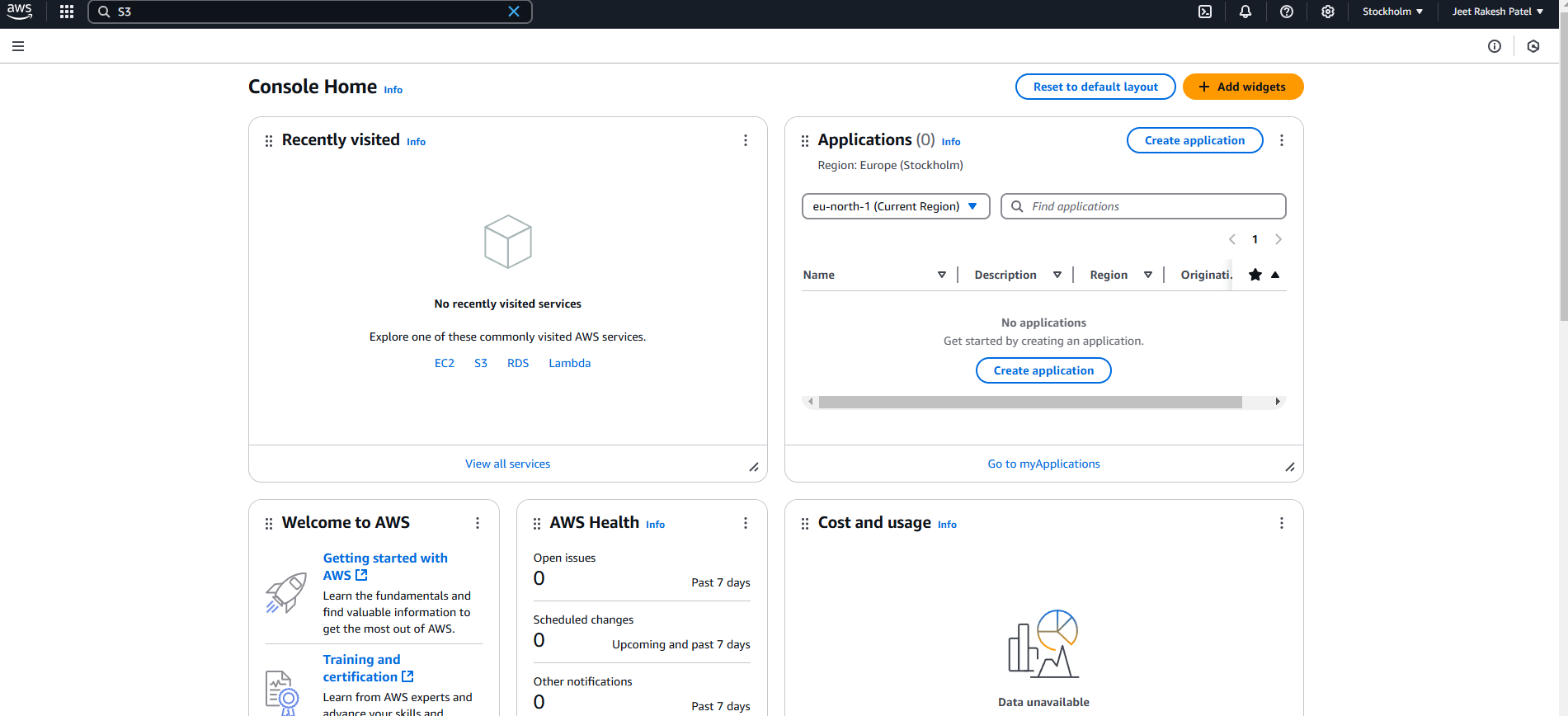
**1. Dataset Selection**

For this project, the chosen dataset is financial\_transactions.csv, which contains detailed financial transaction records. The dataset is large enough to derive meaningful insights and includes diverse columns such as Transaction Amount, Customer Demographics, Transaction Date and Time, and Account Balance. These features allow for comprehensive data cleaning, transformation, and aggregation, enabling tasks like revenue analysis, customer behavior insights, and predictive modeling. The inclusion of both categorical and numerical variables provides opportunities for advanced analysis and machine learning applications.

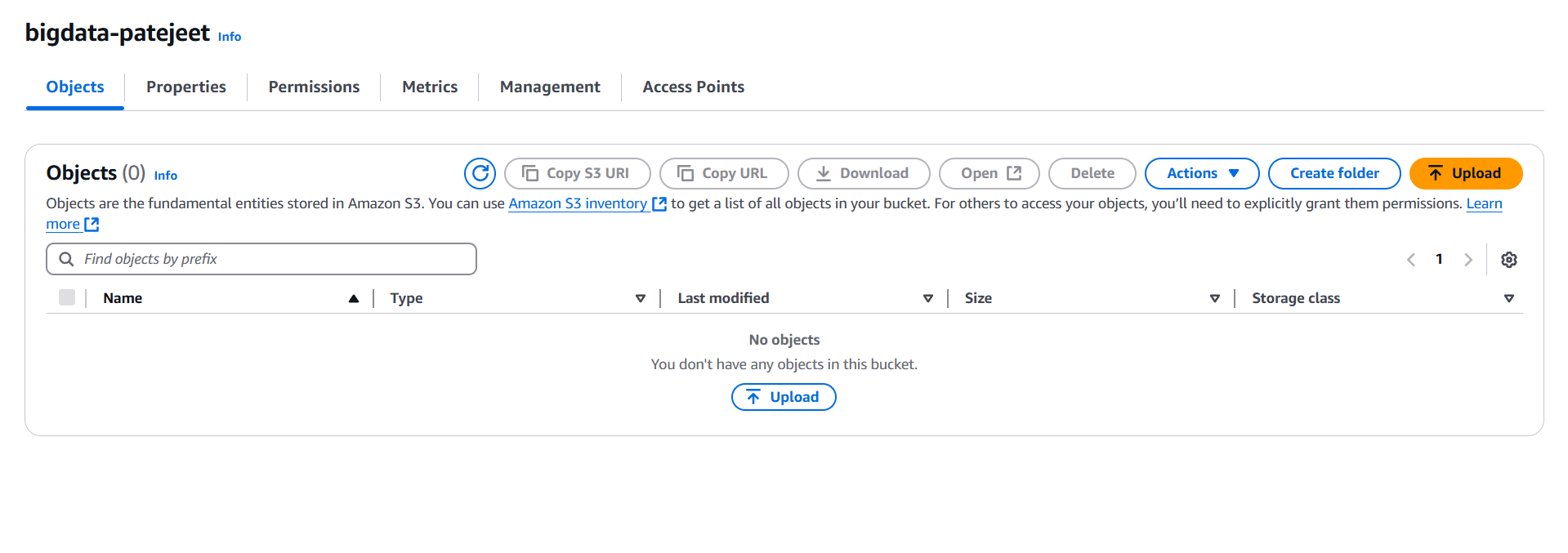
**2. Environment Setup**

1. AWS S3 for Data Storage

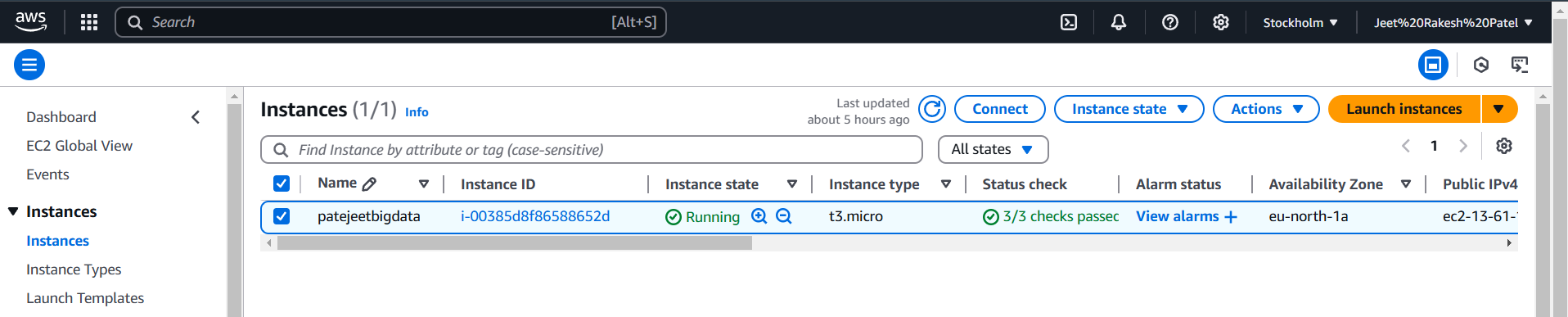
Step 1: Create an S3 bucket to store both raw and processed data



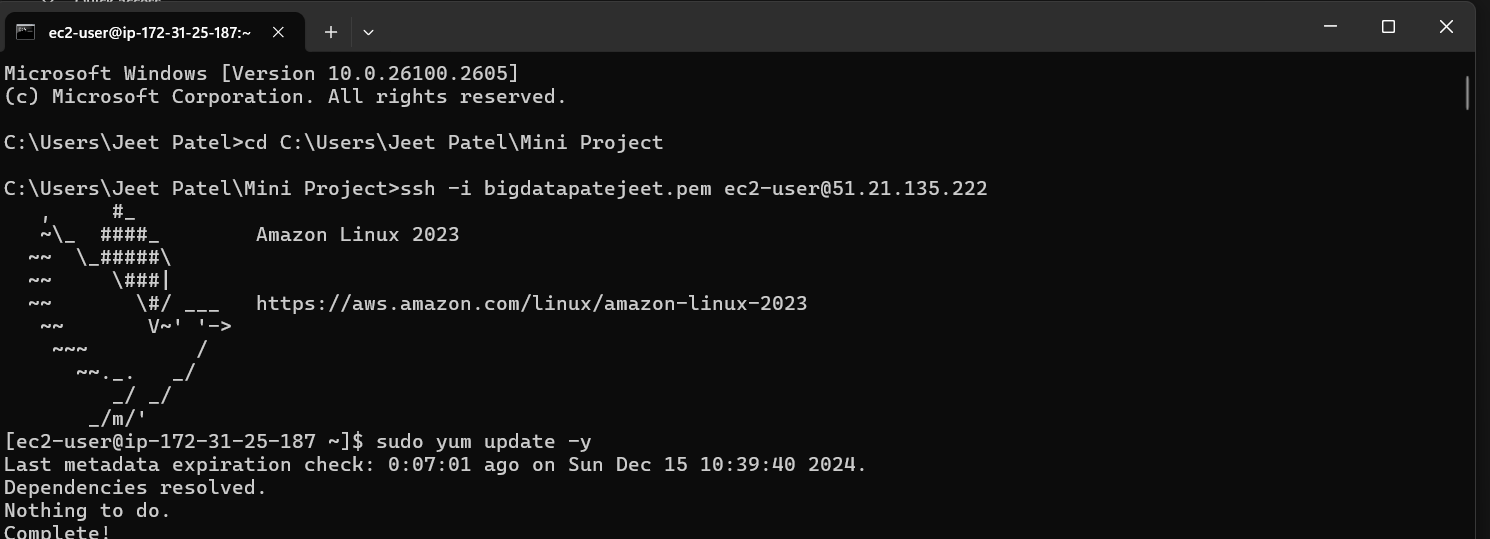
Step 2: Upload the raw dataset to the S3 bucket.



2. Linux Environment with PySpark

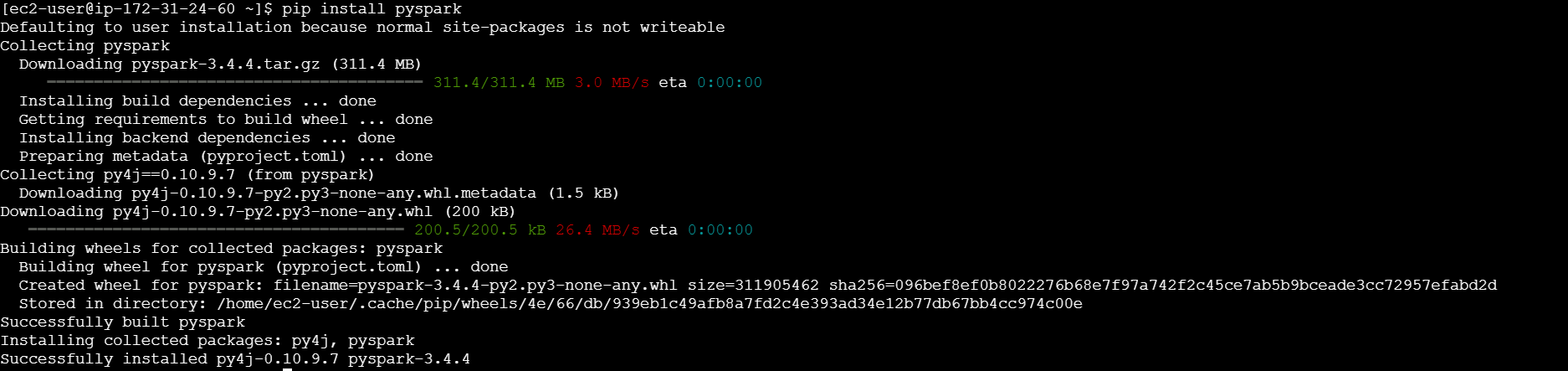


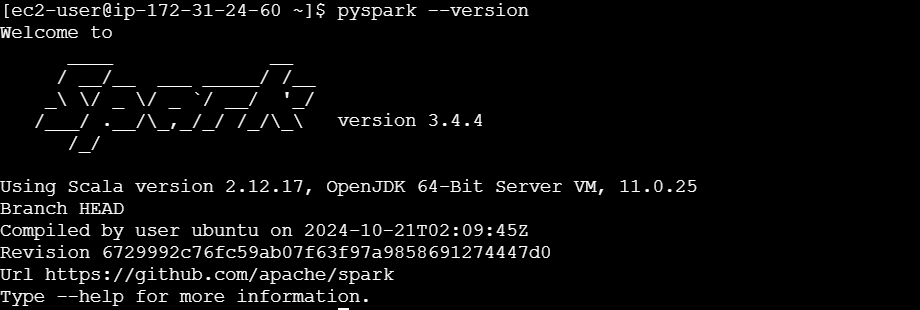
Step 1: Set up a Linux-based environment, either locally or using an AWS EC2 instance.



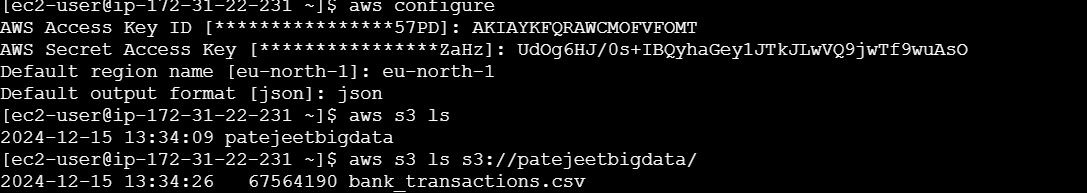
Recommendation: Use an AWS EC2 instance for better scalability and AWS integration.

Step 2: Install PySpark for distributed data processing.





Step 3: Configure AWS CLI to interact with S3 buckets.



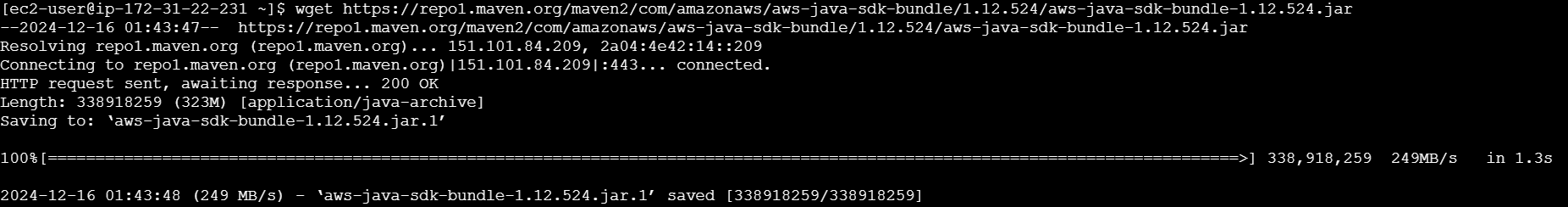
**3. Data Pipeline Tasks**

Task 1: Data Ingestion from S3

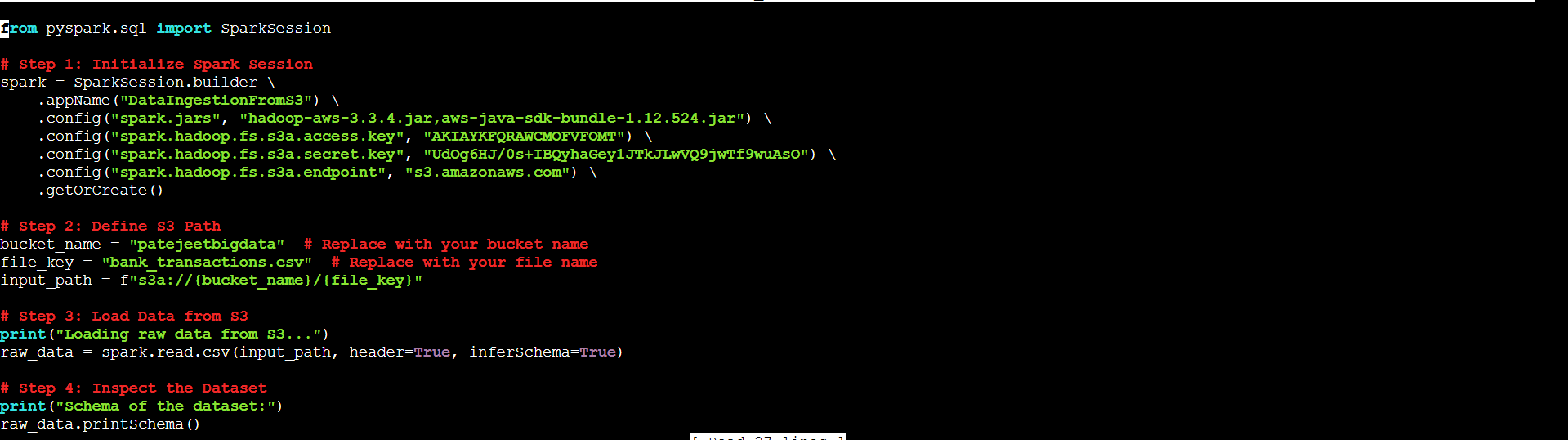
● Objective: Pull raw data from S3 into the PySpark environment.

● Steps: 1. Use AWS CLI or PySpark’s built-in S3 support to load the dataset directly.

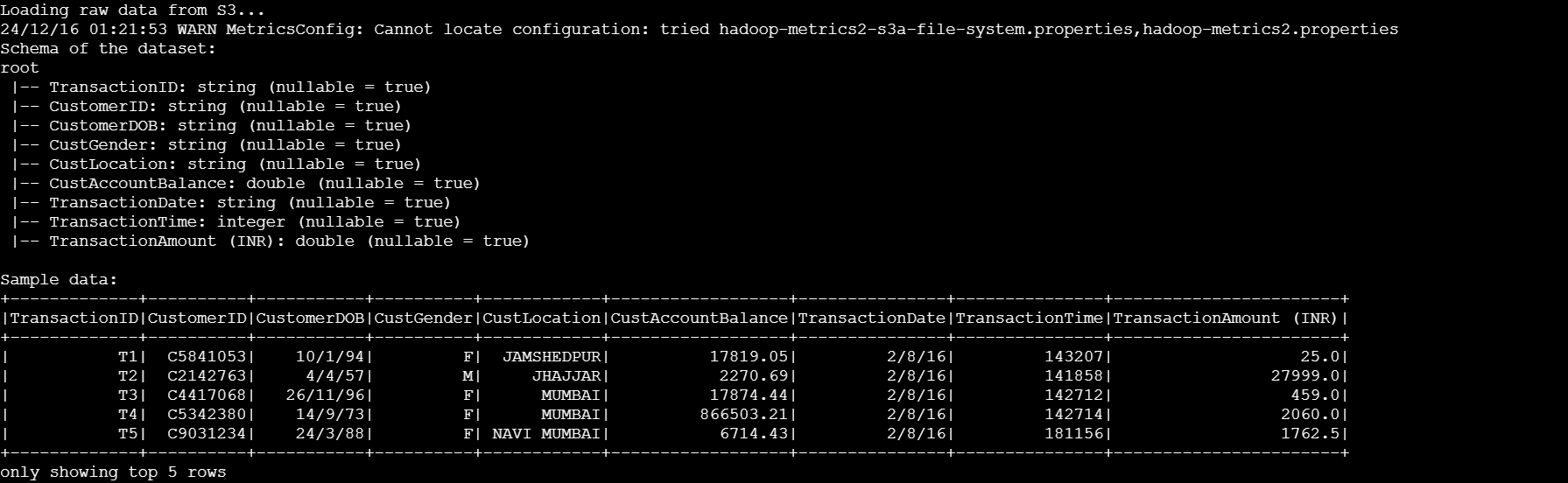
****

****

**Code for Data ingestion**

****

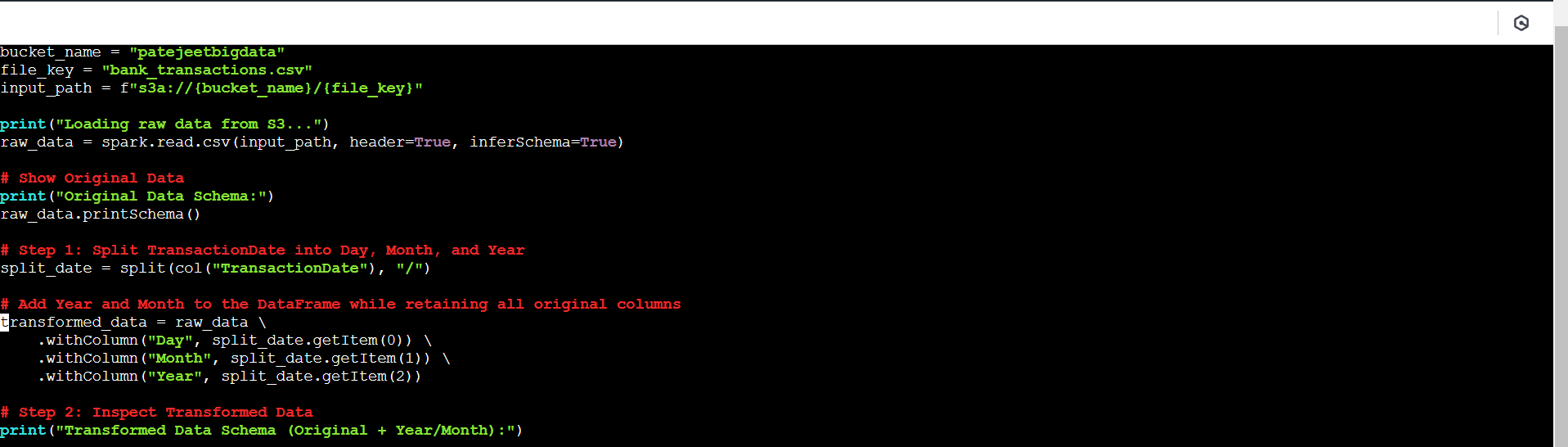
2. Confirm successful ingestion by inspecting the dataset.

****

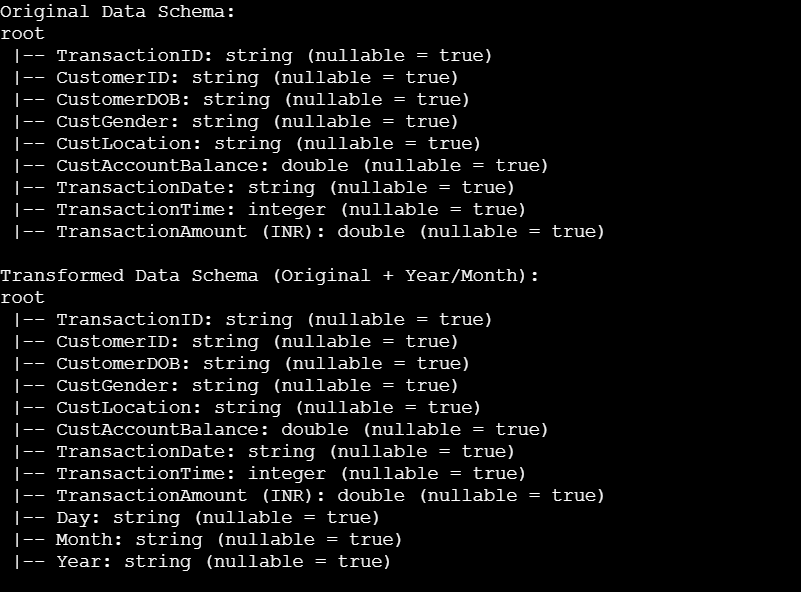
**Task 2: Data Processing with PySpark**

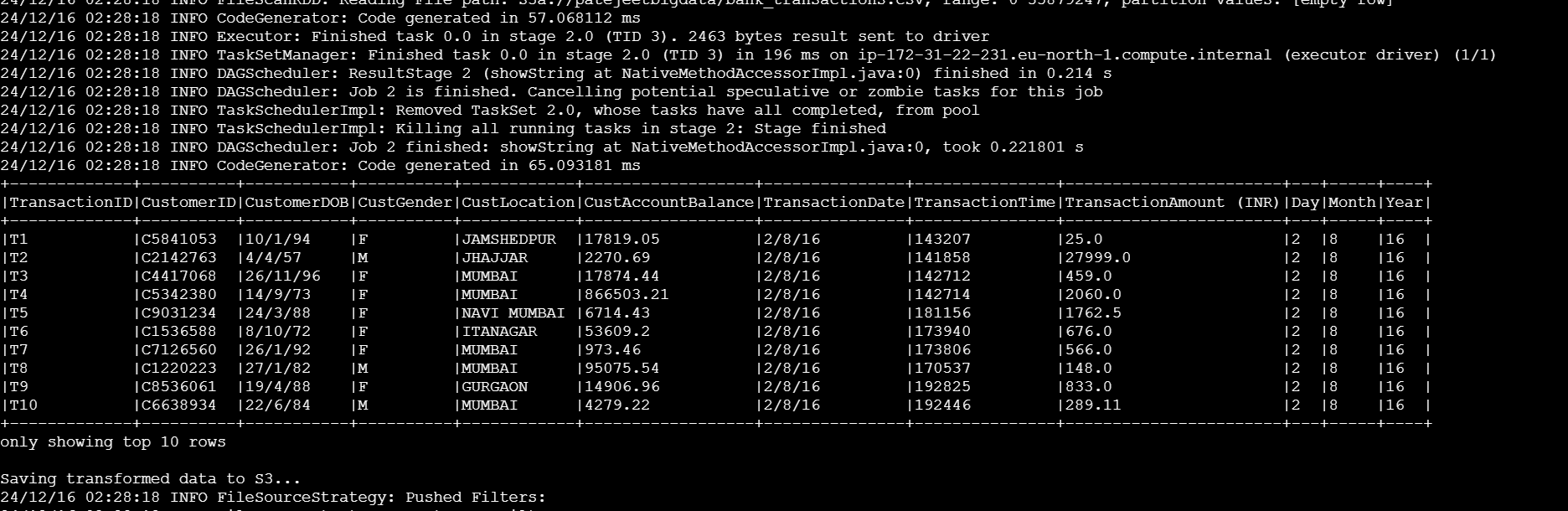
**DATA TRANSFORMATION**

**Create at least 2 new columns (e.g., Year , Month ) to aid in analysis.**

****

**Whole code is uploaded in zip file**





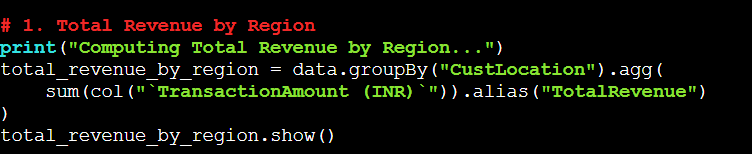
In this task, I created two new columns, Year and Month, by extracting information from the TransactionDate column. This transformation was chosen because analyzing data at a temporal level is critical for identifying trends over time, such as seasonal spending patterns and monthly or yearly revenue growth. By breaking down the date into specific components, it enables more granular analysis, facilitates time-based aggregations, and supports better decision-making, such as tracking monthly transaction trends or annual performance insights.

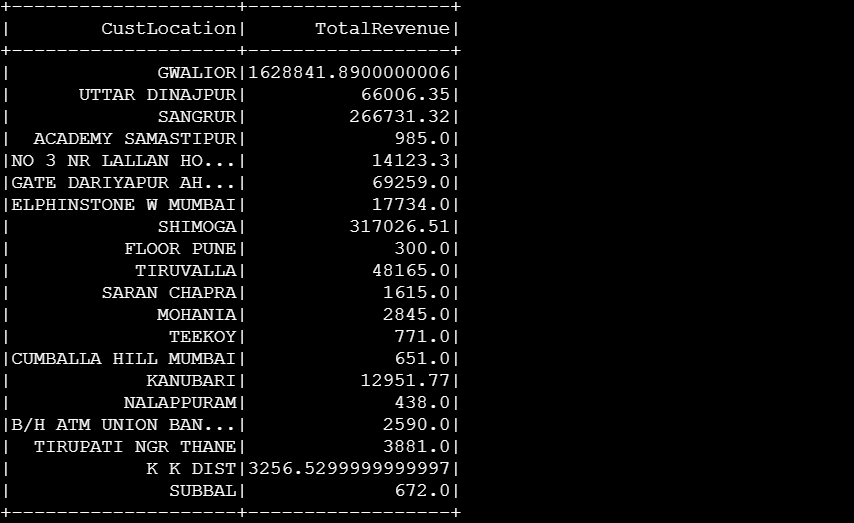
**Task 3: Data Aggregation**

Compute at least 5 key metrics, such as Total revenue by region, Monthly spending trends, Top 10 customers by transaction value (These metrics may vary depending on the specifics of your dataset, but the goal is to aggregate the data in ways that enable meaningful analysis and decision-making.)

**Total Revenue by Region**

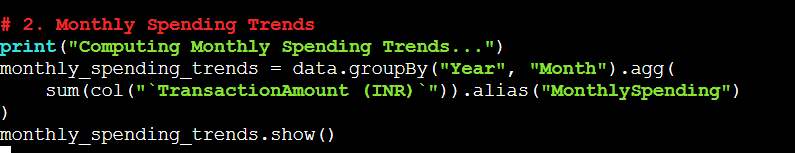
This metric helps identify the regions contributing the most revenue, enabling businesses to focus on high-performing areas and devise strategies for underperforming regions.

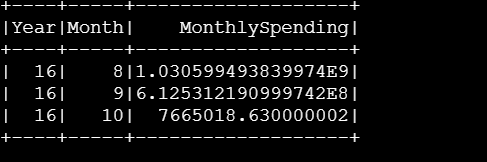
****

****

**Monthly Spend trend**

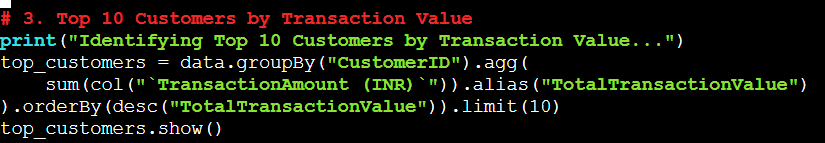
Analyzing monthly spending trends allows for identifying seasonal patterns and variations in customer behavior, which is crucial for forecasting and planning.

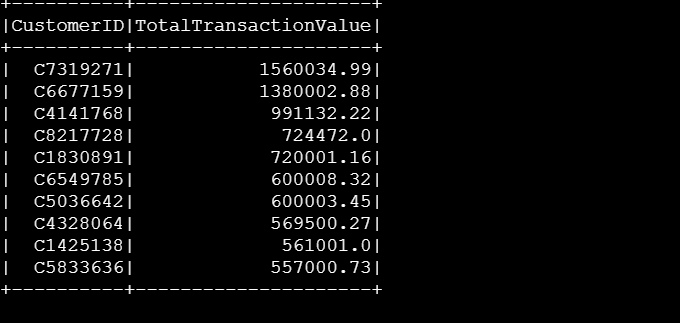
****

****

**Top 10 Customers by Transaction Value:**

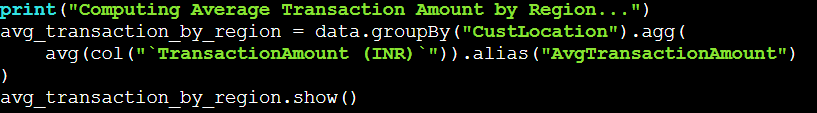
Determining the top customers by transaction value helps recognize high-value customers, allowing businesses to prioritize them for retention and personalized offers.

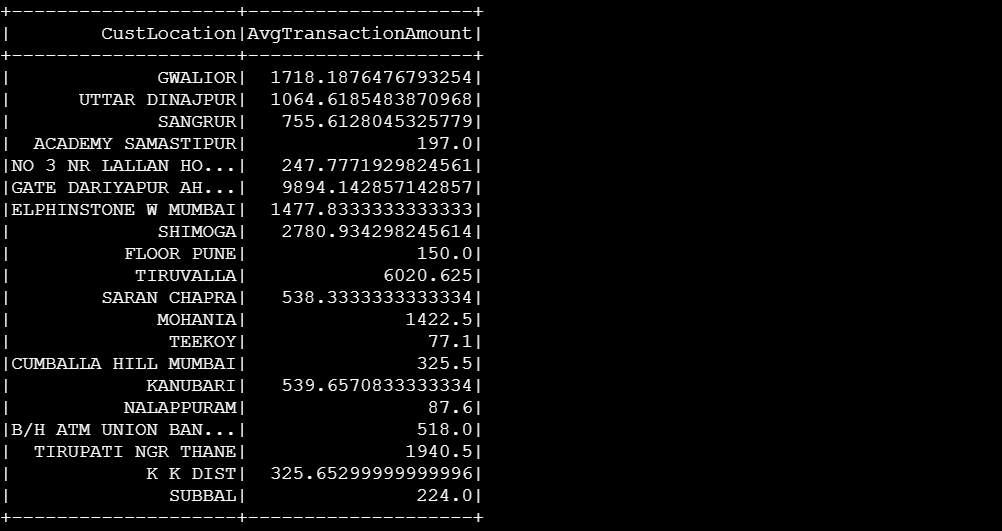
****

****

**Average Transaction Amount by Region:**

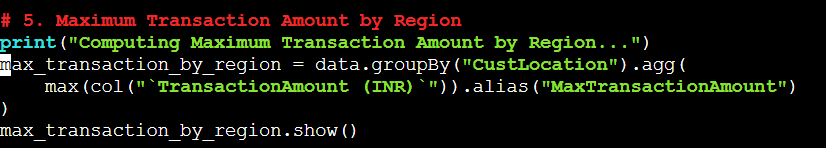
Calculating the average transaction amount by region provides insights into customer spending behavior across different areas, helping understand regional economic differences.

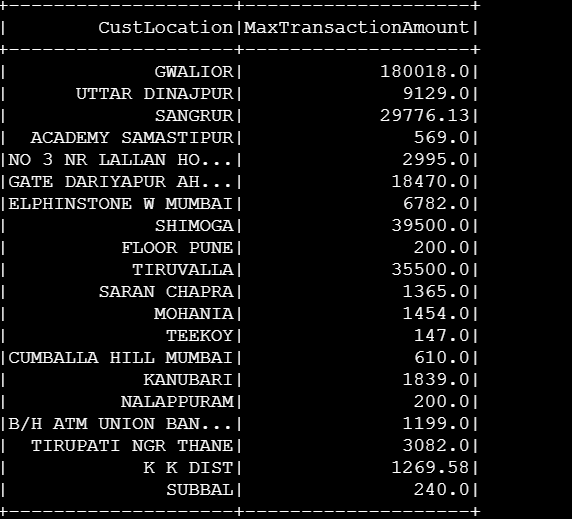
****

****

**Maximum Transaction Amount by Region:**

Identifying the maximum transaction amount in each region highlights significant transactions, which can uncover outliers or exceptional sales opportunities for further analysis.

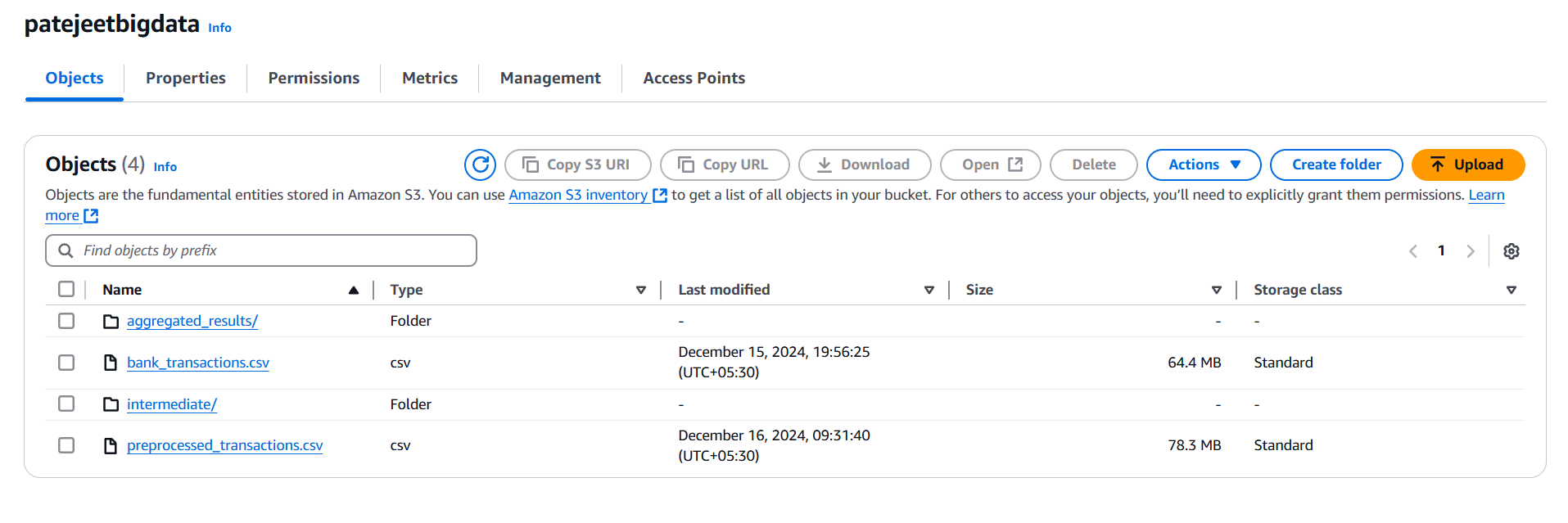
****

****

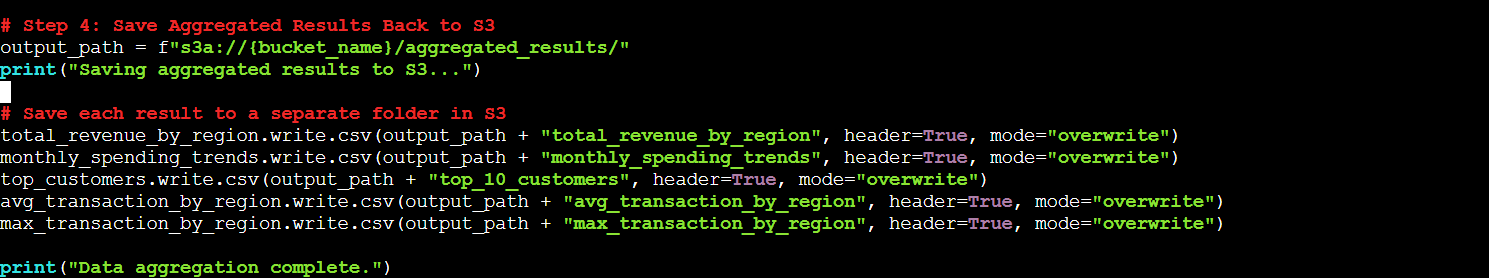
**Task 4: Store Processed Data back in S3**

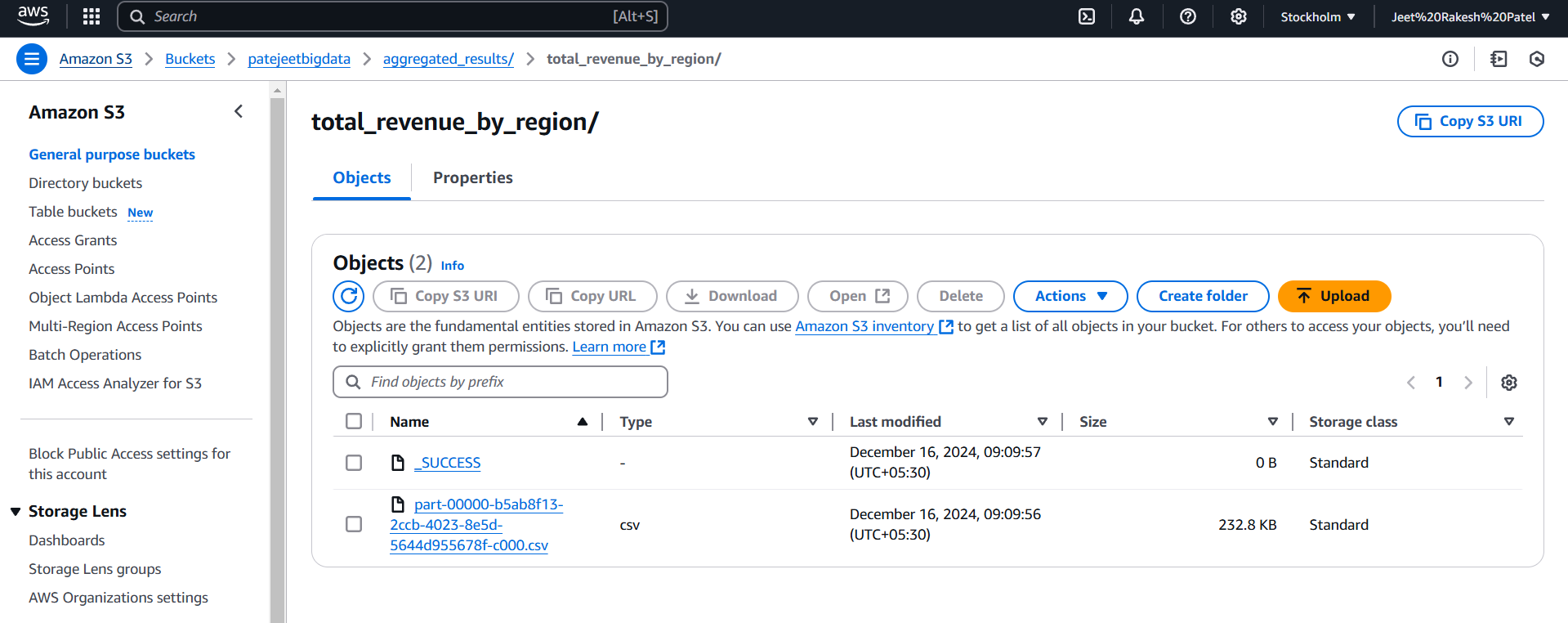
**Storing data back in the S3 code was already written in previous steps**

**Processed Data:**



***Aggregated Data:***

******

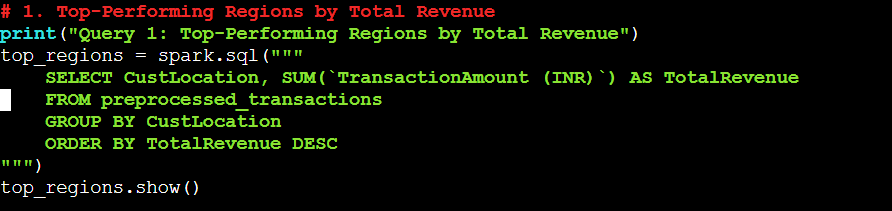
******

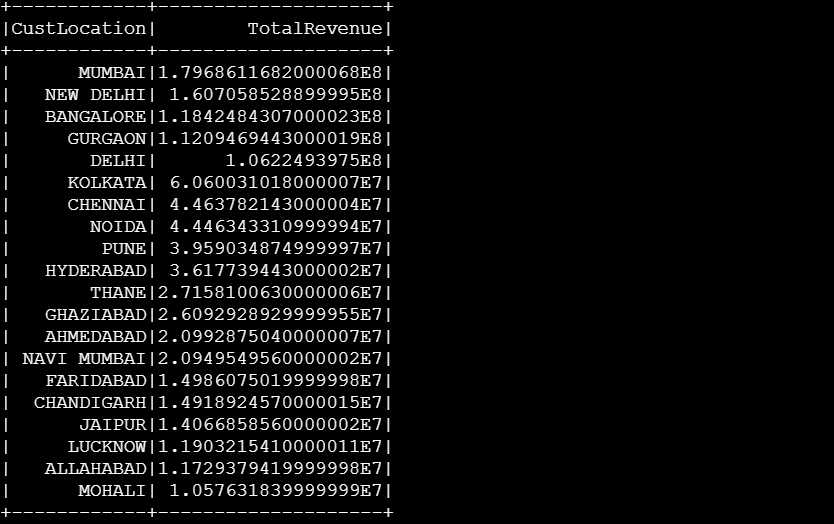
**Task 4:Data Analysis Using Spark SQL**

**● Objective: Use SQL to derive insights (Atleast 5 Queries).**

**Top Performing Regions by Total Revenue**

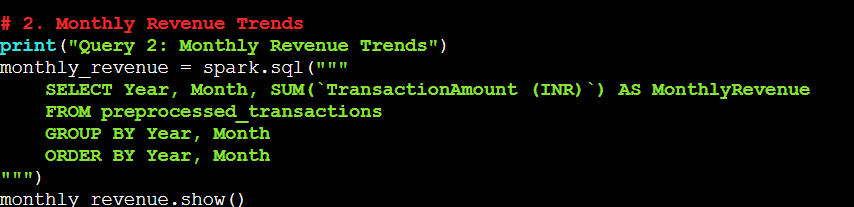
This query helps identify the regions that contribute the highest revenue, enabling businesses to focus their efforts on high-performing areas and analyze the factors driving their success.

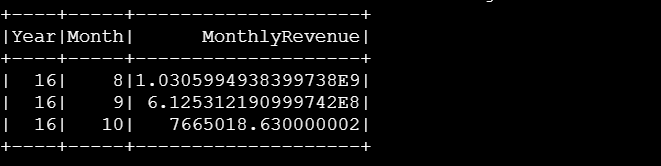
****

****

**Monthly Revenue Trends**

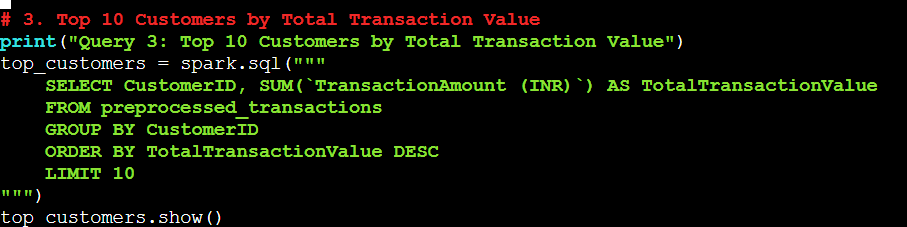
Analyzing monthly revenue trends provides insights into seasonal variations and spending behavior, allowing businesses to plan marketing strategies and resource allocation accordingly.

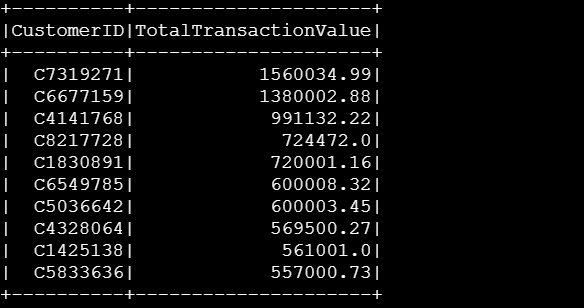
****

****

**Top 10 Customers by Total Transaction Value**

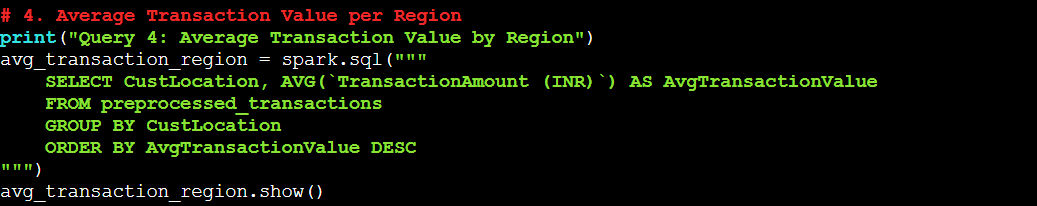
Identifying the top customers based on their total transaction value allows businesses to prioritize high-value customers for retention, rewards, and personalized services.

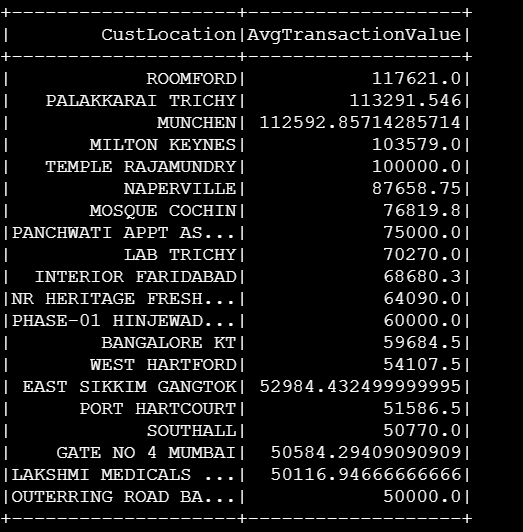
****

****

**Average Transaction Value per Region**

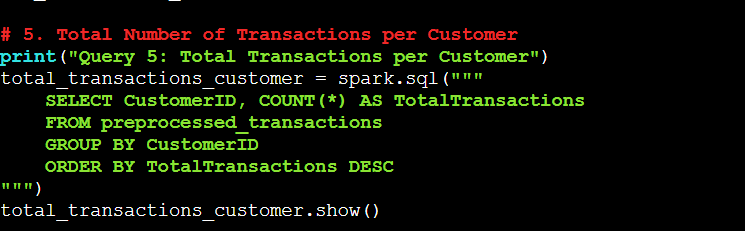
Calculating the average transaction value per region helps in understanding the spending capacity and behavior of customers in different regions, which is valuable for regional strategy development.

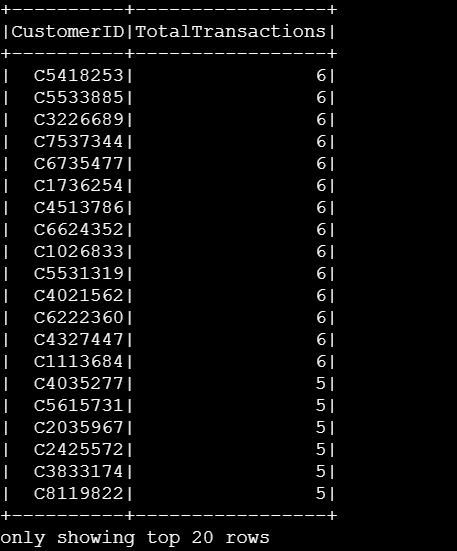
****

****

**Total Number of Transactions per customer**

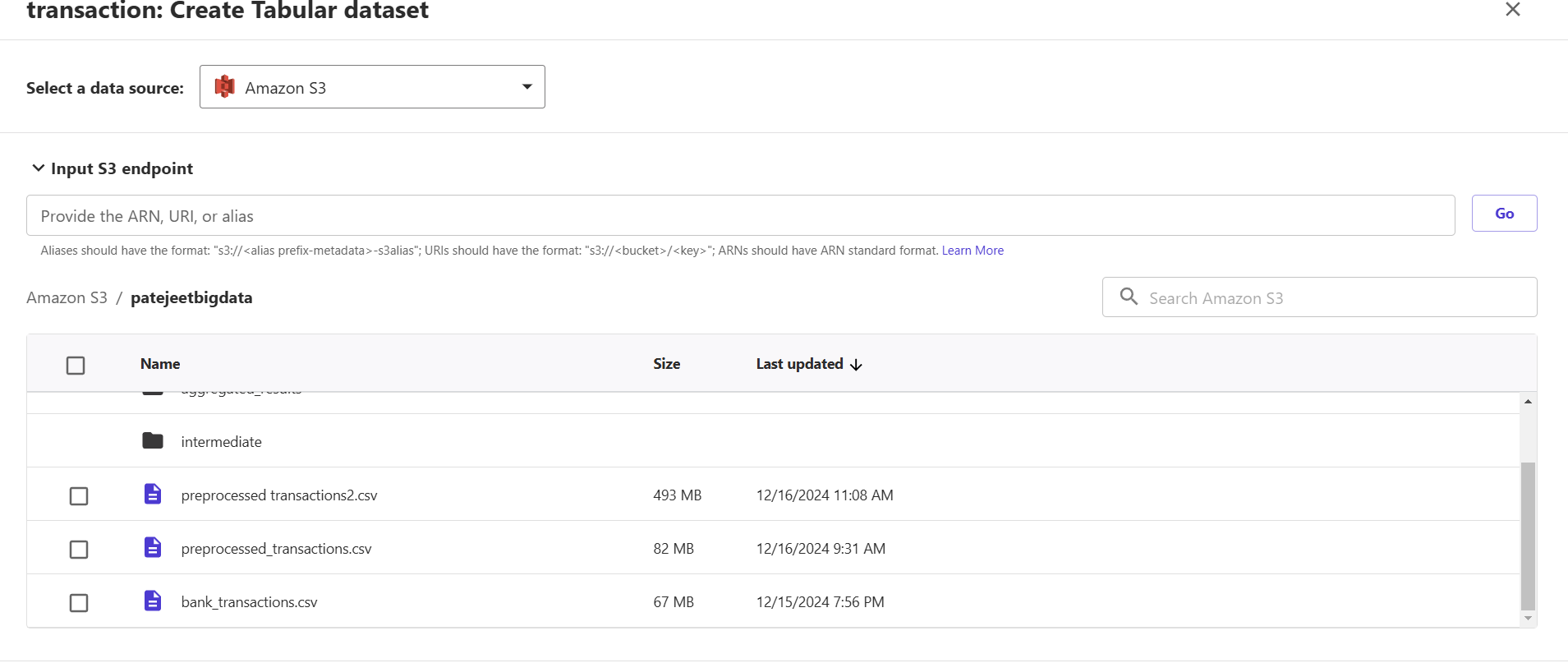
This query reveals customer engagement levels by calculating the number of transactions each customer has made, helping identify active customers and track their purchasing frequency.

****

****

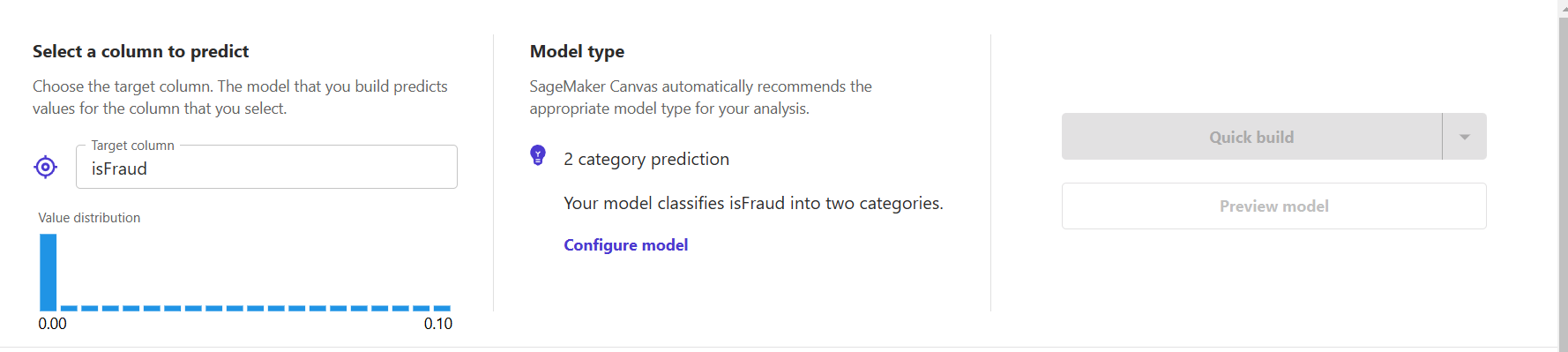
**Task 5: Machine Learning with AWS SageMaker Autopilot**

**1. Import Processed Data: Load the processed dataset from S3 into SageMaker Autopilot.**

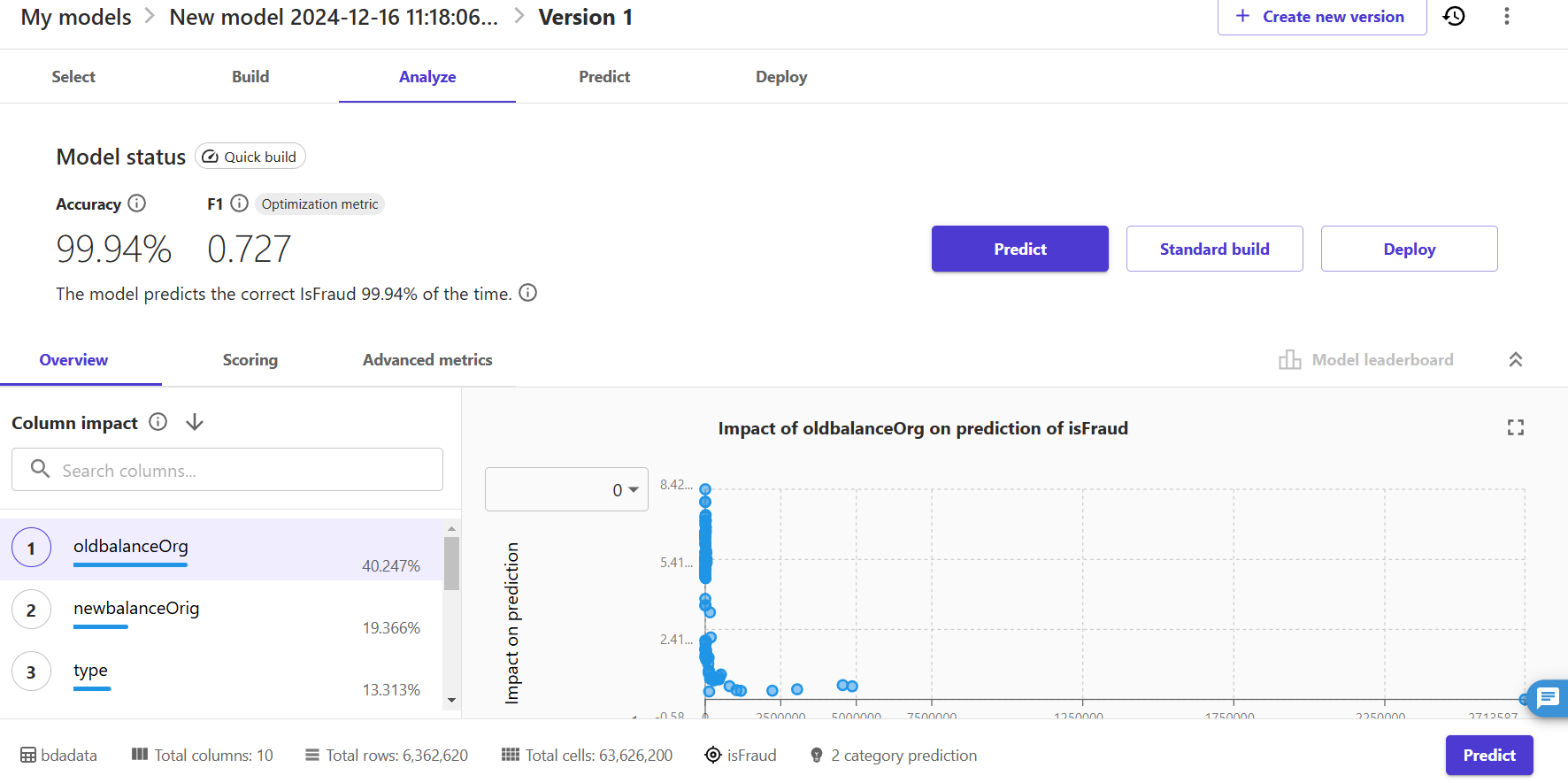
****

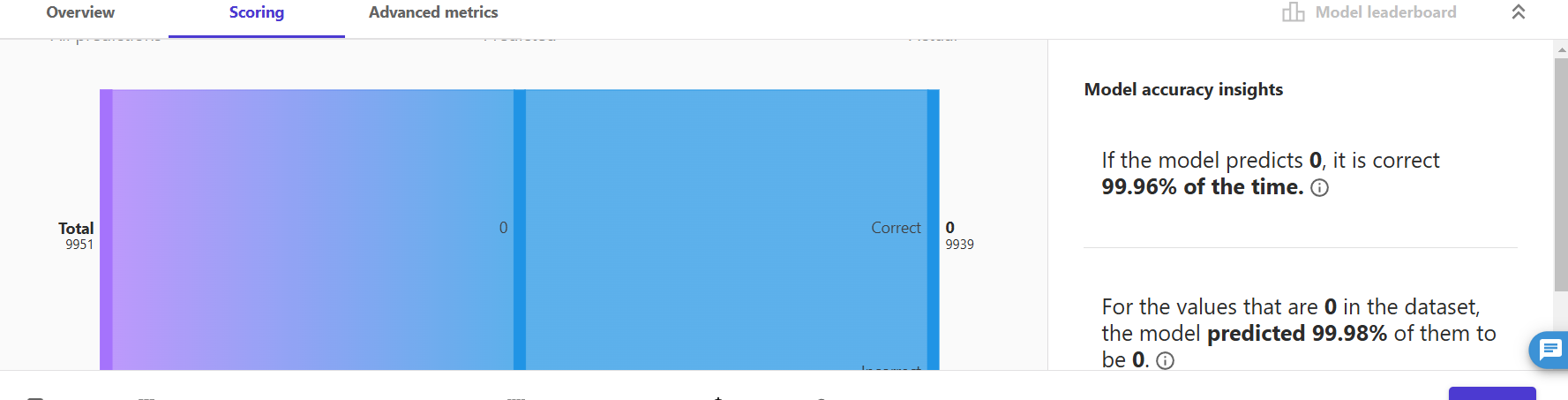
**2. Run Autopilot Experiment:**

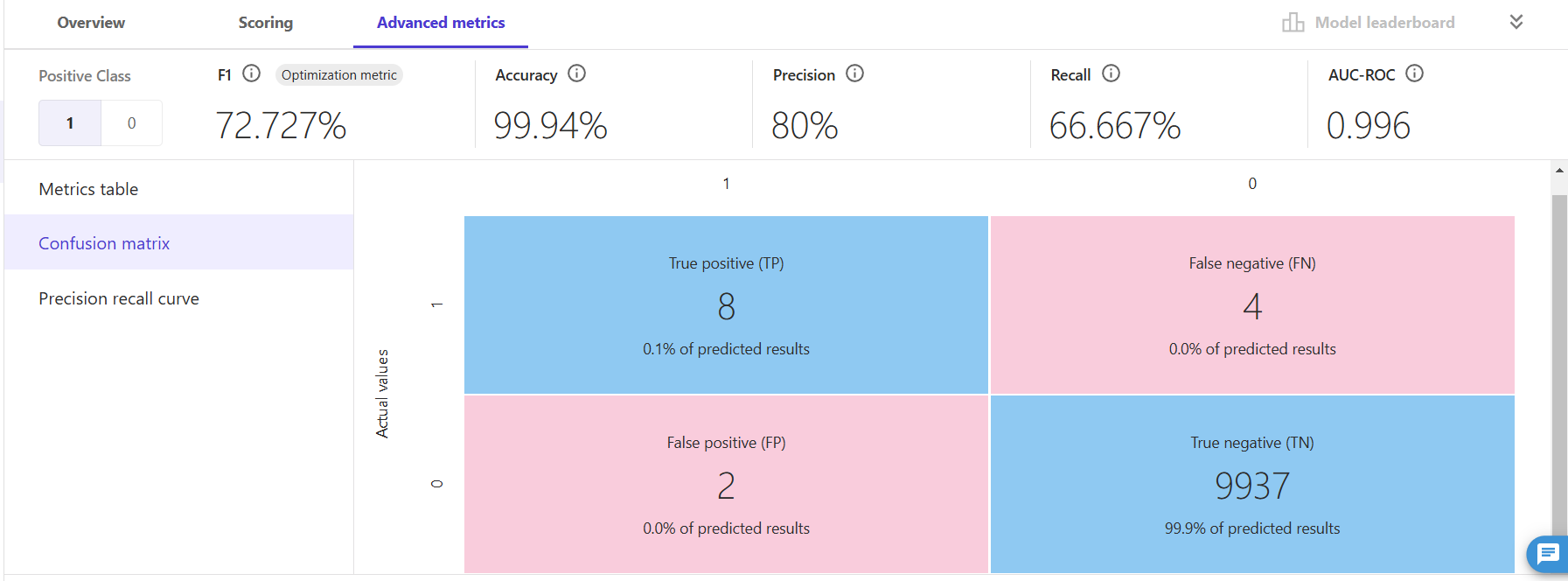
**Select target variable**

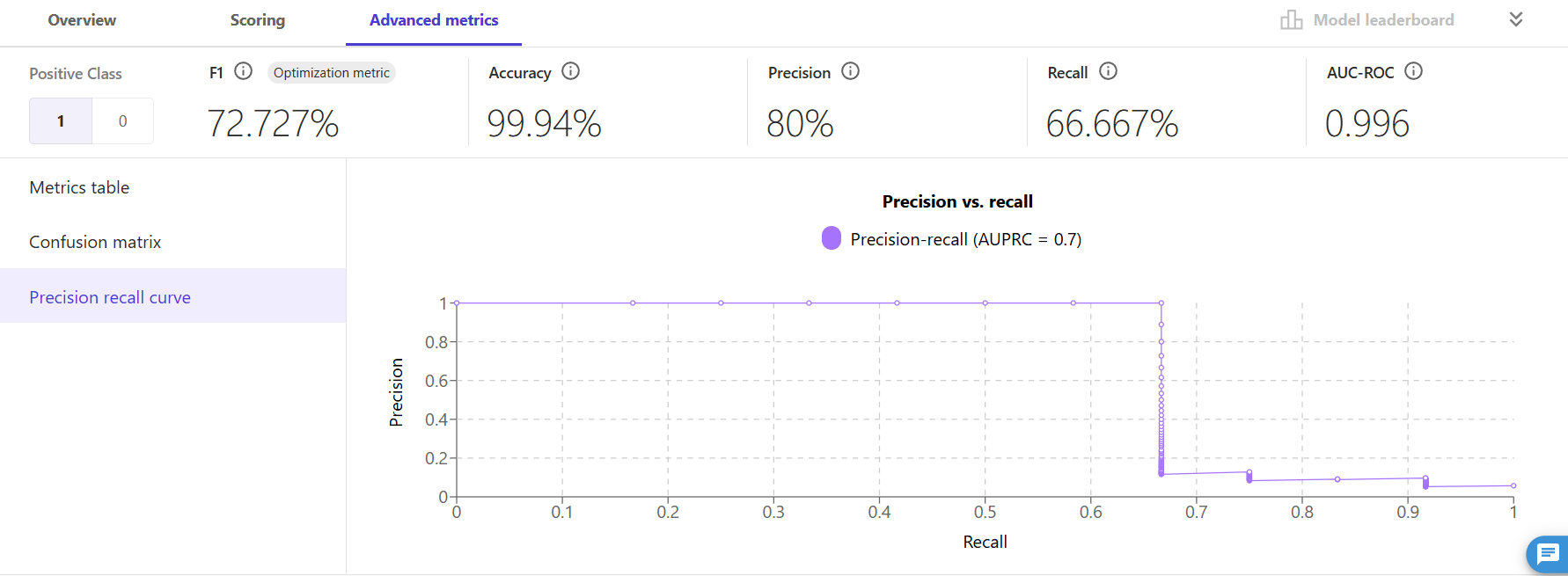
****

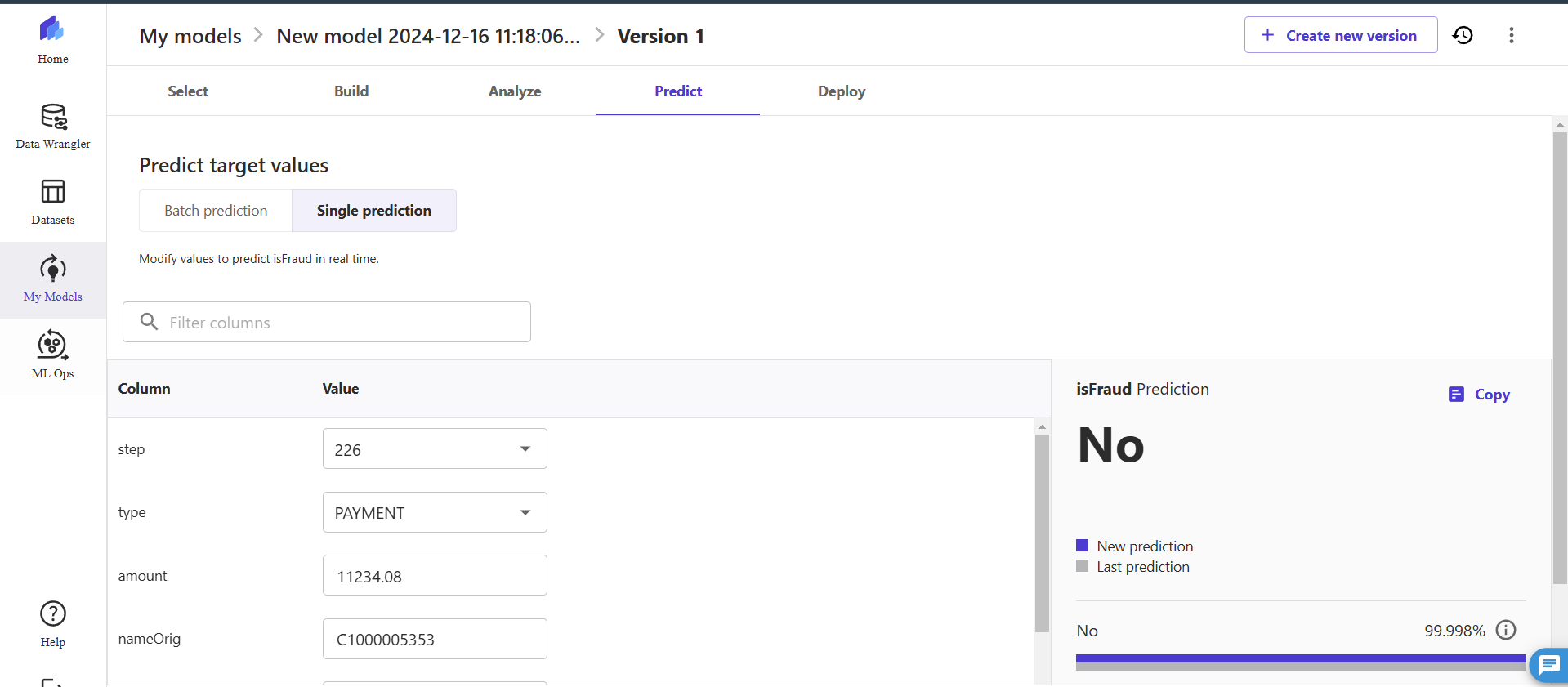
**Run the AutoML process to train and evaluate multiple models.**

****

****

****

****

****

**1. Model Overview**

The trained model was evaluated using several performance metrics, including accuracy, precision, recall, F1 score, and AUC-ROC. The overall model performance indicates a high level of accuracy with some variations in other metrics.

**2. Key Metrics**

**Accuracy (99.94%):**

The model correctly predicted the outcome 99.94% of the time.

This is a very high accuracy score, which suggests the model is able to predict the majority class effectively.

**F1 Score (72.7%):**

The F1 score balances precision and recall. A score of 72.7% indicates the model is fairly capable of detecting the positive class (IsFraud=1), although there is some room for improvement.

**Precision (80%):**

Precision reflects how many of the predicted positives (fraudulent transactions) were actually correct.

A precision of 80% means that 80% of the transactions predicted as fraud were indeed fraudulent.

**Recall (66.67%):**

Recall measures the model’s ability to identify all fraudulent transactions (positive cases).

A score of 66.67% indicates that the model is missing some fraudulent cases, which needs further optimization.

**AUC-ROC (0.996):**

The AUC-ROC score indicates the model’s ability to distinguish between the positive and negative classes.

A score of 0.996 is excellent, as it suggests that the model is highly capable of differentiating between fraudulent and non-fraudulent transactions.

**3. Model Insights**

The Predicted vs. Actual chart shows that the model performs exceptionally well in predicting the majority class (0 - non-fraudulent transactions).

It correctly predicted 99.96% of the 0 class instances, while for 1 (fraudulent transactions), there were 12 incorrect predictions out of 9951 total samples.

This imbalance is typical in fraud detection problems where fraudulent cases are rare, leading to class imbalance issues.

**Ethical Issues and Considerations**

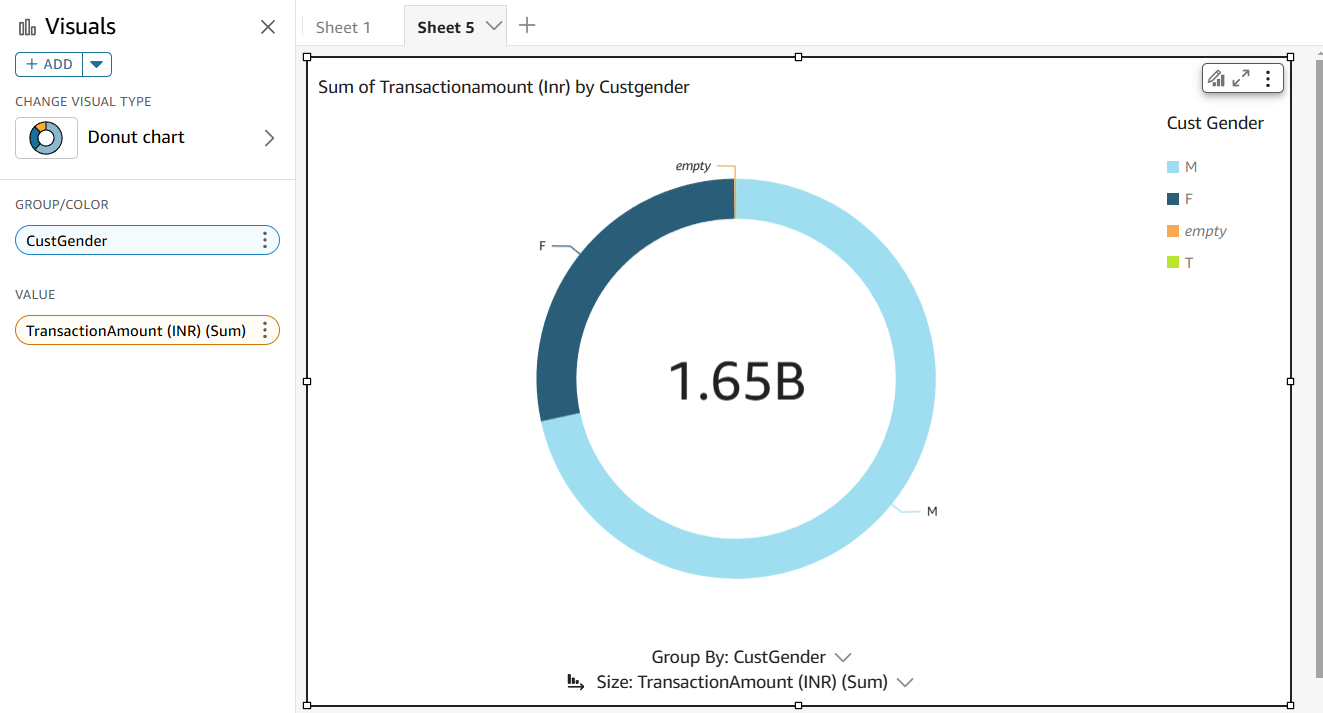
One of the major ethical issues in this project is bias in training data. When the training dataset contains a significant imbalance between fraudulent and non-fraudulent transactions (with the majority class being non-fraudulent), the model may become biased toward predicting the majority class (`0`). This can result in under-prediction of fraudulent transactions (`1`), leading to false negatives where actual fraud goes undetected. To address this, techniques such as class balancing using oversampling (e.g., SMOTE) or undersampling can be applied to improve the recall for the minority class.

Another critical concern is privacy. Financial transaction data often includes sensitive information, such as customer IDs, transaction amounts, and locations, which raises significant privacy concerns. Unauthorized access to this data can lead to breaches of confidentiality and misuse of customer information. To mitigate these risks, it is essential to use encryption for data storage and access, anonymize personally identifiable information (PII), and strictly enforce data governance policies.

The issue of algorithmic fairness also arises, as the model’s predictions may unfairly disadvantage certain groups or regions if the dataset contains biases related to demographics or locations. Such biases can result in discriminatory predictions, leading to ethical issues and eroding customer trust. To ensure fairness, monitoring fairness metrics, regularly auditing the model, and ensuring that the dataset represents diverse groups fairly are necessary steps.

Lastly, model transparency is a concern, particularly with black-box models like those produced by AutoML tools. These models often lack explainability, making it difficult to understand why certain transactions are flagged as fraudulent. This can reduce trust in the model’s predictions, especially in critical use cases such as fraud detection. To address this, explainable AI (XAI) techniques should be utilized to provide insights into the model’s decisions and ensure accountability and transparency.

4. **Visualization**



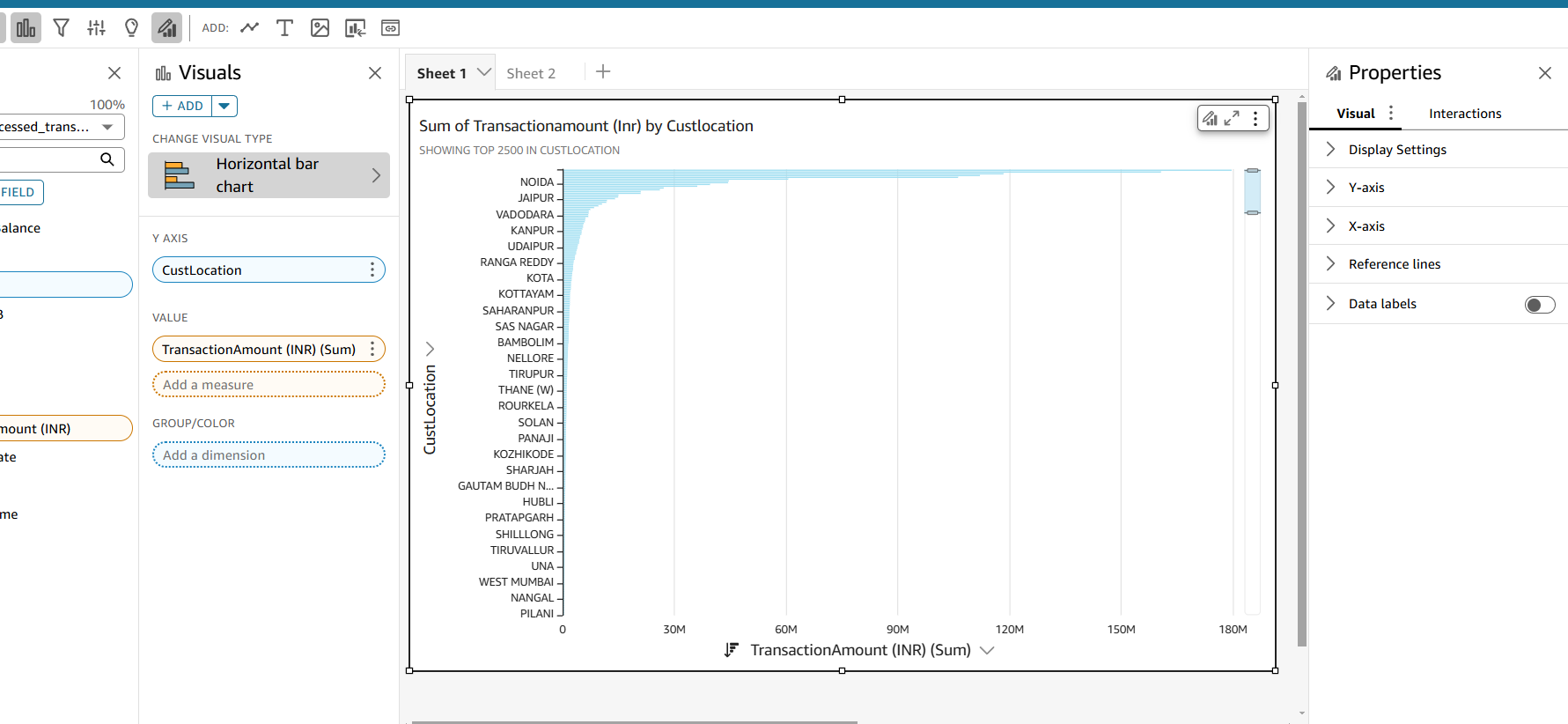
**Donut Chart: Sum of Transaction Amount by Customer Gender**

Description:

This donut chart visualizes the total transaction amount (INR) categorized by customer gender (CustGender). The majority of transactions are associated with the male (M) category, followed by female (F), and a small proportion of missing/empty values.

Insight:

This helps in understanding transaction patterns based on customer demographics, enabling targeted strategies for different gender groups.

****

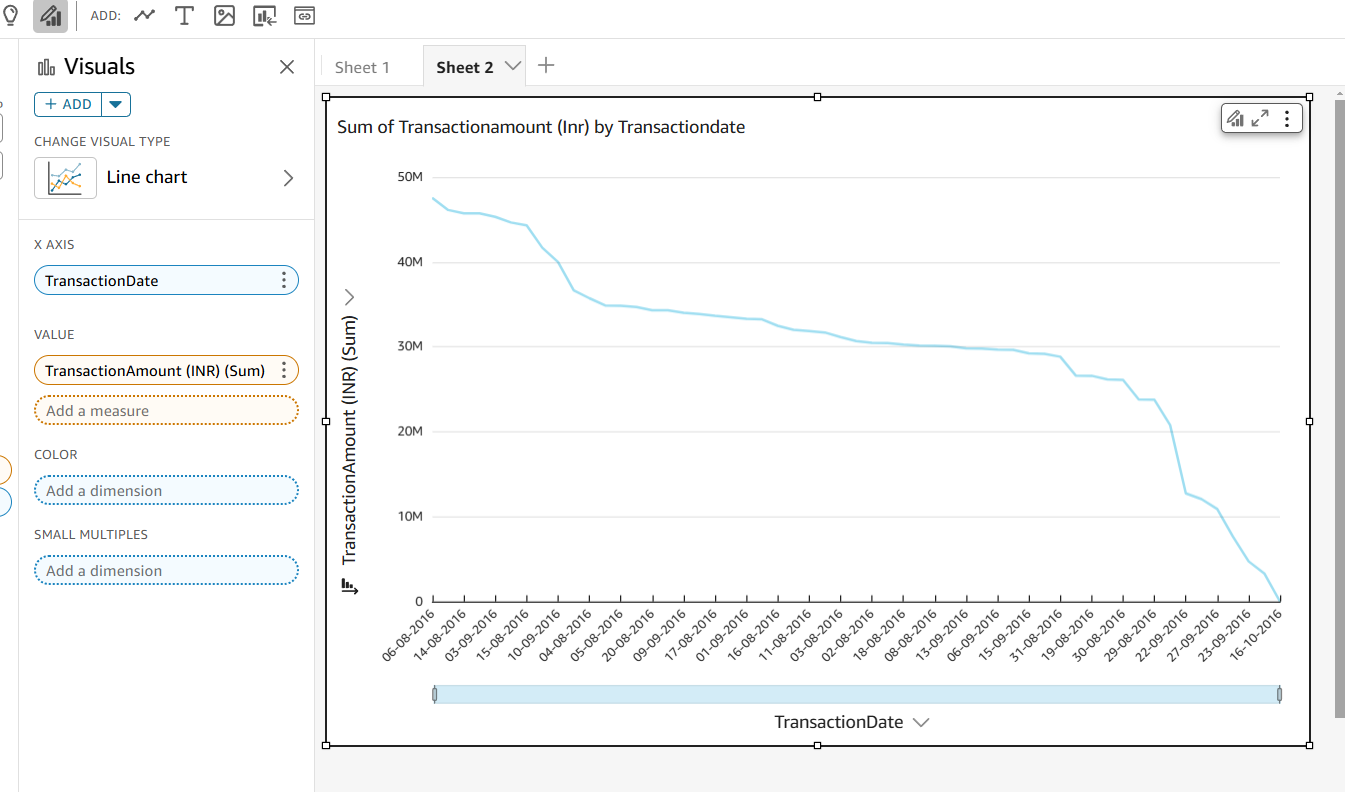
**Horizontal Bar Chart: Sum of Transaction Amount by Customer Location**

Description:

This horizontal bar chart showcases the total transaction amount (INR) aggregated by customer location (CustLocation). It highlights regions contributing the most to the total revenue, with the data sorted in descending order.

Insight:

This visualization identifies top-performing regions in terms of transaction value, which is valuable for regional performance analysis and market focus.

****

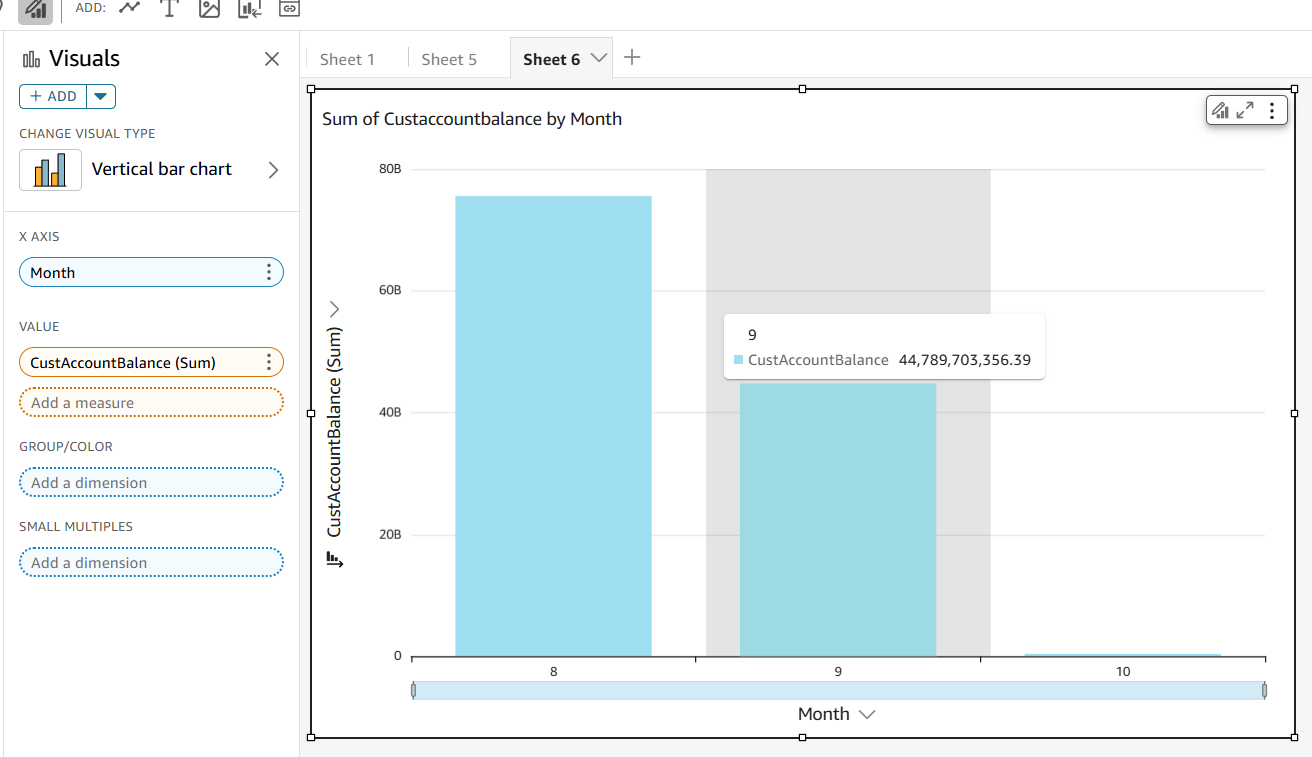
**Line Chart: Sum of Transaction Amount Over Time by Transaction Date**

Description:

This line chart displays the trend of total transaction amounts over time (TransactionDate). The x-axis represents the transaction dates, while the y-axis shows the sum of transaction values.

Insight:

The chart helps identify trends, such as revenue growth or decline over specific dates, and can highlight seasonal or periodic transaction patterns.

****

**Vertical Bar Chart: Sum of Customer Account Balance by Month**

Description:

This vertical bar chart visualizes the total customer account balance (CustAccountBalance) aggregated by month. Each bar corresponds to the sum of account balances in a specific month.

Insight:

This chart provides insights into the monthly account balances, which can help analyze spending behavior, customer financial health, and month-over-month trends.

**Conclusion:**

This project successfully analyzed financial transaction data to uncover meaningful insights, process the data efficiently, and build a robust fraud detection model. By leveraging PySpark for data processing and AWS tools like S3, SageMaker Autopilot, and QuickSight, we derived key metrics such as total revenue by region, monthly spending trends, and top customers by transaction value. Visualizations provided clear insights into customer behaviors, regional performance, and temporal spending patterns.

The machine learning model built using SageMaker Autopilot achieved 99.94% accuracy with an F1 score of 0.727, demonstrating its ability to identify fraudulent transactions effectively. Ethical concerns such as data bias, privacy, and fairness were addressed with mitigation strategies. This project highlights the potential of data-driven approaches to enhance decision-making and improve security in financial domains.

**References**

* Apache Spark Documentation Apache Software Foundation. Apache Spark Overview and Guide. Retrieved from <https://spark.apache.org/docs/latest/>.
* AWS SageMaker Autopilot Amazon Web Services. Train and Deploy Models Automatically with SageMaker Autopilot. Retrieved from <https://aws.amazon.com/sagemaker/autopilot/>.
* AWS QuickSight Documentation Amazon Web Services. Visualize Data Using AWS QuickSight. Retrieved from <https://aws.amazon.com/quicksight/>.
* PySpark API Reference Databricks. PySpark SQL and DataFrame API. Retrieved from <https://spark.apache.org/docs/latest/api/python/>.