# Contents

1. Introduction
2. Problem Statement
3. Discussion
4. Preprocessing the Dataset
5. Methodologies Used
6. Model Implementation
   1. Global Outlier Detection
   2. Contextual Outlier Detection
   3. Combining Outlier Detection
   4. Model Performance and Evaluation
   5. Visualization of Decision Scores
7. Conclusion

# 1. Introduction

Anomaly detection in time-series data is a powerful tool in domains where identifying irregular patterns can yield valuable insights or early warnings for action. In sectors like finance, healthcare, and transportation, understanding anomalies can help identify fraud, detect equipment failures, or prepare for periods of unusual demand. In this project, we explore anomaly detection within the NYC Taxi demand data using machine learning techniques, specifically targeting global, contextual, and local outliers.

The dataset provides hourly demand patterns for taxis in New York City, exhibiting predictable daily and weekly cycles. However, certain events, holidays, or disruptions may lead to spikes or drops in demand, which appear as anomalies in the data. By identifying these outliers, we can gain insights into unexpected changes in taxi demand, which can support city planners, taxi companies, and other stakeholders.

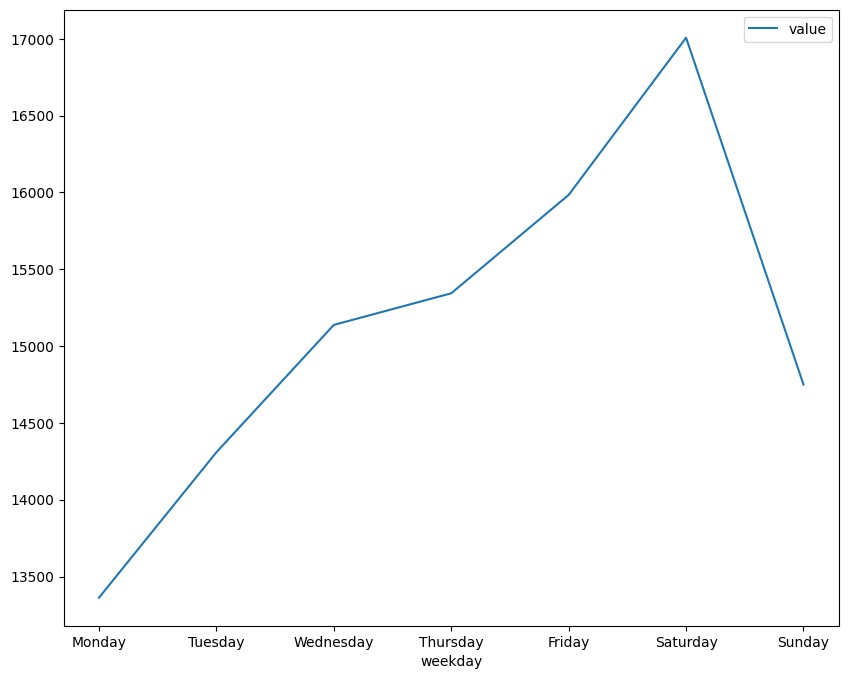
# 2. Problem Statement

The primary objective of this project is to develop an effective machine learning model that accurately identifies anomalies within the NYC Taxi demand data, a time series dataset. In this context, anomalies refer to periods where taxi demand significantly deviates from expected patterns, whether these deviations are unusual across the entire dataset (global outliers) or only within specific contextual subsets, such as certain hours of the day or specific days of the week (contextual outliers).

To accomplish this, we will use **Isolation Forest**, a machine learning algorithm well-suited for anomaly detection, particularly in high-dimensional data, as it isolates anomalies rather than profiling the normal data distribution. The Isolation Forest algorithm isolates data points by partitioning the dataset, identifying instances that require fewer splits as more likely to be anomalous. This approach is efficient, robust to high-dimensional data, and effective in detecting anomalies without requiring extensive parameter tuning.

# 3. Dataset Overview

This dataset, titled "NYC Taxi Demand," contains information on taxi activity within New York City, recorded in half-hour intervals. Each entry has a timestamp indicating the date and time (formatted as "YYYY-MM-DD HH:MM:SS") and a corresponding \*\*value\*\* column representing the number of taxi rides recorded in that interval. Spanning a total of 10,320 entries, the dataset captures a continuous timeline of taxi activity, potentially revealing patterns in demand based on time of day, day of the week, and season. This dataset could be valuable for analyzing temporal patterns in urban mobility, identifying peak hours or unusual spikes in demand, and training models for time series forecasting or anomaly detection. The dataset’s half-hour granularity provides a high-resolution view of fluctuations in taxi usage, which could be useful for applications like predicting taxi demand or assessing city transportation needs.



# 4. Discussion

## 4.1 Anomaly

An anomaly, often referred to as an "outlier," is any data point or sequence that deviates significantly from the expected pattern within a dataset. In the context of credit card transactions, for example, anomalies could include unusually high transaction amounts, rapid successive transactions, or transactions from geographically distinct locations within a short time span. Detecting these anomalies is essential for identifying potentially fraudulent activities and understanding irregularities in data.

## 4.2 Anomaly Detection

Anomaly detection refers to the process of identifying unusual patterns that do not conform to expected behavior. It is a critical tool across industries, helping to preempt issues in areas ranging from fraud detection to equipment maintenance. Effective anomaly detection minimizes false positives (incorrectly flagging normal behavior as anomalous) and false negatives (missing actual anomalies), improving the accuracy and reliability of models deployed in real-world scenarios.

## 4.3 Types of Anomaly Detection

1. **Point Anomalies**: also known as Global anomaly, is a single data instance which can occur in any dataset. It refers to the individual data points that deviates significantly from rest of the data. For instance, in a credit card transaction dataset, a single transaction with a very high amount compared to the average could be a point anomaly.
2. **Contextual Anomalies**: also known as conditional anomaly, are the data points which are considered anomalous only within a specific context or condition. These anomalies might be normal in one context but abnormal in some context. For example, a high expenditure in a holiday shopping season might be normal, but the same amount outside that season could be anomalous.
3. **Collective Anomalies**: also known as group anomaly, it involves a group of data instance that collectively exhibits anomalous behaviour when considered together. Individually the data points might appear normal but their combination is abnormal. For example, a series of small but frequent transactions in a short period may signal fraudulent activity.

## 4.4 Use of Anomaly Detection

Anomaly detection has a wide range of applications across various fields:

* **Financial Services**: Fraud detection in credit card transactions, insurance claims, and tax reporting.
* **Healthcare**: Detecting unusual patient health patterns, which can aid in early diagnosis and prevention.
* **Cybersecurity**: Identifying unusual patterns in network traffic, unauthorized access attempts, and potential security breaches.
* **Retail**: Detecting unusual purchasing behaviors that could indicate either fraud or changing customer preferences.
* **Social Media and Web Monitoring**: Flagging unusual user behavior or harmful content for moderation and security purposes.

## 4.5 Methods for Anomaly Detection

Several machine learning and statistical techniques are commonly used for anomaly detection. Each method is chosen based on the type and distribution of data, the level of anomaly complexity, and real-time performance requirements.

### 1. Statistical Methods

* **Z-Score / Gaussian Distribution**: Measures how far data points are from the mean using standard deviations. While effective for normally distributed data, it may struggle with non-Gaussian distributions.
* **Box Plot and IQR (Interquartile Range)**: Identifies outliers based on the quartile distribution of data. Points outside 1.5 times the IQR are marked as anomalies.

**Best for**: Simple, well-distributed datasets where anomalies are sparse.

### 2. Machine Learning-Based Methods

* **Isolation Forest**: This technique isolates observations by randomly partitioning the data. Points that require fewer partitions to be isolated are more likely to be anomalies. Isolation Forest is effective for high-dimensional data and can be adapted to detect both point and collective anomalies.
* **Support Vector Machine (SVM)**: An SVM-based anomaly detection model (One-Class SVM) aims to find a decision boundary that maximizes the separation between normal and anomalous data. This method works well with datasets where anomalies are few but distinctive.
* **K-Nearest Neighbors (KNN)**: KNN calculates the distance of a point from its nearest neighbors. Points that have larger distances (i.e., are isolated from most of the data) are labeled as anomalies. While effective, KNN can be computationally intensive for large datasets.
* **Clustering Methods** (e.g., DBSCAN, Agglomerative Clustering): These techniques group data based on similarity and mark small or isolated clusters as anomalies. Agglomerative clustering and DBSCAN are effective for collective anomalies, especially in high-dimensional data.

**Best for**: Complex datasets with non-linear relationships, requiring more sophisticated boundary definitions.

### 3. Deep Learning Techniques

* **Autoencoders**: A type of neural network that learns a compressed representation of data. During training, it learns to reconstruct normal data. When an anomaly is passed through the autoencoder, it produces a high reconstruction error, which can be used to flag anomalies.
* **Recurrent Neural Networks (RNN) and LSTM (Long Short-Term Memory) Networks**: Used primarily for time series data, LSTMs learn temporal patterns, making them effective at detecting anomalies in sequences. Anomalies are flagged when observed patterns deviate from the learned sequence.

**Best for**: High-dimensional or sequential data (e.g., time series), where complex relationships between data points exist.

# 5. Preprocessing the dataset

The preprocessing steps taken in this project included:

1. **Loading the Data**: The dataset was imported from an external source and converted into a structured format using Pandas.
2. **Datetime Processing**: The timestamp column was converted to a datetime format to extract features like hour and weekday.
3. **Resampling the Data**: The data was resampled at hourly intervals to standardize it, which is essential for analyzing hourly patterns.
4. **Adding Time-based Features**: We extracted the hour of the day and day of the week as separate columns. These features allow for detecting contextual outliers, as demand patterns typically differ by time of day and day of week.

# 6. Methodologies Used

The approach to anomaly detection involves a layered strategy that combines **global outlier detection** with **contextual analysis** to identify local anomalies:

**6.1 Global Outlier Detection**: This approach uses Isolation Forest, which detects outliers based on the dataset's overall distribution. Isolation Forest is effective because it isolates points that are "rare" in the dataset's global context.

Points that require fewer partitions to isolate are more likely to be anomalies.

**6.2 Contextual Outlier Detection**:

* **Hourly Context**: We calculated hourly mean and standard deviation, flagging points that deviate by more than two standard deviations from the mean for that hour.
* **Weekly Context**: Similarly, we used weekly patterns, flagging outliers that fall outside two standard deviations from the weekday's average demand.

**6.3 Combined Outlier Detection**:

For each point in the dataset, we checked whether it was an outlier in any of the three categories (global, hourly, or weekday). If any category labeled the point as an outlier, it was marked as a "combined" outlier.

# 7. Model Implementations and Evaluations

The primary model used for outlier detection is **Isolation Forest**. Here’s how the implementation was carried out:

1. **Global Outlier Detection with Isolation Forest:**
   1. We trained an Isolation Forest model on the demand data with a contamination level of 0.004 (indicating the approximate fraction of points we expect to be outliers).

○ The model was trained to isolate demand values that deviate substantially from the majority. These are marked as "global" outliers.

1. **Contextual Outlier Detection Using Thresholding:**
   1. **Hourly Outliers**: Calculated hourly means and standard deviations; any value beyond two standard deviations from the mean was flagged as an hourly anomaly.

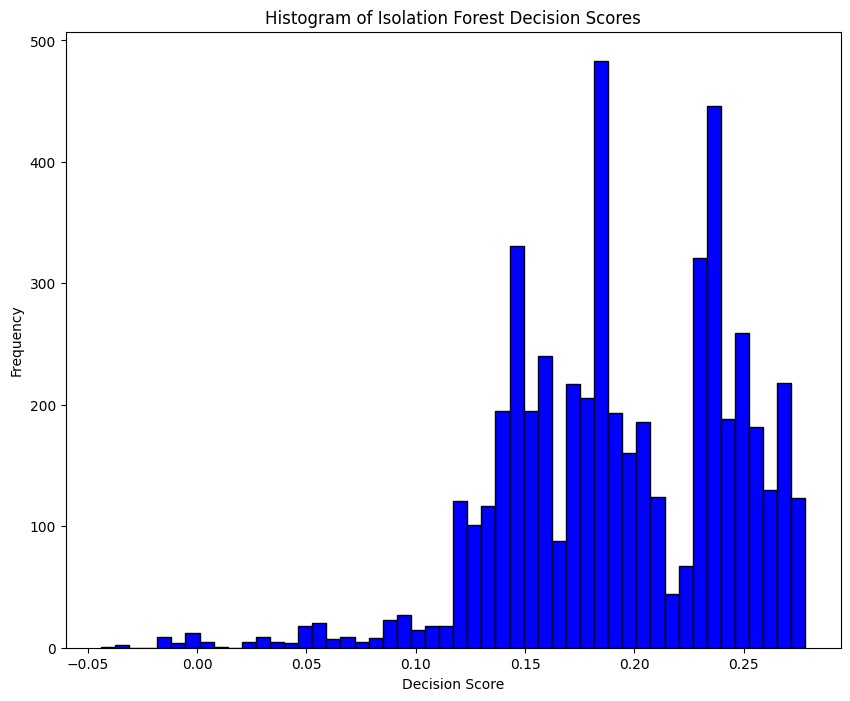
○ **Weekly Outliers**: Calculated weekday means and standard deviations, using the same thresholding technique as above.

1. **Combining Outlier Results:**
   1. If a data point was flagged as an outlier by the global model or in either hourly or weekday contexts, it was labeled as a combined outlier. This combined label captures all unusual behaviors, whether they are rare globally or deviate from expected time-specific norms.
2. **Model Performance and Evaluation:**

To evaluate the model, we used a simple approach, treating global outliers as a baseline or "ground truth" for comparison against the combined outliers.

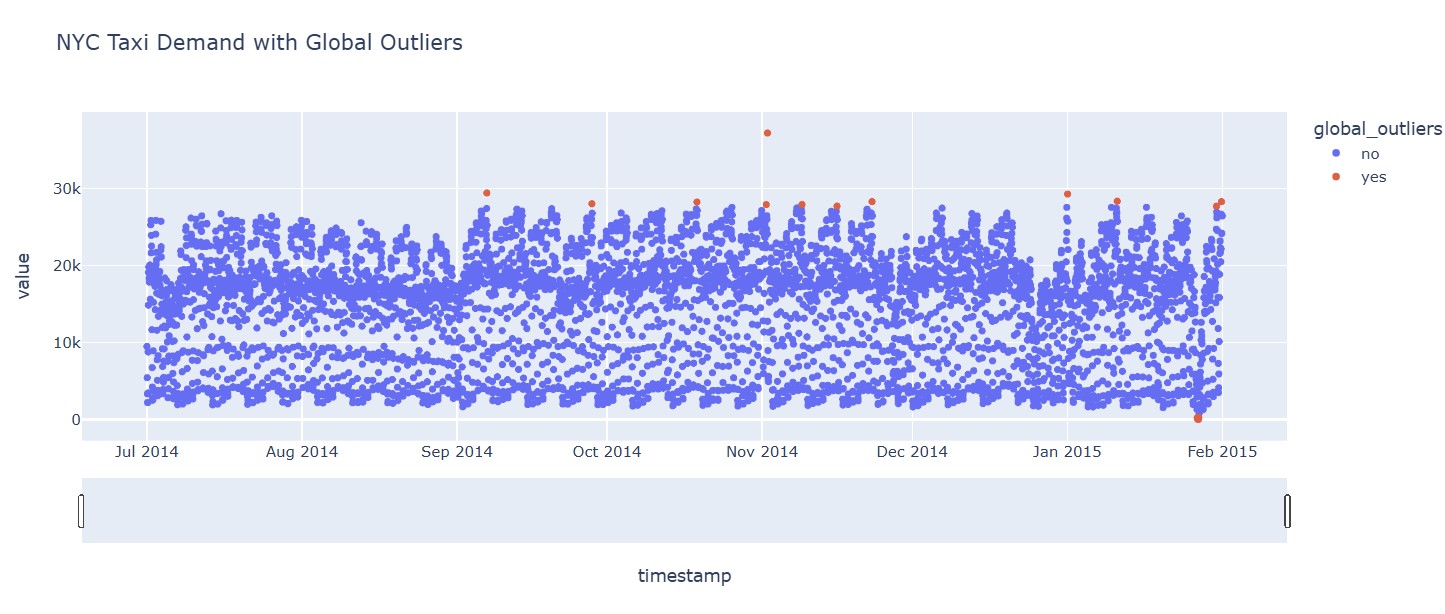
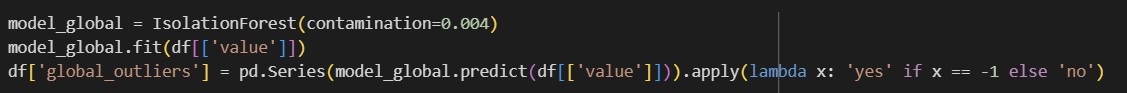
**8.1. Visualization of Decision Scores:**

A histogram of Isolation Forest decision scores was created to visualize the distribution of data points' "normalcy." Points with lower scores are likely to be anomalies, providing insights into the model's confidence levels in its predictions.



**8.2. Isolation Forest:**

Isolation Forest is an unsupervised machine learning algorithm designed for anomaly detection, particularly useful for high-dimensional datasets. Unlike density-based methods, it isolates anomalies by recursively partitioning data, which makes it efficient and effective for outlier detection.



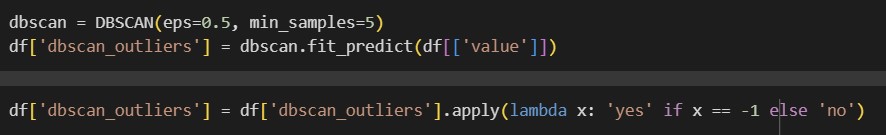
**8.3. DBSCAN:**

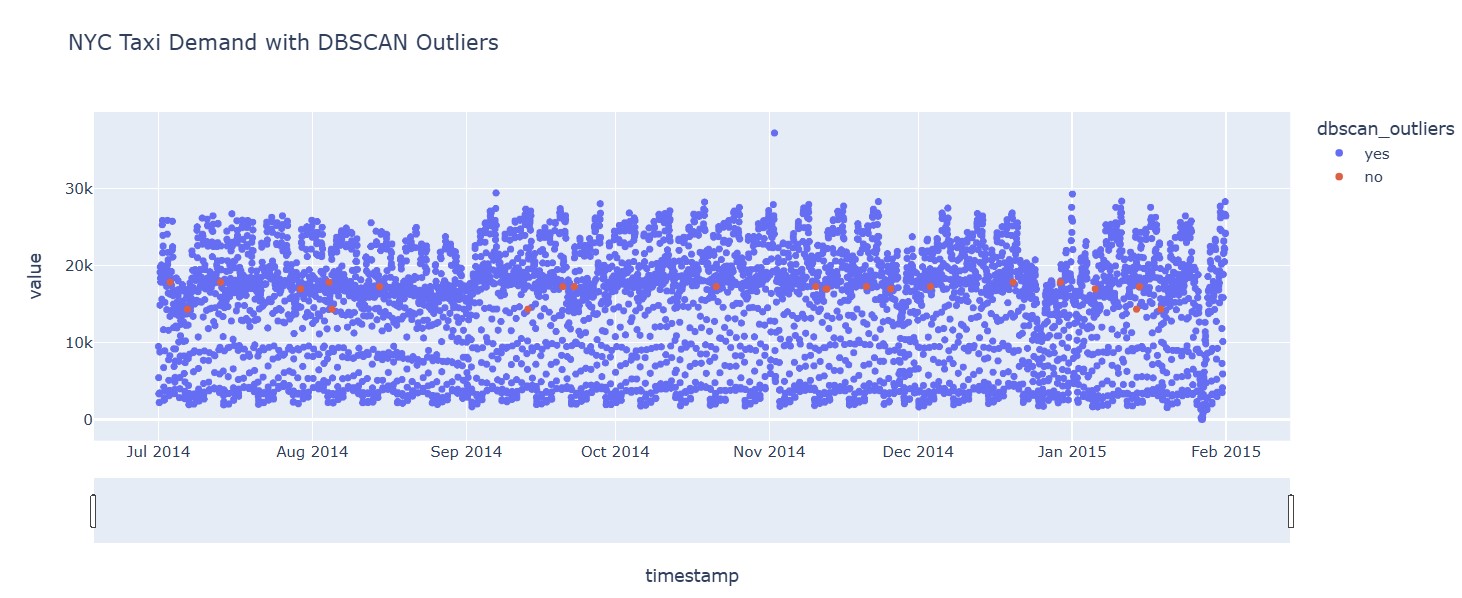
**DBSCAN (Density-Based Spatial Clustering of Applications with Noise)** is an unsupervised machine learning algorithm used for clustering and anomaly detection. It groups points based on density, identifying clusters of arbitrary shapes and flagging outliers.

DBSCAN defines clusters based on two parameters:

* + **Epsilon (eps)**: The maximum distance for points to be considered neighbors.
  + **Min\_samples**: The minimum number of points to form a dense region

(core point).



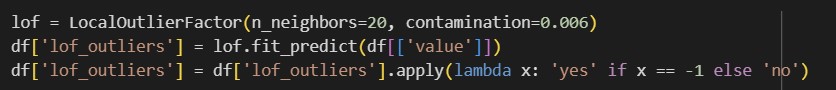


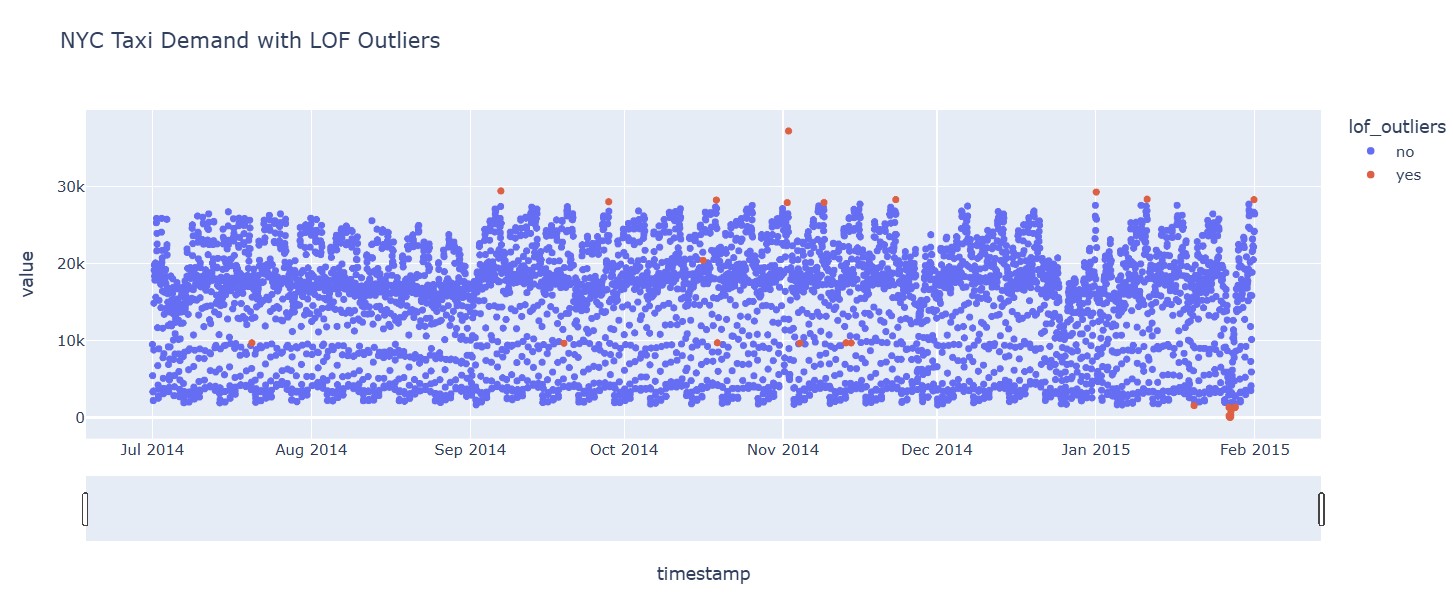
**8.3. Local Outlier Factor (LOF):**

**LOF (Local Outlier Factor)** is an unsupervised anomaly detection algorithm that identifies outliers by comparing the local density of a point with that of its neighbors.

Points with similar densities are considered normal.

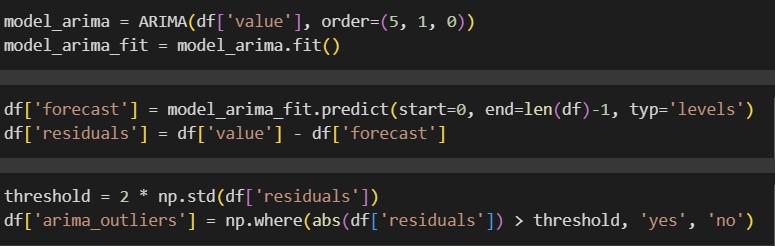
Points with significantly lower densities than their neighbors are labeled as outliers.

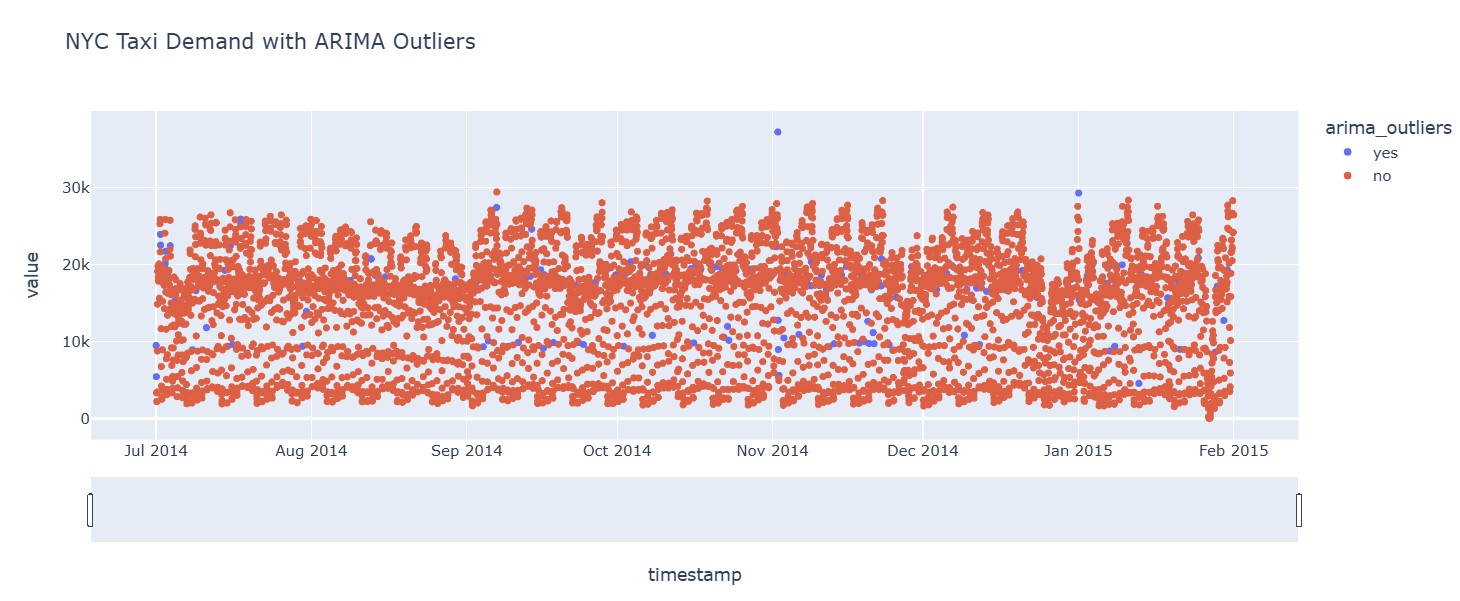




**8.4. ARIMA:**

**ARIMA (AutoRegressive Integrated Moving Average)** is a popular statistical algorithm used for time series forecasting. It is particularly effective for data with trends and seasonality, such as stock prices, sales data, or sensor readings.





**8.5. Evaluation Metrics:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Machine**  **Learning Model** | **Precision** | **Recall** | **F1-Score** | **Accuracy** |
| **Logistic**  **Regression** | 0.11 | 1.00 | 0.20 | 0.97 |
| **DBSCAN** | 0.00 | 1.00 | 0.01 | 0.01 |
| **LOF** | 0.11 | 1.00 | 0.19 | 0.97 |
| **ARIMA** | 0.05 | 1.00 | 0.10 | 0.92 |

# 9. Conclusion

This project demonstrates a practical application of Isolation Forest for anomaly detection in time-series data, specifically targeting the NYC Taxi demand data. Through a combination of global and contextual outlier detection, we were able to identify anomalies in a manner that considers both broad and time-specific patterns.

The results are promising, with decent precision and accuracy, although additional tuning of the model parameters or integrating other contextual features could further enhance performance. This method can be adapted to other time-series datasets where capturing both global and contextual anomalies is necessary for analysis and decision-making.