Wellington Management Group 1

Anomaly Detection for Time Series Data

Student Name and UNI:

Yuhui Li (yl5758), Kittiya Thongpitak (kt2949), Jingwen Zhang (jz3093)

Professor:

Prof. Ali Hirsa

Advisor:

Yossi Cohen

Columbia University

Department of Industrial Engineering and Operations Research

Date:

December 2024

Contents

1 Background

Anomaly detection in financial time series is essential for identifying irregular patterns indicative of significant events, such as fraud or market disruptions. Traditional methods often struggle to adapt to varying economic conditions, leading to inconsistent performance. This project addresses these challenges by developing a comprehensive anomaly detection framework tailored to financial data.

The project framework consists of three key components: regime detection, anomaly detection, and deep learning model refinement. Regime detection identifies distinct economic periods using a silhouette score and a threshold function to assess the introduction of new regimes. Six macroeconomic indicators—GDP, income, CPI, unemployment, interest rates, and trade balance—spanning from 2016 to 2022 were analyzed.

For anomaly detection, algorithms were implemented to flag anomalies based on the consensus of multiple approaches. The distribution of anomalies was analyzed across different economic regimes to understand their behavior under varying conditions. An evaluation system was developed to assess each algorithm's performance, enabling refinement and replacement of less effective models.

Finally, a deep learning model was enhanced by integrating results from the anomaly detection and regime identification phases, improving the model's overall performance.

2 Introduction

This project aims to develop a robust anomaly detection framework for financial time series data. The framework focuses on two main enhancements: improving regime detection and refining anomaly detection.

To enhance regime detection, we experimented with different threshold values and confirmed that the current number of regimes is optimal. Additionally, a merging function was added to the R2RD framework to combine regimes with durations below a specified threshold, ensuring greater stability in regime identification.

For anomaly detection, the SARIMA algorithm was implemented and further refined. Enhanced SARIMA models were applied to individual regimes, enabling the framework to account for seasonality and regime shifts simultaneously. This approach improves the detection of anomalies by tailoring the model to specific economic contexts.

The performance of these methods was evaluated using a comprehensive set of metrics, including precision, recall, F1 score, and AUC-ROC. These metrics were applied during the evaluation of eight different algorithms to ensure robust comparisons.

Furthermore, the enhanced SARIMA model was specifically assessed across four distinct temporal scenarios: Saturday, Sunday, Month Start, and Month End. This analysis, which builds upon a comparison of standard and enhanced SARIMA models, aims to uncover temporal and regime-specific patterns in anomaly occurrences, providing deeper insights into the data.

3 Regime Detection

3.1 Threshold and Assignment Costs

The initial assignment cost for adding an additional regime is 3034895.767444778, with a previously calculated threshold of 3065244.7251192257. A new regime will be added if the assignment cost is less than the threshold. We modified the threshold by adding a scaling factor ρ to the initial threshold, where ρ is a value larger than 0, and we tested around different scaling factors. Various regime counts were tested by silhouette scores, with a regime count of 2 yielding the highest silhouette score (0.4602), suggesting it provides the best clustering quality in terms of separation and compactness.

3.1.1 Threshold Adjustments and Regime Insights

1. Lower Threshold Values ($\rho \leq 1$):
   - Using a lower threshold value restricts the addition of new regimes, resulting in a two-regime division, as shown in Figure 1.
   - While this division aligns with the highest silhouette score and emphasizes longer, sustained regime periods, it may oversimplify the data structure by not capturing subtle but meaningful shifts. Notably, a lower threshold fails to distinguish between the pre-COVID and post-COVID phases, which exhibit different characteristics and should ideally be represented as separate regimes.
2. Higher Threshold Values ($1 < \rho < 1.028$):
   - Increasing the threshold allows for the addition of more regimes beyond the two-regime model, introducing a third regime that distinguishes the COVID period in Figure 2.

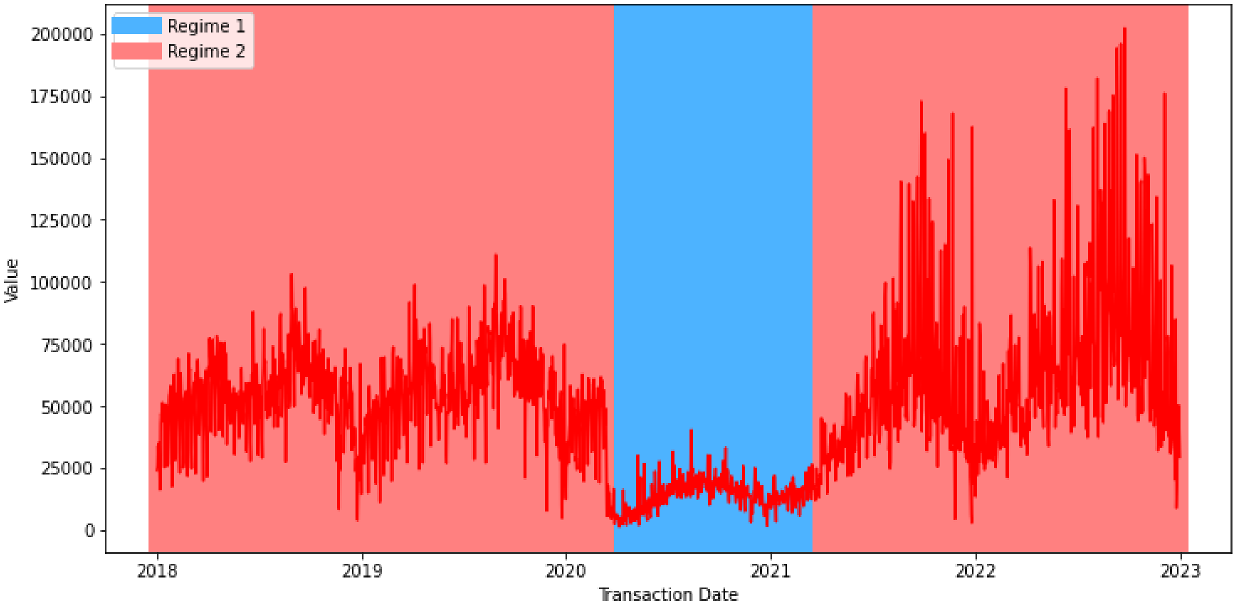


Figure 1: Lower threshold value.

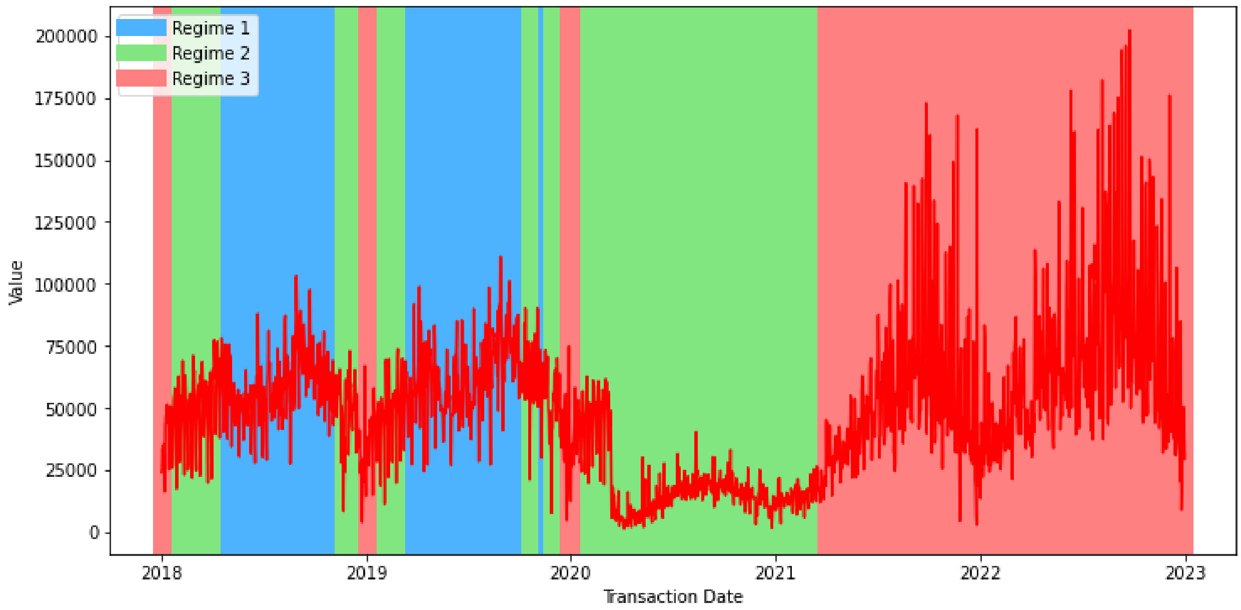
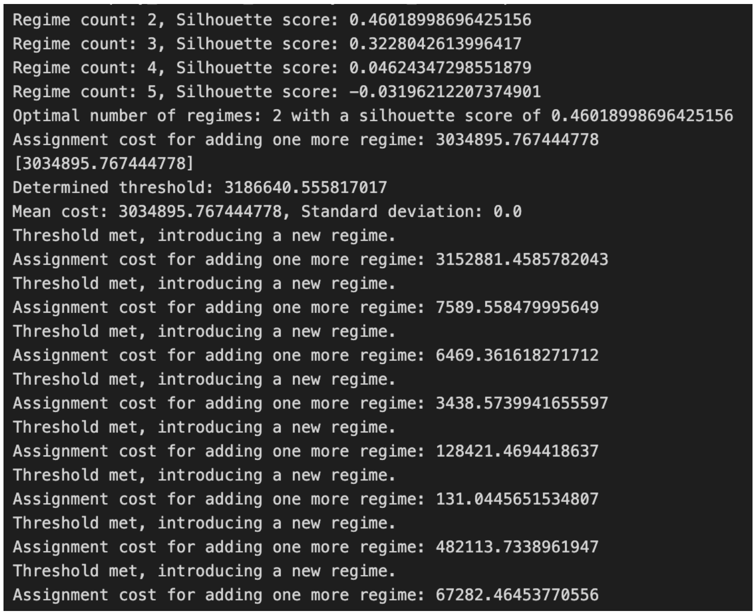


Figure 2: Higher threshold value

- This three-regime configuration captures the significant structural shifts in the data, specifically between pre-COVID, COVID, and post-COVID periods.

3. However, if the threshold is set too high ($\rho \geq 1.028$), the regime-adding process continues, leading to a proliferation of regimes as the assignment cost drops drastically after adding the third regime. This results in over-segmentation, where too many regimes are added, leading to an unreasonable and noisy representation of the data. The graph, in this case, would become cluttered and fail to meaningfully differentiate between sustained regime changes and minor fluctuations.



```
Regime count: 2, Silhouette score: 0.46018998696425156
Regime count: 3, Silhouette score: 0.3228042613996417
Regime count: 4, Silhouette score: 0.04624347298551879
Regime count: 5, Silhouette score: -0.03196212207374901
Optimal number of regimes: 2 with a silhouette score of 0.46018998696425156
Assignment cost for adding one more regime: 3034895.767444778
[3034895.767444778]
Determined threshold: 3186640.555817017
Mean cost: 3034895.767444778, Standard deviation: 0.0
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 3152881.4585782043
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 7589.558479995649
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 6469.361618271712
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 3438.5739941655597
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 128421.4694418637
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 131.0445651534807
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 482113.7338961947
Threshold met, introducing a new regime.
Assignment cost for adding one more regime: 67282.46453770556
```

Figure 3: Assignment costs for new regimes

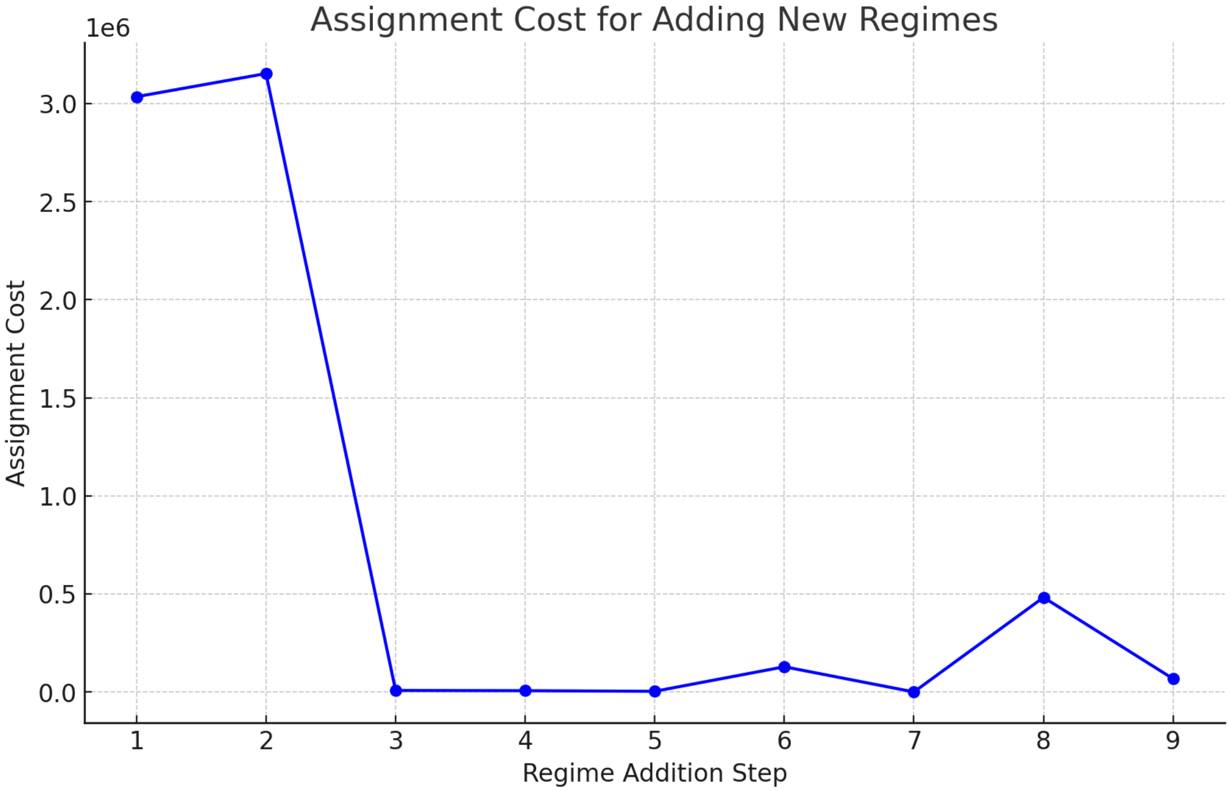


Figure 4: Assignment costs for new regimes

Therefore, the three-regime (adding two regimes) model represents the optimal balance, capturing the major shifts, particularly around the COVID period, without introducing excessive and trivial regimes. This division aligns with the nature of the data by distinguishing between key structural changes without over-segmentation.

3.2 Merging Regimes

To smooth the regime transitions and reduce the number of fragmented regimes, we implemented a function that merges regimes with a duration smaller than a certain threshold. The function iterates through the regime sequence and merges adjacent regimes if the duration of the first regime is less than the threshold.

The updated *plot_regime_detection_with_new_regimes* function introduces logic to merge regimes that fall below a specified and adjustable minimum length ($min\_regime\_length = 150$). This function aims to improve the readability and interpretability of regime segmentation by consolidating shorter, potentially noise-like regimes into adjacent larger ones.

3.2.1 Code Implementation

```
    def plot_regime_detection_with_new_regimes(self, data, model):
        X = data[[’trans_count’, ’spend_amount’]].values
        hidden_states = model.predict(X)
        data.loc[:, ’regime’] = hidden_states

        regime_column = data[’regime’].values
        merged_regimes = np.copy(regime_column)
        current_regime = regime_column[0]
        start_idx = 0
         min_regime_length = 150

        for i in range(1, len(regime_column)):
            if regime_column[i] != current_regime:
                # Check length of the current regime
                regime_length = i - start_idx
                if regime_length < min_regime_length:
                    # Set the range to the previous regime
                    previous_regime_value = merged_regimes[start_idx - 1] if start_idx > 0 else regime_column[i]
                    merged_regimes[start_idx:i] = previous_regime_value

                # Update to the new regime
                current_regime = regime_column[i]
                start_idx = i

        data[’regime’] = merged_regimes

        unique_regimes = np.unique(data.regime)
```

3.3 Results

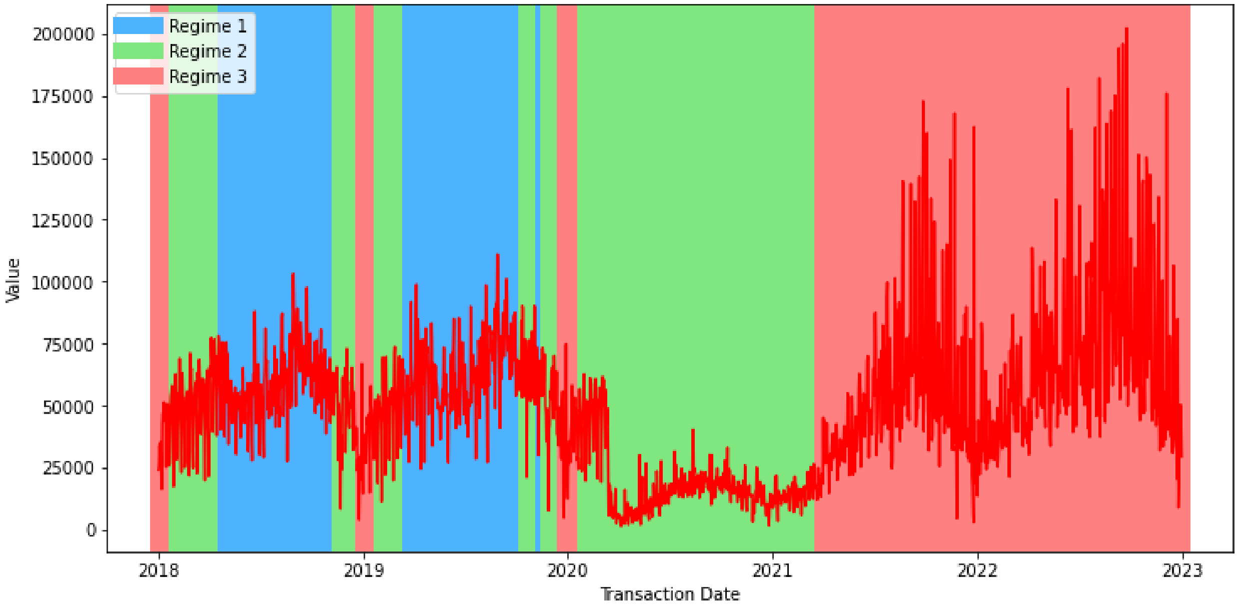


Figure 5: Before merging regimes

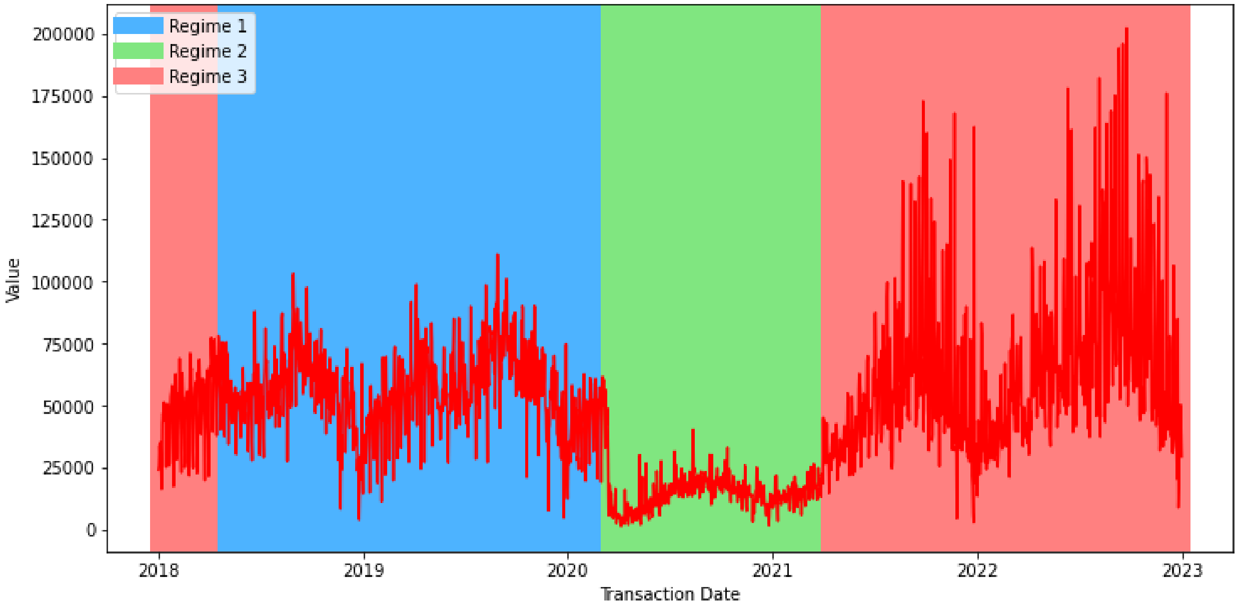


Figure 6: After merging regimes

1. **Reduced noise from short regimes:** By merging shorter regimes, the function reduces over-segmentation caused by fleeting shifts that may not reflect significant structural changes in the data.

2. **Improved interpretability:** The merging of smaller regimes enhances the interpretability of the plot. With fewer, more defined regime shifts, the chart offers a clearer visual representation of major data phases. This adjustment aligns well with the goal of capturing significant shifts (such as pre-COVID, COVID, and post-COVID phases) without unnecessary fragmentation.

3. **Potential drawbacks:** While merging smaller regimes simplifies the plot, it may also obscure legitimate short-term trends that could be relevant in certain analyses. Also, the merging function could be very deterministic using the threshold of duration length. A more sophisticated approach could consider testing the statistical significance of the difference before and after a change in regime.

4 Anomaly Detection Algorithms

Below are the additional algorithms implemented:

4.1 SARIMA

4.1.1 Algorithm Description

SARIMA (Seasonal AutoRegressive Integrated Moving Average) is a time series forecasting model that extends ARIMA by adding seasonal components to account for seasonal patterns in the data.

In the context of anomaly detection, SARIMA is used to predict the expected values of a time series. The residuals (the difference between the actual and predicted values) are measured, and if they exceed a certain threshold, the point is flagged as an anomaly. A dynamically computed threshold (e.g., based on the 95th percentile of residuals) was used. This method is effective for detecting outliers in time series data where seasonal patterns and trends are present. Anomalies are detected by comparing the residuals to the dynamically computed threshold.

4.1.2 Parameter Selection

The SARIMA parameters were chosen as follows:

- **SARIMA Order (p, d, q)**: (1, 1, 1)
  - **p=1**: Captures immediate dependencies in the data.
  - **d=1**: Differencing the data once to remove trends.
  - **q=1**: Models the moving average component of the residuals.

- **Seasonal Order (P, D, Q, S)**: (1, 1, 1, 12) selected to capture seasonality with a 12-period cycle.

- **Dynamic Threshold**: The top 5% of residuals are flagged as anomalies to ensure that only the most extreme deviations from the SARIMA model's predictions are considered anomalies.

4.2 Results

In the experiment conducted for the ARMK company dataset, 128 points out of 2557 data points were flagged as anomalies, representing approximately 5% of the total data. However, in future iterations, it may be necessary to fine-tune the parameters or threshold to adjust the sensitivity of the anomaly detection, as the number of anomalies could vary depending on the data characteristics.

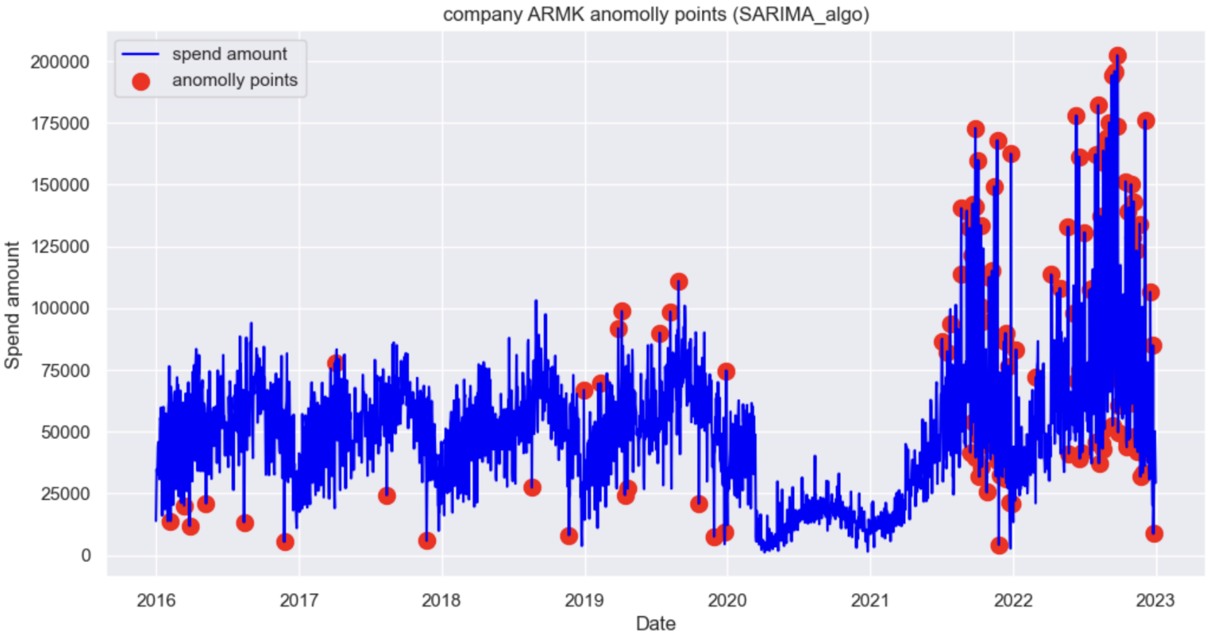


Figure 7: Anomaly points in the SARIMA algorithm

5 Removing Seasonality Effect from Regime Detection

5.1 Initial Approach: SARIMA for Anomaly Detection

Our initial approach to anomaly detection in the time series data involved using the Seasonal Autoregressive Integrated Moving Average (SARIMA) model. This model was chosen due to its ability to handle seasonal patterns effectively while accounting for trends and autocorrelations in the data. By fitting a SARIMA model to the entire dataset, we aimed to capture the underlying structure of the time series and identify deviations that may represent anomalies.

While this approach performed reasonably well in detecting anomalies under stable conditions, we observed inconsistencies in the results, particularly during periods of abrupt changes in the time series' behavior. These inconsistencies led us to suspect the presence of regime shifts in the data, which are characterized by significant structural changes that can influence the dynamics of the time series.

5.2 Enhanced Approach: SARIMA with Regime Detection

To address the limitations of the initial approach, we implemented a regime detection algorithm to identify distinct regimes in the time series data. The algorithm divided the data into three regimes based on changes in its statistical properties. These regime shifts were confirmed through visual inspection and statistical testing, validating the need to incorporate them into our analysis.

In the enhanced approach, we applied the SARIMA model separately to each of the three regimes. By segmenting the data in this way, we aimed to better capture the unique seasonal and structural characteristics within each regime. The fitted SARIMA models for the individual regimes were then combined to construct a unified representation of the time series.

This methodology allowed us to remove the seasonality effect while accounting for the regime shifts in the data. Anomalies were subsequently detected by analyzing the residuals of the combined SARIMA models, with the expectation that this approach would yield more accurate and interpretable results in the presence of regime effects.

5.3 Comparison of Approaches

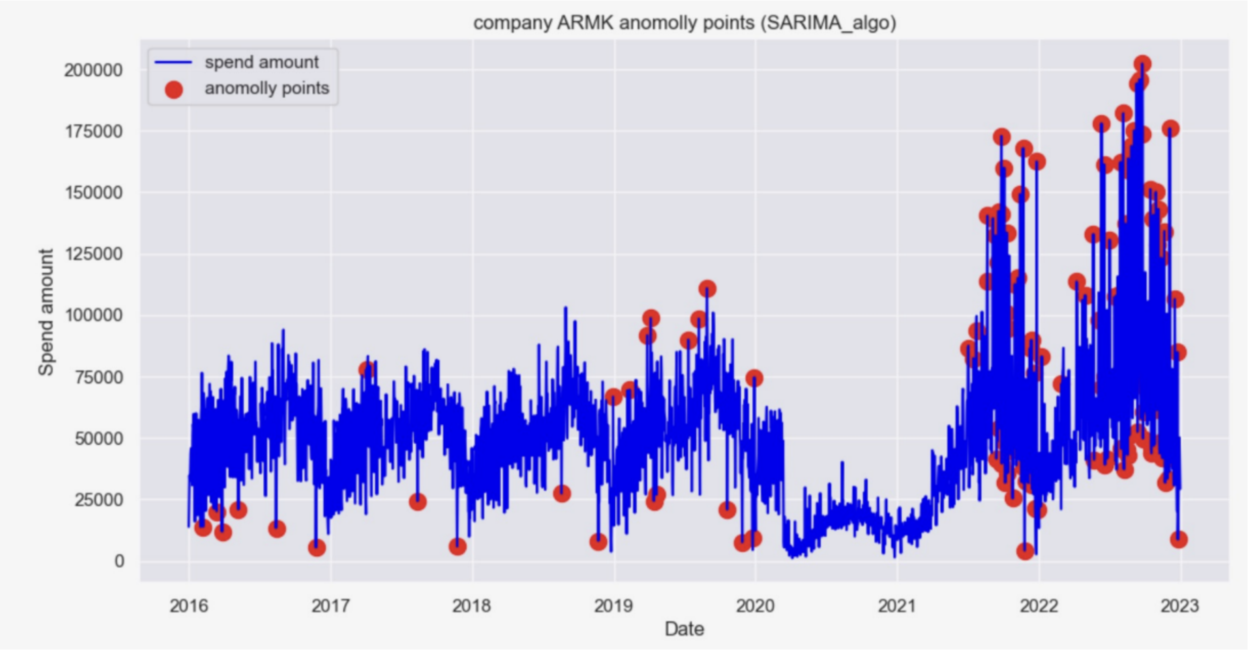


Figure 8: Initial Approach

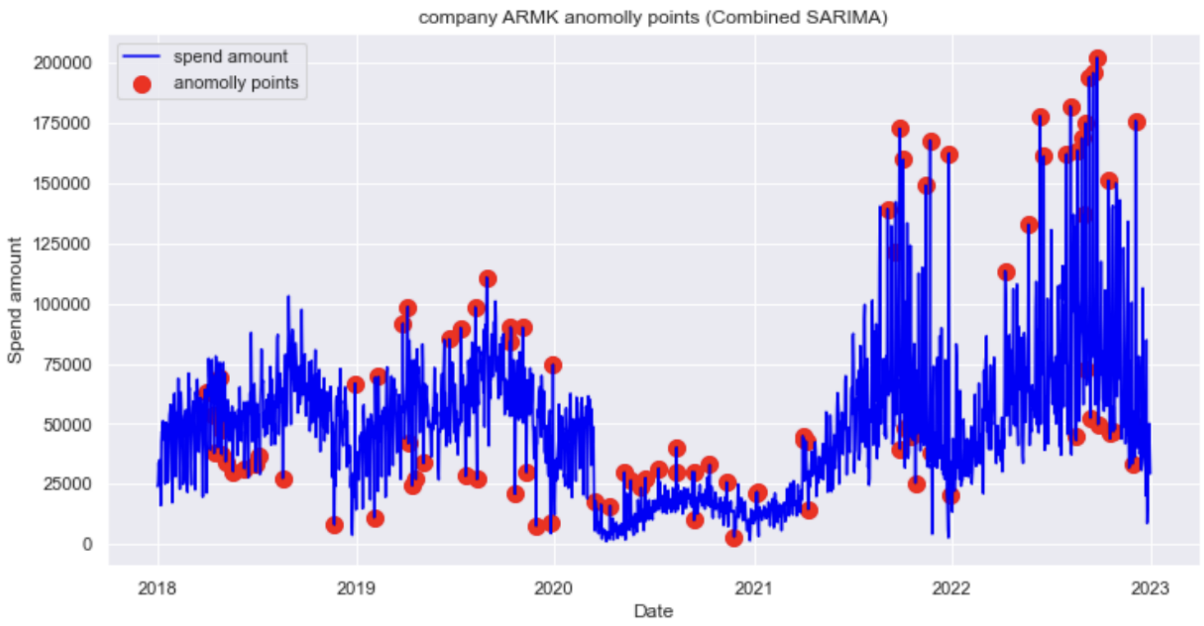


Figure 9: Enhanced Approach

The comparison between the two approaches highlights the importance of incorporating regime detection in time series with structural changes. The initial SARIMA-based approach struggled to adapt to abrupt shifts in the data, leading to false positives or missed anomalies. The enhanced approach, by leveraging regime-specific models, provided a more robust framework for anomaly detection, particularly in the presence of regime shifts.

5.4 Code Implementation

```
class SARIMA_algo(Algos):
def __init__(self, df) -> None:
    super().__init__(df)
    self.result = None
    self.all_results = pd.DataFrame()

@staticmethod
def anomalies(train_data, order=(1, 1, 1), seasonal_order=(1, 1, 1, 12),
    threshold_percentile=95):
    sarima_model = SARIMAX(train_data, order=order,
        seasonal_order=seasonal_order, enforce_stationarity=False,
        enforce_invertibility=False)
    sarima_fit = sarima_model.fit(disp=False)
    predicted = sarima_fit.predict(start=train_data.index[0],
        end=train_data.index[-1])
    residuals = train_data - predicted
    threshold = residuals.abs().quantile(threshold_percentile / 100.0)
    anomalies = (residuals.abs() > threshold).astype(int)
    return anomalies

def sarima_algo(self, company_id, threshold_percentile=95):
    company_data = self.get_company_data(company_id)
```

```
        train_data = company_data['spend_amount']
        anomalies = self.anomalies(train_data, threshold_percentile=threshold_percentile)
        company_data['anomalies'] = anomalies
        self.result = company_data
        return self

    def get_data(self):
        return self.result

    def plot(self):
        ap = anomally_plot(self.result.reset_index())
        ap.plot(self.__class__.__name__)

    def sarima_algo_for_all(self, threshold_percentile=95):
        for id in self.company_ids:
            result = self.sarima_algo(id).get_data()
            self.all_results = pd.concat([self.all_results, result])
        return self.all_results

    regimes = sorted(df['regime'].unique())
    combined_results = pd.DataFrame()
    for regime in regimes:
        regime_data = df[df['regime'] == regime]
        #regime_data['trans_date'] = pd.to_datetime(regime_data['trans_date'])
        sarima = SARIMA_algo(regime_data)
        result = sarima.sarima_algo('ARMK').get_data()
        print(f"Results for regime {regime}:")
        print(result)
        combined_results = pd.concat([combined_results, result])#, ignore_index=True)
        sarima.plot()
    combined_results.to_csv("sarima_results.csv", index=False)
```

5.5 Comparison and Evaluation

5.5.1 Performance Metrics

The table below summarizes the evaluation metrics for the initial and enhanced approaches:

Metric	**Initial Approach**	**Enhanced Approach**
Precision	0.53	0.36
Recall	0.34	0.23
Accuracy	0.93	0.91
F1 Score	0.42	0.28
AUC-ROC	0.66	0.60
AUC-PR	0.23	0.14

Table 1: Comparison of Performance Metrics for Initial and Enhanced Approaches

5.6 Observations

- **Precision:** The precision dropped from 0.53 in the initial approach to 0.36 in the enhanced approach, indicating an increase in false positives.
- **Recall:** The recall decreased from 0.34 to 0.23, meaning the enhanced approach missed more true anomalies.
- **Accuracy:** There was a minor decrease in accuracy, from 0.93 to 0.91, reflecting a slight reduction in overall classification performance.
- **F1 Score:** The F1 score dropped significantly from 0.42 to 0.28, highlighting the trade-off between precision and recall.
- **AUC-ROC:** The AUC-ROC value decreased from 0.66 to 0.60, indicating reduced discriminatory ability.
- **AUC-PR:** The AUC-PR value dropped from 0.23 to 0.14, further emphasizing the reduced performance of the enhanced approach in identifying anomalies.

Currently, the enhanced method shows worse performance compared to the initial approach. However, this could be attributed to the lack of knowledge of the true labels in the dataset. We simulated the "true labels" using a voting system among eight algorithms, but these algorithms were not designed to incorporate the enhanced approach with regime-specific modeling. As a result, the simulated "true labels" may inadvertently ignore the regime effects. Consequently, if our enhanced approach identifies anomalies that differ from the voted labels, these could be classified as false positives. In reality, such instances may highlight the strength of our enhanced model, as it detects seasonality within the context of regime shifts - an aspect that the initial approach and the algorithms forming the voting system may fail to capture.

6 Enhanced SARIMA Anomaly Detection for Different Scenarios

6.1 Implementation of Enhanced SARIMA

The enhanced SARIMA model accounts for both seasonal effects and regime-specific dynamics. To address the limitations of the initial approach, we implemented a regime detection algorithm to identify distinct regimes in the time series data. The algorithm divided the data into three regimes based on changes in its statistical properties. These regime shifts were confirmed through visual inspection and statistical testing, validating the need to incorporate them into our analysis.

In the enhanced approach, we applied the SARIMA model separately to each of the three regimes. By segmenting the data in this way, we aimed to better capture the unique seasonal and structural characteristics within each regime. The fitted SARIMA models for the individual regimes were then combined to construct a unified representation of the time series.

This methodology allowed us to remove the seasonality effect while accounting for the regime shifts in the data. Anomalies were subsequently detected by analyzing the residuals of the combined SARIMA models, with the expectation that this approach would yield more accurate and interpretable results in the presence of regime effects. Moving forward, our work focuses on anomaly detection for different temporal data scenarios.

6.2 Results

Based on the comparison between standard and enhanced SARIMA, it is evident that applying SARIMA to different regimes yields more consistent anomaly detection results. Below are the results obtained using the enhanced SARIMA model for Saturday, Sunday, and Month End/Start datasets.

6.2.1 Saturday

Results for the Saturday scenario showed a total of 261 transactions, with 13 anomalies (5%) detected. The regime distribution was as follows:

- **Regime 0:** 101 transactions, 0 anomalies
- **Regime 1:** 57 transactions, 0 anomalies
- **Regime 2:** 103 transactions, 13 anomalies

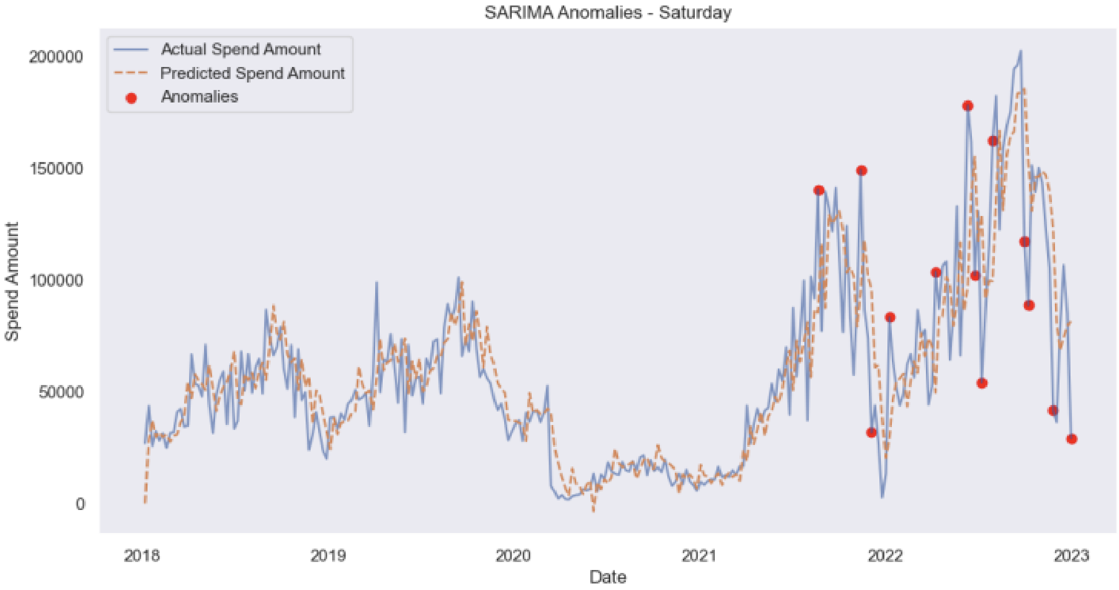


Figure 10: Anomaly Detection for Saturday Scenario

6.2.2 Sunday

Results for the Sunday scenario showed a total of 260 transactions, with 13 anomalies (5%) detected. The regime distribution was as follows:

- **Regime 0:** 101 transactions, 1 anomaly
- **Regime 1:** 57 transactions, 0 anomalies
- **Regime 2:** 102 transactions, 12 anomalies

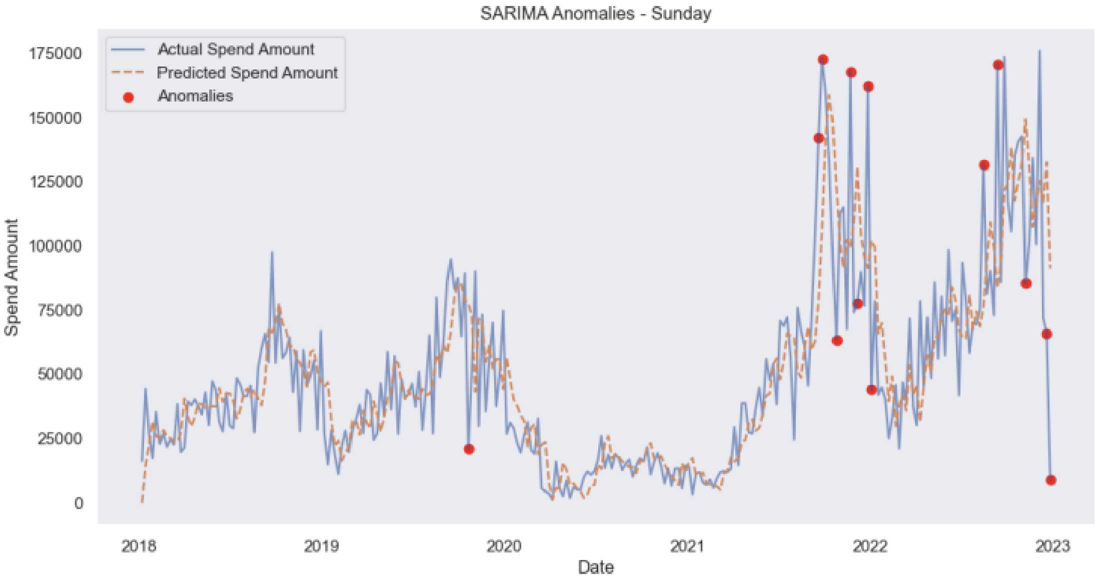


Figure 11: Anomaly Detection for Sunday Scenario

6.2.3 Month Start

Results for the Month Start scenario showed a total of 60 transactions, with 3 anomalies (5%) detected. The regime distribution was as follows:

- **Regime 0:** 23 transactions, 2 anomalies
- **Regime 1:** 13 transactions, 0 anomalies
- **Regime 2:** 24 transactions, 1 anomaly

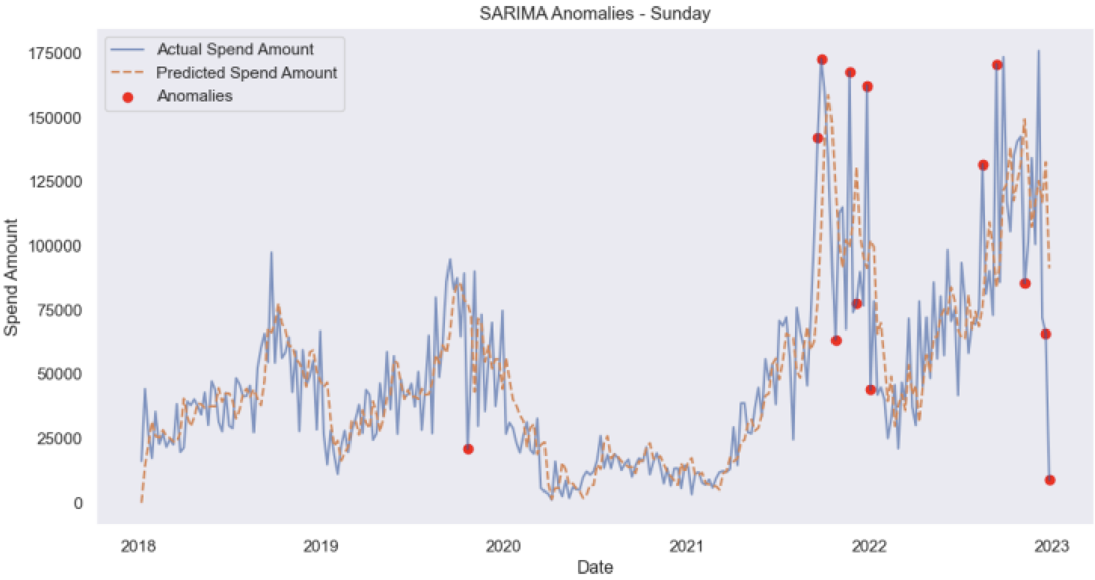


Figure 12: Anomaly Detection for Month Start Scenario

6.2.4 Month End

Results for the Month End scenario showed a total of 60 transactions, with 3 anomalies (5%) detected. The regime distribution was as follows:

- **Regime 0:** 23 transactions, 0 anomalies
- **Regime 1:** 13 transactions, 0 anomalies
- **Regime 2:** 24 transactions, 3 anomalies

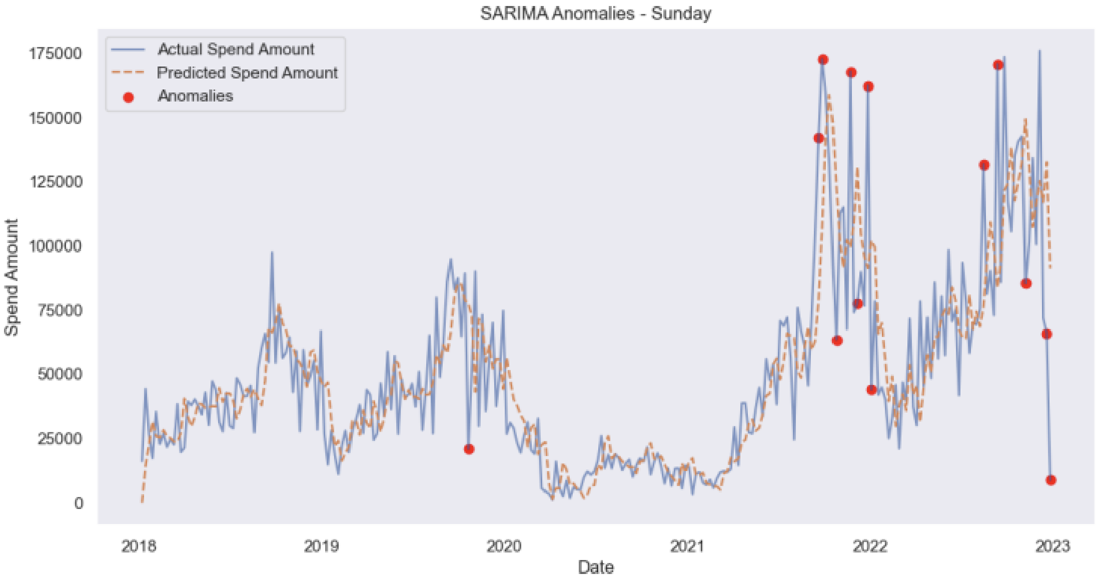


Figure 13: Anomaly Detection for Month End Scenario

6.3 Comparison of Scenarios

Scenario	Regime	Total Transactions	Anomaly Count
Month End	0	23	0
Month End	1	13	0
Month End	2	24	3
Month Start	0	23	2
Month Start	1	13	0
Month Start	2	24	1
Saturday	0	101	0
Saturday	1	57	0
Saturday	2	103	13
Sunday	0	101	1
Sunday	1	57	0
Sunday	2	102	12

The enhanced SARIMA model effectively captured anomalies across all scenarios. Weekends exhibited higher anomaly rates, especially in Regime 2, suggesting irregular consumer behavior. Month Start and Month End anomalies were more modest, likely tied to financial cycle regularities.

7 Further Improvement

In the final section, we listed some suggestions for the next group to work on.

7.0.1 Regime Detection

- We could implement the merging based on testing the significance level of the regime-switching periods.
- Hidden Semi-Markov Model (HSMM) employs a structure where the probability of a change in the hidden state depends on the amount of time that has elapsed since entry into the current state. We can potentially make use of this property and use HSMM to model regime detection, with the hope of obtaining smoother regime transitions.

7.0.2 Anomaly Detection

- Incorporate this enhanced approach into the eight previous models, i.e., implement this approach into our OOP framework, to evaluate its performance more accurately.
- Incorporate external indicators (e.g., economic data, promotions) to contextualize anomalies.
- Develop ensemble models combining SARIMA with machine learning methods.
- Investigate the dynamics of regime transitions for deeper insights.