1. **Problem Statement**

**1.1. Module overview:**

The Dynamic Order Analytics Module (DOAM) is a sophisticated data analysis program designed to monitor and analyze online customer order data in real-time. Utilizing advanced statistical algorithms, DOAM identifies significant deviations in product order volumes, pinpointing products that experience a sudden increase in orders even if they are not typical bestsellers. This system leverages both moving average calculations and time series analysis (e.g., ARIMA models) to detect these anomalies.

**1.2. Consideration** => done 1st version => not suitable for anomaly detection

. To detect products in online customer order data whose order volume suddenly increases significantly, even if they're not overall bestsellers, you can use statistical algorithms like calculating moving averages or employing time series analysis.

. Choose two algorithms for anomaly detection : ex) Moving Average and Standard Deviation, (2) Time Series Analysis (e.g., ARIMA)

=> you can choose right algorithm for anomaly detect

. Find similar results with the variables, thresholds for comparing the data.

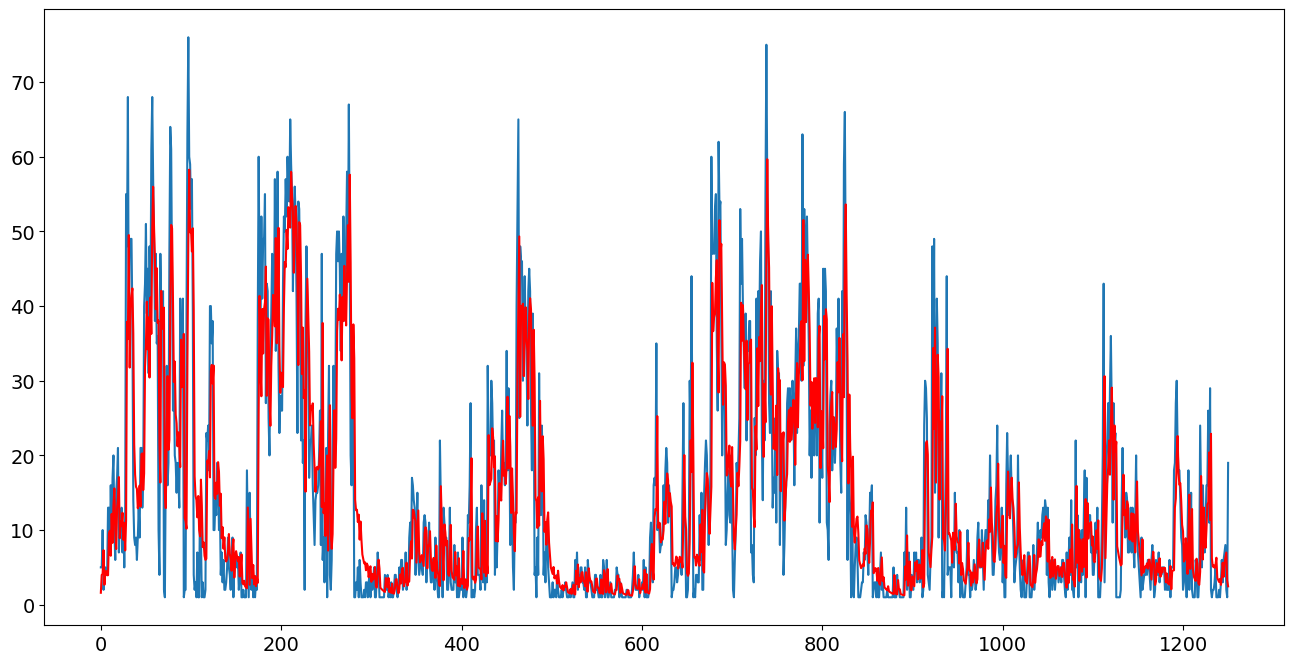
. Define input parameter for query condition (compare date, period,..)

. Define output structure for handling the record data

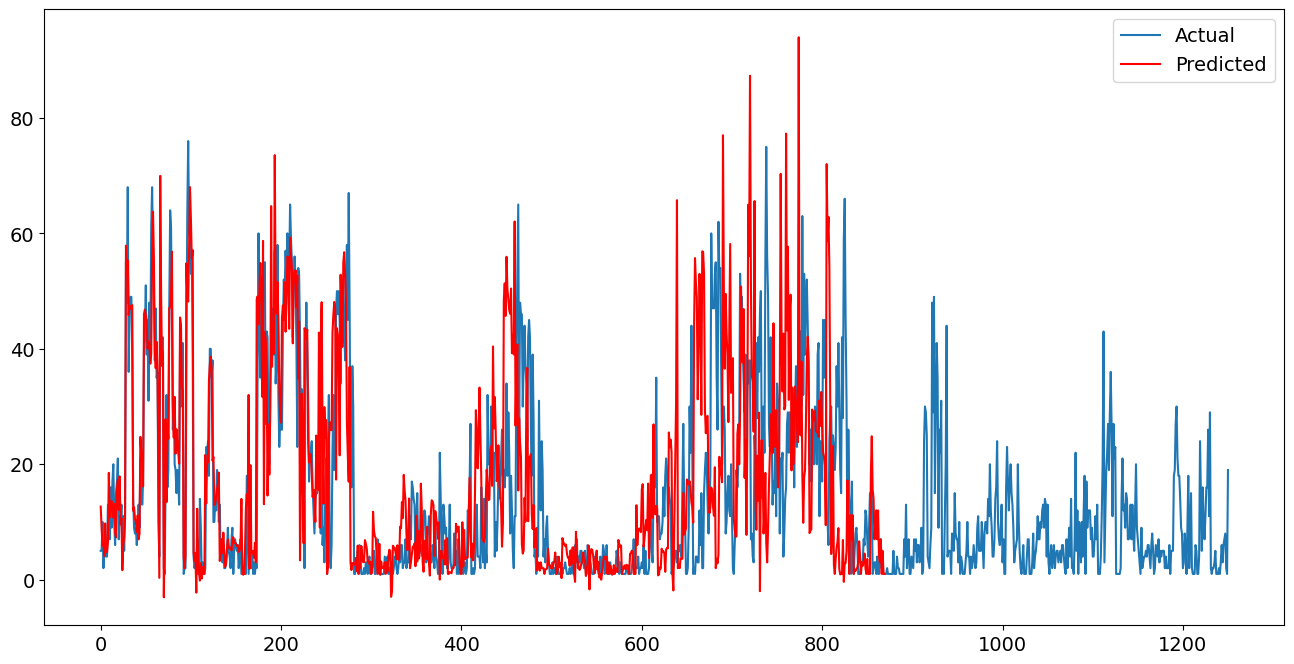
2 methods used for Anomalies Forecasting => ARIMA model: use series of data to predict => 1 value in previous day to predict 1 value in the next day => the following days each item is predicted when calling function

Method 1: Split data by Item: consider all items are similar

Model eval performed by RMSE score: 9.205



Method 2: Split data by Date: consider all items are different (set index item to train with looping items list)



**II. Descriptive methods**

**2.1. Function 1: Detection module (Moving Average and Standard Deviation)**

- Data Preprocessing: used Mysql to retrieve data, Python language to clean data, data transformation, data manipulation from '2015-04-02 00:00:00' to now

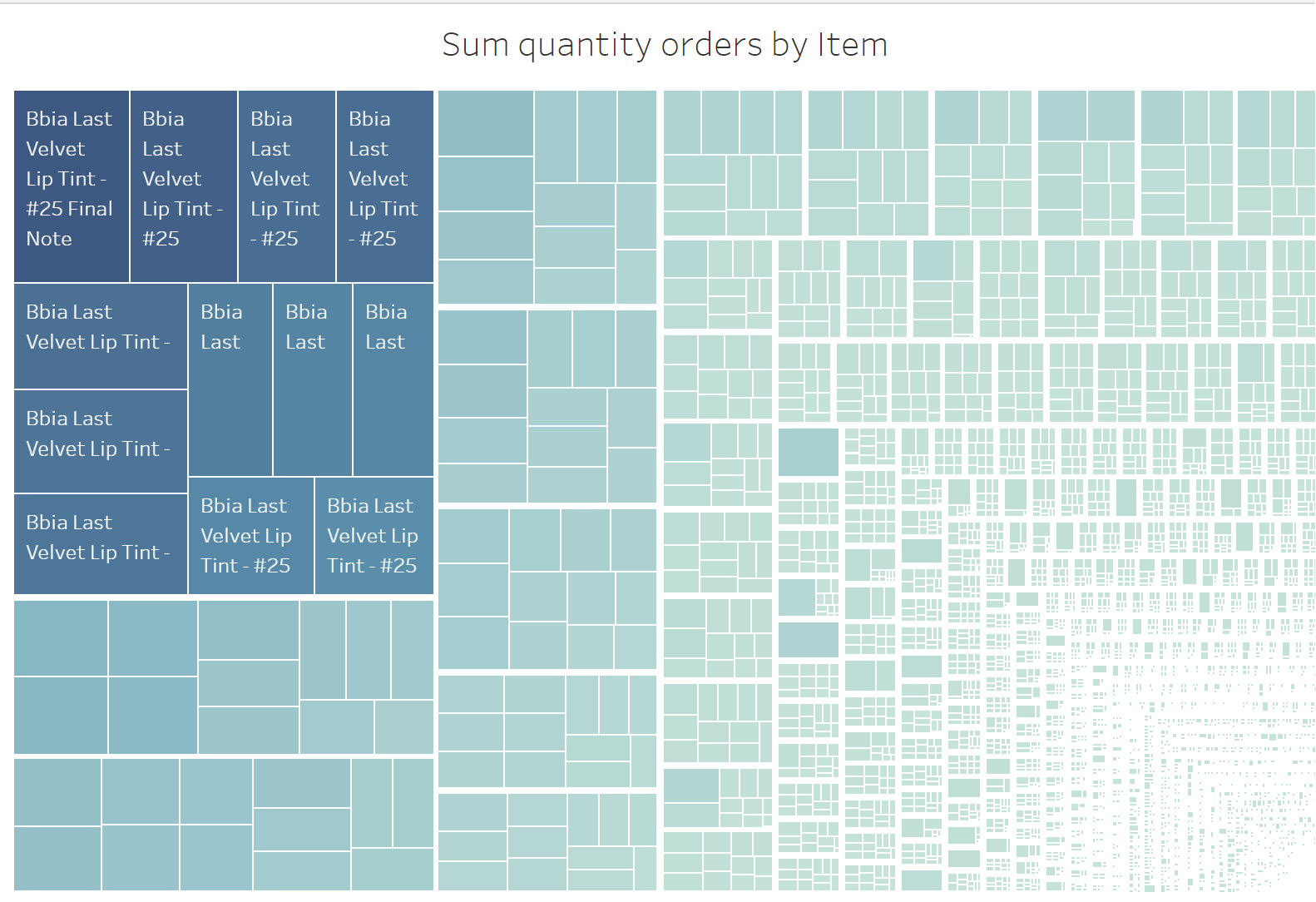
| Variable | Definition | Type of data |
| --- | --- | --- |
| item\_id | id of each item (product\_id) | numerical |
| quantity | quantity orders of each item appears in many bills | numerical |
| post\_date | date with exact timestamp for each item quantity | datetime |
| order\_item\_name | name of each item ordered | text |

*Table 1. Data Description*

- Calculate Moving Average: Compute the average order quantity over the past N months (e.g., 3 months) for each product.

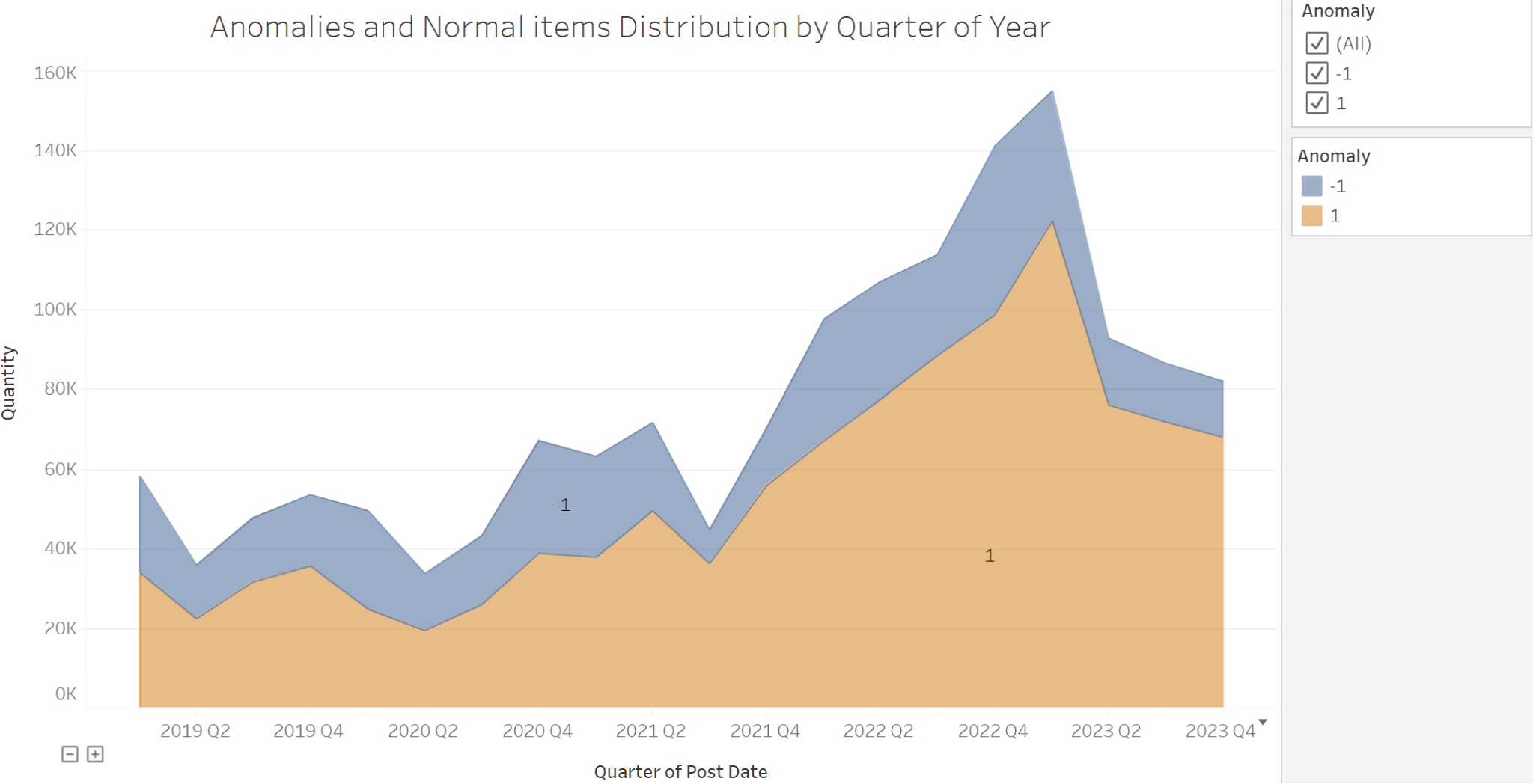
- Calculate Standard Deviation: Calculate the standard deviation of order quantities for the same period.

- Detect Anomalies: If the recent order quantity for a product exceeds the moving average plus a certain number of standard deviations



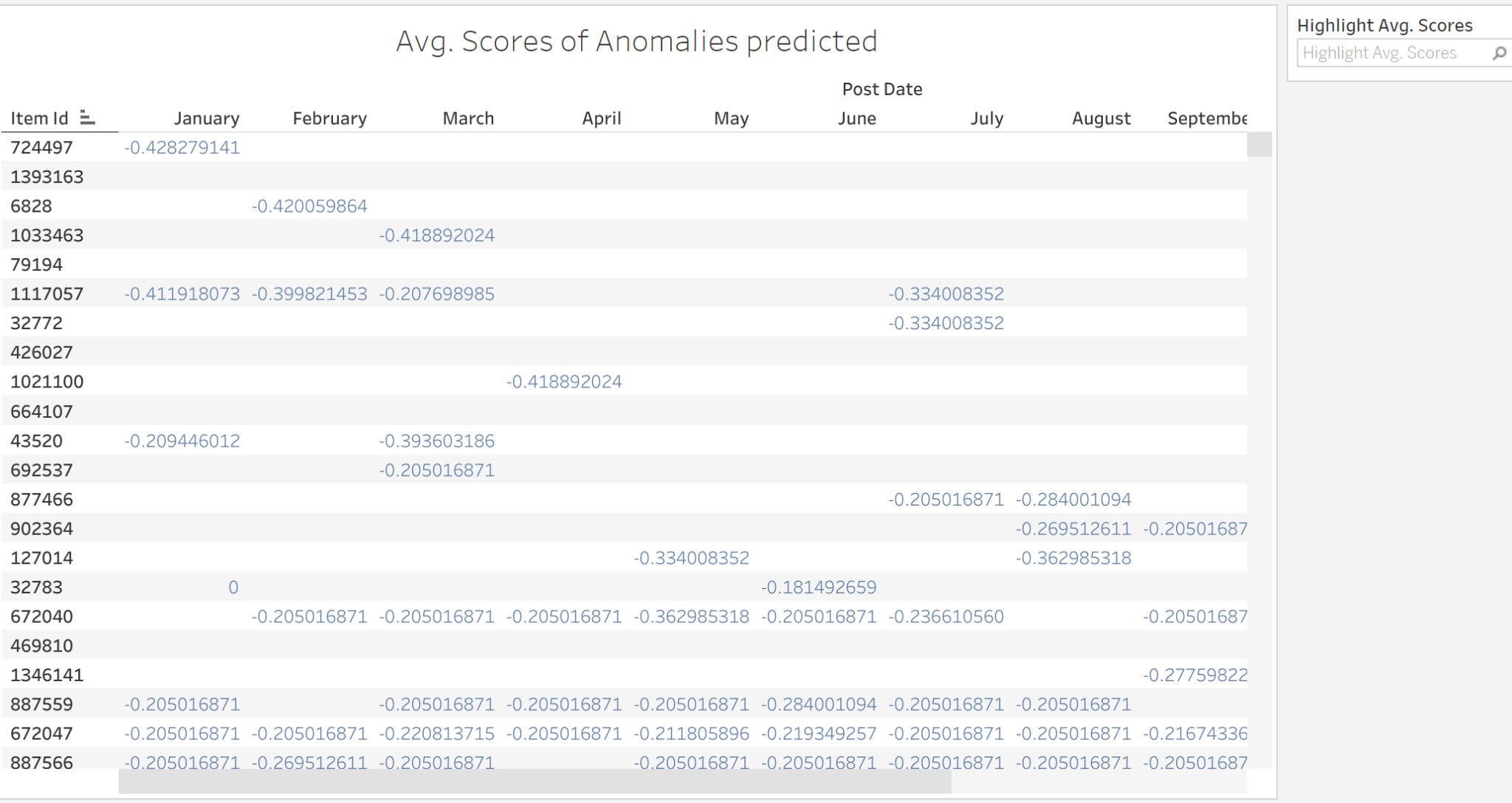
*Figure 1: Sum quantity orders by Item*

The above heatmap show the left-corner is the group of items having the highest quantity orders such as ‘Bbia Last Velvet Lip Tint - #25 Final Note’, ‘Bbia Last Velvet Lip Tint - #25’



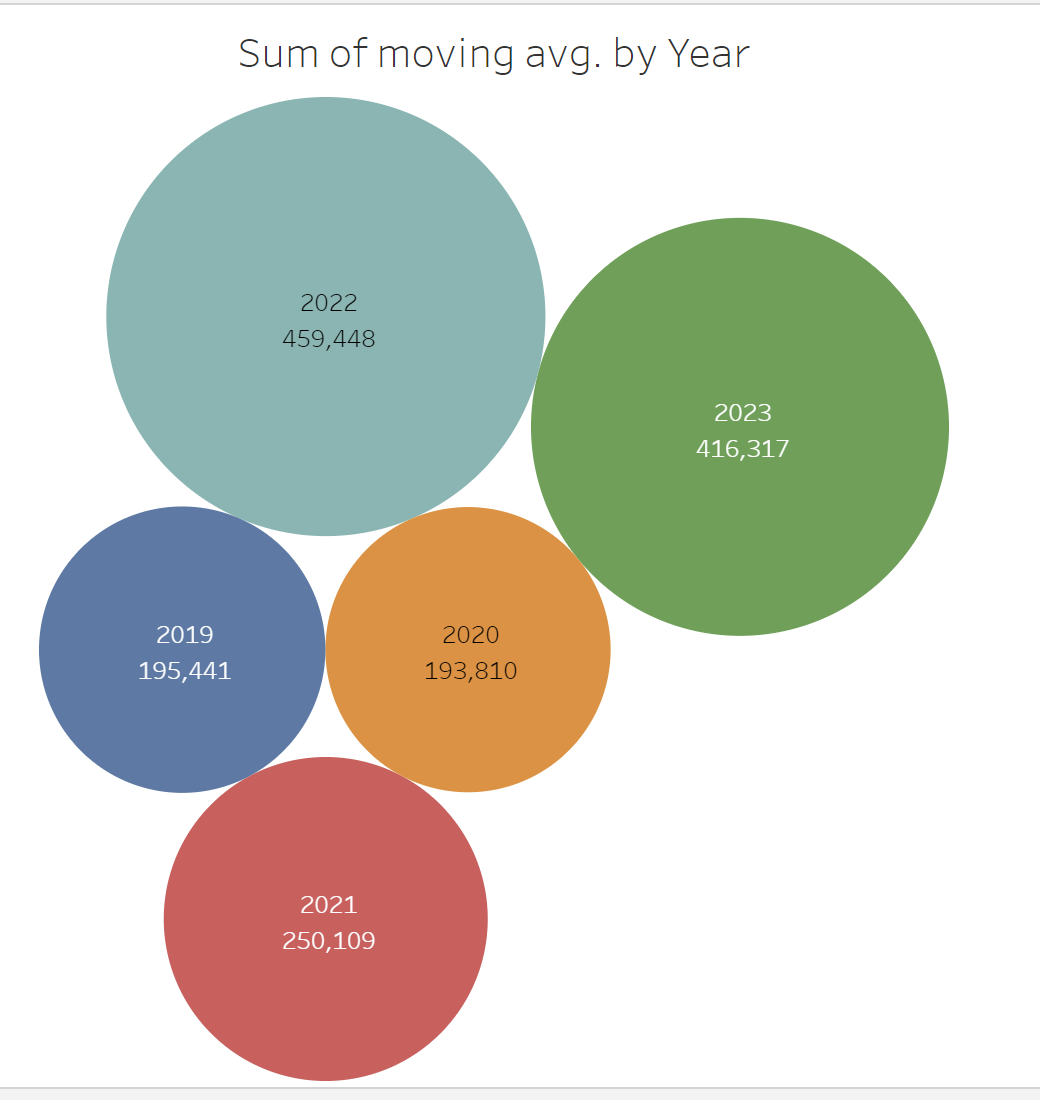
*Figure 2: Anomalies and Normal items Distribution by Quarter of Year*

Anomaly detection results (-1) are showed fewer area distribution than items having normal quantity orders (1)



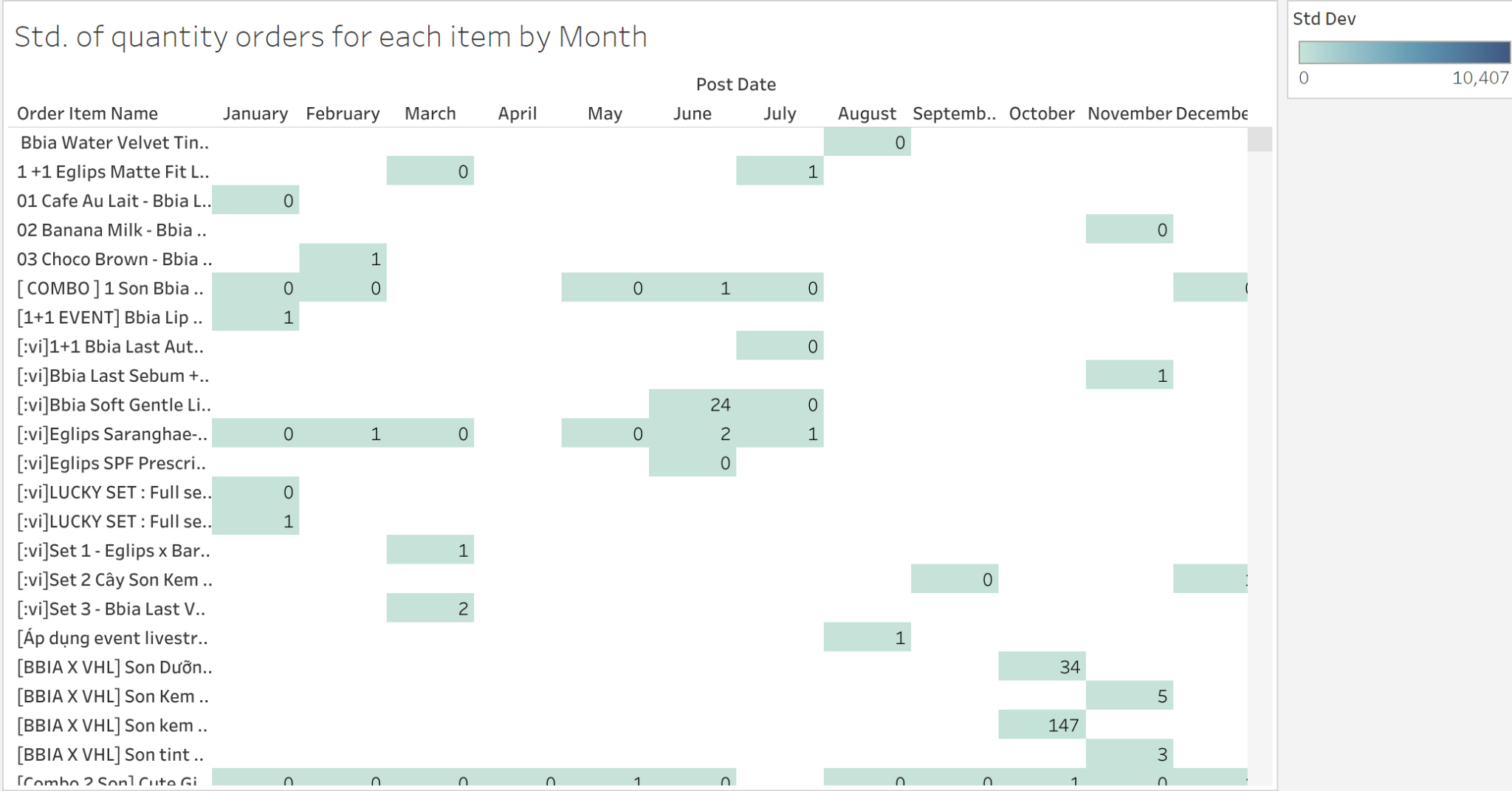
*Figure 3: Avg. Scores of Anomalies prediction*

=> table show exact anomaly detection scores performed my isolation forest algorithm



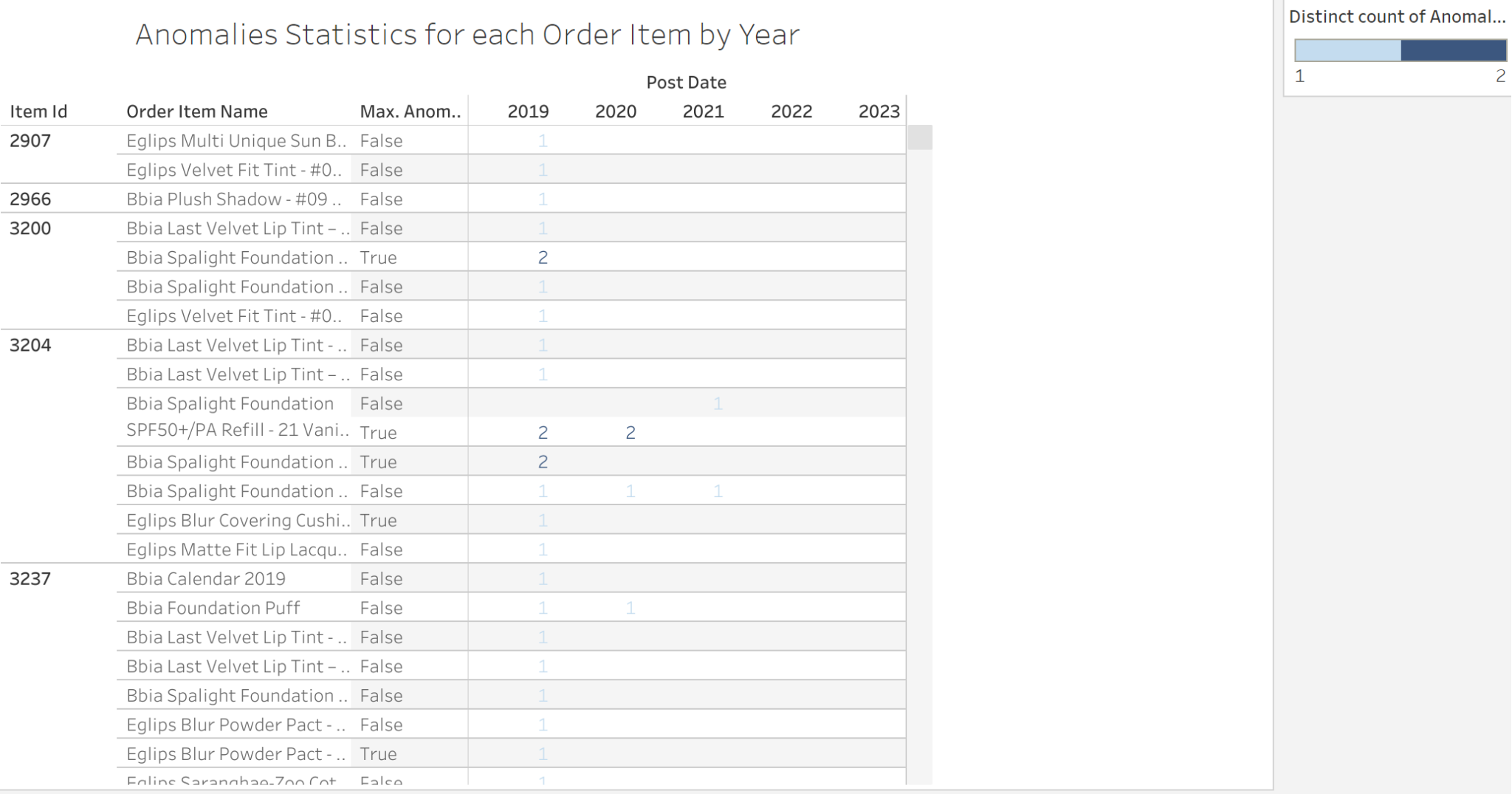
*Figure 4: Sum of moving avg. by Year*

=> We can see that 2023 illustrated the large volume of moving average, meaning there might be a significant change in quantity orders for each item (spike days), followed by 2022 with 416,317 and 459,448, respectively.



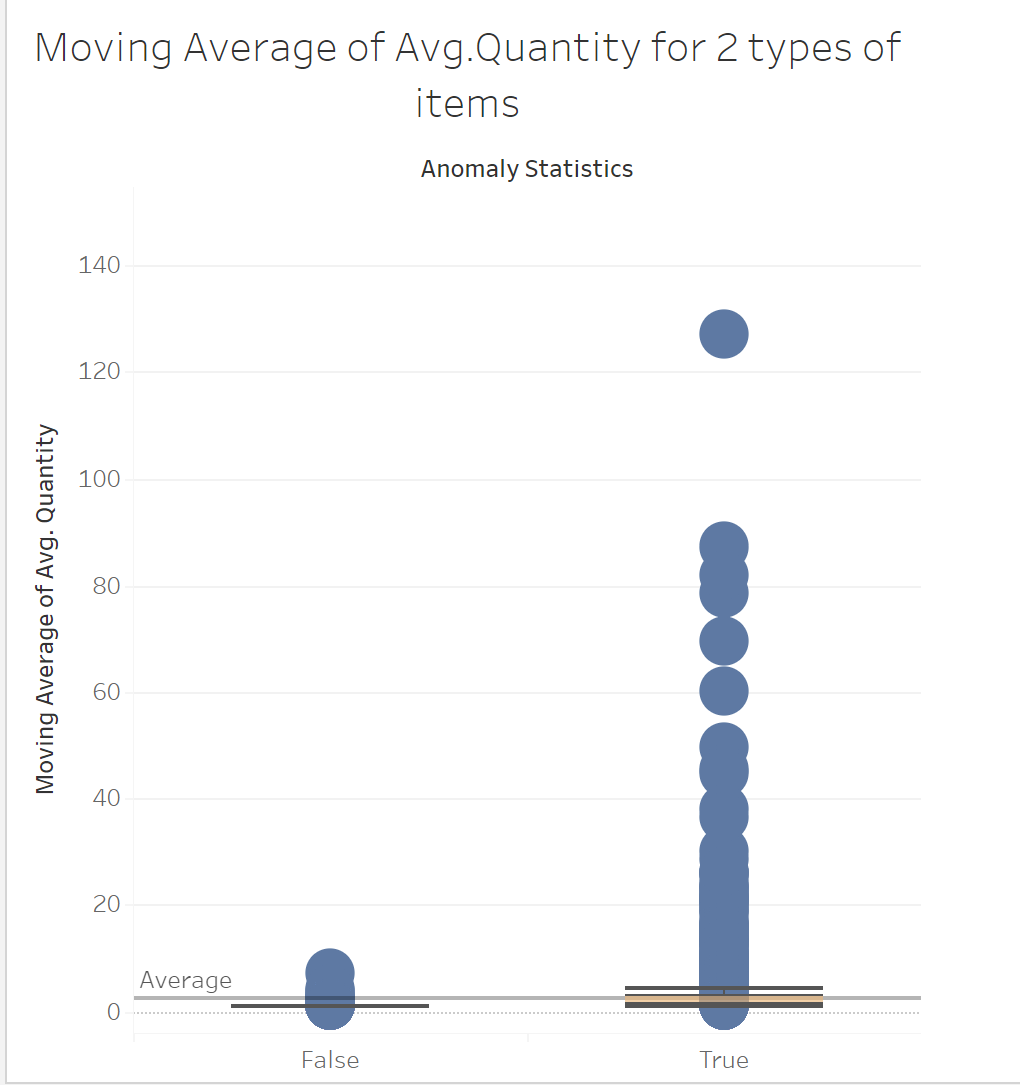
*Figure 5: Standard deviation of quantity orders for each item monthly*

We can conclude that in the middle of the year and at the end of the year, standard deviation of quantity orders appears more often => reasonable because of holidays, sale day,,...



*Figure 6: Anomalies Statistics for each Order Item by Year*

=> This table count number of anomalies detected for each item recently from 5 years ago (2019) to this whole year.



*Figure 7: Moving Average of Avg. Quantity for 2 types of items*

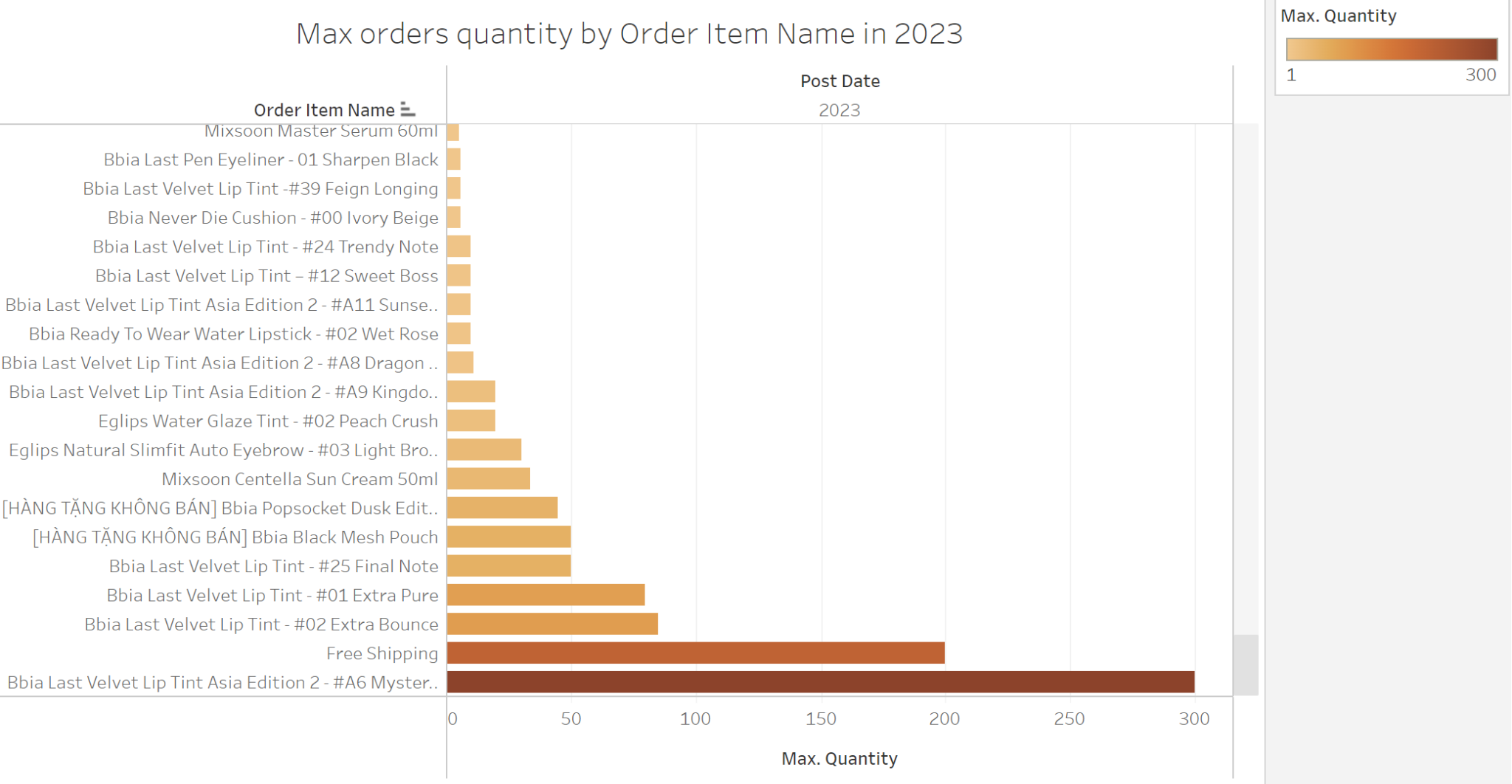
There are 2 types of items defined in statistical method:

**False**: Detected items having normal quantity orders

**True**: Detected items having anomaly patterns

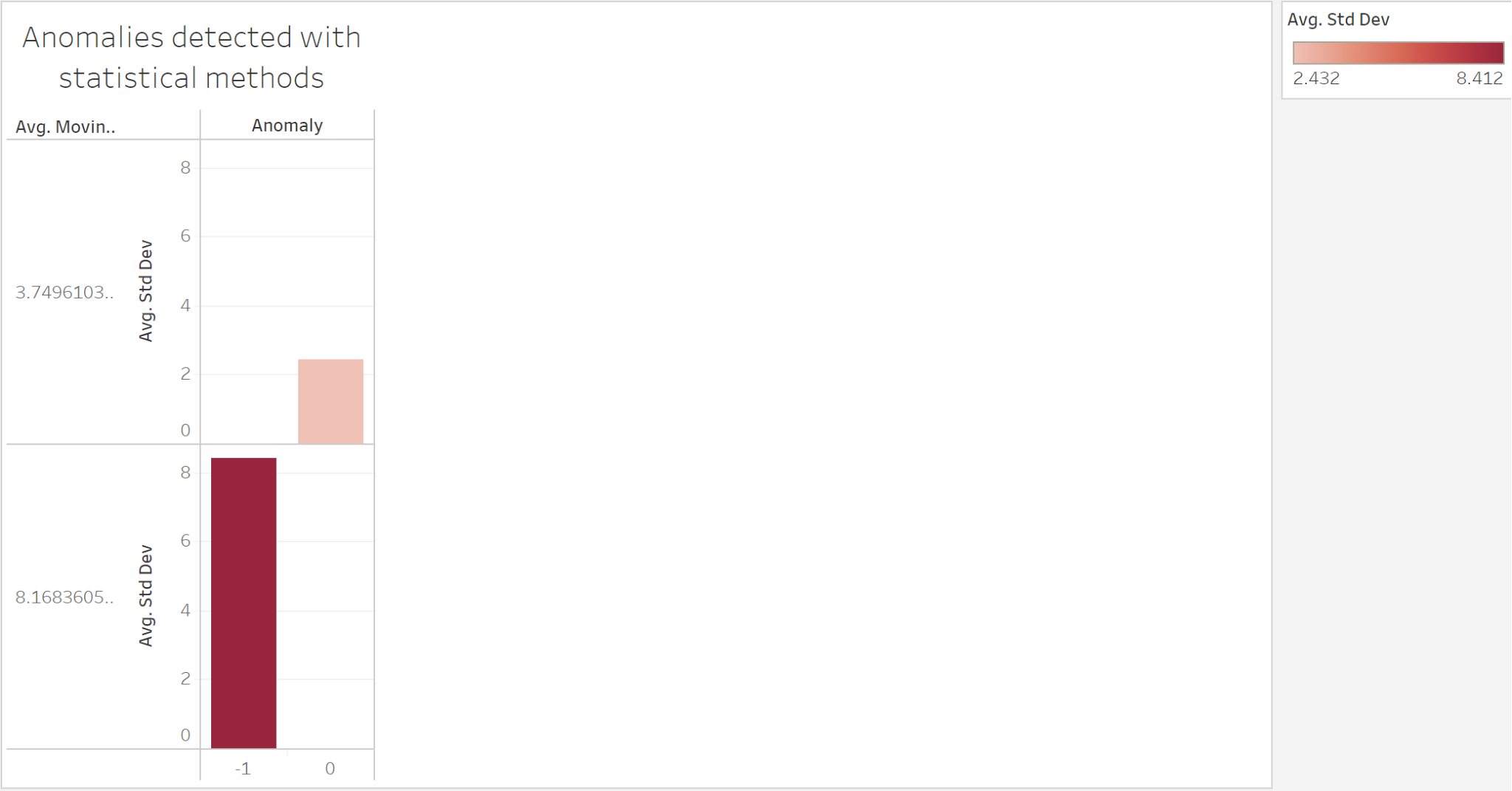
=> There are more outliers in the second group compared to the 1st group

=> average of quantity orders in anomalies group can be also seen higher.



*Figure 8: Distribution of max orders quantity sorted by ASC order, order item name in 2023*

=> This bar chart showed a better view of quantity orders by each item leading to anomaly detection. Item named as ‘Bbia Last velvet Lip Tint Asia Edition 2…’ has max quantity orders (300 in detail)



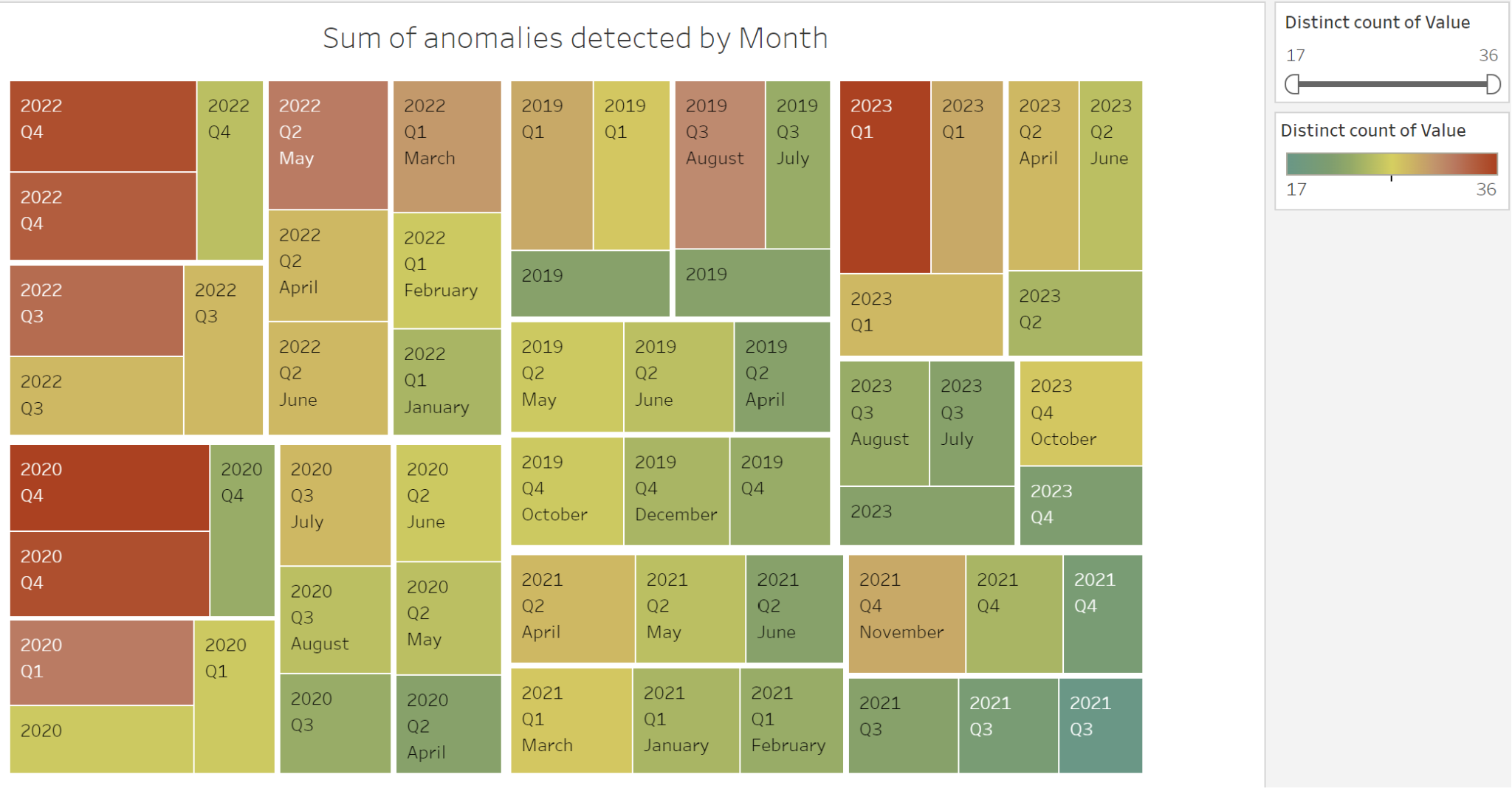
*Figure 9: Anomalies detected with statistical methods*

**2.2. Detection module (Time Series Analysis)**

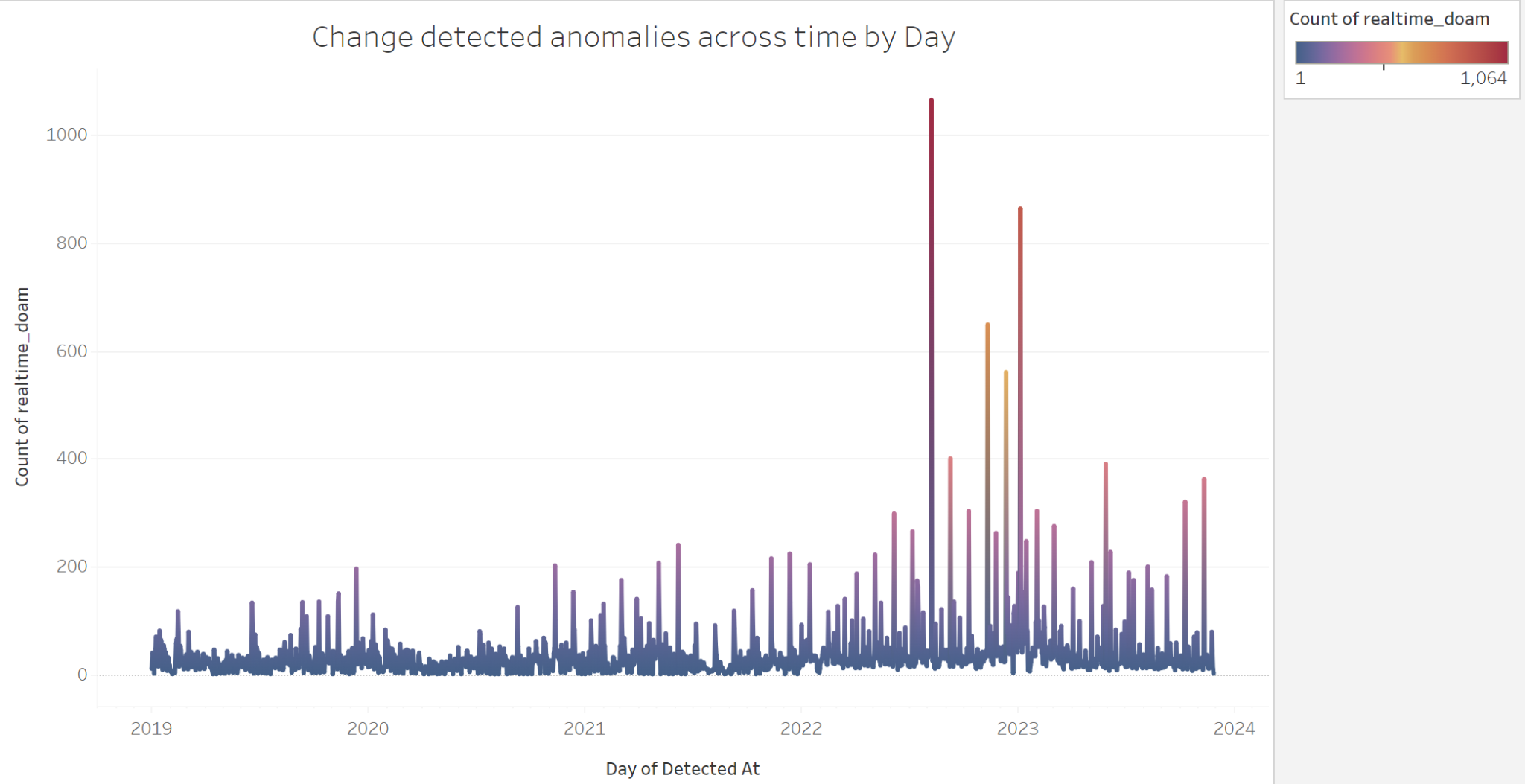
- Data Preprocessing

- Model Selection and Training: Choose a time series model, like ARIMA, and train it with historical data.

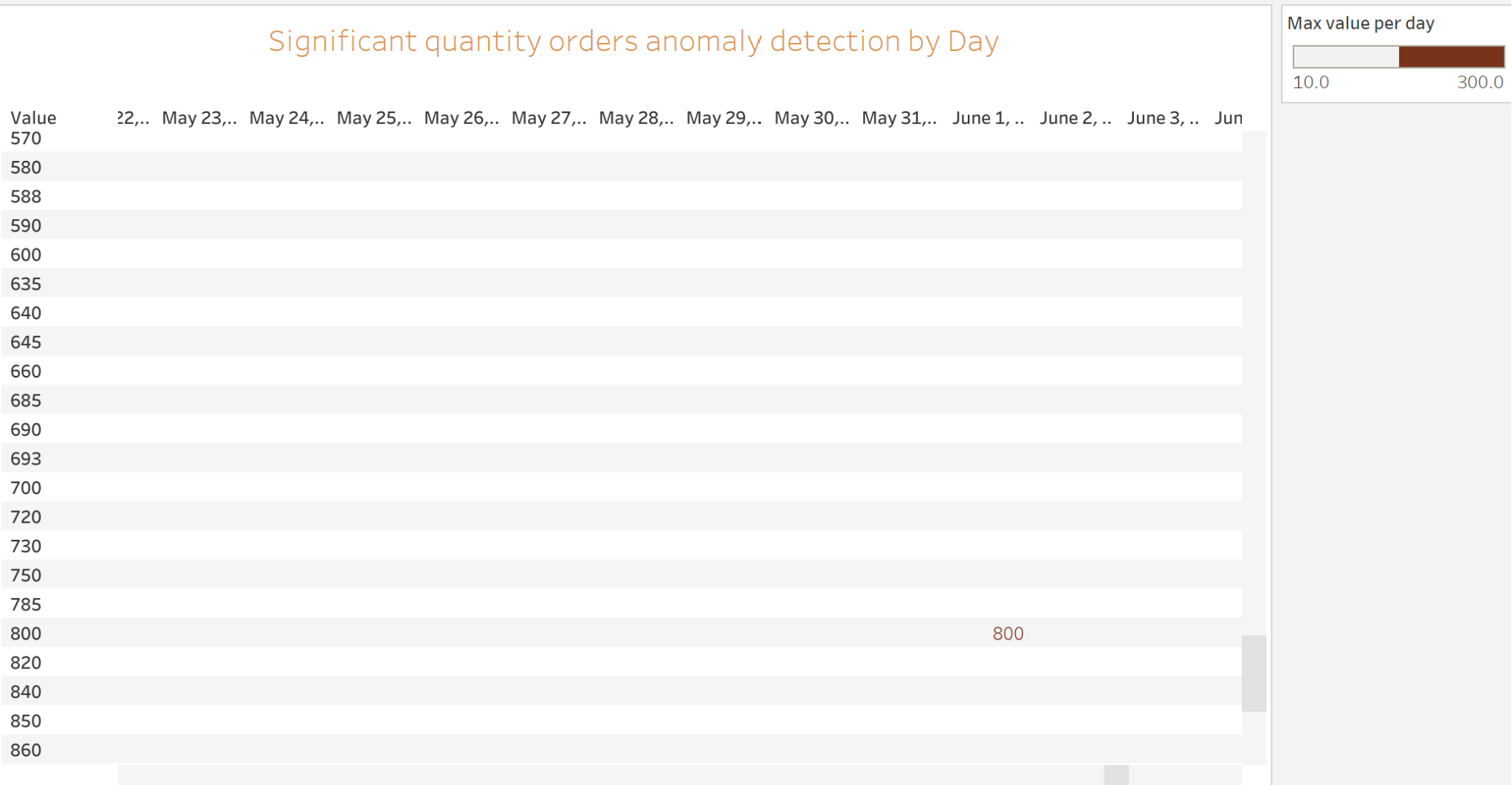
- Prediction and Detection: Use the model to make predictions. If recent order quantities significantly deviate from these predictions, it may indicate an anomaly.



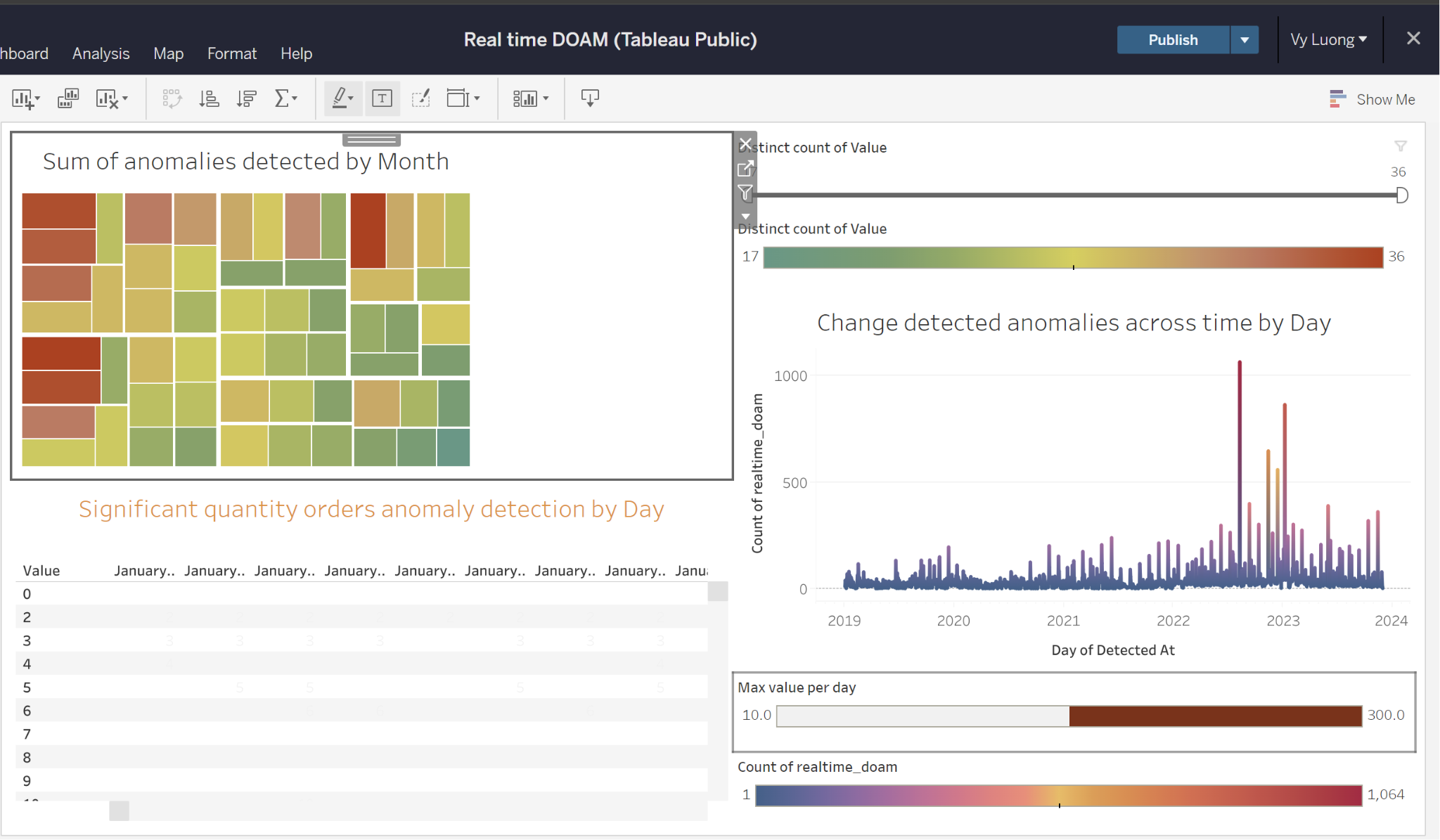
*Figure 9: Sum of anomalies detected monthly*



*Figure 10: Change movement of detected anomalies across time by Day*



*Figure 11: Significant quantity orders anomaly detection by Day*



*Figure 12: Realtime DOAM Dashboard*

2.3. Connect the module to skyadmin system: connected database from the beginning to now

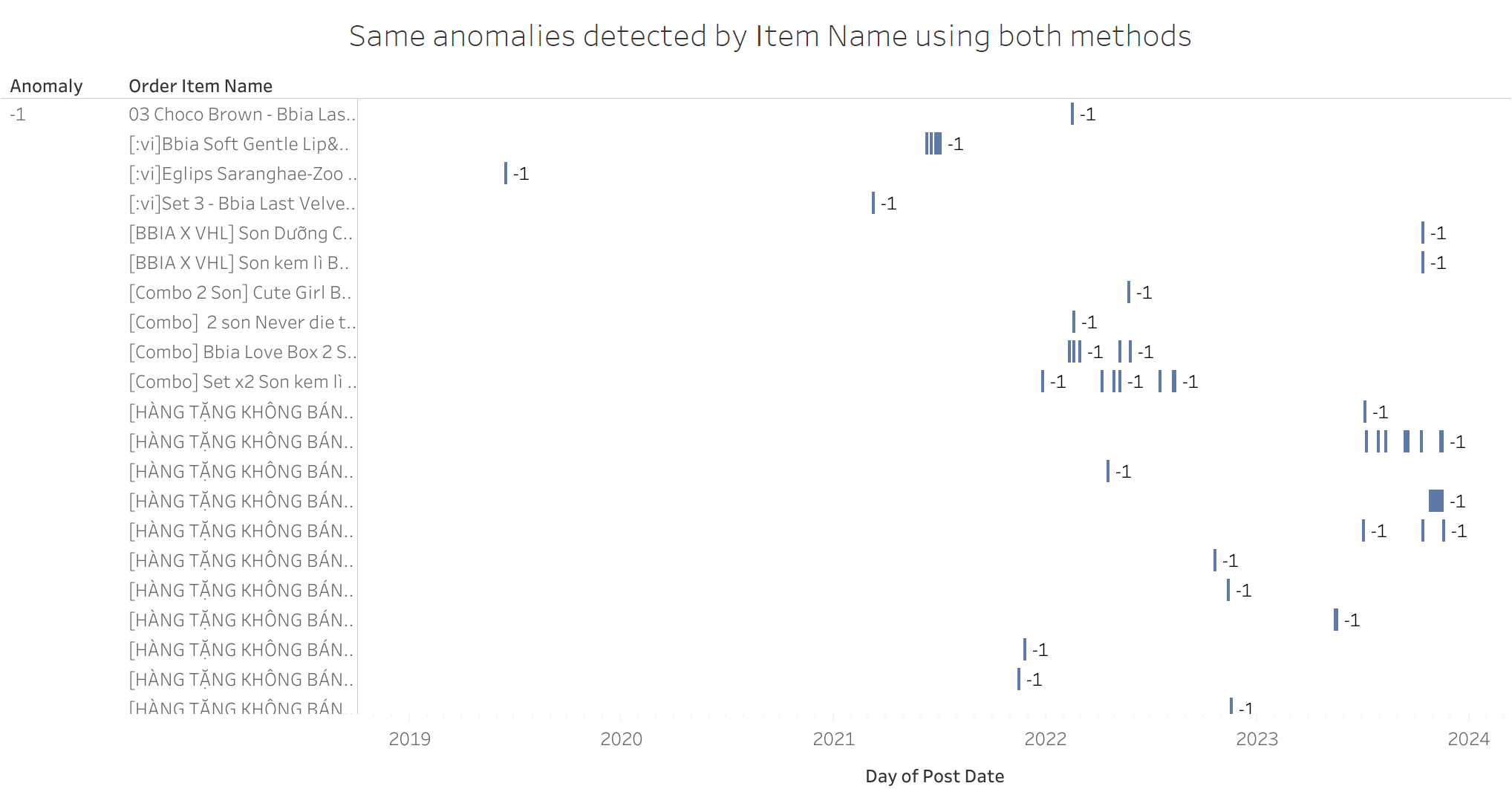
2.4. Real-time Data Streaming and Analysis : Leverage real-time data streaming to continuously collect and analyze data, enabling immediate detection of unusual order volumes. Tools like Apache Kafka

- Stream order data in real-time

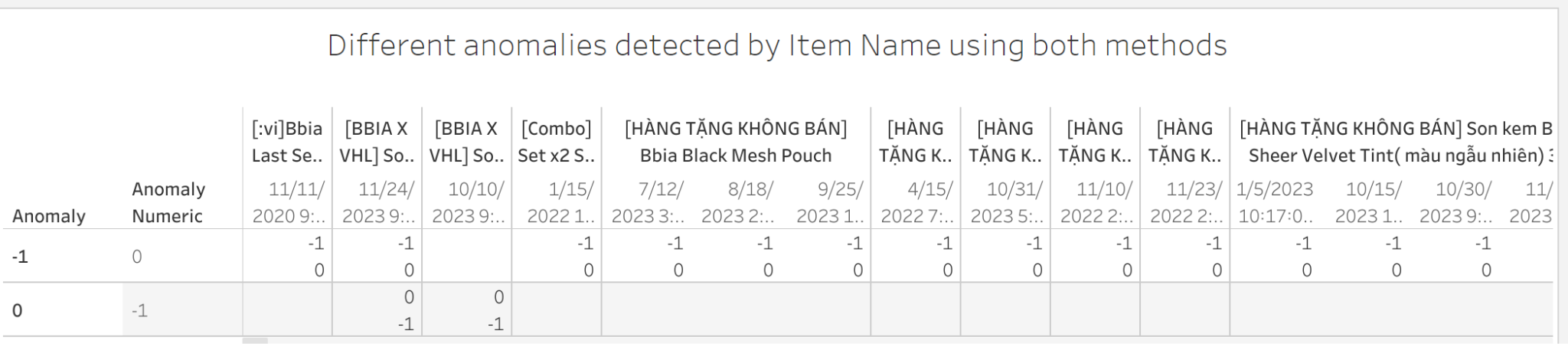
- Apply statistical methods such as moving averages, standard deviation, or time series analysis to the streaming data to detect anomalies.

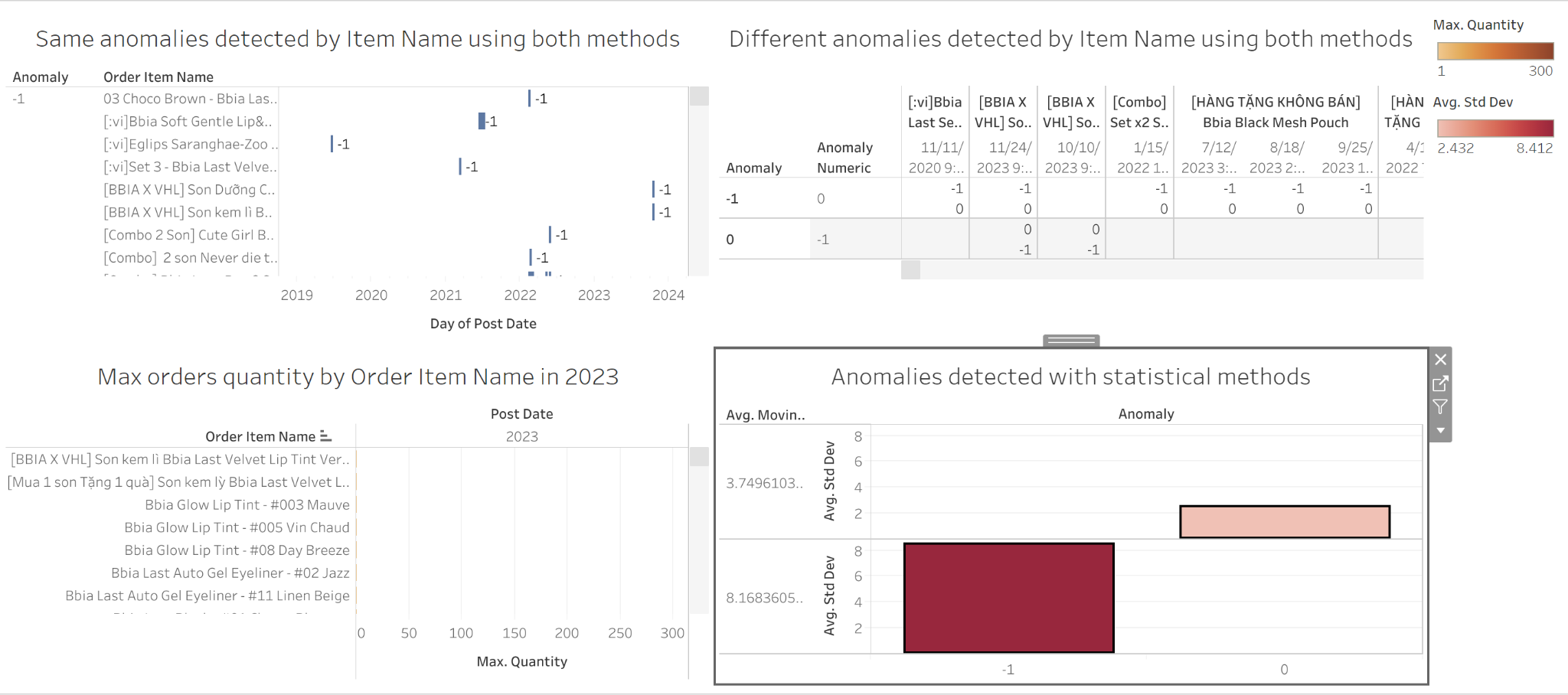
- Generate instant alerts when the order volume significantly exceeds the average.

=> Compare 2 methods: hybrid approach



*Figure 13: Same anomalies detected by Item Name using hybrid methods*





*Figure 14: Same and different detected patterns using both methods*

**III. Predictive methods**

**3.1. Forming the data science problem**

**3.2. Data Preprocessing**

Data set has 4 columns ‘item\_id’, ‘quantity’, ‘post\_date’, ‘order\_item\_name’ with 1,143127 rows

1. **Data Normalization**: remove missing rows with missing value, scale data
2. **Feature Selection**: group by Item\_id and Day to get data quantity orders for each item with exact datetime => 1 feature used for training model is ‘quantity’
3. **Model Building:**

Choose Isolation Forest algorithm for real time anomaly detection

Research: <https://docs.google.com/document/d/1P5kjjp17j4wzM5AFf2zAKya6i7G7r9nQ1548uUc_L10/edit>

***Set 4 parameters:***

**n\_estimator=50** => set the number of base estimators/trees in ensemble is 50

**max\_samples=’auto’** => the number of samples to be drawn to train each base estimator

**contamination=float(0.1**) => expected proportion of outliers in the data set is 10 => used when fitting to define the threshold on the scores of the samples => default value is ‘auto’ => the threshold value will be determined as in the original paper of IF

**max\_feature=1.0** => all the base estimators are not trained with all features available in the dataset => the number of features to draw from the total features to train each base estimator or tree => default value of max features is 1.

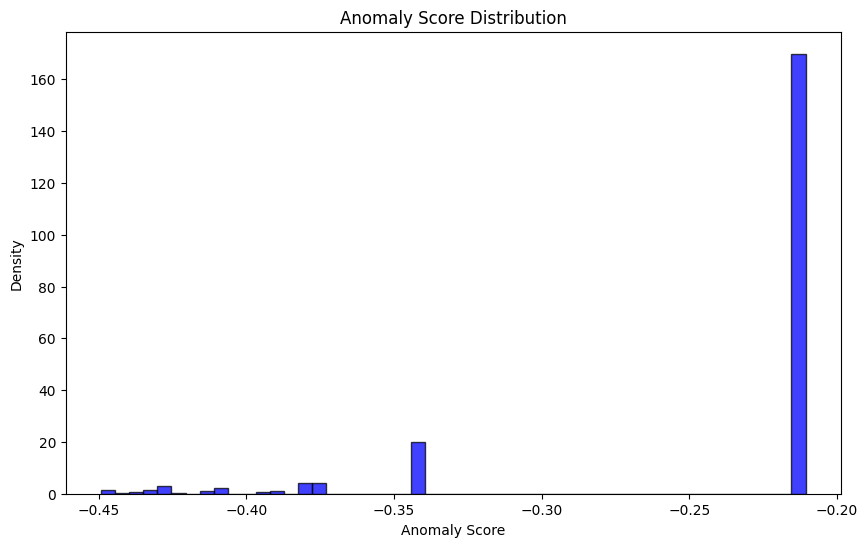
1. **Model Prediction**: having additional scores and anomaly column

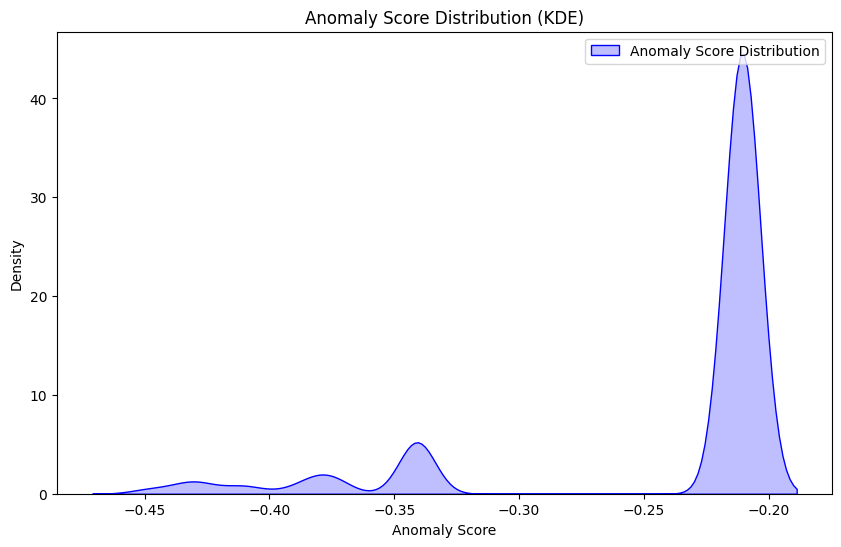
=> find out the values of scores column by calling decision\_function() of the trained model and passing col values as parameter

Each data point in the train set is assigned an anomaly score by this algorithm

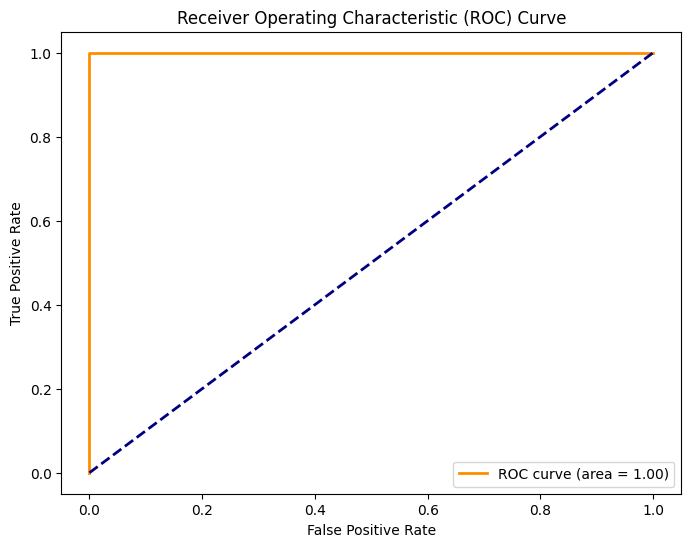
=> define a threshold, & using the anomaly score

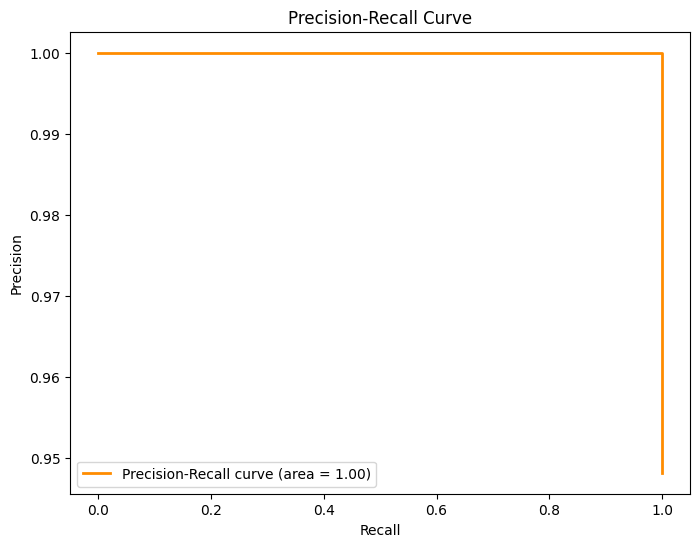
=> may be possible to mark data point as anomalous if its score > the predefined threshold after adding the scores & anomalies for all the rows in the data => Print Anomalies





Evaluating the model => find out the number of outlier present in the data





1. Detected anomalies output (saved in gg sheet, if having new data entries => rerun code => clear and rewrite output in this file)

**IV. Conclusion & Result Recommendation**

* Isolation Forest is the best algorithm for detect real time anomalies because it can capture both small & significant increase of quantity orders even if they're not overall bestsellers
* Using statistical methods such as std & moving average give lower accuracy => manual set threshold from perspective and experiences => can capture only small changes in specific time, need Empirical Adjustments => difficult applied to data streams
* Using hybrid methods to compare similar results with items have quantity orders are normal or not in specific times bring about best result => adjust threshold with anomalies performed by Isolation Forest