JOHN LEWIS
& PARTNERS

Unsupervised Anomaly Detection:

An Investigation of Techniques for Anomaly Detection in Temporal Data

Nadine Mustafa Alkaragholi

A dissertation submitted in partial fulfillment of the requirements
for the MSc in Business Analytics

21st August 2023

Abstract

This dissertation delves into the realm of anomaly detection, with a focus on analyzing unlabeled multivariate retail data sourced from John Lewis. Employing a combination of statistical, machine learning, and deep learning methodologies, the primary objective of this study revolves around pinpointing anomalies, thereby fostering an avenue for substantial improvements in the operational landscape of John Lewis. The study conducts a comprehensive investigation into the performance of four distinct anomaly detection methodologies: Autoregressive Integrated Moving Average (ARIMA), Change Point Detection (CPD), Density Based Spatial Clustering with Noise (DBSCAN), and Convolutional Autoencoder (CAE). The traditionally supervised ARIMA model is adapted to function within an unsupervised learning framework. Notably, the experimentation reveals that the DBSCAN model emerges as the frontrunner in terms of effectiveness, outshining its counterparts. This can be attributed to its precision in identifying anomalies at a granular level, as well as its applicability in various business contexts. In lieu of these findings, the DBSCAN model is selected and was subsequently deployed across the entirety of the dataset. This tailored approach aligns seamlessly with the intrinsic nature of the data and resonates with the specific requirements of John Lewis.

The main Juypter notebook and the .csv file with the identifies anomalies can be found here: `https://drive.google.com/drive/folders/1GPkx82Ps56B1aZPGIxNkeDc4exMWxe67?usp=share_link`.

اللهُ

Acknowledgments

I would like to firstly express my heartfelt gratitude to my biggest cheerleaders, Mama and Baba, who left behind their homeland and sacrificed everything in the pursuit of my hopes and dreams. Your tireless work and unconditional support will never go unnoticed. I am only capable because you believe in me, I hope I've made you proud.

To my soul sisters Ayah and Sarah whose love has been unwavering despite our great geographical distance. Thank you for always picking up the phone whenever I needed to laugh or cry. You've shown me that true friendship is boundless and I am forever grateful for you.

To Adnan, Zeinab, and Manar, you guys are my second set of parents and I think you deserve an acknowledgement here. Thank you for helping me with my math homework, baby sitting me, sending me simsim content, listening to my gossip, making me corn cheese dip whenever I come home, and fixing my resume every week.

To the best friends that I've made in this program, who have carried me through this masters. Thank you for making sure I'm working hard, having fun, ate dinner and got home safe. I would not have survived this masters without you, I love you always.

Lastly, I am deeply grateful to Sai Pandian for his mentorship throughout the course of this dissertation. Your consistent and invaluable guidance has been instrumental in shaping my work. Your willingness to provide support has been an anchor, and I genuinely appreciate your commitment to my success.

Contents

List of Figures

1 Introduction

1.1 Motivation

Many businesses struggle with the daunting task of managing and monitoring the performance of their distributed systems, given the myriad of factors that need to be observed. Unfortunately, errors that can lead to financial losses often go unnoticed, causing significant harm to the business (Raval, 2021). This is where anomaly detection, powered by machine learning techniques, can play a crucial role in identifying and addressing these errors promptly.

Anomaly detection, in essence, involves the use of data mining methods to pinpoint events or observations in a dataset that deviate from normal behavior (Jose, 2023). These anomalous data points, such as shifts in consumer behavior, can serve as indicators of critical incidents or potential opportunities (ibid). To effectively detect anomalies, it is essential to leverage time series data, which encompasses historical information that can facilitate accurate predictions for the future (Chandola, 2009). By forecasting future anomalies, anomaly detection systems can proactively notify businesses and enable timely intervention.

Automating the process of anomaly detection necessitates finding the right combination of supervised and unsupervised machine learning methods. Adopting a hybrid approach ensures comprehensive consideration of all relevant factors within an organization. (Raval, 2021)

1.2 Company Overview

John Lewis Partnership plc is a prominent entity that owns and operates a chain of department stores, supermarkets, and convenience stores across the United Kingdom. Through its brands, John Lewis and Waitrose, the company offers a diverse range of products spanning from grocery items to home appliances (GlobalData, n.d.). However, John Lewis, like many other retailers, faces a significant challenge regarding stock file accuracy. The absence of precise systems to track available stock quantities, schedule replenishments, measure product availability, and manage waste poses various problems. By being able to identify outliers in stock file accuracy, John Lewis can address these issues effectively and mitigate their impact on sales revenue and overall profitability.

1.3 Aim and Objectives

This study aims to investigate and compare different anomaly detection models using unlabeled, multivariate data provided by John Lewis. The primary objective is to develop a robust framework that not only identifies anomalies in historical data but also has the capability to predict future anomalies. By leveraging statistical approaches and advanced machine learning techniques, this study aims to identify the most effective model for detecting anomalous data points, enabling timely alerts to John Lewis regarding errors and potential opportunities.

The focus of this study is to address the challenges faced by John Lewis due to inaccuracies in data related to stock file management. By

analyzing and understanding the patterns and outliers within the data, the anomaly detection framework aims to enhance the accuracy of stock tracking, replenishment scheduling, availability measurement, and waste management processes.

To achieve the research objective, various anomaly detection models have been evaluated and compared. These models include supervised, unsupervised, and hybrid approaches. Each model was trained and tested on the provided data from John Lewis, allowing for a comprehensive assessment of their performance in identifying anomalies

By selecting the most suitable model based on its performance in identifying anomalies, John Lewis can benefit from enhanced decision-making processes and mitigate potential financial losses caused by errors in stock management. Thus, improving the overall operational efficiency and profitability of John Lewis through the implementation of an effective anomaly detection framework.

1.4 Structure

This study is structured into nine distinct sections. The first section initiates by presenting the study's motivation, introducing the company, and outlining the objectives. Subsequently, the second section encompasses a comprehensive literature review that elucidates the concept of anomaly detection. It also delves into an exploration of various approaches, spanning statistical, machine learning, and deep learning techniques. The third section proceeds to introduce the dataset through exploratory data analysis.

This is accompanied by a discussion on data cleaning and the process of feature engineering. The fourth section then delves into the methodology; firstly discussing data preprocessing with dimensionality reduction, and secondly discussing the selected unsupervised learning techniques employed for anomaly detection. Transitioning, the fifth section undertakes the task of evaluating the models' performance. Subsequent to this evaluation, the sixth section offers an insightful discussion of the findings, highlighting the most effective model for full implementation. Next, the seventh section delineates the study's limitations, while the eighth section provides insight into potential avenues for future research. Finally, the ninth section delivers a decisive conclusion, underscored by the study's contributions to John Lewis and the anomaly detection field at large.

2 Literature Review

2.1 Anomalies and Anomaly Detection

What is an anomaly and what is anomaly detection?

An anomaly is generally characterized as patterns within data that deviate from a well-defined notion of normal behavior (Géron, 2019). The identification of anomalies in data can provide crucial actionable insights across various domains, underscoring the significance of anomaly detection. This process, known as anomaly detection, involves detecting patterns or deviations in data that do not adhere to the expected or typical behavior (Abraham, 1989). Anomaly detection finds widespread application in numerous fields, including fraud detection, cyber-security intrusion detection, and retail operations. The pursuit of anomaly detection techniques has a long history within the statistics community, dating back to the 19th century (Edgeworth, 1887). Over time, research groups have developed and refined various methods for anomaly detection, which have subsequently been adapted by different application domains (ibid).

Noise removal, novelty detection and anomaly detection

Before delving into the various techniques used in anomaly detection, it is crucial to grasp the nature of this field. Although noise removal and anomaly detection share a connection, they are distinct concepts with different purposes (Teng et al., 1990), both dealing with unwanted noise in the data (Rousseeuw and Leroy, 1987). Noise in data refers to information that lacks relevance to the analysis and can impede its progress (Chandola, 2009).

Removing noise from data is a common practice to eliminate unwanted elements that may introduce bias or other issues (Huber, 1974). Similarly, novelty detection also pertains to anomaly detection, aiming to identify emerging patterns within the data (Markou and Singh, 2003). However, the key distinction lies in the fact that novel patterns, once detected, can be considered part of the normal behavior (ibid).

Challenges of anomaly detection

Anomaly detection, while conceptually straightforward in detecting deviations from normal behavior, encounters numerous nuanced challenges. One notable hurdle involves defining the boundaries that encompass the normal region, accounting for all possible normal behaviors (Chandola, 2009). Often, the demarcation between normal and anomalous behavior lacks precision. Additionally, normal behavior undergoes constant evolution across diverse domains, rendering historical data potentially inadequate for predicting future patterns (ibid). Another common difficulty in anomaly detection revolves around the availability of labeled data required for training and validating used in machine learning techniques (Boriah, 2008). As previously mentioned, the distinction between noise and anomalies can be blurred, further complicating the differentiation between the two (ibid). In essence, the characteristics of the problem can be summarized as the nature of the data, availability of labels, anomaly types, and desired output.

The types of anomalies

Understanding the various types of anomalies is pertinent in determining the appropriate approach for anomaly detection. The simplest form of

anomaly is known as Point Anomalies, wherein an individual data instance deviates significantly from the rest of the dataset (Géron, 2019). Moving beyond individual instances, Collective anomalies refer to a group of related data points that exhibit anomalous behavior when considered as a whole, even if the individual points may not be considered anomalies on their own (Goldberger et al., 2000). Finally, Contextual Anomalies play a pivotal role in this study. These anomalies are data instances that exhibit abnormal behavior within a specific context (Song et al., 2007). Since this study involves the analysis of time series data, time itself serves as a contextual attribute, determining the position of the anomaly within the sequence (ibid).

Unsupervised anomaly detection

While supervised models excel in accuracy, they often prove inefficient for anomaly detection tasks due to the scarcity of labeled data (Ren et al., 2019). Hence, unsupervised techniques that don't require labeled training data are more widely applicable. Unsupervised techniques operate under the assumption that normal instances are more prevalent in the test data (Géron, 2019). However, if this assumption does not hold true, unsupervised methods may yield a high false alarm rate (ibid). To address these challenges, semi-supervised techniques can be employed, which combine unsupervised methods with a subset of labeled data for training. This adaptation assumes that the test data contains relatively few anomalies, and the model trained on the labeled data remains robust against them (Chandola, 2009). By utilizing these flexible approaches, anomaly detection algorithms can be better equipped to detect and distinguish anomalous instances while minimizing false alarms, thus enhancing their overall effectiveness.

2.2 Business Application of Anomaly Detection

In the business context, anomaly detection is used for fraud detection, network security, predictive maintenance, quality control, and identifying outliers in financial transactions. In real-world scenarios, it finds applications in healthcare monitoring, intrusion detection in smart homes, environmental monitoring, and anomaly detection in sensor networks. Its ability to identify anomalies across different domains has proven instrumental in improving efficiency, minimizing risks, and enhancing safety and security. Ongoing research in anomaly detection continues to drive innovations that benefit a wide range of industries and sectors.

A noteworthy example of anomaly detection in a business application is the study conducted by Kuo and Tsang titled "Detection of price manipulation fraud through rational choice theory: evidence for the retail industry in Taiwan". In this research, the authors utilized labeled data from a retail store in Taiwan, which included instances of labeled fraud transactions. The availability of labeled data allowed the researchers to employ supervised learning methods, namely Bayesian network, Logistic Regression, and Random Forest. The analysis results demonstrated that all the algorithms achieved accuracy scores exceeding 0.8, indicating successful classification and prediction of anomalies with commendable performance. This study serves as a compelling illustration of how anomaly detection techniques can effectively detect and address fraudulent activities in the retail industry, offering valuable insights for fraud prevention and risk management.

Another relevant example of anomaly detection in a business application is Microsoft's introduction of the Azure Anomaly Detector API (Xing, 2019). This powerful anomaly detection service was specifically developed to monitor vast amounts of metrics originating from Bing, Office, and Azure (Ren et al., 2019). The algorithm behind the service employs two unsupervised learning methods to model the unlabeled time series data. Firstly, the Spectral Residual (SR) model utilizes visual saliency detection to transform univariate seasonal data into a saliency map, magnifying the presence of anomalies (Xing, 2019). Secondly, a Convolutional Neural Network (CNN) is trained using synthetic data generated by randomly selecting points in the saliency map and calculating injection values to replace the original points (ibid). The purpose of the CNN is to replace the SR model's single threshold with a CNN discriminator, resulting in enhanced anomaly detection (Ren et al., 2019). This innovative technique has yielded exceptional results, enabling product teams to expedite issue detection, reduce manual efforts, and accelerate the diagnostic process overall (ibid). The Azure Anomaly Detector API exemplifies how anomaly detection methodologies can be effectively integrated into business workflows, improving operational efficiency and decision-making processes.

2.3 Methods of Anomaly Detection

Statistical methods of anomaly detection

Anomaly detection techniques can be categorized into three tasks based on the nature of the data: supervised, unsupervised, and semi-supervised. While anomaly detection is typically considered an unsupervised task due to

the lack of pre-labeled anomalies, there are instances where anomalies are labeled within the dataset. In such cases, supervised learning techniques can be adapted for unsupervised learning tasks in anomaly detection scenarios.

The field of anomaly detection offers a wide range of approaches. Distance-based methods encompass neighbor-based, density-based, and clustering techniques. Statistical methods include histogram-based outlier scores and principal component analysis. Classification techniques such as one-class SVM and isolation forest are also utilized. Additionally, angle-based methods, such as angle-based outlier detection, have gained attention (Jaber, 2023).

Among the notable approaches to anomaly detection, principal component analysis, ARIMA forecasting, change point detection, DBSCAN, and neural network autoencoders stand out. These methods have been employed both individually and in combination with other models to effectively detect anomalies in various types of data. Given the specific characteristics of the data provided by John Lewis, these particular models are highly relevant and applicable. Therefore, this review aims to delve into these specific models, offering a comprehensive understanding of their use cases, relevance and applicability to the study at hand.

The ARIMA model, introduced by Box et al. (2008), is a supervised learning approach for time series modeling and forecasting. Comprising autoregressive (AR), moving average (MA), and integrated (I) components, ARIMA captures dependencies on lagged observations, models short-term fluctuations via moving averages, and employs differencing to transform data

into a stationary series by eliminating trends and seasonality (ibid). Adapting the ARIMA model to unsupervised learning tasks, such as anomaly detection, can provide valuable insights into unusual patterns or outliers in time series data. By leveraging its components, the ARIMA model enables the identification of anomalous events or behaviors that deviate significantly from the expected patterns.

The ARIMA model, when applied to anomaly detection, involves comparing actual observations with predictions generated by the model. Instances where the observed values significantly deviate from the expected values are identified as potential anomalies. Moschini et al. (2021) utilized this approach to tackle the issue of credit card fraud detection. In their study, the researchers employed the ARIMA model to capture the regular spending patterns of customers. By fitting the model to the customers' normal spending behavior, they were able to detect potential fraud by identifying any discrepancies. To detect fraud in the testing set, Moschini et al. (2021) calculated errors by comparing the predicted and actual daily transaction counts. If the Z-Score exceeded a predefined threshold, the corresponding day was flagged as anomalous. The objective of this approach was to create a model of the customers' regular spending behavior, enabling any deviations from it to be considered potential anomalies. The ARIMA model's forecasting capabilities, enabled the researchers to detect and flag suspicious credit card transactions effectively (ibid).

Change point detection is another statistical approach used for anomaly detection, focusing on identifying abrupt variations in time series data. These abrupt changes often indicate transitions or anomalies in the underlying sys-

tem (Kawahara and Sugiyama, 2009). Change point detection algorithms can be classified into two categories: online and offline (Downey, 2008). Offline algorithms consider the entire dataset as a whole, scanning the historical data to detect any changes that occur. On the other hand, online algorithms run concurrently with the monitored process, observing each data point as it becomes available and detecting change points as soon as they occur (ibid).

Among the various frameworks for change point detection, the graph-based framework stands out as particularly relevant for this project. It employs a non-parametric approach that applies a two-sample test on an equivalent graph to determine the presence of a change point within the observations (Friedman and Rafsky, 1979). In this framework, a single window is derived from the time series data, and a change point is reported if it exists within this window (Aminikhanghahi and Cook, 2016). The graph serves as a representation of the time series, with each data point represented as a node, and the edges capturing the relationships between adjacent data points (ibid). Moreover, a search based framework is also particularly relevant in this study. A search based method is a technique used to efficiently solve problems by breaking them down into overlapping subproblems and reusing their solutions (Truong et al., 2020).

Machine Learning methods of anomaly detection

DBSCAN, introduced by Ester et al. (1996), is an unsupervised clustering distance-based approach widely utilized for anomaly detection. It stands out as a powerful algorithm capable of identifying anomalies in large datasets. Unlike other algorithms, DBSCAN not only performs clustering

but also defines anomalies within the data series (ibid). In distance-based approaches such as DBSCAN, the distances between data points play a crucial role in anomaly detection. DBSCAN operates based on two user-defined parameters: epsilon (neighborhood distance) and minpts (minimum number of points). A cluster is formed if the number of neighboring points around a given point exceeds minpts (Celik and Dokuz, 2011). By utilizing these parameters, DBSCAN labels data points as core points, border points, or outliers (anomalous points) (ibid).

What sets DBSCAN apart from typical clustering approaches is its ability to identify outliers that do not fit into any clusters (Ester et al., 1996). This distinction is crucial because traditional clustering methods, such as K-means clustering, often struggle to identify such anomalies. The limitation arises from the fact that K-means requires pre-setting the number of clusters, which may hinder its ability to detect outliers that do not conform to the predetermined cluster structure (Géron, 2019). Celik and Dokuz (2011) argue that DBSCAN outperforms statistical approaches in identifying anomalies, as statistical methods tend to focus solely on extreme values based on predefined thresholds. In contrast, DBSCAN has the capability to uncover anomalies even if they do not exhibit extreme values (ibid). Overall, DBSCAN serves as a robust clustering distance-based algorithm that effectively detects anomalies, making it a valuable tool for anomaly detection tasks, particularly when dealing with large datasets or anomalies that do not conform to typical extreme value patterns.

Deep Learning methods of anomaly detection

Anomaly detection tasks are inherently complex and depend on various factors, such as the data characteristics and the nature of anomalies themselves. As previously mentioned, considerations like the data's multivariate nature or the presence of labeled anomalies impact the selection of suitable models or combinations of models. Consequently, since each dataset is unique, no two deep learning approaches to anomaly detection are identical, adding to their inherent complexity.

One captivating and intricate approach to anomaly detection involves the utilization of neural networks, which have the capability to identify the optimal subspace, capturing non-linear correlations between features (Hinton and Salakhutdinov, 2006). In their research, Chen and Lee (2018) introduce an anomaly detection system based on autoencoders, leveraging their effectiveness in dimensionality reduction. While Principal Component Analysis (PCA) is commonly employed for dimensionality reduction and feature extraction (Shyu et al., 2003), the researchers argue that PCA's linear transformation falls short in capturing non-linear correlations between features. To address this limitation, they propose the use of an autoencoder, specifically a Convolutional Autoencoder (CAE), which efficiently reduces the number of parameters. By sharing parameters between convolutional and deconvolutional layers, the CAE achieves parameter reduction (Hinton and Salakhutdinov, 2006). An autoencoder works by encoding and decoding the data and creating a reconstruction error threshold; anything above or below the threshold is considered to be anomalous (Kingma and Welling, 2019). This particular research endeavor benefited from the availability of labeled data,

allowing it to adopt the autoencoder as a supervised learning approach. As a result, the researchers were able to establish ground truths and treat the anomaly detection task as a supervised learning problem Evaluation results demonstrate that the CAE-based detection method significantly outperforms other detection techniques (Chen and Lee, 2018), highlighting its effectiveness in capturing complex patterns and identifying anomalies with higher accuracy.

In a similar scenario, Ehsani et al. (2022) employed an unsupervised approach using a CAE in conjunction with a classification task for anomaly detection. The CAE, being an unsupervised model, does not depend on the event type and eliminates the need for preprocessing and explicit feature extraction, while still capturing crucial information from the input data (Hinton and Salakhutdinov, 2006). The researchers proposed coupling the anomaly detection task with a supervised classifier to identify the specific type of anomaly present. For this purpose, they employed a Convolutional Neural Network (CNN) as the supervised classifier, utilizing an ensemble learning method to create a labeled dataset (Ehsani et al., 2022).

In this approach, the CAE is responsible for detecting abnormal behavior in the distribution network by comparing it to previously learned normal conditions. It effectively handles high-dimensional data by converting complex non-linear computations into a series of simpler calculations (Masci et al., 2011). By compressing the data into a lower dimension, the CAE eliminates correlations and extracts informative features without the need for pre-processing (ibid). The detected anomalous sequences from the CAE are subsequently classified using the CNN (Ehsani et al., 2022).

Given the lack of fully labeled data to train the model, the researchers utilized bootstrap aggregation to minimize classification errors and prevent overfitting (Lee et al., 2020). This proposed model demonstrated notable advantages, allowing the detection of events regardless of their characteristics, saving time by eliminating the pre-processing step, and exhibiting robustness against small distortions (Ehsani et al., 2022). Remarkably, the CAE method successfully detected 38 out of 46 total anomalous events, with only one false alarm reported. The classification model exhibited outstanding performance, achieving an accuracy of 99.75%, precision of 97.44%, recall of 82.61%, and an F1 score of 89.41% (ibid). These impressive results highlight the effectiveness of the proposed approach in accurately identifying anomalies and classifying them despite the limited availability of labeled data.

The Unsupervised Anomaly Detection for Multivariate Time Series (USAD) approach, proposed by Audibert et al. (2020), combines the concepts of autoencoders and adversarial training inspired by GANs (Generative Adversarial Networks). The underlying idea is that through adversarial training, the encoder-decoder architecture of the autoencoder learns to enhance the reconstruction error of inputs containing anomalies, thus improving anomaly detection (Audibert et al., 2020). This approach offers greater stability compared to methods solely relying on GAN architectures.

Traditionally, autoencoders are used for anomaly detection by measuring the reconstruction error as an anomaly score (Sejnowski and Hinton, 1999). However, they may struggle to detect small anomalies that closely resemble normal data, as the autoencoder aims to imitate normal data closely (ibid). GANs can address this limitation by employing a discriminator that

acts as an anomaly detector (Goodfellow and Mirza, 2014). GANs involve two networks, the generator (G) and the discriminator (D), playing a game against each other. The generator tries to create realistic data, while the discriminator aims to distinguish between real and generated data (ibid). By leveraging the discriminator, GANs can overcome the challenge of detecting subtle anomalies close to normal data.

However, GANs often encounter issues such as mode collapse and non-convergence due to imbalances between the generator and discriminator (Goodfellow and Mirza, 2014). In the USAD approach, these issues are mitigated by integrating the autoencoder architecture into a two-phase adversarial training framework (Audibert et al., 2020). This combined model can effectively identify anomalies by performing accurate reconstructions and maintain stability during adversarial training, thus avoiding problems like collapse and non-convergence (ibid).

The USAD approach yielded impressive results, achieving a precision of 98.51%, a recall of 66.18%, and an F1 score of 79.17% (Audibert et al., 2020). By combining the strengths of autoencoders and adversarial training, USAD demonstrates its effectiveness in anomaly detection tasks, surpassing the limitations of individual techniques.

2.4 Gap in the Literature

Overall, the realm of anomaly detection presents diverse options, each with its own strengths and applications. Researchers and practitioners have leveraged these methods to detect anomalies successfully, addressing the

unique challenges posed by anomalous data. However, a notable gap in the existing literature, which has not been extensively addressed, pertains to the reliance on methods that yield ground truths and the use of metrics to evaluate model performance. Much of the literature available fails to consider scenarios where the use of ground truths is inapplicable. In the case of the dataset provided by John Lewis, the unavailability of ground truths presents a significant challenge. Consequently, this study aims to adapt models in a way that allows for evaluation based on their ability to detect anomalies without relying on traditional ground truth-based metrics.

By focusing on model adaptation that can effectively identify anomalies in an unsupervised manner, this research seeks to bridge the gap in the literature and provide valuable insights into anomaly detection for datasets with limited or no labeled data. The investigation will prioritize methods that showcase their efficacy through their capacity to accurately and comprehensively detect anomalies in the absence of ground truths, ultimately contributing to a more practical and robust anomaly detection framework.

3 Data

3.1 Data Background

The main dataset provided by John Lewis consists of approximately 6.8 million rows and 17 columns. Each row represents a specific period of time and is associated with a date column labeled as "SKACDY_DAY", making it a time series dataset. The dataset includes information about various products. There are five branches of Waitrose represented in the data, and their store locations are identified by branch numbers: 1336, 1414, 1382, 1096, and 304. The product numbers are recorded under the column "NEW_LINE_NUM". To enhance understanding of the data, the data dictionary[1]is as follows:

[1]Please refer to Appendix 10.1 for the data clarifications.

Table 1: Data description

Column	**Description**
SKACDY_DAY	Date
NEW_BU_NUM	Branch number
NEW_LINE_NUM	Product identification
BUY_OFF_NAME	Product category
NEW_LAYOUT_BCODE	Placement code
CASE_SIZE	Items per case
NEW_SUPPLIER_NUM	Supplier number
LINE_SALES_SUS	Sales units per day
LINE_STOCK_SUS	Amount of stock
LINE_WSTG_SUS	Explained wastage. 0 = no wastage, negative value = wastage.
UW_OP_COUNT_SUS	Record that stock has been counted, discrepancies noted (negatives = more stock found, positives = less stock found).
UW_STOCKTAKE_SUS	Bi-annual count of everything in the store, notes stock discrepancies. If any line in the layout group has changed due to a stocktake then every line in the layout group was checked by the stocktake.
AUTO_ADJ_NBAL_SUS	Negative adjustment. Sale of units that were not accounted for in the stock count.
CNTS_NONROUTINE_CT	Non-routine count, the stockfile looks wrong and is flagged. Binary variable, 1 = flagged as wrong, 0 = not flagged.
BR_REC_OOS_CT	Informal count of the stock. Binary variable, 1 = informal count, 0 = no informal count.
SYS_GEN_OOS_CT	The system shows there is no stock. Binary variable, 1 = system flags that there is no stock, 0 = system did not flag.

As evident from the data, this task necessitates a multivariate unsupervised learning approach, given the presence of multiple interdependent variables. Moreover, the anomalies within the data lack labels and represent instances in time where a deviation from the normal pattern occurred.

3.2 Data Cleaning

As mentioned earlier, the initial dataset comprises 6.8 million rows and 17 columns. In order to identify anomalies within the data, no cleaning was performed. Cleaning the data could potentially remove important information or anomalies (outliers) present in the dataset. Additionally, there were no missing values in the dataset, so there was no need to drop any rows or columns

3.3 Feature Engineering

In order to conduct data exploration, several variables were created, but they were only necessary for exploration purposes and were removed during preprocessing. The first variable, "STOCKED," was a binary variable indicating whether a product had been restocked or not. It was created by sorting the data by date, product, and branch number in ascending order and flagging a 1 whenever there was an increase in the "LINE_STOCK_SUS" column. A 0 was assigned if the product had not been restocked. The next variable created was "DIFF_DAYS," which represented the number of days it took for a branch to restock a product. If a product had not been restocked, a 0 was assigned. To create "DIFF_DAYS," another variable called "DAY_INT" was created to calculate the interval of stocking days from the

"STOCKED" column. However, "DAY_INT" was subsequently dropped. The use of these variables are presented in the data exploration section.

The next step in feature engineering was to identify the top seven products from each branch that would be used for the modeling section. Since there were seven product categories[2], the most important products and products present in all branches were selected. Two additional columns were created: "Frequency" and "Present in all branches." "Frequency" identified the product with the highest occurrence for each category "BUY_OFF_NAME" by analyzing the "NEW_LINE_NUM" column. "Present in all branches" indicated whether a product was present across all store branches, with a true value denoting its presence. These variables were created to identify the best product combinations for modeling and experimentation. Therefore, the combinations include the branch number and the product with the highest frequency of occurrence and presence in all branches for each category.

3.4 Data Exploration

The column "NEW_BU_NUM" denotes the branch number for each store location. In Figure 1, the proportion of instances for each row can be observed. The branches show a relatively equal distribution, with branch 1336 having the largest proportion at 20.4% of the total instances, and branch 1096 having the smallest proportion at 19.8% of the total instances. This uniform distribution suggests that each branch has a similar quantity of products available.

[2]Impulse, Deli, Fish, Meat, Poultry and Eggs, Prepared Foods, and Seasonal

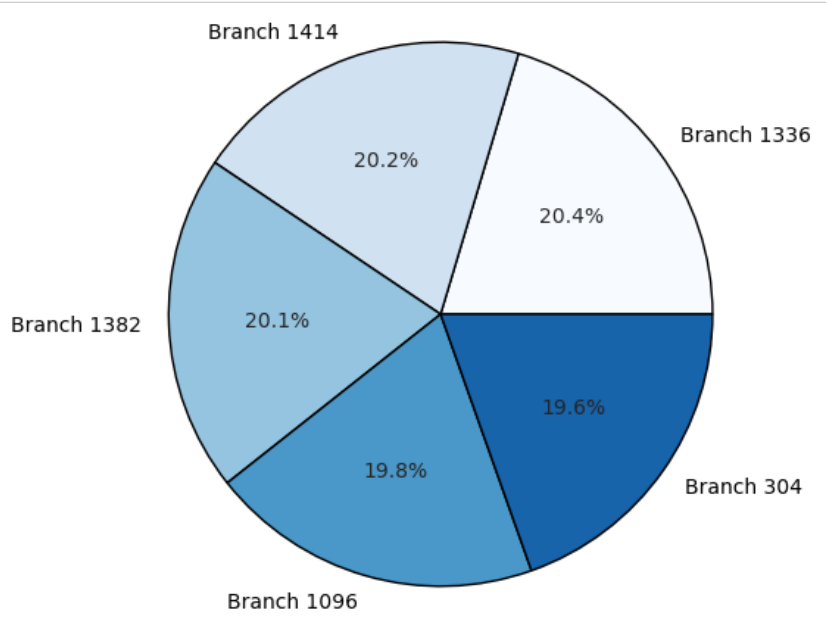

Figure 1: Proportion of Instances for Each Branch

The figure below, Figure 2, presents the number of sales for each product category across all branches. The first five graphs in Figure 2 represent the sales figures for each product category at individual branches, providing insights into category-specific performance at each location. Meanwhile, the last graph in Figure 2 showcases the top-selling product category across all branches, offering a comprehensive overview of the most successful category in terms of sales. The column "LINE_SALES_SUS" provides information about the daily sales units per product, while the column "BUY_OFF_NAME" indicates the product category. Examining the sales numbers for each category reveals a generally consistent pattern across all

branches. Notably, the "Impulse" category emerges as the front runner with the highest number of sales, indicating either a diverse range of impulse products or significant popularity among customers. In contrast, both the "Seasonal" and "Deli" categories exhibit the lowest sales figures. The limited sales in the seasonal category can be attributed to the fact that these products are exclusively available during specific times of the year.

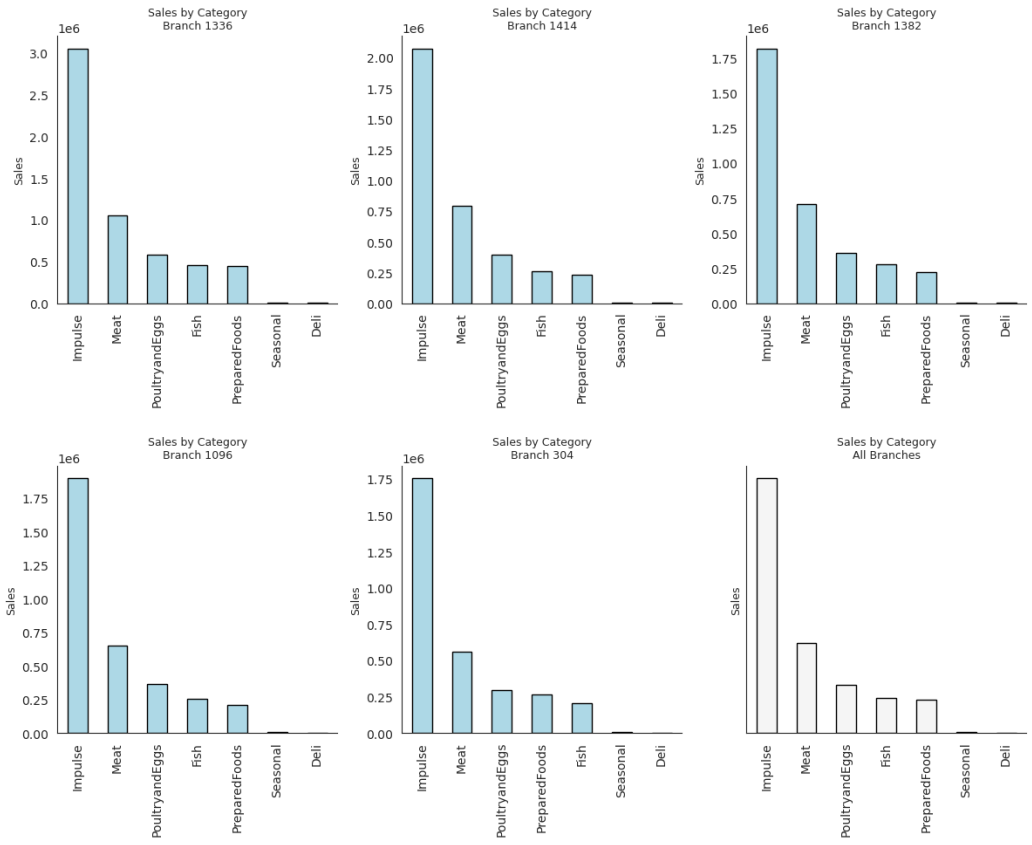

Figure 2: Sales by Product Category

Figure 3 depicts four line graphs, each illustrating the monthly sales and stock levels of four different products at branch 304. The dataset covers a specific time frame, revealing no discernible seasonality in the data. One notable positive aspect is the consistently maintained stock levels below the amount of sales, ensuring that products are adequately stocked and avoiding instances of understocking. However, the amount of stock consistently exceeds the amount of sales, indicating a possible case of overstocking. This raises the risk of increased wastage. Additionally, the figure illustrates intermittent spikes in stock at specific intervals, most notably in January, which can be attributed to the seasonality that often accompanies the commencement of a new year. However, the graph also reveals certain irregularities. For instance, Product 1706732 displays stock increases in July and October, despite exhibiting consistent sales throughout the year. This deviation suggests that the product's seasonality might not necessarily align with its demand. To clarify, consider strawberries as an example: while they are in demand all year round, their distribution peaks during the summer and early fall months. Similarly, the fluctuation in the stock of Product 1706732 may be dictated by its inherent seasonality rather than by changes in consumer demand.

Figure 3: Sales and Stock Over Time

While Figure 3 provides a snapshot of monthly sales, its broad scope limits our ability to discern potential daily anomalies. To address this, Figure 4 complements it with a detailed line graph, which chronicles daily sales for the same products in Branch 304. This granular approach offers insights into specific daily events.

For instance, product 94100 indicates negative sales in October, while product 1706732 displays an abrupt plateau in sales from November to January - both instances hint at potential anomalies. Furthermore, products 159008 and 1687392 exhibit not only consistent sales patterns, but also marked sales surges at the start of the year. These spikes could suggest the influence of seasonal promotions or increased demand of the particular product.

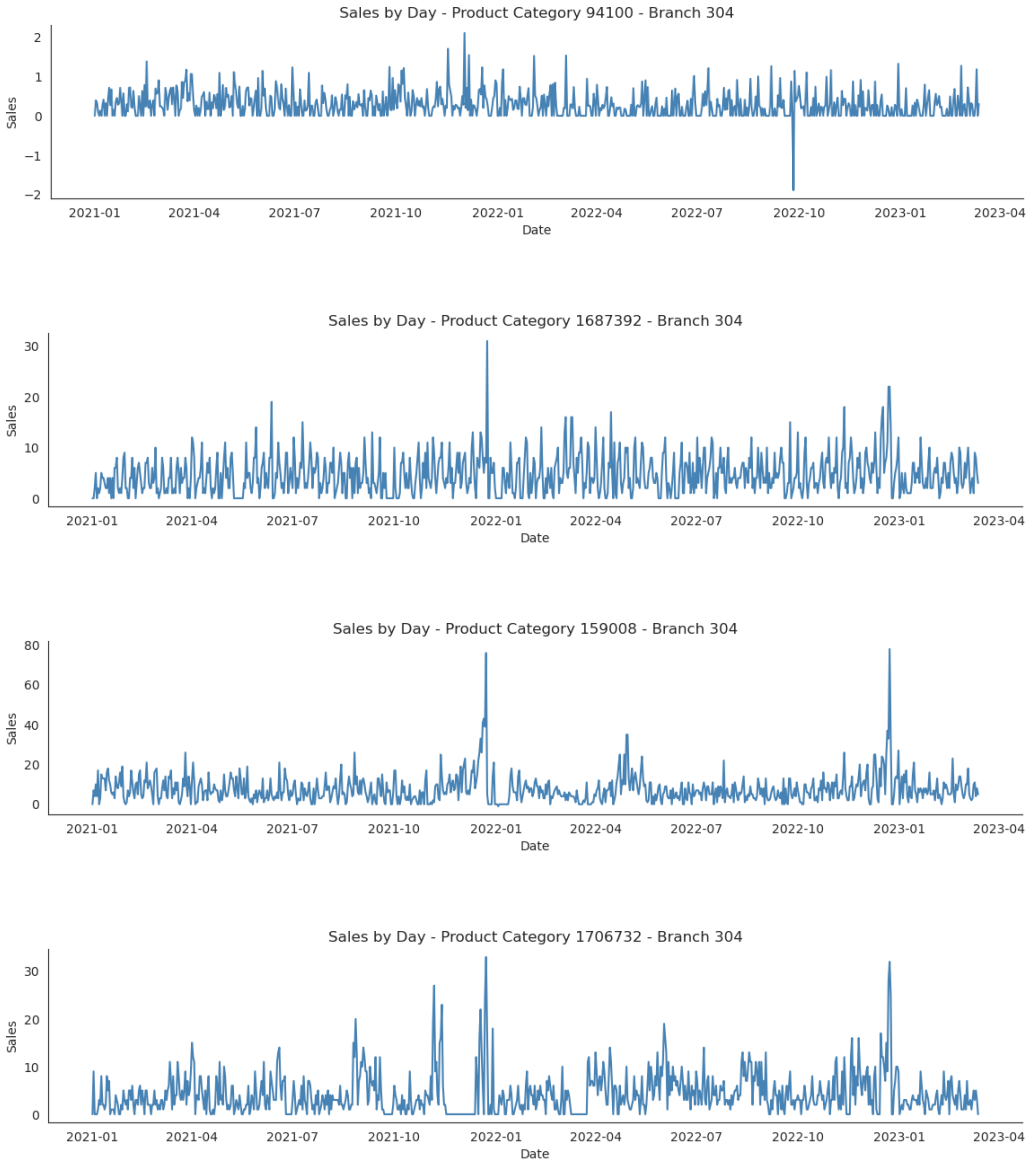

Figure 4: Sales Over Time

Figure 5 presents a histogram that outlines the distribution of sales and stock for the same quartet of products from Branch 304 previously examined. It is clear from this visual representation that neither the sales nor the stock follow a standard normal distribution for the included products. Moreover, each product exhibits a left skew, meaning that extreme values are few, thereby simplifying the process of anomaly detection.

The four plots reveals that the standard deviation of stock exceeds that of sales. This observation reinforces the stability of sales in contrast to stock, a trend initially suggested by Figure 3 and reaffirmed here in Figure 5. Such stock volatility presents a significant business issue for John Lewis. Further, for stock, the second bar reaches the highest point, while for sales, it's the first bar that predominates. This discrepancy indicates a substantial number of days where sales hover around zero, a common phenomenon attributable to the seasonality of demand.

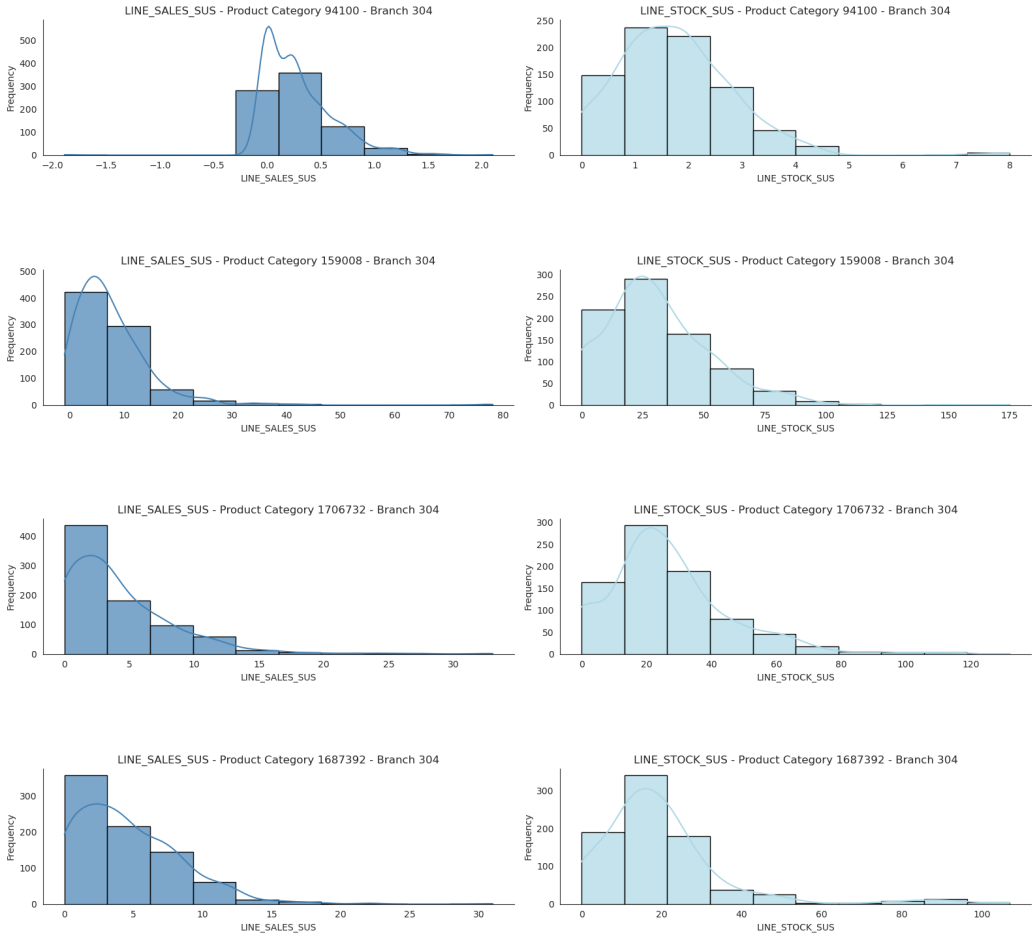

Figure 5: Distribution of Sales and Stock Over Time

The binary variables "CNTS_NONROUTINE_CT", "BR_REC_OOS_CT", and "SYS_GEN_OOS_CT" reflect scenarios where the stockfile seems incorrect, products undergo random counting, or instances where no stock is present. Each instance is marked with either a 0 or a 1, with 1 indicating a flag and 0 representing no flags for these circumstances.

The following figure, Figure 6, breaks down the proportions of 0s and 1s within each category, with gray signifying 0 and blue denoting 1. Within this chart, we see that the stockfile of products under "CNTS_NON_ROUTINE" was flagged for appearing incorrect only 0.134% of the time. Similarly, under "BR_REC_OOS_CT", products were subjected to random counting merely 3.36% of the time. In the case of "SYS_GEN_OOS_CT", there is a higher proportion of 1s, suggesting a lack of stock approximately 8.24% of the time.

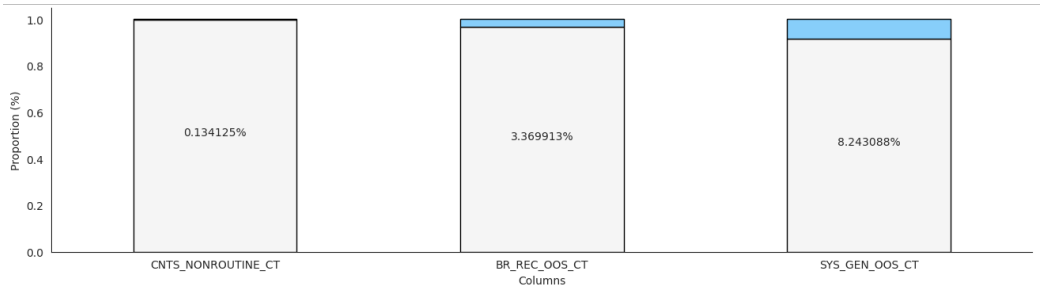

Figure 6: Proportion of True in Binary Columns

Despite these observations, these variables yield limited insights, which makes them likely candidates for exclusion during the modeling phase.

Figures 7 and 8 feature boxplots that depict the distribution of sales and stock across the various product categories in Branch 304. Figure 7 highlights the impulse category as having the most substantial sales and stock levels, marked by considerable outliers in the stock data. However, the dominance of the impulse category in Figure 7 compresses the remaining categories towards the bottom, which hinders a detailed understanding of their distributions.

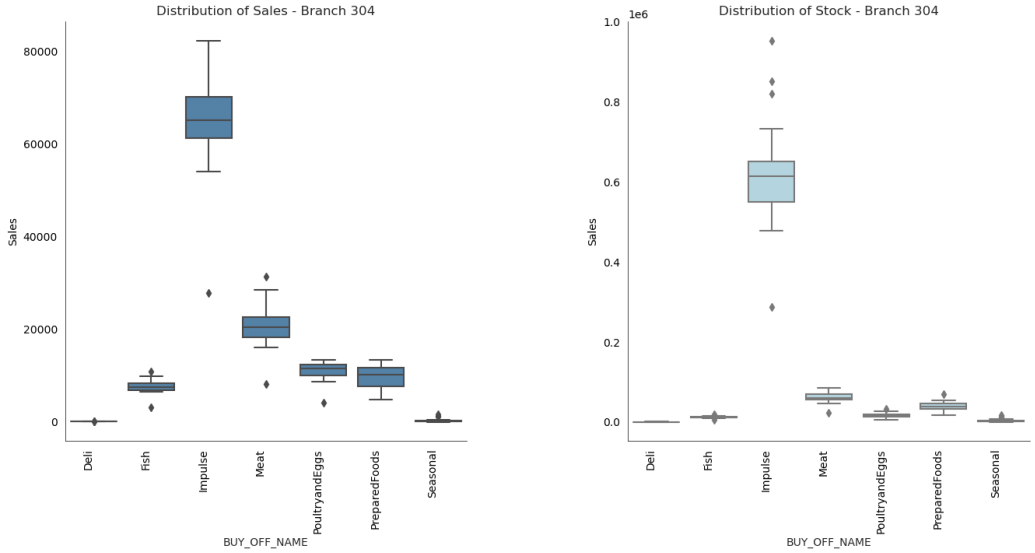

Figure 7: Box Plot - Distribution of Sales and Stock

To better illustrate the spread of these outliers, a logarithmic transformation was applied in Figure 8 to the quantity of sales and stock. This adjustment allows for a more detailed and comprehensive view of the data across all product categories. The log-transformed vertical axes in Figure 8 reveals a broader distribution for seasonal products, punctuated by a significant number of outliers. This suggests a diverse product range, encompassing both niche items and widely popular goods. Furthermore, Figure 8 reveals a trend where sales outliers predominantly occur at the lower end of the spectrum across all product categories. Conversely, stock outliers tend to reside in the upper range. This discrepancy reveals a compelling dynamic between sales and stock trends within the various product categories.

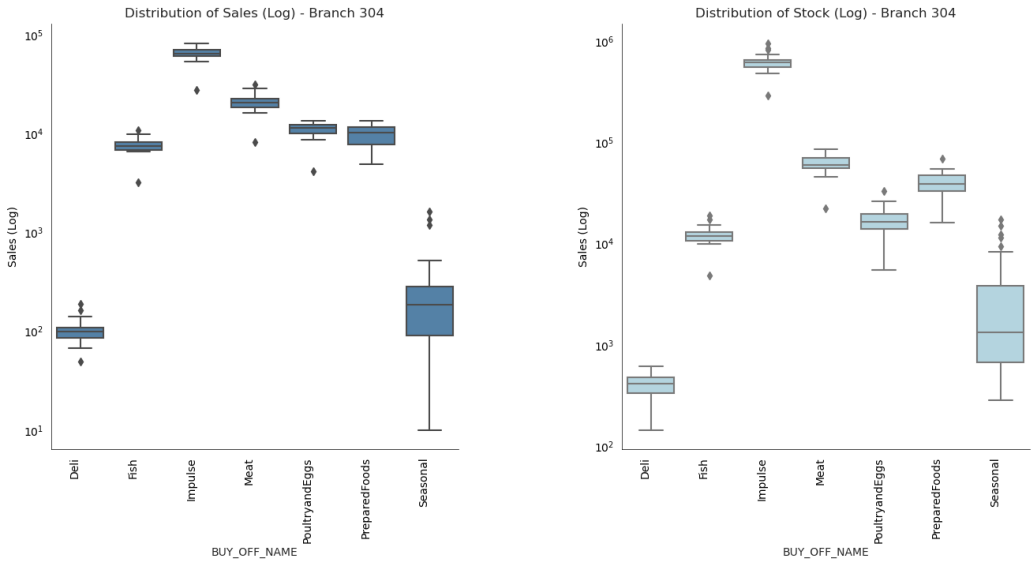

Figure 8: Box Plot - Log-Transformed Distribution of Sales and Stock

Figure 9 depicts histogram plots of the average lead time (in days) for product restocking across different branches. Branch 1336 shows the most efficient restocking procedure with the lowest average lead time of 4.02 days, while Branch 304 experiences the highest average lead time, taking 5.37 days. The mode lead time - the most frequently occurring - is 5.17 days, a figure shared by both Branch 1096 and Branch 1414.

The histogram reveals a left skew for all branches, demonstrating the rarity of products being restocked after approximately $e^4 \approx 55$ days. This pattern suggests a correlation between the speed of restocking and the nature of the products; items that are restocked quickly are likely to be perishable, whereas those replenished after a more extended period are probably non-perishable.

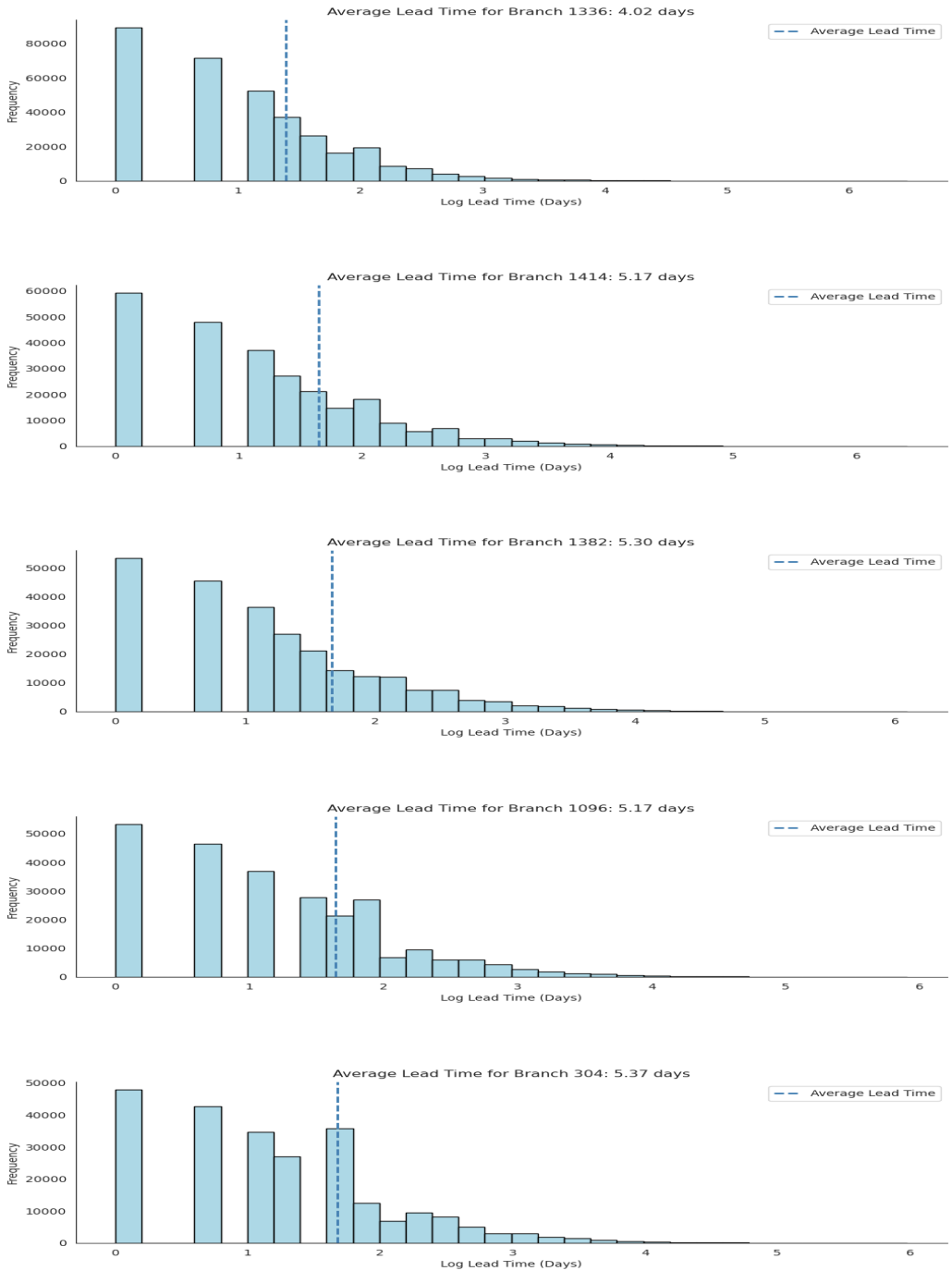

Figure 9: Distribution of Sales and Stock Over Time

4 Methodology

Figure 10 shows a visual representation of the methodology.

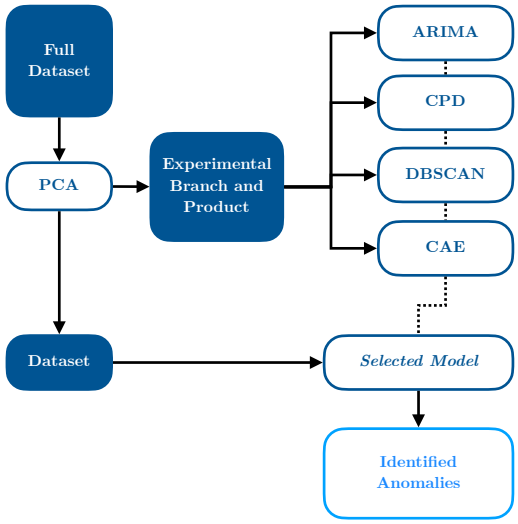

Figure 10: Methodology

The initial phase involves subjecting the numerical variables from the dataset to a PCA (Principal Component Analysis) in order to identify the most significant features. Moving forward, the process of selecting the experimental branch and product is elaborated upon in the section discussed earlier. Following this, the experimental branch and product are subjected to four distinct models: Autoregressive Integrated Moving Average (ARIMA), Change Point Detection (CPD), Density-Based Spatial Clustering of Applications with Noise (DBSCAN), and Convolutional Autoencoder (CAE). By comparing the outcomes of these models, one model is selected. Subsequently, this chosen model is applied to the remaining dataset. Through this application, anomalies within the dataset are identified.

4.1 Data Pre-processing

4.1.1 Principal Component Analysis (PCA)

As manual feature selection cannot be conducted for an unsupervised learning task (there are no predefined target labels) a dimensionality reduction and feature importance method was selected instead. A principal component analysis was run on all of the numerical features in order find the most important features by identifying the explained variance ratio. Principal component analysis is a dimensionality reduction method that is utilized to reduce the size of large datasets by minimizing the number of variables to a smaller number of variables that contain the most important information (Hotelling, 1933). Simply put, the PCA reduces the number of variables within a dataset, whilst retaining as much important information as possible (ibid). *The PCA model conducts dimensionality reduction by taking the following steps:*

Step 1: Standardization

Before performing PCA, standardization is conducted to ensure that the initial variables have comparable scales (Smith, 2002). This is crucial as PCA is sensitive to variable variances, and if there are substantial differences in ranges, it may lead to biased results (ibid). In this study, binary variables[3] were excluded due to the significant variance between 0 and 1, which could introduce bias. Only the continuous variables[4] were included after being

[3]CNTS_NONROUTINE_CT, BR_REC_OOS_CT, and SYS_GEN_OOS_CT.

[4]LINE_SALES_SUS, LINE_STOCK_SUS, LINE_WSTG_SUS, UW_OP_COUNT_SUS, UW_STOCKTAKE_SUS, and AUTO_ADJ_NBAL_SUS.

standardized[5], allowing them to have equal importance and the same scale. The mathematical representation is as such:

$$z = \frac{X - \mu}{\sigma}$$

where X is the value, μ is the mean and σ is the standard deviation.

Step 2: Covariance matrix computation

The goal of this step is to examine the relationships between variables in the input dataset by calculating the covariance matrix. The covariance matrix is a symmetric matrix that shows the covariances between all possible pairs of variables. This matrix helps identify any correlations or redundancies among the variables (Smith, 2002). In this case, 6 features are inputted into the PCA, making it a 6 dimensional array. The covariance matrix is a 6x6 data matrix shown in Appendix 10.2[6]. The covariance matrix contains variances on its diagonal, symmetrically representing correlations between variables as positive or negative values (ibid).

Step 3: Computing the eigenvectors and eigenvalues of the covariance matrix to identify the principal components

In this case, with 6 dimensions, there are 6 eigenvector/eigenvalue pairs associated with the covariance matrix. Eigenvectors represent the directions of the axes holding the most variance (principal components), while eigenvalues indicate the variance within each principal component (Holland, 2019). By ranking the eigenvectors based on eigenvalues (highest to lowest),

[5]StandardScaler is employed.

[6]Adapted from Smith (2002). Variables noted as a, b, c, d, e and f.

the order of significance for the principal components is determined, allowing dimensionality reduction while preserving important information (ibid).

Step 4: Feature vector

In this step, the results are compared and it is decided which features are chosen to be kept based on importance (Smith, 2002). For this, a 95% variance threshold was set, meaning that the features which hold 95% importance will be kept and the others will be discarded. The results were the following:

1. Explained Variance Ratio for LINE_SALES_SUS: 0.2252
2. Explained Variance Ratio for LINE_STOCK_SUS: 0.1823
3. Explained Variance Ratio for LINE_WSTG_SUS: 0.1668
4. Explained Variance Ratio for UW_OP_COUNT_SUS: 0.1665
5. Explained Variance Ratio for UW_STOCKTAKE_SUS: 0.1552
6. Explained Variance Ratio for AUTO_ADJ_NBAL_SUS: 0.1042

Thus, the most important features are the first five, and the last can be discarded as it is the least important.

Step 5: Recasting the data along the principal component axes

In the final step of PCA, the eigenvectors of the covariance matrix form a feature vector. This vector is used to transform the data from the original axes to the principal components (Hotelling, 1933). This transformation is achieved by multiplying the transpose of the original data set by the transpose of the feature vector (ibid).

Figure 11 shows the cumulative variance ratio for the 6 input features in the PCA algorithm. The horizontal line represents the 85% threshold of cumulative variance.

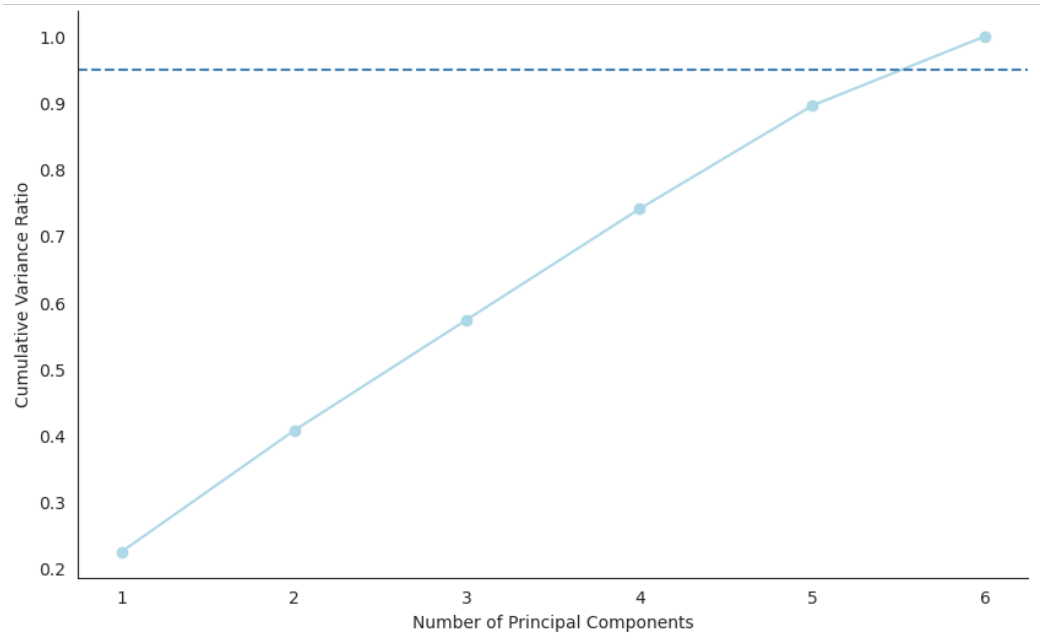

Figure 11: Principal Components Cumulative Variance

4.2 Models

4.2.1 Autoregressive Integrated Moving Average (ARIMA)

The ARIMA (AutoRegressive Integrated Moving Average) model creates its forecasts by computing a simple weighted sum of previous values, which are then rectified by incorporating a moving average. A key assumption of the ARIMA model is that the time series data is stationary (Géron, 2019). The ARIMA model combines the concepts of autoregression (AR),

differencing (I for Integrated), and moving average (MA) to effectively capture the patterns in univariate time series data (Box et al., 2008). Given a variable of interest y and p past periods, the ARIMA model can be described mathematically[7] as follows:

$$y_t^{(d)} = c + \varepsilon_t + \phi_1 y_{t-1}^{(d)} + \phi_1 y_{t-2}^{(d)} + \ldots + \phi_p y_{t-p}^{(d)} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \ldots + \theta_p \varepsilon_{t-q} \quad (1)$$

Where c and ε are constant and error terms respectively. The autoregressive part (AR) uses coefficients $\phi_1, \phi_2, \ldots, \phi_p$ to indicate the strength and direction of the relationship between present and past values of the variable y. If the coefficient is positive, an increase in the variable p periods ago leads to an increase in the variable in the present. On the other hand, the moving average part (MA) ensures the model captures q previous residual errors from an application of a moving average. Once again, the coefficients indicate the strength and direction of this relationship. The integrated part ensures that the model eliminates polynomial trends up to degree d, to achieve stationarity, one of the assumptions of the model (Andres, 2022).

The ARIMA model is used for forecasting but can be used for anomaly detection as it identifies deviations from forecasts or residuals (Moschini et al., 2021). In this study, ARIMA is utilized to see if it can forecast the sales of the experimental product at the experimental branch, the forecast is used to identify the points in time where there are extreme deviations. Firstly, we test for stationarity using an ADFuller Test. The Augmented Dickey-

[7] Andres (2022)

Fuller (ADF) test is a statistical test used to determine whether a time series dataset has a unit root, which indicates if the series has a stochastic trend (Verma, 2021). Table 4 in Appendix 10.3 shows the results of this test.

Moreover, to comply with the key assumption of the ARIMA model of data stationarity, the first and second order differencing of sales is conducted in order to achieve stationarity. The model is then trained on a portion of the historical data and tested on the remaining data. By doing so, areas where the model's results align or misalign with the historical data can be identified, with misalignment indicating potential anomalies.

4.2.2 Change Point Detection (CPD)

Change Point Detection (CPD) is aimed at identifying abrupt changes in time series where the data shows a significant shift in behavior or distribution (Truong et al., 2020). It works by finding the optimal number of change points $C*$ that divides the data into k segments where the dissimilarity -the cost function- between segments is minimized (ibid). There are two main approaches to change point detection: online and offline. Online detection aims to identify changes as soon as they occur, while offline detection involves retrospectively detecting changes once all samples are received (Downey, 2008). Figure 12 shows the algorithms detecting two change points, with a sudden change in the value of the variable of interest.

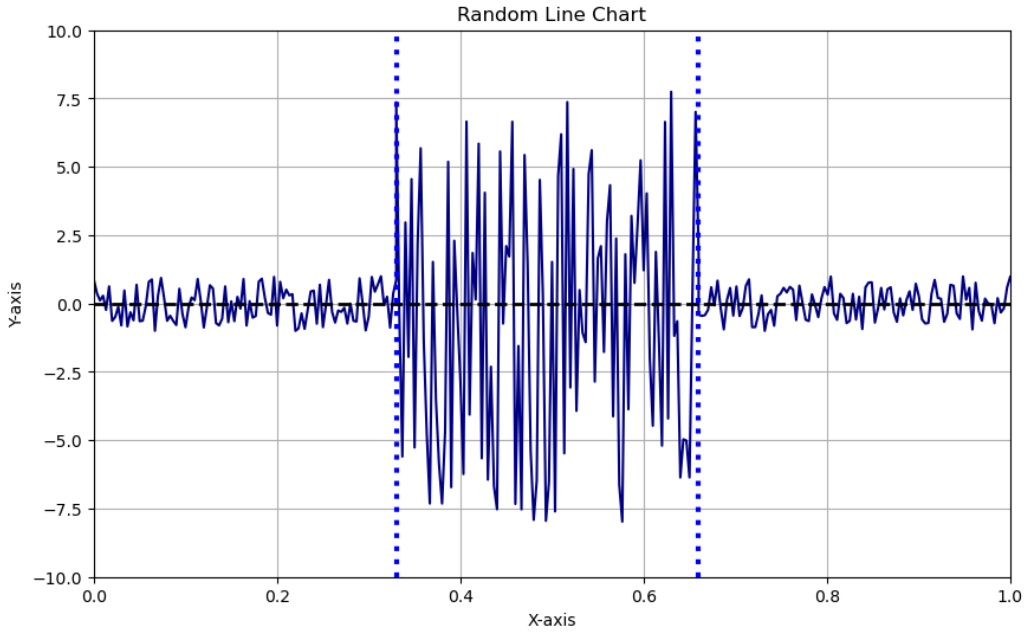

Figure 12: Change Point Detection[8]

In this study, CPD on the 5 features selected by the PCA is carried out, by using a subset of Dynamic Programming known as Pruned Dynamic Programming. Dynamic Programming (DP), is an algorithmic technique that optimally solves problems by breaking them down into overlapping subproblems and reusing solutions (Truong et al., 2020). Whereas, Pruned Dynamic Programming reduces computational complexity by strategically eliminating certain subproblems that are less likely to lead to optimal solutions, significantly reducing unnecessary calculations (ibid).

CPD is utilized to identify anomalies in the sales of the experimental product at the experimental branch. This process involves using the L1 norm

[8]Adapted from Truong et al. (2020).

as the cost function, which helps in detecting abrupt changes in the data. This cost function measures the dissimilarity between segments x_i and x_j is calculated as follows[9]:

$$C(i, j) = |x_t - \mu_{i,j}| \tag{2}$$

Where x_t is a data point at time t within segments i, j and $\mu_{i,j}$ denotes the mean value of the variable of interest in each segment. To ensure that anomalies can be detected throughout the entire dataset, the maximum number of data points allowed between two change points (*jump*) is set to 1. Moreover, to ensure that meaningful and significant segments are identified as change points, a minimum segment size (*min_size*) of 2 data points is specified. Additionally, an allowance of 1.5% of the data to be classified as change points has been set. This ensures that the detection process is sensitive enough to capture potential anomalies. As mentioned previously, to optimize the solution and improve efficiency, Pruned Dynamic Programming is employed, thus helping to converge to an optimal result faster.

4.2.3 Density Based Spatial Clustering with Noise (DBSCAN)

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is an algorithm that identifies clusters in a dataset based on the density of instances (Ester et al., 1996). It operates using two key parameters: epsilon (ε) and a minimum number of samples (*min_samples*). ε defines the maximum distance between instances for them to be considered part of the same

[9]Adapted from Banoula (2023).

neighborhood. Instances within ε distance of each other are considered neighbors (Géron, 2019). The *min_samples* parameter sets the minimum number of instances required in a neighborhood for an instance to be considered a core instance (ibid) (shown in Step 2 in Figure 13). DBSCAN forms clusters by identifying sets of instances that are density-connected to each other through a series of core instances (which represent dense regions in the data) (Ester et al., 1996). An instance p_i and another instance p_j are considered density-connected if there exists a core instance p_k such that both p_i and p_j are in each other's neighborhood (ibid) (shown in Step 3 in Figure 13). Any instance that is not a core instance and does not have a neighborhood is considered an anomaly or noise (shown in Step 4 in Figure 13)(Géron, 2019). The final result is a set of clusters, each containing a group of instances that are density-connected, and any outer instances are considered anomalies or noise.

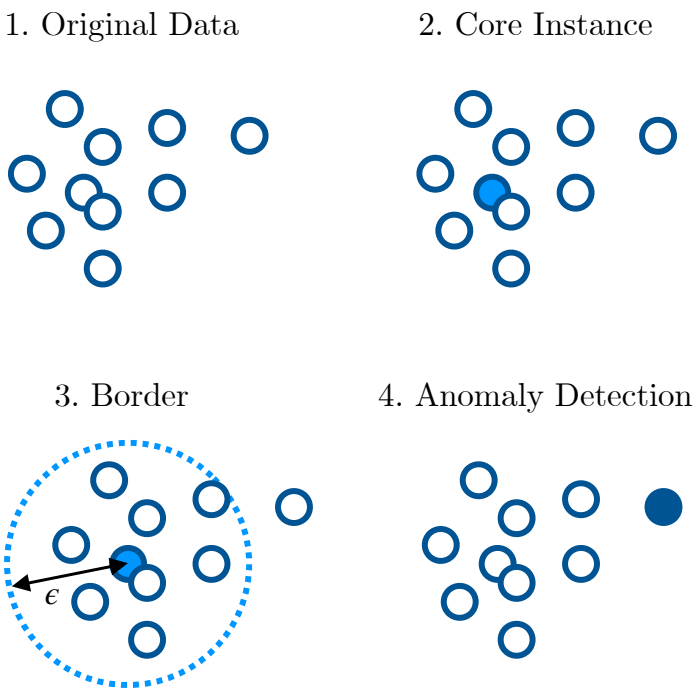

Figure 13: DBSCAN[10]

In this study, DBSCAN is employed to identify anomalies in the data using the same selected features for both the experimental branch and the experimental product. The parameters epsilon and *min_samples* are deliberately set to 1 and 3, respectively. ε is chosen to create dense clusters, which is suitable for handling the very large dataset with dense regions, where anomalies are expected to stand out distinctly. Meanwhile, setting *min_samples* to 3 filters out isolated data points as anomalies and also considers the presence of noise or small clusters, achieving a balanced approach to anomaly detection.

[10]Adapted from Ester et al. (1996).

4.2.4 Convolutional Autoencoder (CAE)

Autoencoders are unsupervised neural networks that learn compact representations of input data called "latent representations" without any explicit labels (Hinton and Salakhutdinov, 2006). They are designed to encode input data into a compressed form (encoding) and then decode it back to closely resemble the original input (decoding) (ibid). Convolutional Autoencoders (CAE) are a specific type of autoencoder that uses convolutional layers, making them particularly effective for image data (Géron, 2019). In this case, these models can be employed for anomaly detection. Using a CAE for anomaly detection allows the model to learn the underlying patterns in the normal data, making it capable of detecting deviations and unusual instances, even if they were not present during the training phase (Masci et al., 2011).

During training, the CAE learns from a dataset containing only normal data samples. It aims to minimize the reconstruction error, which measures how well the autoencoder can recreate the input data from its compressed form. By doing so, it learns to capture essential features present in the normal data (Géron, 2019).

In the anomaly detection phase, the autoencoder is presented with both normal and potentially anomalous data (Hinton and Salakhutdinov, 2006). The input data is encoded into a lower-dimensional representation and then reconstructed using learned filters and activation maps (Masci et al., 2011). The key to anomaly detection lies in analyzing the reconstruction error, the difference between the input data and the reconstructed

output. Normal data points typically have a small reconstruction error since the autoencoder learned to accurately encode and decode them. However, anomalous data points, with unusual patterns or deviations, result in higher reconstruction errors. By setting an anomaly score threshold, data points with high reconstruction errors are flagged as anomalies (ibid).

In this study, the CAE is applied to identify anomalies in the sales of the experimental product at the experimental branch. The model is first trained over 100 epochs using a batch size of 512 instances, and early stopping is implemented to ensure the model stops when the loss in the validation set does not improve in 5 epochs. Then, the reconstruction error threshold is obtained by taking the maximum mean absolute error (MAE) from the training process (Hinton and Salakhutdinov, 2006). The MAE is given by: $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$, where n is the number of observations, y is the real value and $\hat{y}$ is the predicted value (Géron, 2019). Anomalies in the test set are identified by finding the points in time where the test set mean absolute error exceeds the threshold (ibid). Figure 14 shows the architecture of a CAE.

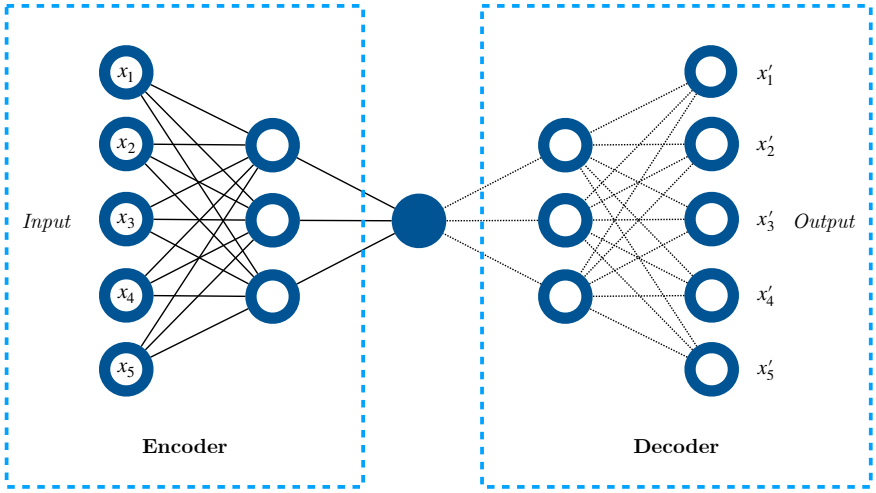

Figure 14: Convolutional Autoencoder[11]

The architecture of the Convolutional Autoencoder (CAE), as illustrated in the figure 14, is structured with an initial input layer. This input layer is followed by a sequence of convolutional hidden layers that function as the encoder. At the center of the architecture is the latent space, which receives and conveys the latent representations to the subsequent decoder stage. Comprising the decoder are additional hidden convolutional layers which generate the output layer.

[11] Adapted from Masci et al. (2011).

5 Evaluation

Due to the unsupervised nature of the models employed and the data, no definitive ground truths are generated. Consequently, the performance evaluation of each model hinges on visual representations of their respective outputs and the model's ability to identify anomalies. The outputs are examined in this section.

5.1 ARIMA Results

In evaluating the results of the applied ARIMA model on the dataset, the model's efficacy was substantially limited due to the characteristics of the data and the inherent constraints of the chosen approach. The primary goal was to detect anomalies in the data by contrasting forecasted sales with the historical sales. However, this endeavor encountered substantial obstacles given that the data is unlabeled, multivariate and unseasonal, which poses difficulties for this traditional supervised learning technique.

This model requires both a target variable and predictive features to produce a reliable output (Pozzolo, 2014). In this scenario, sales was used as the target variable, but the absence of appropriate features in the data meant that the model was essentially attempting to draw insights from a single variable. This is akin to trying to predict the outcome of a multifaceted event based on a single, isolated factor. Consequently, the ARIMA model, without additional predictive features to anchor its forecasts, was incapable of successfully forecasting the sales variable.

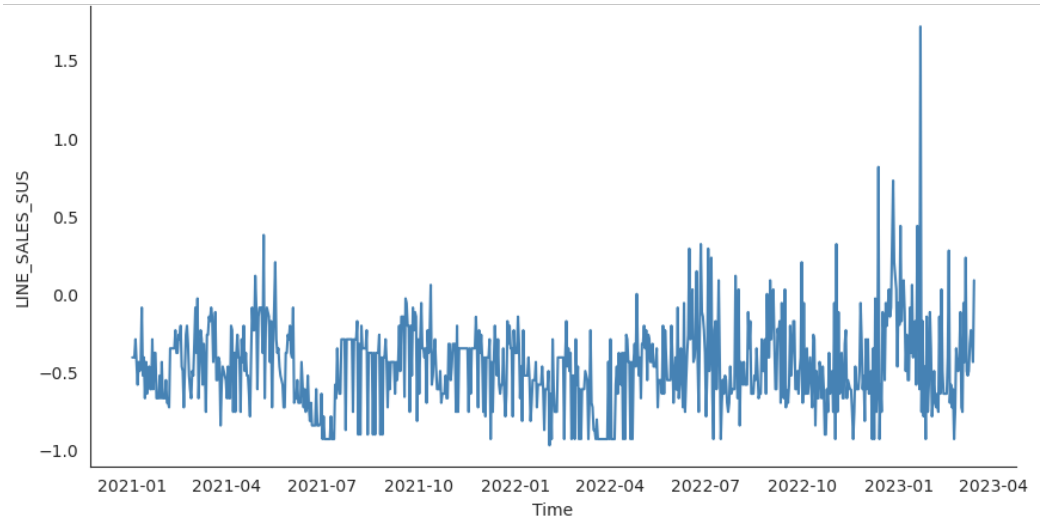

Figure 15: Sales Over Time

The data's lack of seasonality, as evidenced in Figure 15, presents an additional challenge for the ARIMA model. The ARIMA model depends heavily on trend composition and seasonality to generate accurate and meaningful forecasts (Géron, 2019). When these elements are missing, as is the case with this data, the model's ability to leverage past values and errors for predicting future data is greatly impeded (ibid).

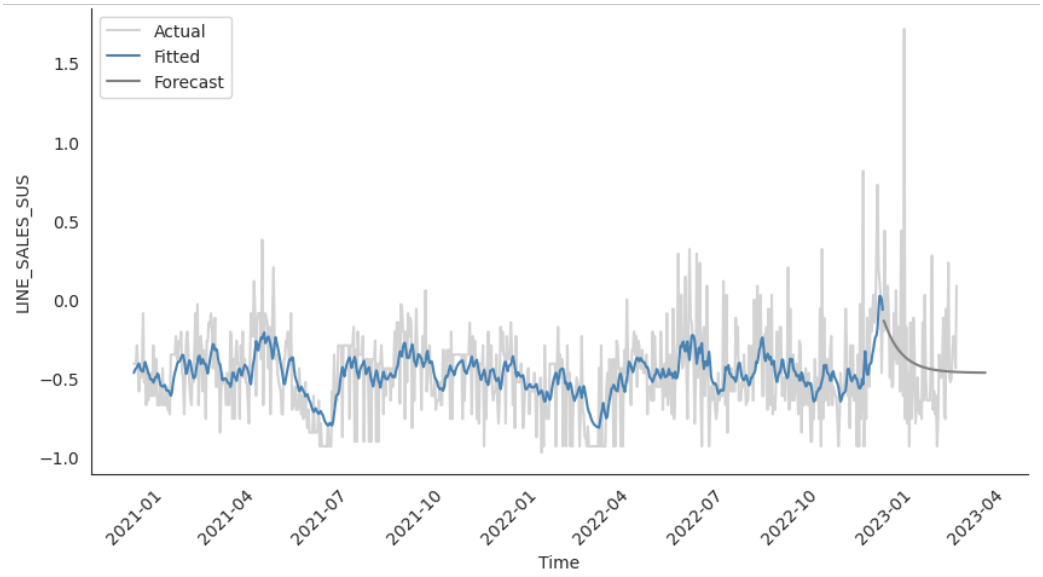

Figure 16: Actual vs Fitted Sales

Lack of seasonality not only reduces the model's predictive power but also increases the risk of overfitting (Adhikari and Agrawal, 2013). In this context, the model concentrates excessively on random fluctuations and noise within the data, resulting in poor generalization to the unseen data (ibid). The consequence of this overfitting risk is demonstrated in Figure 16, where the model fails to accurately represent the variability of the data. Furthermore, the ARIMA model, engineered to capture periodic patterns, struggles when faced with unseasonal data punctuated by anomalies (Box et al., 2008). Random fluctuations in such data hinder the model's pattern detection abilities(ibid). This problem is underscored by the notable deviation from the true data seen in Figure 16 in which the model was unable to mirror the historical data due to the absence of a discernable periodic pattern.

Given the aforementioned challenges, the ARIMA model falls short in accurately forecasting the historical data. The shortcomings are rooted in the data's failure to meet the fundamental prerequisites of the ARIMA model. Applying this model to other variables or expanding its scope to encompass the entirety of the data would likely result in considerable disparities between forecasted and historical values. Since, the identification of anomalies is reliant on discrepancies between historical data and forecasts (Moschini et al., 2021), the model would suggest the presence of anomalies throughout the entire dataset. However, this indication is misleading and reflects the model's inadequacy. As a result of these limitations, the ARIMA model was not selected for further analysis. Instead, the change point detection method was employed as an alternative approach, allowing for a more granular evaluation of the data.

5.2 CPD Results

Next, a Change Point Detection (CPD) method was applied to the dataset. The primary objective of employing this approach was to pinpoint significant deviations in the data across multiple variables, the five most important variables as identified by the PCA[12]. The CPD method yielded promising results, successfully identifying considerable changes in the data patterns.

However, while the CPD approach demonstrated proficiency in highlighting broad deviations, it exhibited limitations in providing a nuanced

[12]LINE_SALES_SUS, LINE_STOCK_SUS , LINE_WSTG_SUS, UW_OP_COUNT_SUS, and UW_STOCKTAKE_SUS.

understanding of anomalies specific to each feature. The primary focus of the CPD model centers on detecting significant deviations based on the importance of each variable (Kawahara and Sugiyama, 2009). Consequently, it may overlook the subtleties and nuances of anomalies occurring within individual features (ibid).

Moreover, the CPD model's inherent subjectivity could potentially influence its output, due to the change point percentage that is arbitrarily set based on user preference. This change point percentage is essentially determined by the number of change points the user intends to identify (Aminikhanghahi and Cook, 2016). However, given the lack of standardized guidelines for setting this change point percentage, the model's results could be significantly impacted. If the percentage is set too low, there is a risk of missing out on critical deviations (Truong et al., 2020). Conversely, setting the percentage too high could lead to the identification of insignificant deviations that may not hold much analytical value (ibid). This challenge underscores the delicate balancing act required to ensure its optimal performance.

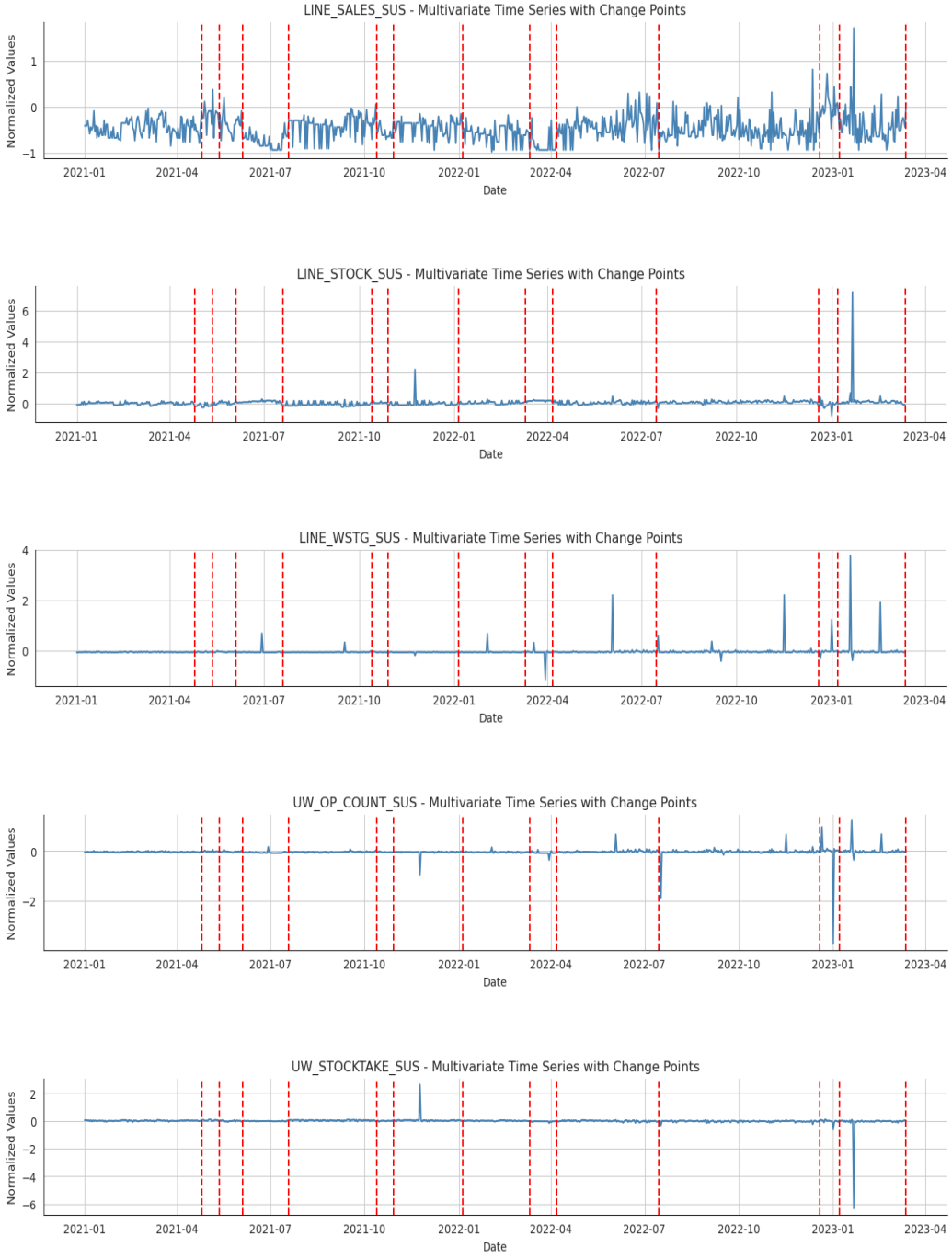

Figure 17: CPD Results

Figure 17 visually illustrates that the identified change points are consistent across each variable. This consistency suggests that the most significant anomalies transpire within a particular time interval. However, it is important to note that not all anomalies are flagged due to the model's prioritization of importance. In the case of the variable "LINE_WSTG_SUS", a notable deviation in June 2022 goes unidentified. Despite its conspicuous nature, this deviation is not deemed significant in comparison to the anomalies observed in the more critical features. Furthermore, the variable "LINE_SALES_SUS" emerges as the one with the most overall importance. This significance is corroborated by the results from the Principal Component Analysis.

In addition to the challenges of subjectivity within the model, such high-level analysis can offer valuable insights into large-scale changes in the data, but it may not be sufficiently detailed for specific anomaly detection tasks where discerning irregularities at the feature level is paramount. Thus, a Density Based Spatial Clustering with Noise (DBSCAN) model was run next to give an even more granular evaluation of the data.

5.3 DBSCAN Results

The DBSCAN algorithm was applied to the dataset and proved to be the most successful method. The primary aim was to identify anomalies in the data, an objective in which the DBSCAN method performed well. It effectively identified several anomalies and pinpointed the exact location of each anomaly in the time series.

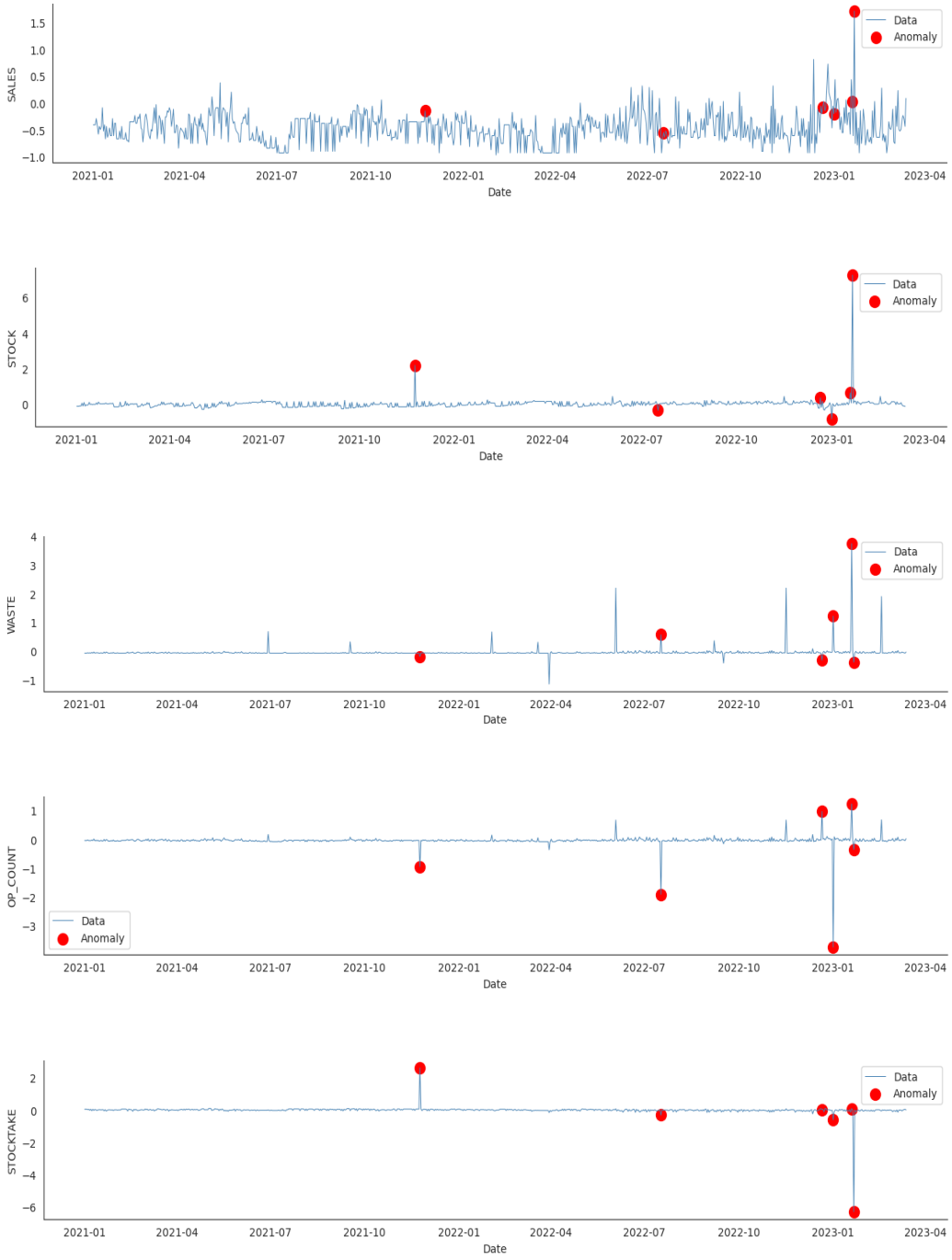

Figure 18: DBSCAN results

As can be seen in Figure 18, 6 anomalies were successfully identified with each variable as a feature, as well as the location of each anomaly in the time series. Upon comparison with the other models —ARIMA, CPD, and CAE—DBSCAN emerged as the superior choice. The principal reasons for its preeminence were threefold. First, DBSCAN's unique capability to distinguish between noise (or outliers) and non-noise data points allowed the model to offer the most detailed evaluation, providing granular insights into each feature's anomalies (Ester et al., 1996). It transcended the broad-brush analysis of the Change Point Detection model and captured the subtleties and nuances of the anomaly patterns at the individual feature level. The DBSCAN also eliminates the subjectivity of the change point detection model by including specified parameters within the algorithm (ibid). Moreover the DBSCAN method supersedes, the ARIMA model, which struggled in this case, as it is more suited to forecasting based on historical trends and seasonality (Géron, 2019).

Second, DBSCAN can identify clusters of arbitrary shapes in the data, making it a more versatile tool for anomaly detection (Celik and Dokuz, 2011). Unlike the CAE, which requires a large amount of data to learn complex data distributions (Hinton and Salakhutdinov, 2006) , DBSCAN can effectively manage with smaller datasets or samples of the dataset. Thirdly, DBSCAN's relative simplicity and lower computational requirements made it easily interpretable even for non-technical stakeholders, while also reducing computational costs. In comparison, the CAE has higher computational requirements and produces results that are more challenging to interpret, particularly for non-experts.

5.4 CAE Results

The last model applied was a Convolutional Autoencoder (CAE), implemented as a method of detecting anomalies in the dataset. The model demonstrated great success in identifying anomalies, solidifying its efficacy as an approach.

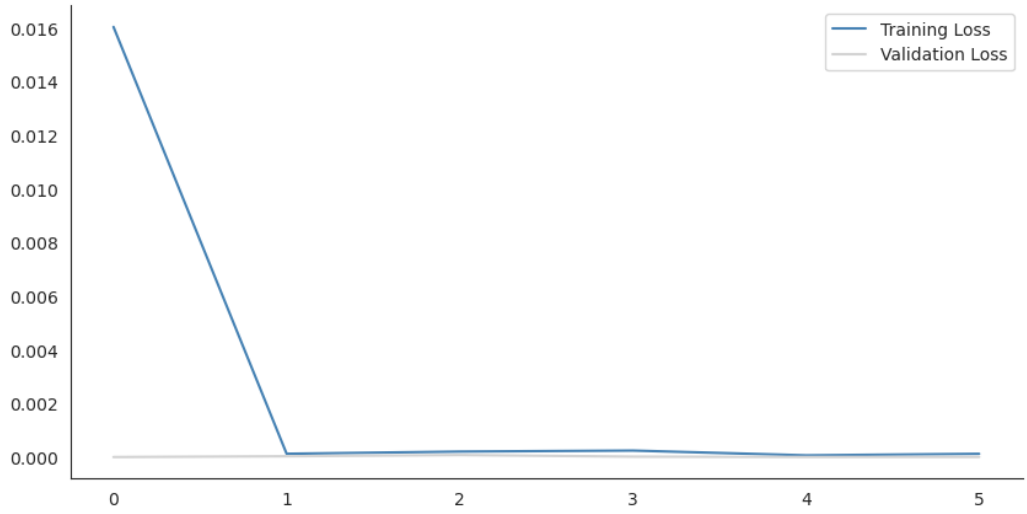

Figure 19: Training and Validation Loss Curves

Figure 19 presents the training and validation loss curves for the model, illustrating an efficient learning process. As depicted, the model did not exhibit any overfitting or underfitting, thereby validating the model's robust training and its ability to generalize well to unseen data. In addition, as highlighted earlier, an early stopping mechanism was integrated into the training process to halt the model training when there's no improvement in the validation loss for five consecutive epochs. This technique is evidenced in Figure 19, demonstrates the effective implementation of early stopping.

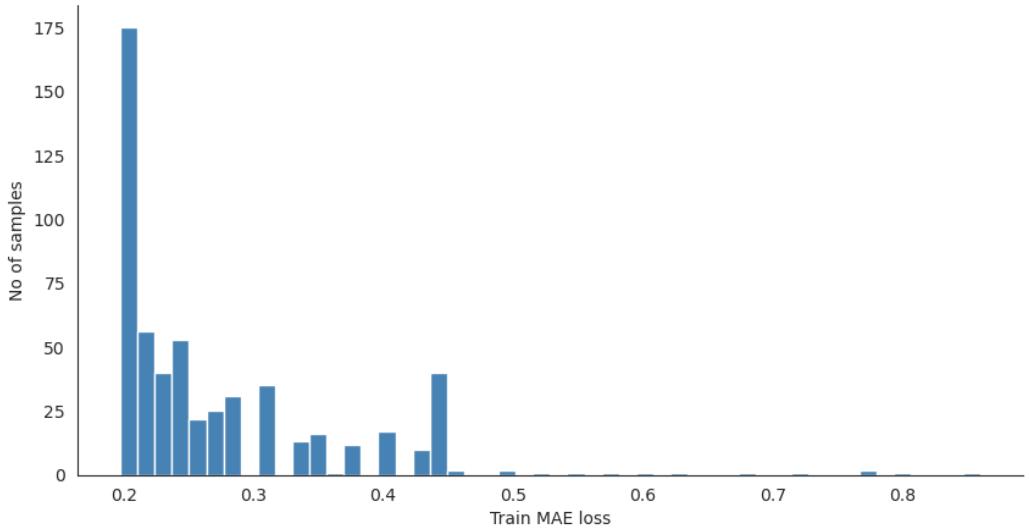

Figure 20: Reconstruction Error Threshold

As illustrated in Figure 20, the CAE demonstrates a reconstruction error threshold of approximately 0.859200. This value signifies the upper limit of tolerable deviation between the original input data and the reconstructed output from the Autoencoder (Chen and Lee, 2018). This particular threshold reflects a reasonable degree of tolerance for discrepancies between the original and reconstructed data. As outlined earlier in the methodology section, the threshold is determined by the Mean Absolute Error (MAE) derived from the model's training process. Any instances in the test set where the MAE surpasses this threshold are deemed to be anomalies (Masci et al., 2011). The reconstruction error threshold hence serves as a critical benchmark for anomaly detection, spotlighting instances in the data that the model struggles to accurately reconstruct (ibid).

Despite its successful application, the CAE was not considered the most optimal approach due to several key limitations. The first drawback of the CAE is its computational complexity. The model demands significant computational resources for training, which could be limiting in contexts where computational efficiency is a priority. Given the nature of the dataset in this study, it consists of numerous branches each carrying a diverse range of products, amounting to over 12,000 unique product/branch combinations. This implies that for comprehensive analysis, the CAE would necessitate training more than 12,000 times. Such a process would significantly contribute to computational costs. Additionally, the model's outputs can be challenging to interpret, particularly for non-technical stakeholders, thus introducing difficulties in its practical application when interpretability is crucial.

An additional consideration is the model's sensitivity to the number of anomalies it can accurately detect. While the CAE was able to identify five anomalies, it fell short of the DBSCAN's performance, which successfully pinpointed an extra anomaly, thereby extracting more meaningful insights from the data. Figure 21 below illustrates the results of the model, highlighting the five anomalies identified.

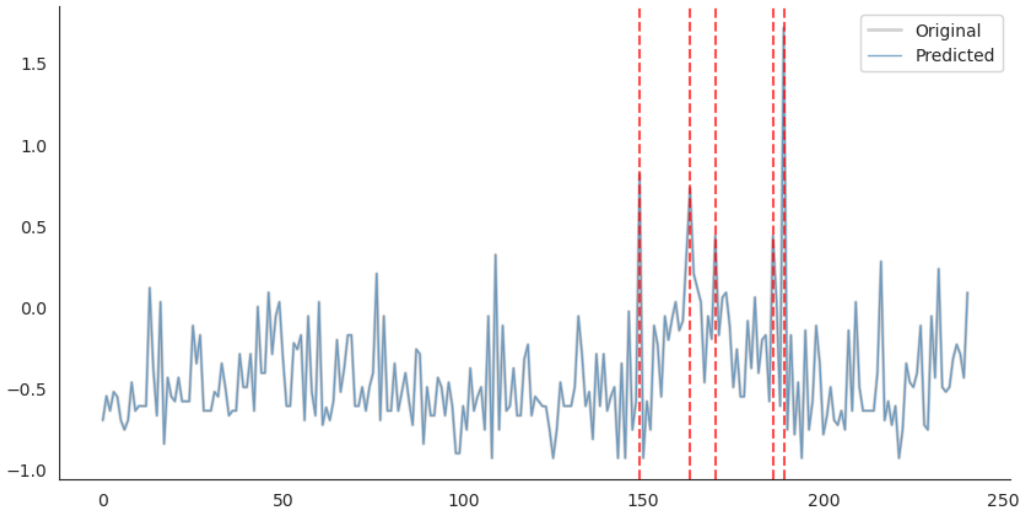

Figure 21: CAE Results

Another crucial aspect to consider is the application of the CAE as a univariate model in this instance, despite its typical use as a multivariate model. The model was unable to perform optimally when applied to multiple variables; as such, it was tested with only one variable—sales (LINE_SALES_SUS)—for trial purposes. This univariate application proved successful; however, it raises an important caveat when comparing the results to DBSCAN.

The DBSCAN algorithm identifies anomalies based on the collective interaction of all variables within the multivariate model, providing a more comprehensive view of the data. In contrast, the CAE, in this context, was limited to identifying anomalies based on a single variable. Consequently, the scope of anomalies detected by the CAE is narrower compared to DBSCAN, thus limiting the comparative analysis between the two models.

A final aspect to consider is the algorithmic design of the Convolutional Autoencoder, which poses a unique challenge in the anomaly detection task. Given that the reconstruction error threshold is determined by the maximum Mean Absolute Error (MAE) in the training set, the CAE will fail to recognize any anomalies within the training data. This creates a trade-off between the ability to identify anomalies across the entire dataset and the accuracy of the anomalies detected. Such a limitation is absent in the DBSCAN algorithm. Its clustering-based approach allows for the detection of anomalies throughout the entire dataset without distinguishing between training and testing data. Thus, DBSCAN offers a more comprehensive scan for anomalies, underscoring its relative advantage in this context.

In light of these observations, while the Convolutional Autoencoder presents a potent strategy for anomaly detection, it is outweighed by the DBSCAN algorithm's advantages in this specific application, including broader coverage of variables, computational efficiency, and user interpretability.

6 Discussion

Due to its successful performance and advantages over the other models, the DBSCAN method was chosen for subsequent analysis on the remaining data. Its proficiency in detecting and locating anomalies, its easy-to-understand output, and its efficient computational demands contributed to it being selected as the most suitable approach for this particular anomaly detection task.

To conduct a comprehensive test for full implementation, the model was run on a set of seven products from each of the five branches. These products were selected by choosing one item from each product category, ensuring that the same product was used across all branches. The selection criteria considered both the frequency of the product in the dataset and its presence in each branch. Therefore, 35 optimal combinations were then used to run the model.

The experimental combination of branch 304 and product 2302160 yielded the results depicted in Table 2. It's important to note that the values are scaled, resulting in an average of 0 for each feature and a standard deviation of 1. This scaling allows us to identify anomalies based on their distance from zero.

Date	SALES	STOCK	WASTE	OP_COUNT	STOCKTAKE
2021-11-24	-0.146248	2.183307	-0.170558	-0.946302	2.623691
2022-07-17	-0.548968	-0.333093	0.590304	-1.904136	-0.313676
2022-12-21	-0.073597	0.367301	-0.299213	0.967029	0.042003
2023-01-01	-0.197310	-0.817258	1.235505	-3.756354	-0.607506
2023-01-19	0.033340	0.650530	3.756354	1.228634	0.090127
2023-01-21	1.714879	7.230984	-0.370579	-0.357708	-6.312871

Table 2: Experimental Branch and Product Anomalies

It can be observed that the Sales feature has only one anomaly for this particular branch and product combination. This anomaly occurred on 2023-01-21 and represented a noticeable spike in sales. Additionally, two anomalies are present in the Stock feature: one on 2023-11-24 and another on 2023-01-21. The latter anomaly seems to be correlated with the sales spike, suggesting that the sudden increase in stock led to a corresponding increase in sales for this item. The most significant number of anomalies (6 anomalies) appeared in the "Waste" and "Op_Count" features. This suggests that for this particular product, "Waste" and "Op_Count" play a crucial role. Furthermore, the "Stocktake" feature showed two major anomalies: a significant spike and a notable dip. Notably, these anomalies occurred at the same time as the anomalies in the "Stock feature". This observation suggests that there may have been a miscount in the stock in both "Stocktake" and "Op_count", resulting in a miscount in the amount of waste. Further, resulting in an unexpected influx or decrease in stock levels, subsequently impacting sales.

The experimental analysis was further expanded by running the 35

combinations to gain a more granular understanding. The results were visualized in Figure 22, which shows the frequency of anomalies by date.

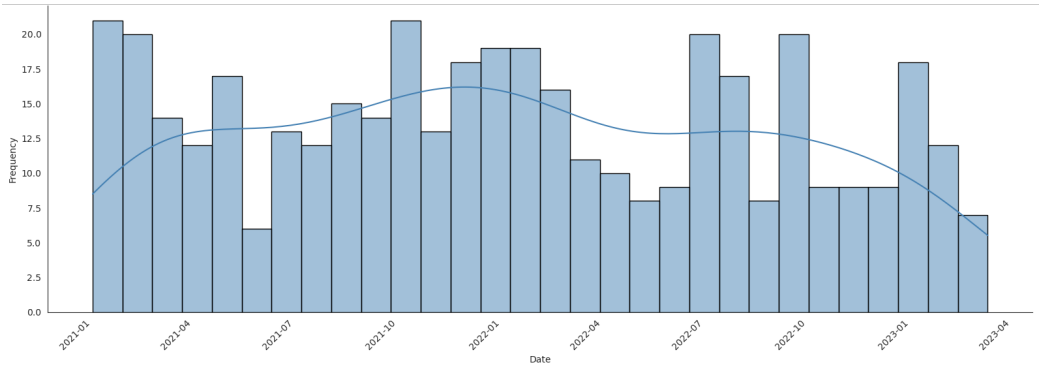

Figure 22: Distribution of Anomalies for 35 Combinations

Interestingly, the data reveals that a majority of anomalies occur at the beginning of the year, particularly in the month of January. Several factors could explain this pattern. One possibility is that there may be a lack of organization or efficiency during seasonal periods of chaos, leading to anomalies in the data. Another potential factor could be the presence of promotions or special offers on certain items during this time of year, causing fluctuations in sales and stock levels.

The DBSCAN algorithm was applied to all 12,000 product/branch combinations for full implementation. Figure 23 displays the results of the DBSCAN run on the entire dataset, and the corresponding histogram reveals that the number of anomalies over time is almost evenly distributed, with minor fluctuations throughout the period under consideration.

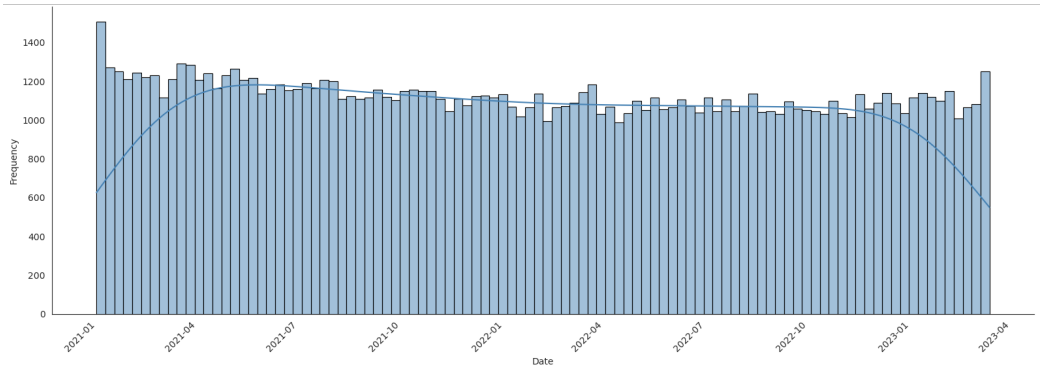

Figure 23: Distribution of Anomalies for Entire Dataset

Additionally, Table 3 highlights the top ten dates with the most anomalies. The date 2021-01-02 stands out with the highest number of anomalies, it can be assumed that this is due to the nature of the time of year. However, the dates with the most anomalies are randomly dispersed throughout the years and do not follow any apparent seasonal pattern. Thus, upon further investigating the full dataset it can be concluded that the nature of the anomalies are complex and non-cyclical.

Date	Frequency
2021-01-02	193
2021-04-10	187
2021-01-01	183
2021-02-13	182
2021-03-24	180
2021-06-16	180
2021-03-25	178
2021-05-18	175
2021-03-16	175
2021-06-09	174

Table 3: Top 10 Anomaly Dates

7 Limitations

While anonymizing the data was crucial for John Lewis to protect confidential company information, it inadvertently posed a drawback for this specific task. The anonymization process resulted in the loss of valuable information and insights that could have been essential for a more comprehensive analysis. For instance, including variables such as product names or promotional information could have provided deeper insights into the reasons and mechanisms behind specific anomalies observed in the data. Due to the absence of such details, the analysis is limited in its ability to understand the full context and implications of the anomalies.

Additionally, the absence of labels in the data necessitated an unsupervised approach to developing anomaly detection algorithms. However, this lack of ground truths in unsupervised learning scenarios posed significant challenges in accurately evaluating model performance. Without labeled data, standard metrics that are typically used to assess model quality were not applicable. As a result, the limited resources available for labeling further exacerbated the issue, restricting the potential for employing supervised learning techniques and hindering the extraction of deeper insights from the dataset. Furthermore, the absence of labels meant that there was no target variable with associated features, rendering the development of predictive models impossible. Consequently, the models created can only identify anomalies within the existing dataset rather than predict future anomalies.

The lack of features and a target variable can be attributed to the anonymization of the dataset, which resulted in the removal of valuable in-

formation. Since the data was limited, the only variable that could have been employed as a target variable for supervised learning was "AUTO_ADJ_NBAL _SUS". Values in this column that were not labeled with a 0 could have been assigned a label of 1, enabling binary classification tasks through supervised learning. However, the problem with this variable is that it has a severe class imbalance issue. In a sample of 2,000 instances, there would be 1,987 zeros and only 13 ones. Given the dataset's size of 6.8 million rows, traditional class balancing algorithms like Synthetic Minority Oversampling Techniques (SMOTE) would generate synthetic data to balance the classes. Nevertheless, this approach would significantly increase the dataset size and create synthetic data that might not be representative of real world data, making it impractical due to the limited time and computational resources available. Therefore, even if it were possible to combine features to create a target variable, the class imbalance and the interdependence of variables would still pose challenges. The relationship between the variables and the presence of complex dependencies make it difficult to achieve meaningful and accurate predictions through traditional supervised learning techniques.

An additional challenge arises from the computational costs and run time associated with certain advanced techniques. For example, implementing a Generative Adversarial Network (GAN) with a CAE has the potential to yield powerful results. However, due to its high expense and time-consuming nature, such a complex algorithm could not be feasibly executed within the allocated resources. As a result, the limitations in computational capabilities prevented the exploration and application of numerous powerful deep learning algorithms, which could have otherwise provided valuable insights.

8 Future Work

Considering the aforementioned limitations discussed above, it is apparent that several alternative methods could have been employed with additional time and resources. One such approach involves utilizing the results of DBSCAN to create labels for the data. By introducing an additional feature that denotes 1 for anomalous observations and 0 for normal instances, it becomes possible to address the class imbalance issue. This new feature would be created based on the interaction of variables that lead to anomalous behavior.

With this feature acting as a target variable, along with other predictive features (assuming the data were not anonymized), supervised learning algorithms could be applied effectively. This integration of supervised learning methods would allow for not only the identification but also the prediction of anomalies. Consequently, leveraging this approach could potentially unlock more comprehensive insights and enable proactive measures to tackle anomalies in the dataset, provided sufficient time and computational resources are available.

Another promising approach that could have been considered is the USAD (Unsupervised Anomaly Detection for Multivariate Time-Series) method developed by researcher Audibert et al. (2020). As outlined in the literature review, this innovative approach combines the strengths of both a CAE and a GAN to overcome the intrinsic limitations of the CAE's reconstruction error threshold and the issues of collapse and non-convergence often encountered by GANs.

The CAE component enables robust feature extraction and reconstruction, while the GAN component enhances the model's ability to capture complex patterns and variations within the data.. By deploying this two-phase model, the USAD method has the potential to provide more accurate and comprehensive anomaly detection, yielding deeper insights into the data. However, the adoption of such an approach requires significant computational resources and careful parameter tuning, which has been limited under the given constraints.

Moreover, an ensemble method can be formed by combining a DBSCAN with a CAE. As highlighted in the results, the DBSCAN identified 6 anomalies in the experimental product/branch pair, while the CAE detected 5. With sufficient time and resources, a detailed analysis of the anomalies identified by the CAE could be conducted to pinpoint their specific locations within the time series data. By comparing these results with those of the DBSCAN, the ensemble model would then assess the discrepancies between the two approaches and identify the most crucial anomalies recognized by both models. This ensemble approach capitalizes on the strengths of both techniques, leveraging the unsupervised nature of the DBSCAN and the powerful feature extraction capabilities of the CAE.

Incorporating real-time data would enable the development of an online CPD algorithm, which could promptly identify deviations as they occur. This proactive approach would empower John Lewis to take immediate action on emerging issues that might potentially have adverse effects on the company. For instance, if an overstock situation arises due to a miscount of the current stock, the algorithm could automatically detect the anomaly and trigger actions to address the problem, such as canceling the order.

By leveraging such an algorithm, John Lewis could optimize its operational efficiency by swiftly mitigating potential risks, providing a competitive advantage in managing supply chain and inventory challenges. Thus, ensuring that the company remains agile and responsive to dynamic market conditions. However, implementing such an online CPD system would necessitate continuous monitoring to effectively capture and respond to anomalies as they unfold.

9 Conclusion and Contributions

Conclusion

In conclusion, this study conducted a thorough investigation to identify the most optimal anomaly detection approach for the nature of the John Lewis data, which lacked labels and required unsupervised or supervised-adapted unsupervised techniques. This investigation highlighted the most applicable models from statistical approaches, machine learning and deep learning approaches. This study commenced with a PCA to extract the most important features, given the absence of labeled data for manual feature selection. Once features were selected based on importance, modeling was conducting. The first model adopted was the ARIMA model. The subsequent attempt to adopt the traditionally supervised ARIMA model, used for forecasting, as an unsupervised anomaly detection task proved unsuccessful due to data characteristics not meeting the model's requirements.

As an alternative, the study adopted CPD, a statistical approach, which proved highly successful in identifying deviations in the time series data. However, a limitation of this method was that deviations were detected based solely on the most important feature. Thus, deviations in less important features might not have been identified.

To gain a deeper understanding of the anomalies, the study explored the machine learning approach DBSCAN, which effectively identified anomalies in every feature, making it the most successful method employed. Subsequently, a deep learning model, the CAE, was also utilized, and it demonstrated excellent performance in identifying anomalies. However, it faced

challenges in handling multiple variables effectively and incurred significant computational costs.

Overall, the study presented a comprehensive analysis of various statistical, machine learning, and deep learning approaches, covering numerous methods for anomaly detection. Among the examined models, the DBSCAN emerged as the most successful approach for the John Lewis data, providing accurate and robust anomaly detection without the need for labeled data.

Contributions

This study makes significant contributions to the field of anomaly detection in industry by thoroughly exploring a range of approaches and identifying the best method for handling large, unlabeled, and multivariate datasets. The contributions of this study can be categorized into technical, operational, and company-specific aspects.

From a technical standpoint, the study developed and presented multiple algorithms with an emphasis on reproducibility. These algorithms can be adapted for various anomaly detection use cases, making them valuable tools for future research and practical applications. Particularly, the DBSCAN model stands out as a promising approach, and it can be used alone or in conjunction with other methods, as proposed in the future work section, to achieve efficient and granular anomaly detection.

In terms of operational impact, this study has successfully identified anomalies in the time series data, providing essential insights that can lead to necessary actions and improvements. The DBSCAN model, along with other investigated models, can enable John Lewis to gain a deeper under-

standing of the anomalies specific to each product in every branch. This valuable information can be instrumental in fine-tuning business operations and processes for enhanced performance and efficiency.

Regarding company-specific contributions, John Lewis can directly integrate the findings of this study to take corrective actions in response to the identified anomalies. These actions can prevent potential issues, optimize inventory management, and ultimately lead to cost savings and increased revenue. Furthermore, leveraging the insights gained from the anomaly detection process can help John Lewis better understand its business performance and operational aspects, enabling the company to make data-driven decisions that further enhance its overall success and competitive edge.

References

Abraham (1989), 'Outlier detection and time series modeling', *Technometrics* **31**(2), 241–248.

Adhikari, R. and Agrawal, R. (2013), 'An introductory study on time series modeling and forecasting', *arXiv preprint arXiv:1302.6613* .

Aminikhanghahi, S. and Cook, D. (2016), 'A survey of methods for time series change point detection', *Knowledge and Information Systems* . Accessed: 13 August 2023.
URL: *https://link.springer.com/article/10.1007/s10115-016-0987-z*

Andres, D. (2022), 'Introduction to arima models - ml pills, machine learning pills'. Accessed: 13 August 2023.
URL: *https://mlpills.dev/time-series/introduction-to-arima-models/*

Audibert, J. et al. (2020), Usad, *in* 'Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining [Preprint]'.

Banoula, M. (2023), 'What is cost function in machine learning: Simplilearn'. Accessed: 14 August 2023.
URL: *https://www.simplilearn.com/tutorials/machine-learning-tutorial/cost-function-in-machine-learning*

Boriah (2008), Similarity measures for categorical data: A comparative evaluation, *in* 'Proceedings of the 2008 SIAM International Conference on Data Mining'.

Box, G. E., Jenkins, G. M. and Reinsel, G. C. (2008), *Time Series Analysis: Forecasting and Control*, 4th edn, John Wiley & Sons, Hoboken, NJ, USA.

Celik, M. and Dokuz, A. (2011), Anomaly detection in temperature data using dbscan algorithm, *in* 'IEEE'. Accessed: 13 August 2023.
URL: *https://ieeexplore.ieee.org/document/5946052/*

Chandola (2009), 'Anomaly detection', *ACM Computing Surveys* **41**(3), 1–58.

Chen, Z. and Lee, B. (2018), Autoencoder-based network anomaly detection, *in* 'IEEE Conference'. Accessed: 13 August 2023.
URL: *https://ieeexplore.ieee.org/document/8363930*

Downey, A. B. (2008), 'A novel changepoint detection algorithm', *arXiv preprint arXiv:0812.1237* .

Edgeworth, F. Y. (1887), 'On discordant observations', *Philosophical Magazine* **23**(5), 364–375.

Ehsani, N., Aminifar, F. and Mohsenian-Rad, H. (2022), 'Convolutional autoencoder anomaly detection and classification based on distribution pmu measurements', *IET Generation, Transmission & Distribution* **16**(14), 2816–2828.

Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. (1996), A density based algorithm for discovering clusters in large spatial databases with noise, *in* 'KDD-96 Proceedings', pp. 226–231.

Friedman, J. H. and Rafsky, L. C. (1979), 'Multivariate generalizations of the wald-wolfowitz and smirnov two-sample tests', *The Annals of Statistics* **7**(4), 697–717.

GlobalData (n.d.), 'John lewis partnership plc company profile - john lewis partnership plc overview', `https://www.globaldata.com/company-profile/john-lewis-partnership-plc/`. Accessed: 09 August 2023.

Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. C., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K. and Stanley, H. E. (2000), 'Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals', *Circulation* **101**(23), e215–e220.
URL: *http://circ.ahajournals.org/cgi/content/full/101/23/e215*

Goodfellow, I. and Mirza, M. (2014), 'Generative adversarial nets'. Accessed: 13 August 2023.
URL: *https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf*

Géron, A. (2019), *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd edn, O'Reilly Media, Inc.

Hinton, G. and Salakhutdinov, R. (2006), 'Reducing the dimensionality of data with neural networks', *Science* **313**(5786), 504–507.

Holland (2019), 'Principal components analysis pca - uga', `http://strata.uga.edu/8370/handouts/pcaTutorial.pdf`. Accessed: 09 August 2023.

Hotelling, H. (1933), 'Analysis of a complex of statistical variables into principal components', *Journal of Educational Psychology* **24**(6), 417–441.

Huber, P. (1974), *Robust Statistics*, Wiley, New York.

Jaber, S. (2023), '2023 edge of ai technology report'. Accessed: 10 August 2023.
URL: *https://www.wevolver.com/article/2023-edge-ai-technology-report*

Jose (2023), 'How anomaly detection can help today's business world', `https://blog.accubits.com/how-anomaly-detection-can-help-todays-business-world/`. Accessed: 09 August 2023.

Kawahara, Y. and Sugiyama, M. (2009), Sequential change-point detection based on direct density-ratio estimation, *in* 'SIAM International Conference on Data Mining', pp. 389–400.

Kingma, D. and Welling, M. (2019), 'An introduction to variational autoencoders', *Foundations and Trends® in Machine Learning* **12**(4), 307–392.

Kuo, C. and Tsang, S.-S. (2022), 'Detection of price manipulation fraud through rational choice theory: Evidence for the retail industry in taiwan', *Security Journal [Preprint]* .

Lee, T.-H., Ullah, A. and Wang, R. (2020), Bootstrap aggregating and random forest, *in* P. Fuleky, ed., 'Macroeconomic Forecasting in the Era of Big Data', Springer International Publishing, New York, pp. 389–429.

Markou, M. and Singh, S. (2003), 'Novelty detection: A review-part 1: Statistical approaches', *Signal Processing* **83**(12), 2481–2497.

Masci, J., Meier, U., Cireşan, D. C. and Schmidhuber, J. (2011), Stacked convolutional auto-encoders for hierarchical feature extraction, *in* 'Advances in Neural Information Processing Systems', pp. 1097–1105.
URL: *https://link.springer.com/chapter/10.1007/978-3-642-21735-7*$_7$

Moschini, G. et al. (2021), 'Anomaly and fraud detection in credit card transactions using the arima model', *MDPI* . Accessed: 13 August 2023.
URL: *https://www.mdpi.com/2673-4591/5/1/56*

Pozzolo, A. (2014), 'Learned lessons in credit card fraud detection from a practitioner perspective', *Expert Systems with Applications* **41**, 4915–4928.

Raval (2021), 'Why your business needs anomaly detection', `https://datatechvibe.com/data/why-your-business-needs-anomaly-detection/`. Accessed: 09 August 2023.

Ren, H. et al. (2019), Time-series anomaly detection service at microsoft, *in* 'Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining [Preprint]'.

Rousseeuw, P. J. and Leroy, A. M. (1987), *Robust Regression and Outlier Detection*, John Wiley & Sons, Inc.

Sejnowski, T. and Hinton, G. (1999), *Unsupervised Learning: Foundations of Neural Computation*, Massachusetts Institute of Technology, Cambridge, Massachusetts.

Shyu, M.-L. L., Chen, S.-C., Sarinnapakorn, K. and Chang, L. (2003), A novel anomaly detection scheme based on principal component classifier, Technical report, DTIC Document.

Smith (2002), 'A tutorial on principal components analysis - otago', `http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf`. Accessed: 09 August 2023.

Song, X., Wu, M., Jermaine, C. and Ranka, S. (2007), 'Conditional anomaly detection', *IEEE Transactions on Knowledge and Data Engineering* **19**(5), 631–645.

Teng, H., Chen, K. and Lu, S. (1990), Adaptive real-time anomaly detection using inductively generated sequential patterns, *in* 'Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy', IEEE Computer Society Press, pp. 278–284.

Truong, C., Oudre, L. and Viyatis, N. (2020), 'Selective review of offline change point detection methods', *arXiv preprint arXiv:1801.00718* .
URL: *https://arxiv.org/pdf/1801.00718.pdf*

Verma, Y. (2021), 'Complete guide to dickey-fuller test in time-series analysis'. Accessed: 14 August 2023.
URL: *https://analyticsindiamag.com/complete-guide-to-dickey-fuller-test-in-time-series-analysis/*

Xing, T. (2019), 'Overview of sr-cnn algorithm in azure anomaly detector'. Accessed: 13 August 2023.

URL: *https://techcommunity.microsoft.com/t5/ai-customer-engineering-team/overview-of-sr-cnn-algorithm-in-azure-anomaly-detector/ba-p/982798*

10 Appendix

10.1 Data

LINE_STOCK_SUS = End of day stock. It is calculated using the following equation:

LINE_STOCK_SUS =Stock Yesterday + Deliveries Today − Sales Today
− Explained Wastage Today (LINE_WSTG_SUS)
− Unexplained Wastage Today (UW_OP_COUNT_SUS)

Moreover, if LINE_STOCK_SUS = 0 and AUTO_ADJ_NBAL_SUS ≠ 0, LINE_SALES_SUS = AUTO_ADJ_NBAL_SUS + Other expected stock

10.2 PCA

The covariance matrix for the PCA takes the following form:

$$\begin{bmatrix} \mathrm{Cov}(a,a) & \mathrm{Cov}(a,b) & \mathrm{Cov}(a,c) & \mathrm{Cov}(a,d) & \mathrm{Cov}(a,e) & \mathrm{Cov}(a,f) \\ \mathrm{Cov}(b,a) & \mathrm{Cov}(b,b) & \mathrm{Cov}(b,c) & \mathrm{Cov}(b,d) & \mathrm{Cov}(b,e) & \mathrm{Cov}(b,f) \\ \mathrm{Cov}(c,a) & \mathrm{Cov}(c,b) & \mathrm{Cov}(c,c) & \mathrm{Cov}(c,d) & \mathrm{Cov}(c,e) & \mathrm{Cov}(c,f) \\ \mathrm{Cov}(d,a) & \mathrm{Cov}(d,b) & \mathrm{Cov}(d,c) & \mathrm{Cov}(d,d) & \mathrm{Cov}(d,e) & \mathrm{Cov}(d,f) \\ \mathrm{Cov}(e,a) & \mathrm{Cov}(e,b) & \mathrm{Cov}(e,c) & \mathrm{Cov}(e,d) & \mathrm{Cov}(e,e) & \mathrm{Cov}(e,f) \\ \mathrm{Cov}(f,a) & \mathrm{Cov}(f,b) & \mathrm{Cov}(f,c) & \mathrm{Cov}(f,d) & \mathrm{Cov}(f,e) & \mathrm{Cov}(f,f) \end{bmatrix}$$

10.3 ARIMA

The table below shows the results for the AD Fuller Test on sales.

Table 4: ADF Test Results

Statistic	**Value**
ADF Statistic	-5.0841330294589095
p-value	$1.5086296904873828 \times 10^{-5}$
Critical Values	
1%	-3.438602251755426
5%	-2.8651823762743245
10%	-2.5687095387840673

10.4 Project Management

The Gannt chart below shows how each individual task was spaced out across the term and the table below shows the dates and minutes for the 8 meetings with the supervisor.

Date	Week	Literature Review	Data Exploration	Data Visualisation	Data Pre-processing	Model Designs	Model Training	Entire Dataset	Dissertation	Presentation
24/04/2023	1									
01/05/2023	2									
08/05/2023	3									
15/05/2023	4									
22/05/2023	5									
29/05/2023	6									
05/06/2023	7									
12/06/2023	8									
19/06/2023	9									
26/06/2023	10									
03/07/2023	11									
10/07/2023	12									
17/07/2023	13									
24/07/2023	14									
31/07/2023	15									
07/08/2023	16									
14/08/2023	17									

(a) Project Management

Meeting Date	Meeting Minutes
01/05/2023	Introduced project to supervisor. Explained the data variables, the structure/size of the data, and the nature of the data (unlabeled). Walked through some initial exploratory analysis, and gave me tips on how to proceed, i.e. analyze specific products. Discussed an issue with computing power from John Lewis.
02/06/2023	Walked supervisor through the completed exploratory data analysis. Discussed which visualizations were the most important and least important. Supervisor suggested that I do further analysis to try to answer business questions.
09/06/2023	Presented the data analysis which answered business questions. Discussed potential ideas for modeling approaches. Explained research from the literature review which gave ideas for potential models to explore, for example, ARIMA, isolation forest, and variational autoencoder.
23/06/2023	Presented modeling attempts for CPD model and DBSCAN model and discussed issues to be improved with the models. Discussed what the anomalies are and how I will define them as well as my overall research question. Further, discussed potentially engineering a target variable for supervised learning.
07/07/2033	Presented completed models and methodology. Discussed issues to be improved in models, for example, potentially implementing a moving batch size for the autoencoder. Decided that supervised learning would not work for this particular project. However, I presented how I attempted to adapt an ARIMA model for an unsupervised learning task.
21/07/2023	Discussed progress of writing. Decided on the best structure possible for the paper, for example, setting the literature review before the data exploration. Moreover, discussed the expectations for the presentation. Supervisor advised that I include the debate mate questions at the end of the presentation.
04/08/2023	Discussed the final writing stages of the paper and the inclusion of the time management section. Decided the best approach for presenting the time management. Also discussed, including a "gap in the literature" section to my literature review.
18/08/2023	Walked through presentation. Supervisor gave me tips on how to improve presentation.

(a) Meetings Minutes

Figure 25: Project Management and Meeting Minutes