Universiteit Leiden

Master Computer Science

Automated Machine Learning of Customer-Specific Gaussian Mixture Models for Unsupervised Anomaly Detection

Name: Yan Liang
Student ID: s2434806

Date: [16/07/2021]

Specialisation: Advanced Data Analytics

1st supervisor: Prof. Dr. Ir. Joost M.W. Visser
2nd supervisor: Dr. Wojtek J. Kowalczyk

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Acknowledgements

First and foremost, I would like to express my sincere gratitude to Prof. Dr. Ir. Joost M.W. Visser and Dr. Wojtek J. Kowalczyk for their guidance, support and supervision through my internship and the subsequent thesis project. Thank you for constantly challenging me and providing insightful feedback that led me to explore new areas of inquiry.

Furthermore, my humble appreciation go out to the entire team at Avery Dennison for providing me with the opportunity to tackle real world data science challenges with a talented and highly motivated team. I would especially like to thank Nicole Ostlund, Data Scientist at Avery Dennison, for supervising me during the initial phase of my internship and introducing me to the challenges that are being researched within the sales and finance departments. In addition, I would like to thank Marc van Vliet for supervising me during the latter stages of my internship, providing me his expansive knowledge and his guidance in finalizing my internship research project successfully. Finally I would like to thank Chris Osinski, Ana O. Corrales, Sandra Tarr and Giovanni Matarazzo of the Business Intelligence and Sales teams for supporting me in this journey.

Additionally, I would like to thank my fellow classmates at LIACS for providing me with many insightful discussions and memorable times. In particular, I would like to thank Ruduan Plug for providing the elegant thesis layout and design, and also for helping me peer-review the experiment and pointing me towards new techniques. In addition, I would like to thank my cousin, Dr. MD. Liang Chen, for inspiring me to take this journey abroad.

Finally and most importantly, my endless appreciation go out to my parents for always supporting me no matter what. For providing me with a warm home where I could explore my interests and for providing me with the courage to expand my horizons and explore the world.

Abstract

Background Avery Dennison [1] is a global material company specializing in producing labelling and functional material products. Because of the variety of the products, it is hard for the sales team to keep track of every customer. In the sales orders, the changes of some customers are hard to detect not only because of the great amount of the entire customers in the company, but also because changes are various under many aspects and categories.

Aim The paper focuses on detecting customer abnormal buying patterns based on unlabelled annual sales transaction orders by using the unsupervised learning methods. Automation of anomaly detection is the main goal to support the sales department in identifying abnormal customer behaviors.

Method We use three design science research cycles in guiding our project. In the relevance cycle, we do the data exploration and get a baseline idea of anomalous data by using an unsupervised method DBSCAN. In the design cycle, we elaborate and give a detail evaluation for an automated forecasting model in detecting abnormal orders and propose a mixture model for finding anomalies. In the rigor cycle, we validate our model through labelling a small sample of anomalous data supported by sales representatives.

Results We find three locally optimal models for daily, weekly and yearly aggregated transaction data, and purpose a prophet-derived Gaussian mixture model in the form of the bi-modal distribution for different product categories. In addition, we provide a methodology to verify the performance of the unsupervised model on labelled sample data.

Conclusion We conclude that our approach can give us the insight in not only the anomaly distribution and the statistical properties of anomalies per customer and per item but also helping identify the cause and effect of transaction data anomalies for sales department.

Contents

1 Introduction

Avery Dennison [1] is a global material company specializing in products such as labels and packaging material. Operating in over 50 countries, Avery Dennison addresses a diverse global market with complex financial dynamics. As the market behaviour of the various product lines can vary greatly by customer, region and time, the analysis of sales dynamics needs to incorporate these components to produce versatile predictive and diagnostic models.

The primary challenge that we analyze in this study is abnormal customer behaviour within this complex system. In daily sales, the changes of customers buying patterns are hard to detect, not only because of the great amount of the entire customers in the company, but also because the changes are variable under product, spatial and temporal properties. Another challenge is there is no defined standard for anomalous behaviour within the domain, and therefore no objective labelling of transactional data. Therefore, it is necessary to understand and define the conditions under which customer buying patterns are abnormal, which form the basis for our anomaly detection model.

In this thesis, the data we focus on is sales orders data which we analyze for Avery Dennison, focusing on the transactions through the EU data warehouse. The sales orders dataset contains both categorical variables such as customer name and numerical variables such as ordered value in Euros. In the dataset, the daily orders are recorded if a customer makes an order for a specific product with the relevant information such as ordered volume and value.

We use three design science research cycles [2] in guiding the project. Since the project cooperates with sales department, in the initial cycle (i.e. relevance cycle), we transform the requests into research questions and define the abnormal buying patterns. On the next cycle which is design cycle, we build our anomaly detection model and do the relevant experiments and evaluate the model performance. On the final cycle (i.e. rigor cycle), some result validation works have been done by the sales representatives.

The rest of the thesis is organized as follows. Section 2 provides the related works for time series analysis and existing unsupervised anomaly detection methods. In section 3, we give preliminaries about the design science research and the time series analysis. In section 5, we elaborate the details about the sales transaction data and do the data exploration in unsupervised analysis of the sales transaction data to get a baseline idea of anomalous data. We discuss the baseline time series exploration, provide a detail evaluation of the automated forecasting model, purpose a mixture model in the form of a bi-modal distribution for anomalies in section 6 and label a small sample of validation data supported by sales representatives. We conclude our contributions, answer our research questions and discuss about future work in section 7.

1.1 Research Statement

There are some questions that need to be stated for this research.

1. Can we use statistical uncertainty modelling to effectively detect anomalies in sales data?

2. How can we utilize unsupervised learning to create and optimize an anomaly detection model to classify our data?

3. How can the performance of the resulting anomaly detection models be determined in a practical setting?

2 Related Work

In the presence of unlabelled data, typical classification models that train on class labels cannot be used without making assumptions over the class of each instance. In addition, the classification of a data element may depend largely on the underlying distribution of the data. To rectify this we must look towards unsupervised methods, that allow us to extract analyzable patterns or metrics from which we can estimate the probability for a data point being an outlier that can be classified as an anomaly.

Clustering is one of the unsupervised learning methods. There are some popular clustering algorithms in unsupervised machine learning such as K-Means described in detecting unknown network intrusions or attacks [3], however they are susceptible to strong outliers unless the clusters are disjoint. Another clustering methods is called DBSCAN which is used in clustering and detecting noise by grouping the reachable neighbors and finding the outliers [4].

In time series analysis, the stationary time series means the changes of values do not depend on time. For our project, we focus on the non-stationary time series which the values change through time and are affected by trend, seasonalities, etc. Changepoint analysis detects the points that affect the changes on the time series, Dette et al. [5] purpose the changepoint and correlation analysis for non-stationary time series in detecting relevant changepoints.

The goal of our project is to detect time series anomalies. Since the data is unlabelled, we focus on unsupervised anomaly detection methods. Munir et al. [6] purpose a convolutional neural network (CNN) based prediction model called DeepAnT for unsupervised learning on a time-scale that used the unlabelled data in predicting the time series normal behaviour and DeepAnt can be applied for both uni-variant and multi-variant. Audibert et al. [7] purpose an UnSupervised Anomaly Detection for multivariate time series (USAD) model based on an adversarial autoencoder architecture in detecting anomalies while doing unsupervised training.

For detected anomalies, a mixture of a bi-modal distribution model can be performed. Zong et al. [8] purpose an advanced Gaussian Mixture Model (GMM) with the non-linear autoregressive component for univariate time series in satisfying the stationarity and ergodicity conditions. Zong et al. [8] purposes a similar GMM architecture by using auto-encoding method in modeling the error. Braei and Wagner [9] conduct a survey on the state of the art in evaluating and comparing the anomaly detection models not only in statistical approaches such as auto-regression but also deep neural network methods such as CNN on univariate time series.

Preliminaries

3

In this chapter, we discuss about the preliminaries of our research. In section 3.1, we discuss the three cycles of design science and how we use design science in guiding our project. In section 3.2, we describe the preliminaries of the stochastic effects and auto-correlation for time-series data.

3.1 Design Science

The project described in this thesis is an application of the design science research. The idea of the three design science research cycles (i.e. relevance cycle, rigor cycle, design cycle) is brought by Hevner [2] where he claims that these three cycles help enhance the understanding of design science in the Information System (IS) field. In [10], Iivari summarizes twelve theses in order to give an overview of the main properties of IS as a design science based on ontology, epistemology, methodology, and ethics.

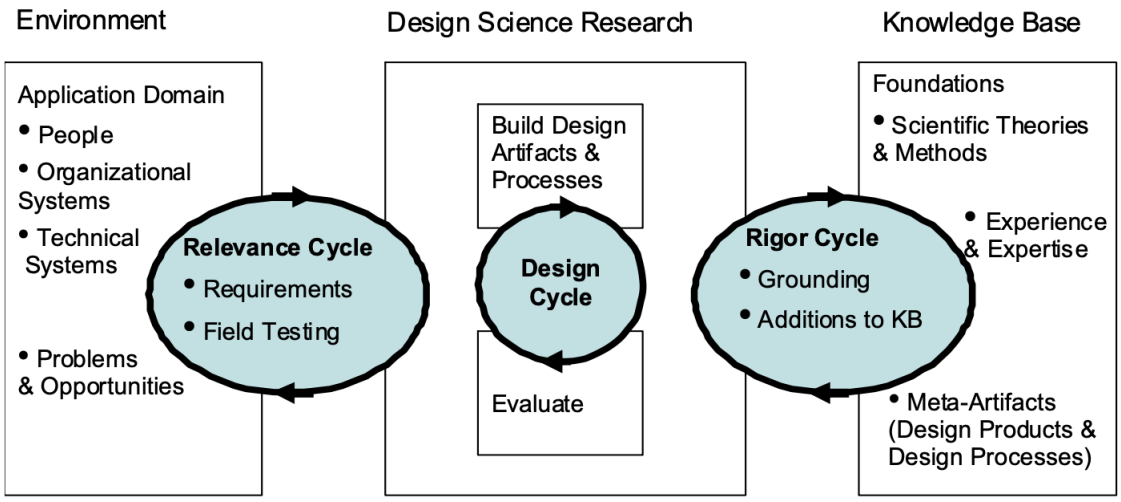


Fig. 3.1.: Design Science Research Cycle [2]

Figure 3.1 shows the three design science research cycles introduced by Hevner [2]. In general, the **Relevance Cycle** represents the initiate stage of the design science research project that connects both the application domain such as people, organizational systems and the requirements such as research problems while evaluating if the design artifact meets the final

research results criteria. If the requirements are wrong or the design artifact will have negative impact in practice. In this context, field testing is used to evaluate the impact of the prototype to which demonstrates the performance and usability before it is used in operation. This process is used to validate designs in the current design cycle before they make it to the market.

The **Design Cycle** is bridge that connects both the relevance cycle and the rigor cycle. It provides the evaluation for the design artifact to the relevance cycle after applying the scientific research methods provided in the rigor cycle based on the requirement inputs from the relevance cycle. Since the changes of methods in the rigor cycle and the feed-backs provided in the relevance cycle, design cycle is the core of the design science research and it may involve multiple of iterations.

The **Rigor Cycle** indicates the process of collecting and developing the scientific theories and methods from both the innovative experience and expertise of the research domain and the already existed meta-artifacts in the application domain [2]. In this thesis, we consider and test both sources mentioned above.

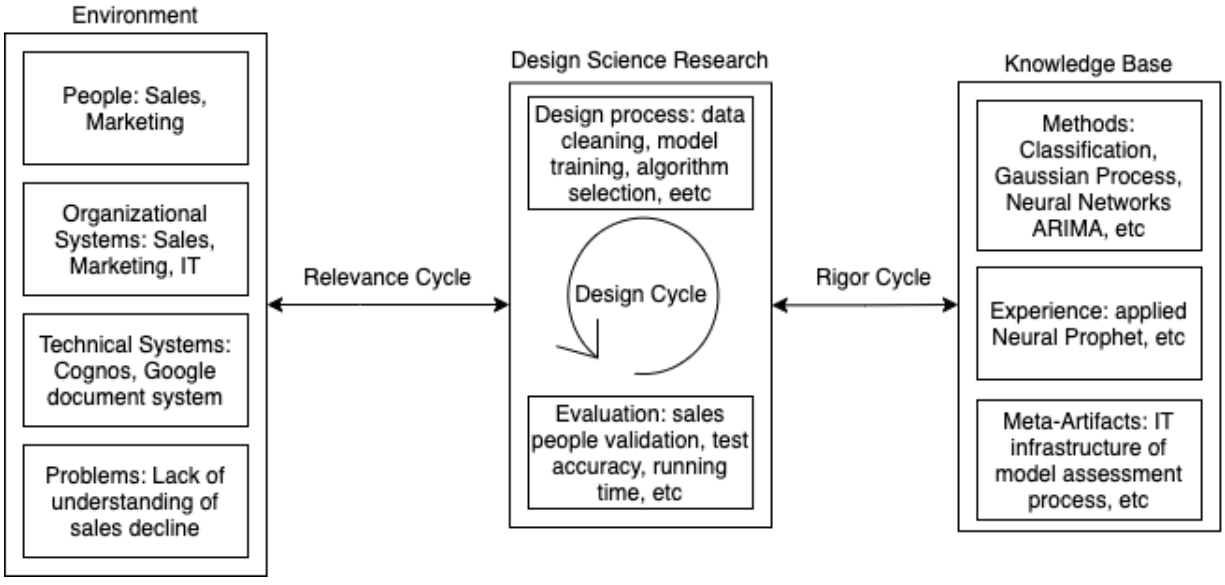


Fig. 3.2.: Application of Design Science Research Cycle

The application of our research project on three cycles of design science is shown in Figure 3.2. At the initial stage of our project, the application environment includes sales representatives to gain insights of the transactional data, organizational systems such as the Sales Department, Marketing Department, and IT Department, and technical systems which are Cognos system

and Google Cloud Platform (GCP). Cognos is a business intelligence tool created by IBM, it helps users with and without technical background to manage and analyze the data easily. In our project, we get the recent three-year sales orders data from Cognos and use GCP to share documents and communicate with each other. The problem that drives us to do the anomaly detection research is the unexplained decline in sales of some products. Sales people find that there have been declined sales patterns among some customers in recent years and marketing people are interested in understanding the sales dynamics for various products. Both of the sales and marketing departments would like to know the changing patterns in detail on a time-scale basis.

During the design cycle, we continuously get feedbacks from sales and marketing people and improve the model during the design cycle. The design process contains the data preprocessing, algorithm selection, model training, etc. Figure 3.3 shows the design process flow chart of our project.

The rigor cycle provides the existing knowledge base (i.e. methods) with additional extensions of the original approaches, the new meta-artifacts, and the learning experience from doing the research experiments and field testing [10]. In Figure 3.2, we consider the existing unsupervised learning methods such as DBSCAN, and auto-regression and also the innovative approach such as Prophet. Experiment, as mentioned in [10], is one of the additional knowledge that represents the cutting-edge theories or methods in the research application domain. In our project, we consider unsurprising methods such as the application of DBSCAN and Prophet. Another additional knowledge is meta-artifact which means the existent design products and processes applied in the past research domain. As mentioned in [11], meta-artifact is to improve the development of the information system. After reviewing the meta-artifacts mentioned in [12] and [13], we propose the new meta-artifact shown in Figure 3.3.

3.2 Time Series Analysis

In comparison to cross sectional data, which is used to compare the sample differences within sample space for various instances, for time series data we find that there is ordering in the data with serial dependence over the observations [14].

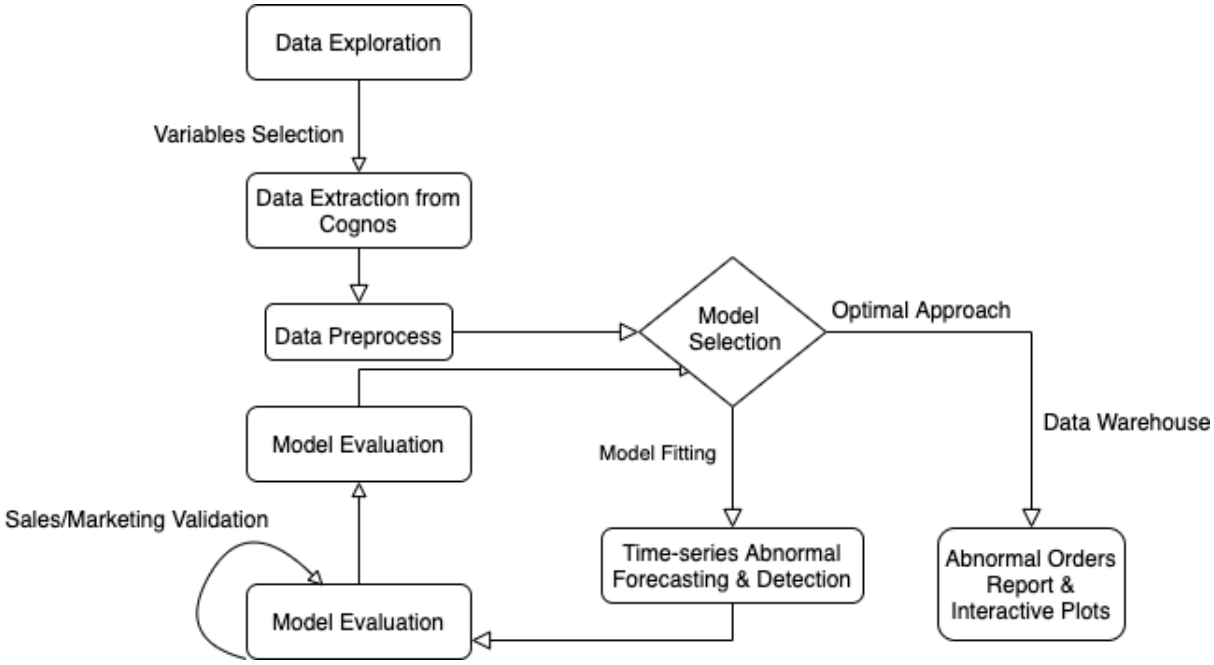


Fig. 3.3.: Flow chart for the design process followed in this thesis.

This ordering is temporal in nature, as such each sample within the sample space can be expressed as the values at a unique point in time and we can define a time differential with respect to two discrete samples in a time-continuous sample space.

In comparison to a continuous function g, where we can determine a derivative by taking the limit $\lim_{\epsilon \to 0} \frac{g(x+\epsilon)-g(x)}{\epsilon}$, real-world data is bound to a sampling rate f giving us a discrete waveform with no definition of the derivative for this sampling function. This means that traditional calculus cannot be applied, rather we must look towards probabilistic modelling methods. Consider that between each time-step in the discrete signal, there is uncertainty over the subsequent possible values, which are resolved at the next measuring interval [15]. As the result of this, the higher the measuring frequency, the lower the uncertainty over the possible values between each sample.

3.2.1 Temporal Stochastic Effects

With time series data comes new challenges, as we not only have to estimate the effects between each of the predictor attributes $x_1, x_2, ..., x_{n-1}, x_n$ with $x_i \in \mathbf{X}$ where X is the vector of attributes in relation to the data, but we also have to consider the change of each of these attributes as a function of

the temporal attribute $\mathbf{T}$ to give observations $y_0, y_1, ..., y_{t-1}, y_t$ where $y_i \in \mathbf{Y}$ and $t \in \mathbf{T}$. This property is called the dynamic causal effect of $\mathbf{X}$, which can be modelled as a sequence of coefficients $\beta_1, \beta_2, ..., \beta_{n-1}\beta_n$ for each attribute $x \in \mathbf{X}$ denoting the effect a change in t has on the realisation of observation $y \in \mathbf{Y}$.

With this property a complicating factor of the model comes into play, as the temporal variable changes, so do the statistical properties of the attributes for non-stationary data [16]. However, as the attributes themselves change in shape and distribution, so do the co-correlated attributes. This sequence of events is what we consider a temporal dynamic model, which is governed by a stochastic process indexed by time and attributes that have causal relationships that can be estimated by inference [17]. Each individual observation within a time series based model is also considered a realization of the stochastic process.

We can describe this type of system with a mixed-effects model that combines deterministic effect attributes with temporal non-determinism, which we can describe as a generalized mixed-effects model as shown in Equation 3.1. In this respect, individual predictors $\mathbf{X}$ and their estimated coefficients $\boldsymbol{\beta}$ are given by the first term, the covariates $\mathbf{Z}$ and their measured interaction coefficient $\boldsymbol{\gamma}$ are given by the second term and the final term gives the random effect on each of the realizations of Y.

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{\gamma} + \boldsymbol{\epsilon} \tag{3.1}$$

Where residuals are defined by the normally distributed vector $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{R})$ with $\mathbf{R} = \boldsymbol{\sigma}^2\,\mathbf{I}$, the static component is given by the parametric vector $\boldsymbol{\beta} \sim N(\boldsymbol{\mu}, \boldsymbol{\sigma})$ and the stochastic component is given by the triangular Cholesky factorization [18, 19] of the variance co-variance matrix $\mathbf{G} = \mathbf{L}\mathbf{D}\mathbf{L}^T$ resulting in the model matrix of Equation 3.2 for the non-deterministic component.

$$\mathbf{G} = \begin{bmatrix} \sigma^2_{\gamma 1} & \sigma^2_{\gamma 1,2} \\ \sigma^2_{\gamma 2,1} & \sigma^2_{\gamma 2} \end{bmatrix} \overset{\text{i.i.d}}{=} \begin{bmatrix} \sigma^2_{\gamma 1} & 0 \\ 0 & \sigma^2_{\gamma 2} \end{bmatrix} \tag{3.2}$$

From the Cholesky factorization we can derive the random effects model parameters distributed by $\boldsymbol{\gamma} \sim N(\mathbf{0}, \mathbf{G})$ to get the Whittaker-Henderson factorization [20] in Equation 3.3 of the model given by Equation 3.1.

$$\begin{bmatrix} \mathbf{X}^T\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}^T\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}^T\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}^T\mathbf{R}^{-1} + \mathbf{G}^{-1} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}^T\mathbf{R}^{-1}\mathbf{Y} \\ \mathbf{Z}^T\mathbf{R}^{-1}\mathbf{Y} \end{bmatrix} \tag{3.3}$$

For the generalized linear mixed-effects model using ordinary-least squares (OLS) parameter estimation this introduces a challenge, similar to cross-sectional regression, several assumptions have to be made over the data which have a big influence on the quality of the final model.

An assumption unique to time series is that the temporal distance between each index within the sample is constant, in other words the sampling rate f must be a constant [21]. An assumption that is harder to satisfy is the normal distribution of the response variable. For this assumption to hold the variance of the response must be approximately constant over all t, which is hardly ever the case in real world data [22]. With the right processing techniques however, we might transform the data to satisfy this assumption to some extent while estimating the effect of the inevitable factor of auto-correlation between temporal sampling for model diagnostics.

3.2.2 Serial and Auto-Correlation

Within a stochastic process we assume that the random errors are uncorrelated, since the model won't take the co-variances into account, as holds with the assumption of independent and identically distributed residuals using the Cholesky factorization in Equation 3.2. By assuming uncorrelated errors and independence of variables we can simplify the propagation of errors into the partial derivatives of variances in our model with respect to each of the attributes over the observations $y_i \in \mathbf{Y}$ and attributes $x_i \in \mathbf{X}$.

$$\sigma_Y^2 = \sum_i^n \left(\frac{\partial \mathbf{Y}}{\partial x_i} \sigma_{xi} \right)^2 = \left(\frac{\partial \mathbf{Y}}{\partial x_1} \sigma_{x1} \right)^2 + \left(\frac{\partial \mathbf{Y}}{\partial x_2} \sigma_{x2} \right)^2 + \ldots + \left(\frac{\partial \mathbf{Y}}{\partial x_n} \sigma_{xn} \right)^2 \tag{3.4}$$

To determine the presence of correlated errors for an attribute over temporal attribute T, we analyse the presence of auto-correlation in the residuals

using the Durbin–Watson test [23]. Consider the residual ϵ in a simple linear regression model defined by Equation 3.5.

$$\mathbf{Y} = \boldsymbol{\alpha}_0 + \boldsymbol{\alpha}_1 \mathbf{X} + \boldsymbol{\epsilon} \tag{3.5}$$

Then for an auto-correlated error, in other words modelling that residual ϵ depends on ϵ_{t-1}, we can define the residual at time t as Equation 3.6.

$$\boldsymbol{\epsilon} = \boldsymbol{\upsilon} + \boldsymbol{\rho}\boldsymbol{\epsilon}_{t-1}, \quad \boldsymbol{\upsilon} \sim N(\mathbf{0}, \boldsymbol{\sigma}^2\mathbf{I}) \tag{3.6}$$

And $\boldsymbol{\rho}$ denotes the correlation matrix, which is given by the Pearson correlation coefficient over elements of $\mathbf{X}$ and $\mathbf{Y}$ in Equation 3.7.

$$\boldsymbol{\rho} = \begin{bmatrix} \rho_{x1,y1}, & \cdots & \rho_{x1,yn} \\ \vdots & \ddots & \vdots \\ \rho_{xn,y1} & \cdots & \rho_{xn,yn} \end{bmatrix}, \quad \rho_{x,y} = \frac{\mathbb{E}[(\hat{x} - \mu_X)(\hat{y} - \mu_Y)]}{\sigma_X \sigma_Y} \tag{3.7}$$

Here ρ denotes the measure of correlation of the residual with itself in regards to the residual at the previous time step for attribute x_i and realisation y_i. For this assumption to hold, we must apply some test to determine if the hypothesis $H_0 : \rho = 0, H_a : \rho \neq 0$ holds to a measure of significance α. This can be tested with a Durbin-Watson statistics given by Equation 3.8.

$$\mathcal{D} = \frac{\sum_{t=2}^{T}(\epsilon_t - \epsilon_{t-1})^2}{\sum_{t=1}^{T} \epsilon_t^2} \tag{3.8}$$

In other words, we divide the sum of squared differences of the residual between time steps by the residual sum of squares. The closer the difference between the residuals is to the value of the baseline residual at t, the less evidence there is for auto-correlation to one degree of time differential. Values $\mathcal{D} > 1$ tend to be positively auto-correlated, while values $\mathcal{D} < 1$ are negatively auto-correlated, and $\mathcal{D} = 1$ means there no auto-correlation between single time-steps [24]. The size of the confidence interval around the critical value of $\mathcal{D}$ depends positively on the amount of regressors within the model and negatively on the sample size.

4 Approaches

In this chapter, we discuss about the related approaches used for our experiments. Since the sales transaction data is unlabelled and discrete (see section 5 in detail), we use the unsupervised clustering methods to discover the hidden pattern of the customer buying behaviors. To detect abnormal sales orders on a time scale, we preform the unsupervised anomaly detection methods for our experiments. For optimizing our anomaly detection model, we use the grid search in automatically selecting the hyperparameters. To analyze the anomalies labelled from our anomaly detection model, we purpose a Gaussian mixture model in the form of a bi-modal distribution.

4.1 Unsupervised Clustering Methods

Since the sales data is unlabelled, distributed over time and hierarchically grouped by product type and customer, we will need to use unsupervised clustering method to identify clusters and patterns within the data. To achieve the goal of detecting abnormal customers, and ultimately abnormal patterns, within the sales data, we utilize two methods which are Principal Component Analysis (PCA) and Density-based Spatial Clustering of Applications with Noise (DBSCAN) described in this section. The result of DBSCAN of applying 2 dimensions of PCA data is shown in section 5.3.

4.1.1 Principal Component Analysis (PCA)

Clustering can help divide aggregated customer data with similar sales patterns into hierarchical groups, which can then be mapped in proximity to find outliers. To deal with the high dimensionality of the data, PCA is used to generate representation mappings for the original data into few numbers of principal components [25].

Given a data matrix $\mathbf{X}$ with m variables and n objects, the covariance matrix for $\mathbf{X}$ is shown in Equation 4.1

$$Cov(\mathbf{X}) = \begin{bmatrix} Cov(x_1, x_1) & ... & Cov(x_1, x_i) & ... & Cov(x_1, x_m) \\ ... & ... & ... & ... & ... \\ Cov(x_j, x_1) & ... & Cov(x_j, x_i) & ... & Cov(x_j, x_m) \\ ... & ... & ... & ... & ... \\ Cov(x_m, x_1) & ... & Cov(x_m, x_i) & ... & Cov(x_m, x_m) \end{bmatrix} \tag{4.1}$$

where x_i represents the i-th variable in data matrix with $i \in \{1, 2, ..., m\}$. The covariance between two variables x and y is $Cov(x, y) = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})/(n-1)$. For PCA the weights for the representations on the covariance of the data matrix $\mathbf{X}$ are estimated by finding matrix $\mathbf{v}$ that maximizes variance over the covariance matrix $Cov(\mathbf{X})$ as in Equation 4.2 where λ represents the scale of the variance. For m variables, there are m vectors of weights so that $\mathbf{v}$ represents the weights for the new variables (i.e. principle components) [25].

$$(Cov(\mathbf{X}) - \lambda_i \mathbf{I})\mathbf{v_i} = \mathbf{0} \tag{4.2}$$

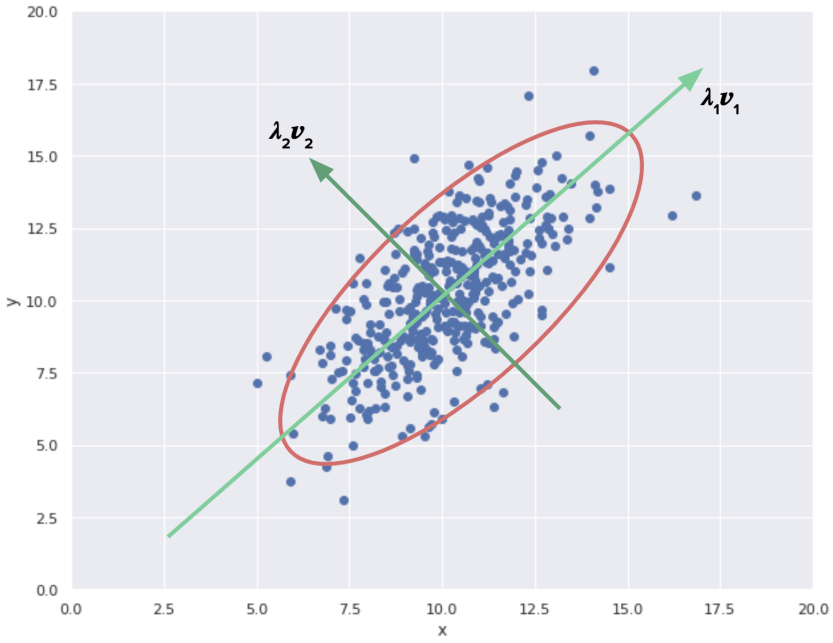


Fig. 4.1.: Principle Component Example in 2 Dimensional Space Illustration

Each principle component is uncorrelated with each other and ordered by the explained variance defined by λ [26]. Figure 4.1 gives an example in visualizing the principle components in a 2 dimensional space on the original example dataset with 2 variables. In the plot, it is clear that the example data

has a positive correlation. After using the PCA, the first principle component in the light green arrow shows the direction of the data represented by the weight vector $\mathbf{v_1}$ with the highest variance represented as the vector length scaled by λ_1. The first principle component explains the most of variance and the second principle component shown in the darker green arrow is uncorrelated with the first component since $\mathbf{v_2}$ is perpendicular with $\mathbf{v_1}$. The length of second principle component vector λ_2 is smaller than that of the firs component which indicates that the second principle component explains less variance than the first component. In our unsupervised clustering experiment, we use PCA to capture most of the variance among our four numerical variables and reduce the dimensionality to 2 principle components.

4.1.2 DBCAN

From the PCA, we use Density-based Spatial Clustering of Applications with Noise (DBSCAN) to initially identify customer-based outliers. DBSCAN provides a more reliable clustering method in light of within-cluster outliers [4]. There two parameters used in DBSCAN, *minPts* represents the minimum number of points (i.e. threshold) for the density of the neighborhood including the current point itself. Another parameter is the radius ϵ which means the distance in measuring the scale of the neighborhood. Higher radius means the neighborhood scale is bigger which can include more points.

In DSBCAN model, the neighbors for the point within its the radius with at least minPts in the neighborhood is called *core point* which belongs to the same cluster [27]. The point that stays within the radius of the core point and does not reach the minPts neighbors of itself is represented as the *border point* and it has the same cluster as the core point since the border point is within the neighborhood of the core point (i.e. density connected), in addition, the core points are density reachable with each other [28]. The point that is not density reachable with the core point, in order words, the point does not connect to the core point within its neighborhood is considered as *noise* [27]. In our experiment, we use DBSCAN to detect the noises which are the abnormal customers to discover the customer buying behavior pattern further.

The illustration of DBSCAN cluster model is shown in Figure 4.2 generated by Schubert et al. [28] with *minPts* $= 4$ and radius ϵ is the length of the green

arrow. In the plot, point A and the other red points are core points since each of them has the neighbors including themselves great than the threshold. Point B and C are the border points since they are density connected and reachable with the core points but do not reach the minPts requirement for their neighborhoods. Point N is neither density connected nor reachable, thus, it is the noise/outlier. The points A, B and C are within the same cluster. By clustering based on densities, outliers within a cluster are detected by density even in non-linearly separable clusters [28].

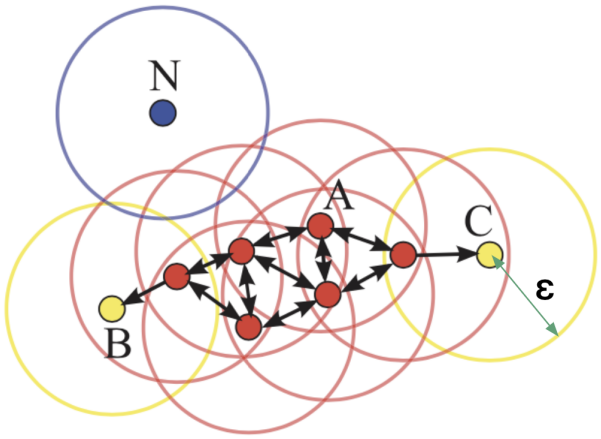


Fig. 4.2.: DBSCAN Cluster Model Illustration [28]

The pseduecode of DBSCAN algorithm shown in algorithm 1 is originated from [28]. The DBSCAN algorithm iterates each point p in the points set P and detects if the current point is core point or not based on the number of connected neighbors N. If p is the core point, then it will be assigned into a cluster, otherwise, it will be labeled as an outlier. Since some of the outliers are connected to the core points but do not reach the minPts neighbors, DBSCAN solves this issue by expanding the neighborhood based on the detected core points and then reaching the border points while relabeling them to the cluster.

In the algorithm 1, the function RangeQuery is called two times. In the first time, DBSCAN searches for the neighbors for the unlabelled point in the beginning and then it expends the neighborhood in searching for the neighbors of the border points in the second time. The pseduecode of RangeQuery function given in algorithm 2 compares the Euclidean distance between the selected point and other points q in P with the given radius ϵ iteratively and selects the reachable neighbors for the point p.

Algorithm 1: DBSCAN Alogrithm Pseduecode [28]

input : Point set P
input : Radius ϵ
input : Minimum points *minPts*
input : Distance function *dist*
output : Point labels label, default *null*

foreach *point* $p \in$ *points* P **do**
  if label$(p) \neq null$ **then continue** `// skip labelled points`
  Neighbors $N \leftarrow$ `RangeQuery`$(P, dist, p, \epsilon)$
  if $|N| < minPts$ **then** `// label non-core points`
    label$(p) \leftarrow$ Outlier
    continue
  $c \leftarrow$ new cluster label
  label$(p) \leftarrow c$
  Subset $S \leftarrow N \setminus \{p\}$
  foreach *point* $q \in$ *points* S **do** `// Relabel non-core points`
    if label$(q) =$ *Outlier* **then** label$(q) \leftarrow c$
    if label$(q) \neq null$ **then continue**
    Neighbors $N \leftarrow$ `RangeQuery`$(P, dist, q, \epsilon)$
    label$(q) \leftarrow c$
    if $|N| < minPts$ **then continue**
    $S \leftarrow$ `Union(S,N)`
  end
end

Algorithm 2: Pseduecode of RangeQuery Function for DBSCAN

Function `RangeQuery`$(P, dist, q, \epsilon)$**:**
  foreach *point* $q \in$ *points* P **do**
    $dist \leftarrow \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$ `// Euclidean dist with n-space`
    if $dist < \epsilon$ **then**
      Neighbors $N \leftarrow$ add q
    end
  end

4.2 Unsupervised Anomaly Detection

In this section, we perform the unsupervised anomaly detection for our experiments in detecting abnormal orders on a time scale. We use auto-regression as our baseline model and primarily look at an time-series forecasting algorithm provided by Facebook called `Prophet` [29]. Prophet is a method that can be used to model time-series data with strong seasonality and handles outliers and changepoints well from an automated optimization procedure.

4.2.1 Auto-Regression

Although we have described a method to describe hierarchical anomalies in customer behaviour with clustering, determining anomalies on the level of individual transactions requires a more intricate approach that takes into account the auto-regression, stationarity and causality [17].

An important factor relating to sequential data we have discussed so far are serial- and auto-correlation. This describes that observations at t given by y_t may be dependant on the set of previous observations $y_0, ..., y_{t-1}$, where δt is constant for some periodic sampling frequency f.

To model this sequential relationship between observations, we can specify a model that includes regressor elements on previously observed time steps [30]. Such a linear model specification using all points 0 to $t-1$ is given in Equation 4.3.

$$y_t = \beta_0 + \sum_{i=1}^{t-1} \beta_i y_{t-i} + \epsilon_t \tag{4.3}$$

Within this model, the auto-correlation is modelled as $\rho(y_{t-1}, y_t)$ estimated as coefficient β_{t-1}. In this case we are considering the auto-correlation between a single time step, which is considered the serial correlation with lag $k = 1$, which forms the AR(1) model $y_t = \beta_0 + \beta_1 y_{t-1}$. Previously we have defined the general auto-regressive model in Equation 4.3, where the model is comprised of the auto-regressive components of order $k = 1$ to $k = t - 1$, given by $\sum_{k=1}^{t-1} \text{AR}(k)$, plus an intercept and error term to estimate y_t.

Such an auto-regressive model is prone to overfitting, because we estimate n parameters β for n samples, which does not generalize well outside of the training data. For that purpose the lag component is modelled from a moving window of size l. One approach may be to take the last l data points and discard older data points when considering estimation of y_t, which encourages exploitation of current trends [31]. While for example, an AR(7) model would be appropriate to model the exact difference in observation as compared to the previous week, which considers $\rho(y_{t-7}, y_t)$, and sample points y_{t-6} to y_{t-1} as random variables in a random process that link the two observations probabilistically.

4.2.2 Prophet Model

Prophet is a time-series automated forecasting model [13] developed by Facebook AI Research, which can be used in detecting anomalies on a timescale. This can be performed because Prophet can model the sales patterns overtime and provide the expected value with an associated confidence interval. For data that are out of the confidence interval, it can be considered as an outlier since it deviates from the expected value boundaries. Prophet contains three main components which are trend, seasonality and holidays [29]. The general piece-wise formula can be defined below

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t \tag{4.4}$$

where g(t) represents trend, s(t) is seasonality, h(t) means holidays effect and ϵ_t means the error.

Trend Component

The trend model contains two implementations, the first is a piecewise growth model that is built for nonlinear growth such as the population growth [29]. The generalized saturating growth model given in Equation 4.6 is developed from the baseline logistic growth model with the formula:

$$f(x) = \frac{L}{1 + e^{-k(t-t_0)}} \tag{4.5}$$

where L is the maximum value of the curve, k is the growth rate and t_0 is the initial time. In Equation 4.6, C is the carrying capacity, m is the offset parameter for the time t. According to Taylor and Letham [29], there are two differences between the piecewise logistic model and the standard model. The first one is that the carrying capacity C is a variable C(t) that can be changed based on time t. The second is that the growth rate k is changeable. The growth rate k is adjusted to $k + \boldsymbol{a}(t)^\intercal \boldsymbol{\delta}$ at time t. $\boldsymbol{\delta} \in 0, 1^S$ represents the rate adjustments vector and $\boldsymbol{a}(t)$ indicates the vector that helps sum up the rate adjustments to the time t with the number of S changepoints [32] at time t.

$$g(x) = \frac{C}{1 + exp(-(k + \boldsymbol{a}(t)^\intercal \boldsymbol{\delta})(t - (m+)\boldsymbol{a}(t)^\intercal \boldsymbol{\gamma}} \tag{4.6}$$

Another implementation for trend model is a piecewise linear model [33] which is the trend model we use for our experiments. The reason is that

growth model focuses on the pattern similar with the logistical growth, however, our transaction data does not show such growing pattern based on our previous data analysis. The linear model formula shown in Equation 4.7 where k, $\boldsymbol{\delta}$, and m are the growth rate, rate adjustments and offset parameter respectively. γ_j represents $-s_j\delta_j$ to make the function continuous where s_j is the changepoint time with $j \in 1, S$ and δ_j is the rate changes at time s_j.

$$g(x) = (k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})t + (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma}) \tag{4.7}$$

When there is no specifications for changepoint dates are assigned, the trend model first selects a large number of changepoints (i.e. default is 25) based on the history and then apply the sparse prior for the rate change $\boldsymbol{\delta}$ for the regularization purpose [29]. Sparse prior is $\delta_j \sim Laplace(0, \tau)$ for each changepoint $j \in 1, S$, in order words, sparse prior has the Laplace distribution shown in Figure 4.3 which is controlled by τ (i.e. changepoint prior scale) that can be tuned for the model flexibility. As the changepoint prior scale goes to 0, it does not have an impact on the growth rate which turns the model to standard logistical and linear model [29].

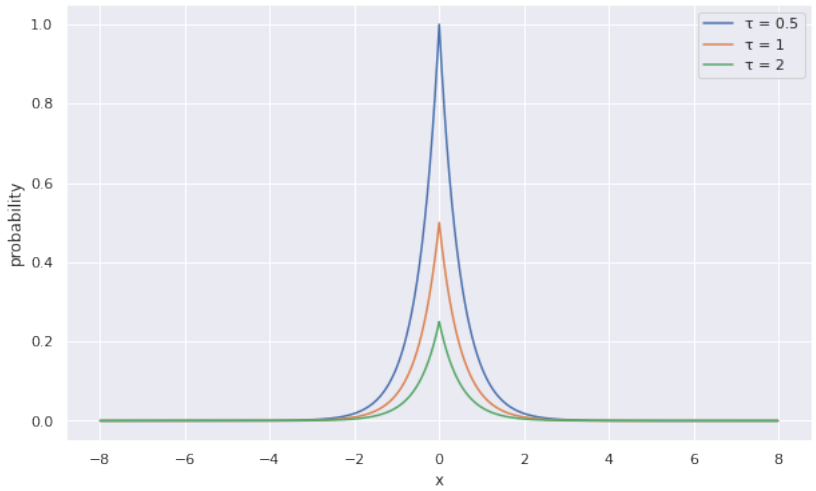


Fig. 4.3.: Laplace Distribution with changing τ

In forecasting, the trend model provides future rate changes (i.e. uncertainty) in order to estimate the future pattern based on the historical patterns. The assumption made by Taylor and Letham [29] is that the forecast uncertainty has the same change rates in the average frequencies and extents as the historical ones.

$$\forall j > T, \begin{cases} \delta_j = 0 & \text{w.p. } \frac{T-S}{T} \\ \delta_j \sim \text{Laplace}(0, \lambda) & \text{w.p. } \frac{S}{T} \end{cases} \tag{4.8}$$

In the uncertainty formula (see Equation 4.8) where $\lambda = \frac{1}{S}\sum_{j=1}^{S}|\delta_j|$ in replacing the historical change prior scale τ described in Equation 4.7 with the maximum likelihood estimate to make sure the randomly generated future changepoints have the average frequencies based on the historical pattern [29]. Figure 4.4 shows an example of changes in uncertainty during the one year (i.e. 365 days) forecast, it is clear that the prediction patter has the similar rate changes compared to the historical changing patterns from 2020 January to 2021 January.

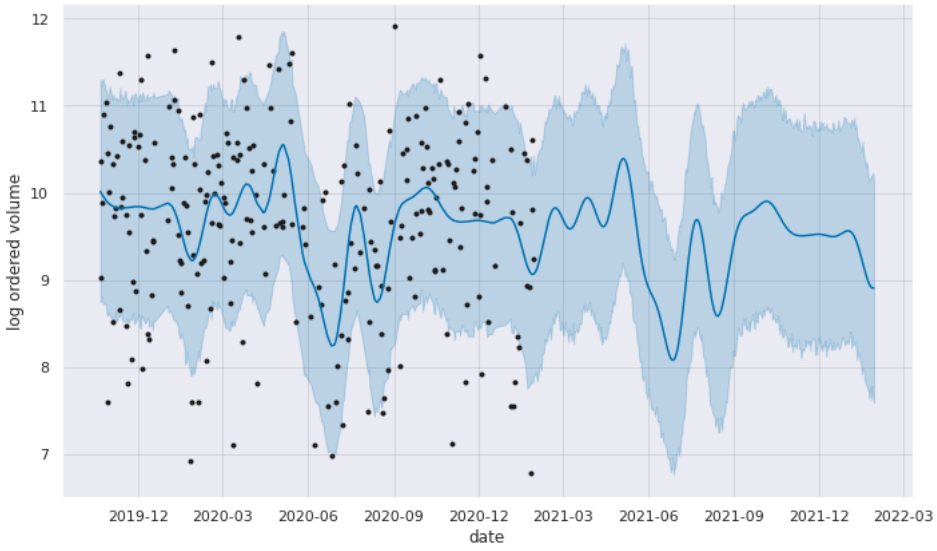


Fig. 4.4.: Trend uncertainty example with one year forecast with log ordered volume as y-axis.

The Laplace distribution parametric over $\mu = 0, \beta = \lambda$ provides us with the distribution over random variable $\delta_j \sim \text{Laplace}(0, \lambda)$, which we can evaluate to the Laplacian distribution as in Equation 4.9.

$$\text{Laplace}(x \in X \mid \mu = 0, \beta = \lambda) = \frac{1}{2\lambda}\exp\left(-\frac{|x|}{\lambda}\right) \tag{4.9}$$

Seasonal Component

The seasonality component uses the Fourier series model to provide the periodic seasonal effect. The seasonal component is deterministic since the result is determined by the model parametric inputs, with the same set of parameters, the output will remain the same. A standard Fourier series function (see Equation 4.10) is a linear periodic combination of sines and cosines where P represents the period that shows the same pattern along

the time and k is the specified number of waves appeared during the time period P. A_0, A_k and B_k are the Fourier coefficients, $A_0 = \frac{1}{\pi}\int_{\pi}^{-\pi} f(x)\,dx$ represents the average value within the time range $(-\pi, \pi)$, A_k function is $\frac{1}{\pi}\int_{\pi}^{-\pi} f(x)\cos kx\,dx$ and B_k function is $\frac{1}{\pi}\int_{\pi}^{-\pi} f(x)\sin kx\,dx$.

$$f(x) = \frac{A_0}{2} + \sum_{k=1}^{\infty}(A_k \cos\frac{2\pi kx}{P} + B_k \sin\frac{2\pi kx}{P}) \tag{4.10}$$

In Figure 4.5, it shows the changes of Fourier series with $k = \{3, 10\}$ in weekly $P = 7$ and yearly $P = 365.25$ periods. When $k = 3$ and the period P is 7, the function is shown in Equation 4.11 where the coefficients $A_0 = 0.49, A_1 = 1.52, A_2 = -2.48, A_3 = -4.26, B_1 = 0.63, B_2 = 0.01, B_3 = 0.19$. From the figure, there are less curves compared to the curve with the same period but higher $k = 10$. However, when we compare $k = 3, P = 7$ curve with $k = 3$ and period is 365.25 (i.e. yearly) curve, the yearly curve has less waves during the period $(-\pi, \pi)$. Therefore, less period value and higher k number can cause the Fourier series becomes more volatile.

$$\begin{aligned} f(x) = \frac{A_0}{2} + A_1 \cos\frac{2\pi x}{7} \quad &+A_2 \cos\frac{2\pi 2x}{7} + A_3 \cos\frac{2\pi 3x}{7} \\ + B_1 \sin\frac{2\pi x}{7} \quad &+B_2 \sin\frac{2\pi 2x}{7} + B_3 \sin\frac{2\pi 3x}{7} \end{aligned} \tag{4.11}$$

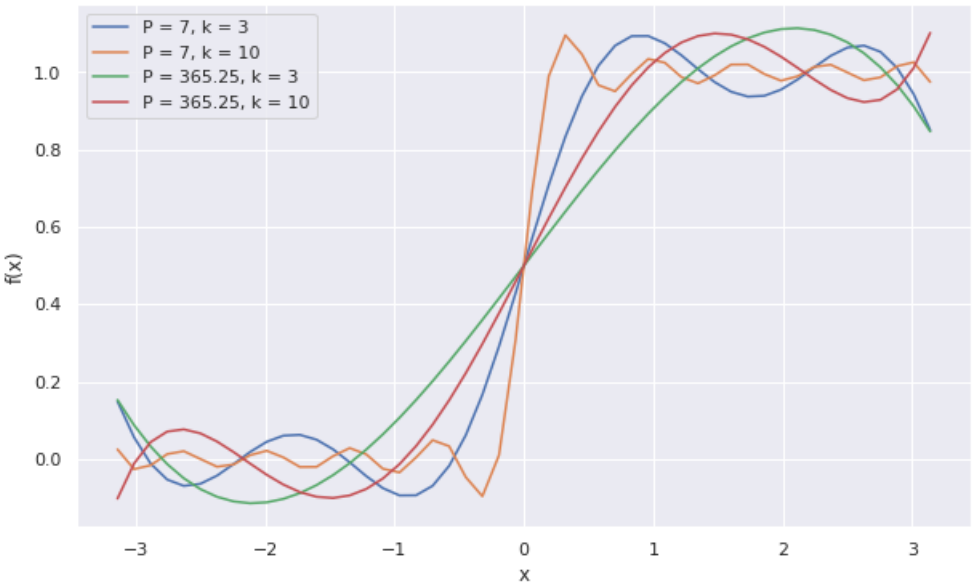


Fig. 4.5.: Fourier series with $k = \{3, 10\}$ in weekly and yearly periods.

Taylor and Letham [29] purpose to add smoothing in the seasonal model by making the coefficients to have the normal distribution with mean equals to 0, they also find that $k = \{3, 10\}$ suit most of the weekly and yearly seasonality without providing over-fitting results.

Table 4.1.: A List of Events in United Kingdom

Date	Event
1 January 2019	New Year's Day
1 January 2020	New Year's Day
1 January 2021	New Year's Day
31 December 2020	New Year Holiday [Scotland]
17 March 2021	St. Patrick's Day [Northern Ireland]
17 March 2021	St. Patrick's Day [Northern Ireland] (Observed)
12 July 2021	Battle of the Boyne [Northern Ireland]
2 August 2021	Summer Bank Holiday [Scotland]
30 November 2021	St. Andrew's Day [Scotland]
27 December 2021	Christmas Day
26 December 2021	Boxing Day
1 January 2021	New Year Holiday [Scotland] (Observed)

Holidays / Events Component

The event model is deterministic since the output is the same for the given holiday inputs unless we change them and non-differentiable because each holidays or events represent one or multiple days and they are independent with each other assumed by Taylor and Letham [29]. An instance of a list of events in United Kingdom is shown in Table 4.1, it is clear that event such as "New Year's Day" happens every year and in Scotland the New Year Holiday starts on December 31st.

$$h(t) = [\mathbf{1}(t \in D_1), ..., \mathbf{1}(t \in D_L)]\boldsymbol{\kappa} \tag{4.12}$$

According to Taylor and Letham [29], the event model formula given in Equation 4.12 shows the holiday effects in forecasting where D_L is a set of event dates for holiday L, $\boldsymbol{\kappa}$ is a smooth prior that has a normal distribution with mean equals to 0 and it is applied to the previous event matrix $[\mathbf{1}(t \in D_1), ..., \mathbf{1}(t \in D_L)]$.

4.3 Automated Optimization Technique

To automate features and configuration, we use the method discussed in [34] on how to automate features and configurations in order to find the best

model. We use an automated machine learning procedure to optimize the hyperparameters based on a grid search based approach.

Grid search as a optimization technique helps in finding the optimal solution for the model by tinning the specific parameter values of a model. In selecting the optimal hyperparameters of the model, we perform the grid search on the daily, weekly and monthly transactions data. We use cross-validation, a validation method, for our grid search in estimating the overall general performance of the model and selecting the optimal pair with the best cross-validation performance while prevent overfitting [35]. The idea of the cross-validation is that we use the a given period of historical data in predicting a given time period value for multiple times with a specific cutoff period in between.

During optimization, we perform the grid search on a three-dimensional space with three hyperparameters X, Y, Z which represent changepoint prior scale, seasonality prior scale and holiday prior scale respectively. We consider 4 values for each hyperparameter and there are 64 combinations in total. The sets of values for each hyperparameter are $X \in \{0.01, 0.05, 0.1, 0.5\}$, $Y \in \{0.01, 0.1, 1.0, 10.0\}$, and $Z \in \{0.01, 0.1, 1.0, 10.0\}$. Through grid search, we compare the performance of each pair and select the best pair by using cross-validation evaluation scores.

4.4 Anomaly Analysis

Classifying outlier observations when no inherent class labels are present requires us to model the distribution of the process generating the data. This allows for estimation of the sales observation across customers, and makes it possible to model observations that fall outside of the expected level of uncertainty over the previous observations. After the anomaly detection experiment, each sales order is classified as one of the labels which are *normal*, *negative anomaly*, and *positive anomaly*. Negative anomaly means an order that has extreme low ordered volume and positive anomaly is an abnormal order with high ordered volume. Under the assumption of normality, a Gaussian mixture model can model the anomalies as a bimodal distribution (i.e. a distribution with two combined normal distributions), which are situated beyond the confidence interval above and below the expected value.

Since we have three labels for the sale orders, assume that we have a mixture distribution of three Gaussian components, with components for under-performing sales (i.e. negative anomaly), normal sales and over-performing sales (i.e. positive anomaly). When we observe a series of points $x_1, ..., x_n$ from random variables X_k, we assume that each x_i has been sampled from one of the $k = 3$ distributions, which associates x_i with a classification label $z_i \in Z_k$. In our anomaly analysis experiment, the set of labels is $Z = \{-1, 0, 1\}$ and -1 is under-performing 0 is normal, and 1 means over-performing.

Since z_i is not observed, but has to be inferred, they are latent variables within the mixture model. Given a proportion π_k associated to each of the k distributions, where $\sum^k \pi_i = 1$, the marginal probability model given for a random variable generating X_k conditional to a label Z_k can be shown in Equation 4.13.

$$\sum_{j=1}^{k} P(X_i = x \mid Z_i = j)P(Z_i = j) = \sum_{j=1}^{k} P(X_i = x \mid Z_i = j)\pi_j \tag{4.13}$$

For a continuous random variable we can then give the probability density and mass functions as Equation 4.14.

$$f(x) = \sum_{j=1}^{k} \pi_j f_X(x \mid Z_j) \quad p(x) = \sum_{j=1}^{k} \pi_j p(x \mid Z_j) \tag{4.14}$$

Under the assumption of a given constant k and each distribution being Gaussian or by making an assumption over the true labels $z_i \in Z$, the weights π_k and the conditional probabilities can be estimated with Maximum Likelihood Estimate (MLE) in order to get the distribution that maximizes the probability of data [36]. Therefore, we are able to use such Gaussian mixture model in modeling the anomalies.

5 Data Analysis

According to Avery Dennison [1], there are two types of sales data which are currently available. The first one is called "sales invoice" which contains an aggregated and brief information about the sales orders happened daily. The second one, which is the data we use for this project, is called "sales orders", it contains abundant daily orders with sufficient information. After consulting the sales department, we decide to continue working on our experiments based on sales orders data. For reasons of confidentiality, the data is only for company's internal usage and disclosure of the customers names is prohibited. The transaction dataset extracted from IBM Cognos system [37] is the subset from the Avery Dennison EU data warehouse. The time range for our dataset covers the sales orders dates starting from October 2019 to January 2021.

5.1 Sales Order Variables

Since there are a large number of variables that can be used and analyzed in the sales orders data, it is important for us to consider what variables are fundamental and important for both sales and marketing departments. The unrelated variables can have negative influence on our anomaly detection results [38] and also produce noise for our transaction data. In addition, the outlier detection result will be used by the sales department, so it is important for them to decide which variables are necessary for them during the daily workflow. Table 5.1 shows our selected numerical and categorical variables without ordering in the Design Cycle stage (see Section 2.1 Design Science).

From Table 5.1a, the column "Unique Labels" shows the number of labels this categorical variable has. It is clear that in our dataset, we have 20 categorical variables, and variables such as `Sales Order Number`, `Product Code` have a large number of labels. The customer contains customer from inner-companies (i.e. the sub-branch company that orders the products within the company) which will be filtered in the preprocessing stage. We can find that within 495 ordering dates, everyday there is a large number of orders from different customers with various products. That's why the

Table 5.1.: Sales Order Data Variables Name List

(a) Categorical list

No.	Categorical Variable	Unique Labels
1	Sales Order Number	469857
2	Customer Name	2992
3	Country Name	67
4	Region Name	6
5	Product Code	5995
6	Product Name	5627
7	Product Category 0	6
8	Product Category 1	34
9	Product Category 2	89
10	Product Category 3	44
11	Standard Finance Reporting Flag	1
12	Division	1
13	Customer Segmentation Code	4
14	Customer Segmentation Name	4
15	Sales Order Date	495
16	Bulk Specialty	6
17	Adhesive Technology	9
18	Product Line Team	7
19	Liner Group	5
20	Face Material	1472

(b) Numerical list

No.	Numerical Variable	Value Range
1	Average Sales Price in Euro	(0,2870)
2	Quantity Ordered	(-130219,3468000)
3	Value Ordered	(-1418136, 1446156)
4	Material Cost in Euro	(0, 891960)

anomaly detection for each customer is needed since there are over 5000 products. In our experiments, we mainly focus on `Product Category 2` which is an important categorical variable for sales people. From Table 5.1a, it has 89 labels which is larger than other `Product Category` variables but is not as large as the number labels that product code has.

The reason for choosing `Product Category 2` is because in sales orders `Product Code` can be changed even it represents the same product, however, by using `Product Category 2`, we are not only able to see the product attributes but also keep track of the current product.

From Table 5.1b, the column "Value Range" represents the range of each numerical variable. The `Average Sales Price` (ASP) shows the average product price for each order. `Quantity Ordered` and `Value Ordered` show the the quantity and the value in euro for each individual sales order respectively. `Material Cost` is the total cost for each sales order. If one sales order

Table 5.2.: Numerical Variables Distribution Summary

	ORDERED_VALUE_EUR	QTY_ORDERED_SQM	MATERIAL_COST_EUR	ASP_EUR
count	601010.00	601005.00	601010.00	599704.00
mean	3232.70	7100.69	1695.10	0.65
std	7261.07	16920.25	3934.88	4.63
min	-1418135.84	-130218.52	0.00	0.00
25%	766.00	1340.00	354.07	0.36
50%	1458.80	2640.00	682.60	0.46
75%	3087.50	6120.00	1529.07	0.67
max	1446156.00	3468000.00	891960.00	2870.00

has a negative ordered value or quantity, it means this order is a balance order which is commonly used in sales area [39]. For our experiment, we don't expect negative orders shown in our testing stage, therefore, we sum up both the positive order and the negative order for the same product on the same day in order to eliminate the negative influence of the balance order.

Figures 5.1, 5.2, 5.3 and 5.4 show the frequency distributions of four numerical variables listed in Table 5.1b. Table 5.2 provides the summary statistics of these quantitative variables.

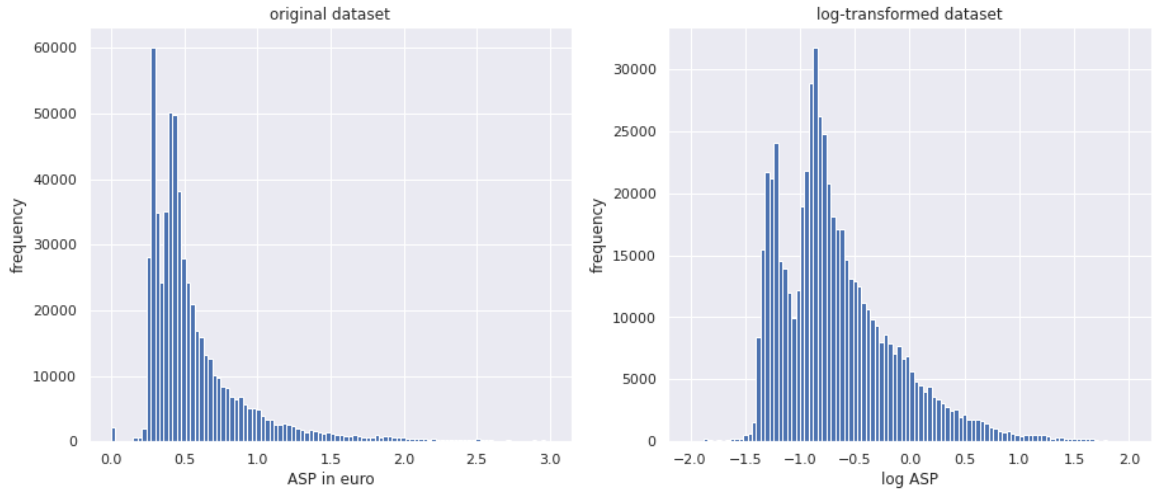


Fig. 5.1.: Total Average Sales Price Distribution

Figure 5.1 shows the ASP original dataset distribution in the range between 0 and 3 euros and the log-transformed ASP distribution. From the plot and the Table 5.2, we can find that the distribution is skewed to the right side with standard deviation 4.63 and 75% orders concentrate between 0 to 0.67 euro. This indicates that many products have low ASP per square meter. Moreover, there are 25% orders between 0 and 0.36 euro which means less orders have been made under 0.36 euro compared to the large amount of

orders with ASP greater than 0.36 euro. After the log transformation, the ASP distribution is less skewed and has two peaks.

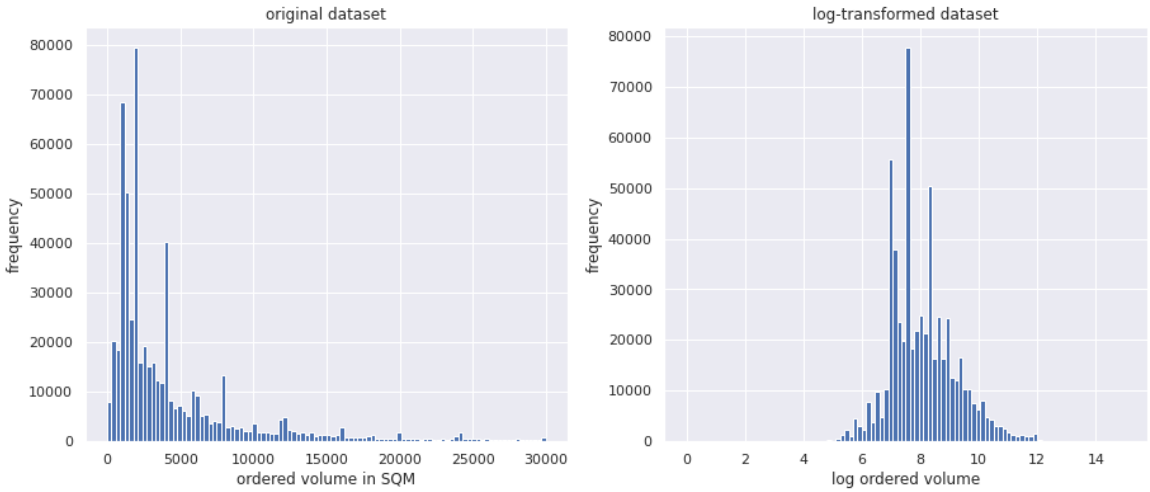


Fig. 5.2.: Total Ordered Volume Distribution

From Figure 5.2 and Table 5.2, we can tell the ordered values in the transaction data are highly spread out with standard deviation 7261.08. A high frequency of orders are concentrated around 2500 with lower frequencies after 5000. It is interesting to find there is a small peak when the volume is around 4000. After transformation, the log ordered volume is close to normal distribution except some high peaks. Since it is close to normal distribution, we decide to use the log ordered volume for our experiments to detect anomalies.

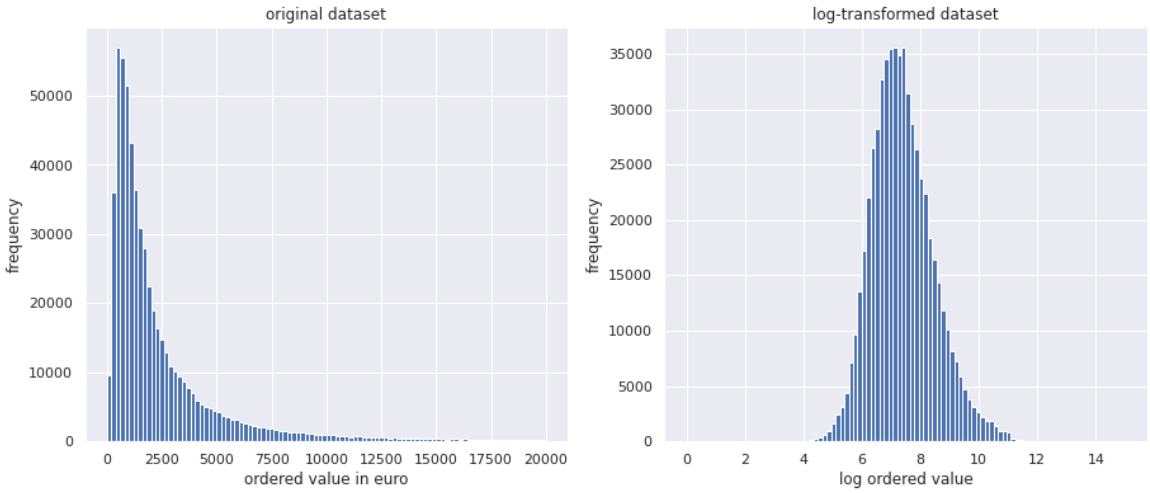


Fig. 5.3.: Total Order Value Distribution

Figure 5.3 shows the frequency distribution of the total ordered value in the range between 0 and 20000 before log transformation and the log-

transformed distribution. From the left plot, it is highly skewed to the right wit high variation which is 16920.25 from Table 5.2 and it indicates most of orders are concentrated between 0 and 2500 euros. Since the mean (7100.69 euro) is a lot higher than the ordered value at 50% orders (2640.00 euro), it means there are some orders with high ordered value which causes the mean greater than the median ordered value. After transformation, it is clear that the log ordered value has a near-normal distribution with mean around 7.

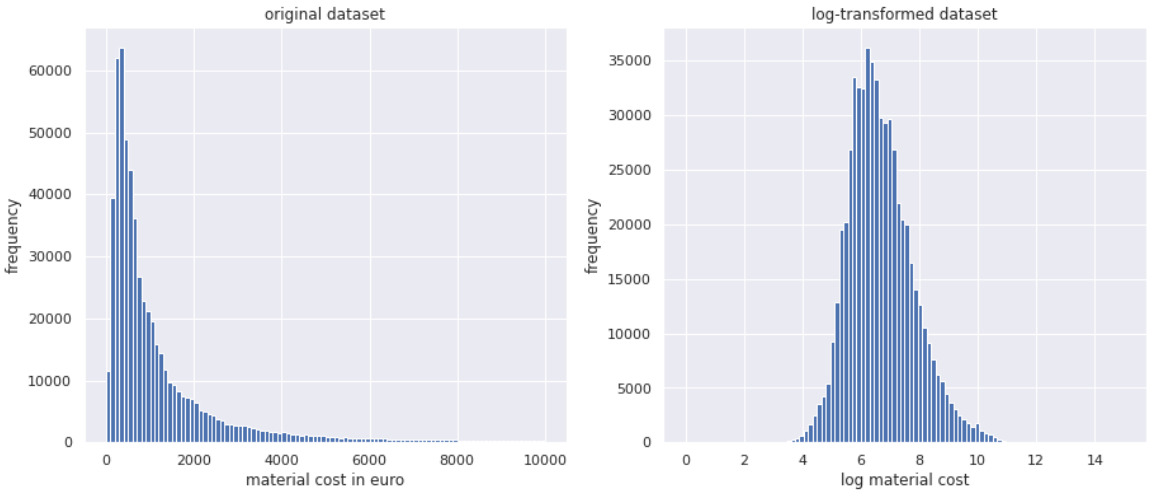


Fig. 5.4.: Total Material Cost Distribution

From Figure 5.4, in the left plot, the distribution shape is skewed to the right side with a high standard deviation 3934.88 from Table 5.2. Most of orders material costs are concentrated between 300 and 1500 euros. The mean is greater then the median (682 euros) indicates there are some orders that have a high material costs. For log-transformed material cost, the distribution in the right plot is approximately normal [40].

5.2 Preprocessing

In this section, we do the preprocessing for our sales order dataset, since Cognos can only store the recent years data, we select the sales data from 2019 to 2021. The entire dataset before preprocessing has millions of daily sales orders, and almost 3000 customers in total. However, the sales data includes Avery intercompany orders and also some orders with missing values, therefore, it's necessary to clean the data before continuing with our experiments [41].

Table 5.3.: Sales Data Filtering Condition

Variable Name	Condition
REGION_CODE	except 5
CUSTOMER_SEGMENTATION_CODE	except STD
PRODUCT_CATEGORY_0	except Tapes
STANDARD_FINANCE_REPORTING_FLAG	except flag tagged as N
DIVISION_NAME	except GDE division
DIVISION_NAME	except PPD division
DIVISION_NAME	except orders with empty division
CUSTOMER_SEGMENTATION_NAME	except Internal Company orders

First, we select customers that meet our criteria, in other words, customers that do not belong to Avery Dennison (i.e. not internal companies). Specifically, we filter the original 2019-2021 dataset by using the conditions shown in Table 5.3.

Second, we clean the sales dataset through the following three steps:

1. Dealing with missing value in various variables such as Bulk specialty, Ordered value, etc. From the sales department, we know that the orders with missing values can be system errors, the wrong listed order, and orders that are incomplete. In this step, we removing the sales orders with missing numerical or categorical variables shown in Table 5.1.

2. Dealing with negative ordered value and ordered quantity. The negative order after merging is due to the deferred refund related to customer behaviors such as product exchanging, product refunding. Here we merge the negative orders with the same orders on the same day to do the order corrections.

3. Encoding `Sales Order Date` to the numerical timestamp. Here we use Python `datetime` package to transform the sales date with the format `year-month-day` to a numerical timestamp variable.

Third, because the sales dataset contains the daily orders with the same products and customer but in various quantities, we aggregate such same orders on the daily basis, therefore, there's only one sales order from the same customer for the same product in one day. By doing this step, we reduce the number of the original millions of daily orders to around 0.6 million of orders in total.

5.3 Sales Data Exploration

After finishing the preprocessing, from what we've discussed in Design Science, we explore the dataset on both categorical and numerical variables to study what variables are we interested in detecting anomalies and what variables are less important from both the sales representatives perspective and data exploration perspective. For numerical variables in Table 5.1b, we check the correlation among these variables [42] while adding another numerical variable timestamp (i.e. date) which we generate from previous preprocessing stage. From Figure 5.5, it's obvious that the sales volume, value, and cost are high correlated since the number are around 0.9. The correlations between ASP and other variables are close to zero, this indicates that there's no strong correlation for ASP with cost, value, and date. The correlations between date and volume, date and cost, date and ASP are close to zero which show no meaningful relationship. However, the correlation between date and value is 0.24, this means there's a moderate but not strong positive relationship [43]. The moderate relationship between date and value indicates the price increase.

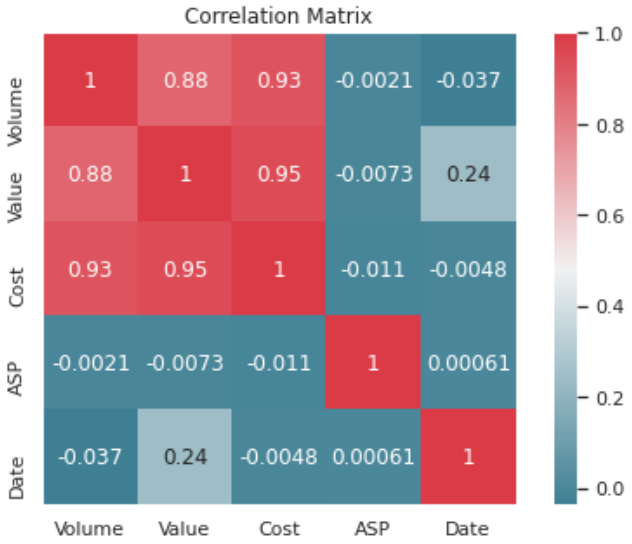


Fig. 5.5.: Correlation Matrix for numerical variables and date

As we discussed previously, we consider `Product Category 2` as an important categorical variable for our anomaly detection model. Therefore, we plot the category 2 labels to see the buying pattern for each label. After reprocessing, we have 81 unique labels for category 2 products, Figure 5.6

shows the distribution of these labels. Each dot represents one label, the left plot shows the overall category 2 products distribution based on the total ordered quantity and the ASP per product, the y-axis on the right plot indicates the total volume less than 10 millions. In the plot, each category 2 label is transformed into an ID number (i.e. "PP85" is represented as `Cat2_ID=5`). From Figure 5.6, we can find that most of the category 2 products are concentrated under 10 million ordered quantity with ASP less than 3 euros. There are some products which have a large amount of sales volume but with low ASP, these products are usually the popular or common products that most customers tend to buy. However, there are some products with high ASP close to 7 euros but with low ordered quantity. These are the special and unusual products for few customers.

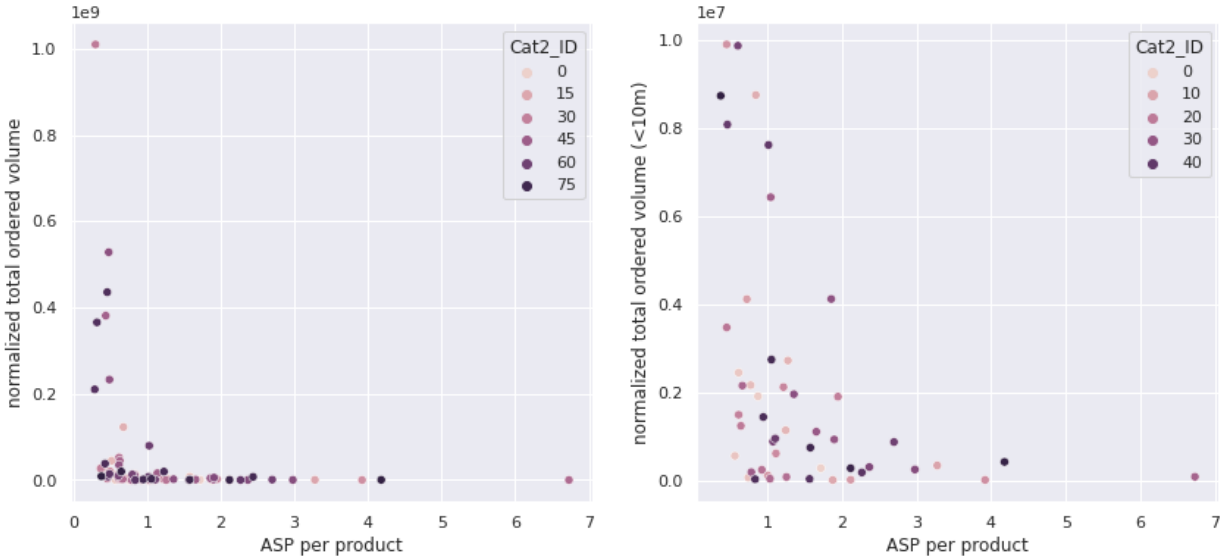


Fig. 5.6.: Category 2 products overview with ASP and ordered volume

From Table 5.1a, we consider the the abnormal behaviors among customers from various countries. Therefore, it is necessary to have a visualization for the distributions of the total number of customers, the total ordered quantity (i.e. volume), the total ordered value in each country [44]. Figure 5.8 shows the geographic map of these distributions for each country. By comparing the three different maps, it's clear that over 100 customers are concentrated in Europe, there's only one customer in the North America (NA) region. This is because the dataset we got came from Avery Dennison EU data warehouse, so we're unable to access the data in the NA region. Both the distributions of total ordered value and volume look similar. However, with its low amount of number of customers, Russia has higher ordered value compared to other countries with the corresponding higher of volume in

thousands of square meters (sqm). This indicates that compared to many sales orders concentrated in EU countries such as Germany, UK, Spain, Italy, Russia although has a small number of customers (i.e. around 10), it still has competitive ordered value and volume. The customers in Russia can be considered as big customers.Figure 5.7 shows the overview of the countries with the highest sales ordered volumes.

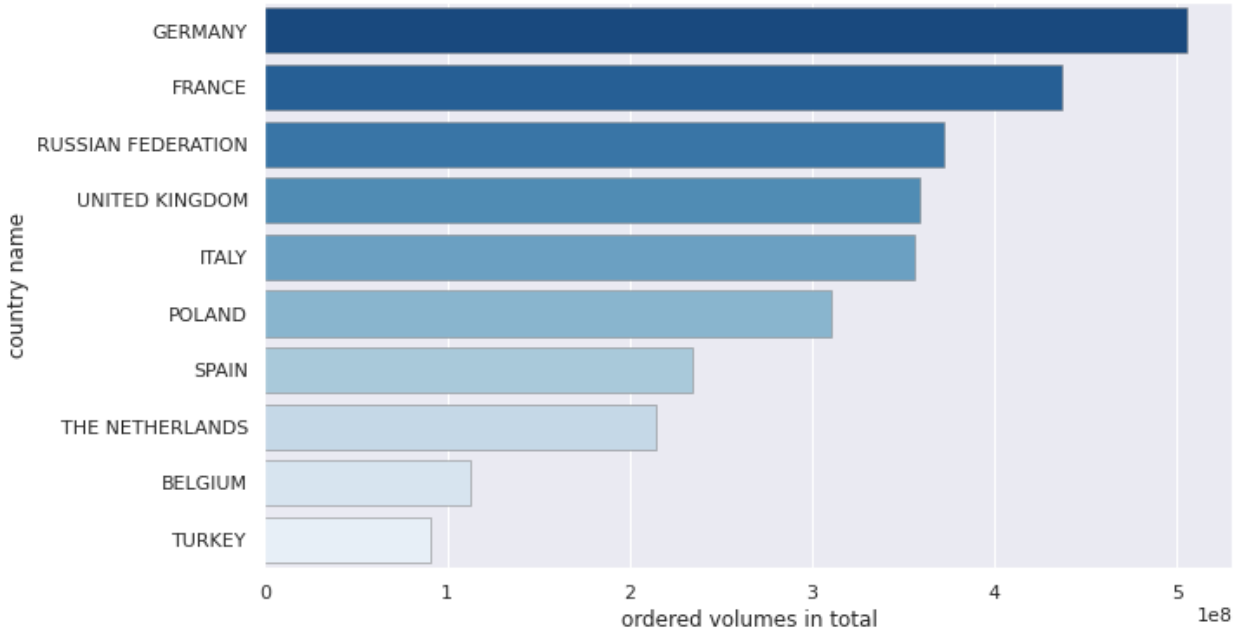


Fig. 5.7.: Top 10 countries ordered by the total sales volume.

Since our abnormal detection is based on time-series sales data, it's important to have an overview of how the sales ordered quantity and ASP change during the time. Figure 5.9 shows the changes of one UK customer for the top 5 ordered most category 2 products from 2019 October to 2021 January. In Figure 5.9a, we can find products such as MC and Digital PE are concentrating under 20000 square meters with a stable trend. We can also find that this customer tends to buy more quantities for products such as PE85 and PP50 with volatile changes through time. Figure 5.9b shows products PE85 and PP50 have stable average sales prices which are lower than the ASP of PP60 and Digital PE. MC has the lowest ASP. Therefore, we know that the changes for various category 2 products on the time-scale are different, for example, some show seasonal trend (i.e. PE85) and some have stable trend without any volatile changes (i.e. Digital PE).

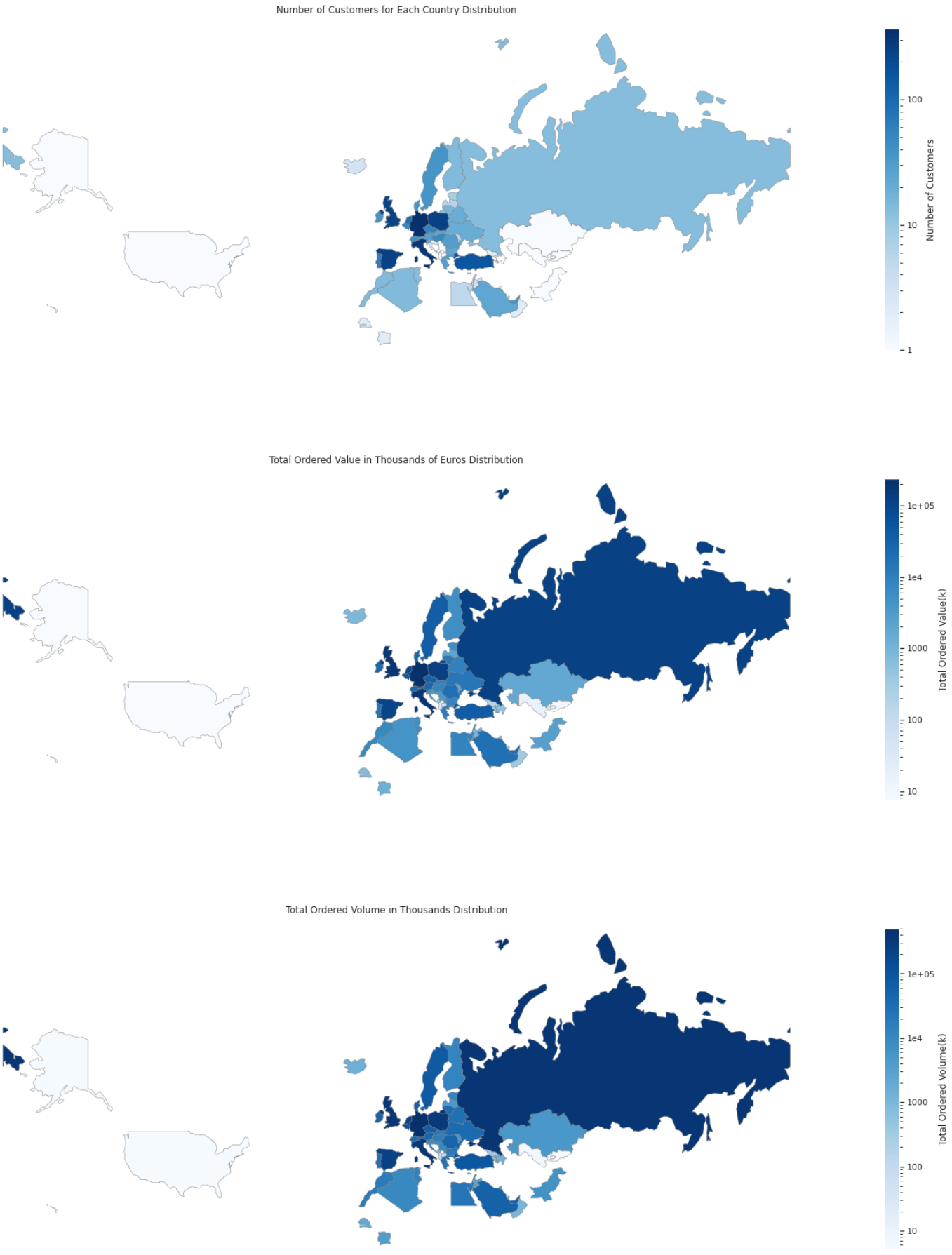
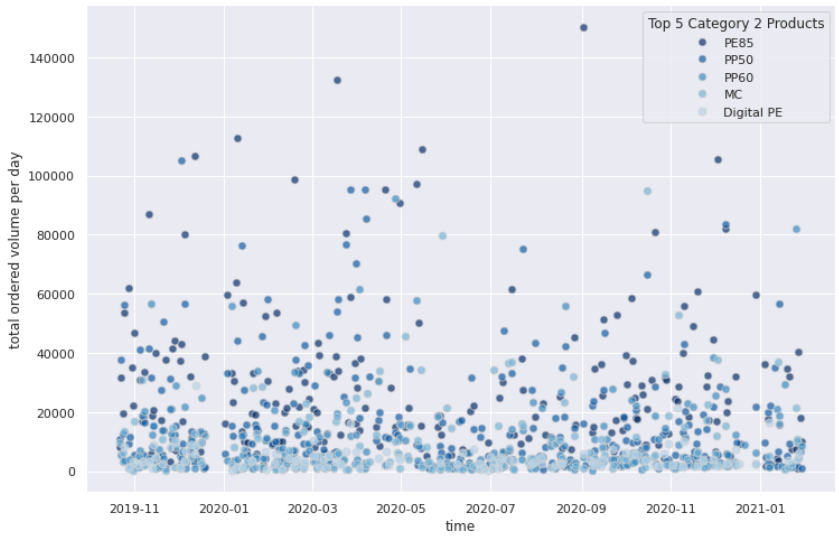
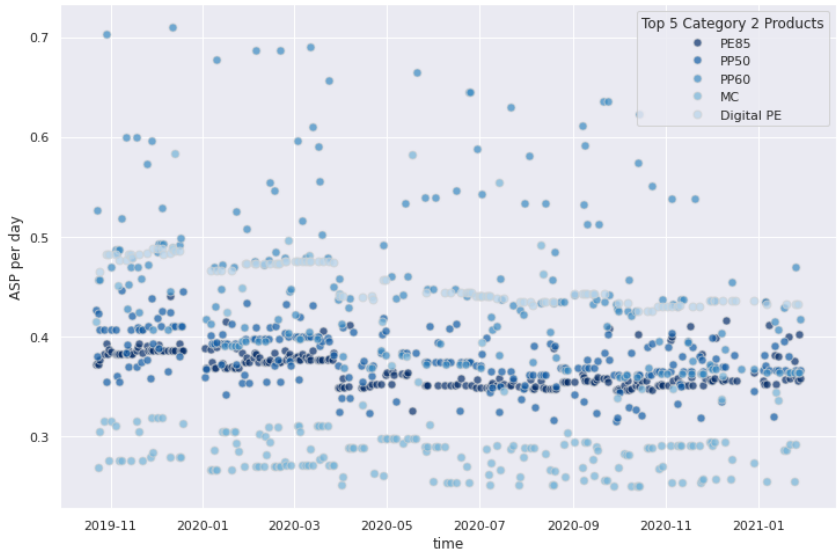


Fig. 5.8.: Geographic map for the distributions of customers, the total ordered value, the total ordered volume in each country.



(a) Volume Changes through Time



(b) Average Sales Price Changes through Time

Fig. 5.9.: Changes of volume and ASP on the time-scale for top 5 ordered most products.

In addition, since the sales department is interested in finding the difference of the customers buying patterns, the buying frequency distribution can provide information about how often the customer buys the specific product [45]. The buying frequency distribution for the same UK customer with the top 5 ordered most products shown in Figure 5.10. The x-axis represents the number of days between two consecutive orders for the same product category 2 products in a log-scale and y-axis shows the frequency in a stack visualization. Included are the kernel density estimations (KDE) using Gaussian kernels that indicate the density distribution of the buying frequencies. From the plot, it's clear that there are two peaks which happen when the

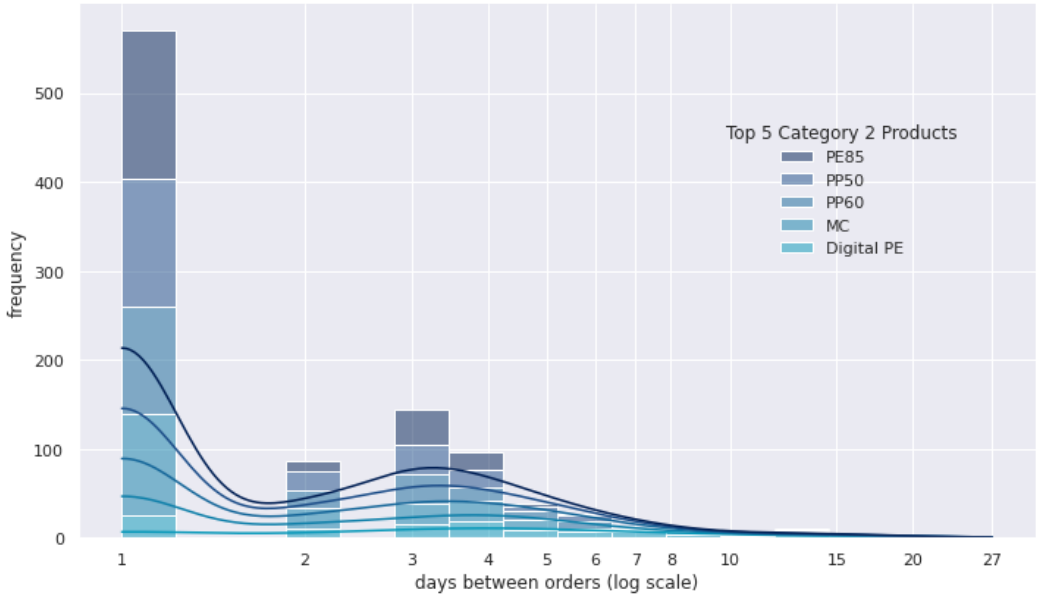


Fig. 5.10.: Buying frequency distribution based on the days between two consecutive orders.

days between two orders is 1 and 3 to 4. Most products are ordered on the next day, however, there are lower amount of these top 5 products ordered on the next 3 or 4 days as well. PE85 has the highest KDE which means that PE85 will be more likely ordered within 5 days and Digital PE has the lowest KDE which indicates the customer tends to get this products within 5 days with low likelihood and the buying patter is not stable.

As we discussed in section 4.1, an unsupervised clustering method DBSCAN has been used in studying the hidden patterns among customers. In this experiment, we use 4 numerical variables described in Table 5.1b to do the clustering and identify noise, in other words, to find abnormal customers (i.e. outliers). At the beginning, we aggregate all the transactions for each customer and take average for each numerical variable and then do the standardization by using the `StandardScaler()` method from `sklearn` to make sure all the numerical variables are standardized. Before we use DBSCAN, we reduce the dimensionalities through Principal component analysis (PCA) (see section 4.1). Figure 5.11 shows the amount of variance captured on the y-axis based on the number of components we include in the x-axis. A rule of thumb is to preserve around 80% of the variance, therefore, we decide to keep 2 principle components for DBSCAN method in detecting abnormal customers (i.e. noise).

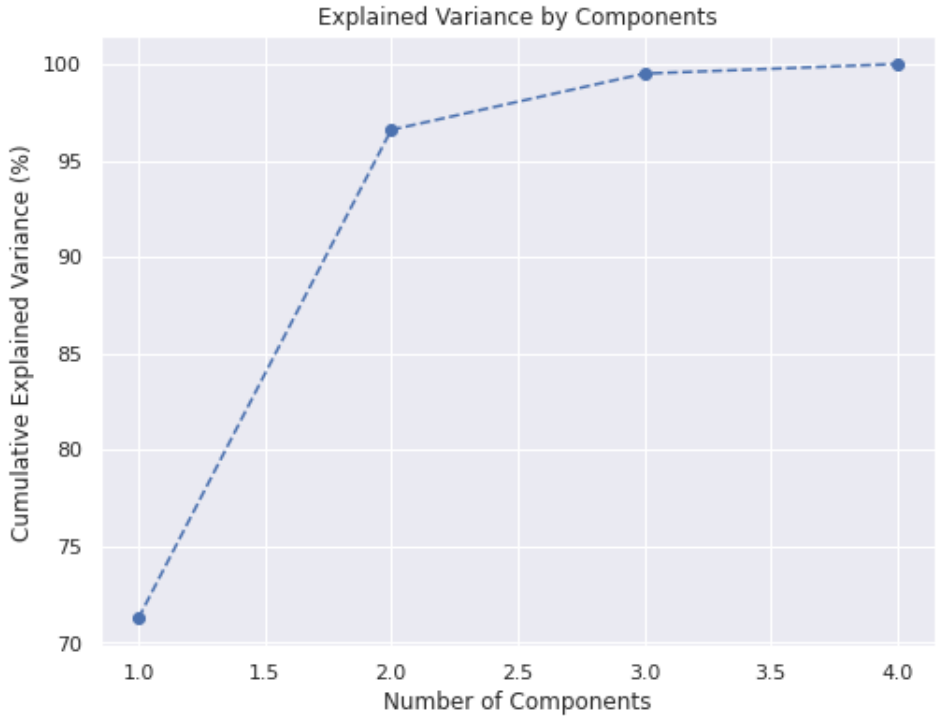


Fig. 5.11.: PCA Components

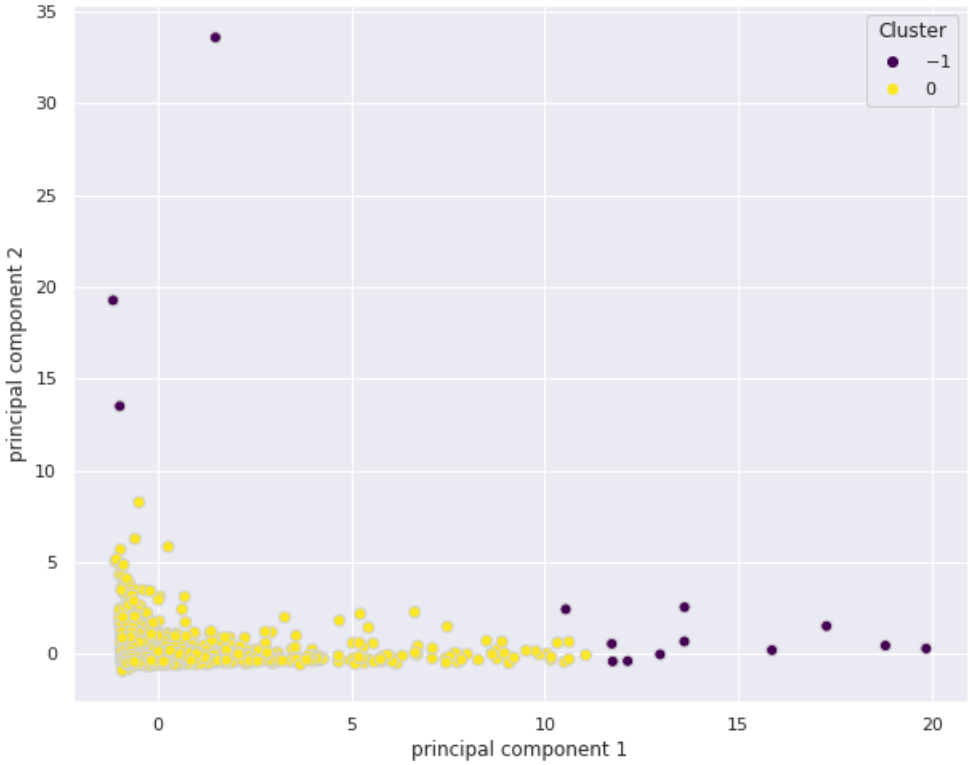


Fig. 5.12.: DBSCAN with PCA Components Result

For DBSCAN method, we set `eps=4` and `min_samples=100`. `eps` indicates the maximum distance between two samples, lower eps means more number of

clusters can be found. `min_samples` represents the number of samples can be considered as a core point in a cluster. Here we are interested in using DBSCAN to detect the noise (i.e. abnormal customers). Figure 5.12 shows the DBSCAN clustering result, each dot represents one customer. From the plot, label -1 indicates the abnormal customer and label 0 means the customers in the normal cluster. We can find that most of the customers are concentrated together and there are 14 abnormal customers. These abnormal customers all have abnormal buying patterns in ordered value, ordered volume, ASP, and metrical cost. There are three abnormal customers that have low ordered value and ordered volume but have high ASP such as 14, 19, and 33 euros respectively. Most of the customer are abnormal because of the high average ordered value over 100000, average volume over 30000 and average material cost over 20000.

6 Anomaly Detection

In this chapter we introduce various unsupervised methods that allow us to extract anomalies based on the property of uncertainty. Hypotheses are generated over the uncertainty with significance ($\alpha = 0.05$), to provide a binary determination over each data point whether they are within the normal range or outside the confidence interval and thus anomalous.

6.1 Auto-Regressive Model

Using the first year of sales data, we split into forecasting test window of 100 elements and use the previous data to fit an auto-regressive model (see section 4.2.1). For the model we use a multiple auto-regression model of order AR(1) through AR(7), which is fitted using Conditional Maximum Likelihood (CML). Seasonal dummies are included with an assumed seasonality of 30 time steps, and assumed are that the trend β_0 is a constant value. The forecast window and the fitted model are shown in Figure 6.1 with the residuals, KDE, Normal Q-Q and Correlogram given in Figure 6.2.

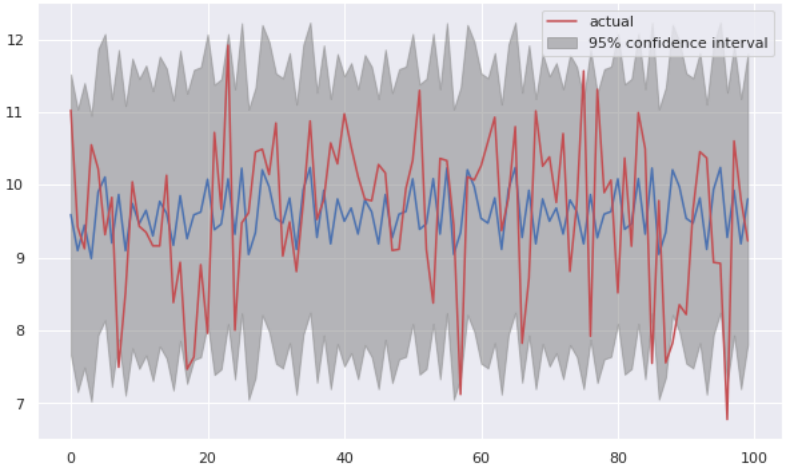


Fig. 6.1.: Autoregression Prediction with 95% Confidence Interval (in blue) vs Actual Time Series Data (in red)

While the forecast contains all actual data within the 95% CI, the model underfits as can be seen from the relatively large and unstable residuals [46]. Because the data is approximately normally distributed, shown in the Normal Q-Q graph [47], and the auto-correlation is lower than $\alpha_\rho = 0.2$, a more rigorous time-series model would be able to provide a better fit on the data while adhering to the assumption of normality and stationarity [48].

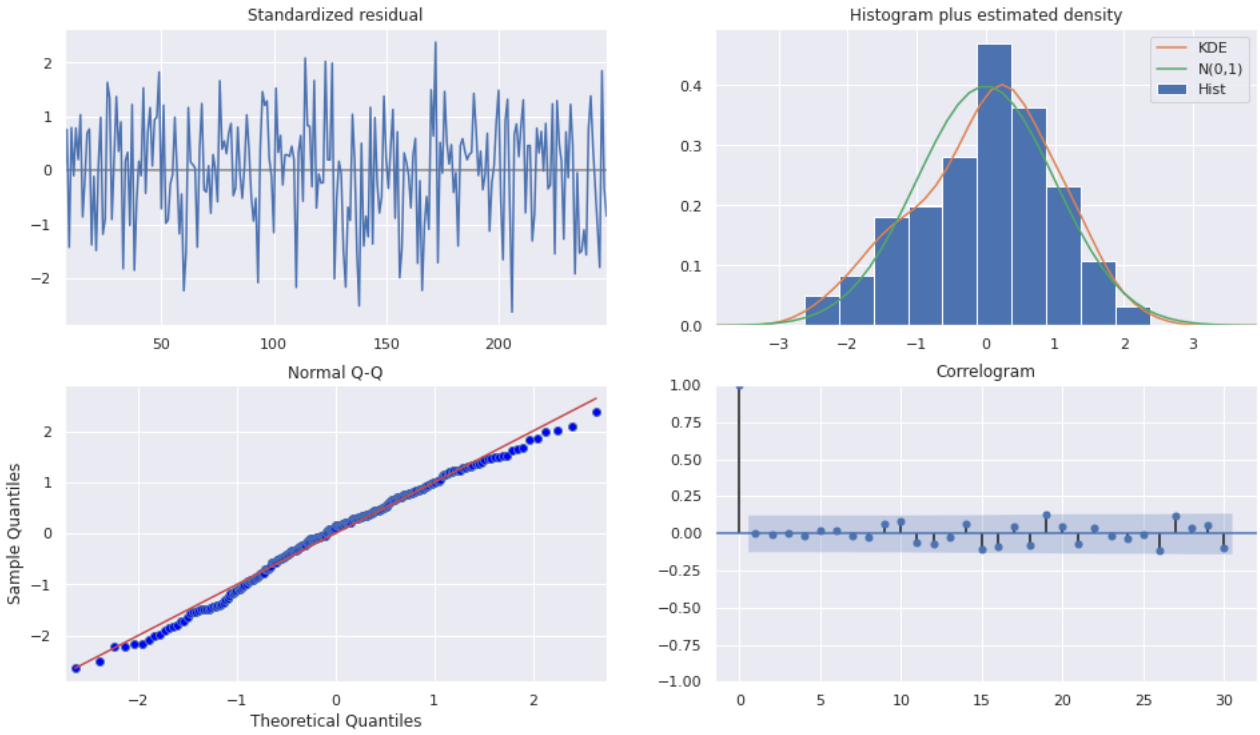


Fig. 6.2.: Autoregression Diagnostic Diagram

6.2 Anomaly Detection with Daily, Weekly, and Monthly Transactions

Different parameters can change the estimated/predicted value and confidence interval which lead to different anomalies when using the Prophet model. In addition, the model settings are different based on various daily or non-daily (i.e. weekly, monthly) transaction data which can cause the different performance. Therefore, it's important to have an insight on the relevant parameters and how the different settings can affect the predictions overtime. In this section, we give an overview on how various changes in parameters and settings can affect the model performance on anomaly detection.

6.2.1 Parameters

From the previous discussion, there are various parameters related to the three model components can be tuned and make an impact in generating the result of anomalies. According to Taylor and Letham [29], there are 4 tunable parameters and the deceptions are shown below:

1. `changepoint_prior_scale`: Changepoint prior scale τ controls the model flexibility described in the trend component (see Equation 4.7). A small changepoint prior scale can cause the trend underfitted and a large scale can cause the trend becomes overfitting. In the trend model, the default is 0.05.

2. `seasonality_prior_scale`: Seasonality prior scale controls the seasonal component flexibility (see Equation 4.10). Smaller seasonality prior scale will have lower magnitude of seasonality and higher scale will make the component becomes more volatile. The default seasonality prior scale is 10 without any regularization.

3. `holidays_prior_scale`: Holidays prior scale controls the flexibility of the holiday component (see Equation 4.12). Lower holidays prior scale can have less impact on the holiday component, higher scale can make the holiday magnitude becomes more unstable and may cause overfitting. The default is 10 that indicates no regularization.

4. `seasonality_mode`: There are two options for seasonality mode, one is "additive" which means the seasonality effect will be added constantly to the trend. If seasonality does not show additive effect but it grows together with trend, then "multiplicative" is a better option . The seasonality mode default is additive as we discussed in Equation 4.4.

In our experiments, we test all these 4 tunable parameters mentioned above with various values by using the daily, weekly and monthly transaction data in order to see how the model perform and the predictions differences with various parameter values. For seasonality, since the model does not contain monthly seasonality by default, we add the monthly seasonality with period $P = 30.5$ and Fourier order $k = 3$. In the Prophet model, weekly seasonality has $P = 7$ with $k = 3$ and yearly seasonality has $P = 365.25$ with $k = 10$ by default.

To show the predictions, we use the same UK customer that we discussed in the previous section (see Figure 5.10 and Figure 5.9) for one of the top 5 category 2 products called PE85. From the plots, we know that the customer orders PE85 really frequently and also has a high ordered volume with volatile changes through time. In addition, we use the natural log-transformed ordered volume in testing our experiments in order to avoid

getting the negative predictions. Therefore, it is meaningful to show the different prediction results and see how anomalies change under various set of parameter values.

6.2.2 Daily

In this subsection, we test the model with daily sales transaction data for different parameters. Table 6.3 shows the sets of parameter values that we choose for testing our experiments. In addition, in comparing how different seasonalities affect the prediction results, we also test weekly, monthly, and yearly seasonalities for our daily transaction data. For the baseline model and the test cases listed in the Table 6.3, the default seasonalities are monthly and yearly seasonalities. The reason why we do not use weekly seasonality is that most of the customers do not buy products daily so it is hard to capture the weekly trend and most of the products have the monthly and year periodic seasonalities.

Table 6.1.: Baseline model and 4 cases with different sets of parameter values.

name	baseline	case 1	case 2	case 3	case 4
Changepoint Prior Scale	0.05	0.05	0.5	0.05	0.05
Seasonality Prior Scale	10	10	10	0.01	10
Holidays Prior Scale	0.01	0.01	0.01	0.01	10
Seasonality Mode	additive	multiplicative	additive	additive	additive

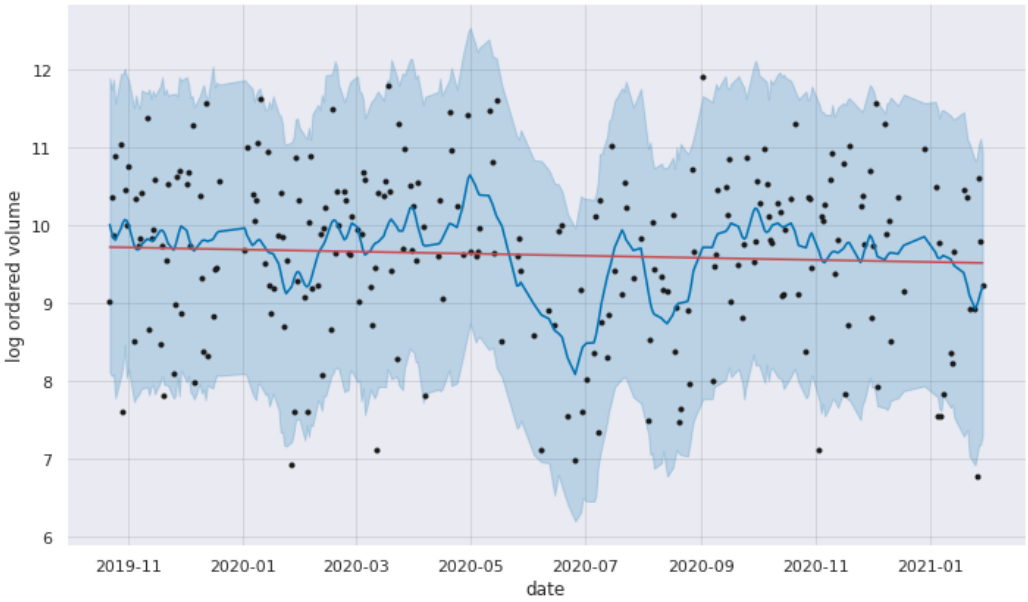


Fig. 6.3.: Daily Baseline Prediction Result

The baseline model prediction result is shown in Figure 6.3, each dot represents a daily order, the blue upper and lower boundaries represent 95% confidence intervals, the blue line means the predicted fitting result for log ordered volume, and the red line is the trend overtime. Since we set the threshold of the changepoints as 0.01, no obvious changepoints have been detected in the baseline model. The dots out of boundaries (i.e. dots are not in the blue shaded area) are considered as anomalies. From the baseline model, the prediction result is volatile and the ordered volume has a rapid decrease between 2020 May and July. The overall trend is negative since it is decreasing, the prediction blue line contains trend, seasonality and holiday effects.

(a) case 1 **(b)** case 2

(c) case 3 **(d)** case 4

Fig. 6.4.: 4 cases with different sets of parameter values in comparing to the baseline model.

In comparison, Figure 6.4 shows the 4 cases described in Table 6.3. From Figure 6.4a, after changing the seasonality mode from additive to multiplicative, there are small changes in the prediction such as the peaks shown in the baseline during 2020 March to April disappear in case 1 model and the predicting line becomes more stable during 2021 January. This is because with multiplicative mode, the seasonality grows together with trend which will make the seasonality effect less volatile when the trend is negative.

Figure 6.4b shows the prediction result after increasing the changepoint prior scale from the default in the baseline 0.05 to 0.5. It is obvious that there are additional 4 vertical dash lines appeared on the plot, these are the changepoints that have been used with the magnitude of rate change greater than the threshold 0.01. The red trend line is decomposed into 5 subsections, the trend increases till 2020 January and then decreases until 2021 January in different decreasing speed. In addition, increasing changepoint prior scale increases the sensitivity of the seasonality component since the peaks happened during 2020 March and 2021 January become higher compared to the same peaks in the baseline model.

In Figure 6.4c, the case 3 model has a lower seasonality prior scale 0.01 compared to the default one 10. It is clear that the prediction line becomes flatter and closer to the trend line. This is because there are less seasonality effect has been added to our piece-wise model and the magnitude of seasonality determines if the prediction is volatile or not. The seasonality peaks in the plots are also diminishing compared to the baseline model in Figure 6.3.

Case 4 prediction result shown in Figure 6.4d is more volatile than the baseline since we increase the holiday prior scale from 0.01 to 10. Therefore, we can clearly see with higher holiday scale, the magnitude of holiday component increases in specific dates which cause the peaks during 2020 March and 2021 January become higher. The overall prediction remains the same since the holiday component only effects the specific number of dates.

To show how different seasonalities affect the prediction result, we test weekly, monthly, and yearly seasonalities with the same baseline model parameter values listed in Table 6.3. The seasonality differences plot is shown in Figure 6.5 where the red line represents the trend component. Since seasonality is another component, the trend remains the same for each plot. In Figure 6.5a, the prediction result is stable in general but it shows many small weekly peaks overtime which indicates the weekly buying trend. There is a gap between 2020 May to 2020 July because the customer did not order ever day, there is no weekly periodic trend that can be captured during that time. Therefore, we decide to not consider about the weekly seasonality to avoid overfitting. Figure 6.5b shows the monthly periodic effect, it is clear that the prediction becomes more stable than weekly result. Monthly seasonality captures the change happened in a monthly period, however, it

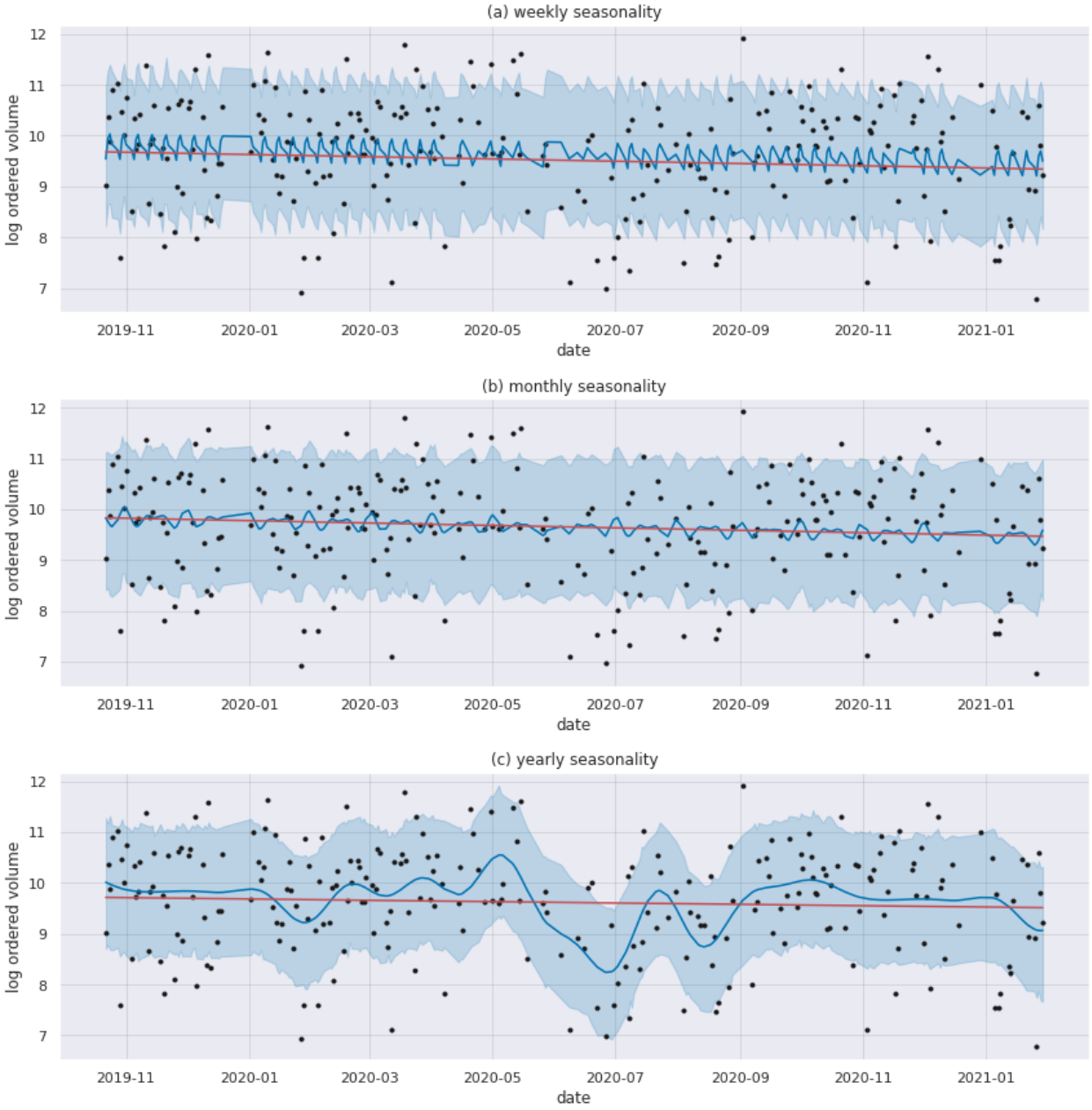


Fig. 6.5.: Weekly, Monthly and Yearly Seasonality Differences

does not capture the decreasing in ordered volume during 2020 May to July. For yearly seasonality (see Figure 6.5c), it captures the overall changes and it is the main seasonality we use for our daily transaction data. From the plot, the yearly prediction is the most volatile result compared to weekly and yearly, it also successfully captures the decreasing pattern from 2020 May to 2020 July and other changing patterns. Therefore, for our daily transaction data, we use monthly and yearly seasonality as our baseline seasonality component.

6.2.3 Weekly

Since in our transaction data, many customers do not order the products daily, it is necessary to see if the aggregated weekly or monthly transaction data can show better anomaly detection results. To test how the model changes based on the weekly data, we continue testing our baseline model and the other 4 cases listed in Table 6.3. In order to show the effects of changing parameter values, we use the same UK customer discussed in the previous section 6.2.2. For the weekly transaction data, each aggregated weekly order date starts on the first day of the week. It is possible that for some products, there are empty weeks with 0 ordered volume. For seasonality component, we use monthly and yearly seasonality for our baseline model and 4 cases.

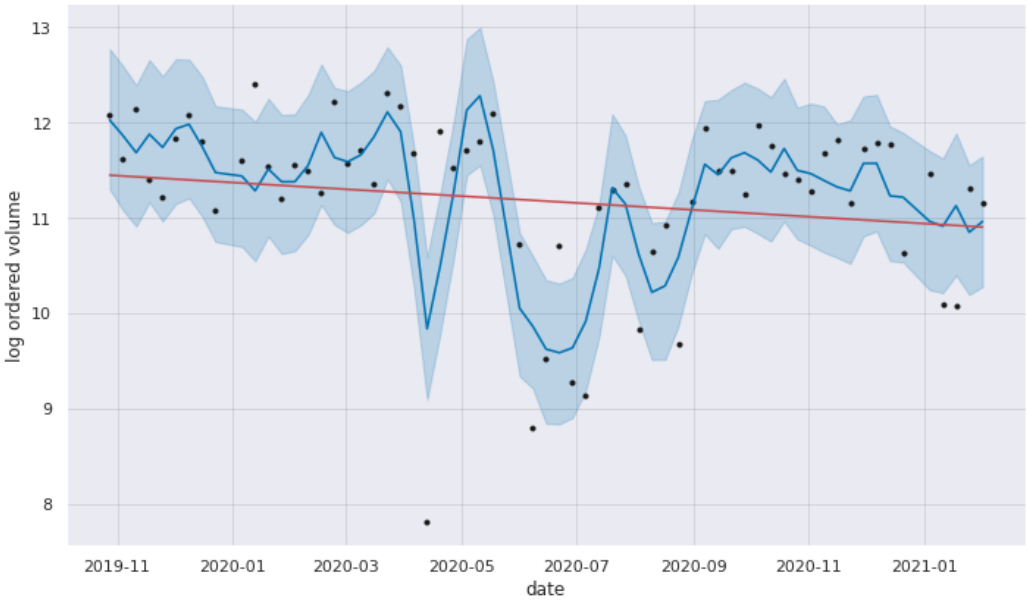
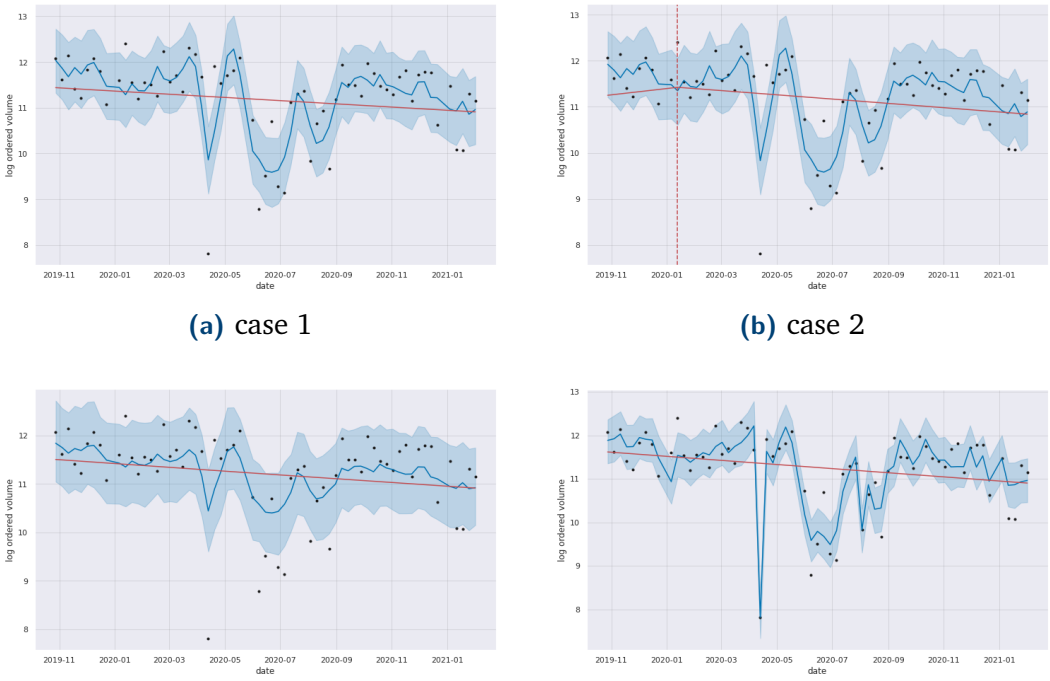


Fig. 6.6.: Weekly Baseline Prediction Result

Figure 6.6 shows the weekly baseline prediction, after aggregating daily transaction data into weekly, the number of orders with low ordered volume decreases and most of the weekly ordered volume concentrates between 11 and 12. The trend (see red line in the plot) decreases overtime which is similar with the trend in daily baseline model. From the plot, there are 10 weeks that are out of boundaries which can be considered as anomalies.

The prediction result for the 4 cases is shown in Figure 6.7. After changing the seasonality model to multiplicative, in Figure 6.7a, the prediction result is really similar to the baseline model except there are slightly changes on the lower and upper boundaries during 2020 July. The number of weekly

anomalies has the same number in baselines result which is 10. We can tell that seasonality model does not have a significant impact for the prediction result.



(a) case 1 **(b)** case 2

(c) case 3 **(d)** case 4

Fig. 6.7.: 4 cases with different sets of parameter values in comparing to the baseline model.

Figure 6.7b shows the result after increasing the changepoint prior scale to 0.5. It is clear that there is a changepoint (see vertical dashed red line) that has the rate change over 0.01 and it turns the trend into two segments. The trend increases till 2020 January and then decreases. Compared to the daily result in Figure 6.4b, less changepoints are used. The number of weekly anomalies in case 2 is 12 which is greater than that in the baseline model. Higher changepoint scale can cause the model becomes more volatile and detect more anomalies.

After decreasing the seasonality prior scale to 0.01, it is clear that in case 3 shown in Figure 6.7c the prediction is flatter and less volatile since the model have less seasonality scale. The smooth prediction curve detects 11 anomalies in total and most of them have low ordered volume. Such anomalies are inaccurate since they are not the real anomalies as these can be explained by monthly and yearly seasonalities.

Figure 6.4d shows the result when holidays prior scale is 10. It is obvious that when we have a strong holiday effect, one of the previous anomalies detected in the baseline during 2020 May is no longer an anomaly since it can be explained by the holiday events. The total numbers of anomalies in case 4 is 15 which has the highest anomaly amount among all cases. Strong holiday effect causes the model becomes more flexible which may detect anomalies that are actually normal orders (i.e. false positive).

6.2.4 Monthly

For monthly sales transaction data, comparing to the weekly level, there are less aggregated monthly orders. To test our baseline model and 4 cases, we continue using the same UK customer data as we discussed in the previous section. For the same customer with the same product, the number of transactions for daily orders is 249, for aggregated weekly orders is 64, and for aggregated monthly orders is only 16 data points which means that this customer buys PE85 for 16 months in total during the period from 2019 October to 2021 January. Because of the few monthly data points, in order to avoid overfitting, we change seasonality prior scale of the baseline model to 0.01 rather than remaining 10. The sets of parameter values for the baseline model and 4 cases is shown in Table 6.2.

Table 6.2.: Baseline model and 4 cases with different sets of parameter values for monthly data.

name	baseline	case 1	case 2	case 3	case 4
Changepoint Prior Scale	0.05	0.05	0.5	0.05	0.05
Seasonality Prior Scale	0.01	0.01	0.01	0.5	0.01
Holidays Prior Scale	0.01	0.01	0.01	0.01	10
Seasonality Mode	additive	multiplicative	additive	additive	additive

The baseline model prediction result is shown in Figure 6.8. After aggregating the daily transaction data to monthly level, each dot represents the log-transformed ordered volume sum for the related month. The date for each monthly orders is the last day of the corresponding month. In the baseline plot, the trend decreases overtime, there is no significant changepoints with change rate over 0.01, and the model detects 5 anomalies in total.

The prediction results for the 4 cases is shown in Figure 6.9. There is no significant changes when we change seasonality mode from additive to multiplicative (see Figure 6.9a), the baseline result and the case 1 result are

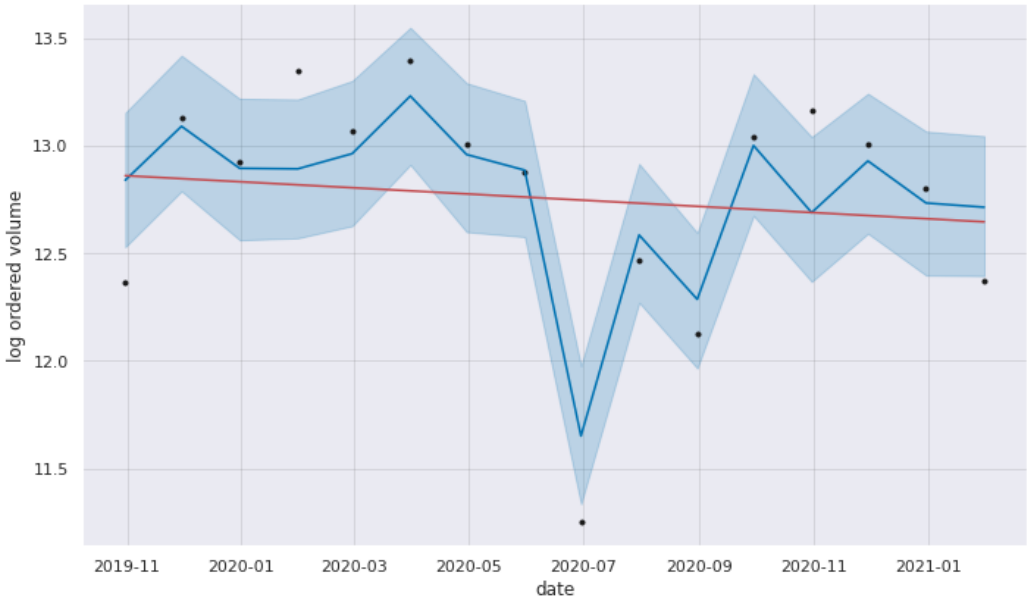
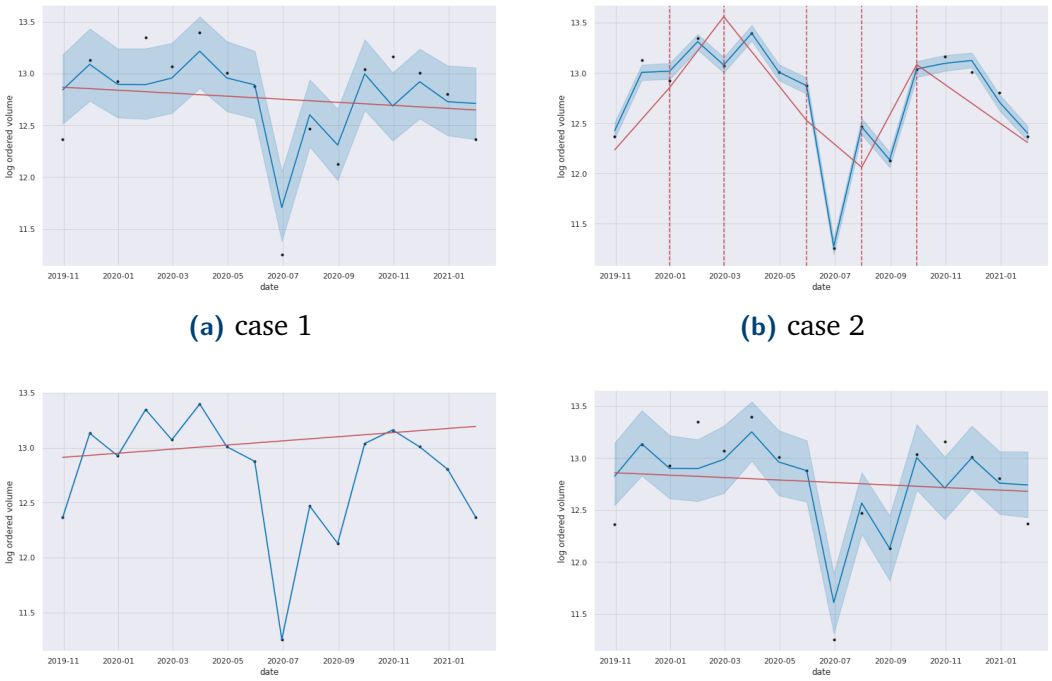


Fig. 6.8.: Monthly Baseline Prediction Result

similar. The boundaries in case 1 during 2021 January is wider than that in baseline case because multiplicative mode can make seasonality become more flexible, therefore, the total anomalies detected in case 1 is 4. In case 2, we increase the changepoint prior scale from 0.05 to 0.5 and the result shown in Figure 6.9b has smaller boundaries and more fitted prediction results than the baseline model results. The reason why the confidence intervals are small is that the predictions are closer to the actual values with the high changepoint scale which causes the confidence interval becomes smaller. In the case 2 plot, there are 5 significant changepoints impact the trend and make the trend becomes more flexible and predicts the overall ordered volume pattern correctly. It is interesting to find that with fewer data, it is more susceptible to changing trend for the model. The total number of detected anomalies is 4 which is the same amount in case 1.

In case 3, we increase the seasonality prior scale from 0.01 to 0.5 and the result is shown in Figure 6.9c. From the plot, there is no confidence intervals, the red trend line does not predict the correct downward pattern, and all the monthly data points are connected together. This is because with high seasonality scale on monthly level, the model causes overfitting and it gives too much credence to seasonality with few data points. Case 4 we increase the holiday prior scale from 0.01 to 10, however, there is not significant effect that we have discovered in daily and weekly prediction results. This is because holiday component focuses on the specific event dates. On monthly



(a) case 1

(b) case 2

(c) case 3

(d) case 4

Fig. 6.9.: 4 cases with different sets of parameter values in comparing to the baseline model.

level for the same even date, there is no ordered volume records thus cannot have significant influences on the prediction results. The anomalies detected in case 4 is same to the the baseline anomaly results.

6.3 Optimization

As we discussed in section 4.3, we preform an optimization approach grid search and evaluate our model by using cross-validation method to find the optimal model with best hyperparameters for our aggregated daily, weekly and yearly transaction data.

The evaluation metrics that we use are Mean Squared Error (MSE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Median Absolute Percentage Error (MdAPE), and Coverage. MSE measures the mean of the square difference between the predicted value and the actual value, the formula is $\frac{1}{n}\sum(y-\hat{y})^2$ where n is the total number of orders in our experiment, $\hat{y}$ means the prediction. RMSE with formula $\sqrt{\frac{1}{n}\sum(y-\hat{y})^2}$ is the root mean squared difference between prediction

and actual value. MAE measures the mean of the absolute difference between the predicted value and the actual value, the formula is $\frac{1}{n}\sum|(y-\hat{y}|$. MSE, RMSE, and MAE are all the metrics in showing the variation of the errors, the lower MSE/RMSE/MAE means the model has a better prediction result.

The formula for MAPE is $\frac{1}{n}\sum|\frac{(y-\hat{y}}{y}|$ which measures the model prediction accuracy in percentage, lower MAPE means the model performs better. MdAPE is the median score selected from a range of the absolute percentage errors (APEs) where the formula is $|\frac{y-\hat{y}}{y}|$ for each time. If MDAPE is less than MAPE, it means the distribution of APEs is skewed to the right which indicates that most predictions have smaller errors. If MDAPE is greater than MAPE, the APEs distribution is skewed to the left and most predictions have higher errors. Coverage measures the coverage probability of the actual value under the confidence interval of the randomly generated sample. Since our confidence interval for the model is 95%, ideally the coverage should close to 95% if there is enough data and it is perfectly normally distributed. However, because our data contains anomalies and is only close to normal distribution, so the coverage value should always smaller than the ideal value. Lower coverage means the model fails to include the normal orders and have more false positives. Higher coverage means the model performs better and have less false positives.

Table A.1 shows the model performance with different hyperparameters for our daily sales transaction data starting with the lowest RMSE score. It is clear that for daily data, the model with 0.05 changepoint prior scale, 10 holidays prior scale and 0.01 seasonality prior scale performs the best with MSE equals to 1.4455 and RMSE equals to 1.2023. Both MAPE and MDAPE are low and MDAPE is smaller than MAPE indicates the APE distribution is skewed to right and most error are small. The coverage is 85.89% which is closer to 95% indicates that the model detects the anomalies without many false positives. From Table A.1, the high holiday prior scale increases has model flexibility and also the predicting accuracy. Seasonality is less important than the default value 10 in our baseline model (see Table 6.3). From the table, RMSE increases rapidly as the seasonality prior scale increases, this is because higher seasonality scale can cause the prediction result becomes more volatile which can leads to overfitting.

The model performance for weekly data after grid search is shown in Table A.3. The model that performs the best with the smallest RMSE score

which is 1.2486 has 0.1 changepoint prior scale, 0.01 holidays prior scale, and 0.01 seasonality prior scale. Comparing to the best hyperparameter derived in the model for daily data, the model for weekly data focuses more on the changepoint prior scale that controls the flexibility of the trend. Holiday prior scale and seasonality prior scale are not as important as the changepoint scale. MAPE is lower than MAE means most of predictions have small errors. Higher seasonality prior scale causes the weekly model more unstable and performs worse. The coverage for the best model is 55.88%, this is because there are less data points for weekly data which deceases the coverage since the generated region covers less actual weekly orders.

Table A.3 shows the grid search performance result for monthly data. Unlike the daily model focuses on holiday prior scale and the weekly model focuses on changepoint prior scale, the monthly model focuses on both changepoint prior scale and the seasonality prior scale. The model with 0.1 changepoint scale, 0.01 holiday scale and 0.01 seasonality scale performs the best with MSE equals to 0.7943 and RMSE equals to 0.8912. It is clear that on monthly level, the model focuses more on the trend and the seasonality components rather than holiday component. Comparing with the baseline model with the default changepoint scale 0.05, the model is more flexible.

6.4 Anomaly Analysis

In this section, we analyze the detected anomalies by using our daily sales transaction data in order to haven an insight on the normal orders and anomalies distributions. In the end, we purpose the binomial models for anomalies based on different products. After using the optimal model for daily data derived from grid search (see Table A.1) with 0.05 changepoint prior scale, 10 holidays prior scale and 0.01 seasonality prior scale, we analyze the abnormal results with the aim of finding the patterns for anomalies under different products.

To have an overview of the anomalies, first we look at the top 5 countries that have the high amount of anomalies and the plot is shown in Figure 6.10. The plot shows the distribution of abnormal orders for each country on a timescale, Poland has the highest number of anomalies and Germany has the second highest number of anomalies. In the plot, there is no specific anomaly pattern for each country since anomalies are widely distributed. However, it

is clear that when log ordered volume is around 8, there is a gap between the upper anomalies and lower anomalies, this is because for our anomaly results, there are positive anomalies and negative anomalies. The positive anomaly means that the customer orders more than the expected value and the negative anomaly means that the customer orders less than the prediction. From Figure 6.10, most of anomalies for Poland is more concentrated and there are some anomalies for Germany and United Kingdom have extremely high ordered volume. Some anomalies for Italy have extremely low ordered volume.

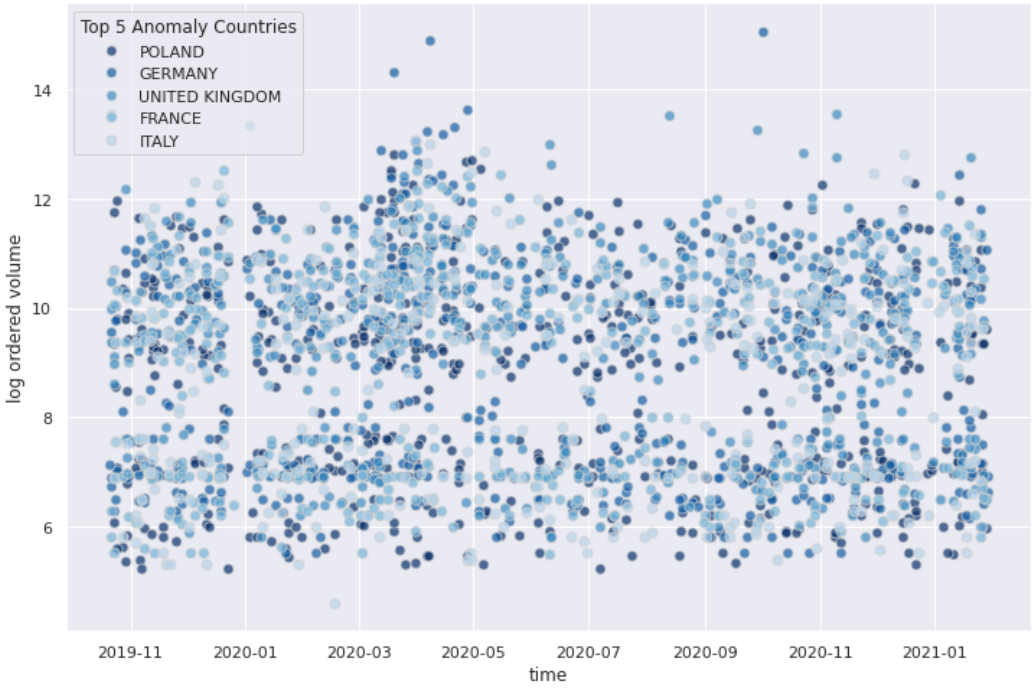


Fig. 6.10.: Top 5 Countries with High Numbers of Anomalies

In addition, we distribute the anomalies for customers (see Figure 6.10) and study how customers behave overtime and if there are a group of customers that have many anomalies during one period. In the plot, different dot color represents different customers and each dot means an abnormal order. It is clear that there are a large amount of anomalies clustered from 2020 March to 2020 May. During this period, most of the anomalies are positive anomalies which indicate that customers buy products with a high ordered volume. There are small gaps during 2020 December and 2021 December, this is because of the holiday seasonality that most of countries have Christmas and New Year which decreases the number of anomalies. Another reason is that not many customer place their orders during holidays. Most of the anomalies are concentrate between 6 to 12 log ordered volume, however,

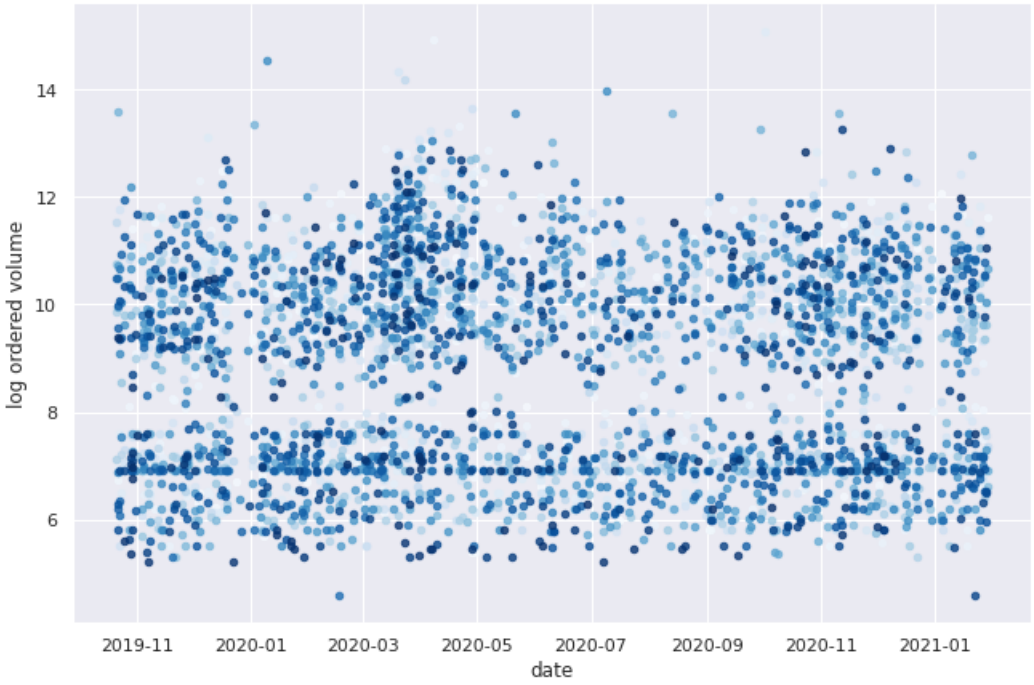


Fig. 6.11.: Customer Anomalies Distribution Overtime

some customers have more orders with extremely high ordered volume than orders with extremely low ordered volume.

Fig. 6.12.: Three Products Examples with Anomalies Overtime

To have an insight of anomalies distribution with different products, we select 3 top ordered products (i.e. PE85, PP50, and PP60) among customers as our examples. The anomalies for each product overtime is shown in Figure 6.12 where y-axis is the ordered date and x-axis is the log ordered volume for each product. The two labels are negative anomaly which is an abnormal order with lower volume than prediction and positive anomaly which is an

abnormal order with higher volume than expected. It is obvious that from the plot, negative anomalies and positive anomalies are separated from each other. The anomaly distributions for each product are different, the positive anomalies are concentrated during 2020 March to 2020 May in product PE85. The anomalies in PP50 are widely spread and there are some overlaps with both negative and positive anomalies happened during 2020 March which indicates the customers behave differently during the period, some customers have more extreme low ordered volume than expected. There are more positive anomalies in buying product PP60 which means that the customer buying patterns are more volatile and the customers are diverse since the range of both negative and positive anomalies are wide.

(a) Three Product Examples Normal Orders Distribution

(b) Three Product Examples Abnormal Orders Distribution

Fig. 6.13.: Frequency distributions comparisons for three product examples.

An compassion between the normal orders and anomalies frequency distributions for the three products is shown in Figure 6.13 where y represents the log ordered value, count is the frequencies for the log-transformed volume, and the line represents the kernel density estimation (KDE) of the distribution. In Figure 6.13a, the frequency distribution of normal orders in product PE85 is close to normal distribution and it is slightly skewed to right, most of the normal orders are concentrated when log ordered volume is 8. PP50 has

smaller distribution than PE85 with lower ordered frequency around 500 in general, the distribution is also close to normal distribution and there are two small peaks. It is clear that PP60 has a distribution with high ordering frequencies greater than 2500 with the log ordered volume around 8 the distribution is skewed to right. There are also two peaks in PP60 normal order around 8. The normal order frequency distribution indicates that different products have different ordering patterns in not only ordering frequencies and ordered volumes individually.

In comparing with the normal order distributions, the anomalies frequency distributions shown in Figure 6.13b perform differently for each product and are more close to normal distributions than that in normal order distributions. The anomaly with label 1 means a positive abnormal order, the anomaly with label -1 is a negative abnormal order. In Figure 6.13b, the positive anomalies distributions are separated from the negative anomalies distributions with small overlaps. It is clear that the frequencies of anomalies are much smaller than the normal order frequencies. In addition, in PE85 distribution, more negative orders are detected than positive orders and positive orders have a wider distribution than negative anomaly distribution. In PP50, both distributions are similar except the negative distribution is more concentrated around log ordered volume 7. The negative anomaly distribution in PP60 has a peak with around 160 anomalies, more positive anomalies have been detected than negative anomalies and the positive anomaly distribution is approximately to normal distribution.

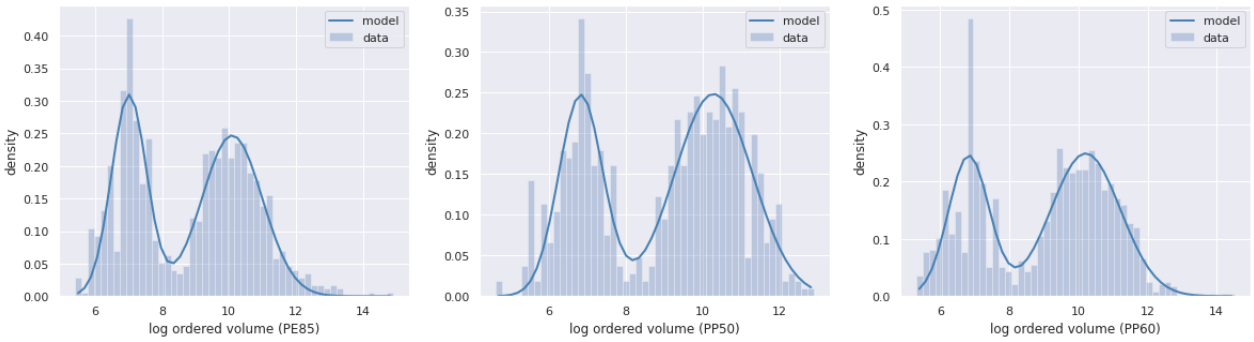


Fig. 6.14.: Products Anomaly Model

From Figure 6.13b, both the negative and positive abnormal order distributions are approximately to normal distributions which lead to a mixture of two normal distributions, in other words, such mixture can be considered as a bimodal distribution for abnormal orders. Rossi [49] describes the mixtures of univariate and multivariate normal distributions. As we discussed in the

section 4.4, based on the detected anomalies labels {-1, 1}, we are able to conduct a bimodal distribution that contains two Gaussian components for each product. Derived from Equation 4.14, the bimodal model probability density formula can be defined as

$$p(x) = A_1 \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\mu_1}{\sigma_1}\right)^2\right) + A_2 \cdot \exp\left(-\frac{1}{2}\left(\frac{x-\mu_2}{\sigma_2}\right)^2\right) \tag{6.1}$$

where A_1,A_2 are the proportions of the first and second distributions which can be estimated with MLE Rossi [49], μ is the distribution mean, σ represents the standard variance of the distribution. The bimodal distribution model for each product is shown in Figure 6.14. The estimated values for parameters and the standard deviation errors on the parameters (i.e. sigma) of each product bimodal model are listed in Table 6.3.

Table 6.3.: Bimodal Distribution Parameters and Standard Deviation Errors Summary for three Product Examples

	params PE85	sigma PE85	params PP50	sigma PP50	params PP60	sigma PP60
μ_1	7.011393	0.045694	6.821975	0.060565	6.823961	0.078975
σ_1	0.525920	0.046411	0.574510	0.061389	0.589295	0.080764
A_1	0.309155	0.022837	0.247063	0.022187	0.244937	0.027824
μ_2	10.094075	0.074418	10.278292	0.079381	10.209560	0.100557
σ_2	0.905030	0.077090	1.012411	0.082477	1.007424	0.104694
A_2	0.247172	0.017548	0.248694	0.016870	0.249258	0.021447

From Figure 6.14 and Table 6.3, the bi-modal model for PE85 estimates anomalies distribution under a maximized likelihood estimate with the first distribution mean $\mu_1 = 7.01$ and second distribution with mean $\mu_2 = 10.09$, the standard deviations are relatively small with $\sigma_1 = 0.53$ and $\sigma_2 = 0.90$ respectively leading to a bimodal distribution. Since μ_1(i.e. the mean of the first distribution component) in the table is less than μ_2, it indicates that the ratio of under-performing to over-performing sales is evaluated at $A_1/A_2 = 0.309155/0.247172 \approx 1.250769$, thus anomalies below the mean of the prophet model are more common than above.

In the PP50 bi-modal model, we can find that the second distribution has a larger standard deviation of 1.01 than that of the first distribution of 0.57. In addition, since μ_1 in PP50 is also less than μ_2, we can state that anomalies below μ_1 are more common than above μ_1. The performance of PP60 is similar with pp50 bi-modal model. Overall, the positive anomalies distribution has higher standard deviation than the negative anomalies distribution resulting in greater dispersion for sales below expectation. The σ columns shows the standard deviation errors for each parameter, from Table 6.3, all of the errors

are proportionally insignificant which results that our bi-modal models for anomalies have good estimation performance. Table A.4 shows additional bi-modal model parameter results for 10 products.

6.5 Validation

In this Section we discuss the validation of the anomaly detection model that we created. We do this by collecting feedback on the prediction results from various sales people. Also, we discussed with them the overall usability of the model. The following summary shows our validation process and result:

- Preparing and presenting anomaly reports to 5 sales people. The anomaly reports contain variables such as ordered volume, ordered value, customer names, product category 2 etc. The report also includes the reasons why we report this order as an anomaly, the reason could be the ordered volume is lower than the expected value. The report uses the aggregated monthly data, thus the detected anomalies are on the monthly level.

- Ask for structured feedback by presenting them with a spreadsheet in which they can indicate which anomalies were considered valid and why. In the anomaly report, we ask sales representatives to help us validate if the detected anomaly is a real anomaly and is there any normal orders that should be considered as an anomaly.

- Only 2 sales people provided useful feedback for the validation in the end, covering 5 customers. Many sales representatives provide useful feedback in improving the anomaly report, unfortunately, not many validations for the detected anomalies have been done for most of the customers because lacking of sales representatives. Therefore, we are only able to gather the validation results from 5 customers.

We use 3 different measurements for our validation results. Here are the formulas for our metrics:

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

Precision measures the percentage of real anomalies among all detected anomalies.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Recall measures the percentage of real anomalies to the results which include both the real anomalies and the false normal orders (i.e. orders detected as normal orders but should be anomalies).

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{Total}}$$

Accuracy measures the percentage of real anomalies and real normal orders from the total orders.

Table 6.4.: Validation Result for 5 Customers

Customer	Precision (%)	Recall (%)	Accuracy (%)	Num of Anomalies (%)
A	67.86	83.33	91.90	14.40
B	54.17	91.67	90.16	18.22
C	55.56	83.33	89.58	14.58
D	59.72	87.21	91.78	20.82
E	61.90	84.56	92.39	17.35

The feedback form summary is shown in Table 6.4, we can find that the average precision is around 60% which indicates that from all detected anomalies, there are 60% anomalies are correctly detected. High precision means that the model has less false anomalies (i.e. normal orders detected as anomalies). The average recall is high and it is around 85%, it means the model predicts less false normal orders, in other words, the model performs well in detecting more potential anomalies without detecting them as normal orders. The most of the accuracy results for customers is above 90%, this is because the accuracy accounts for both the real anomalies and the real normal orders. From this validation we conclude that the anomaly detection model is useful for detecting the real anomalies without detecting normal orders as anomalies.

7 Conclusion

In this thesis we set out to perform time series analysis through unsupervised learning on unlabelled sales data, in order to detect anomalies in the sales volume and patterns. We approach this by using an automated time-series prediction model, elaborating each component within the model, and finding three optimal models for our daily, weekly, and monthly aggregated sales transactions data.

Based on our detected anomalies, we purpose a mixture model in the form of a bi-modal anomaly probability density specification, which is uniquely optimized for the baseline behaviour per customer in relation to specific product categories. By starting from a baseline and then optimizing models for behaviour of individual customers, a more reliable model of uncertainty can be achieved in comparison to a model that assumes all customers have identically distributed sales patterns.

Furthermore, we have used a small sample of validation data labelled by sales representatives to validate the estimation of anomalies for a set of five specific customers, which on average have shown an estimated accuracy of 91.16% on the anomaly classification of the validation data. This shows that successful identification of anomalies can be achieved through unsupervised methods, by taking advantage of the distribution of the data itself and modelling the uncertainty as a classification criteria based on binary hypotheses on the confidence interval.

In addition, this approach has given us insight into the distribution of the anomalies, while also giving the statistical properties of anomalies in the sales data categorically per customer and item. These properties can then be used to perform causal inference, leading to direct business insights that can help identify the cause and effect of sales data anomalies.

Below we will provide explicit answers to our research questions, enumerate our contributions, and describe possible avenues of future work.

7.1 Can prediction models for time series data be used to detect anomalies in sales data?

We have demonstrated that it is indeed possible to design an anomaly detection model for sales data based on an uncertainty model for time series data. We use an auto-regressive model as our baseline model, however, the model shows under-fitting and is not appropriate for customer-specific modelling where data is more sparse. Therefore, we move on to the more rigorous prophet model that utilizes automated machine-learning to introduce change-points and models three deterministic components, which are the trend, seasonality, and holidays components.

We elaborate each component and build the optimal model through grid search for daily, aggregated weekly and monthly data respectively in detecting daily and non-daily anomalies. In general, the anomalies are the outliers out of the 95% confidence interval based on the the observations under the uncertainty model. We discover that the anomalies have a bi-modal distribution under maximum likelihood. In order to analyze the detected anomalies, we propose that an optimized mixture model for anomalies under different combinations of customers and product categories. Based on the validation data, labelled and provided by the sales team, measured over the uncertainty model and evaluated for precision, recall, and accuracy, we conclude that the time-series prediction models can be used to reliably detect sales anomalies.

7.2 How can we utilize unsupervised learning to create and optimize an anomaly detection model to classify our data?

While theoretically, given the availability of class labels on the data, either supervised or non-supervised methods could be used. In supervised training,

the labelled data is necessary so that we can use the supervised or semi-supervised training methods. In addition, most labelling is performed by human annotators, which is a costly and time consuming process that may be also biased if not accounting for a significant inter-rater agreement sample. Unsupervised learning on the other hand makes no assumptions over the classification labels of the data, and instead uses the inherent distribution of the data to determine what constitutes as an anomaly.

We find that in a practical setting, collecting sufficient labels to enable reliable supervised learning proved challenging. Due to lacking resources in terms of experts that can reliably label the data, especially in consideration of the volume and veracity of the data that is being recorded every day of the operation. Even if we assume that enough resources are available for labelling, this process would take place after orders are already processed. While if we apply unsupervised methods, sales data can be analyzed in real time without requiring labelling in order to adjust the model in light of new data. What constitutes as an anomaly within the sales patterns changes over time, so the most efficient model requires quick adaptation to adjust to market trends.

To create an optimize our unsupervised anomaly detection model, we discuss the auto-regression as our baseline model to determine anomalies (see section 6.1) in extracting anomalies. Then we elaborate an automated time-series prediction model and find three local optimal models for our daily, weekly and monthly aggregated sales transaction data in classifying our data through grid search which is a hyperparameter optimization method. By using the labels generated by the anomaly detection model, we purpose a Gaussian mixture model in modeling the anomalies as a bimodal distribution.

7.3 How can the performance of the resulting anomaly detection models be determined in a practical setting?

The conditions for validating the performance of the prophet uncertainty model are based on post-classification of previous sales data. This can be

done by taking a sample of the data, then having a team of sales experts analyze these for anomalies and classify these data. Since these are limited samples from the original data, classification of small samples is significantly more feasible than classification of all data for training a supervised model. With these samples, represented by the classification labels of individual domain experts for the sales data, we can use an inter-rater agreement metric to estimate the true label of these data points. From these labels we can estimate the classification performance of the uncertainty model, while comparing it to the mean and standard deviation within classification performance of the human annotators.

To validate the estimation of the detected anomalies, we create reports to present detected anomalies to sales representatives and collect their feedback. With this method, we were able to collect the labelling feedback from two users for five customers as a small validation set but not as a means to train or model. Based on the validation result, we use three measurements (i.e. precision, recall, and accuracy) to evaluate our model performance.

7.4 Contributions

Throughout this project we made the following contributions:

- Data analysis for sales transaction data in preprocessing, data exploration, and detecting abnormal customers by using DBSCAN with PCA components for customer-based sales aggregates.

- Unsupervised training by using the piece-wise automated prediction model for aggregated daily, weekly and yearly transaction data with the baseline model and four different settings of trend, seasonality and holiday components.

- Automated Prophet machine learning model optimization by efficient hyper-parameter optimization through forecast-based grid search and evaluation of model performance via MSE, RMSE, MAE, MAPE, MDAPE and Coverage.

- Anomaly analysis on optimal prophet-based uncertainty model, leading to a Gaussian mixture model in the form of a bi-modal distribution

for detected anomalies in the form of abnormal orders with unique parametric models under maximum likelihood for each combination of customers and product categories.

- Description and example of a validation procedure, where a small sample set is labelled by human annotators using inter-rater agreement as validation from which an estimation of model performance can be generated through the precision, recall and accuracy metrics.

7.5 Future Work

In our experiments, we have a baseline auto-regressive model in the beginning and then move on to the automated forecasting model in detecting anomalies since the baseline model underfits our transaction data. It is interesting that for different aggregated data (i.e. daily, weekly and monthly data), the optimal model has different hyperparameters. Therefore, in the future, we hope to automate the selection process and build the specific model for each customer.

Zong et al. [8] purposes a similar architecture which is a Gaussian Mixture Model (GMM) to model the error by using auto-encoding instead of auto-regression as the bi-modal model purposed by us. Thus, we would like to compare both unsupervised models and evaluate which model performs better in detecting anomalies. Munir et al. [6] purpose a convolutional neural network (CNN) based prediction model for unsupervised learning on a time-scale. We would like to compare our current optimal anomaly detection model with this CNN model and to improve the unsupervised learning method further.

In addition, since we only have a small sample of validation data labelled by two sales representatives, there is not enough labelled data for us to do the semi-supervised or supervised training by using other models such as Long short-term memory (LSTM), recurrent neural network (RNN). Therefore, in the future, we hope to gather more labelled validation data in order to perform the supervised training.

Appendix

A

A. Python Code

```
def detect_anomalies(forecast):
    forecasted = forecast[['ds','trend', 'yhat', 'yhat_lower', '
    yhat_upper', 'fact']].copy()

    forecasted['anomaly'] = 0
    forecasted.loc[forecasted['fact'] > forecasted['yhat_upper'],
     'anomaly'] = 1
    forecasted.loc[forecasted['fact'] < forecasted['yhat_lower'],
     'anomaly'] = -1
    #anomaly importances
    forecasted['importance'] = 0
    interval_range = forecasted['yhat_upper'] - forecasted['
    yhat_lower']
    # calculate importance of anomalies
    forecasted.loc[forecasted['anomaly'] ==1, 'importance'] = \
        (forecasted['fact'] - forecasted['yhat_upper'])*0.5/
    interval_range
    forecasted.loc[forecasted['anomaly'] ==-1, 'importance'] = \
        (forecasted['yhat_lower'] - forecasted['fact'])*0.7/
    interval_range

    return forecasted
```

Listing A.1: Anomaly Detection

```
def weekly_abn(output, product_cat2):
  test_filter = (output.PRODUCT_CATEGORY_2 == product_cat2)
  df_new = output.loc[test_filter, output.columns].reset_index(
    drop=True)

  week_num = []
  year_num = []
  # add week & year columns
  for i in df_new.SALES_ORDER_DATE:
      week_num.append(i.week)
      if (i.week == 1) & (i.month == 12):
        year_num.append(i.year+1)
      else:
        year_num.append(i.year)
```

```
    df_new['WEEK_NUM'] = week_num
    df_new['YEAR_NUM'] = year_num

    # get the empty weeks
    empty_week = df_new[df_new['ORDERED_VALUE_EUR_SUM'] <= 0]
    # remove 0 ordered value lines
    df_new = df_new.drop(np.where(df_new['ORDERED_VALUE_EUR_SUM']
     <= 0)[0]).reset_index(drop=True)
    df_prop = df_new.copy()
    df_prop = df_prop.rename(columns={'SALES_ORDER_DATE':'ds','
     ORDERED_VALUE_EUR_SUM':'y'})
    # unsupervised anomaly detection model
    m = Prophet(
        daily_seasonality=False,
        yearly_seasonality=True,
        weekly_seasonality=False,
        seasonality_mode='additive',
        changepoint_prior_scale=0.05,
        seasonality_prior_scale=0.01,
        changepoint_range=0.8,
        holidays_prior_scale=10,
        interval_width=0.95
        )
    # add monthly seasonality
    m.add_seasonality(name='monthly', period=30.5, fourier_order=3)
    m.add_country_holidays(country_name='UK')
    start_time = time.time()
    m.fit(df_prop)
    runtime = time.time() - start_time

    prop_pred = m.predict(df_prop[['ds','y']])
    prop_pred['fact'] = df_prop['y']
    for i in prop_pred[prop_pred.yhat_lower < 0].yhat_lower.index:
        prop_pred['yhat_lower'][i] = 0

    new_prop = prop_pred[['ds','fact', 'trend', 'yhat', 'yhat_lower
     ', 'yhat_upper']]
    detect_anb = detect_anomalies(new_prop).drop('ds',axis=1)
    df_detect = pd.concat([df_prop.reset_index(drop=True),
     detect_anb], axis=1)
    # return the detection result with importance and reasons
    Abn_Value = week_value_detection(df_detect)

    return Abn_Value, df_detect
```

Listing A.2: Anomaly Detection For Weekly Data

B. Training Plots and Tables

AutoReg Model Results

Dep. Variable:	y	No. Observations:	249
Model:	Seas. AutoReg(7)	Log Likelihood	-339.490
Method:	Conditional MLE	S.D. of innovations	0.984
Date:	Thu, 08 Jul 2021	AIC	0.282
Time:	00:05:24	BIC	0.830
Sample:	7	HQIC	0.503
	249		

	coef	std err	z	P>|z|	[0.025	0.975]
intercept	5.0287	1.284	3.917	0.000	2.512	7.545
seasonal.1	0.1040	0.501	0.208	0.835	-0.877	1.085
seasonal.2	0.7488	0.497	1.508	0.132	-0.225	1.722
seasonal.3	0.0098	0.497	0.020	0.984	-0.964	0.984
seasonal.4	0.8481	0.497	1.708	0.088	-0.125	1.821
seasonal.5	-0.4281	0.498	-0.860	0.390	-1.404	0.548
seasonal.6	0.0182	0.504	0.036	0.971	-0.970	1.006
seasonal.7	0.9078	0.486	1.869	0.062	-0.044	1.860
seasonal.8	0.6383	0.490	1.303	0.193	-0.322	1.598
seasonal.9	0.0474	0.497	0.096	0.924	-0.926	1.021
seasonal.10	0.0084	0.507	0.017	0.987	-0.984	1.001
seasonal.11	0.5007	0.504	0.993	0.321	-0.487	1.489
seasonal.12	-0.1582	0.497	-0.318	0.750	-1.132	0.816
seasonal.13	0.5838	0.502	1.163	0.245	-0.400	1.568
seasonal.14	0.7999	0.499	1.604	0.109	-0.178	1.777
seasonal.15	-0.1503	0.497	-0.303	0.762	-1.124	0.823
seasonal.16	0.5042	0.498	1.012	0.311	-0.472	1.480
seasonal.17	-0.1886	0.501	-0.376	0.707	-1.170	0.793
seasonal.18	0.5480	0.501	1.094	0.274	-0.434	1.530
seasonal.19	0.1327	0.498	0.267	0.790	-0.842	1.108
seasonal.20	0.2909	0.503	0.578	0.563	-0.696	1.278
seasonal.21	-0.0869	0.495	-0.176	0.861	-1.057	0.883
seasonal.22	0.4927	0.501	0.983	0.326	-0.490	1.475
seasonal.23	0.2882	0.494	0.584	0.559	-0.679	1.255
seasonal.24	-0.1370	0.498	-0.275	0.783	-1.112	0.838
seasonal.25	0.5229	0.497	1.053	0.292	-0.451	1.496
seasonal.26	-0.0423	0.497	-0.085	0.932	-1.016	0.931

seasonal.27	0.3073	0.497	0.619	0.536	-0.666	1.281
seasonal.28	0.2867	0.496	0.578	0.563	-0.685	1.259
seasonal.29	0.7614	0.498	1.528	0.126	-0.215	1.738
y.L1	0.1110	0.064	1.735	0.083	-0.014	0.236
y.L2	0.1116	0.064	1.751	0.080	-0.013	0.237
y.L3	0.0788	0.065	1.221	0.222	-0.048	0.205
y.L4	-0.0522	0.065	-0.805	0.421	-0.179	0.075
y.L5	0.0203	0.065	0.312	0.755	-0.107	0.148
y.L6	0.0943	0.065	1.462	0.144	-0.032	0.221
y.L7	0.0859	0.065	1.331	0.183	-0.041	0.212

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	1.2099	-0.0000j	1.2099	-0.0000
AR.2	0.8262	-1.0483j	1.3347	-0.1438
AR.3	0.8262	+1.0483j	1.3347	0.1438
AR.4	-0.5439	-1.3539j	1.4590	-0.3108
AR.5	-0.5439	+1.3539j	1.4590	0.3108
AR.6	-1.4361	-0.6890j	1.5929	-0.4288
AR.7	-1.4361	+0.6890j	1.5929	0.4288

Table A.1.: Grid Search Performance Results for Daily Data

Changepoint Prior Scale	Holidays Prior Scale	Seasonality Prior Scale	MSE	RMSE	MAE	MAPE	MDAPE	Coverage	Runtime
0.05	10.0	0.01	1.4455	1.2023	0.9689	0.1043	0.0859	0.8589	10.67
0.05	0.1	0.01	1.5141	1.2305	0.9931	0.1066	0.089	0.8506	11.0
0.05	1.0	0.01	1.5271	1.2358	0.9966	0.1071	0.089	0.8465	11.06
0.05	0.01	0.01	1.5473	1.2439	1.0059	0.1081	0.0894	0.8506	10.52
0.1	0.01	0.01	1.5481	1.2442	1.0031	0.1078	0.0882	0.8506	10.68
0.1	1.0	0.01	1.5507	1.2453	1.0043	0.1078	0.0897	0.8382	10.79
0.1	10.0	0.01	1.5668	1.2517	1.0095	0.1084	0.0897	0.8382	11.24
0.1	0.1	0.01	1.5737	1.2545	1.0133	0.1088	0.0907	0.8465	11.05
0.01	10.0	0.01	1.7613	1.3271	1.0642	0.1153	0.0882	0.8465	10.44
0.01	0.01	0.01	1.7627	1.3277	1.0647	0.1154	0.0881	0.8465	10.55
0.01	0.1	0.01	1.7653	1.3286	1.0645	0.1154	0.088	0.8423	10.21
0.01	1.0	0.01	1.7656	1.3288	1.0649	0.1155	0.0882	0.8423	10.16
0.5	0.1	0.1	2.3364	1.5285	1.2278	0.1306	0.1087	0.7593	11.5
0.05	0.01	0.1	2.3411	1.5301	1.2335	0.1328	0.1163	0.751	10.45
0.5	10.0	0.01	2.3473	1.5321	1.2222	0.1312	0.1052	0.7552	11.28
0.5	0.01	0.1	2.351	1.5333	1.2304	0.131	0.1069	0.751	11.16
0.5	1.0	0.01	2.3647	1.5378	1.226	0.1317	0.1038	0.7386	11.38
0.5	0.1	0.01	2.3658	1.5381	1.2273	0.1318	0.1037	0.7427	11.53
0.5	0.01	0.01	2.3792	1.5425	1.2323	0.1322	0.1068	0.7427	11.52
0.1	0.01	0.1	2.3903	1.5461	1.2456	0.1342	0.1145	0.7427	10.26
0.05	0.1	0.1	2.4142	1.5538	1.2521	0.1346	0.1167	0.7261	10.52
0.1	0.1	0.1	2.4184	1.5551	1.2525	0.1344	0.1111	0.7178	10.36
0.05	10.0	0.1	2.4331	1.5598	1.2555	0.1348	0.1168	0.7303	10.26
0.1	10.0	0.1	2.4571	1.5675	1.2635	0.1354	0.1089	0.7303	10.63
0.05	1.0	0.1	2.4703	1.5717	1.2643	0.1356	0.1166	0.7261	10.51
0.1	1.0	0.1	2.5248	1.5889	1.276	0.1374	0.1131	0.7303	10.26
0.5	1.0	0.1	2.5864	1.6082	1.2966	0.1387	0.1114	0.722	10.89
0.5	10.0	0.1	2.592	1.61	1.2976	0.1388	0.1106	0.722	10.84
0.01	0.1	0.1	2.7566	1.6603	1.3505	0.1466	0.1248	0.7635	10.11
0.01	1.0	0.1	2.9004	1.703	1.3922	0.1505	0.1298	0.7427	10.16
0.01	0.01	0.1	2.9369	1.7137	1.4024	0.1517	0.1298	0.7344	10.06
0.01	10.0	0.1	2.9628	1.7213	1.4175	0.1534	0.1326	0.7261	10.17
0.05	0.01	1.0	5.098	2.2579	1.7784	0.1883	0.1487	0.6017	10.84
0.01	0.01	1.0	5.4701	2.3388	1.9132	0.2041	0.1609	0.5934	10.25
0.05	0.1	1.0	5.8348	2.4155	1.8752	0.1985	0.153	0.5685	10.73
0.1	0.01	1.0	5.9125	2.4316	1.8966	0.1993	0.1552	0.5726	10.41
0.01	0.1	1.0	5.9297	2.4351	1.9292	0.2058	0.157	0.6224	10.11
0.05	10.0	1.0	6.3252	2.515	1.9025	0.2009	0.1515	0.5768	10.35
0.1	0.1	1.0	6.5282	2.555	2.0074	0.2123	0.1692	0.527	10.39
0.05	1.0	1.0	6.5605	2.5613	1.9483	0.2058	0.161	0.5768	10.62
0.01	1.0	1.0	6.5924	2.5676	1.9787	0.2106	0.1551	0.639	10.21
0.01	10.0	1.0	6.6072	2.5704	1.9793	0.2107	0.1551	0.639	10.39
0.1	10.0	1.0	7.3458	2.7103	2.0721	0.2185	0.1735	0.5394	10.57
0.1	1.0	1.0	7.8405	2.8001	2.1719	0.2285	0.18	0.5228	10.56
0.5	0.01	1.0	85.6819	9.2565	6.1104	0.6432	0.3307	0.3029	10.37
0.5	1.0	1.0	93.2215	9.6551	6.5982	0.6931	0.3788	0.278	10.53
0.5	0.1	1.0	100.1222	10.0061	6.5086	0.6848	0.3344	0.3071	10.64
0.5	10.0	1.0	105.1307	10.2533	6.7218	0.7072	0.3638	0.3112	11.34
0.01	0.01	10.0	207.1187	14.3916	6.5962	0.6792	0.1753	0.5394	12.29
0.05	0.01	10.0	268.7251	16.3928	6.9886	0.7146	0.1617	0.5643	10.5
0.1	0.01	10.0	276.3333	16.6233	7.1135	0.7276	0.1718	0.5311	10.5
0.01	0.1	10.0	351.9445	18.7602	8.0914	0.8294	0.1801	0.5519	10.03
0.1	0.1	10.0	356.1351	18.8715	8.3951	0.8581	0.1925	0.4979	10.3
0.05	0.1	10.0	363.6006	19.0683	8.051	0.8235	0.1712	0.5394	10.62
0.01	1.0	10.0	377.4384	19.4278	8.5427	0.8798	0.167	0.5934	10.2
0.05	1.0	10.0	395.1735	19.879	8.5655	0.8794	0.1667	0.5519	10.15
0.1	1.0	10.0	396.9258	19.923	8.8869	0.913	0.2164	0.4647	10.3
0.05	10.0	10.0	397.4212	19.9354	8.5859	0.8817	0.1664	0.5519	10.19
0.1	10.0	10.0	399.8304	19.9958	8.9579	0.9195	0.206	0.4813	10.42
0.01	10.0	10.0	411.6025	20.288	8.7916	0.9035	0.1812	0.556	14.25
0.5	0.01	10.0	470.0088	21.6797	13.3727	1.3837	0.5805	0.2656	10.1
0.5	0.1	10.0	513.7011	22.665	14.2724	1.4777	0.6422	0.2573	10.47
0.5	1.0	10.0	536.6881	23.1665	14.5217	1.5034	0.6538	0.2614	10.58
0.5	10.0	10.0	571.7395	23.9111	15.1599	1.5725	0.6727	0.2656	10.38

Table A.2.: Grid Search Performance Results for Weekly Data

Changepoint Prior Scale	Holidays Prior Scale	Seasonality Prior Scale	MSE	RMSE	MAE	MAPE	MDAPE	Coverage	Runtime
0.1	0.01	0.01	1.5591	1.2486	0.9642	0.0912	0.0742	0.5588	13.1
0.05	0.01	0.01	1.5971	1.2638	1.0175	0.0954	0.0788	0.5882	12.97
0.01	0.01	0.01	1.6	1.2649	1.0182	0.0955	0.0793	0.5882	12.38
0.01	0.1	0.01	2.0049	1.4159	1.1098	0.1051	0.0831	0.3088	12.15
0.05	0.1	0.01	2.0161	1.4199	1.1139	0.1055	0.083	0.3088	12.89
0.01	10.0	0.01	2.0273	1.4238	1.1125	0.1054	0.0799	0.2941	11.69
0.01	1.0	0.01	2.0321	1.4255	1.114	0.1056	0.0801	0.2941	12.19
0.05	10.0	0.01	2.034	1.4262	1.1145	0.1056	0.0805	0.2941	12.6
0.05	1.0	0.01	2.0387	1.4278	1.1164	0.1058	0.0805	0.2941	13.51
0.1	0.1	0.01	2.4782	1.5742	1.2035	0.1132	0.0957	0.3382	13.78
0.1	10.0	0.01	2.4862	1.5768	1.2008	0.1129	0.0978	0.3382	13.78
0.1	1.0	0.01	2.4868	1.5769	1.201	0.1129	0.098	0.3382	14.64
0.5	0.01	0.01	5.3381	2.3104	1.8022	0.1661	0.1347	0.5	51.93
0.5	0.01	0.1	5.6613	2.3794	1.9387	0.1762	0.1381	0.3088	50.72
0.5	10.0	0.01	5.7102	2.3896	1.8307	0.1707	0.1253	0.5147	52.94
0.5	0.1	0.01	5.7466	2.3972	1.8455	0.1719	0.125	0.4706	53.99
0.5	1.0	0.01	7.7872	2.7905	2.0543	0.1907	0.1299	0.5147	54.19
0.1	0.01	0.1	8.949	2.9915	2.262	0.2027	0.144	0.2647	48.79
0.01	0.01	0.1	12.1213	3.4816	2.3231	0.2115	0.1343	0.25	12.24
0.05	0.01	0.1	12.2148	3.495	2.4753	0.2211	0.1401	0.2794	17.64
0.5	10.0	0.1	13.2388	3.6385	2.7321	0.251	0.1823	0.1912	51.6
0.5	1.0	0.1	13.3167	3.6492	2.7584	0.2506	0.1793	0.1912	51.6
0.1	10.0	0.1	15.2068	3.8996	3.1195	0.2822	0.2015	0.0441	49.95
0.5	0.1	0.1	15.4924	3.936	3.0565	0.278	0.201	0.1618	50.95
0.1	0.1	0.1	15.8841	3.9855	3.1831	0.2875	0.2019	0.0441	50.34
0.05	10.0	0.1	16.06	4.0075	3.1536	0.2861	0.1949	0.0588	22.91
0.05	1.0	0.1	16.1326	4.0165	3.1633	0.2873	0.1921	0.0588	15.26
0.1	1.0	0.1	16.9646	4.1188	3.2651	0.294	0.2039	0.0441	50.72
0.05	0.1	0.1	18.119	4.2566	3.3256	0.3005	0.198	0.0735	19.43
0.01	0.1	0.1	19.4584	4.4112	3.4012	0.3089	0.2012	0.0441	12.32
0.01	1.0	0.1	20.5991	4.5386	3.4293	0.3088	0.1995	0.0441	12.33
0.01	10.0	0.1	21.069	4.5901	3.5056	0.316	0.1958	0.0441	13.91
0.1	10.0	1.0	527.5081	22.9675	17.1499	1.5298	1.2333	0.0294	49.73
0.1	1.0	1.0	636.9448	25.2378	18.5738	1.6571	1.426	0.0294	52.6
0.05	0.1	1.0	642.8104	25.3537	19.2657	1.7218	1.5737	0.0441	48.57
0.1	0.1	1.0	727.5591	26.9733	20.5511	1.853	1.6096	0.0294	50.16
0.1	0.01	1.0	757.2583	27.5183	18.6167	1.6772	1.019	0.1912	48.91
0.05	1.0	1.0	780.6468	27.9401	20.1725	1.7855	1.5672	0.0588	49.25
0.01	1.0	1.0	878.0198	29.6314	21.5925	1.9189	1.642	0.0294	12.23
0.05	10.0	1.0	887.4243	29.7897	21.2581	1.9107	1.6148	0.0294	14.04
0.5	0.1	1.0	950.3293	30.8274	22.9503	2.0665	1.6129	0.0882	54.0
0.05	0.01	1.0	1030.3539	32.0991	22.0905	2.0081	1.276	0.1324	14.1
0.5	0.01	1.0	1124.8984	33.5395	23.0679	2.0545	1.4386	0.1765	51.38
0.01	0.1	1.0	1352.734	36.7795	24.9236	2.1849	1.6027	0.0441	48.42
0.5	1.0	1.0	1375.2986	37.085	25.3766	2.3791	1.614	0.0441	55.23
0.5	10.0	1.0	1383.2789	37.1925	26.4114	2.4357	1.6211	0.0735	102.53
0.01	0.01	1.0	1485.763	38.5456	25.4684	2.2767	1.4399	0.1471	49.44
0.01	10.0	1.0	2549.3814	50.4914	30.3573	2.7182	1.719	0.0294	15.18
0.1	1.0	10.0	10147.2748	100.7337	64.4038	5.8501	2.0374	0.0441	52.12
0.05	10.0	10.0	10255.0617	101.2673	66.7488	6.0517	3.4872	0.0294	48.5
0.1	10.0	10.0	10369.2322	101.8294	68.7206	6.2311	3.7769	0.0294	50.7
0.05	1.0	10.0	10505.3359	102.4955	69.0249	6.2515	4.065	0.0294	13.94
0.01	1.0	10.0	10890.1883	104.3561	71.1375	6.4636	4.93	0.0294	12.51
0.5	10.0	10.0	11688.1171	108.1116	71.9147	6.5344	4.6578	0.0294	55.01
0.1	0.1	10.0	12042.8786	109.7401	76.7218	6.9319	5.7164	0.0294	50.42
0.01	0.1	10.0	14343.3898	119.7639	82.2462	7.508	5.6333	0.0294	13.1
0.01	10.0	10.0	17775.5382	133.3249	92.02	8.361	6.5728	0.0294	13.81
0.5	0.1	10.0	21790.4637	147.6159	91.481	8.5439	5.1351	0.0441	59.98
0.1	0.01	10.0	34917.7503	186.8629	98.8713	9.1879	3.4152	0.1324	49.58
0.05	0.01	10.0	35088.131	187.3183	99.1123	9.1845	3.3766	0.1618	12.69
0.01	0.01	10.0	35376.6309	188.0868	102.9723	9.5784	3.4613	0.1324	19.49
0.05	0.1	10.0	36635.8401	191.4049	118.6949	10.9211	6.5818	0.0294	14.85
0.5	0.01	10.0	37739.237	194.2659	92.1178	8.6914	2.0408	0.2059	59.92
0.5	1.0	10.0	87253.6528	295.3873	145.7674	14.0736	4.215	0.0588	57.98

Table A.3.: Grid Search Performance Results for Monthly Data

Changepoint Prior Scale	Holidays Prior Scale	Seasonality Prior Scale	MSE	RMSE	MAE	MAPE	MDAPE	Coverage	Runtime
0.1	0.01	0.1	0.7943	0.8912	0.7279	0.059	0.039	0.0	28.42
0.5	0.01	0.1	1.3462	1.1603	0.9659	0.078	0.0573	0.05	87.98
0.5	0.1	0.1	1.4432	1.2013	0.9846	0.0786	0.067	0.2	89.58
0.05	1.0	0.1	1.5146	1.2307	0.9754	0.079	0.0594	0.0	47.49
0.01	0.01	0.01	1.5626	1.25	1.0713	0.086	0.0687	0.0	12.64
0.5	0.01	0.01	1.635	1.2787	1.0376	0.083	0.0537	0.4	100.32
0.5	10.0	0.01	1.6622	1.2893	1.0917	0.0872	0.0634	0.35	133.8
0.05	0.01	0.01	1.6954	1.3021	1.1189	0.0896	0.0724	0.0	32.62
0.01	0.1	0.01	1.7075	1.3067	1.1154	0.0898	0.0756	0.0	14.37
0.05	0.1	0.01	1.7174	1.3105	1.1252	0.0903	0.0778	0.0	31.98
0.01	10.0	0.01	1.7278	1.3145	1.1305	0.0908	0.0774	0.0	14.69
0.05	10.0	0.01	1.7386	1.3186	1.1337	0.091	0.0778	0.0	27.35
0.05	1.0	0.01	1.7509	1.3232	1.1354	0.0911	0.0782	0.0	24.91
0.5	0.1	0.01	1.7694	1.3302	1.117	0.0892	0.0653	0.35	132.87
0.01	1.0	0.01	1.7796	1.334	1.1511	0.0924	0.0756	0.0	13.8
0.01	0.01	0.1	1.7807	1.3344	1.1316	0.0909	0.0757	0.0	16.01
0.5	1.0	0.01	1.8094	1.3451	1.1441	0.0913	0.0711	0.35	115.56
0.1	0.01	0.01	1.8875	1.3738	1.2021	0.0961	0.0876	0.0	61.84
0.1	10.0	0.01	1.9009	1.3787	1.1868	0.095	0.0858	0.05	79.49
0.1	0.1	0.01	1.9093	1.3818	1.1922	0.0954	0.0853	0.05	76.82
0.1	1.0	0.01	1.9322	1.39	1.1984	0.0959	0.0863	0.05	72.83
0.5	10.0	0.1	2.0164	1.42	1.1856	0.0962	0.0813	0.0	122.66
0.01	0.1	0.1	2.2354	1.4951	1.1499	0.0938	0.0646	0.0	16.45
0.5	1.0	0.1	2.363	1.5372	1.298	0.1041	0.0918	0.05	101.24
0.1	10.0	0.1	2.4783	1.5743	1.3319	0.1067	0.0833	0.0	59.42
0.05	0.1	0.1	2.5095	1.5841	1.2798	0.1033	0.0768	0.0	48.84
0.1	1.0	0.1	2.551	1.5972	1.3302	0.1074	0.0921	0.0	58.38
0.05	10.0	0.1	2.5746	1.6046	1.32	0.105	0.0805	0.0	22.65
0.01	1.0	0.1	2.579	1.6059	1.3683	0.1095	0.0915	0.0	16.86
0.01	10.0	0.1	2.7068	1.6452	1.4232	0.1138	0.1048	0.0	16.0
0.1	0.1	0.1	2.7337	1.6534	1.4732	0.1179	0.0968	0.0	49.01
0.05	0.01	0.1	2.8642	1.6924	1.4267	0.1138	0.0851	0.0	22.46
0.1	0.1	1.0	208.0027	14.4223	11.711	0.9273	0.9889	0.0	43.7
0.5	1.0	1.0	351.582	18.7505	16.1105	1.2646	1.0739	0.0	109.25
0.5	0.1	1.0	404.5528	20.1135	15.1492	1.2042	0.9987	0.05	90.62
0.1	10.0	1.0	498.5677	22.3286	16.153	1.285	0.6716	0.1	68.17
0.5	10.0	1.0	534.1418	23.1115	18.2524	1.4829	1.1663	0.05	127.16
0.1	1.0	1.0	593.102	24.3537	19.4353	1.5465	1.1685	0.0	50.66
0.05	0.1	1.0	604.503	24.5866	19.1085	1.5194	1.3566	0.0	16.36
0.5	0.01	1.0	709.3814	26.6342	18.4875	1.4757	1.061	0.05	81.09
0.05	1.0	1.0	888.5921	29.8093	18.9843	1.5054	0.7805	0.0	14.16
0.01	0.01	1.0	906.8947	30.1147	19.5468	1.5534	1.2212	0.0	12.33
0.01	1.0	1.0	929.6245	30.4897	18.8232	1.48	0.9143	0.0	12.46
0.05	10.0	1.0	946.3264	30.7624	21.9092	1.7768	1.2597	0.0	30.86
0.05	0.01	1.0	1020.9509	31.9523	24.1085	1.9038	1.374	0.0	20.14
0.1	0.01	1.0	1055.1698	32.4834	21.6756	1.7246	0.8876	0.0	40.52
0.01	0.1	1.0	1517.3268	38.9529	29.2268	2.2946	2.3084	0.0	12.97
0.01	10.0	1.0	3181.703	56.4066	42.3688	3.4025	2.0553	0.0	42.74
0.5	0.1	10.0	9884.5049	99.4208	71.5861	5.6937	3.8603	0.0	94.94
0.1	0.01	10.0	10670.0023	103.2957	76.5094	6.2105	4.5318	0.0	21.83
0.05	0.01	10.0	12898.8021	113.5729	84.2855	6.6963	4.5776	0.0	14.68
0.5	1.0	10.0	13348.3784	115.5352	70.0621	5.4873	1.8417	0.05	130.22
0.5	0.01	10.0	17812.0614	133.4618	99.2043	7.9114	5.3313	0.0	90.4
0.1	1.0	10.0	19570.4902	139.8946	90.8144	7.4798	4.3841	0.0	61.69
0.01	10.0	10.0	20136.8019	141.9042	105.899	8.4871	5.2828	0.0	31.53
0.01	0.1	10.0	20967.2534	144.8007	102.9399	8.3023	6.677	0.0	14.2
0.1	10.0	10.0	21423.7609	146.3686	109.1792	8.5352	6.8439	0.0	43.43
0.1	0.1	10.0	22370.9742	149.5693	104.4122	8.2065	6.5063	0.0	72.6
0.05	0.1	10.0	23284.5184	152.5927	104.8498	8.4475	5.3665	0.0	32.61
0.05	1.0	10.0	24156.98	155.4252	107.4946	8.5323	5.3078	0.0	16.01
0.05	10.0	10.0	45510.0224	213.3308	144.4524	11.6052	8.0889	0.05	68.71
0.01	0.01	10.0	45530.2625	213.3782	141.4875	11.3895	7.5303	0.0	12.1
0.01	1.0	10.0	52580.6301	229.3047	151.081	11.8729	7.3663	0.0	17.54
0.5	10.0	10.0	63213.7482	251.4234	176.0804	14.276	7.2534	0.0	129.12

Table A.4.: Bimodal Distribution Parameters and Standard Deviation Errors (Sigma) Summary for 10 Products

	μ_1	σ_1	A_1	μ_2	σ_2	A_2
params PE85	7.011393	0.52592	0.309155	10.094075	0.90503	0.247172
sigma PE85	0.045694	0.046411	0.022837	0.074418	0.07709	0.017548
params PP50	6.821975	0.57451	0.247063	10.278292	1.012411	0.248694
sigma PP50	0.060565	0.061389	0.022187	0.079381	0.082477	0.01687
params PP60	6.823961	0.589295	0.244937	10.20956	1.007424	0.249258
sigma PP60	0.078975	0.080764	0.027824	0.100557	0.104694	0.021447
params Thermal ECO	7.154781	0.350253	0.414061	10.346966	1.269719	0.181288
sigma Thermal ECO	0.036625	0.037263	0.036955	0.159829	0.169374	0.019613
params TT Uncoated	7.110139	0.287267	0.374780	10.060893	0.962612	0.277377
sigma TT Uncoated	0.031390	0.031495	0.035274	0.077458	0.078631	0.019350
params Decorative Uncoated	6.317687	0.607491	0.342194	9.680521	0.768912	0.245010
sigma Decorative Uncoated	0.119432	0.129495	0.057342	0.180932	0.182589	0.049929
params MC	7.068254	0.458789	0.424382	10.419040	1.002428	0.181639
sigma MC	0.033547	0.033760	0.026627	0.115340	0.117598	0.018107
params Special Film Other	6.811508	0.525339	0.317501	10.453161	1.270285	0.176336
sigma Special Film Other	0.114210	0.115579	0.057695	0.331811	0.393028	0.038951
params Decorative Coated	6.323099	0.552525	0.235768	10.030882	1.219497	0.230721
sigma Decorative Coated	0.120144	0.128447	0.043172	0.180168	0.198827	0.029454
params Thermal Top	6.934644	0.540794	0.365774	10.391814	0.968904	0.193375
sigma Thermal Top	0.052263	0.052753	0.030360	0.131786	0.134407	0.022779

References

B

Literature

[1] Avery Dennison. *Innovative Packaging Materials Solutions and Technologies: Avery Dennison*. `https://www.averydennison.com/en/home.html`. Accessed: 2020-11-25 (cit. on pp. ii, 1, 25).

[2] Alan R Hevner. „A three cycle view of design science research“. In: *Scandinavian journal of information systems* 19.2 (2007), p. 4 (cit. on pp. 1, 5, 6).

[3] Iwan Syarif, Adam Prugel-Bennett, and Gary Wills. „Unsupervised clustering approach for network anomaly detection“. In: *International conference on networked digital technologies*. Springer. 2012, pp. 135–145 (cit. on p. 3).

[4] K. Sheridan, Tejas G. Puranik, Eugene Mangortey, et al. „An Application of DBSCAN Clustering for Flight Anomaly Detection During the Approach Phase“. In: 2020 (cit. on pp. 3, 14).

[5] Holger Dette, Weichi Wu, and Zhou Zhou. „Change point analysis of correlation in non-stationary time series“. In: *Statistica Sinica* 29.2 (2019), pp. 611–643 (cit. on p. 3).

[6] Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. „DeepAnT: A Deep Learning Approach for Unsupervised Anomaly Detection in Time Series“. In: *IEEE Access* 7 (2019), pp. 1991–2005 (cit. on pp. 3, 64).

[7] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. „Usad: Unsupervised anomaly detection on multivariate time series“. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020, pp. 3395–3404 (cit. on p. 3).

[8] Bo Zong, Qi Song, Martin Renqiang Min, et al. „Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection“. In: *ICLR*. 2018 (cit. on pp. 4, 64).

[9] Mohammad Braei and Sebastian Wagner. „Anomaly detection in univariate time-series: A survey on the state-of-the-art“. In: *arXiv preprint arXiv:2004.00433* (2020) (cit. on p. 4).

[10] Juhani Iivari. „A paradigmatic analysis of information systems as a design science“. In: *Scandinavian Journal of Information Systems* 19 (Jan. 2007), pp. 39– (cit. on pp. 5, 7).

[11] Juhani Iivari. „Towards information systems as a science of meta-artifacts“. In: *Communications of the Association for Information Systems* 12 (Jan. 2003), pp. 568–581 (cit. on p. 7).

[12] Shin-Yuan Hung, David C Yen, and Hsiu-Yu Wang. „Applying data mining to telecom churn management“. In: *Expert Systems with Applications* 31.3 (2006), pp. 515–524 (cit. on p. 7).

[13] Emir Zunic, Kemal Korjenic, Kerim Hodzic, and Dzenana Donko. „Application of Facebook's Prophet Algorithm for Successful Sales Forecasting Based on Real-world Data“. In: *arXiv preprint arXiv:2005.07575* (2020) (cit. on pp. 7, 18).

[14] J. Cryer. „Time Series Analysis“. In: 1986 (cit. on p. 7).

[15] P. Young and S. Shellswell. „Time series analysis, forecasting and control“. In: *IEEE Transactions on Automatic Control* 17 (1972), pp. 281–283 (cit. on p. 8).

[16] N. Huang, Zheng Shen, S. Long, et al. „The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis“. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 454 (1998), pp. 903 –995 (cit. on p. 9).

[17] Cosimo Magazzino. „Wagner's Law and Augmented Wagner's Law in EU-27. A Time-Series Analysis on Stationarity, Cointegration and Causality“. In: *International Research Journal of Finance and Economics* 89 (Apr. 2012), pp. 205–220 (cit. on pp. 9, 17).

[18] L. Freitag, S. Knecht, C. Angeli, and M. Reiher. „Multireference Perturbation Theory with Cholesky Decomposition for the Density Matrix Renormalization Group“. In: *Journal of Chemical Theory and Computation* 13 (2017), pp. 451 –459 (cit. on p. 9).

[19] Xiaoning Kang and X. Deng. „An improved modified cholesky decomposition approach for precision matrix estimation“. In: *Journal of Statistical Computation and Simulation* 90 (2020), pp. 443 –464 (cit. on p. 9).

[20] H. Weinert. „Efficient computation for Whittaker-Henderson smoothing“. In: *Comput. Stat. Data Anal.* 52 (2007), pp. 959–974 (cit. on p. 10).

[21] R. Pascual-Marqui. „Instantaneous and lagged measurements of linear and nonlinear dependence between groups of multivariate time series: frequency decomposition“. In: *arXiv: Methodology* (2007) (cit. on p. 10).

[22] G. Chen, Min Gan, and Guo long Chen. „Generalized exponential autoregressive models for nonlinear time series: Stationarity, estimation and applications“. In: *Inf. Sci.* 438 (2018), pp. 46–57 (cit. on p. 10).

[23] B. Bercu and Fr'ed'eric Proia. „A sharp analysis on the asymptotic behavior of the Durbin-Watson statistic for the first-order autoregressive process“. In: *arXiv: Statistics Theory* (2011) (cit. on p. 11).

[24] P. Turner. „Critical values for the Durbin-Watson test in large samples“. In: *Applied Economics Letters* 27 (2019), pp. 1495 –1499 (cit. on p. 11).

[25] I. T. Jolliffe. „Principal Component Analysis“. In: 2002 (cit. on pp. 12, 13).

[26] Hervé Abdi and Lynne J Williams. „Principal component analysis“. In: *Wiley interdisciplinary reviews: computational statistics* 2.4 (2010), pp. 433–459 (cit. on p. 13).

[27] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. „A density-based algorithm for discovering clusters in large spatial databases with noise.“ In: *kdd*. Vol. 96. 34. 1996, pp. 226–231 (cit. on p. 14).

[28] Erich Schubert, J. Sander, M. Ester, H. Kriegel, and Xiaowei Xu. „DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN“. In: *ACM Trans. Database Syst.* 42 (2017), 19:1–19:21 (cit. on pp. 14–16).

[29] Sean J Taylor and Benjamin Letham. „Forecasting at scale“. In: *The American Statistician* 72.1 (2018), pp. 37–45 (cit. on pp. 16, 18–20, 22, 40).

[30] William B. Nicholson, Ines Wilms, J. Bien, and D. Matteson. „High Dimensional Forecasting via Interpretable Vector Autoregression“. In: *J. Mach. Learn. Res.* 21 (2020), 166:1–166:52 (cit. on p. 17).

[31] Serena Ng and P. Perron. „Lag Length Selection and the Construction of Unit Root Tests with Good Size and Power“. In: *Econometrica* 69 (2001), pp. 1519–1554 (cit. on p. 17).

[32] Michael W Robbins, C. Gallagher, and R. Lund. „A General Regression Changepoint Test for Time Series Data“. In: *Journal of the American Statistical Association* 111 (2016), pp. 670 –683 (cit. on p. 18).

[33] Lingjian Yang, Songsong Liu, S. Tsoka, and L. Papageorgiou. „Mathematical programming for piecewise linear regression analysis“. In: *Expert Syst. Appl.* 44 (2016), pp. 156–167 (cit. on p. 18).

[34] Matthias Feurer, Aaron Klein, Katharina Eggensperger, et al. „Efficient and robust automated machine learning“. In: *Advances in neural information processing systems*. 2015, pp. 2962–2970 (cit. on p. 22).

[35] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. *A practical guide to support vector classification*. 2003 (cit. on p. 23).

[36] Atefeh Daemi, H. Kodamana, and Biao Huang. „Gaussian process modelling with Gaussian mixture likelihood“. In: *Journal of Process Control* 81 (2019), pp. 209–220 (cit. on p. 24).

[37] S. Gautam. „IBM Cognos Business Intelligence v10: The Complete Guide“. In: 2012 (cit. on p. 25).

[38] Srikanth Thudumu, P. Branch, Jiong Jin, and Jugdutt Singh. „A comprehensive survey of anomaly detection techniques for high dimensional big data“. In: *Journal of Big Data* 7 (2020), pp. 1–30 (cit. on p. 25).

[39] M. Fujii and Akihiko Takahashi. „A Mean Field Game Approach to Equilibrium Pricing with Market Clearing Condition“. In: *ERN: Dynamic Stochastic General Equilibrium Models (Topic)* (2020) (cit. on p. 27).

[40] W. Manning and J. Mullahy. „Estimating Log Models: To Transform or Not to Transform?“ In: *NBER Working Paper Series* (2001) (cit. on p. 29).

[41] R. Hughes, J. Heron, J. Sterne, and K. Tilling. „Accounting for missing data in statistical analyses: multiple imputation is not always the answer“. In: *International Journal of Epidemiology* 48 (2019), pp. 1294 –1304 (cit. on p. 29).

[42] J. H. Steiger. „Tests for comparing elements of a correlation matrix.“ In: *Psychological Bulletin* 87 (1980), pp. 245–251 (cit. on p. 31).

[43] Jacob Cohen, P. Cohen, S. West, and L. Aiken. „Applied multiple regression/-correlation analysis for the behavioral sciences“. In: 1975 (cit. on p. 31).

[44] Choo-Yee Ting, Chiung Ching Ho, Hui-Jia Yee, and Wan Razali Matsah. „Geospatial Analytics in Retail Site Selection and Sales Prediction“. In: *Big data* 6 1 (2018), pp. 42–52 (cit. on p. 32).

[45] J. Little. „Integrated measures of sales, merchandising, and distribution“. In: *International Journal of Research in Marketing* 15 (1998), pp. 473–485 (cit. on p. 35).

[46] M. Boldin and M. N. Petriev. „On the Empirical Distribution Function of Residuals in Autoregression with Outliers and Pearson's Chi-Square Type Tests“. In: *Mathematical Methods of Statistics* 27 (2018), pp. 294–311 (cit. on p. 39).

[47] M. Pardo and R. Alonso. „A generalized Q–Q plot for longitudinal data“. In: *Journal of Applied Statistics* 39 (2012), pp. 2349 –2362 (cit. on p. 39).

[48] M. Skitmore and G. Runeson. „Bidding models: testing the stationarity assumption“. In: *Construction Management and Economics* 24 (2006), pp. 791 –803 (cit. on p. 39).

[49] Peter E. Rossi. „Bayesian Non- and Semi-parametric Methods and Applications“. In: 2014 (cit. on pp. 56, 57).

[50] David Arthur and Sergei Vassilvitskii. *k-means++: The advantages of careful seeding*. Tech. rep. Stanford, 2006.

[51] Zhexue Huang. „Extensions to the k-means algorithm for clustering large data sets with categorical values“. In: *Data mining and knowledge discovery* 2.3 (1998), pp. 283–304.

[52] Zhexue Huang and Michael K Ng. „A fuzzy k-modes algorithm for clustering categorical data“. In: *IEEE transactions on Fuzzy Systems* 7.4 (1999), pp. 446–452.

[53] Ching-Hsue Cheng and You-Shyang Chen. „Classifying the segmentation of customer value via RFM model and RS theory“. In: *Expert systems with applications* 36.3 (2009), pp. 4176–4184.

[54] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, et al. „Long-term recurrent convolutional networks for visual recognition and description“. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 2625–2634.

[55] P. McNicholas. „Mixture Model-Based Classification“. In: 2016.

[56] D. Reynolds. „Gaussian Mixture Models“. In: *Encyclopedia of Biometrics*. 2015.

[57] Leena Kalliovirta, Mika Meitz, and Pentti Saikkonen. „A Gaussian mixture autoregressive model for univariate time series“. In: *Journal of Time Series Analysis* 36.2 (2015), pp. 247–266.

Index

C

List of Figures

List of Tables