FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Sales Forecasting for SaaS

Joel Fernando da Costa Silva Coelho

Mestrado em Engenharia Informática e Computação

Supervisor: Ana Paula Rocha

April 6, 2022

Sales Forecasting for SaaS

Joel Fernando da Costa Silva Coelho

Mestrado em Engenharia Informática e Computação

April 6, 2022

Abstract

The evolution of technology through the years directly correlates with the rapid development of e-commerce (EC), which consists of buying and selling goods or services using an electronic network, primarily the internet. As a result of this, it is essential to adopt strategies to manage the inventory of a business, where sales forecasting plays a crucial role. Affected by many external and internal factors, e-commerce sales (ECS) are normally diversified with linear and non-linear characteristics, which makes it a challenge to develop a model that can predict ECS volume.

Traditional time series models are a type of prediction models that can tackle this problem, from which we can distinguish Seasonal Autoregressive Integrated Moving Average (SARIMA), Holt-Winters, and Prophet models. Furthermore, supervised machine learning models that classify data and predict outcomes are also to be explored and implemented, such as Support-vector machine (SVM), Logistic Regression, and Decisions Trees. Finally, deep learning models that are widely used in extracting high-level abstract features such as DeepAR and others provided by Keras and GLuonTS frameworks can also tackle sales forecasting problems. We aim to compare the use of traditional and deep learning models in the electronic commerce sales forecasting problem. A real dataset will be used to evaluate the proposed models in real scenarios efficiently. One additional challenge is to handle new information in real-time, which will be done by implementing a large-data process system that will be responsible for updating data and forecasting models.

In the end, only machine learning models were implemented with good forecasting results. Overall they produced good results. More work should be done in order to explore different types of models and also to speed up the process of the data treatment and modelling stages of this forecasting problem.

Keywords: Artificial Intelligence, Deep Learning, Machine Learning, Time-series models, Data Processing

Resumo

A evolução da tecnologia ao longo do tempo está diretamente relacionada com o rápido desenvolvimento na área do e-commerce (EC). Este tipo de comércio consiste na compra e venda de bens ou serviços na internet. Desta forma, como em qualquer tipo de negócio, é importante adotar estratégias, como a previsão de vendas. que facilitam a gestão de inventário de uma empresa. No entanto, há uma série de fatores internos e externos que afetam as vendas de ecommerce (ECS) .Adicionalmente, os dados do ECS apresentam características lineares e não lineares que elevam a dificuldade de desenvolver um modelo eficiente.

Os modelos tradicionais de series temporais podem ser utilizados para resolver o problema em questão, entre eles destacam-se: o Seasonal Autoregressive Integrated Moving Average (SARIMA), Holt-Winters e Prophet. Por outro lado, modelos de machine learning que classificam dados e prevêm resultados podem também ser explorados e implementados como por exmeplo: Support-vector machine (SVM), Logistic Regression e Decisions Trees. Por último, modelos de deep learning capazes de extrair dados com alto grau de generalização, como o DeepAR e outros fornecidos por frameworks como o Keras e GLuonTS podem também ser utilizados. Uma vez que o objetivo principal deste projeto é a comparação do desempenho de diferentes modelos de previsão de ECS, será usado um dataset real. Este dataset será usado para avaliar de forma eficiente os modelos propostos. Um desafio adicional passa por saber lidar com atualização dos dados em tempo real. Desta forma, será implementada uma plataforma de processamento de dados em larga escala que será responsável por atualizar os dados e os modelos implementados.

Em suma, apenas modelos de previsão de machine-learning foram implementados, obtendo bons resultado, no entanto, outros tipos de modelos de previsao. De salientar também, novas formas de como o tratamento, processo de dados e treino dos modelos são feitos devem ser exploradas de modo a que se possa obter melhores resultados.

Acknowledgements

I want to thank Professor Ana Paula Rocha for the opportunity presented and Jumpseller's team, especially Filipe Gonçalves, the company's supervisor, for all the feedback and guidance provided throughout this dissertation.

To all the people who have accompanied me since the first time I stepped into a college, I would like to thank you for the time that we shared, the memories that we created, the mutual help that we gave each other that have contributed to finish my journey as an academy student. Namely to the people that I met during my bachelor's degree in Instituto Superior de Engenharia do Porto and later on Faculdade de Engenharia da Universidade do Porto during my master's. I want to give a special thanks to a friend of mine that I have known for a long time, Susana Lima, and a "brother from another mother," Guilherme Sousa, who helped me tremendously in this journey.

To my family, I would like to thank my parents for always giving me the basics, the love, the affection, and care since the day that I was born, and the ones that have given me the necessary conditions to succeed well in life.

These and other people that I've met have contributed to having the mentality of never giving up and always giving my best in any condition.

Joel Fernando da Costa Silva Coelho

"Success is not final, failure is not fatal,
it is the courage to continue that counts."

Winston S. Churchill

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
ANN	Artificial Neural Network
AWS	Amazon Web Services
CNN	Convolutional Neural Network
CV	Coefficient of Variation
DFS	Deep Feature Synthesis
DL	Deep Learning
DNN	Deep Neural Network
EC	Ecommerce
ECS	E-commerce sales
ETL	Extract Transform and Load
FBNN	FeedBackward Neural Network
FFNN	FeedForward Neural Network
GNB	Gaussian Naive Bayes
HW	Holt-Winter's
Light GBM	Light Gradient Boosting Machine
LSTM	Long Short Term Memory
LR	Logistic Regression
MAE	Mean Absolute Error
MAD	Mean Absolute Deviation
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MRE	Mean Relative Error
MSE	Mean Square Error
NA	Not Available
NaN	Not an Number
NN	Neural Network
PFE	Percentage Forecast Error
R^2	Root Mean Square Error
RF	Random Forest
RFE	Recursive Feature Elimination
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
SARIMAX	Seasonal Autoregressive Integrated Moving Average Exogenous Variables
SFS	Sequential Feature Selector
SFFS	Sequential Forward Floating Selection
SQL	Structured Query Language
SVM	Support Vector Machines
XGBoost	Extreme Gradient Boosting

Chapter 1

Introduction

1.1 Context

E-commerce (EC) consists of the sale of goods or services on the internet, introduced in 1979, the early internet days. Since then, there have been improvements in internet security and connectivity, the development of secure and accessible payment gateways, widespread consumer and business adoption, and lately, lockdowns, travel bans and retail closures have contributed to this growing type commerce. As a result of this growth, inventory management is crucial for any business to free up capital [38].

Sales forecasting is the prediction of future sales revenue. Despite being more accurate for short and medium shelf-life products, it can help a business reduce its sales and product returns, which negatively impact a retail chain. Given the current commerce conditions, there are a lot of internal (known to the retailer), and external factors that affect sales and demand: [7]

- **Internal factors** - Promotions, Discounts, Cannibalization, Organization Structure, Profit policy
- **External factors** - Competition, Supply (if applicable), Seasonality, Inflation rate, Abnormal events (pandemics, health crisis), Product need

These factors can contribute to under-stocking, where customers cannot buy the product they are searching for, or over-stocking, which leads to a loss of the company's revenue and increases product's waste, storage cost and maintenance [48]. Therefore, these factors are the main reason why sales forecasting is a crucial topic in commerce in general and also in e-commerce.

1.2 Motivation

All commercial organisations should decide on their strategic development due to a competitive, technological and growing environment. Currently, the standard elements of defining a market strategy and competitive factors within a developing technological and regulatory environment are forecast dependent [29].

Due to a large quantify of product's data (time series) over a given period (forecast horizon), forecasting sales accurately is a delicate task that potentiates the growth of a retail chain. To avoid over-stocking and under-stocking of products that negatively impact a company, decisions about space allocation, availability, ordering and pricing product are essential aspects that are directly correlated with demand forecasting [43].

Additionally, one key area that retailers should consider when choosing a forecasting model/software is the appropriate stocking process for highly seasonal and promotional products to increase its accuracy. Besides existing strategic promotional methods (% discount, buy x get y free), a proper broadcasting method (social media, advertises) directly correlates with its success. As a result of this, different promotion types, when applied at the right time (Christmas, Black Friday), are a naive approach to inventory management and can lead to costly mistakes when neglected [14]. Finally, forecasting with various and complex product data does typically not result in better accuracy in traditional methods, thus choosing and tuning a proper model and its data is a significant challenge [6].

1.3 Objectives

The dissertation's main objective is to compare traditional and deep learning models in the electronic commerce sales forecasting problem on a real dataset. It can be decomposed into the following subgoals:

- Analyse and select performance measurements that best suit each model
- Analyse, treat and adapt the data to each model
- Analyse and implement multiple approaches to training and test data
- Train, validate and apply time series methods to the problem: SARIMA, SARIMAX, PROPHET and Holt Winter's
- Train, validate and apply multiple machine learning models approaches to the problem
- Train, validate and apply multiple deep learning models approaches to the problem
- Identify and incorporate features that best suit each model
- Implement a large-data process system that will be responsible for updating data and forecasting models
- Explore different variables on models when applicable: seasonality, promotions

Recent studies appear to indicate that forecast accuracy increases with the use of deep learning models when analysing a large volume of e-commerce sales (ECS) while requiring minimal manual work [23] [36].

1.4 Document structure

The following chapters are structured as follows:

1. Literature Review, Chapter 2, presents a brief background on state of the art on usage of AI techniques on sales forecasting and the necessary background to understand the topics explored in this dissertation. It covers background for the different demand forecast models, ways to validate them, and a preliminary literature review of the available technologies.

2. Problem Statement, Chapter 3, explains the main problem of this dissertation and the proposed plan to address it. It covers a guideline and a complete description of the different tasks of the proposed solution.

3. Conclusions and Future Work, Chapter 5, provides an initial conclusion of the dissertation planning work.

Chapter 2

Literature Review

This chapter discusses the state of the art of the demand forecast domain: background of the subject and the related work done in demanding forecast in general and demanding forecasting in e-commerce in particular.

2.1 Forecasting models

This section presents additional background on well-known forecast models.

2.1.1 Time-series models

Time-series forecasting takes an existing series of data $x_{i-n}, \ldots, x_{i-2}, x_{i-1}, xi$ and estimates x_{i+1}, x_{i+2}.... information values. Data is present in a broad period. Thus information about data at various time intervals can be extracted. Usually, the size of the data correlates directly with the efficiency of a time-series model. [22] [20]. As demand forecasting depends on the seasonality and trend of the industry, the models to be discussed are:

- Holt-Winter's (HW) - utilizes exponential smoothing, weighted moving average of recent time-series [50].
- Seasonal Autoregressive Integrated Moving Average (SARIMA) - takes into account seasonality trends but instead captures moving averages throughout time-series [33].
- Prophet - is a "procedure developed for forecasting time series data based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects" [49].

2.1.1.1 Holt Winters

First introduced in the 1960s, this model is an extension of the exponential smoothing method. The calculation of the prediction is based upon of all data values present in a series [19] [34]. The smoothed series can take the following form:

$$S_t = \alpha y_t + (1-\alpha) S_{t-1} \tag{2.1}$$

$S_1 = y_1$, which means that:

$$S_t = \alpha y_t + (1-\alpha) y_{t-1} + \alpha(1-\alpha)^2 y_{t-2} + ... + (1-\alpha)^{t-1} y_1 \tag{2.2}$$

We can conclude that the smoothed series depends on all past values, giving more importance to the latest ones 2.2.

- S_t is the smoothed series
- α is the smoothing parameter
- S is smoothing constant between $0 < \alpha < 1$

To tackle the problem of using exponential smoothing for seasonal data that includes cycles or tends, HW uses a modified form of exponential smoothing. It comprises three smoothing equations, ℓ_t, b_t and s_t, with corresponding smoothing parameters α, β^* and γ. These equations are applied to a series with a trend constant seasonal component using the additive and multiplicative methods. The additive HW component 2.3 is used when the seasonal variations are roughly constant through the series, while the multiplicative method 2.4 is preferred when the seasonal variations are changing proportionally to the level of the series.

$$\begin{aligned} \ell_t &= \alpha(Y_t - s_{t-p}) + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^*(\ell_t - \ell_t - 1) + (1-\beta^*) b_{t-1} \\ s_t &= \gamma(Y_t - \ell_t) + (1-\gamma) s_{t-p} \end{aligned} \tag{2.3}$$

$$\begin{aligned} \ell_t &= \alpha \tfrac{Y_t}{s_{t-p}} + (1-\alpha)(\ell_{t-1} + b_{t-1}) \\ b_t &= \beta^*(\ell_t - \ell_t - 1) + (1-\beta^*) b_{t-1} \\ s_t &= \gamma \tfrac{Y_t}{\ell_t} + (1-\gamma) s_{t-p} \end{aligned} \tag{2.4}$$

Where:

- ℓ_t is the level equation
- b_t is the trend equation
- s_t is the seasonality equation
- m is the frequency of the seasonality, i.e the number of seasons in a year.

2.1.1.2 SARIMA(X)

The Seasonal ARIMA model mostly mentioned as SARIMA (p,d,q)x(P, D, Q)s, where p, d, q and P, D, Q are non-negative integers that refer to the polynomial order of the autoregressive (AR), integrated (I), and moving average (MA) parts of the non-seasonal and seasonal components of the model, respectively [52]. Seasonal ARIMA(X) can be composed of 5 parts [11]:

- Seasonality (S).
- Autoregression (AR) - a model composed of a changing variable that regresses its own lagged, or prior, values.
- Integrated (I) - a difference of raw observations (data values and previous values) to allow the time series to be stationary.
- Moving average (MA) - dependency between an observation and a residual error from a moving average model applied to lagged observations
- Exogenous variables (X).

The SARIMA model can take the following form:

$$\varphi_p(B)\Phi_P(B^s)\nabla^d\nabla_s^D yt = \theta_q(B)\Theta Q(B^s)\varepsilon_t \tag{2.5}$$

Where:

- y_t is the forecast variable
- $\varphi_p(B)$ is the regular AR polynomial of order p
- $\theta_q(B)$ is the regular MA polynomial of order q
- $\Phi_P(B^s)$ is the seasonal AR polynomial of order P
- $\Theta Q(B^s)$ is the seasonal MA polynomial of order Q
- ∇^d is the differentiating operator
- ∇_s^D is the seasonal differentiating operator
- B is the back ship operator, responsible for operating y_t by shifting it one point in time
- ε_t is the white noise process
- S is the seasonal period

Similarly, the SARIMAX model is generally expressed mathematically as:

$$\varphi_p(B)\Phi_P(B^s)\nabla^d\nabla_s^D yt = \beta_k x_{k,t}{}' + \theta_q(B)\Theta Q(B^s)\varepsilon_t \tag{2.6}$$

Where:

- $x_{k,t}$ is vector including k^{th} explanatory input variables at time t
- β_k is the coefficient value of the k^{th} exogenous input variable

2.1.1.3 Prophet

Prophet is based on a decomposable time series model with three main components: trend, seasonality and holidays [49]. They can be expressed in the following equation:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t \tag{2.7}$$

Where:

- $g(t)$ is the trend function responsible for modelling the non-periodic changes in the value of the time-series
- $s(t)$ represents the periodic changes (daily, weekly, yearly seasonality)
- $h(t)$ represents the effects of holidays that occur on potentially irregular schedules over one or more days
- ε_t is the error term, any idiosyncratic changes which the model does not accommodate

In conclusion, this specification is similar to a generalized additive model (GAM) [18], a class of regressions models with potentially non-linear smoothers applied to the regressors.

2.1.2 Machine learning models

Machine learning (ML) is a branch of Artificial Intelligence (AI) with its root in computational statistics.It focuses on estimating the behavior or outcome in a specific time and context using computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field [55]. ML can be subdivided in three main approaches, namely, supersized learning, unsupervised learning and reinforcement learning. Nevertheless, the focus of this thesis is the supervised learning approach. Supervised learning is the construction of algorithms that can produce general patterns and hypotheses, which predicts future instances using externally supplied instances. This prediction focuses on classification and regression based on known features previously learned from the training data [46] [8].

The models to be discussed are:

- Support Vector Machines (SVM) - a set of supervised learning methods
- Random Forest (RF) - composed of different decision trees, each with the same nodes
- Logistic Regression (LR) - a statistical model that uses a logistic function to model a binary dependent variable
- Boosting algorithms: AdaBoost, Light Gradient Boosting Machine (Light GBM) and Extreme Gradient Boosting (XGBoost) - a family of algorithms that converts weaker learners to strong ones
- Gaussian Naive Bayes (GNB) - supervised learning algorithm based on Gaussian variant of the Naive Bayes algorithm

2.1.2.1 Support Vector Machines

SVMs use an implicit mapping of input data into a high-dimensional feature space defined by a kernel function. A kernel function is responsible for return the inner product between the images of two data points in the feature space. It is in the feature space that the learning phase takes place, while the data points only appear inside dot products with other points, i.e. the "kernel trick" [21] [42]. The following equation demonstrates how the kernel functions.

$$k(x,x^{'}) = (h\Phi(x),\Phi(x^{'})) \tag{2.8}$$

Where:

- Φ is the implicit mapping
- $(h\Phi(x),\Phi(x^{'}))$ is the inner product
- $x,x^{'}$ are the data points
- k is the kernel function

2.1.2.2 Random Forest

RF is composed of different decision trees, i.e. it classifies cases by commencing at the tree's root and passing through it unto a leaf node. Different rules for tree growing, tree combination, self-testing and post-processing are also used on RF. It is also robust to over-fitting and more stable in the presence of outliers. It uses the Gini index to measure the prediction power of features, based on the principle of impurity reduction, and also non-parametric as it does not rely on data belonging to a particular type of distribution [39]. A binary split of the Gini index can take the following form:

Where:

$$Gini(n) = 1 - \sum_{j=1}^{2}(p_j)^2 \tag{2.9}$$

Where:

- p_j is the relative frequency of class j in the node n

2.1.2.3 Logistic Regression

LR is a predictive analysis used to describe data and explain the relationship between one dependent variable and one or more independent variables (nominal, ordinal, interval, or ratio-level). It is an algorithm that learns a model for binary classification and is a statistical method for analyzing a dataset, where independent variables determine an outcome. Based on probabilities, odds and logarithm of odds [24]. These can be expressed using the following equation:

$$odds = \frac{f(E)}{1 - f(E)'} \tag{2.10}$$

Where:

- $f(E)$ is the probability of and event E
- $1 - f(E)$ is the probability of a no event

2.1.2.4 Boosting

Boosting based algorithms sequentially combine simple rules to form an ensemble that focuses on increasing the performance of the single ensemble member, i.e boosted [30]. This can expressed using the following equation:

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x). \tag{2.11}$$

Where:

- α_t is the coefficient with which h_t is combined

In this case, AdaBoost algorithms initially assign equal weights to each training observation. It uses multiple weak models and assigns higher weights to those observations for which misclassification was observed [35].

Similarly, Gradient Boost algorithms sequentially create new models from an ensemble of weak models with the idea that each new model can minimize the loss function. The loss function is measured by the gradient descent method. Finally, to avoid over-fitting, a stop criteria of the creation of new models is used. LightGBM and XGBoost are decision tree-based algorithms that use the gradient boost method, and the main difference is the split method used on the decision tree. LightGBM uses Gradient-based one-side sampling (GOSS), while XGBoost uses a histogram-based algorithm to filter the observations. [35].

2.1.2.5 Gaussian Naive Bayes

GNB consists of assigning the label of the class that maximizes the posterior probability of each sample, presupposing that the voxel contributions are independent and follow a Gaussian distribution [31]. The GNB decision rule can take the following form:

$$\delta_i^k = -\sum_{j=1}^{v} \left(\frac{(x_{ij}^{TE} - \hat{u}_j^k)^2}{2\hat{\sigma}_j^2}\right) + log(p_k) \tag{2.12}$$

Where:

- δ^k is the discriminant function, i.e. the sum of the squared distances to the centroid of each class k, across all voxels in the searchlight

- i is the sample
- σ is the variance
- p_k is the logarithm of the a-priori probability computes in the training set

2.1.3 Deep Learning Models

Deep Learning (DL) is a machine-learning method based on the characterization of data learning. Mostly done by feeding raw data to a machine to automatically discover patterns needed for detection or classification of data, known as representation learning. By composing simple but non-linear modules that transform the representation at a higher, slightly more abstract level, very complex functions can be learned. Features are not pre-defined but instead are learned from data using a general-purpose learning procedure [26]. The models to be discussed are:

- Artificial Neural Network (ANN) - consists of input of layers of neurons, hidden layers of neurons, and final layer of output neurons [54].
- Recurrent Neural Network (RNN) - used to process sequences of data [57].
- Convolutional Neural Network (CNN) - used in computer vision tasks, it is based on convolutional layers [44].

2.1.3.1 ANN

ANNs principle is similar to how a human brain works. They can be used in classification or pattern recognition tasks. Neural Network (NN) layers are independent, i.e. a specific layer can have an arbitrary number of nodes (bias nodes), its primary role is to provide nodes with trainable value (constant), Figure 2.1 represents an architecture of a neural network. Additionally, the activation function (defined as the output of a node given an input or set of input) also determined by the bias value. NNs can be used in classification problems, where the input and output nodes will match the input and output features. On the other hand, it has an input and an output node when used in function approximations. ANNs can be shallow or deep, i.e., shallow when there is only one hidden layer or deep when there are more than one. The term Deep Neural Network (DNN) is used with deep ANNs [16]. One key difference in ANNs is the flow between input and output nodes. It can be either FeedForward (FFNN), seen in Figure 2.2, or FeedBackward (FBNN), seen in Figure 2.3. The difference is that in FFNNs, signals travel one way only (input to output). In FBNNs, networks can have signals travelling in both directions [5].

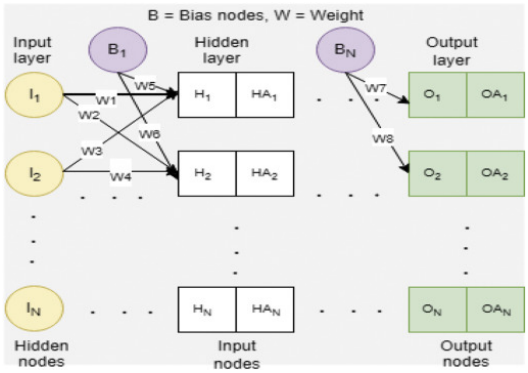


Figure 2.1: Neural Network Architecture (source: [16])

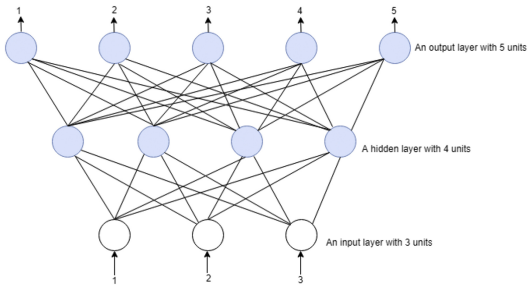


Figure 2.2: Feed Forward Network (source: [16])

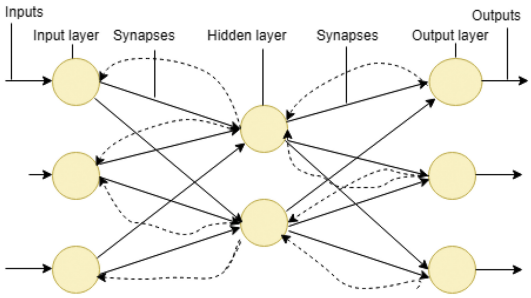


Figure 2.3: Feed Backward Network (source: [16])

2.1.3.2 RNN

To introduce a notion of time, RNNs are FFNNs augmented by the inclusion of edges that "stretch" adjacent time steps, called recurrent edges. The recurrent edges can form cycles that connect a node to itself across time (backpropagation), seen in Figure 2.4 [28]. This can be expressed using the following equation:

$$\begin{aligned} h^{(t)} &= \sigma(W^{hx}x^{(t)} + W^{hh}h^{(t-1)} + b_h) \\ \hat{y}^{(t)} &= softmax(W^{yh}h^{(t)} + b_y). \end{aligned} \tag{2.13}$$

Where:

- $x^{(t)}$ is the current data point (input)
- $h^{(t)}$ is the hidden node values
- $\hat{y}^{(t)}$ is the output
- W^{hx} is the matrix of conventional weights between the input and the hidden layer
- W^{hh} is the matrix of reecurrent weights between the hidden layer and itself at adjacent time steps
- b_h is a bias parameter
- b_y is a bias parameter

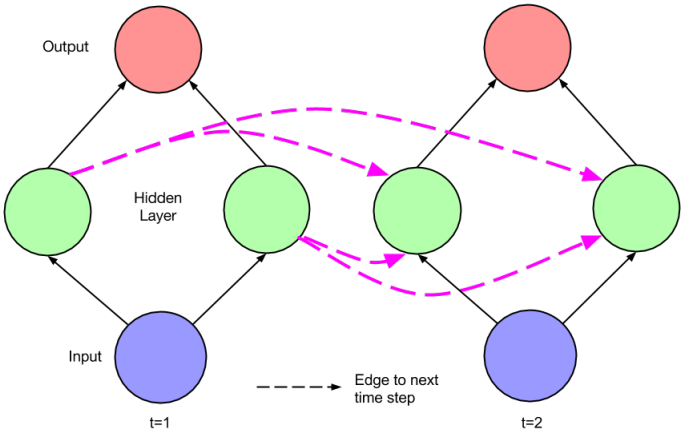


Figure 2.4: Recurrent Neural network example (source: [28]

Examples of RNNs implementations:

- Bidirectional recurrent neural networks - connect two hidden layers of opposite directions to the same output. The output layer can get information from backwards and forward states simultaneously [41].
- DeepAR - Forecasting method based on autoregressive RNNs [36]
- Gated recurrent units (GRU) - Uses the concept of gates, similar to neural networks, each gate has its own biases and weights. It is composed of two gates: a reset and an update gate [12].
- Long Short Term Memory (LSTM) - Similar to GRU, besides having a reset and update gate, it has a forget and output gate [12].

2.1.3.3 CNN

CNNs are a type of neural networks used for processing grid pattern data (images). It receives an input that sequentially goes through a series of layers (processing). One layer feeds its output to the next one, increasing the complexity of features extracted. A CNN model can be composed of 5 layers [56], as seen on Figure 2.5:

- Input - resizes data
- Convolutional - filter data to extracts features from it
- Pooling - preserves important information as large data is shrinking down. It also preserves the best fit for each feature
- Rectified linear unit layer (ReLU) - swaps every negative number of the pooling layer with 0s.
- Fully connected layer - Converts high level filtered images into categories with labels

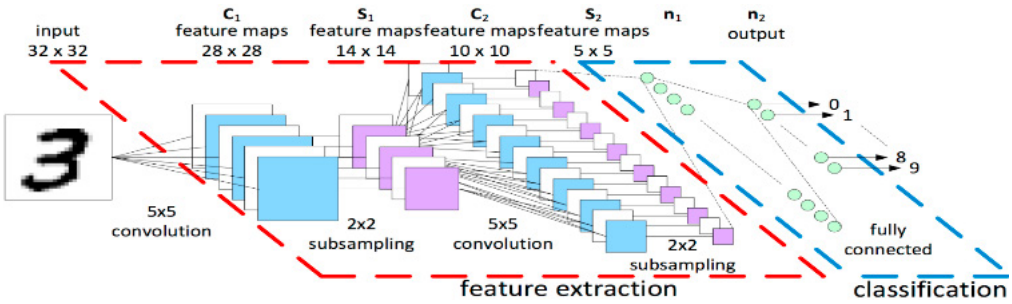


Figure 2.5: CNN example (source: [56]

2.2 Issues relating to forecasting models

This section presents forecast models' evaluation measures that can be used to calculate their performance.

2.2.1 Model Update Strategies

One key concept that should not be neglected is concept drift. Concept drift is used as a generic term to describe changes in data. As data is organized in the form of data streams that are constantly updated, the performance of learning models decays since the testing data comes from a different distribution than the training data, as seen in Figure 2.6. To tackle this problem, prediction models must have mechanisms to be continuously evaluated and to be able to adapt to changes in data over time [59]).

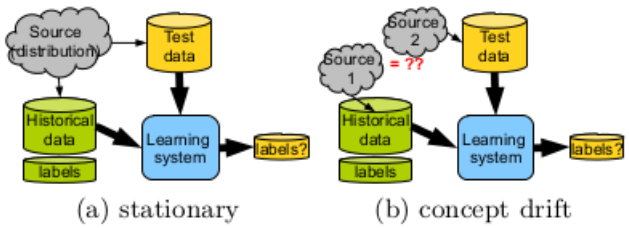


Figure 2.6: Stationary supervised learning (a) and learning under concept drift (b). (source: [59])

Prediction models' learning may come from two different approaches [9]:

- Online - Iterative and incremental, where the model is being updated along with the data, this can be made in real-time or in batches, seen on Figure 2.7.
- Batch (Offline) - Model is trained using the entire dataset once, seen in Figure 2.8.

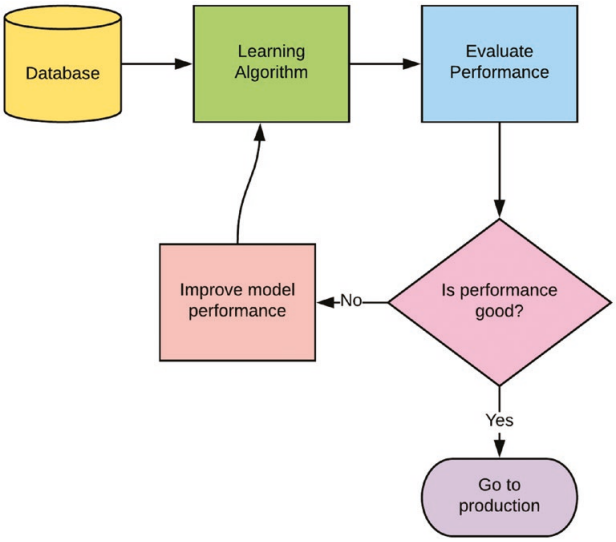


Figure 2.7: Batch learning (source: [9])

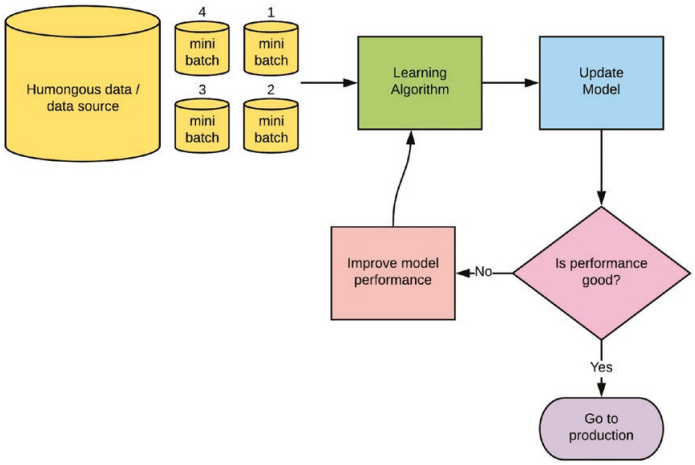


Figure 2.8: Online Learning (source: [9])

2.2.2 Evaluation Measures

Prediction learning models should be assessed using various evaluation measures considering the type of model and the specification of each task. In sales forecasting tasks, the main objective is to correctly calculate the demand forecast with the minimal margin of error possible (directional and size). The presented measures can be used to better evaluate a model [25], [37], [4], [1] .

$$Bias = \frac{\sum(Yt - Ft)}{n}. \tag{2.14}$$

The bias, which is the average of the forecast errors, measures the direction of the error. A model performs the best when this value is closer to 0.

$$MAD = \frac{\sum |Y_t - F_t|}{n} \tag{2.15}$$

The mean absolute deviation (MAD) calculates the amount of error. The smaller the value, the better.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} \tag{2.16}$$

The mean absolute error (MAE) measures the absolute average distance between the real data and the predicted data.

$$MSE = \frac{\sum(Y_t - F_t)^2}{n} \tag{2.17}$$

The mean square error (MSE) calculates the dispersion of errors. The smaller the value, the better. The square root of the MSE comes from the standard deviation of the errors.

$$MRE = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i - \hat{y}_i|}{y_i} 100\% \tag{2.18}$$

The mean relative error (MRE) is an approximation error measure based on the mean of the relative errors.

$$MAPE = \frac{\sum \frac{|Y_t - F_t|}{Y_t}}{n} \tag{2.19}$$

The mean absolute percentage error (MAPE) calculates the average of the absolute values of percentage errors. The lower the value, the better. MAPE also attempts to consider the effect of the magnitude of the values.

$$PFE = \frac{2s_e}{\hat{Y}_{t+1}} 100\% \tag{2.20}$$

The percentage forecast error (PFE) calculates the relative dispersion of the mean like the Coefficient of Variation (CV). It provides a conservative estimate of the accuracy of the model.

$$RMSE = \sqrt{MSE} \tag{2.21}$$

The Root Mean Square Error (RMSE) is the standard deviation of the prediction errors, i.e. it measures the quality of the fit between the actual data and the predicted model.

$$R^2 = \frac{\sum_{i=1}^{n} (y_i - f(x_i))^2}{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \tag{2.22}$$

The Coefficient of Determination (R^2) is used to analyze how a second variable can explain differences in one variable by dividing the residual amount of squares by the total sum of squares.

2.2.3 Error measurement selection

According to the literature [45], there are suggestions regarding the selection of error measurements to different tasks, these are:

- If the forecast performance is evaluated for time series with a similar scale data where data was preprocessed, then it is advisable to use MAE, MdAE or RMSE
- Percentage errors are not advised due to non-symmetry.
- If data contains outliers, the forecast horizon is large, identical values are not present and the normalization factor is not equal to 0, it is advisable to use MASE

2.3 Related Work

This section presents information about the state of the art of demand forecasting and its applications.

2.3.1 Sales Forecasting

Information (retail type, prediction models, evaluation measures and variables) about the literature review of demand forecast is present in table 2.1

Table 2.1: Demand forecast studies

Reference	Retail type	Type of data	Models	Additional variables	Evaluation measures	Problems	Best model	Best type of model	Important notes	Type of study
İşlek and Öğüdücü [58] 2015	Food retail	Sales invoices from 2011-2013	Bayesian networks	-	MAPE	-	Bayesian network machine learning	One for each food chain distribution warehouse	Moving average of sales. Development of product ontology to predict sales on new products. Cluster warehouses according to sale amount of products. Use of Protégé (onthology framework) to construct a product ontology to fix cold start problems.	Academic
Li et al. [27] 2018	60 different types (e-commerce)	Jan 2014 - March 2017	ARIMA, ARIMA-NARNN, NARNN	-	MRE, R^2, *RMSE*	It is easy to cause underftting and overftting because of poor control of the model structure. Prediction of linear components is not as efective as the ECS-ARIMA model.	ARIMA-NARNN	Global	-	Academic
Ali et al. [6] 2009	Grocery chain (pasta, sauces, noodles, and ready-made meals)	11936 SKU-week combination 76 weeks	RT, SVR, RT (explicit features), SVR (explicit features)	Forecast using promotions	MAE, MAPE	1 category tested. Clustering of stores (identify the basis for information) could be researched	Regression tree using explicit features (without promotions)	Global	Important features: SKU (average, sum, trend, standard deviation) of the past 4–12 weeks, and stocks, such as promotion stock	Academic
Singh et al. [47] 2020	Multiple types (e-commerce)	100000 transactional order history from 2016 to 2018 of a braizllean e-commerce store	RF, XGBoost, ARIMA, SARIMA	-	Accuracy, RMSE, MAE, R^2	Dataset timespam (only 20 months) and limited dataset	SARIMA	Global	-	Academic
Gurnani et al. [15] 2017	Rossman Store, Drug store company	Jan 2013 - July 2015 (Kaggle)	ARIMA, ARNN, XGBoost, SVM, Hybrid ARIMA-ARNN, Hybrid ARIMA-XGBoost, Hybrid ARIMA-SVM, STL Decomposition	-	MAE, RMSE	-	STL Decomposition (Seasonal - snaive, trend - ARIMA, remainder - XGBoost)	Global (drug)	-	Academic

Table 2.1: Demand forecast studies

Reference	Retail type	Type of data	Models	Additional variables	Evaluation measures	Problems	Best model	Best type of model	Important notes	Type of study
Pavlyshenko [32] 2019	Rossman Store, Drug store company	Jan 2013 - July 2015 (Kaggle)	ExtraTree, ARIMA, Random Forest, Lasso, Neural Network, Stacking	-	MAE	Insufficient data analysis	Stacking (ExtraTrees, Neural Network, Lasso)	Global (drug)	Important features: Date (month, week, year) and Promotions	Academic
Kilimci et al. [23] 2019	SOK Market in Turkey	106 weeks of sales	3 Regression models, Support Vector Regression, Exponential Smoothing Model, Holt-Trends, Holt-winters, 3 Two-level models (mix between Regression, Holt-Winters and Exponential Smoothing Model), MLFFANN	-	MAPE, MAD	Very difficult to compare with other studies because of the difference in techniques and datasets used. Variety of data. DL approaches (models sued and hyperparameters). Explore new heuristics on the Proposed Integration Strategy	Integration Strategy with DL (MLFFANN)	One for each type of product (only metric used)	Select the best performing forecasting method to future forecast or choose the best performing of the current week and calculate the prediction by combining weighted predictions of winners (Proposed Integration Strategy). PCA feature extraction. Data cleaning (Hadoop SPARK). Dimensionality Reduction. K means cluster	Academic
Tugay and Oguducu [51] 2020	Fashion store (e-commerce)	2015-2017 (3126648 orders)	Generelizaed linear model (large class of conventional linear regression models), DT, GBT	-	RMSE	Size of data too small. Only 85000 records were used due to performance constraints, fields and attributes, used in this analysis were insufficient for the further analysis, and Accuracy rate, Error rate, Precision, Recall, Kappa	GBT	Global (fashion)	Exploratory Analysis, Outlier detection, Forecasting and trends (clusters)	Academic

2.3.2 Available Technologies

Information about available demand forecast technologies can be found in table 2.2

Table 2.2: Available Technologies

Name	Cloud based	Demand forecast	Product extension (forecast profit, stock levels)	Integrations	Additional features
Inventory Planner (e-commerce)	Yes	Yes	Yes	Aggregate multichannel sales (Shopify, Amazon)	Create purchase orders, Manage cash flow with open-to-buy planning
SavvyCube (e-commerce)	Yes	Yes	Yes	Aggregate multichannel sales (Shopify, Magnento, Google Analytics)	Sales analytics, Product Analytics
Infor Demand Planning	Yes	Yes	Yes	Infor Ming.IE Integration, ERPs	Scenario management, Inventory optimization, Omnichannel demand planning
SAP Advanced Planning and Optimization	Yes	Yes	Yes	ERP	Supply Chain Cockpit, Supply chain collaboration, Industry-specific applications
Xant	Yes	Yes	Yes	SalesForce, Microsoft, SAP	Predictive Pipeline (automatic forecast pipeline)

Chapter 3

Dataset

This chapter discusses the dataset used in this thesis project. The dataset information is subdivided into its overview, the work done to migrate the existing data, data understanding, and preparation.

3.1 Dataset Overview

The dataset used in this dissertation and the migration of data to Amazon Web Services (AWS) Redshift, an Extract Transform Load (ETL) tool are discussed in more detail in this chapter.

The following dataset was gathered from Jumpseller's e-commerce stores containing around 39 million orders from 50 000 stores operating in different retail types. As the literature suggests [17] a minimum of 50 monthly data points (50/12 $\tilde{4}$.16 years), the dataset's period ranges from November 2017 to November 2021. Nevertheless, there are historical records since 2011. Figure 3.1 summarizes the tables used on this dataset. All tables are present on a standard MYSQL database except for the table Visits, which is present on a standard AWS S3 (a simple Simple Storage Service) bucket.

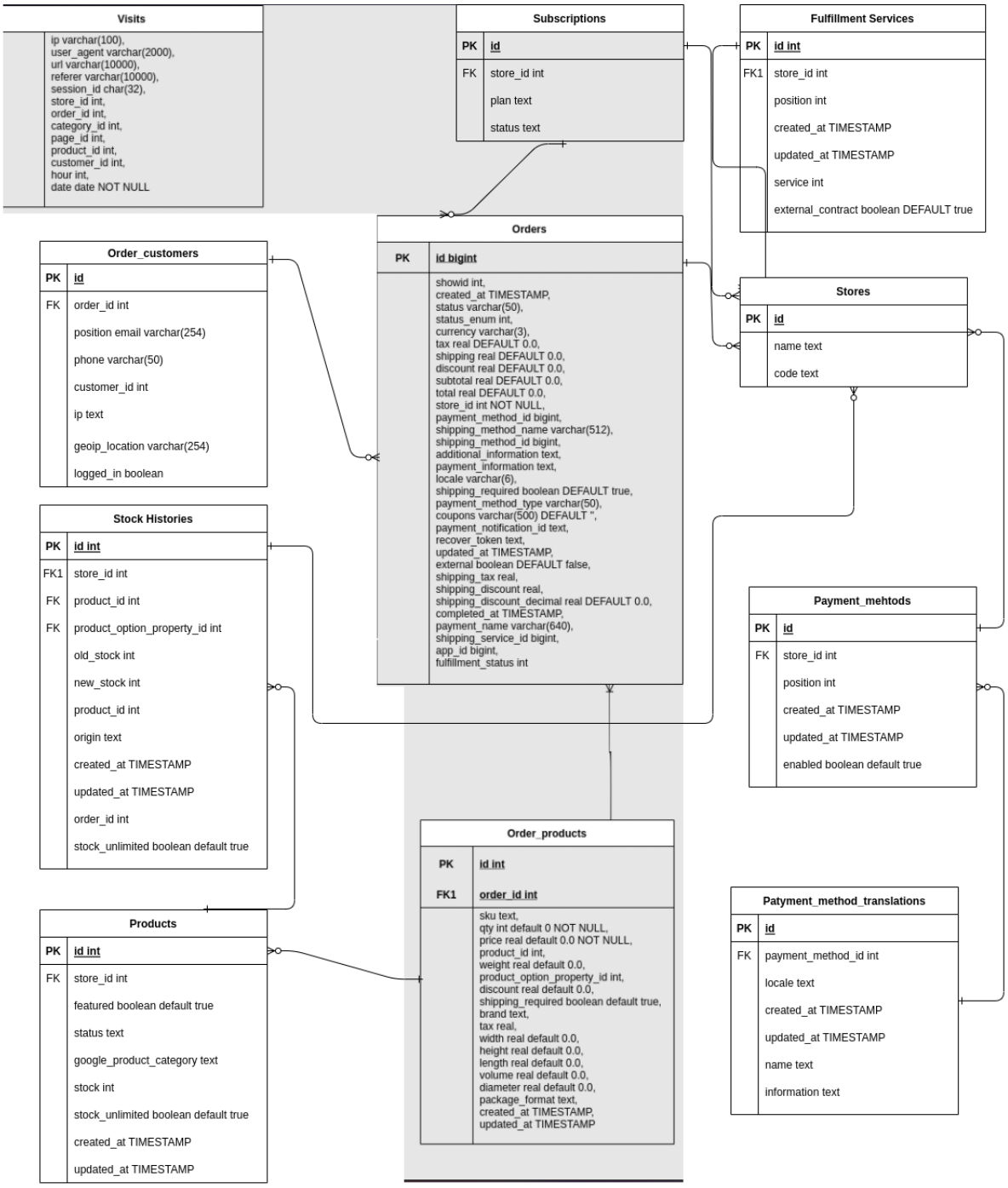


Figure 3.1: Database Entity Relationship Diagram

- Visits - information about page visits
- Order customer - customer information
- Order products - order product information

- Order - order information
- Payments - payments services that store has
- Fulfillment services - fulfilment services that store has
- Stock histories - history of stock
- Products - remaining product information (some order products are not present here)
- Translation tables - The value of the object in the store's default language
- Stores - Information about the eCommerce store
- Subscriptions - Information about the store's subscription

3.2 Redshift Migration

In order to choose the best large-data process system, a series of tests were made using AWS Redshift and AWS Redshift Spectrum, 2 Online Analytic Processing (OLAP) cloud solutions. AWS Redshift is a warehouse solution based on standard Structured Query Language (SQL), that can efficiently handle a large quantity of data thanks to its Massively parallel Processing (MPP) computing. It is used to process real-time analytics and combine multiple data sources. On the other hand, AWS Redshift Spectrum joins data from Amazon S3 into AWS Redshift. It is a serverless query processing engine.

A series of 18 tests were made using the Orders, Order Customers, Order Products, and other tables not present in the final dataset: Traffic Sources and Addresses tables, which are present on a standard Mysql database. In addition, ten different tests were made using the Visits table.

The dummy dataset used on the tests consists of data comprehended between April 2021 to May 2021: 1058895 Visit entries, 100000 Order entries, 200000 Order Product entries, 10000 Order Customer entires, and other tables not present on the final dataset, 100000 Address entries, and 10000 Traffic Source entries.

The 18 different tests are the following:

test 1

SELECT SUM(orders.total) FROM orders WHERE orders.store_id = 99701 AND orders.status IN ('Pending Payment', 'Paid') AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' OR orders.completed_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' AND orders.completed_at IS NOT NULL);

test 2

SELECT SUM(orders.total) FROM orders WHERE orders.store_id = 99701 AND orders.status IN ('Pending Payment', 'Paid') AND (orders.completed_at IS NULL AND orders.created_at

BETWEEN ’2021-04-10 23:00:00’ AND ’2021-05-31 22:59:59’ OR orders.completed_at BETWEEN ’2021-04-10 23:00:00’ AND ’2021-05-31 22:59:59’ AND orders.completed_at IS NOT NULL);

test 3

SELECT SUM(orders.total) FROM orders WHERE orders.store_id = 99701 AND orders.status IN (’Pending Payment’, ’Paid’) AND (orders.completed_at IS NULL AND orders.created_at BETWEEN ’2021-04-01 23:00:00’ AND ’2021-07-31 22:59:59’ OR orders.completed_at BETWEEN ’2021-04-01 23:00:00’ AND ’2021-07-31 22:59:59’ AND orders.completed_at IS NOT NULL);

test 4

SELECT orders.total FROM orders WHERE orders.store_id = 99701 AND orders.status IN (’Pending Payment’, ’Paid’) AND (orders.completed_at IS NULL AND orders.created_at BETWEEN ’2021-05-10 23:00:00’ AND ’2021-05-15 22:59:59’ OR orders.completed_at BETWEEN ’2021-05-11 23:00:00’ AND ’2021-05-15 22:59:59’ AND orders.completed_at IS NOT NULL);

test 5

SELECT orders.total FROM orders WHERE orders.store_id = 99701 AND orders.status IN (’Pending Payment’, ’Paid’) AND (orders.completed_at IS NULL AND orders.created_at BETWEEN ’2021-04-10 23:00:00’ AND ’2021-05-31 22:59:59’ OR orders.completed_at BETWEEN ’2021-05-10 23:00:00’ AND ’2021-05-31 22:59:59’ AND orders.completed_at IS NOT NULL);

test 6

SELECT orders.total FROM orders WHERE orders.store_id = 99701 AND orders.status IN (’Pending Payment’, ’Paid’) AND (orders.completed_at IS NULL AND orders.created_at BETWEEN ’2021-04-01 23:00:00’ AND ’2021-07-31 22:59:59’ OR orders.completed_at BETWEEN ’2021-04-01 23:00:00’ AND ’2021-07-31 22:59:59’ AND orders.completed_at IS NOT NULL);

test 7

SELECT order_products.product_id, SUM(qty) AS sum_qty, SUM(qty * price) AS sum_total FROM order_products LEFT OUTER JOIN orders ON orders.id = order_products.order_id WHERE order_products.product_id IS NOT NULL AND order_products.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN ’2021-05-10 23:00:00’ AND ’2021-05-15 22:59:59’ OR orders.completed_at BETWEEN ’2021-05-10 23:00:00’ AND ’2021-05-15 22:59:59’ AND orders.completed_at IS NOT NULL) AND orders.status IN (’Pending Payment’, ’Paid’)) GROUP BY order_products.product_id ORDER BY sum_total desc LIMIT 10;

test 8

SELECT order_products.product_id, SUM(qty) AS sum_qty, SUM(qty * price) AS sum_total FROM order_products LEFT OUTER JOIN orders ON orders.id = order_products.order_id WHERE order_products.product_id IS NOT NULL AND order_products.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN ’2021-04-10 23:00:00’ AND ’2021-07-31 22:59:59’ OR orders.completed_at BETWEEN ’2021-04-10 23:00:00’ AND ’2021-07-31 22:59:59’ AND orders.completed_at IS

NOT NULL) AND orders.status IN ('Pending Payment', 'Paid')) GROUP BY order_products.product_id ORDER BY sum_total desc LIMIT 10;

test 9

SELECT traffic_sources.referral_url FROM traffic_sources WHERE traffic_sources.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' OR orders.completed_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' AND orders.completed_at IS NOT NULL));

test 10

SELECT traffic_sources.referral_url FROM traffic_sources WHERE traffic_sources.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' OR orders.completed_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' AND orders.completed_at IS NOT NULL));

test 11

SELECT DISTINCT orders.created_at, orders.completed_at, orders.total, orders.status, addresses.country, addresses.region, orders.payment_name, orders.shipping_method_name FROM orders LEFT OUTER JOIN addresses ON addresses.order_id = orders.id AND addresses.type = 'OrderBillingAddress' WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' OR orders.completed_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' AND orders.completed_at IS NOT NULL) AND orders.status IN ('Pending Payment', 'Paid') ORDER BY COALESCE(orders.completed_at, orders.created_at);

test 12

SELECT DISTINCT orders.created_at, orders.completed_at, orders.total, orders.status, addresses.country, addresses.region, orders.payment_name, orders.shipping_method_name FROM orders LEFT OUTER JOIN addresses ON addresses.order_id = orders.id AND addresses.type = 'OrderBillingAddress' WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' OR orders.completed_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' AND orders.completed_at IS NOT NULL) AND orders.status IN ('Pending Payment', 'Paid') ORDER BY COALESCE(orders.completed_at, orders.created_at);

test 13

SELECT orders.id, orders.created_at, orders.completed_at, orders.status FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' OR orders.completed_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' AND orders.completed_at IS NOT NULL);

test 14

SELECT orders.id, orders.created_at, orders.completed_at, orders.status FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN

'2021-04-10 23:00:00' AND '2021-07-31 22:59:59' OR orders.completed_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' AND orders.completed_at IS NOT NULL);

test 15

SELECT order_customers.customer_id, orders.created_at, orders.completed_at FROM order_customers LEFT OUTER JOIN orders ON orders.id = order_customers.order_id WHERE order_customers.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' OR orders.completed_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 22:59:59' AND orders.completed_at IS NOT NULL));

test 16

SELECT order_customers.customer_id, orders.created_at, orders.completed_at FROM order_customers LEFT OUTER JOIN orders ON orders.id = order_customers.order_id WHERE order_customers.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' OR orders.completed_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 22:59:59' AND orders.completed_at IS NOT NULL));

test 17

SELECT order_customers.customer_id FROM order_customers WHERE order_customers.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 23:00:00' OR orders.completed_at BETWEEN '2021-05-10 23:00:00' AND '2021-05-15 23:00:00' AND orders.completed_at IS NOT NULL));

test 18

SELECT order_customers.customer_id FROM order_customers WHERE order_customers.order_id IN (SELECT orders.id FROM orders WHERE orders.store_id = 99701 AND (orders.completed_at IS NULL AND orders.created_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 23:00:00' OR orders.completed_at BETWEEN '2021-04-10 23:00:00' AND '2021-07-31 23:00:00' AND orders.completed_at IS NOT NULL));

Table 3.1: Results for MYSQL tables

Test number	Amazon Redshift	Amazon Redshift Spectrum
test 1	13897458,9	5847027767
test 2	15939279,3	5946942196
test 3	21508199,4	5613572308
test 4	885010279,8	6639227726
test 5	3069413389	5743753126
test 6	6360793996	5654152828
test 7	8980832	8194938903
test 8	7744178,5	6812091909
test 9	39657177,6	6791674556

Table 3.1: Results for MYSQL tables

Test number	Amazon Redshift	Amazon Redshift Spectrum
test 10	54548377,1	2127894949
test 11	432345668,4	2417881725
test 12	1345273527	1961863334
test 13	1544052222	2037843901
test 14	10462905943	1578415966
test 15	327065812,3	1926012288
test 16	1484579211	1480700754
test 17	40166146,8	1743014503
test 18	538206582,3	1454996769
Sum in ns	26652088280.4	73972005508
Sum in s	26,65	73,97

The 10 tests for visit tables are the following:

Test 1

SELECT * FROM <database>.visits_parquet_improved limit 10;

Test 2

SELECT store_id, DATE(year || '-' || month || '-' || day) AS date, COUNT(DISTINCT(session_id)) AS visits FROM <database>.visits_parquet_improved WHERE session_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-06-23') AND DATE('2021-06-24') GROUP BY store_id, year, month, day ORDER BY year, month, day

Test 3

SELECT avg(CASE WHEN url LIKE 'avg(CASE WHEN url LIKE 'FROM <database>.visits_parquet_impro WHERE product_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-06-24') AND DATE('2021-06-25') GROUP BY store_id, year || '-' || month || '-' || day

Test 4

SELECT store_id, DATE(year || '-' || month || '-' || day) AS date, COUNT(DISTINCT(session_id)) AS visits FROM <database>.visits_parquet_improved WHERE session_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-04-23') AND DATE('2021-07-24') GROUP BY store_id, year, month, day ORDER BY year, month, day

Test 5

SELECT store_id, DATE(year || '-' || month || '-' || day) AS date, COUNT(DISTINCT(session_id)) AS visits FROM <database>.visits_parquet_improved WHERE session_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-05-23') AND DATE('2021-06-24') GROUP BY store_id, year, month, day ORDER BY year, month, day

Test 6

SELECT store_id, DATE(year || '-' || month || '-' || day) AS date, COUNT(DISTINCT(session_id)) AS visits FROM <database>.visits_parquet_improved WHERE session_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-05-20') AND DATE('2021-05-24') GROUP BY store_id, year, month, day ORDER BY year, month, day

Test 7

SELECT store_id, DATE(year || '-' || month || '-' || day) AS date, COUNT(DISTINCT(session_id)) AS visits FROM <database>.visits_parquet_improved WHERE session_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-05-10') AND DATE('2021-05-25') GROUP BY store_id, year, month, day ORDER BY year, month, day

Test 8

SELECT avg(CASE WHEN url LIKE 'avg(CASE WHEN url LIKE 'FROM <database>.visits_parquet_impro WHERE product_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-05-20') AND DATE('2021-05-24') GROUP BY store_id, year || '-' || month || '-' || day

Test 9

SELECT avg(CASE WHEN url LIKE 'avg(CASE WHEN url LIKE 'FROM <database>.visits_parquet_impro WHERE product_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-05-10') AND DATE('2021-05-25') GROUP BY store_id, year || '-' || month || '-' || day

Test 10

SELECT avg(CASE WHEN url LIKE 'avg(CASE WHEN url LIKE 'FROM <database>.visits_parquet_impro WHERE product_id IS NOT NULL AND DATE(year || '-' || month || '-' || day) BETWEEN DATE('2021-04-10') AND DATE('2021-07-25') GROUP BY store_id, year || '-' || month || '-' || day

cluster type / nº nodes	Redshift dc2.large / 1	Spectrum dc2.large/ 1	Redshift ra3.4xlarge / 2	Spectrum ra3.4xlarge / 2	Redshift dc2.8xlarge / 2	Spectrum dc2.8xlarge / 2
test 1	0,397	1,839	0,0182	1,746	0,0188	2,233
test 2	0,775	5,274	0,573	2,973	0,651	3,318
test 3	0,671	5,307	0,152	2,656	2,074	4,511
test 4	47,212	97,214	7,132	15,285	7,894	12,739
test 5	20,197	44,736	3,549	9,074	3,78	9,144
test 6	2,434	8,082	0,296	4,185	0,302	4,307
test 7	8,129	20,401	0,585	6,817	0,677	7,475
test 8	2,294	2,481	0,276	4,413	0,271	5,082
test 9	6,993	6,206	0,536	4,837	0,503	6,406
test 10	39,727	27,2	7,807	12,887	3,983	20,612
Sum in s	128,834	218,743	20,928	68,879	18,29	75,832

AWS Redshift provides the best results when querying different data based on the tests made. Thus, this tool was chosen to develop the dataset.

Chapter 4

Data Preparation

Data preparation is the process of collecting, cleaning, and consolidating data in order to be used to predict the sales of a given product on a specific date. The main tasks are data cleaning, data transformation, and data reduction. In this chapter this tasks will be described in detail.

4.1 Data Understanding

The first phase of data understanding relies on studying the data integrity: accuracy, completeness, consistency, timeliness, validity, and uniqueness [40]. This information can be summarised in 4.1

Table 4.1: Dataset's data integrity

Dimension	Definition	Verdict
Accuracy	The degree to which data correctly describes the "real world" object or event being described	Since the data comes from a real e-commerce sales dataset, it is assumed to be accurate.
Completeness	The proportion of stored data against the potential of "100% complete"	In case of description, the tables with "product" prefix were not always present to all the order products. Information about visits is only available from March 2020.
Consistency	The absence of difference, when comparing two or more representations of a thing against a definition	The data is consistent. No discrepancies were found while joining the different tables
Timeliness	The degree to which data represent reality from the required point in time	All of the information needed to predict the sales of the product is available at the date of its request
Validity	Data are valid if it conforms to the syntax (format, type, range) of its definition	No invalid data was found. The orders were always saved after their request
Uniqueness	Nothing will be recorded more than once based upon how that thing is identified. It is the inverse of an assessment of the level of duplication	All the data is considered unique. Each order is associated with its products

It would be infeasible to analyze every order product data, the top 10 products that are present in more orders and ten that are less present. a Chilean-based clothing and sporting goods-based e-commerce store was chosen to conduct the studies present on the Data Understanding and Preparation phases.

- Seasonality of sales
- Relation between the quantity and price of products sold
  - Products sold
  - Number of sales

- Relation with discount data
- Outliers
- Missing values

Information about the chosen products can be summarised in the following table:

Table 4.2: Product information

Product name	Product description	Product options available	Product Category	Number of orders that contained this product	Average number of products present in order	Average price (in Chilean pesos)	Quantity standard deviation	Price standard deviation	Max number of products sold at an order	Max price of a product	Minimum number of products at an order	Minimum price of product
Product 1	A sports mask	Yes	Naroo mask	805	1.57	19990	1.72	0	40	19990	1	19990
Product 2	A sports mask	Yes	Naroo mask	513	1.41	19990	0.74	0	5	32990	1	32990
Product 3	A sports mask	Yes	Naroo mask	344	1.32	29990	0.74	0	6	29990	1	29990
Product 4	An hygienic mask	Yes	Seo	219	1.47	14990	0.70	0	4	14990	1	14990
Product 5	Smart bycycle trainer	No	Cyber Rodillos 2021	186	1	599990	0	0	1	599990	1	599990
Product 6	Bycyle speedometer	No	Garmin ciclismo	159	1.02	70115.79	0.14	3544.40	2	79990	1	79990
Product 7	Electrolyte Supplement Capsule	Yes	Isotonicos	148	1.08	28165.68	0.27	3862	2	29990	1	29990
Product 8	A fitness book	No	Libros	130	1.02	19205.38	0.19	976.5	3	19990	1	17990
Product 9	A heart rate strap	No	Bandas cardiacas	120	1.06	69240	0.23	3204.81	2	79990	1	59990
Product 10	Smart bycyle trainer	No	Tacx	118	1.01	744905.25	0.09	15112.33	2	749990	1	699990
Product 11	Sports eyewear	No	Seo	1	1	159990	0	0	1	159990	1	159990
Product 12	Therapeutic Gels	Yes	Apel	1	1	19990	0	0	40	19990	1	19990
Product 13	Frontal Snorkel	Yes	Seo	1	1	29990	0	0	1	29990	1	29990
Product 14	A sports mask	Yes	Seo	1	1	339990	0	0	1	339990	1	339990
Product 15	GPS silicone case	Yes	Seo	1	1	13989.64	0	0	1	13989.64	1	13989.64
Product 16	Microphiber towel	No	Toallas microfibra	1	2	6990	0	0	2	6990	2	6990
Product 17	A sports backpack	Yes	Mochilas	1	1	89990	0	0	1	89990	1	89990
Product 18	A sports backpack	Yes	Victorinox	1	1	59990	0	0	1	59990	1	59990
Product 19	Sports strap	No	Seo	1	1	26990	0	0	1	26990	1	26990
Product 20	Swim goggles	No	Lentes de nado	1	1	9990	0	0	1	9990	1	9990

4.1.1 Product analysis

This subsection will discuss the seasonality, relevant information, and the outliers of the top 10 products present in more orders and an overall analysis of the store's orders. For each product, the seasonality analysis, moving average of the total daily price of products sold, the distribution plot of daily data, outlier Z-score (better suited for normally or approximately normally distributed features) using 3 as a threshold and IQR (better suited for skewed data) analysis of the total price and quantity sold each day of each product is also discussed. Outlier capping and trimming are explored on the IQR analysis. A Shapiro-Wilk test will be used to determine whether or not the product's sample comes from a normal distribution. The outlier analysis is present in the appendix.

4.1.1.1 Product 1

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around May 2020 during the covid-19's first pandemic wave, stabilizing its value months after, as seen on figures 4.1 4.2 4.3 4.4. Overall, most of its values are suited above the mean. Only 2.89% and 2.16% of the price and quantity fields are considered outliers using the z_score method, and 2.89% and 3.16% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

Figure 4.1: Seasonality analysis

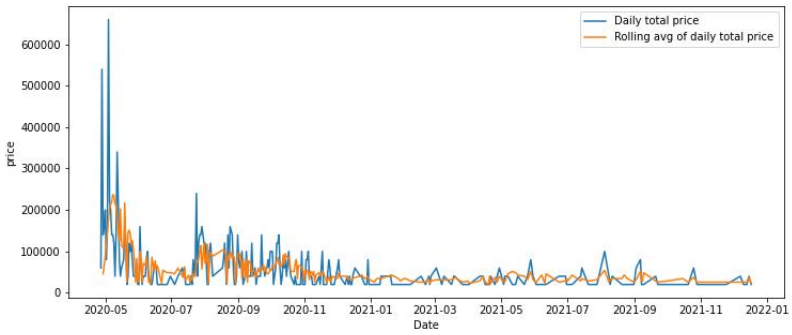


Figure 4.2: Moving average of the daily total price of products sold

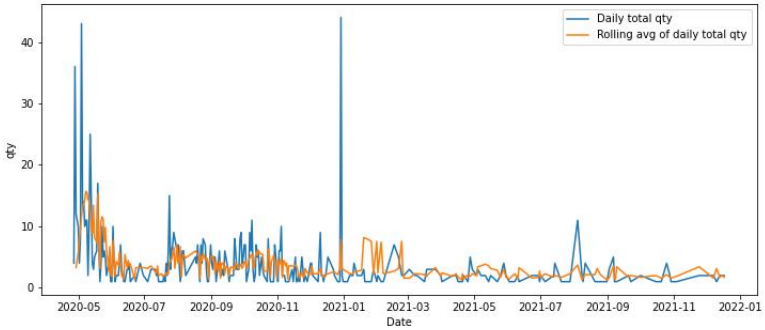


Figure 4.3: Moving average of the daily total quantity of products sold

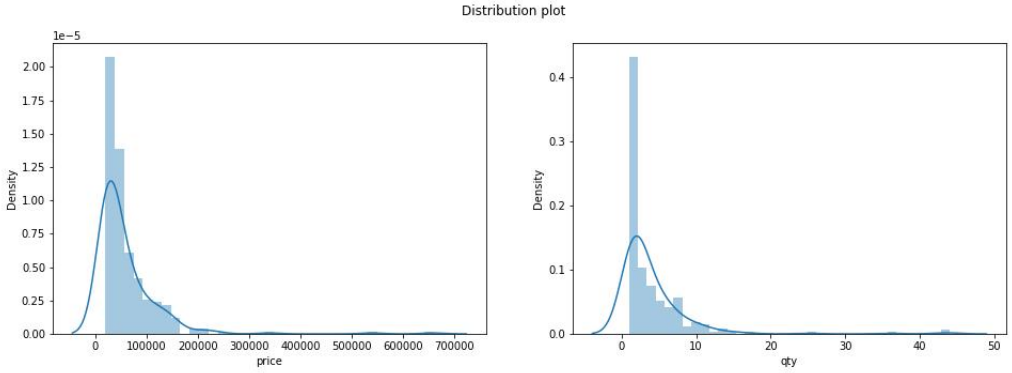


Figure 4.4: Distribution plot of daily data

4.1.1.2 Product 2

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around October 2020 during the covid-19's second pandemic wave, stabilizing its

value months after, as seen on figures 4.5 4.6 4.7 4.8. Overall, most of its values are suited above the mean. Only 3.04% and 2.17% of the price and quantity fields are considered outliers using the z_score method, and 3.47% and 3.04% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

Figure 4.5: Seasonality analysis

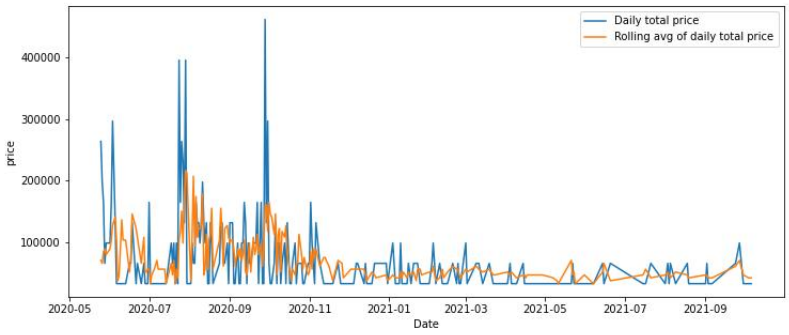


Figure 4.6: Moving average of the daily total price of products sold

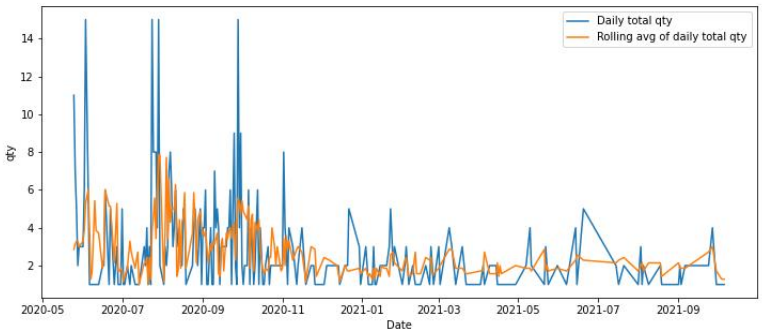


Figure 4.7: Moving average of the daily total quantity of products sold

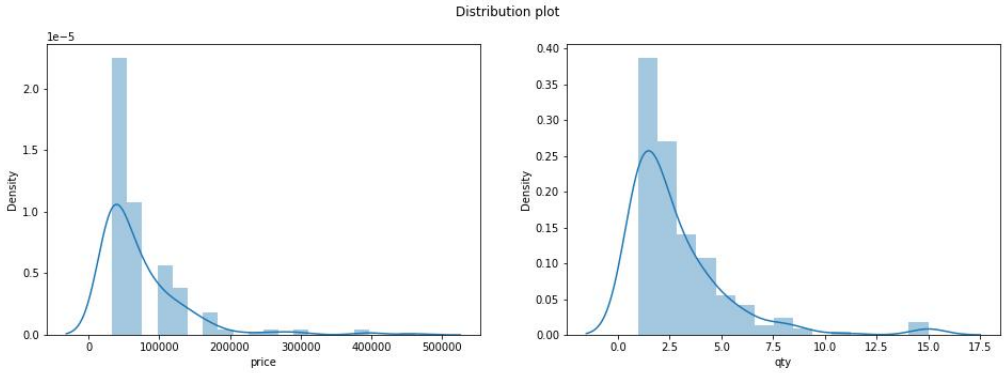


Figure 4.8: Distribution plot of daily data

4.1.1.3 Product 3

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around May 2020 during the covid-19's first pandemic wave, stabilizing its value months after, as seen on figures 4.9 4.10 4.11 4.12. Overall, most of its values are suited above the mean. Only 2.80% and 3.50% of the price and quantity fields are considered outliers using the z_score method, and 6.29% and 7.69% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

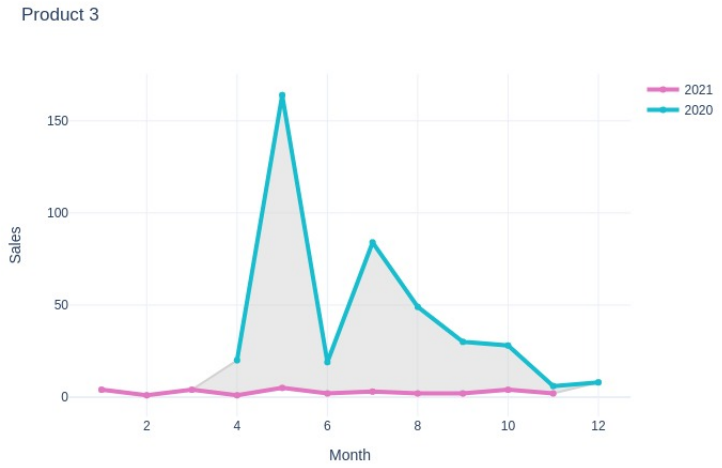


Figure 4.9: Seasonality analysis

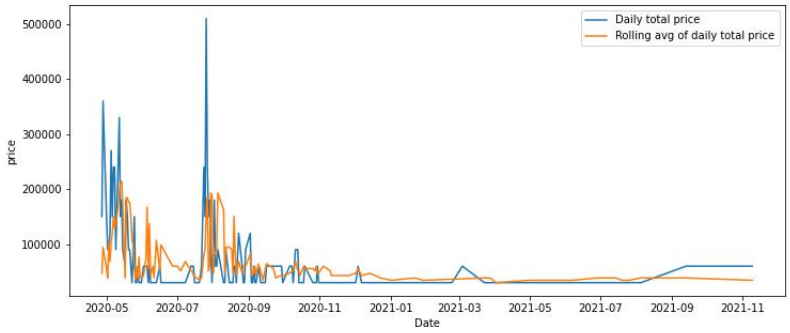


Figure 4.10: Moving average of the daily total price of products sold

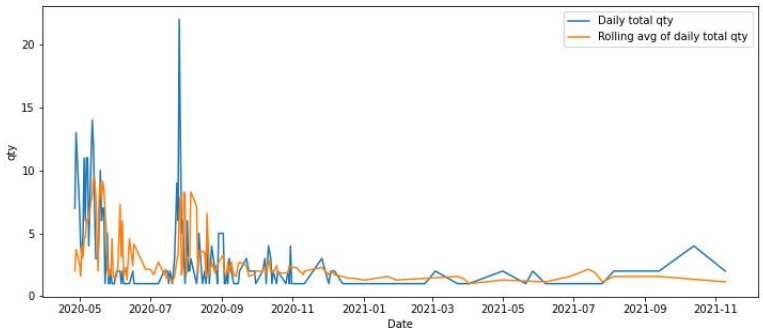


Figure 4.11: Moving average of the daily total quantity of products sold

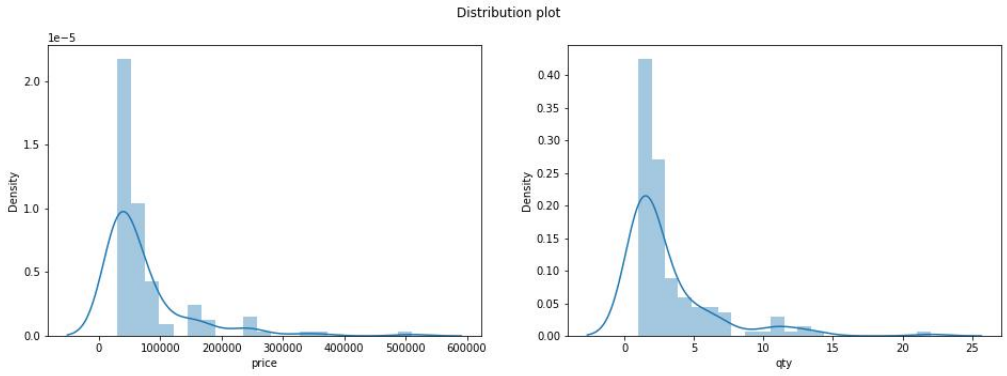


Figure 4.12: Distribution plot of daily data

4.1.1.4 Product 4

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around September 2020 during the covid-19's second pandemic wave, stabilizing its value months after, as seen on figures 4.13 4.14 4.15 4.16. Overall, most of its values are suited above the mean. Only 1.37% and 0% of the price and quantity fields are considered outliers using the z_score method, and 1.37% and 0% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

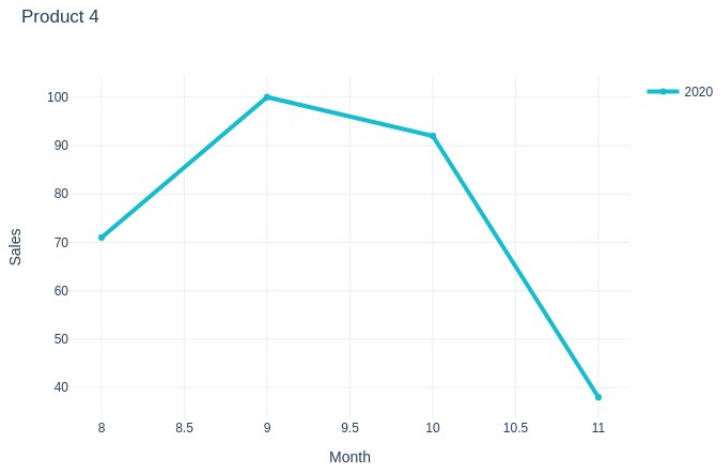


Figure 4.13: Seasonality analysis

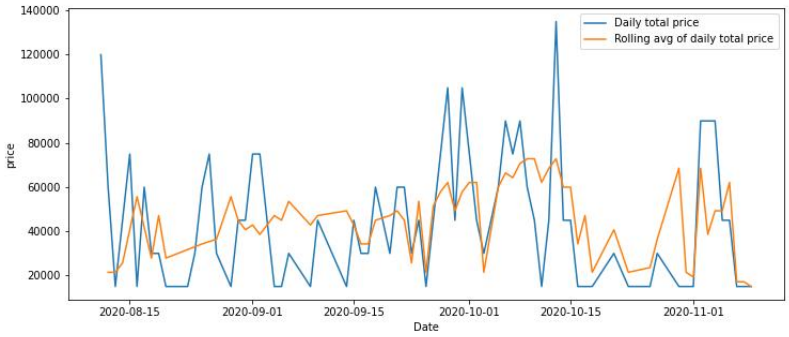


Figure 4.14: Moving average of the daily total price of products sold

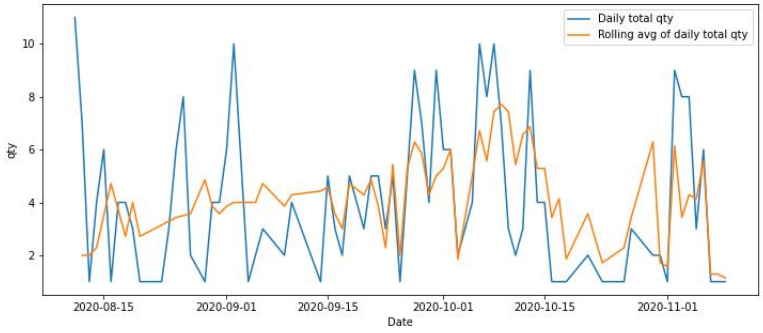


Figure 4.15: Moving average of the daily total quantity of products sold

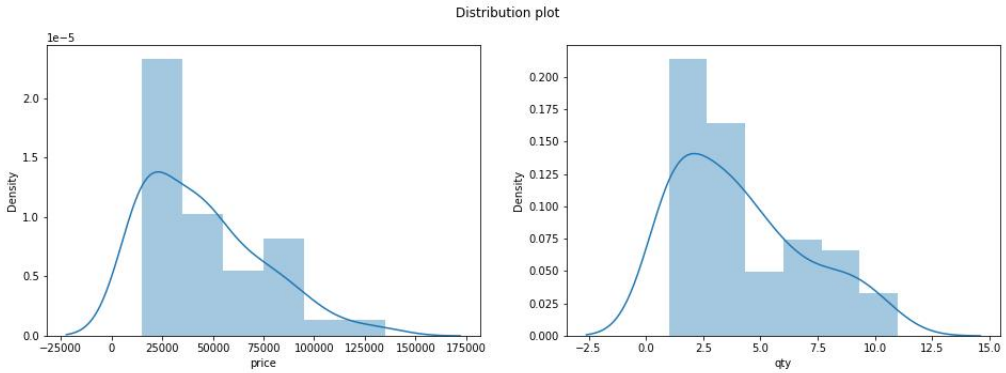


Figure 4.16: Distribution plot of daily data

4.1.1.5 Product 5

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around July 2020 at the end of covid-19's first pandemic wave and in April 2021, the

end of covid 19's second pandemic wave, stabilizing its value months after. Overall, most of its values are suited above the mean, as seen on figures 4.17 4.18 4.19 4.20. Only 4.51% and 4.51% of the price and quantity fields are considered outliers using the z_score method, and 24.81% and 24.81% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

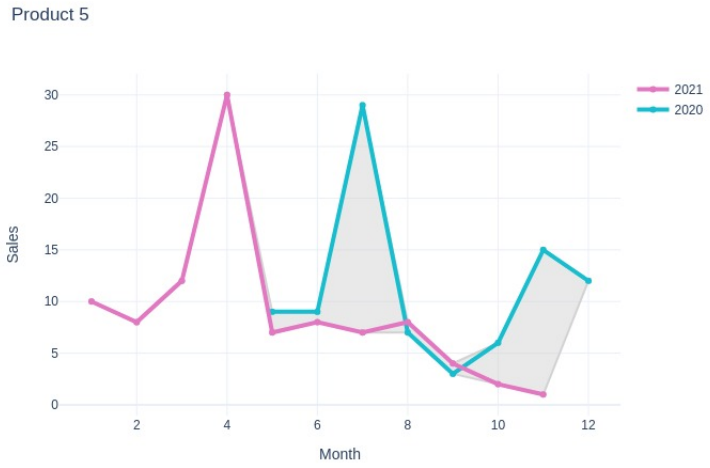


Figure 4.17: Seasonality analysis

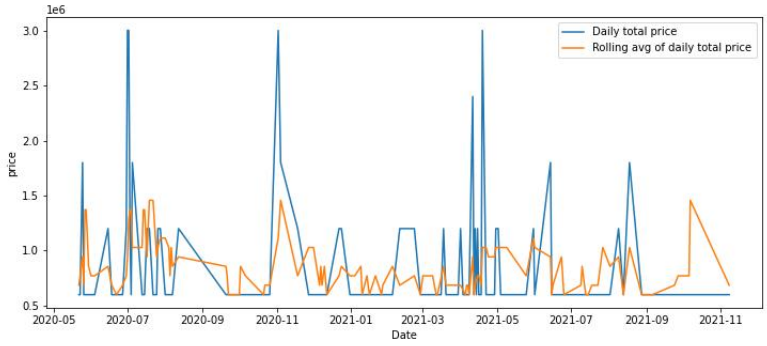


Figure 4.18: Moving average of the daily total price of products sold

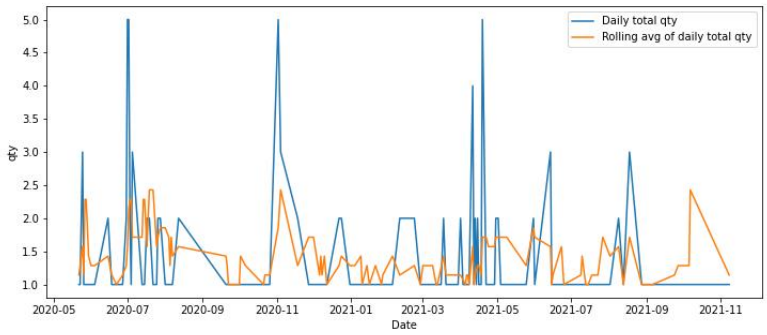


Figure 4.19: Moving average of the daily total quantity of products sold

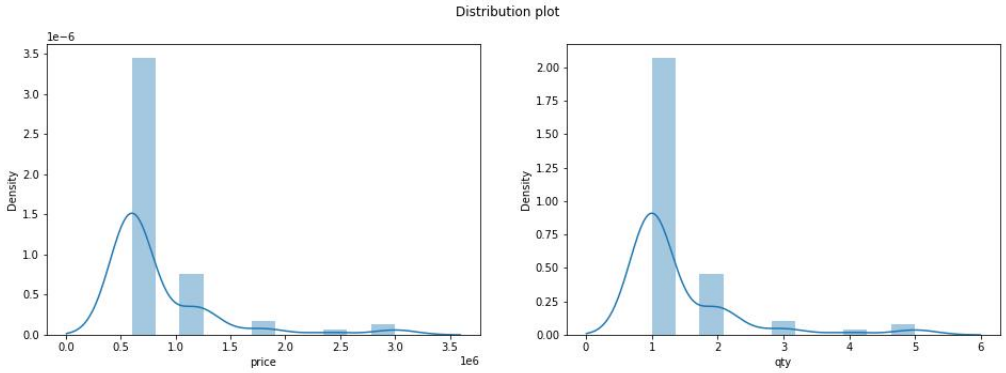


Figure 4.20: Distribution plot of daily data

4.1.1.6 Product 6

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around November 2020 during covid-19's second pandemic wave, stabilizing its value months after, as seen on figures 4.21 4.22 4.23 4.24. Overall, most of its values are suited above the mean. Only 2.74% and 2.74% of the price and quantity fields are considered outliers using the z_score method, and 12.32% and 20.55% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

Figure 4.21: Seasonality analysis

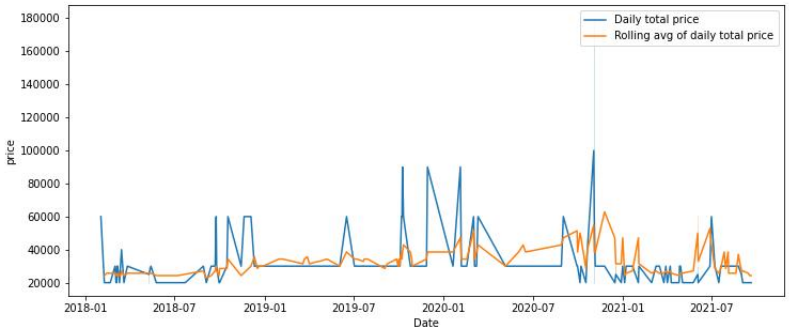


Figure 4.22: Moving average of the daily total price of products sold

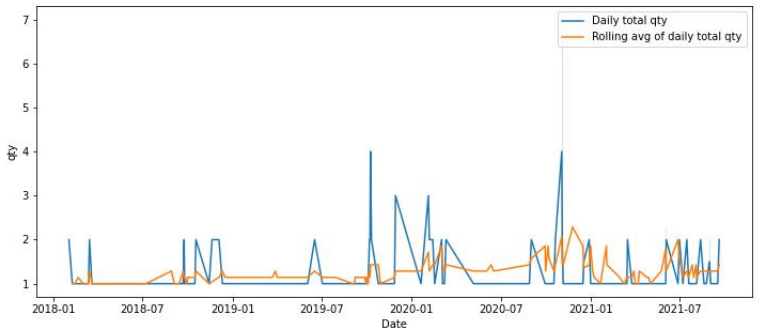


Figure 4.23: Moving average of the daily total quantity of products sold

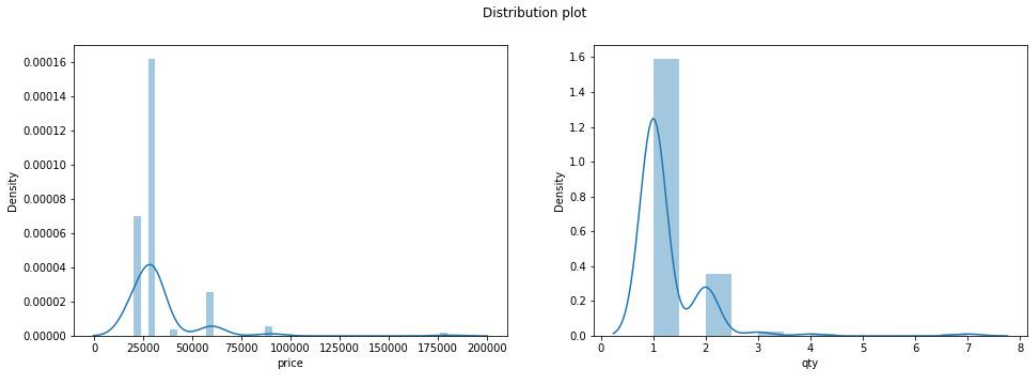


Figure 4.24: Distribution plot of daily data

4.1.1.7 Product 7

This product's total price and quantity sold daily trend is primarily high in 2020, having its maximum values around May 2020 during covid-19's second pandemic wave, stabilizing its value months after, as seen on figures 4.25 4.26 4.27 4.28. Overall, most of its values are suited above the mean. Only 4.42% and 1.77% of the price and quantity fields are considered outliers using the z_score method, and 32.74% and 23.01% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

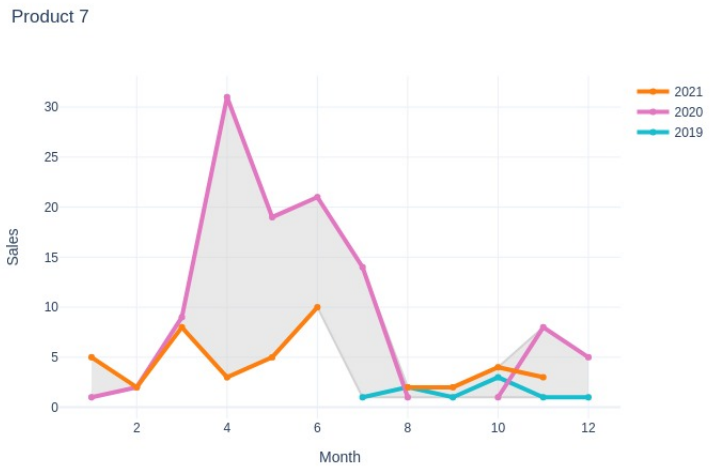


Figure 4.25: Seasonality analysis

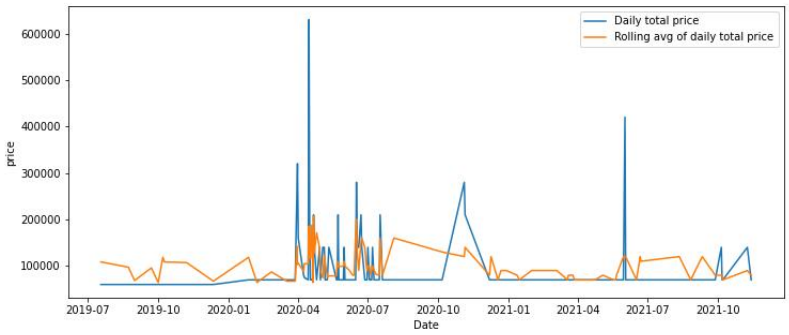


Figure 4.26: Moving average of the daily total price of products sold

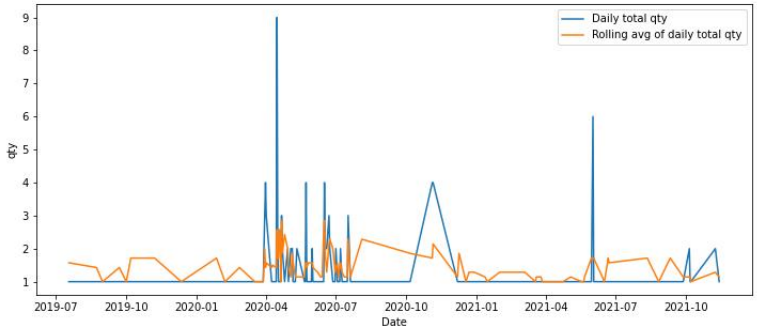


Figure 4.27: Moving average of the daily total quantity of products sold

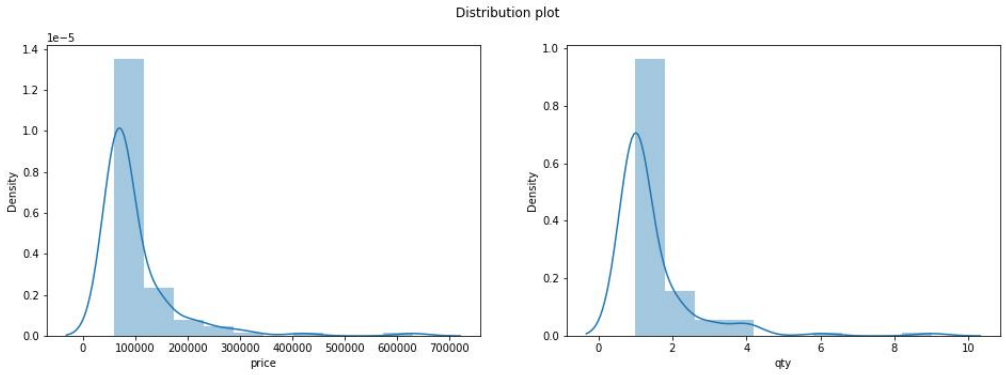


Figure 4.28: Distribution plot of daily data

4.1.1.8 Product 8

This product's total price and quantity sold daily trend is primarily high in 2018, having its maximum values around August 2018, stabilizing its value months after. Overall, most of its values are

suited above the mean, as seen on figures 4.29 4.30 4.31 4.32. Only 3.05% and 3.82% of the price and quantity fields are considered outliers using the z_score method, and 15.27% and 16.00% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

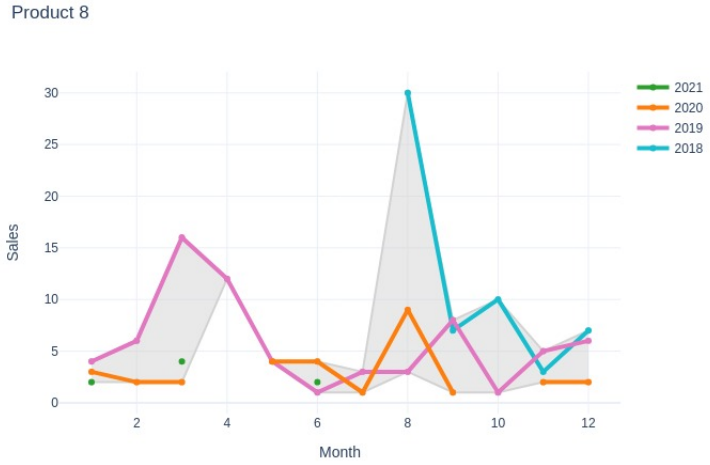


Figure 4.29: Seasonality analysis

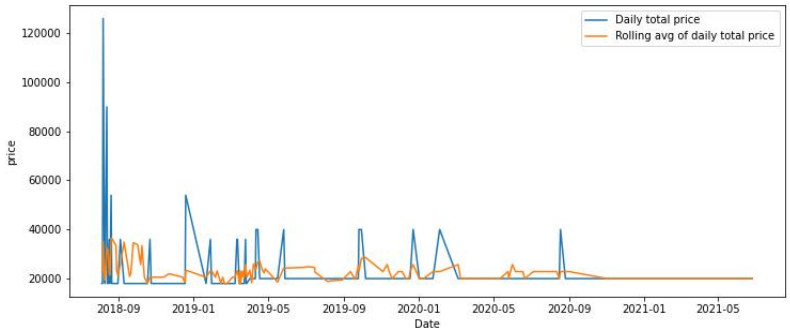


Figure 4.30: Moving average of the daily total price of products sold

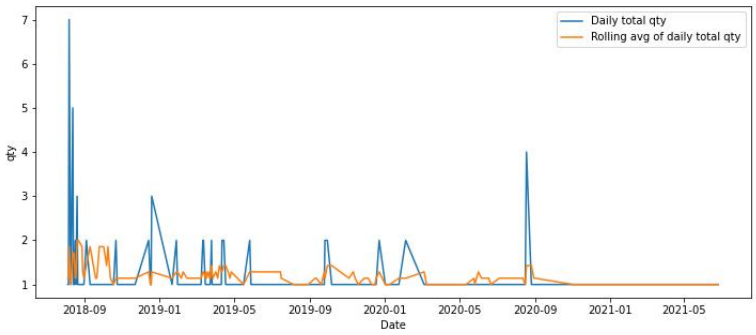


Figure 4.31: Moving average of the daily total quantity of products sold

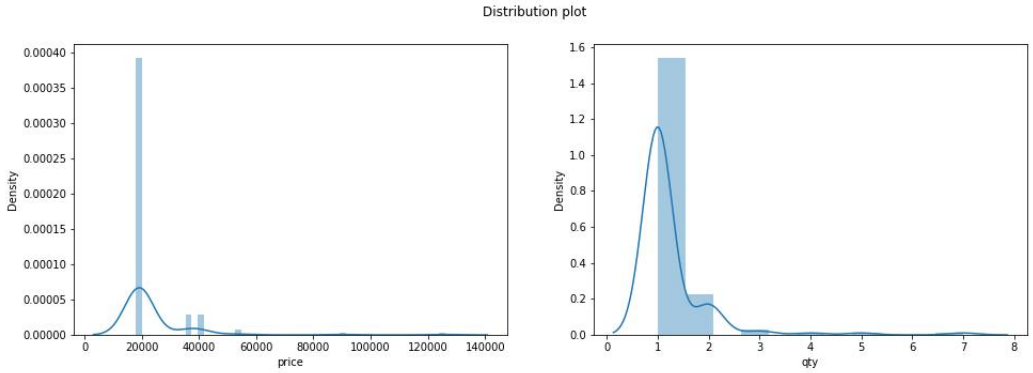


Figure 4.32: Distribution plot of daily data

4.1.1.9 Product 9

This product's total price and quantity sold daily trend is primarily high in 2018, having its maximum values around October 2018, stabilizing its value months after. Overall, most of its values are suited above the mean, as seen on figures 4.33 4.34 4.35 4.36. Only 3.79% and 3.79% of the price and quantity fields are considered outliers using the z_score method, and 24.05% and 12.66% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

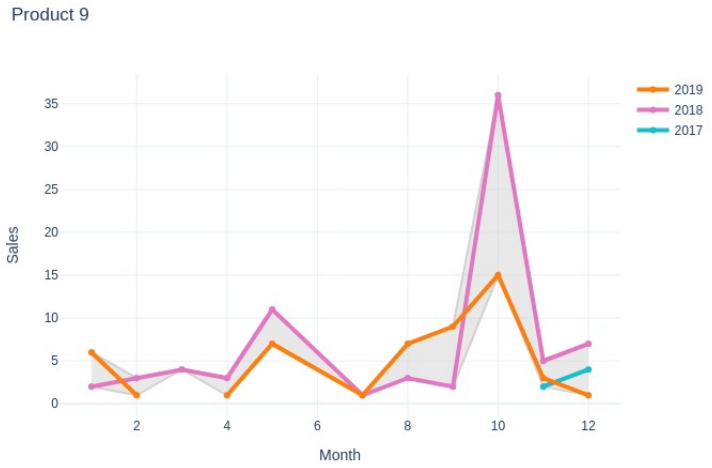


Figure 4.33: Seasonality analysis

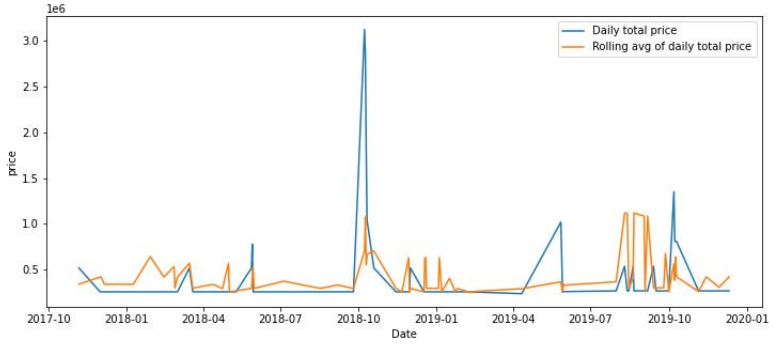


Figure 4.34: Moving average of the daily total price of products sold

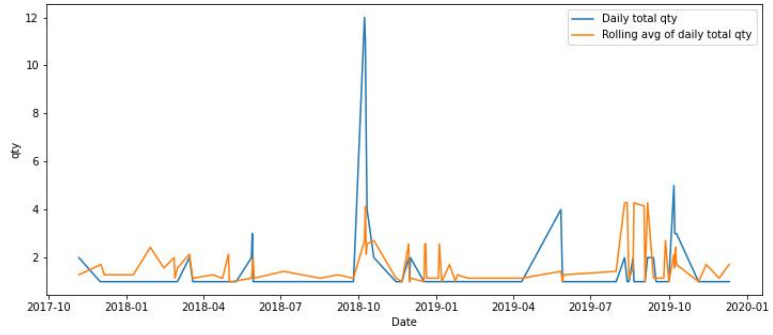


Figure 4.35: Moving average of the daily total quantity of products sold

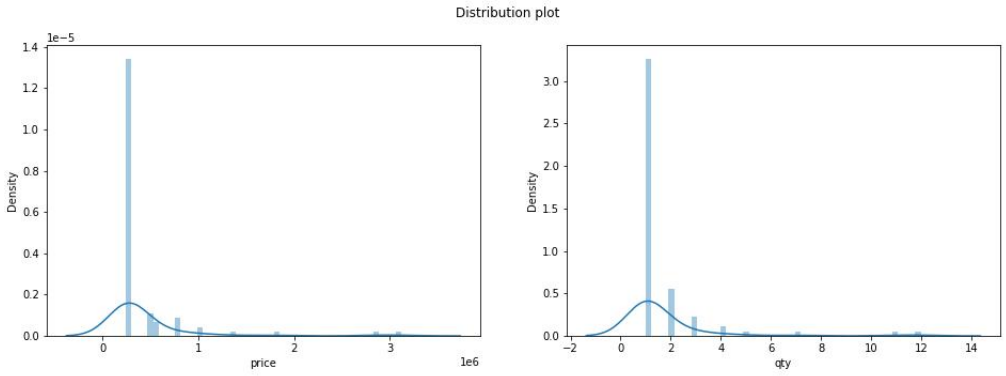


Figure 4.36: Distribution plot of daily data

4.1.1.10 Product 10

This product's total price and quantity sold daily trend is primarily high in 2018, having its maximum values around June 2021, stabilizing its value months after. Overall, most of its values are suited above the mean, as seen on figures 4.37 4.39 **??** 4.40. Only 3.96% and 1.98% of the price and quantity fields are considered outliers using the z_score method, and 23.76% and 18.81% using the IQR trimmed method. As the p-values of the Shapiro-Wilk tests are less than 0.05, the null hypothesis is rejected. There is sufficient evidence that the sample data does not come from a normal distribution, suggesting that the IQR outlier method is the better option.

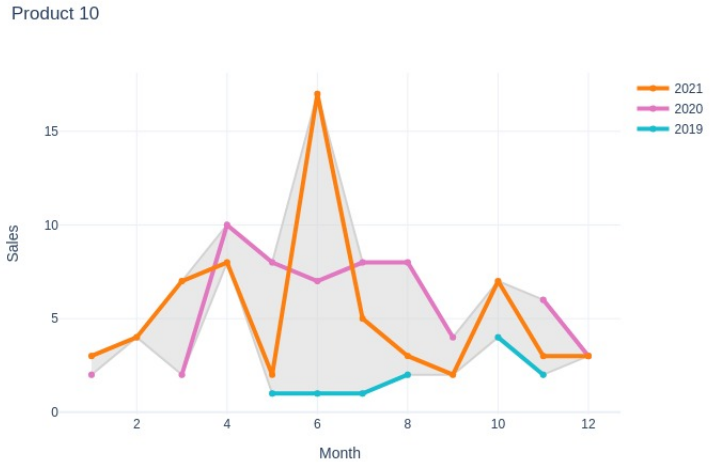


Figure 4.37: Seasonality analysis

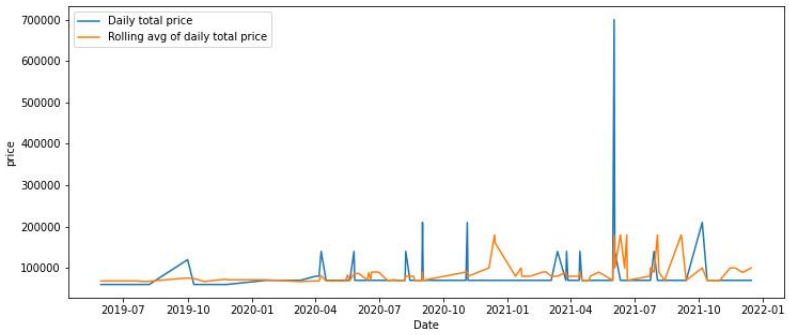


Figure 4.38: Moving average of the daily total price of products sold

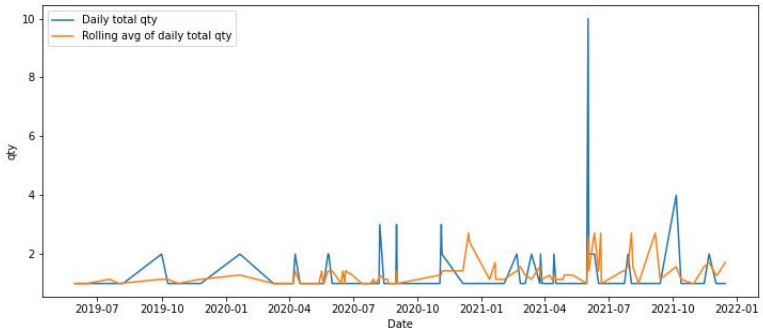


Figure 4.39: Moving average of the daily total quantity of products sold

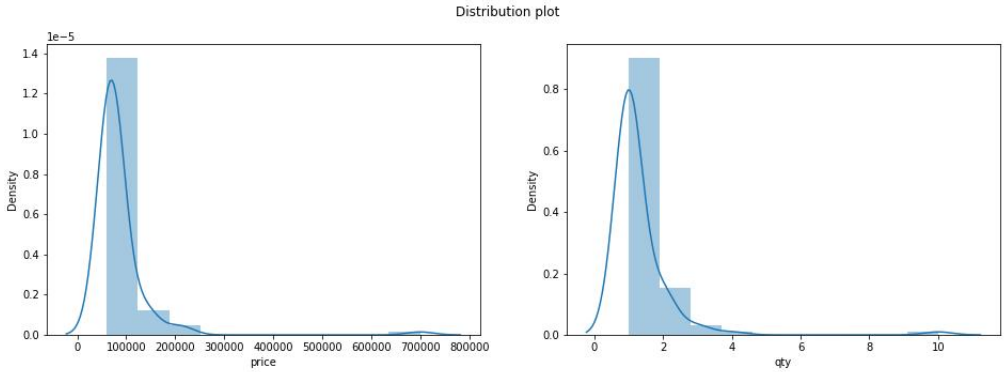


Figure 4.40: Distribution plot of daily data

4.1.2 Store

The store's monthly analysis consists of paid orders 4.48, products sold 4.42, the percentage of discount of the order's total value 4.48, and the amount of each order's status 4.44. As the years

go by, globally, the total number of paid orders and products sold also increase, incrementing a lot since the first covid 19 pandemic (around the 2nd quarter of 2020). The percentage of discount in orders total has increased between 2018 and 2020. This percentage has been decreasing during 2021, with a tendency to increase lately (October 2021). Finally, the number of canceled and abandoned orders has been increasing slowly lately while the number of paid orders spiked after the first covid 19's first pandemic and has remained steady.

Figure 4.41: Paid orders

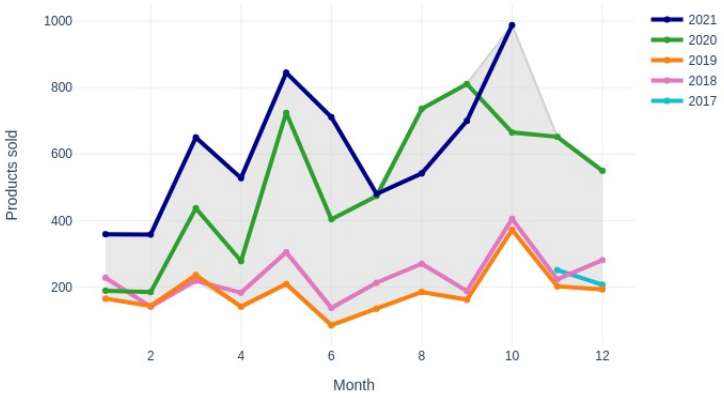


Figure 4.42: Products sold

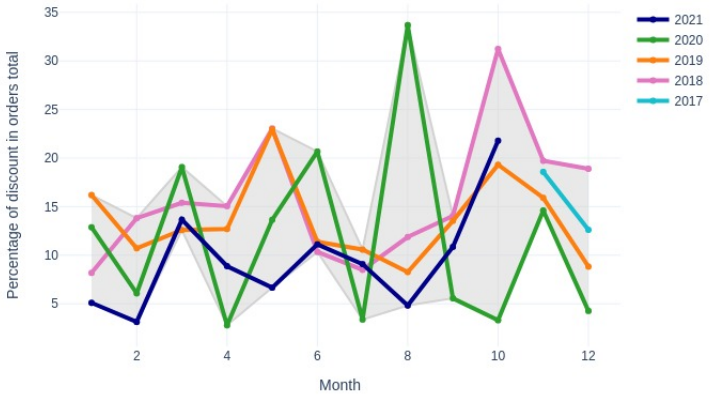


Figure 4.43: Percentage of discount

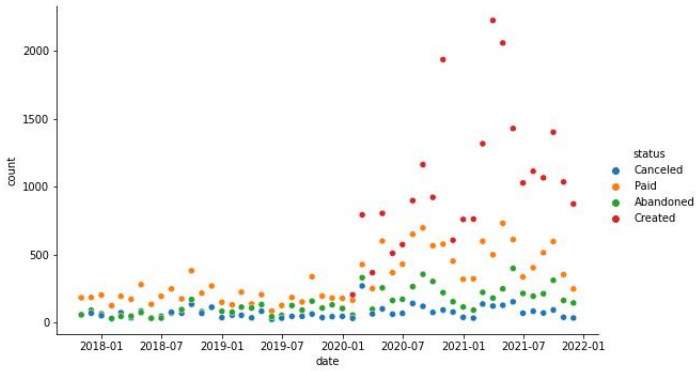


Figure 4.44: Type of order by status

The availability of data is present in almost all tables. The tables that do not have such information are related to the order customers' information, the stock history, and payment information and product information. By comparing the correlation of data of all products vs. 10 most sold vs. 10 less sold products, the data about the order product's price is correlated to the order customer's and if the product is featured or not on the store's page, while the order product's qty is mostly correlated to the order customer's data, as seen of figures 4.45 4.46 4.47 4.48

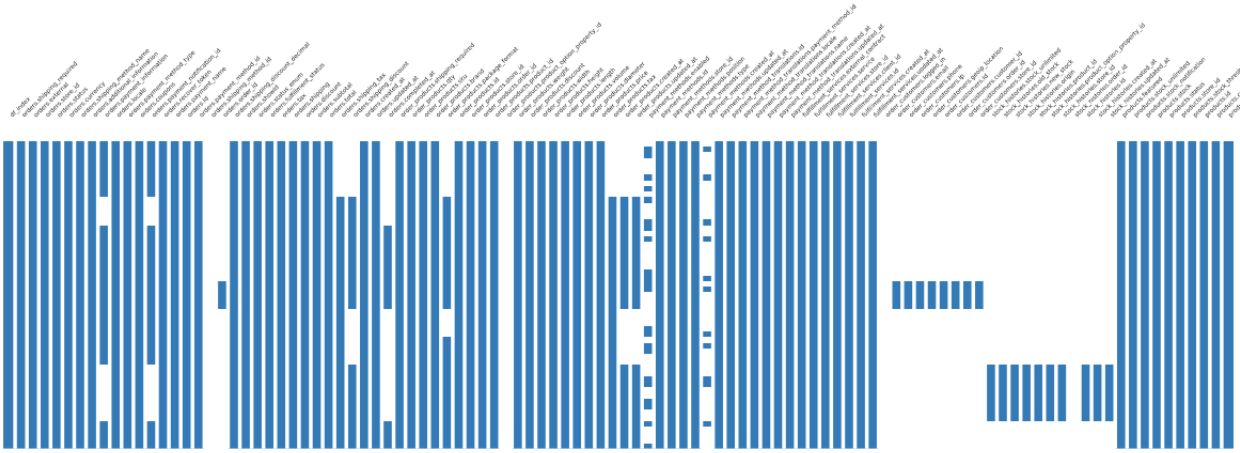


Figure 4.45: Data availability

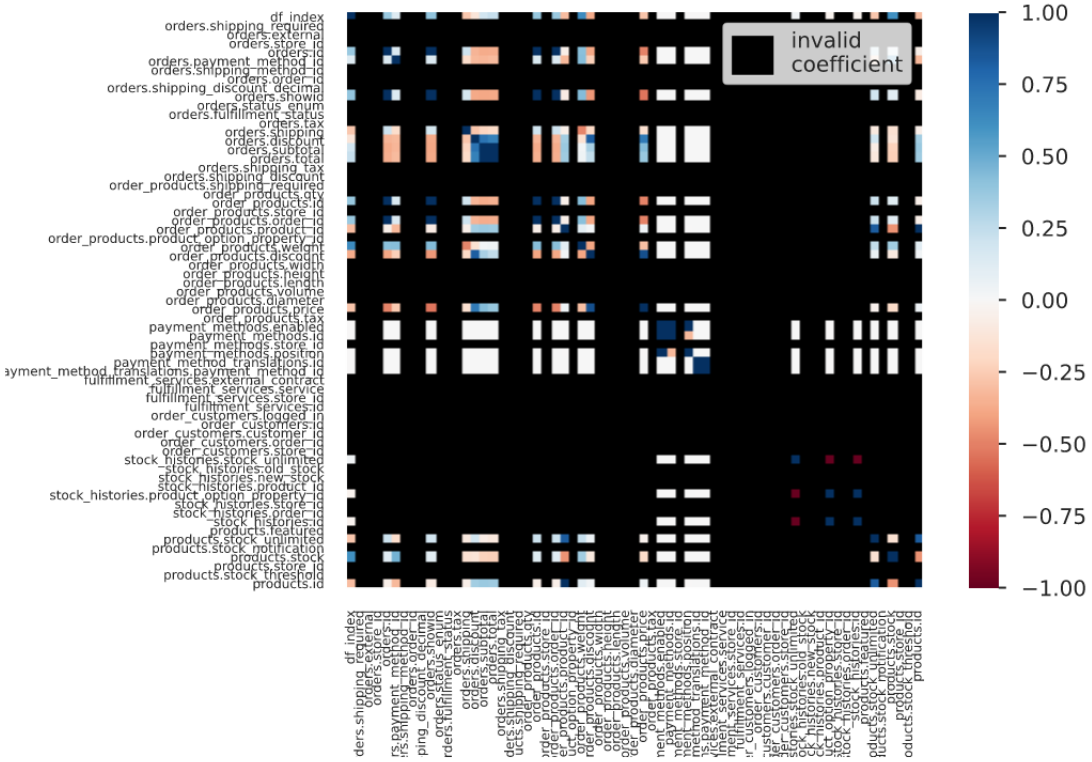


Figure 4.46: Correlation for 10 less sold products

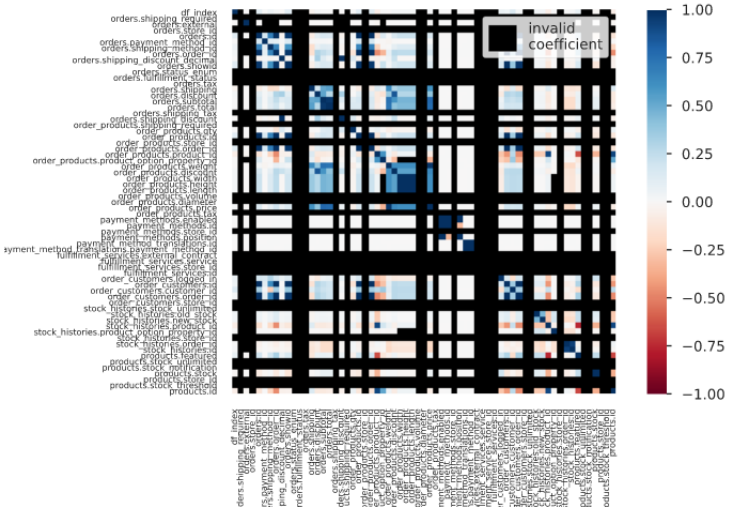


Figure 4.47: Correlation for 10 most sold products

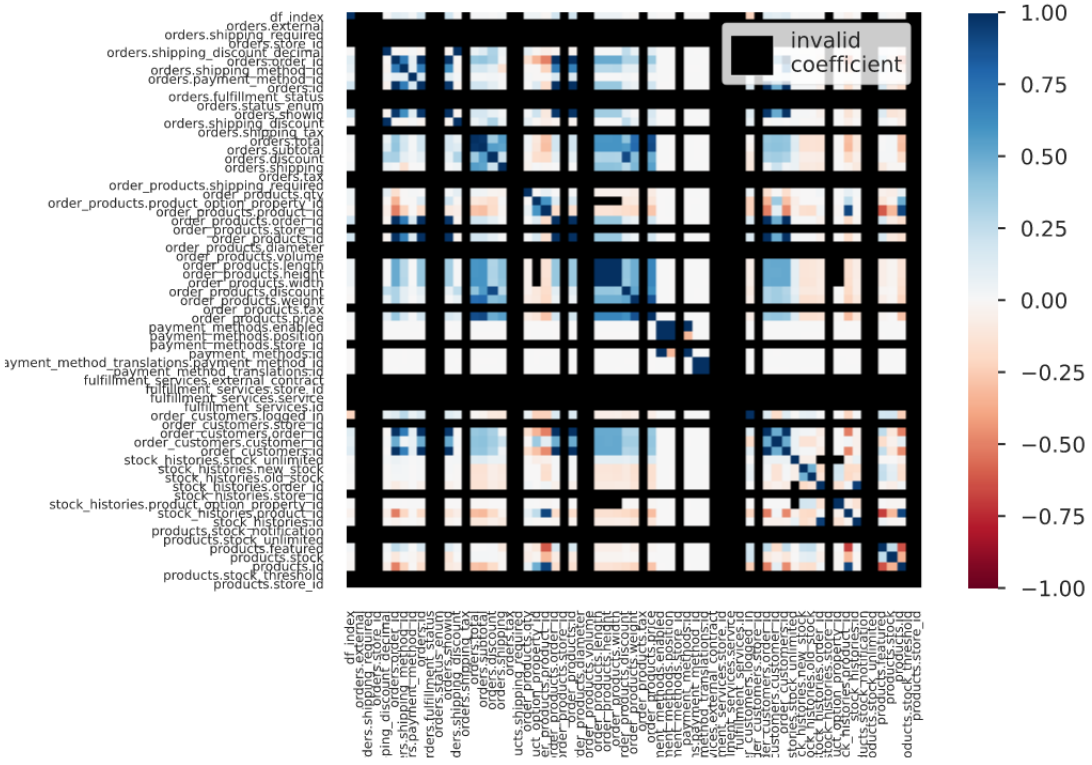


Figure 4.48: Correlation for all sold products

4.2 Data cleaning

Data cleaning is the process of correcting or removing missing, duplicated, outliers, noise, and inconsistent values in a dataset. By neglecting the data cleaning of the dataset when combining data from multiple sources (tables) 3.1, the performance of the sales forecasting models to be implemented can be compromised [10].

- Missing values - As it is infeasible to predict the missing values, especially the product information, missing values were only analysed on 4.1.2, and they will not be replaced.
- Duplicated values - apply a distinct select on the dataset's queries
- Outliers - As explained on 4.1.1 subsection, the IQR and Z_score methods are used to remove outliers during the modelling phase.
- Noise - Only real store's data are used on the dataset, noise data can be filtered by accessing the subscriptions table and checking the store's plan 3.1.
- Inconsistent values - As redshift has already implemented a python library to connect to its services, the datatypes are already converted to its SQL equivalent, so there is no need to convert column types and fix Not a Number (NaN) and Not Available (NA) values.

4.3 Data Transformation

Data transformation is converting raw data into a structure that will better suit the forecasting models to be implemented. This phase consists of generalisation, normalisation, data aggregation, and features creation [13].

In order to access the data better and to predict the total sale quantity of a product in a given day, the table "order_data_by_day" was created 4.49. This table aggregates the data later merged with the final dataset.

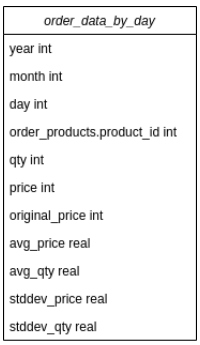


Figure 4.49: Order_data_by_day Entity Relationship Diagram

- year - The year of the date
- month - The month of the date

- day - The day of the date
- order_products.product_id - Unique identifier of the product that will be predicted
- qty - Total quantity of product sold in a given date
- price - Total price of product sales in a given date
- original_price - The original price of the product
- avg_price - Average of the total price of product sales
- avg_qty - Average of the total quantity of product sold in a given date
- stddev_price - Standard deviation total price of product sales
- stddev_qty - Standard deviation of the total quantity of product sold in a given date

The data present on 4.49 and 3.1 were transformed into materialized views (present on redshift). A materialized view contains a precomputed result set, based on an SQL query over one or more base tables. Materialized views are beneficial for speeding up predictable and repeated queries. Instead of performing resource-intensive queries against large tables (such as aggregates or multiple joins) [2].

The dataset used in the feature creation stage was the set of 20 products discussed in 4.1.1. The following set of features was created taking also into account the ones mentioned on the table 2.1:

- visits
- product information
- stock history
- moving average of sales
- SKU information
- average, sum, trend, the standard deviation of the past 4-12 weeks (sales and stocks)
- date (sales, stocks)
- seasonality
- payment information
- fulfillment information

Besides these features, the python library Featuretools was also used to create automatic features [3]. This library uses Deep Feature Synthesis (DFS), an automated method for performing feature engineering on relational and temporal data. The data used to generate these automatic features are the ones represented on 3.1. In total, around 4200 features were created.

In order to select the best features that are important to the variable that we want to predict (total quantity of product sold in a given date), the Sequential Forward Floating Selection (SFFS),

Sequential Forward Selection (SFS), and Recursive Feature Elimination (RFE) wrapper methods were used. The best features selected by each method are present in the tables 4.4 4.3.

Table 4.3: RFE best features

Feature name
orders.coupons
orders.payment_method_id
orders.shipping_discount
MAX(order_products.order_products.discount)_x
MODE(order_products.order_products.sku)_x
NUM_UNIQUE(order_products.YEAR(order_products.updated_at))_x
orders.orders.coupons
orders.orders.shipping_discount_decimal
orders.MAX(order_products.order_products.discount)
orders.MODE(order_products.order_products.sku)
products.COUNT(order_products)
products.MEAN(order_products.order_products.discount)
products.MIN(order_products.order_products.discount)
products.MIN(order_products.order_products.weight)
products.MODE(order_products.order_products.sku)
products.SKEW(order_products.order_products.qty)
products.STD(order_products.order_products.price)
products.SUM(order_products.order_products.discount)
products.SUM(order_products.order_products.price)
products.SUM(order_products.order_products.product_option_property_id)
products.SUM(order_products.order_products.shipping_required)
products.MONTH(products.updated_at)
products.WEEKDAY(products.updated_at)
products.YEAR(products.updated_at)
MEAN(order_products.order_products.discount)_y
MODE(order_products.order_products.sku)_y
STD(order_products.order_products.product_option_property_id)_y
MONTH(products.created_at)
MAX(order_products.orders.orders.order_id)
MAX(order_products.orders.orders.showid)
MEAN(order_products.orders.orders.discount)
MEAN(order_products.orders.orders.order_id)
MEAN(order_products.orders.orders.payment_method_id)
MEAN(order_products.orders.orders.shipping_method_id)
MIN(order_products.orders.orders.total)
MODE(order_products.DAY(order_products.created_at))_y
MODE(order_products.DAY(order_products.updated_at))_y
MODE(order_products.WEEKDAY(order_products.created_at))_y
SKEW(order_products.orders.orders.order_id)
SKEW(order_products.orders.orders.shipping_discount_decimal)
STD(order_products.orders.orders.discount)
STD(order_products.orders.orders.shipping_discount_decimal)
STD(order_products.orders.orders.subtotal)
STD(order_products.orders.orders.total)
SUM(order_products.orders.orders.discount)
SUM(order_products.orders.orders.fulfillment_status)
SUM(order_products.orders.orders.shipping_discount_decimal)
SUM(order_products.orders.orders.shipping_method_id)
SUM(order_products.orders.orders.showid)
SUM(order_products.orders.orders.subtotal)

Table 4.4: SFS and SFFS best features

Aggregate cv score	Feature name
0.1761068664805 71	order_products.id
0.3188667076040938	stddev_qty
0.3591155936311019	stddev_price
0.3908615901869402	avg_qty
0.4099526444532799	STDorder_products.orders.orders.subtotal
0.4198009518020641	NUM_UNIQUEorder_products.order_products.sku_y
0.4346594684784757	products.STDorder_products.order_products.qty
0.4435273105356084	products.SKEWorder_products.order_products.price
0.4500128091823094	products.MODEorder_products.order_products.sku
0.4563519663644746	products.MAXorder_products.order_products.price

Table 4.4: SFS and SFFS best features

Aggregate cv score	Feature name
0.4626927469963249	orders.YEARorders.completed_at
0.4660313140521667	orders.WEEKDAYorders.completed_at
0.4701152180254608	orders.MONTHorders.created_at
0.4765887555388102	orders.DAYorders.updated_at
0.4809672155017277	orders.DAYorders.completed_at
0.4842072133608847	orders.SUMorder_products.order_products.discount
0.488068059562051	orders.STDorder_products.order_products.qty
0.4913056812880054	orders.STDorder_products.order_products.length
0.493813181357241	orders.MODEorder_products.order_products.sku
0.4964264099088126	orders.MEANorder_products.order_products.product_option_property_id
0.4984401382113859	orders.MAXorder_products.order_products.qty
0.5004336080338538	orders.MAXorder_products.order_products.price
0.504213485727092	orders.orders.created_at
0.5059090113498244	orders.orders.updated_at
0.507298897148361	orders.orders.subtotal
0.5098705977693967	orders.orders.coupons
0.5122191713776061	MONTHorder_products.updated_at
0.5139783259526796	MONTHorder_products.created_at
0.5149975703268792	order_products.created_at
0.5159573910092197	order_products.updated_at
0.5168983834635987	order_products.price
0.522190779315448	order_products.tax
0.5230897648197324	order_products.sku
0.523914474990607	orders.id
0.5245472770948525	MODEstock_histories.WEEKDAYstock_histories.created_at
0.5251408410744598	SUMorder_products.products.products.stock
0.5256773783902551	SKEWorder_products.products.products.stock
0.5262026860663366	MODEorder_products.YEARorder_products.created_at_x
0.5267021129014834	YEARorders.created_at
0.5271576823093118	WEEKDAYorders.updated_at
0.5276190029877619	WEEKDAYorders.created_at
0.528365428768566	COUNTorder_customers
0.529622522185125	SUMorder_products.order_products.weight_x
0.5303233035229333	STDorder_products.order_products.product_option_property_id_x
0.5308478986266627	STDorder_products.order_products.price_x
0.5313273879931852	MINorder_products.order_products.qty_x
0.5334990758777471	MINorder_products.order_products.discount_x
0.5342354626462907	MAXorder_products.order_products.discount_x
0.5357421530860035	orders.shipping_discount_decimal
0.5366348876151339	orders.shipping_method_name

4.4 Modelling

Modelling is the final and crucial step in determining the quality and accuracy of future predictions in new situations [53]. In this case, the modelling phase is used to:

- Find the best parameters
- Find the best train/test split on data
- Find the best outlier strategy

Due to time concerns, only machine learning models were implemented: Logistic Regression, Gaussian Naive-Bayes, Random Forest, and Adaboost 2.1.2. Three split strategies were adopted:

- Leave the dataset as it comes from the query
- Sort the dataset by the date of order creation
- Sort the dataset by the quantity of product sold

- Sort the dataset by the order_products.product_id

Furthermore, three outlier removal strategies on the training data were applied:

- Leave the dataset as it is
- Using the IQR method
- Using the Z_score method

Finally, the final dataset contains 1719 entries of the 4.49 table from 4 different stores. The train/test split adopted was 70/30. The leading e-commerce stores analysed are home delivery of gifts; flower shop; consumer electronics and entertainment, and sports equipment stores. The metrics used were: MSE, R2, MAPE, MAE, RMSE, and Accuracy. The model tuning phase first fitted the best parameters for the best MSE value and then the best MAPE. The features used were the RFE's best 50 features. The SFS's best 50 features were not explored due to technical limitations. The table with the results and the real prediction plot will be discussed for each sorting method. Additional information about the individual results of each algorithm is present in the appendix.

4.4.1 Unsorted dataset

The results when using an unsorted dataset will be presented and discussed in this subsection. In this case, the best model that produced the most accurate results was the Gaussian Naive Bayes when using the dataset. On the other hand, its predictions are farther from the actual values, as seen in the table 4.5 and the figure 4.50.

Table 4.5: Results

outlier_detection	order_by	Model	MSE	R2	MAPE	MAE	RMSE	Accuracy
normal	normal	gaussiannb_model	3.169	-5.561	0.169	0.424	1.78	0.74
iqr	normal	gaussiannb_model	4.795	-8.927	0.586	0.841	2.19	0.626
z_score	normal	gaussiannb_model	4.795	-8.927	0.586	0.841	2.19	0.626
iqr	normal	lr_model	0.907	-0.878	0.768	0.833	0.952	0.194
normal	normal	lr_model	10.698	-21.15	2.508	2.659	3.271	0.043
normal	normal	rf_model	123.335	-254.372	9.654	11.076	11.106	0.0
normal	normal	adaboost_model	469.256	-970.617	18.745	21.651	21.662	0.0
iqr	normal	rf_model	113.93	-234.898	9.302	10.651	10.674	0.0
iqr	normal	adaboost_model	426.953	-883.028	17.887	20.651	20.663	0.0
z_score	normal	rf_model	113.93	-234.898	9.302	10.651	10.674	0.0
z_score	normal	lr_model	32.419	-66.124	5.009	5.651	5.694	0.0
z_score	normal	adaboost_model	426.953	-883.028	17.887	20.651	20.663	0.0

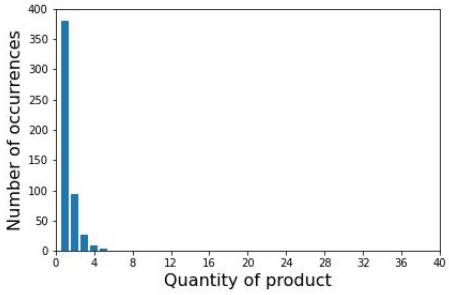


Figure 4.50: Real results

4.4.2 Sort the dataset by the date of order creation

The results when sorting the dataset by the date of order creation will be presented and discussed in this subsection. In this case, the best model that produced the most accurate results was the Gaussian Naive Bayes when using the dataset and the Logistic Regression model when using both outlier removal approaches. In the end, the Logistic Regression models are better than the Gaussian Naive Bayes due to having minimal errors towards the actual prediction values, as seen in the table 4.6 and the figure 4.51.

Table 4.6: Results

outlier_detection	order_by	Model	MSE	R2	MAPE	MAE	RMSE	Accuracy
normal	orders.created_at	gaussiannb_model	3.126	-5.128	0.155	0.413	1.768	0.748
iqr	orders.created_at	lr_model	0.632	-0.239	0.139	0.349	0.795	0.748
z_score	orders.created_at	lr_model	0.632	-0.239	0.139	0.349	0.795	0.748
iqr	orders.created_at	gaussiannb_model	5.322	-9.433	0.641	0.888	2.307	0.618
z_score	orders.created_at	gaussiannb_model	5.322	-9.433	0.641	0.888	2.307	0.618
normal	orders.created_at	lr_model	0.934	-0.831	0.774	0.845	0.966	0.188
normal	orders.created_at	rf_model	93.655	-182.603	8.471	9.651	9.678	0.0
normal	orders.created_at	adaboost_model	1494.422	-2928.691	33.439	38.651	38.658	0.0
iqr	orders.created_at	rf_model	245.469	-480.222	13.637	15.651	15.667	0.0
iqr	orders.created_at	adaboost_model	1494.422	-2928.691	33.439	38.651	38.658	0.0
z_score	orders.created_at	rf_model	321.835	-629.931	15.588	17.921	17.94	0.0
z_score	orders.created_at	adaboost_model	1494.422	-2928.691	33.439	38.651	38.658	0.0

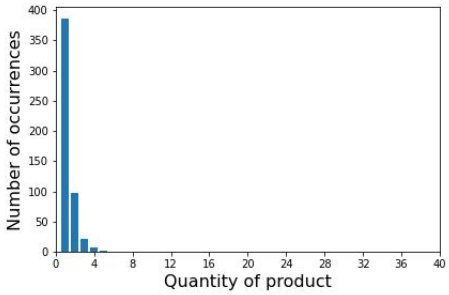


Figure 4.51: Real results

4.4.3 Sort the dataset by the quantity of product sold

The results when sorting the dataset by the number of products sold will be presented and discussed in this subsection. In this case, the best model that produced the most accurate results was the Gaussian Naive Bayes when using the dataset and the Logistic Regression model using the z_score outlier removal method. The Logistic Regression model is better than the Gaussian Naive Bayes due to having minimal errors towards the actual prediction values, as seen in the table 4.7 and the figure 4.52.

Table 4.7: Results

outlier_detection	order_by	Model	MSE	R2	MAPE	MAE	RMSE	Accuracy
normal	order_products.product_id	gaussiannb_model	0.597	-0.249	0.142	0.345	0.773	0.738
z_score	order_products.product_id	lr_model	0.597	-0.249	0.142	0.345	0.773	0.738
iqr	order_products.product_id	gaussiannb_model	6.442	-12.479	0.334	0.585	2.538	0.711
z_score	order_products.product_id	gaussiannb_model	6.442	-12.479	0.334	0.585	2.538	0.711
iqr	order_products.product_id	lr_model	7.527	-14.75	2.435	2.671	2.744	0.017
normal	order_products.product_id	rf_model	77.169	-160.474	7.661	8.754	8.785	0.0
normal	order_products.product_id	lr_model	22.147	-45.343	4.149	4.659	4.706	0.0
normal	order_products.product_id	adaboost_model	427.109	-892.718	17.877	20.655	20.667	0.0
iqr	order_products.product_id	rf_model	196.457	-410.084	12.155	13.992	14.016	0.0
iqr	order_products.product_id	adaboost_model	427.109	-892.718	17.877	20.655	20.667	0.0
z_score	order_products.product_id	rf_model	825.403	-1726.143	24.795	28.721	28.73	0.0
z_score	order_products.product_id	adaboost_model	427.109	-892.718	17.877	20.655	20.667	0.0

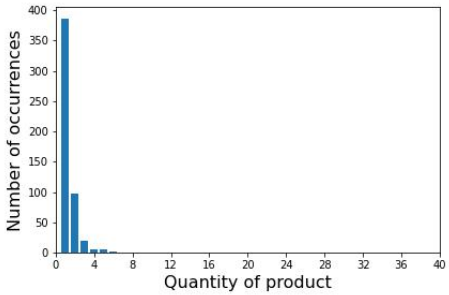


Figure 4.52: Real results

4.4.4 Sort the dataset by the order_products.product_id

The results when sorting the dataset by the order_products.product_id will be presented and discussed in this subsection. In this case, the best model that produced the most accurate results was the Gaussian Naive Bayes when using the dataset and the Logistic Regression model using the z_score outlier removal method. Both models produce the same error compared to the actual prediction values, as seen in the table 4.8 and the figure 4.8.

Table 4.8: Results

outlier_detection	order_by	Model	MSE	R2	MAPE	MAE	RMSE	Accuracy
normal	order_products.product_id	gaussiannb_model	0.597	-0.249	0.142	0.345	0.773	0.738
z_score	order_products.product_id	lr_model	0.597	-0.249	0.142	0.345	0.773	0.738
iqr	order_products.product_id	gaussiannb_model	6.442	-12.479	0.334	0.585	2.538	0.711
z_score	order_products.product_id	gaussiannb_model	6.442	-12.479	0.334	0.585	2.538	0.711
iqr	order_products.product_id	lr_model	7.527	-14.75	2.435	2.671	2.744	0.017

Table 4.8: Results

outlier_detection	order_by	Model	MSE	R2	MAPE	MAE	RMSE	Accuracy
normal	order_products.product_id	rf_model	77.169	-160.474	7.661	8.754	8.785	0.0
normal	order_products.product_id	lr_model	22.147	-45.343	4.149	4.659	4.706	0.0
normal	order_products.product_id	adaboost_model	427.109	-892.718	17.877	20.655	20.667	0.0
iqr	order_products.product_id	rf_model	196.457	-410.084	12.155	13.992	14.016	0.0
iqr	order_products.product_id	adaboost_model	427.109	-892.718	17.877	20.655	20.667	0.0
z_score	order_products.product_id	rf_model	825.403	-1726.143	24.795	28.721	28.73	0.0
z_score	order_products.product_id	adaboost_model	427.109	-892.718	17.877	20.655	20.667	0.0

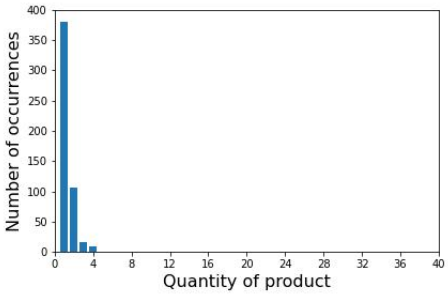


Figure 4.53: Real results

Chapter 5

Conclusions and Future work

In this chapter, the limitations, the future work, and the conclusions of the work done on this thesis will be described in better detail.

5.1 Limitations and Future Work

Some aspects can be addressed for all the limitations and criticism of the development phase to achieve better forecasting results. When migrating the data to an ETL platform to better process the dataset, its main limitation was not only the amount of data to be migrated but also its access time. Implementing new algorithms would be something to tackle as only 4 Machine Learning models were implemented. Adding new algorithms such as Deep Learning and Time Series Forecasting would be nice to compare their performance to the Machine Learning ones. The limitation of the original dataset size (156227) involving orders from 100 different stores should also be addressed in the future to have a more diversified dataset. Different modelling techniques such as ensemble model prediction, multiple train/test splits, different models for each type of store, and different models for the number of products sold annually should also be explored in future studies. Using a faster and dedicated server to merge the data and train the models would be something to consider as this process takes a while to finish on the computer used to develop this thesis. Identifying and incorporating new features that better suit each model and not one model would also increase the performance of the forecasting models to be implemented. Finally, as noted in the model's results, sometimes the model's prediction value is mostly only one value (quantity of product sold), this is also something to be explored in the future. Due to time concerns and inexperience in the different areas of thesis englobes, these limitations were not addressed during the development of this thesis.

5.2 Conclusions

Taking into account the initial objectives planned 1.3 these were achieved:

- Analyse and select performance measurements that best suit each model

- Analyse, treat and adapt the data to each model
- Train, validate and apply multiple machine learning models approaches to the problem
- Implement a large-data process system that will be responsible for updating data and forecasting models
- Explore different variables on models when applicable: seasonality, promotions

As explained on 5.1 the work that was done was not the one expected in the beginning. Overall, even with the difficulties that appeared in the development of this thesis, the main goal of predicting when a product's stock will end was not achieved nevertheless, an intermediate goal of predicting the quantity of product sold in a day was achieved with an accuracy of 0.748 with minimal dispersion of errors. In terms of improvement, increasing the dataset's variety and the type of forecasting models should be a priority.

References

[1] Coefficient of Determination (R Squared): Definition, Calculation, . URL https://www.statisticshowto.com/probability-and-statistics/coefficient-of-determination-r-squared/.

[2] Creating materialized views in Amazon Redshift - Amazon Redshift, . URL https://docs.aws.amazon.com/redshift/latest/dg/materialized-view-overview.html.

[3] Featuretools | An open source framework for automated feature engineering Quick Start, . URL https://www.featuretools.com/.

[4] Climate Research 30:79. *Clim Res*, page 4, 2005.

[5] Oludare Isaac Abiodun, Aman Jantan, Abiodun Esther Omolara, Kemi Victoria Dada, Nachaat AbdElatif Mohamed, and Humaira Arshad. State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938, November 2018. ISSN 2405-8440. doi: 10.1016/j.heliyon.2018.e00938. URL https://www.sciencedirect.com/science/article/pii/S2405844018332067.

[6] Özden Gür Ali, Serpil Sayın, Tom van Woensel, and Jan Fransoo. SKU demand forecasting in the presence of promotions. *Expert Systems with Applications*, 36(10):12340–12348, December 2009. ISSN 09574174. doi: 10.1016/j.eswa.2009.04.052. URL https://linkinghub.elsevier.com/retrieve/pii/S0957417409004035.

[7] Nari Sivanandam Arunraj and Diane Ahrens. A hybrid seasonal autoregressive integrated moving average and quantile regression for daily food sales forecasting. *International Journal of Production Economics*, 170:321–335, December 2015. ISSN 0925-5273. doi: 10.1016/j.ijpe.2015.09.039. URL https://www.sciencedirect.com/science/article/pii/S0925527315003783.

[8] Taiwo Oladipupo Ayodele. *Types of Machine Learning Algorithms*. IntechOpen, February 2010. ISBN 978-953-307-034-6. doi: 10.5772/9385. URL https://www.intechopen.com/books/new-advances-in-machine-learning/types-of-machine-learning-algorithms. Publication Title: New Advances in Machine Learning.

[9] Ekaba Bisong. *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. Apress, 1st ed. edition edition, September 2019. ISBN 978-1-4842-4469-2.

[10] Meta S. Brown. *Data Mining For Dummies*. For Dummies, 1st edition, 2014. ISBN 1118893174.

[11] James Chen. Autoregressive Integrated Moving Average (ARIMA). URL https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp.

[12] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv:1412.3555 [cs]*, December 2014. URL http://arxiv.org/abs/1412.3555. arXiv: 1412.3555 version: 1.

[13] Kenneth Ezukwoke. Data Transformation for Machine Learning. URL https://www.academia.edu/40436475/Data_Transformation_for_Machine_Learning.

[14] Robert Fildes, Shaohui Ma, and Stephan Kolassa. Retail forecasting: Research and practice. *International Journal of Forecasting*, December 2019. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2019.06.004. URL https://www.sciencedirect.com/science/article/pii/S016920701930192X.

[15] Mohit Gurnani, Yogesh Korke, Prachi Shah, Sandeep Udmale, Vijay Sambhe, and Sunil Bhirud. Forecasting of sales by using fusion of machine learning techniques. In *2017 International Conference on Data Management, Analytics and Innovation (ICDMAI)*, pages 93–101, February 2017. doi: 10.1109/ICDMAI.2017.8073492.

[16] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. "O'Reilly Media, Inc.", September 2019. ISBN 978-1-4920-3261-8. Google-Books-ID: HHetDwAAQBAJ.

[17] John Hanke and Dean Wichern. *Business Forecasting*. Pearson, Upper Saddle River, N.J, 9th edition edition, February 2008. ISBN 978-0-13-230120-6.

[18] Trevor Hastie and Robert Tibshirani. Generalized Additive Models: Some Applications. *Journal of the American Statistical Association*, 82(398):371–386, June 1987. ISSN 0162-1459. doi: 10.1080/01621459.1987.10478440. URL https://www.tandfonline.com/doi/abs/10.1080/01621459.1987.10478440. Publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1987.10478440.

[19] Charles C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 20(1):5–10, January 2004. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2003.09.015. URL https://www.sciencedirect.com/science/article/pii/S0169207003001134.

[20] Sudan Jha, Eunmok Yang, Alaa Almagrabi, Ali Bashir, and Gyanendra Prasad Joshi. Comparative analysis of time series model and machine testing systems for crime forecasting. *Neural Computing and Applications*, May 2020. doi: 10.1007/s00521-020-04998-1.

[21] Alexandros Karatzoglou, David Meyer, and Kurt Hornik. Support Vector Machines in *R*. *Journal of Statistical Software*, 15(9), 2006. ISSN 1548-7660. doi: 10.18637/jss.v015.i09. URL http://www.jstatsoft.org/v15/i09/.

[22] Vipul Kedia, Vamsidhar Thummala, and Kamalakar Karlapalem. *Time Series Forecasting through Clustering - A Case Study*. January 2005. Pages: 191.

[23] Zeynep Hilal Kilimci, A. Okay Akyuz, Mitat Uysal, Selim Akyokus, M. Ozan Uysal, Berna Atak Bulbul, and Mehmet Ali Ekmis. An Improved Demand Forecasting Model Using Deep Learning Approach and Proposed Decision Integration Strategy for Supply Chain. *Complexity*, 2019:e9067367, March 2019. ISSN 1076-2787. doi: 10.1155/2019/9067367. URL https://www.hindawi.com/journals/complexity/2019/9067367/. Publisher: Hindawi.

[24] Murat Kirişci. Comparison of artificial neural network and logistic regression model for factors affecting birth weight. *SN Applied Sciences*, 1(4):378, March 2019. ISSN 2523-3971. doi: 10.1007/s42452-019-0391-x. URL https://doi.org/10.1007/s42452-019-0391-x.

[25] Ronald Klimberg, George Sillup, Kevin Boyle, and Vinay Tavva. Forecasting performance measures - What are their practical meaning? *Advances in Business and Management Forecasting*, 7:137–147, November 2010. ISSN 978-0-85724-201-3. doi: 10.1108/S1477-4070(2010)0000007012.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015. ISSN 0028-0836, 1476-4687. doi: 10.1038/nature14539. URL http://www.nature.com/articles/nature14539.

[27] Maobin Li, Shouwen Ji, and Gang Liu. Forecasting of Chinese E-Commerce Sales: An Empirical Comparison of ARIMA, Nonlinear Autoregressive Neural Network, and a Combined ARIMA-NARNN Model. *Mathematical Problems in Engineering*, 2018:e6924960, November 2018. ISSN 1024-123X. doi: 10.1155/2018/6924960. URL https://www.hindawi.com/journals/mpe/2018/6924960/. Publisher: Hindawi.

[28] Zachary C. Lipton, John Berkowitz, and Charles Elkan. A Critical Review of Recurrent Neural Networks for Sequence Learning. *arXiv:1506.00019 [cs]*, October 2015. URL http://arxiv.org/abs/1506.00019. arXiv: 1506.00019.

[29] Shaohui ma, Stephan Kolassa, and Robert Fildes. *Retail forecasting: research and practice*. October 2018. doi: 10.13140/RG.2.2.17747.22565.

[30] Ron Meir and Gunnar Rätsch. An introduction to boosting and leveraging. In *Advanced Lectures on Machine Learning, LNCS*, pages 119–184. Springer, 2003.

[31] Marlis Ontivero-Ortega, Agustin Lage-Castellanos, Giancarlo Valente, Rainer Goebel, and Mitchell Valdes-Sosa. Fast Gaussian Naïve Bayes for searchlight classification analysis. *NeuroImage*, 163:471–479, December 2017. ISSN 1053-8119. doi: 10.1016/j.neuroimage.2017.09.001. URL https://www.sciencedirect.com/science/article/pii/S1053811917307371.

[32] Bohdan Pavlyshenko. Machine-Learning Models for Sales Time Series Forecasting. *Data*, 4:15, January 2019. doi: 10.3390/data4010015.

[33] Adhistya Permanasari, Indriana Hidayah, and Isna Alfi Bustoni. *SARIMA (Seasonal ARIMA) implementation on time series to forecast the number of Malaria incidence*. October 2013. ISBN 978-1-4799-0423-5. doi: 10.1109/ICITEED.2013.6676239. Pages: 207.

[34] Md. Habibur Rahman, Umma Salma, Md Hossain, and Md Tareq Ferdous Khan. Revenue Forecasting using Holt–Winters Exponential Smoothing. *Research & Reviews: Journal of Statistics*, 5:19–25, December 2016.

[35] Saifur Rahman, Muhammad Irfan, Mohsin Raza, Khawaja Moyeezullah Ghori, Shumayla Yaqoob, and Muhammad Awais. Performance Analysis of Boosting Classifiers in Recognizing Activities of Daily Living. *International Journal of Environmental Research and Public Health*, 17(3), February 2020. ISSN 1661-7827. doi: 10.3390/ijerph17031082. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7038216/.

[36] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, July 2020. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2019.07.001. URL https://www.sciencedirect.com/science/article/pii/S0169207019301888.

[37] Neil Salkind. *Encyclopedia of Research Design*. SAGE Publications, Inc., 2455 Teller Road, Thousand Oaks California 91320 United States, 2010. ISBN 978-1-4129-6127-1 978-1-4129-6128-8. doi: 10.4135/9781412961288. URL http://methods.sagepub.com/reference/encyc-of-research-design.

[38] Behrang Samadi, Azamat Nogoev, Assc. Prof. Dr. Rashad Yazdanifard, Shahriar Mohseni, and Meera Menon. *The Evolution and Development of E-Commerce Market and E-Cash*. October 2011. doi: 10.1115/1.859858.

[39] Alessia Sarica, Antonio Cerasa, and Aldo Quattrone. Random Forest Algorithm for the Classification of Neuroimaging Data in Alzheimer's Disease: A Systematic Review. *Frontiers in Aging Neuroscience*, 9, 2017. ISSN 1663-4365. doi: 10.3389/fnagi.2017.00329. URL https://www.frontiersin.org/articles/10.3389/fnagi.2017.00329/full. Publisher: Frontiers.

[40] Ashley (CDC/ONDIEH/NCBDDD) (CTR) Satterfield. The Six Dimensions of EHDI Data Quality Assessment. page 4.

[41] Mike Schuster and Kuldip Paliwal. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45:2673–2681, December 1997. doi: 10.1109/78.650093.

[42] Bernhard Schölkopf and AJ Smola. Learning With Kernels. *Cambridge: MIT Press. Schölkopf, B., Mika, S., Burges, C. J., P. Knirsch, K.-R. M., Rätsch, G., & Smola, A. J*, page 2000, November 2002.

[43] Brian Seaman. Considerations of a retail forecasting practitioner. *International Journal of Forecasting*, 34(4):822–829, October 2018. ISSN 0169-2070. doi: 10.1016/j.ijforecast.2018.03.001. URL https://www.sciencedirect.com/science/article/pii/S0169207018300293.

[44] Neha Sharma, Vibhor Jain, and Anju Mishra. An Analysis Of Convolutional Neural Networks For Image Classification. *Procedia Computer Science*, 132:377–384, January 2018. ISSN 1877-0509. doi: 10.1016/j.procs.2018.05.198. URL https://www.sciencedirect.com/science/article/pii/S1877050918309335.

[45] Maxim Shcherbakov, Adriaan Brebels, N.L. Shcherbakova, Anton Tyukov, T.A. Janovsky, and V.A. Kamaev. A survey of forecast error measures. *World Applied Sciences Journal*, 24:171–176, January 2013. doi: 10.5829/idosi.wasj.2013.24.itmies.80032.

[46] Amanpreet Singh, Narina Thakur, and Aakanksha Sharma. A review of supervised machine learning algorithms. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, pages 1310–1315, March 2016.

[47] Karandeep Singh, P M Booma, and Umapathy Eaganathan. E-Commerce System for Sale Prediction Using Machine Learning Technique. *Journal of Physics: Conference Series*, 1712:012042, December 2020. ISSN 1742-6588, 1742-6596. doi: 10.1088/1742-6596/1712/1/012042. URL https://iopscience.iop.org/article/10.1088/1742-6596/1712/1/012042.

[48] R. S. Soni and D. Srikanth. Inventory forecasting model using genetic programming and Holt-Winter's exponential smoothing method. In *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*, pages 2086–2091, May 2017. doi: 10.1109/RTEICT.2017.8256967.

[49] Sean J. Taylor and Benjamin Letham. Forecasting at scale. Technical Report e3190v2, PeerJ Inc., September 2017. URL https://peerj.com/preprints/3190. ISSN: 2167-9843.

[50] Güzin Tirkeş, Cenk Güray, and Nese Celebi. Demand Forecasting: A Comparison Between The Holt-Winters, Trend Analysis and Decomposition Models. *Tehnicki Vjesnik*, 24:503–509, September 2017.

[51] Resul Tugay and Sule Gunduz Oguducu. Demand Prediction Using Machine Learning Methods and Stacked Generalization. *arXiv:2009.09756 [cs, stat]*, September 2020. URL http://arxiv.org/abs/2009.09756. arXiv: 2009.09756.

[52] Stylianos I. Vagropoulos, G. I. Chouliaras, E. G. Kardakos, C. K. Simoglou, and A. G. Bakirtzis. Comparison of SARIMAX, SARIMA, modified SARIMA and ANN-based models for short-term PV generation forecasting. In *2016 IEEE International Energy Conference (ENERGYCON)*, pages 1–6, April 2016. doi: 10.1109/ENERGYCON.2016.7514029.

[53] Sander van Cranenburgh, Shenhao Wang, Akshay Vij, Francisco Pereira, and Joan Walker. Choice modelling in the age of machine learning - Discussion paper. *Journal of Choice Modelling*, 42:100340, March 2022. ISSN 1755-5345. doi: 10.1016/j.jocm.2021.100340. URL https://www.sciencedirect.com/science/article/pii/S1755534521000725.

[54] Sun-Chong Wang. Artificial Neural Network. In Sun-Chong Wang, editor, *Interdisciplinary Computing in Java Programming*, The Springer International Series in Engineering and Computer Science, pages 81–100. Springer US, Boston, MA, 2003. ISBN 978-1-4615-0377-4. doi: 10.1007/978-1-4615-0377-4_5. URL https://doi.org/10.1007/978-1-4615-0377-4_5.

[55] Yang Xin, Lingshuang Kong, Zhi Liu, Yuling Chen, Yanmiao Li, Hongliang Zhu, Mingcheng Gao, Haixia Hou, and Chunhua Wang. Machine Learning and Deep Learning Methods for Cybersecurity. *IEEE Access*, 6:35365–35381, 2018. ISSN 2169-3536. doi: 10.1109/ACCESS.2018.2836950. Conference Name: IEEE Access.

[56] Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*, 9(4):611–629, August 2018. ISSN 1869-4101. doi: 10.

1007/s13244-018-0639-9. URL https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9. Number: 4 Publisher: SpringerOpen.

[57] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent Neural Network Regularization. *arXiv:1409.2329 [cs]*, February 2015. URL http://arxiv.org/abs/1409.2329. arXiv: 1409.2329.

[58] İrem İşlek and Şule Gündüz Öğüdücü. A retail demand forecasting model based on data mining techniques. In *2015 IEEE 24th International Symposium on Industrial Electronics (ISIE)*, pages 55–60, June 2015. doi: 10.1109/ISIE.2015.7281443. ISSN: 2163-5145.

[59] Indrė Žliobaitė, Mykola Pechenizkiy, and João Gama. An Overview of Concept Drift Applications. In Nathalie Japkowicz and Jerzy Stefanowski, editors, *Big Data Analysis: New Algorithms for a New Society*, volume 16, pages 91–114. Springer International Publishing, Cham, 2016. ISBN 978-3-319-26987-0 978-3-319-26989-4. doi: 10.1007/978-3-319-26989-4_4. URL http://link.springer.com/10.1007/978-3-319-26989-4_4. Series Title: Studies in Big Data.

Chapter 6

Apendix

In this chapter figures related to chapter3 and 4 spreadsheet of results will be displayed. The link to the spreadsheet is also made available.

6.1 Chapter 3 appendix

	price
count	277.000000
mean	56027.951801
std	47841.585903
min	19990.000000
25%	19990.000000
50%	39980.000000
75%	79960.000000
max	255707.549628
shapiro p value	0.000000
shapiro test statistic	0.562616
outlier_percentage	2.888087

Figure 6.1: Product 1 Z-score qrice

	qty
count	277.000000
mean	3.727826
std	3.492128
min	1.000000
25%	1.000000
50%	2.000000
75%	5.000000
max	19.151965
shapiro p value	0.000000
shapiro test statistic	0.536356
outlier_percentage	2.166065

Figure 6.2: Product 1 Z-score quantity

	price
count	269.000000
mean	50829.591078
std	37452.812921
min	19990.000000
25%	19990.000000
50%	39980.000000
75%	59970.000000
max	159920.000000
outlier_percentage	2.888087

Figure 6.3: Product 1 IQR price

	price
count	277.000000
mean	54268.880866
std	41967.087672
min	19990.000000
25%	19990.000000
50%	39980.000000
75%	79960.000000
max	169915.000000
outlier_percentage	0.000000

Figure 6.4: Product 1 IQR capped price

	qty
count	267.000000
mean	3.262172
std	2.518837
min	1.000000
25%	1.000000
50%	2.000000
75%	4.000000
max	11.000000
outlier_percentage	3.610108

Figure 6.5: Product 1 Quantity IQR

	qty
count	277.000000
mean	3.541516
std	2.864559
min	1.000000
25%	1.000000
50%	2.000000
75%	5.000000
max	11.000000
outlier_percentage	0.000000

Figure 6.6: Product 1 IQR capped price

	price
count	230.000000
mean	71306.084344
std	53426.474246
min	32990.000000
25%	32990.000000
50%	65980.000000
75%	98970.000000
max	264793.879818
shapiro p value	0.000000
shapiro test statistic	0.652737
outlier_percentage	3.043478

Figure 6.7: Product 2 Z-score qrice

	qty
count	230.000000
mean	2.718914
std	2.123992
min	1.000000
25%	1.000000
50%	2.000000
75%	4.000000
max	10.270066
shapiro p value	0.000000
shapiro test statistic	0.701371
outlier_percentage	2.173913

Figure 6.8: Product 2 Z-score quantity

	price
count	222.000000
mean	64493.963964
std	40162.864678
min	32990.000000
25%	32990.000000
50%	49485.000000
75%	98970.000000
max	197940.000000
outlier_percentage	3.478261

Figure 6.9: Product 2 IQR price

	price
count	230.000000
mean	69135.565217
std	46445.386978
min	32990.000000
25%	32990.000000
50%	65980.000000
75%	98970.000000
max	197940.000000
outlier_percentage	0.000000

Figure 6.10: Product 2 IQR capped price

	qty
count	223.000000
mean	2.493274
std	1.721276
min	1.000000
25%	1.000000
50%	2.000000
75%	3.000000
max	8.000000
outlier_percentage	3.043478

Figure 6.11: Product 2 Quantity IQR

	qty
count	230.000000
mean	2.676087
std	1.985338
min	1.000000
25%	1.000000
50%	2.000000
75%	4.000000
max	8.500000
outlier_percentage	0.000000

Figure 6.12: Product 2 IQR capped price

	price
count	143.000000
mean	70295.118981
std	62880.670236
min	29990.000000
25%	29990.000000
50%	59980.000000
75%	89970.000000
max	291754.004774
shapiro p value	0.000000
shapiro test statistic	0.625660
outlier_percentage	2.797203

Figure 6.13: Product 3 Z-score qrice

	qty
count	143.000000
mean	2.991258
std	2.969329
min	1.000000
25%	1.000000
50%	2.000000
75%	3.500000
max	12.937468
shapiro p value	0.000000
shapiro test statistic	0.657078
outlier_percentage	3.496503

Figure 6.14: Product 3 Z-score quantity

	price
count	134.000000
mean	57518.134328
std	39617.350042
min	29990.000000
25%	29990.000000
50%	29990.000000
75%	59980.000000
max	179940.000000
outlier_percentage	6.293706

Figure 6.15: Product 3 IQR price

	price
count	143.000000
mean	65223.006993
std	48581.490506
min	29990.000000
25%	29990.000000
50%	59980.000000
75%	89970.000000
max	179940.000000
outlier_percentage	0.000000

Figure 6.16: Product 3 IQR capped price

	qty
count	132.000000
mean	2.280303
std	1.672929
min	1.000000
25%	1.000000
50%	2.000000
75%	3.000000
max	7.000000
outlier_percentage	7.692308

Figure 6.17: Product 3 Quantity IQR

	qty
count	143.000000
mean	2.662587
std	2.085171
min	1.000000
25%	1.000000
50%	2.000000
75%	3.500000
max	7.250000
outlier_percentage	0.000000

Figure 6.18: Product 3 IQR capped price

	price
count	73.000000
mean	44939.681176
std	29148.693301
min	14990.000000
25%	14990.000000
50%	44970.000000
75%	59960.000000
max	132696.725831
shapiro p value	0.000005
shapiro test statistic	0.880407
outlier_percentage	1.369863

Figure 6.19: Product 4 Z-score qrice

	qty
count	73.000000
mean	4.123288
std	2.828158
min	1.000000
25%	2.000000
50%	4.000000
75%	6.000000
max	11.000000
shapiro p value	0.000024
shapiro test statistic	0.898742
outlier_percentage	0.000000

Figure 6.20: Product 4 Z-score quantity

	price
count	72.000000
mean	43720.833333
std	27416.010987
min	14990.000000
25%	14990.000000
50%	44970.000000
75%	59960.000000
max	119920.000000
outlier_percentage	1.369863

Figure 6.21: Product 4 IQR price

	price
count	73.000000
mean	44867.328767
std	28933.600302
min	14990.000000
25%	14990.000000
50%	44970.000000
75%	59960.000000
max	127415.000000
outlier_percentage	0.000000

Figure 6.22: Product 4 IQR capped price

	qty
count	73.000000
mean	4.123288
std	2.828158
min	1.000000
25%	2.000000
50%	4.000000
75%	6.000000
max	11.000000
outlier_percentage	0.000000

Figure 6.23: Product 4 Quantity IQR

	qty
count	73.000000
mean	4.123288
std	2.828158
min	1.000000
25%	2.000000
50%	4.000000
75%	6.000000
max	11.000000
outlier_percentage	0.000000

Figure 6.24: Product 4 IQR capped price

	price
count	133.000000
mean	826346.540956
std	461620.986960
min	599990.000000
25%	599990.000000
50%	599990.000000
75%	599990.000000
max	2426439.986774
shapiro p value	0.000000
shapiro test statistic	0.521248
outlier_percentage	4.511278

Figure 6.25: Product 5 Z-score qrice

	qty
count	133.000000
mean	1.377267
std	0.769381
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	4.044134
shapiro p value	0.000000
shapiro test statistic	0.521248
outlier_percentage	4.511278

Figure 6.26: Product 5 Z-score quantity

	price
count	100.000000
mean	599990.000000
std	0.000000
min	599990.000000
25%	599990.000000
50%	599990.000000
75%	599990.000000
max	599990.000000
outlier_percentage	24.812030

Figure 6.27: Product 5 IQR price

	price
count	133.000000
mean	599990.000000
std	0.000000
min	599990.000000
25%	599990.000000
50%	599990.000000
75%	599990.000000
max	599990.000000
outlier_percentage	0.000000

Figure 6.28: Product 5 IQR capped price

	qty
count	100.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	24.812030

Figure 6.29: Product 5 Quantity IQR

	qty
count	133.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	0.000000

Figure 6.30: Product 5 IQR capped price

	price
count	146.000000
mean	31981.276284
std	14224.653283
min	19990.000000
25%	19990.000000
50%	29990.000000
75%	29990.000000
max	87711.584370
shapiro p value	0.000000
shapiro test statistic	0.518787
outlier_percentage	2.739726

Figure 6.31: Product 6 Z-score qrice

	qty
count	146.000000
mean	1.237402
std	0.502630
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	3.330331
shapiro p value	0.000000
shapiro test statistic	0.415857
outlier_percentage	2.739726

Figure 6.32: Product 6 Z-score quantity

	price
count	128.000000
mean	27177.343750
std	4849.699801
min	19990.000000
25%	19990.000000
50%	29990.000000
75%	29990.000000
max	39980.000000
outlier_percentage	12.328767

Figure 6.33: Product 6 IQR price

	price
count	146.000000
mean	29373.424658
std	7425.072199
min	19990.000000
25%	19990.000000
50%	29990.000000
75%	29990.000000
max	44990.000000
outlier_percentage	0.000000

Figure 6.34: Product 6 IQR capped price

	qty
count	116.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	20.547945

Figure 6.35: Product 6 Quantity IQR

	qty
count	146.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	0.000000

Figure 6.36: Product 6 IQR capped price

	price
count	113.000000
mean	97160.439854
std	60699.930486
min	59990.000000
25%	69990.000000
50%	69990.000000
75%	69990.000000
max	335299.851727
shapiro p value	0.000000
shapiro test statistic	0.483390
outlier_percentage	4.424779

Figure 6.37: Product 7 Z-score qrice

	qty
count	113.000000
mean	1.413499
std	0.894027
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	4.862693
shapiro p value	0.000000
shapiro test statistic	0.466469
outlier_percentage	1.769912

Figure 6.38: Product 7 Z-score quantity

	price
count	76.000000
mean	69990.000000
std	0.000000
min	69990.000000
25%	69990.000000
50%	69990.000000
75%	69990.000000
max	69990.000000
outlier_percentage	32.743363

Figure 6.39: Product 7 IQR price

	price
count	113.000000
mean	69990.000000
std	0.000000
min	69990.000000
25%	69990.000000
50%	69990.000000
75%	69990.000000
max	69990.000000
outlier_percentage	0.000000

Figure 6.40: Product 7 IQR capped price

	qty
count	87.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	23.008850

Figure 6.41: Product 7 Quantity IQR

	qty
count	113.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	0.000000

Figure 6.42: Product 7 IQR capped price

	price
count	131.000000
mean	22625.364685
std	8979.929567
min	17990.000000
25%	17990.000000
50%	19990.000000
75%	19990.000000
max	62706.386871
shapiro p value	0.000000
shapiro test statistic	0.405371
outlier_percentage	3.053435

Figure 6.43: Product 8 Z-score qrice

	qty
count	131.000000
mean	1.211207
std	0.538920
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	3.556057
shapiro p value	0.000000
shapiro test statistic	0.366793
outlier_percentage	3.816794

Figure 6.44: Product 8 Z-score quantity

	price
count	111.000000
mean	19125.135135
std	995.320748
min	17990.000000
25%	17990.000000
50%	19990.000000
75%	19990.000000
max	19990.000000
outlier_percentage	15.267176

Figure 6.45: Product 8 IQR price

	price
count	131.000000
mean	19715.190840
std	1668.962990
min	17990.000000
25%	17990.000000
50%	19990.000000
75%	19990.000000
max	22990.000000
outlier_percentage	0.000000

Figure 6.46: Product 8 IQR capped price

	qty
count	110.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	16.030534

Figure 6.47: Product 8 Quantity IQR

	qty
count	131.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	0.000000

Figure 6.48: Product 8 IQR capped price

	price
count	79.000000
mean	411212.295664
std	362587.763283
min	239990.000000
25%	259990.000000
50%	259990.000000
75%	269990.000000
max	1918435.678744
shapiro p value	0.000000
shapiro test statistic	0.415174
outlier_percentage	3.797468

Figure 6.49: Product 9 Z-score qrice

	qty
count	79.000000
mean	1.591571
std	1.384125
min	1.000000
25%	1.000000
50%	1.000000
75%	1.500000
max	7.367068
shapiro p value	0.000000
shapiro test statistic	0.417927
outlier_percentage	3.797468

Figure 6.50: Product 9 Z-score quantity

	price
count	60.000000
mean	262489.833333
std	4648.400670
min	254990.000000
25%	259990.000000
50%	259990.000000
75%	269982.500000
max	269990.000000
outlier_percentage	24.050633

Figure 6.51: Product 9 IQR price

	price
count	79.000000
mean	267394.936709
std	10617.466364
min	244990.000000
25%	259990.000000
50%	259990.000000
75%	269990.000000
max	284990.000000
outlier_percentage	0.000000

Figure 6.52: Product 9 IQR capped price

	qty
count	69.000000
mean	1.144928
std	0.354607
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	2.000000
outlier_percentage	12.658228

Figure 6.53: Product 9 Quantity IQR

	qty
count	79.000000
mean	1.284810
std	0.496356
min	1.000000
25%	1.000000
50%	1.000000
75%	1.500000
max	2.250000
outlier_percentage	0.000000

Figure 6.54: Product 9 IQR capped price

	price
count	101.000000
mean	81692.852682
std	37024.817894
min	59990.000000
25%	69990.000000
50%	69990.000000
75%	69990.000000
max	292128.120901
shapiro p value	0.000000
shapiro test statistic	0.269829
outlier_percentage	3.960396

Figure 6.55: Product 10 Z-score qrice

	qty
count	101.000000
mean	1.261421
std	0.629932
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	4.403501
shapiro p value	0.000000
shapiro test statistic	0.324428
outlier_percentage	1.980198

Figure 6.56: Product 10 Z-score quantity

	price
count	77.000000
mean	69990.000000
std	0.000000
min	69990.000000
25%	69990.000000
50%	69990.000000
75%	69990.000000
max	69990.000000
outlier_percentage	23.762376

Figure 6.57: Product 10 IQR price

	price
count	101.000000
mean	69990.000000
std	0.000000
min	69990.000000
25%	69990.000000
50%	69990.000000
75%	69990.000000
max	69990.000000
outlier_percentage	0.000000

Figure 6.58: Product 10 IQR capped price

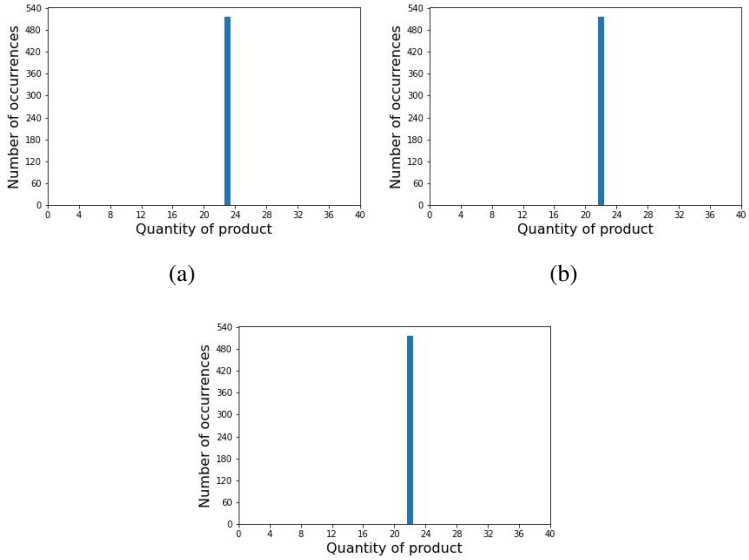
	qty
count	82.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	18.811881

Figure 6.59: Product 10 Quantity IQR

	qty
count	101.000000
mean	1.000000
std	0.000000
min	1.000000
25%	1.000000
50%	1.000000
75%	1.000000
max	1.000000
outlier_percentage	0.000000

Figure 6.60: Product 10 IQR capped price

6.2 Chapter 4 appendix



(a) (b)

(c)

Figure 6.61: Adaboost unsorted dataset results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier

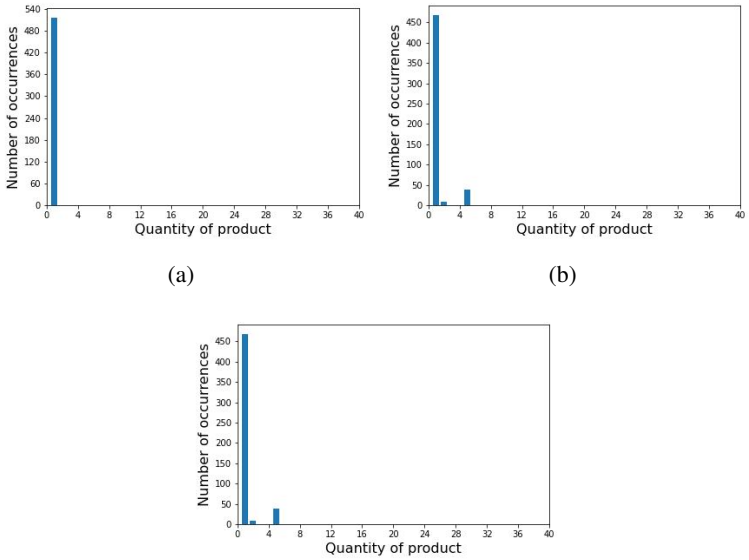


Figure 6.62: Gaussian Naive-Bayes unsorted dataset results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier

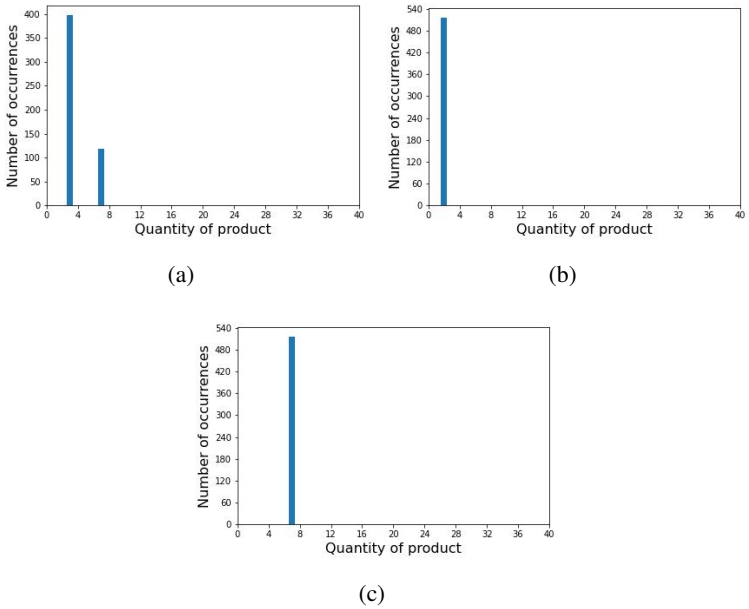
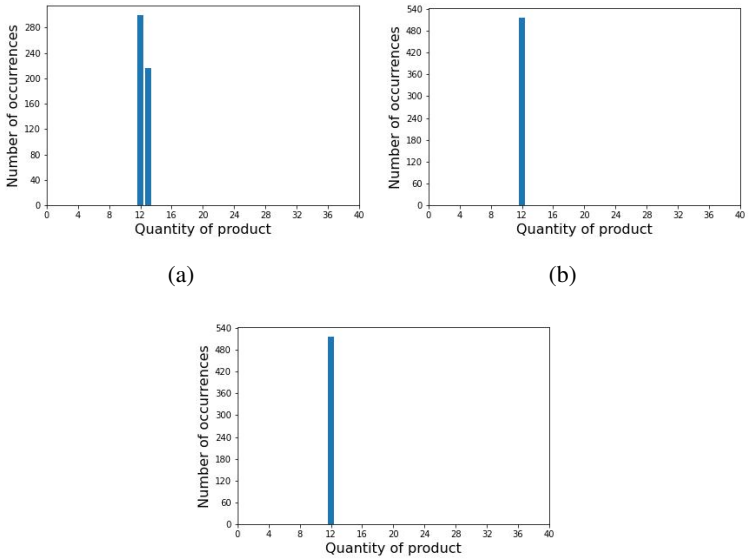


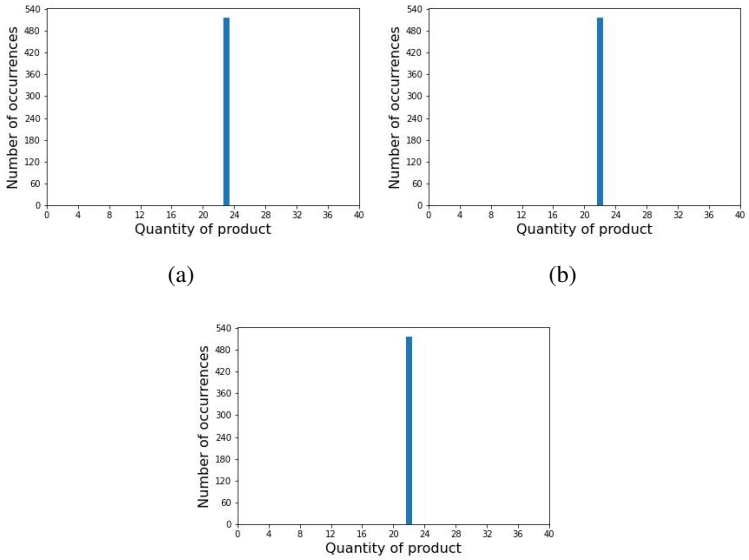
Figure 6.63: Logistic Regression unsorted dataset results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier



(a) (b)

(c)

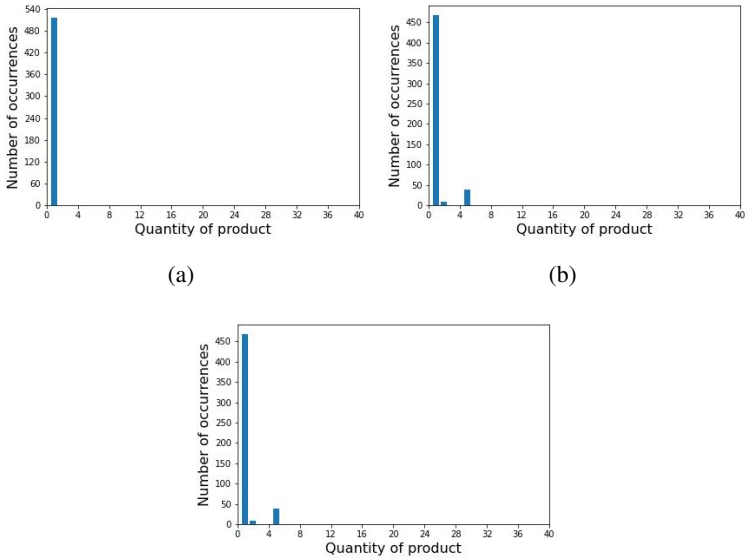
Figure 6.64: Random forest unsorted dataset results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier



(a) (b)

(c)

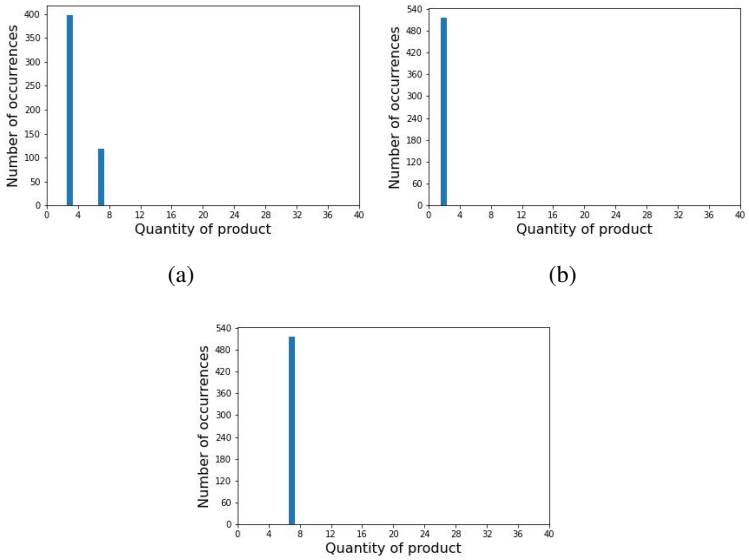
Figure 6.65: Adaboost sort the dataset by the date of order creation results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier



(a) (b)

(c)

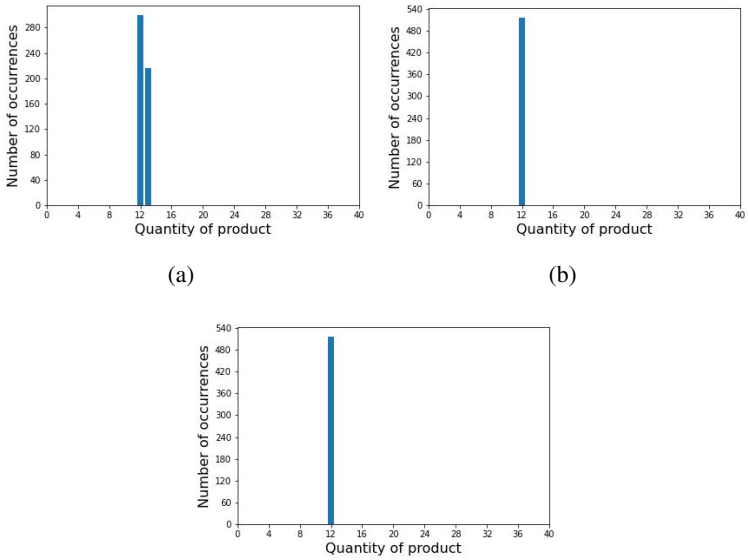
Figure 6.66: Gaussian Naive-Bayes sort the dataset by the date of order creation results (a) Normal outlier (b) IQR outlier (c) Z_score outlier



(a) (b)

(c)

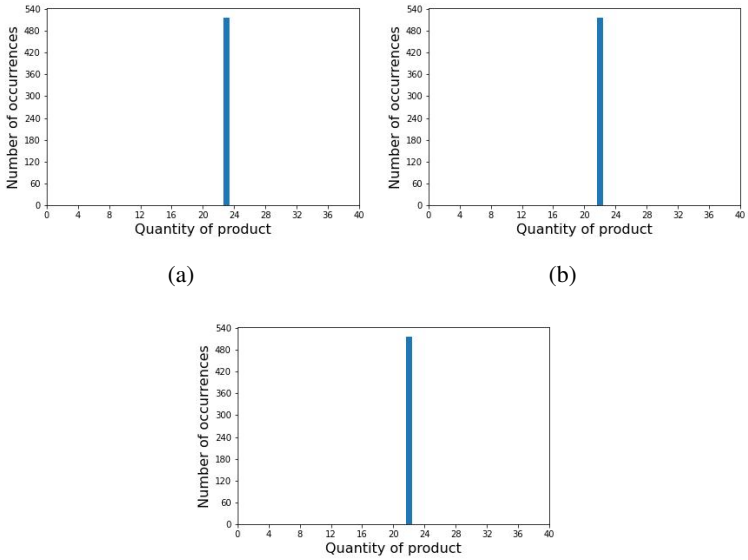
Figure 6.67: Logistic Regression sort the dataset by the date of order creation results (a) Normal outlier (b) IQR outlier (c) Z_score outlier



(a) (b)

(c)

Figure 6.68: Random forest sort the dataset by the date of order creation results (a) Normal outlier (b) IQR outlier (c) Z_score outlier



(a) (b)

(c)

Figure 6.69: Adaboost sort the dataset by the quantity of product sold results (a) Normal outlier (b) IQR outlier (c) Z_score outlier

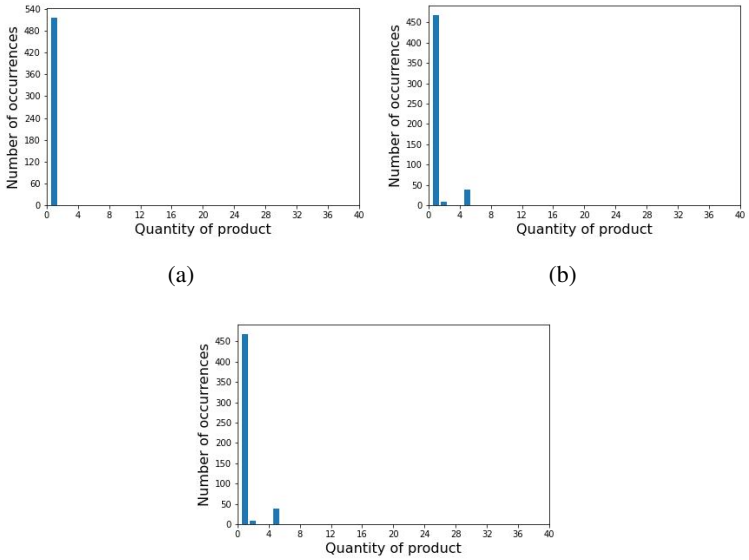


Figure 6.70: Gaussian Naive-Bayes sort the dataset by the quantity of product sold results (a) Normal outlier (b) IQR outlier (c) Z_score outlier

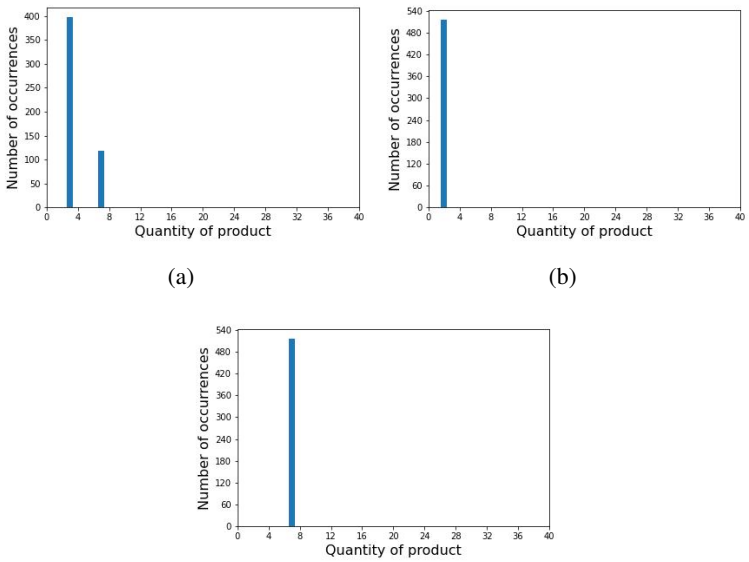


Figure 6.71: Logistic Regression sort the dataset by the quantity of product sold results (a) Normal outlier (b) IQR outlier (c) Z_score outlier

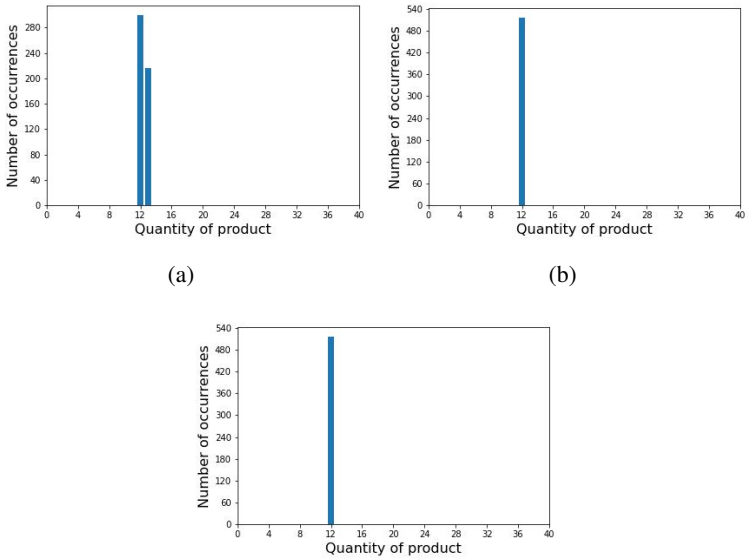


Figure 6.72: Random forest sort the dataset by the quantity of product sold results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier

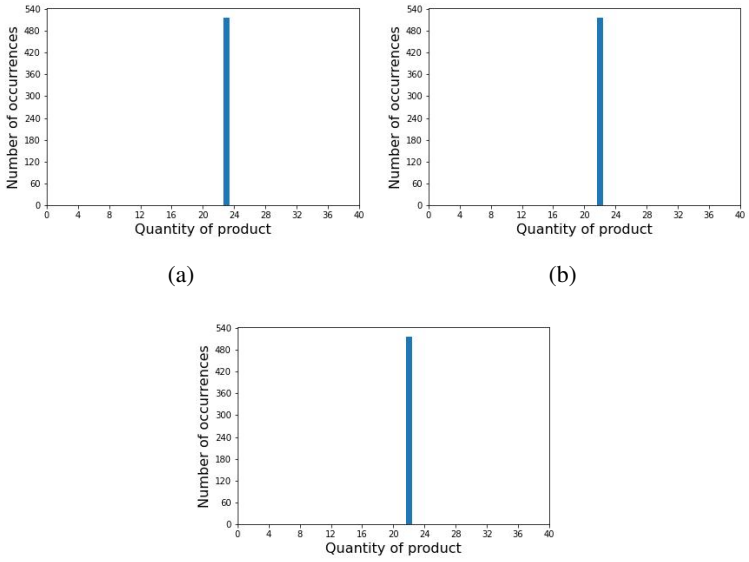


Figure 6.73: Adaboost sort the dataset by order_products.product_id results
(a) Normal outlier (b) IQR outlier (c) Z_score outlier

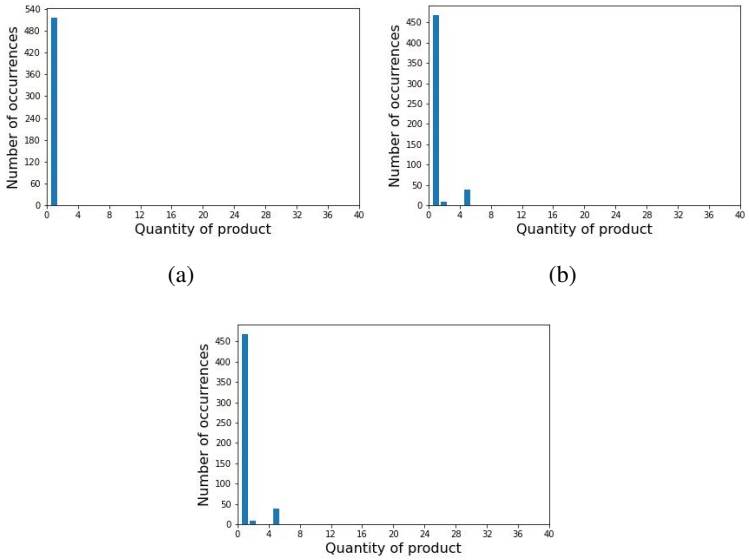


Figure 6.74: Gaussian Naive-Bayes sort the dataset by order_products.product_id results (a) Normal outlier (b) IQR outlier (c) Z_score outlier

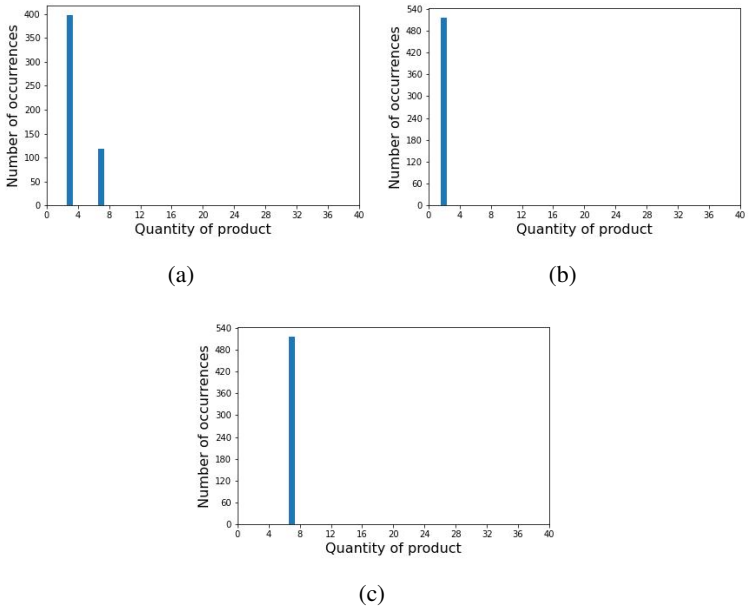


Figure 6.75: Logistic Regression sort the dataset by order_products.product_id results (a) Normal outlier (b) IQR outlier (c) Z_score outlier

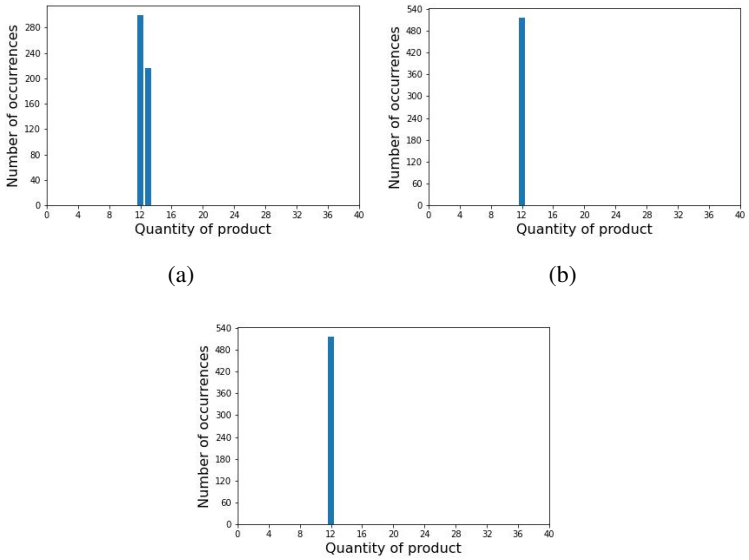


Figure 6.76: Random forest sort the dataset by order_products.product_id results (a) Normal outlier (b) IQR outlier (c) Z_score outlier