Improving Supply Chain Efficiency in the Market via Predictive Analysis Techniques

Deekshita Prakash Savanur, Gouri Benni, Krishna Sameera Surapaneni, Sonali Arcot

Department of Applied Data Science, San Jose State University

DATA 270 : Data Analyst Process

Dr. Eduardo Chan

May 17, 2023

Abstract

Supply chain is a crucial part of any corporate operation, and its effectiveness directly affects revenue. Prior research conducted on enhancing the efficiency of demand prediction in supply chain processes has shown that Random Forest and XGBoost exhibit considerable potential in the domain. This project aims to leverage the insights gleaned from previous studies to apply them in a distinct facet of the supply chain, specifically addressing the crucial aspect of timely delivery. The project aims to increase customer satisfaction through faster shipping of products by using predictive analytics to analyze data collected from DataCo company and identify patterns to forecast any delays in shipment and improve supply chain efficiency. It also aims to locate bottlenecks to boost customer satisfaction and profitability. For this project, Random Forest, XGBoost, SVR and KNN were applied. The models predict the days to ship the order to the customer. The best-performing model was chosen based on the R2 score and the suitability of the dataset. Results revealed that the XGBoost model had outperformed the other models in predicting supply chain efficiency, with an R2 score of 89%. By finding the days it takes to ship a product and the factors affecting it, insights were drawn on improving supply chain efficiency on the transportation end. Integrating real-time data streams and exploring advanced machine-learning techniques can further enhance supply chain optimization.

Introduction

Project Background and Executive Summary

Project Background

Over the past decade, the supply chain industry has undergone significant changes due to factors such as globalization, e-commerce, sustainability, and digitalization. The COVID-19 pandemic further accelerated these changes, resulting in supply chain disruptions for many companies. To address these challenges and ensure optimal performance, companies are turning to new technologies and techniques, such as predictive analysis. The predictive analysis involves using data, statistical algorithms, and machine learning to identify patterns and trends, enabling businesses to make informed decisions and optimize their supply chain processes.

Efficient supply chain management involves optimizing every aspect of the supply chain, including procurement, production, inventory management, and logistics. This requires a holistic approach that emphasizes communication, coordination, and collaboration among all stakeholders. To achieve optimal supply chain efficiency, companies must leverage technology and data analytics to gain insights into their operations and identify areas for improvement. This includes implementing tools such as inventory management software, transportation management systems, and demand forecasting tools to optimize inventory levels, reduce transportation costs, and improve demand planning. Another critical aspect of improving supply chain efficiency is the streamlining of processes to eliminate inefficiencies. This involves identifying bottlenecks and other areas of waste and implementing process improvements to reduce lead times and improve throughput.

The goal of this research paper is to explore the latest predictive analysis techniques in the supply chain industry and their potential to improve supply chain efficiency and reduce costs.

The paper will examine current research and case studies to identify best practices and provide insights into how businesses can leverage predictive analysis to optimize their supply chain operations. The research paper is aimed to help businesses improve their supply chain efficiency, reduce costs, and remain competitive in today's dynamic and rapidly-changing market.

Needs and Importance

Improving supply chain efficiency using predictive analysis techniques has become increasingly important for businesses due to several reasons. Firstly, the high costs associated with inefficient supply chains can come in the form of increased inventory, longer lead times, and higher transportation expenses. Predictive analytics have a notable impact on a company's profitability, making it an important area for improvement. Secondly, supply chain disruptions can have severe consequences for businesses, both in terms of costs and customer satisfaction. The COVID-19 pandemic highlighted the risks of supply chain disruptions, with companies experiencing them. This can lead to lost revenue, delayed shipments, and low customer satisfaction. Therefore, improving supply chain efficiency and resilience has become a top priority for businesses to mitigate the risks associated with these disruptions.

Thirdly, the growth of e-commerce has increased customer expectations for faster delivery times, putting pressure on companies to optimize their supply chain operations. Businesses need to optimize their supply chain operations to meet these expectations and remain competitive in the market. Finally, predictive analysis techniques can provide businesses with valuable insights into their supply chain operations, enabling them to make data-driven choices and optimize their processes. Predictive analysis involves using data, statistical algorithms, and machine learning to identify patterns and trends. By applying predictive analysis to their supply

chain operations, businesses can optimize inventory levels, improve transportation efficiency, and identify potential risks before they occur.

Target Problem

It is impossible to exaggerate the significance of on-time delivery in the context of supply chain management. Delivering the goods to the customer is important, but so does accomplishing so within the specified time range. This dedication to promptness is crucial in determining client happiness and how they view the brand as a whole. Stock shortages may cause problems in this process. Even a brief time of shortages can cause unanticipated delivery delays. These setbacks may cause timetable conflicts, jeopardize client relationships, and harm the reputation of the business. Additionally, it may prompt customers to look for alternatives, potentially favoring rivals who can ensure ongoing service.

This emphasizes how crucial it is to have a strong, efficient system for managing inventory in place to uphold supply chain performance criteria. A system like this guarantees consistent product availability, strikes an equilibrium between excess and lack of resources, and permits easy and quick deliveries.

However, the function of an effective inventory management system goes beyond only preserving adequate stock levels. It includes predicting demand patterns, comprehending market dynamics, and selecting products wisely. It involves employing technologies and information to anticipate needs, manage a seamless supply chain, and deliver goods on schedule to satisfy consumer expectations.

The difficulty of precisely forecasting demand is a significant barrier to effectively managing this problem. The erratic nature of customer demand is the main cause of this problem. Several elements, including industry changes, sales activities, cyclical trends, and even global

events, can have a significant impact on purchasing patterns. These factors might result in abrupt changes in demand, making accurate forecasting difficult. This unpredictable nature can lead to situations of overstocking or understocking, each of which can hinder the efficient operation of the supply network and, as a result, have an impact on how quickly products are delivered.

Motivation and Goals

In order to remain competitive, it is essential for manufacturers and retailers in all industries to accurately predict delivery delay. Machine learning can be particularly useful in predicting the shipment delay for products, such as those in the fresh food, technology, and fashion sectors. Compared to traditional statistical methods, machine learning can provide a higher level of accuracy in delay forecasting, resulting in better inventory management throughout the supply chain. This can reduce the occurrence of stockouts, improve product availability for consumers, and increase profitability. As a result, machine learning offers a significant advantage in predicting delay for various products. Effective supply chain management has always relied on accurate delay forecasting to ensure timely stock replenishment, improved capacity management, and optimal sales and revenue. In addition, delay forecasting facilitates management and decision-making, and enables future growth and expansion plans. To achieve accurate delay forecasting, it is necessary to conduct thorough research into a variety of variables, ranging from sales history patterns to specific events in the commercial calendar, such as Christmas. This comprehensive examination of factors affecting a company's supply infrastructure is crucial to maintaining business preparedness, continuity, and a positive end-user experience.

One of the primary goals of supply chain management is to optimize efficiency and minimize costs. This involves identifying and eliminating inefficiencies, reducing waste, and

ensuring that resources are used effectively. By optimizing their supply chain, organizations can increase their profitability and remain competitive in their respective markets.

Project Approaches and Methods

The research approach offers a high-level strategic plan. It sets a plan to carry out the research's objectives and explains the project's scope. This involves choosing a conventional, linear technique like the Waterfall model or an increasingly dynamic, iterative approach like Agile, depending on the specific requirements of the research. The research aims to improve delivery timeliness and streamline inventory management.

Data Collection. It is the next step, where relevant data is collected from different sources, including historical sales data, production data, inventory data, supplier data, customer data, and more. The data collected must be verified to ensure that it is accurate and reliable. The collected data may be in different formats, such as structured, semi-structured, or unstructured, and may require processing and transformation to make it suitable for machine learning models.

Data Preparation. It is a crucial step in any machine learning project, including supply chain management. Data preprocessing techniques such as data normalization, data cleaning, feature engineering, and data integration may be employed to transform the collected data into a format that can be used by machine learning models. Data normalization is necessary to ensure that the data is in the same range and scale, which facilitates better model training. Data cleaning is essential to remove any missing or duplicate data, which can adversely affect the model's performance. Feature engineering involves selecting the most relevant features that are important for predicting the target variable, and data integration involves combining data from different sources to create a unified dataset.

Feature Selection. It is another crucial step in machine learning, where the most relevant features are selected from the dataset to predict the target variable. Feature selection may involve statistical tests and feature ranking methods to identify the most critical features. Selecting the right features is essential to reduce model complexity and improve model accuracy.

Model Selection. It is the next step, where the appropriate machine learning algorithms for the problem at hand are chosen. Various algorithms such as Random forest, XGBoost, SVR and K-NN may be compared, and the best-performing model selected. The selected model must be appropriate for the problem's scale, complexity, and data availability.

Evaluation. Once the model has been selected, it is trained on the prepared dataset and evaluated to determine its performance. Different evaluation metrics such as R2 score, MSE, MAE and RMSE, may be used to measure the model's performance. The selected model is evaluated against a validation dataset to ensure that the model is generalizable and can accurately predict the target variable.

Maintenance and Monitoring. The final step involves monitoring the model's performance and maintaining it by updating it with new data and retraining it periodically to ensure its continued accuracy and relevance. This step is crucial to ensure that the model's performance does not deteriorate over time, and the model continues to provide accurate and relevant predictions.

Expected Project Contributions and Applications

This project will use models like Random Forest, KNN, SVR, XGBoost and combined together, these can potentially contribute to the field of supply chain management in several ways:

Improved Forecasting Accuracy. The use of predictive analysis techniques can help in improving inventory management and production planning. This, in turn, can help in reducing inventory costs and improving delivery times.

Real-Time Decision-Making. The combination of these models can allow for real-time decision-making in supply chain management, enabling companies to react quickly to changes in demand or supply, and adjust production and inventory levels accordingly.

Increased efficiency. By leveraging the power of predictive analytics, companies can optimize their supply chain processes, reduce waste, and improve overall efficiency. This can lead to significant cost savings and increased profitability.

Enhanced Customer Satisfaction. By improving supply chain efficiency, companies can improve their delivery times and order accuracy, leading to increased customer satisfaction and loyalty.

Advancements in the Field. The use of advanced machine learning models such as XGBoost and Random Forest in supply chain management can contribute to the development of new and innovative approaches to logistically improve delivery times.

Overall, the research can potentially provide significant contributions to the field of supply chain management by improving forecasting accuracy, enabling real-time decision-making, increasing efficiency, enhancing customer satisfaction, and advancing the field through the use of advanced machine learning techniques.

The proposed models will improve the accuracy by combining multiple weaker models, leading to better predictions and more informed decisions. Predictive analytics techniques can also help companies identify potential supply chain disruptions, such as delays in delivery or unexpected changes in demand, and take preventive measures to minimize the impact of these

disruptions. By analyzing the relationships between different variables, such as demand and price, companies can make data-driven decisions about inventory levels, production schedules, and supply chain optimization. If a company's regression analysis shows that a particular supplier has a history of delays, they can take steps to either find an alternative supplier or maintain safety stock levels to prevent stockouts.

One of the most significant real-world applications of predictive analysis in supply chain management is inventory management. By ensuring timely delivery, businesses can optimize inventory levels to meet customer needs while reducing excess inventory, leading to reduced storage costs and freeing up capital for other operations.

Another application is transportation management, where predictive analytics can help optimize routes and modes of transportation, reducing delivery times and transportation costs. Predictive analysis techniques can also identify potential disruptions in the supply chain, such as, political events, weather patterns or supplier performance. By identifying risks in advance, businesses can develop contingency plans to mitigate the effect of these disruptions, ensuring they can continue to meet customer demand.

Predictive analysis techniques can also help with supplier selection and management. By analyzing supplier performance data, businesses can identify reliable suppliers who deliver high-quality products on time and at reasonable costs, reducing the risk of supply chain disruptions and ensuring consistent quality for customers.

Project Requirements

Functional Requirements

Focusing on particular functional needs that focus on maximizing inventory levels, locating bottlenecks, and guaranteeing prompt product delivery to customers is necessary for

enhancing supply chain efficiency using predictive analysis. Implementation of predictive analytics starts with gathering data from the supply chain's sources, including suppliers, producers, distributors, and retailers. This is accomplished using a variety of techniques, including data scraping and API interfaces. In the context of rate limits, it is essential to adhere to the specific guidelines set by data providers to ensure smooth data retrieval. A maximum rate limit of 1000 API calls per hour was by the supplier's data source. Complying with these rate limits prevents disruptions in data acquisition and facilitates a consistent flow of information. The acquisition of data from suppliers, producers, distributors, and retailers necessitates appropriate licensing agreements to ensure lawful and ethical usage. Securing licenses from relevant data providers grants the necessary permissions to access and utilize their data. For instance, obtaining a data license from a major distributor might involve an annual fee allowing access to their comprehensive sales and inventory datasets although the dataset for this project is open source. Access permissions play a crucial role in maintaining data confidentiality and security throughout the project. By implementing a role-based access control system, individuals involved in the project are assigned specific access rights based on their responsibilities. Additionally, the data is scalable and stored securely, with appropriate backup and disaster recovery strategies in place. Gathering data from multiple sources and putting it into a common format for analysis is followed in data integration. Tools like ETL (extract, transform, load) procedures, data warehouses, or data lakes are used for this. Data integrity, consistency, and completeness is ensured during the integration process.

Data cleansing is done by finding and fixing data errors and inconsistencies. Techniques like data profiling, outlier detection, and data imputation are followed by using scalable solutions. Following this, large data volumes are supported by fault-tolerant, scalable distributed

databases. The security of data should be guaranteed. The type of data and business needs is taken into consideration while choosing a predictive analytics solution. This could entail utilizing statistical analysis software like Python or machine learning frameworks like TensorFlow or PyTorch. Tools like Tableau or Jupyter Notebooks are used for data exploration and visualization. Also, key performance indicators and business objectives are used to establish performance metrics. This involves processing data in real-time and calculating metrics like forecast accuracy or inventory turnover using programs. It is important to combine predictive analytics models with current supply chain management platforms like ERP and WMS. This entails integrating data flow between systems via APIs, or transforming data so that it is compatible with current systems. Infrastructure that is scalable and fault-tolerant is used for integration.

AI Powered Requirements

In today's market, improving supply chain efficiency through the use of AI-powered models and predictive analytics approaches offers faster lead times, lower costs, and higher customer satisfaction. It aims to make use of these strategies to create advanced models that can forecast demand, enhance inventory control, and simplify logistics.

KNN (K-Nearest Neighbors). In the realm of supply chain management, KNN models are leveraged to forecast the relationship between dependent and independent variables. These models prove valuable in predicting supplier performance, lead times, and shipping costs. By leveraging the collective predictions of neighboring data points, KNN enhances the accuracy of predictive models, effectively addressing a range of supply chain challenges, including demand forecasting and inventory optimization.

Random Forest. A powerful ensemble learning technique known as Random Forest finds utility in the supply chain domain. By constructing multiple decision trees and amalgamating their outcomes, Random Forest models excel in demand forecasting and inventory management. With the ability to capture complex relationships and patterns, these models enable accurate predictions, aiding in proactive decision-making and efficient supply chain operations.

XGBoost. In supply chain management, XGBoost emerges as a highly promising model. Extending the capabilities of gradient boosting, XGBoost combines an ensemble of decision trees to enhance prediction accuracy. This robust technique is effectively employed in addressing diverse supply chain challenges, including demand forecasting and inventory optimization. By leveraging the strengths of XGBoost, supply chain professionals can derive valuable insights to improve decision-making and operational efficiency.

Support Vector Regression (SVR). Within supply chain management, SVR plays a vital role in forecasting product demand and optimizing inventory levels. It is instrumental in identifying potential supply chain challenges or bottlenecks, allowing for proactive risk mitigation. For instance, if SVR predicts a surge in demand for a specific product, supply chain adjustments can be made to ensure sufficient inventory levels are maintained to meet the expected demand. SVR's ability to handle non-linear relationships and high-dimensional data makes it a valuable tool in supply chain forecasting and optimization.

These are a few illustrations of the kinds of models that could be applied in an AI-powered system to increase supply chain efficiency through the use of predictive analytics methods.

Data Requirements

Businesses must gather and analyze a variety of data types in the effort to improve on-time delivery using predictive analytics. This could include information on sales and demand, stock levels, measurements for suppliers' performance, effects from outside markets, manufacturing statistics, and historical trends. An extensive data set like this is essential for making accurate demand projections, adjusting inventory levels, reducing supplier-related risks, identifying potential bottlenecks, raising product quality, and cutting costs. Organizations may get valuable insights and take informed decisions by analyzing this abundance of data, enabling them to respond to customer needs more effectively, gain a competitive edge, and improve supply chain performance as a whole.

The two datasets provided by DataCo, a reputable provider of supply chain management services, for the analysis. The first of these, the "Product Dataset," provides in-depth information concerning the products, whereas the second, the "Order Dataset," contains extensive information about the orders as shown in Figure 1 and Figure 2. The "Product Dataset", provides in-depth information concerning the products, whereas the second, the "Order Dataset", contains extensive information about the orders. These datasets together include 44 columns and 180519 records total.

Figure 1

Snapshot of the Product Dataset

	Order Id	Product Card Id	Product Category Id	Product Description	Product Image	Product Name	Product Price	Product Status	Shipping Mode
0	77202.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
1	75939.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
2	75938.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
3	75937.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
4	75936.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class

Figure 2

Snapshot of the Order Dataset

	Type	Days for shipping (real)	Days for shipment (scheduled)	Benefit per order	Sales per customer	Delivery Status	Late_delivery_risk	Category Id	Category Name	Customer City	...	Order Item Profit Ratio	Order Item Quantity	Sales	Order
0	DEBIT	3.0	4.0	91.250000	314.640015	Advance shipping	0.0	73.0	Sporting Goods	Caguas	...	0.29	1.0	327.75	314.64
1	TRANSFER	5.0	4.0	-249.089996	311.359985	Late delivery	1.0	73.0	Sporting Goods	Caguas	...	-0.80	1.0	327.75	311.35
2	CASH	4.0	4.0	-247.779999	309.720001	Shipping on time	0.0	73.0	Sporting Goods	San Jose	...	-0.80	1.0	327.75	309.72
3	DEBIT	3.0	4.0	22.860001	304.809998	Advance shipping	0.0	73.0	Sporting Goods	Los Angeles	...	0.08	1.0	327.75	304.80
4	PAYMENT	2.0	4.0	134.210007	298.250000	Advance shipping	0.0	73.0	Sporting Goods	Caguas	...	0.45	1.0	327.75	298.25

The 'Product Dataset', in contrast, has fewer records. The attributes in this collection include a range of product-specific information, and each record represents a unique product. Although the datasets volumes vary, both are crammed with relevant data that may be used to optimize supply chain processes and speed up delivery. As a result of the abundance of data points, it is possible to undertake in-depth analyses that are both detailed and thorough, which opens the door for more successful supply chain initiatives.

Project Deliverables

Table 1 describes the deliverables and their respective due dates.

Table 1

Deliverables Including Reports, Prototypes, Development Applications, and/or Production Applications

Deliverable	Description	Due Dates
Project Proposal	Document outlining the objectives, scope, methodology, and expected outcomes for Improving Supply Chain efficiency	02/17/2023

Deliverable	Description	Due Dates
Work Breakdown Structure (WBS)	A hierarchical decomposition of the project's tasks into smaller, more manageable components following CRISP-DM methodology	03/03/2023
Gantt Chart	A visual representation of the project schedule, showing the start and end dates of every task and their dependencies.	03/10/2023
PERT Chart	A network diagram that displays the sequence of tasks and their dependencies, as well as the estimated durations and critical path.	03/17/2023
Project Management Plan	A document that describes how the project will be managed, including a project schedule, milestones, and resource allocation plan.	03/31/2023
Data Collection Plan	A document that outlines the specific data that will be gathered and how it will be collected.	03/31/2023
Data Exploration	Performing Exploratory Data Analysis on the retrieved data using data visualization and statistical analysis.	04/14/2023
Data Cleaning and Preprocessing Code	Code to perform data cleaning and preprocessing tasks, such as removing duplicates, filling in missing values, and transforming data types.	04/14/2023
Predictive Models	Code to build predictive models using exponential smoothing, double exponential smoothing, ARIMA, adaptive boosting to forecast demand, optimize inventory, and reduce lead times in the supply chain.	04/21/2023
Performance Metrics	Defining the key performance indicators that will be used to evaluate the efficiency of the predictive models.	05/05/2023
Model Evaluation/ Selection Report	A document that describes the process of selecting the best-performing predictive models and the evaluation criteria used.	05/05/2023
Production-Ready Predictive Models	Final versions of the predictive models that are optimized for production use.	05/12/2023

Deliverable	Description	Due Dates
Project Presentation	A presentation that summarizes the key aspects of the project, including the problem statement, methodology, findings, and conclusions.	05/12/2023
Final Project Report	A document that summarizes the project's objectives, methodology, findings, and conclusions.	05/17/2023

Note. The table provides a list of the key deliverables for the project. Each deliverable is described briefly in the second column.

The first deliverable is the Project Proposal, which will outline the objectives, scope, methodology, and expected outcomes of the project. The next three deliverables are planning documents that will help to structure the project: the Work Breakdown Structure (WBS), Gantt Chart, and PERT Chart. The three deliverables after that are related to data management and exploration: the Data Management Plan, Data Collection Plan, and Data Exploration Plan. These will define how data will be collected, stored, and managed throughout the project and outline the specific data that will be collected and how it will be analyzed. The Data Cleaning and Preprocessing Code deliverable will perform data cleaning and preprocessing tasks, while the Predictive Models deliverable will build predictive models to forecast demand, optimize inventory, and reduce lead times in the supply chain. The Performance Metrics deliverable will define the key performance indicators (KPIs) that will be used to evaluate the effectiveness of the predictive models.

The Model Evaluation and Selection Report will describe the process of selecting the best-performing predictive models and the evaluation criteria used. The Production-Ready Predictive Models deliverable will be the final versions of the predictive models that are optimized for production use. The Final Project Report will summarize the project's objectives,

methodology, findings, and conclusions, while the Project Presentation will be a summary of the key aspects of the project. Completing these deliverables will help to ensure the successful completion of the project and provide a clear roadmap for the project team.

Technology and Solution Survey

The technological survey presents several models and methods that are commonly used for demand forecasting in supply chain management. Each model has its strengths and weaknesses, and the choice of model depends on the specific problem and characteristics of the data. The technological survey covers several machine learning models and time series forecasting methods that are commonly used for demand forecasting in supply chain management. These models include XGBoost Regressor, Support Vector Regression (SVR), K-nearest neighbors (KNN), and Random Forest.

Vairagade et al. (2019) developed a demand forecasting model using RF and ANN and compared their performance with traditional time series forecastings models such as ARIMA and exponential smoothing. The authors found that both RF and ANN outperformed the traditional models regarding forecasting accuracy, with RF showing slightly better performance than ANN. Random forest (RF) has shown promising results for demand forecasting in supply chain management, as demonstrated in the research paper.

GuangHui (2012) found that SVR outperformed traditional time series forecastings models such as ARIMA and exponential smoothing in predicting demand for supply chain management. The study noted that SVR can effectively capture nonlinear relationships in data, which is particularly relevant in supply chain management where demand patterns can be influenced by various factors.

Kilimci et al. (2019) developed an improved demand forecasting model using a deep learning approach and proposed a decision integration strategy. The authors used a variant of recurrent neural networks (RNNs) called long short-term memory (LSTM) networks for demand forecasting and compared their model with other popular time series forecasting models such as ARIMA, exponential smoothing, and ANNs. The authors found that the LSTM-based deep learning model outperformed the traditional models regarding forecasting accuracy.
While Kilimci et al. (2019) did not use RF in their study, the comparison between LSTM-based deep learning and ANNs suggests that machine learning models, in general, may perform better than traditional models such as ARIMA and exponential smoothing.

According to the research paper by Praveen et al. (2019), ANN modeling can be an effective approach for improving inventory management and reducing supply chain costs. The authors used a time-series forecasting model based on ANN to forecast future demand for a product in a supply chain. The model was trained using historical sales data and other relevant factors such as seasonality, promotions, and holidays. The authors reported a significant improvement in the inventory turnover ratio and a reduction in inventory holding costs.

In summary, both ANN and RF have shown promising results for demand forecasting in supply chain management. RF has a slight advantage over ANN. The models have the advantage of being able to capture complex nonlinear relationships between inputs and outputs, making them suitable for modeling time-series data with multiple factors affecting demand.

The research papers by Seyedan and Mafakheri (2020), Lee and Mangalaraj (2022), and Aamer et al. (2020) indicate that ANNs have shown promising results in terms of accuracy and efficiency in demand forecasting. They are capable of learning nonlinear relationships between variables and can make accurate demand predictions even when data is noisy or has missing

values. They can also handle large and complex datasets with a high number of features. They are versatile and can handle a variety of input data types, including time series data, categorical data, and continuous data.

The use of ANNs for demand forecasting in supply chain management has been discussed in several research papers by Seyedan and Mafakheri (2020), Aamer et al. (2020), and Lee and Mangalaraj (2022).

The authors, Islam and Amin (2020) used a dataset of inventory and sales data for a company that sells electronic components. The researchers attributed Random Forest's superior performance to its ability to handle large datasets and identify complex relationships between variables. It is an ensemble learning method that creates multiple decision trees and combines their predictions to make a final decision. This approach allows Random Forest to avoid overfitting and improve accuracy. The study suggests that it is a promising machine-learning technique for predicting backorders and improving supply chain efficiency. Its ability to handle complex and large data and its high accuracy and processing speed makes it a suitable tool for supply chain managers to forecast demand and plan inventory levels.

In the paper by Sarhani and Afia (2014), the authors support vector regression (SVR) which was used as an intelligent system for supply chain demand forecasting. The authors aimed to improve the accuracy of demand forecasting by applying SVR to a real-world dataset from a Moroccan company. The SVR model was compared to other traditional models such as linear regression, moving average, and exponential smoothing. The results of the study showed that SVR outperformed the traditional models in terms of accuracy and reliability. The authors found that SVR had a higher forecasting accuracy, as well as lower root mean squared error (RMSE) and mean absolute percentage error (MAPE) values compared to the traditional models.

The paper by Seyedan and Mafakheri (2020) highlights the use of Decision Trees for demand forecasting, given their ability to handle nonlinear relationships between variables and provide interpretable models. Similarly, the paper by Aamer et al. (2020) suggests that Decision Trees are effective in demand forecasting and outperform other machine learning models in terms of accuracy and speed.

Furthermore, the paper by Lee and Mangalaraj (2022) recommends Decision Trees for demand forecasting in supply chain management due to their ability to handle both categorical and numerical data and provide interpretable results. Compared to other machine learning models such as K-Nearest Neighbors, Artificial Neural Networks, and Decision Trees, Random Forests are easier to interpret and provide actionable insights for decision-making. They are also more computationally efficient and can handle high-dimensional datasets, missing values, and outliers, which are common in supply chain management datasets.

Literature Survey of Existing Research

The paper by Mitra et al. (2022) focuses on developing a demand forecasting model for a multi-channel retail company using a hybrid machine learning approach. The authors conducted a comparative study of various demand forecasting models, including time-series models, machine learning models, and hybrid models, to determine which approach is the most effective in predicting demand. The study used a dataset from a real-world multi-channel retail company and evaluated the models' performance based on several metrics, such as Mean Absolute Percentage Error (MAPE) and Root Mean Squared Error (RMSE). The authors developed a novel hybrid machine-learning approach that combines time series and machine-learning models to improve the accuracy of demand forecasting. They found that the proposed hybrid model outperformed all other models in predicting demand for the retail company. The paper concludes

that the proposed hybrid machine learning approach is an effective method for demand forecasting in multi-channel retail companies and can provide valuable insights for decision-making processes related to inventory management and supply chain management.

Cadavid et al. (2018) provide an overview of the recent trends in applying machine learning techniques to demand and sales forecasting. For this paper, a systematic literature review to analyze relevant studies from various databases was conducted and three major goals were identified, namely improving forecasting accuracy, reducing forecasting error, and enhancing forecasting efficiency. The reviewed results indicated that machine learning techniques have been increasingly used in demand and sales forecasting and have demonstrated better performance compared to traditional statistical methods. However, it also highlights some limitations and challenges of using machine learning techniques in this domain, including data availability and quality, model interpretability, and ethical issues. Overall, the paper provides valuable insights into the current trends in applying machine learning to demand and sales forecasting and highlights the potential benefits and limitations of using these techniques in real-world applications.

The research paper by Kot et al. (2011) aims to develop a theoretical framework for inventory management based on demand forecasting. The authors review the existing literature on inventory management and demand forecasting to identify the main challenges and opportunities in this field. They argue that accurate demand forecasting is crucial for effective inventory management and propose a theoretical framework that integrates these two areas. The paper discusses the key elements of the proposed framework, including demand forecasting models, inventory control policies, and performance evaluation methods. The authors emphasize

the importance of selecting appropriate forecasting models and inventory policies based on the product, market, and supply chain characteristics.

Xi and Sha (2014) reviewed the existing literature on inventory management and demand forecasting to identify the main challenges and opportunities in this field. They propose an inventory management system that integrates demand forecasting, inventory control policies, and optimization techniques to improve inventory performance. The paper discusses the key elements of the proposed system, including demand forecasting models, inventory control policies, and optimization techniques such as linear programming and simulation. The authors emphasize the importance of selecting appropriate forecasting models and inventory policies based on the product, market, and supply chain characteristics.

The research paper by Toktay and Wein (2001) explores a forecasting production inventory system with stationary demand, with the goal of understanding how various inventory and production control policies affect system performance. The authors generated simulated demand data using a Monte Carlo simulation method to test the system's behavior under different conditions. One potential limitation of the study is that it assumes a stationary demand process, which may not reflect real-world scenarios where demand fluctuates over time. Additionally, the study only considers a single product and does not account for other external factors that could impact the system's performance. The study's findings suggest that the performance of the forecasting-production-inventory system is highly dependent on the chosen control policies. Overall, the research paper provides valuable insights into the behavior of forecasting-production-inventory systems, highlighting the importance of carefully selecting inventory and production control policies to optimize system performance.

The author Schlegel (2014), argues that traditional methods of managing supply chain risk, such as safety stock and contingency planning, are no longer sufficient to address the complexities and uncertainties of today's business environment. The article explores how big data can be used to collect and analyze data from various sources, such as social media and sensors, to identify potential risks and opportunities. By using predictive analytics, supply chain managers can leverage big data to forecast demand, optimize inventory, and identify potential disruptions in the supply chain. The article provides real-world examples of how big data and predictive analytics have been successfully used to manage supply chain risk, including real-time monitoring of inventory levels and addressing supplier risk. The article suggests that big data and predictive analytics can provide valuable insights to supply chain managers for managing risk and improving supply chain performance. The author emphasizes the importance of investing in data infrastructure, analytics expertise, and data-driven decision-making processes to fully leverage the potential of big data and predictive analytics in supply chain management.

The study by Dhanush et al. (2021) aims to solve the challenges of supply chain management, such as the lack of transparency, inefficiency, and high operational costs, by proposing a blockchain-aided predictive time series analysis model. The authors used two datasets from a supply chain system for electronic products to test the proposed model. The first dataset consisted of time series data of the order delivery process, while the second dataset included blockchain data related to the same order delivery process. The paper acknowledges several limitations and challenges, including the small size of the dataset and the complexity of the proposed model. Additionally, the study assumes that all supply chain actors will use the same blockchain platform, which may not always be feasible in practice. The results of the study show that the proposed model can improve the accuracy of demand forecasting by 15% and

reduce the lead time of order delivery by 30%. The authors also found that the use of blockchain technology can enhance the transparency and traceability of the supply chain system, leading to improved efficiency and reduced operational costs. The paper proposes a blockchain-aided predictive time series analysis model for supply chain management and highlights the potential of blockchain technology to improve transparency and traceability in the supply chain system.

The authors Sheremetov et al. (2013), emphasize the importance of accurate forecasting in the oil and gas supply chain, where demand and production are subject to various uncertainties and fluctuations. The paper reviews the existing literature on time series forecasting and identifies the most commonly used techniques, including ARIMA models, exponential smoothing, and artificial neural networks. The article provides examples of how time series forecasting techniques have been applied to the upstream oil and gas supply chain, such as forecasting production rates, estimating reserves, and optimizing drilling schedules. The authors demonstrate the benefits of using time series forecasting techniques, such as improved accuracy, reduced costs, and better decision-making. The paper suggests that time series forecasting techniques can provide valuable insights for managing the upstream oil and gas supply chain. The authors recommend that oil and gas companies invest in data infrastructure and analytics expertise to fully leverage the potential of time series forecasting techniques for improving forecasting accuracy and supply chain performance.

The aim of the paper by Carbonneau et al. (2008) is to explore the use of machine learning techniques for supply chain demand forecasting, specifically focusing on the application of artificial neural networks (ANNs) and support vector machines (SVMs) to predict future demand for a particular product. The dataset used in the paper consists of historical sales data for a consumer product, along with information about promotions, marketing campaigns, and other

factors that may impact demand. The authors use this data to train and test their machine learning models, and evaluate their performance based on metrics such as mean absolute percentage error (MAPE) and root mean squared error (RMSE). The authors also discuss several limitations and challenges in the use of machine learning for demand forecasting, including the need for high-quality data, the difficulty of capturing complex demand patterns, and the risk of overfitting when training models on historical data. They propose several strategies for addressing these challenges, such as incorporating external data sources and using ensemble techniques to combine multiple models. In terms of results, the authors find that both ANNs and SVMs are effective for demand forecasting. They also note that the inclusion of external data sources can significantly improve the accuracy of the forecasts. The authors conclude that machine learning has the potential to significantly improve demand forecasting in supply chain management, but that further research is needed to address the challenges and limitations identified in the study.

The authors, Zwißler and Hermann (2012), explain the types of supply chain risks that the electronics industry faces, such as product obsolescence, quality issues, and supply chain disruptions. The chapter stresses the importance of identifying and mitigating supply chain risks to ensure the resilience and sustainability of the supply chain. The paper presents a framework for managing supply chain risks, which involves identifying, assessing, mitigating, and monitoring risks. The authors also discuss the role of information technology in supply chain risk management, suggesting that real-time monitoring of supply chain data can help to identify and address potential risks. The paper provides examples of how supply chain risk management has been applied in the electronics industry, including contingency planning and supplier evaluation systems. In conclusion, the paper suggests that effective supply chain risk management is crucial for the long-term success of the electronics industry. The authors recommend that companies in

the electronics industry adopt a comprehensive and integrated approach to supply chain risk management, including risk identification, assessment, mitigation, and monitoring, as well as the use of information technology to support these efforts.

The goal of the paper by Praveen et al. (2019) is to find out if using artificial intelligence (AI) can help a pharmaceutical company manage its inventory better and reduce costs. The authors studied one company and its distribution centers. The authors looked at past sales and inventory data to create a new way of predicting how much of each product will be sold in the future. They also used a type of computer program called an artificial neural network (ANN) to figure out how much inventory to keep at each distribution center. The authors talked about some of the challenges they faced in creating this new approach, like the fact that the supply chain process is complicated and that it's important to have good data to make accurate predictions. The results of the study showed that using the new approach did lead to cost savings and better inventory management. The AI-based time-series forecasting helped to reduce the amount of inventory that had to be kept, while the ANN modeling helped to make sure there was enough of each product available. The authors suggest that using AI in this way could be helpful for other companies too, but they also point out that more research is needed to see if this approach can be improved even further.

The main objective of the paper by Khan et al. (2020) is to develop an accurate demand forecasting model for the retail industry by leveraging business intelligence and machine learning techniques. The traditional forecasting methods have limitations in handling complex data sets and dynamic market trends. Therefore, the authors proposed a model that can address these limitations and provide more accurate demand forecasting. The authors used a dataset from a retail store chain in South Korea that included daily sales data for a period of two years. The

data contained information about products, stores, and customers. Additionally, external data sources such as weather data were used to further enhance the accuracy of the forecasting model. The paper highlights some of the challenges and limitations associated with the proposed model, including managing and analyzing large amounts of data, the complexity of the model, and the lack of consideration of external factors like economic conditions and competition. The results of the study show that the proposed model outperforms traditional forecasting methods and can provide an accuracy improvement of up to 15%. The paper presents an effective demand forecasting model for the retail industry that leverages business intelligence and machine learning techniques. The study emphasizes the potential of these techniques to improve the accuracy of demand forecasting and suggests future research to address the limitations and challenges of the proposed model.

The paper by Tarallo et al. (2019) explores the use of machine learning in predicting the demand for fast-moving consumer goods (FMCG). The study uses historical sales data from a Brazilian retail chain to develop a predictive model for FMCG demand. The goal is to determine whether machine learning techniques can provide more accurate predictions than traditional statistical methods. The methodology used in the study involves the application of three machine learning models - Random Forest, Gradient Boosting, and Support Vector Regression - to the sales data. The models are evaluated based on their ability to predict future demand accurately. The results suggest that machine learning techniques can indeed improve the accuracy of FMCG demand forecasting, with the Random Forest model performing the best. The study has some limitations, including the use of data from a single retail chain in Brazil and a limited number of FMCG products. The authors suggest that future research should explore the applicability of the proposed model to other retail chains and a more extensive range of FMCG products. Overall,

the paper provides insights into the potential of machine learning in improving demand forecasting accuracy for FMCG products.

A paper by Belhadi et al.(2021) presents a novel approach for predicting credit risk of SMEs investing in agriculture 4.0 through supply chain finance. The study employs an ensemble machine learning approach that combines several models to improve prediction accuracy. The authors collected data from a Moroccan bank that provides supply chain finance to agricultural SMEs, including information on their financial performance, supply chain partners, and other factors that could influence credit risk. The data underwent preprocessing, and feature selection techniques were applied to identify the most relevant variables for the model. Results show that the model can help financial institutions assess the creditworthiness of SMEs investing in agriculture 4.0 through supply chain finance. The paper contributes to the growing literature on the use of machine learning techniques in financial risk management and highlights the importance of using advanced machine learning methods in financial decision-making. However, the study's limitations include a small sample size and the use of data from a single bank in a specific country. The authors suggest that future research should investigate the proposed model's applicability in other countries and industries. This paper has practical implications for banks and financial institutions that provide supply chain finance to agricultural SMEs.

The paper by Filali et al. (2021) explores the applications of machine learning (ML) in supply chain management. The goal of the study is to identify and analyze various ML techniques used in supply chain management and their potential benefits. The authors conducted a literature review to identify relevant studies and articles published between 2015 and 2020. They classified the studies based on the type of ML technique used and the supply chain management function, such as demand forecasting, inventory management, and logistics

optimization. The authors found that ML techniques, including artificial neural networks, support vector machines, and decision trees, have been applied in various supply chain management functions, such as demand forecasting, inventory management, and logistics optimization. The results suggest that ML techniques can improve supply chain performance by providing accurate and timely information for decision-making. However, the authors also noted several limitations, including the need for high-quality data, the complexity of ML algorithms, and the need for expertise in ML and supply chain management. Overall, the paper provides valuable insights into the potential benefits of using ML techniques in supply chain management and highlights the need for further research in this area. The study has practical implications for supply chain managers, who can use ML techniques to improve their decision-making and enhance their supply chain performance.

The paper by Aviv (2003) presents a time-series approach to inventory management in supply chains. The objective of the study was to develop a mathematical framework for predicting inventory levels based on past demand patterns. The authors argue that traditional inventory management methods that rely on constant demand assumptions may not be suitable for modern supply chains where demand patterns are often variable and uncertain. The methodology used in the study involves developing a time-series model that uses historical demand data to predict future demand patterns. The model is then used to determine optimal inventory levels based on expected demand and supply chain costs. The results of the study suggest that the time-series approach can improve inventory management in supply chains by providing more accurate demand forecasts and reducing inventory costs. The model does not account for lead times, which can be a significant factor in inventory management. Overall, the

paper provides a valuable contribution to the literature on supply chain inventory management and highlights the potential benefits of using time-series models for predicting demand patterns.

The paper by Parzen (1961) presents a new approach to time series analysis, with a particular focus on smoothing methods. Parzen introduces a class of methods for smoothing and interpolating time series, based on convolution and generating functions, and shows that this approach provides a unified framework for a variety of smoothing techniques. The paper does not use any specific dataset, but provides a theoretical framework for analyzing time series data. One limitation/challenge of this approach is the need to choose an appropriate smoothing parameter, which can affect the performance of the method. Additionally, the paper assumes a stationary time series, which may not always be applicable to real-world data. Overall, the paper concludes that the proposed approach provides a powerful tool for analyzing time series data, particularly in the context of forecasting and prediction, and can be applied to a variety of fields, including economics, engineering, and meteorology.

The goal of the paper by Aamer et al. (2020) is to review the existing literature on the use of machine learning in demand forecasting for supply chain management. The authors aim to provide insights into the current state of research, identify gaps and future directions for research, and present practical implications for supply chain managers. The study reviews various machine learning models, such as neural networks, decision trees, and support vector machines, that have been used for demand forecasting in supply chain management. The authors also discuss the data preprocessing and feature selection techniques employed in these studies. The study found that machine learning models have shown promising results in improving demand forecasting accuracy in supply chain management. The authors also identified several limitations, such as the need for large amounts of data and the difficulty of interpreting the results of some machine

learning models. The paper highlights the importance of selecting appropriate machine learning models based on the nature of the data and the forecasting task, as well as the need for further research on the integration of machine learning models into existing supply chain management systems. The findings of the paper have practical implications for supply chain managers in terms of improving demand forecasting accuracy and enhancing supply chain performance.

Data & Project Management Plan

Data Management Plan

Data Collection Approaches

Efficient data collection is an essential aspect of improving supply chain efficiency in the market through predictive analysis. Data collection approaches involve various methods of gathering, and integrating data from different sources such as the data from existing sources such as industry reports, government databases, or market research studies. Internal data sources like sales, inventory, and shipping data, and external sources such as weather and social media data are used. Historical data provides insights into past supply chain performance to identify areas for improvement and develop predictive models that anticipate future supply chain disruptions. Predictive analytics techniques are employed to analyze the data and identify trends, patterns, and insights which are utilized to optimize the supply chain process and enhance efficiency. The selection of data collection methods is dependent on the specific project goals, data sources, and ethical and legal implications of data collection and use.

Data Management Methods

Data management methods are necessary for making informed decisions, improving business processes and protecting sensitive data for efficiency of the project which is focused on predictive analysis. These methods ensure that the data collected is accurate, reliable, accessible,

and secure and these include integration, quality, security, governance, and preservation of data. In data integration the data is combined from multiple sources, while data quality ensures the accuracy and consistency of the data and data security is important for protecting the data from unauthorized access or corruption. Data governance establishes policies and procedures for managing data and it also ensures compliance with relevant regulations and standards. Finally, data preservation ensures the long-term preservation and accessibility of the data. Carefully selecting and integrating data management methods, will help derive valuable insights from the data and make impactful changes to their supply chain operations.

Data Storage Methods

Storing data on AWS provides several benefits, including scalability, security, and accessibility. AWS allows to store and retrieve any amount of data, from anywhere on the web, making it highly accessible. For the project aimed at improving supply chain efficiency, storing the dataset on AWS can be a suitable option since it offers scalable and secure storage solutions for organizations of all sizes. Amazon S3 is a durable object and highly scalable storage service that provides cost-effective storage for large datasets.

Git is used for version control in the project which provides several benefits, including ease of use and flexibility. It also allows the creation of branches for the project for development and testing purposes, making it easy to experiment with new features without affecting the main codebase. Additionally, Git provides a robust set of tools for code and data management, including merging, and conflict resolution, ensuring that changes to the code and data are properly tracked and managed, certifying that the project is always up-to-date and accurate.

The data transformation process is converting the raw data to analyze and process easily, allowing to gain insights and make informed decisions. Several techniques are used for data

transformation, including data cleaning, data aggregation, and data normalization. Data cleaning where all inconsistencies, errors, and duplicate records are removed from the dataset to ensure accuracy and consistency. Data aggregation involves grouping and summarizing data to identify trends and patterns, providing valuable insights into supply chain efficiency. In Data normalization the data is organized into a standard format, reducing redundancy and minimizing data anomalies. Normalization is applied to improve the accuracy and consistency of the data, making it easier to analyze and process. Machine learning algorithms are used for data transformation and these algorithms automatically identify patterns and trends in the data, helping to make data-driven decisions that improve supply chain efficiency (see Appendix A).

The folder structure is designed to be intuitive and help ourselves easily locate and access the data they need. A typical structure is that it includes top-level folders for raw data, cleaned data, processed data, and analysis results. Within each folder, subfolders are created to categorize data based on specific criteria. The file naming convention is consistent, informative, and designed to enable ourselves to easily identify the content of each file. The convention includes elements such as the data source, date and time stamp, and a description of the data. Using a clear naming convention, helps easily understand the content of each file, reducing the time required to search for specific data.

Clear guidelines for all the folder structure, file naming convention, and file format is established at the outset of the project to ensure that all team members follow the same practices. This helped in managing the data easily, sharing and analyzing effective decision-making and improving supply chain efficiency.

Data Usage Mechanisms

Data usage mechanisms refer to the processes and methods used to access, manipulate, and analyze data. It helps in improving supply chain efficiency as it enables effective management, manipulation, and analysis of data. Storing data in GitHub involves creating a new repository and adding relevant data files to it. Git manages changes to data over time by creating branches for development and testing purposes, tracking changes to code and data, and merging changes to the main branch. It also provides a platform for managing data, keeping track of changes, team collaboration, and ensuring that the project is properly managed and backed up resulting in improved supply chain efficiency. This empowers the team to make data-driven decisions and optimize the supply chain process for maximum efficiency.

Data ownership is essential to identify who owns and is responsible for the data which is used in the project. Clear roles and responsibilities are defined in Table 2. Standards for data quality are established to ensure that the data is accurate, complete, consistent, and relevant. The data security policies and procedures are developed to safeguard data from unauthorized access, modification, or disclosure. Data access and sharing the data and how it can be shared is determined and ensures that privacy and confidentiality are maintained.

Table 2

Task Responsibilities and Descriptions

Task	Responsible Member	Task Description
Data collection	Gouri Benni	Ensure that the data is collected from reliable sources, is of high quality, and is stored in a secure manner

Task	Responsible Member	Task Description
Ethics and legal compliance	Deekshita Prakash Savanur	Ensure that project team adheres to ethical standards in the collection, handling, usage of data and also whether the project team is in compliance with laws and regulations
Reports and visualization	Krishna Sameera Surapaneni, Sonali Arcot	Analyze the data, draw insights and conclusions, and present the findings in a clear and concise report; responsible for visualizing the data in graphs, charts, and other visualizations

Note. The table describes the tasks and responsibilities of each team member for data collection and usage.

Project Development Methodology

Efficiency in the supply chain is a crucial component that can affect the overall success. Supply chain strategies can be enhanced by analyzing vast amounts of data to obtain insights and forecast future events with the aid of predictive analytics techniques. Big data analytics or Intelligent System Development Life Cycle (ISDLC) are used to create these intelligent systems. Breaking down the project into smaller, manageable tasks is an important step in ensuring the success of the project which is focused on improving supply chain efficiency. According to Schröer et al. (2021), the challenges of incorporating the CRISP-DM process model with other methods and frameworks are determined, and the requirement for domain-specific adaptations when using it in various contexts. It emphasizes the significance of addressing these difficulties in order to successfully implement the model in practical applications. The first objective is the problem identification step where the issues are specified i.e shorter lead times, late shipment,

delay in delivery, and increasing customer satisfaction, that the intelligent system will be trying to solve. There are other factors involving figuring out the problem's scale and the project's precise goals, and locating important stakeholders.

The second stage is the data collection step where pertinent data from diverse sources are collected. The dataset will include information regarding the flow, goods quantity, location etc. For the model to be accurate and dependable, the information should be varied and indicative of supply chain processes. This involves collecting relevant data from various sources such as DataCo, DataUN, etc. Data preprocessing is required to guarantee that the data is prepared for analysis after it has been collected. Data from several sources can be merged into a single dataset using data integration techniques. The collected data contained inconsistencies, errors, and duplicates, which were affecting the quality of the data. Therefore, the data is cleaned and processed to ensure that it is accurate and consistent. Once the data is cleaned, it is transformed so it can be easily analyzed and processed which involves techniques such as data integration, data aggregation, and data normalization.

In the data analysis step, the trends and patterns in the data can be found using machine learning approaches like random forest, support vector regression, k-nearest neighbors, and XGBoost. A predictive model can be created using the previous analysis in the model development step. The ML model is trained using the historical data and tested to make sure it is correct. Model evaluation step entails evaluating the model's performance using performance metrics like accuracy, and precision and comparing anticipated and actual outcomes.

The data analysis results are presented in a graphical or visual format, using Tableau, MatplotLib and Seaborn as visualizations tools for making it easier to interpret and understand. This involves creating charts, graphs, and other visual representations of the data, highlighting

trends and patterns that may not be easily identifiable from the raw data. Interactive and dynamic dashboards in Tableau allow users to explore data in real-time and create dashboards, reports, and visualizations. Matplotlib is used to create static and interactive visualizations for analyzing and communicating data insights, while Seaborn specializes in statistical data visualization, creating visually appealing charts such as heat maps and violin plots.

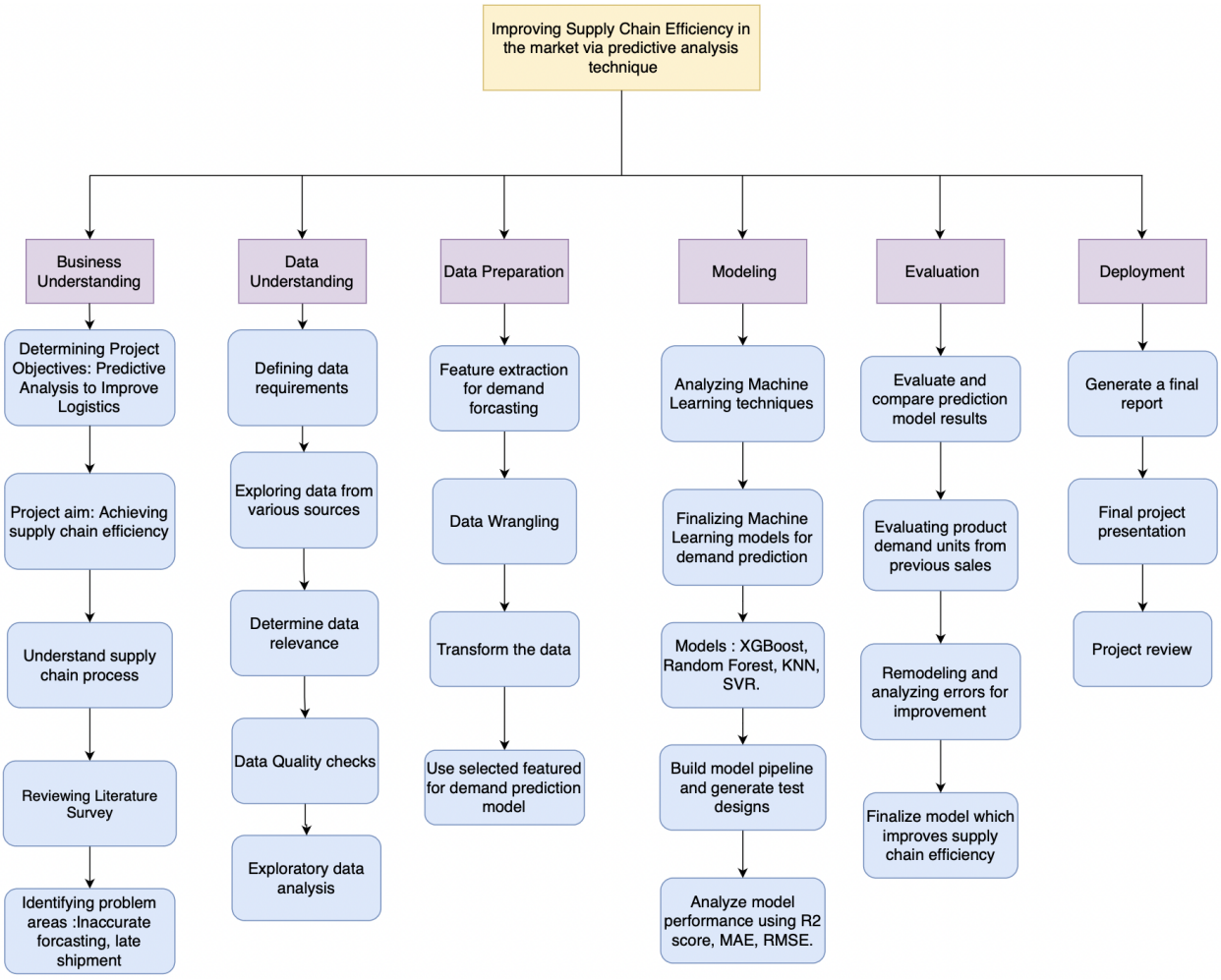
Finally, deployment of the model can be done into systems for supply chain management and will allow it to be utilized to optimize a number of different supply chain operations. This can entail delivery delays of a product. As new information becomes available, the model can be revised and improved over time, ensuring that it stays precise.

Project Organization Plan

Figure 3 describes the predictive analytics techniques to improve supply chain effectiveness in the marketplace. It entails data gathering and preparation, the creation and validation of prediction models, and the use of data analysis to spot supply chain inefficiencies. Proposed optimization techniques will be implemented, followed by monitoring and effectiveness testing. In the end, the project will result in a stronger supply chain system, which will boost customer satisfaction, reduce delivery delays, cut expenses, and raise revenue.

Figure 3

Work Breakdown Structure (WBS)



Business Understanding Phase

Business understanding in a project involves gaining a deep understanding of the business problem or objective that the project is intended to solve, and identifying the relevant data sources, variables and metrics needed to achieve this objective. It helps in setting realistic goals and developing an effective strategy for implementing the project. The Business Understanding phase for this project consisted of the following tasks:

Determining the Project Objectives. The initial step in the Business Understanding phase is project objectives definition. In this case, the objective was to improve supply chain efficiency in the market through predictive analysis techniques.

Understanding the Supply Chain Processes. To enhance supply chain efficiency, there is a need to gain a deep understanding of the supply chain processes involved in the market. This includes identifying the primary stakeholders in the supply chain, determining the flow of goods and services, and pinpointing the critical points in the supply chain process.

Conducting a Literature Survey. A complete review of the current state of supply chain management is conducted, including any previous work related to predictive analysis techniques in the industry. This helps identify any existing solutions or best practices and reveals any gaps or limitations in the current literature.

Identifying Problem Areas. Two specific problem areas in the supply chain were identified, that the project aims to address, which are inaccurate predicting and late shipment.

Data Understanding Phase

Data understanding in a project involves acquiring and exploring the relevant data sources, evaluating their quality and relevance, and gaining a comprehensive understanding of the variables and relationships between them. This phase helps in the next phase, that is identifying the data preparation requirements, selecting the appropriate features, and refining the project objectives and methodology. For the project, the Data Understanding phase consisted of the following tasks:

Defining Data Requirements. The first step is to identify the data required to achieve the project's objectives. The necessary data fields and sources were determined, including historical shipment and order data, inventory levels, and supplier information.

Exploring Data from Various Sources. The data is collected from various sources such as DataCo, DataUN. The data is extracted, transformed, and loaded into a centralized data repository for analysis.

Determining Data Relevance. After collecting the data, its relevance to the project objectives is assessed. The data fields are reviewed and those that are relevant to supply chain visibility and predictive analysis are identified.

Performing Data Quality Checks. The data quality checks are conducted to ensure the accuracy and consistency of the data. Any missing values, duplicates, outliers, and data inconsistencies are identified.

Conducting Exploratory Data Analysis. EDA is done to gain insights into the data and identify any patterns or relationships.

Data Preparation Phase

Data preparation in a project involves processing, transforming and cleaning the raw data obtained in the data understanding phase to create a structured, accurate and relevant dataset for analysis. This phase includes selecting and extracting the relevant features, cleaning and handling missing or invalid data, and transforming the data into the appropriate format. Proper data preparation is critical to ensure the accuracy and effectiveness of the machine learning models. For the project, the Data Preparation phase consisted of the following tasks:

Defining Feature Extraction. The key features that are most relevant to the problem stated are identified, including historical shipment data, sales data, inventory levels, and supplier information etc. These features are selected based on their ability to contribute to the accuracy of the predictive model.

Data Wrangling. Data wrangling is performed to clean and prepare the data for analysis. This involves correcting the identified missing values, outliers, and inconsistencies in the data. It ensures that the data is in the proper format for analysis.

Transforming the Data. The data is transformed to enhance its usefulness for the predictive model. Data normalization, scaling, and feature engineering are performed to improve the accuracy of the model. This includes techniques such as one-hot encoding, feature scaling, and data imputation.

Creating a Data Pipeline. Finally, a data pipeline is created that streamlines the process of data preparation, feature selection, and model building. The pipeline enables the team to automate several tasks and reduce the time required for model development.

Modeling Phase

Modeling phase in a machine learning project involves developing and testing various machine learning models to find the best fit for the project objectives. This phase includes selecting the appropriate algorithm, fine-tuning the model hyperparameters, and analyzing the performance of the model using various metrics. The modeling phase is crucial in building accurate and effective predictive models for the project. For the project, the Modeling phase consisted of the following tasks:

Analyzing Machine Learning Techniques. Various machine learning techniques and algorithms suitable for the prediction are analyzed, including RF, KNN, SVR, SVM and XGBoost. The strengths and weaknesses of each approach are identified and the most appropriate ones for the project are selected.

Finalizing Machine Learning Models. Based on the analysis, the machine learning models that would be used for demand prediction are finalized. The models are selected such that

they provide the best accuracy and performance while also taking into account the complexity and interpretability of the models.

Building Models. The selected machine learning models are built, including RF, SVR, and XGBoost. Ensemble techniques are used to combine multiple models to improve the prediction accuracy.

Building Model Pipeline and Generating Test Designs. A model pipeline that automates the process of data preprocessing, feature extraction, and model building is built. Test designs are also generated for model performance evaluation using techniques such as cross-validation and train-test split.

Analyzing Model Performance. The performance of the models are analyzed using metrics such as accuracy score, ROC curve, and confusion matrix. Sensitivity analysis is conducted to check the robustness of the models under different scenarios and data conditions.

Fine-tuning Models for Better Performance. Based on the analysis, the models are fine-tuned by adjusting hyperparameters, feature selection, and model architecture. The process is repeated for model building, testing, and evaluation until the achieve the desired level of accuracy and performance.

Evaluation Phase

The evaluation phase in a machine learning project involves assessing the performance of the model and its ability to meet the project objectives. This phase includes comparing the performance of different models, evaluating the accuracy of predictions using appropriate metrics, and identifying areas for improvement. Proper evaluation is crucial to ensure that the machine learning model is reliable and can make accurate predictions. For this project, the Evaluation phase consisted of the following tasks:

Evaluating and Comparing Predictive Model Results. The results of the developed models are evaluated and compared to identify the best-performing model that could accurately predict product demand. Various performance metrics such as R-squared, MAE, and MSE are used to assess the models' accuracy and reliability.

Evaluating Delivery Delays from Past Delays. The product delivery delays from previous data are evaluated to determine the accuracy of the developed models. The predicted values are compared with the data to identify any discrepancies and improve the models accordingly.

Remodeling and Analyzing Errors for Improvement. The developed models are remodeled and the errors are analyzed to improve the models' performance. The sources of errors are identified and necessary changes are made to the models to minimize the errors and improve accuracy.

Finalizing the Model which Best Improves Supply Chain Visibility. After analyzing the model results and errors, the model that best improved supply chain visibility is finalized. The model that provided the most accurate and reliable predictions of delay in delivery and aligned with the project aim is selected.

Deployment Phase

Deployment in a project involves integrating the developed models into the production environment and making them available for use by end-users. This phase includes generating a final report, testing the models in real-world scenarios and providing user training and support. The deployment phase is crucial to ensure that the developed models are utilized effectively and produce the desired results. For this project, the Deployment phase consisted of the following tasks:

Schedule Plan of Action for Deployment. A plan of action is developed for deploying the developed models in a production environment. The plan includes timelines, resources, and responsibilities to ensure a smooth deployment process.

Generate Final Report. A final report is generated that summarizes the project's objectives, methodologies, and findings. The report included details about the project background and introduction, data and project management plan, data engineering and model development.

Project Review and Presentation. The review of the entire project is conducted, highlighting the successes and challenges encountered during the project's implementation.

Project Resource Requirements & Plan

Hardware Requirements

Hardware requirements refer to the specific hardware components needed for a device or system to perform particular tasks or functions effectively. The requirements may vary depending on the software and applications being used and the type of task or function being performed. For the project, a MacBook computer system is used with specific hardware configurations.

It's essential to have the appropriate hardware specifications for the smooth execution of the project. Table 3 provides detailed information about the system's CPU, GPU and other relevant hardware components and their respective functions.

Table 3

Hardware Configurations

Hardware	Configuration	Memory
8 – Core CPU with 4 efficiency cores and also 4 performance cores.	256GB SSD	8GB Memory
7 – Core GPU	Configurable to 512GB SSD, 1Tb, or 2TB SSD	Configurable to 16GB

Note. The hardware configurations are described above, where the cost for a 256GB SSD for 8 – Core CPU is 600$ and for 7 – Core GPU is 320$.

Software Requirements

Additionally to hardware requirements, software requirements as shown in Table 4 are also an important consideration for any study that involves technology. In summary, the study requires specific software applications, including Microsoft Word, Jupyter Notebook, and Python programming language. It's important to ensure that the required software versions are used and that compatibility is verified, as this can significantly impact the smooth execution of the project.

Table 4

Software Specifications

Software/Libraries	Version	Purposes
Python (Libraries including Pandas, Numpy, Matplotlib, Seaborn and scikit - learn)	3.9.13	Data Preprocessing, Cleaning, Analysis, visualizations, Machine Learning Model Building.
Tableau	2023.1	Visualizing and analyzing the data.

Software/Libraries	Version	Purposes
GitHub	3.8.0	Version control and to track changes to code over time.

Note. The software specifications are described above, where the versions used for Python, SQL, Tableau, and GitHub are 3.9.13, 8.0.

29, 2023.1, and 3.8.0 respectively.

Tools and Licenses

In addition to hardware and software requirements, the successful completion of the project also requires specific tools and licenses as shown in Table 5. It is crucial to ensure that all tools and licenses used are up-to-date and that any licensing requirements are met to avoid any legal or ethical issues.

Table 5

Tools Specifications

Tools	License	Purpose
Jupyter Notebook	Open Source	Widely used for data analysis, visualizations, and Machine Learning tasks
JIRA	Cloud Licenses	Project management and issue tracking capacity
Draw.io	Free	Web-based diagramming and visual representations, such as flowcharts, network diagrams.
GitHub	Free	Create a new repository and track changes to code over time, and manage issues and feature requests.
Microsoft Excel	Free	Data analysis and reporting

Note. The above table demonstrates the tools specifications, licenses and purposes for Jupyter Notebook, JIRA, Draw.io, AWS, GitHub and Microsoft Excel.

Specifications, Costs and Justification

Each hardware and software component as shown in Table 6 and Table 7 must meet or exceed the minimum specifications required to ensure the project's successful execution. The project costs will be based on the requirements and specifications of each component, including hardware, software, tools, and licenses.

Table 6

Hardware Specifications and Costs

Hardware	Duration	Cost	Justification
8 – Core CPU with 4 efficiency cores and 4 performance cores.	4 months	$500	Efficient parallel processing for handling complex machine learning algorithms
7 – Core GPU	4 months	$300	Accelerate computations and enhance model training

Note. The hardware configurations approximate cost for the duration of the project is mentioned in the table.

Table 7

Resource Specifications

Resources	Duration	Cost	Justification
JIRA	4 months	$0	Assign tasks, set target dates, and monitor performance.

Resources	Duration	Cost	Justification
Grammarly	3 months	$60	Language inaccuracies, spelling errors, and punctuation flaws, suggest alternatives for word selection and sentence organization.
Scribbr	1 year	$0	Precision, lucidity, and consistency of scholarly documents.

Note. The above table demonstrates the resource specifications for JIRA, Grammarly, and Scribbr where the costs are 0$, 60$ and 0$ respectively. The justifications are also mentioned.

Project Schedule

The project schedule is a critical element in project management, and it helps the team to plan and execute the project within the set timelines and budget.

Gantt Chart

The Gantt chart is a popular tool used to create a project schedule that represents the tasks, timeline, responsible team members, and the status of deliverables. For the project, a WBS Gantt chart tool is used to break down the project into smaller components and create a project schedule. The Gantt chart shows the start and end dates of each task, their duration, and dependencies between tasks. The tasks for each sprint are planned and prioritized based on their importance and dependencies on other tasks. For example, dataset gathering cannot be started until the defining project requirements phase is completed, and accuracy cannot be evaluated until modeling is completed. Responsible team members are assigned to each task and due dates set for each deliverable.

The Gantt chart includes dependencies between the stories, highlighting the order in which the tasks need to be completed. By following the Gantt chart, the project team can ensure that the project is completed on time and within budget. Each story has a set of tasks that need to be completed within the time frame of the sprint. Story points are used to estimate the effort

required for each task and tasks of the same difficulty level are distributed equally among team members to ensure a balanced workload. The tasks in each sprint were interdependent, meaning that the completion of one task was required before the next task could begin. These dependencies are mapped out using the Gantt chart by connecting the bars representing the tasks with arrows. This shows the critical path of the project and the team ensures that tasks are completed in the correct sequence.

Throughout the project, progress of each task is tracked and updated in the Gantt chart to reflect the status of deliverables. The Gantt chart helps identify potential issues and delays, adjust timelines and resources as needed, and ensure that the project remains on track and within budget. All team members equally participated in creating and assigning the tasks and preparing the Gantt chart (see Appendix B), which helped to ensure that everyone was on the same page regarding project timelines and deliverables.

The project is broken down into six epics, each consisting of a number of stories:

Business Understanding. It has four stories - defining problem statements, establishing project goals, evaluating project requirements and project planning; with few of the stories containing one or more tasks. This epic is handled in the first sprint. In this sprint, the team focused on defining the problem statement, establishing project goals, and identifying project requirements. This sprint is essential for laying the foundation for the rest of the project.

Data Understanding. The second sprint focuses on this epic, including exploring datasets, identifying relevant sources, and finalizing the dataset. This phase is crucial in ensuring that the project team has the right data to work with (see Appendix B).

Data Preparation. This is the goal for the third sprint. It includes cleaning the dataset, integrating data from various sources, and performing data warehousing. This phase helps to ensure that the data is properly prepared for modeling (see Appendix B).

Modeling. This epic is taken care of in the fourth sprint. It involves selecting and building models. The project team will conduct research to find the appropriate data models, finalize the models, and train and test the models (see Appendix B).

Evaluation. The fifth sprint focuses on this. This epic focuses on evaluating the models, comparing the results, and creating visualizations and dashboards to present the results. This phase helps the team to understand which model provides the most accurate results (see Appendix B).

Deployment. The sixth sprint focuses on this epic of the project, which contains generating a final report, and presenting the project. This phase ensures that the project is properly deployed and the results are communicated effectively (see Appendix B).

PERT Chart

A PERT chart, also known as a Project Evaluation and Review Technique chart, is a visual tool used to represent a project's schedule. Its purpose is to display the sequence of tasks involved in the project, the estimated duration for each job, and the interdependencies between them. The PERT chart typically provides more detail than the project's Work Breakdown Structure (WBS) and less detail than a Gantt chart. The level of detail in the PERT chart is chosen to provide a clear and concise overview of the project schedule.

For the project, a milestone-based PERT chart is created that accurately illustrates the dependencies for all tasks and activities essential for accurately representing the project schedule. The chart focuses on the sequence of tasks needed to complete the project, their

estimated durations, and the dependencies between them. The critical path for the project is identified as: Start - Understanding Business Objectives - Defining and understanding Data Requirements - EDA - Checking Data Quality - Transforming Data - Splitting and testing data - Model building - Analyzing model performance - Testing models - Analyzing errors and remodeling - Deploying model - Presenting final project - End

The critical path identified is the path with the most optimal and necessary tasks for project completion. Any delay in these tasks will affect the project completion. The number of optimistic days for each task were assigned based on previous experience. The remaining tasks Conducting Market Research, Conducting an Impact Analysis, Identifying stakeholders, Conducting feasibility study, Track and manage inventory, Exploring Additional Modeling Techniques and Implementing Advanced Visualization will not affect the coming together of the project and are additional steps hence they are not used for this project. There is a slack of 1 - 2 days in each of these activities and the pessimistic timeline is under 56 days which is the total time to complete the project.

The PERT chart created accurately illustrates the interrelationships between the tasks and activities, including finish-to-start and other dependencies. Team members responsible for each task or activity are consulted to confirm dependencies, identify any missing dependencies, and correct any incorrect relationships. In conclusion, the milestone-based PERT chart is a helpful tool for managing the project and ensuring its success. The chart provides a clear and concise overview of the project schedule, accurately illustrates task dependencies, and identifies the critical path for project completion. The Dependency list in the PERT Chart is shown in Table 8.

Table 8

Dependency list in PERT Chart

Activity	Description	Predecessor/Dependency
1	Project Start	-
2	Understood Business Objectives	-
3	Stakeholders Identified	-
4	Defined & Understood Data Requirements	2
5	Market Research done	2
6	EDA done	4
7	Successfully finished feasibility study	3
8	Checked Data Quality	6
9	Data Transformed	7,8
10	Split and tested data	9
11	Built the model	10
12	Analyzed Models Performance	11
13	Models tested	12
14	Additional modeling techniques explored	12
15	Finished analyzing errors and remodeling done	13
16	Model deployed	14,15
17	Final report submitted	16

Note. The above table demonstrates the activity list for the milestones mentioned in the PERT chart, with their Predecessors/Dependencies.

Figure 4 shows the PERT chart, while Figure 5 and Figure 6 give a close up view of the PERT chart.

Figure 4

PERT Chart

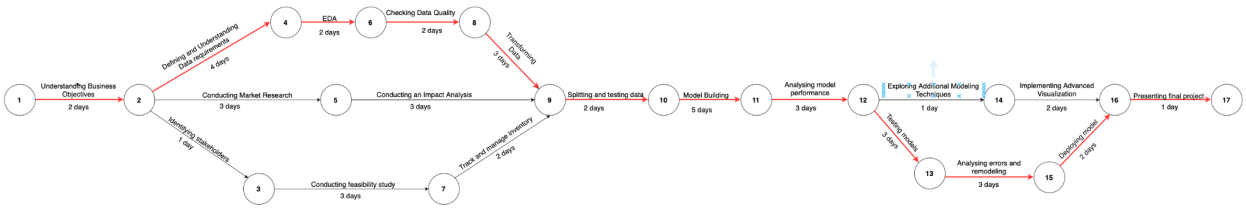


Figure 5

Close Up View of PERT Chart

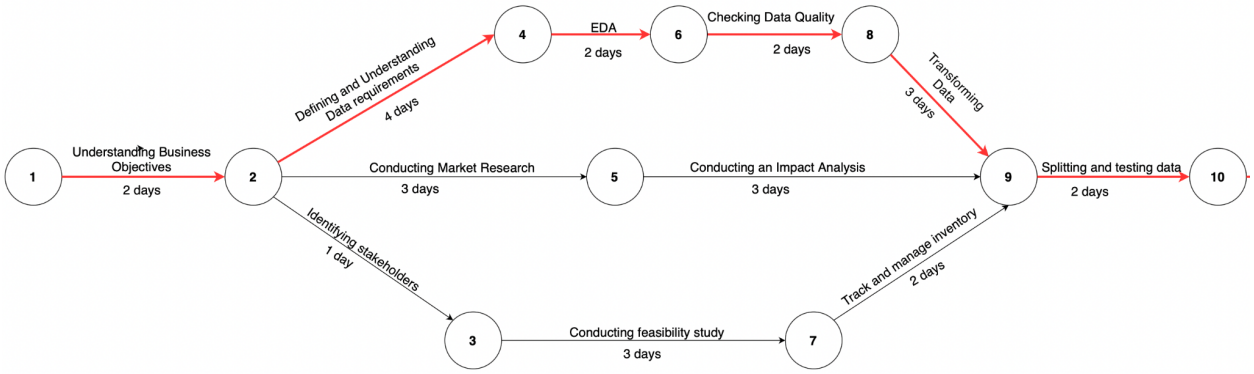
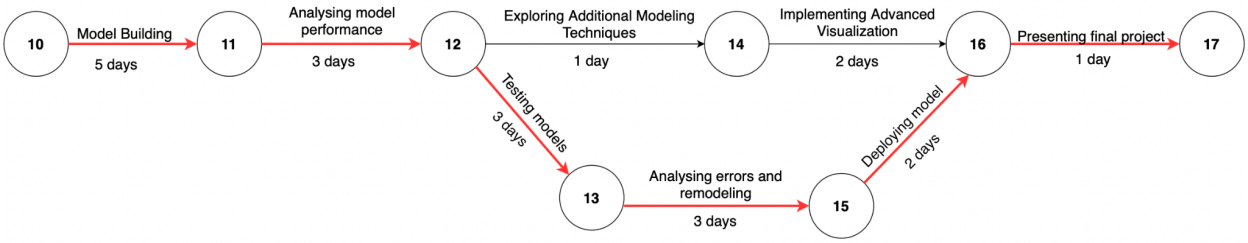


Figure 6

Close Up View of PERT Chart



Data Engineering

Data Process

To extract valuable insights and information, data processing entails the collection, modification, and analysis of the dataset. By identifying relevant data sources, the supply chain data has been collected from the Mendeley data website. Once the data has been obtained, the chosen raw data from the previously mentioned site is added to the stages required for data processing, at the input stage. The next stage involves cleaning the data using the Python Libraries. This includes eliminating duplicates, adding missing numbers, and making sure the data is formatted consistently. The preprocessed datasets are then utilized for additional analytics to gain beneficial details, trends and patterns about the supply chain process. Based on the gained insights from the data analysis, machine learning techniques are performed. The models that have been chosen are Support Vector Regression (SVR), K-NN, Random Forest and XGBoost models. The dataset is then divided into three datasets, of which 60% will be utilized for training and 20% for validating and 20% for testing using the historical data. The final step involves visualizing the data using graphs and charts that summarizes the conclusions.

Data Collection

Data Collection Requirements

'Order Dataset' and 'Product Dataset,' two sizable datasets from DataCo, this substantial amount of data emphasizes the necessity of a carefully planned data collection method. Figure 7 describes the data collection plan in detail. In summary, this dataset offers a valuable resource for analyzing and enhancing supply chain systems.

Figure 7

Data Collection Plan Table

Data Collection Plan

Project number: Project Group 3 | Project title: Improving Supply C | Project leader: Gouri Benni | Date: 03/21/2023

Description of the data collection

Altogether, the data has been collected from two datasets, Orders and Products. Both these datasets were obtained from the Mendeley data website. This dataset is used by the DataCo company. This data was documented between the years 2015 to 2018.

What will be done with the data once it has been collected?

		Identify how well the process is meeting **customer** needs		Analyze a **problem**, or identify the causes of variation	
	Yes	Obtain **exploratory view** of the process		Test a **hypothesis** about the process output	
		Evaluate the **measurement system**		Test a **hypothesis** about the effects of one or more inputs	
		Check the **stability** of the process (is it in control?)		**Control** a process input or monitor a process output	
		Conduct a **capability** study		**Other reason…**	

Key Variables - A summary of the chosen input variables (Y's) and/or output variables (X's)

		1	2	3	4	5	6
What?	**Variable title**	Delivery_Delay	Market	Type	Delivery_Status	Late_Delivery_Risk	Shipping_Mode
	Input (X) or output (Y) variable?	X	Y	Y	Y	Y	Y
	Unit of measurement	Days	-	-	-	1/0	-
	Data type	Attribute	Attribute	Attribute	Attribute	Attribute	Attribute
	Collection method	Automated	Automated	Automated	Automated	Automated	Automated
	If manual						
MSA	Gauge/instrument	N/A	N/A	N/A	N/A	N/A	N/A
	Location	N/A	N/A	N/A	N/A	N/A	N/A
	Gauge calibrated?	N/A	N/A	N/A	N/A	N/A	N/A
	Measurement system checked?	N/A	N/A	N/A	N/A	N/A	N/A
	Precision (R&R) adequate?	N/A	N/A	N/A	N/A	N/A	N/A
	Accuracy adequate?	N/A	N/A	N/A	N/A	N/A	N/A
orical	Historical data exist?	Yes	Yes	Yes	Yes	Yes	Yes
	Source of historical data	https://data.mendeley.com/datasets/8gx2fvg2k6/5#:~:text=Dataset%20of%20Supply%20Chain%2C%20which%20allows%20the%20use,Structured%20Data%20with%20Unstructured%20Data%20for%20knowledge%20generation					
	Historical data representative/reliable?	Yes	Yes	Yes	Yes	Yes	Yes
	Mean	-0.5658	N/A	N/A	N/A	0.5482	N/A
	Upper specification limit	2.0000	N/A	N/A	N/A	1.0000	N/A
	Lower specification limit	-4.0000	N/A	N/A	N/A	0.0000	N/A
	Standard deviation	1.4909	N/A	N/A	N/A	0.4976	N/A
	Target						
mplir	Minimum sample size (MSS)	N/A	N/A	N/A	N/A	N/A	N/A
	Sampling frequency	N/A	N/A	N/A	N/A	N/A	N/A
	Sub-grouping needed?	N/A	N/A	N/A	N/A	N/A	N/A
	Sub-group size	N/A	N/A	N/A	N/A	N/A	N/A
	Stratification needed? (time, shift)	N/A	N/A	N/A	N/A	N/A	N/A
Who?	**Data collector**	Deekshita P Savanur	Deekshita P Savanur	Deekshita P Savanur	Deekshita P Savanur	Deekshita P Savanur	Deekshita P Savanur
	Operational definition exist?	Yes	Yes	Yes	Yes	Yes	Yes
	Data collector trained?	Yes	Yes	Yes	Yes	Yes	Yes
	Resources available for data collector?	Yes	Yes	Yes	Yes	Yes	Yes
When?	**Start date**	13-Mar	13-Mar	13-Mar	13-Mar	13-Mar	13-Mar
	Due date	22-Mar	22-Mar	22-Mar	22-Mar	22-Mar	22-Mar
	Duration (in days)	9d	9d	9d	9d	9d	9d

Note. Data collection specifications added.

Dataset Samples

Figure 8 shows the Product table that comprises various data types, including integers, objects, and floats. Among the columns, Order Id and Product Price are represented as float values, while Product Status is represented as an integer value. Additionally, the dataset includes a categorical variable called Shipment Mode, which contains four categories: First Class, Same Day, Second Class, and Standard Class. Shipment Mode is encoded as an object data type as shown in Figure 9.

Figure 8

Sample of Product Table

	Order Id	Product Card Id	Product Category Id	Product Description	Product Image	Product Name	Product Price	Product Status	Shipping Mode
0	77202.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
1	75939.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
2	75938.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
3	75937.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class
4	75936.0	1360.0	73.0	NaN	http://images.acmesports.sports/Smart+watch	Smart watch	327.75	0	Standard Class

Note. Dataset of the product details.

Figure 9

Datatypes of the Columns in Product Table

```
Order Id              float64
Product Card Id       float64
Product Category Id   float64
Product Description   float64
Product Image          object
Product Name           object
Product Price         float64
Product Status          int64
Shipping Mode          object
dtype: object
```

Figure 10 shows the Order table that contains object and float data types. Some of the significant columns in the dataset include Type, Days for Shipping (real), Benefit per Order, Late_delivery_risk, Delivery Status, Order Status, and Order ID. Type is represented as an object

value, while Days for Shipping (real), Benefit per Order, Late_delivery_risk, and Order ID are represented as float values. Delivery Status and Order Status are also object values as shown in Figure 11.

Figure 10

Sample of the Order Table

	Type	Days for shipping (real)	Days for shipment (scheduled)	Benefit per order	Sales per customer	Delivery Status	Late_delivery_risk	Category Id	Category Name	Customer City	...	Order Item Profit Ratio	Order Item Quantity	Sales	Order
0	DEBIT	3.0	4.0	91.250000	314.640015	Advance shipping	0.0	73.0	Sporting Goods	Caguas	...	0.29	1.0	327.75	314.64
1	TRANSFER	5.0	4.0	-249.089996	311.359985	Late delivery	1.0	73.0	Sporting Goods	Caguas	...	-0.80	1.0	327.75	311.35
2	CASH	4.0	4.0	-247.779999	309.720001	Shipping on time	0.0	73.0	Sporting Goods	San Jose	...	-0.80	1.0	327.75	309.72
3	DEBIT	3.0	4.0	22.860001	304.809998	Advance shipping	0.0	73.0	Sporting Goods	Los Angeles	...	0.08	1.0	327.75	304.80
4	PAYMENT	2.0	4.0	134.210007	298.250000	Advance shipping	0.0	73.0	Sporting Goods	Caguas	...	0.45	1.0	327.75	298.25

Note. Dataset of the order details.

Figure 11

Datatypes of the Columns in Order Table

```
Type                            object
Days for shipping (real)       float64
Days for shipment (scheduled)  float64
Benefit per order              float64
Sales per customer             float64
Delivery Status                 object
Late_delivery_risk             float64
Category Id                    float64
Category Name                   object
Customer City                   object
Customer Country                object
Customer Email                  object
Customer Fname                  object
Customer Id                    float64
Customer Lname                  object
Customer Password               object
Customer Segment                object
Customer State                  object
Customer Street                 object
Customer Zipcode               float64
Department Id                  float64
Department Name                 object
Latitude                       float64
Longitude                      float64
Market                          object
Order City                      object
Order Country                   object
Order Customer Id              float64
order date (DateOrders)         object
Order Id                       float64
Order Item Cardprod Id         float64
Order Item Discount            float64
Order Item Discount Rate       float64
Order Item Id                  float64
Order Item Product Price       float64
Order Item Profit Ratio        float64
```

In the Order table, there are two features ‘Days for shipping (real)’ and ‘Days for shipment (scheduled)’ from which the target variable ‘Delivery delay’ is derived. Delivery delay is obtained by subtracting the value of ‘Days for shipment (real)’ from ‘Days of shipment (scheduled)’.

Delivery delay = Days for shipment (scheduled) - Days for shipping (real)

Data Exploration Plan

Exploratory data analysis (EDA) is a critical step in data analysis to understand the data structure, patterns, relationships, and distributions. In this regard, EDA is carried out on the raw dataset to gain insights into the data. It is a process that is iterative, and further EDA tools, including visualization of data and statistical analysis, may be used to extract more information from the data and guide subsequent data processing processes.

To assess the quality of the data, completeness and accuracy scores are calculated for each column in the dataset using a custom Python function as shown in Figure 12. Completeness is defined as the percentage of non-missing values in a column, while accuracy is defined as the percentage of non-null values in a column. The completeness and accuracy scores are stored in a new dataframe, and an overall quality score is calculated as the mean of the mean quality scores across all columns.

The results of the data quality assessment showed that the dataset has high completeness and accuracy scores across all columns, with an overall quality score of 0.99. This indicates that the dataset is well-structured, with few missing or null values, and is suitable for further analysis. However, this initial data exploration only provides a basic assessment of data quality, and further exploratory analysis is needed to gain a deeper understanding of the dataset and identify potential issues or outliers.

Figure 12

Completeness and Accuracy Scores of All the Data Types

Quality Scores:

	Completeness	Accuracy
Type	0.99959	0.99959
Days for shipping (real)	0.99959	0.99959
Days for shipment (scheduled)	0.99959	0.99959
Benefit per order	0.99959	0.99959
Sales per customer	0.99959	0.99959
Delivery Status	0.99959	0.99959
Late_delivery_risk	0.99959	0.99959
Category Id	0.99959	0.99959
Category Name	0.999485	0.999485
Customer City	0.999518	0.999518
Customer Country	0.999496	0.999496
Customer Email	0.999518	0.999518
Customer Fname	0.99948	0.99948
Customer Id	0.99948	0.99948
Customer Lname	0.999463	0.999463
Customer Password	0.99948	0.99948
Customer Segment	0.999452	0.999452
Customer State	0.999507	0.999507
Customer Street	0.99959	0.99959
Customer Zipcode	0.999574	0.999574
Department Id	0.99959	0.99959
Department Name	0.99959	0.99959
Latitude	0.999458	0.999458
Longitude	0.999513	0.999513
Market	0.99959	0.99959
Order City	0.99959	0.99959
Order Country	0.999496	0.999496
Order Customer Id	0.999496	0.999496
order date (DateOrders)	0.999469	0.999469
Order Id	0.999242	0.999242
Order Item Cardprod Id	0.999552	0.999552
Order Item Discount	0.999557	0.999557
Order Item Discount Rate	0.99959	0.99959
Order Item Id	0.99959	0.99959
Order Item Product Price	0.999546	0.999546
Order Item Profit Ratio	0.99959	0.99959
Order Item Quantity	0.99959	0.99959
Sales	0.999568	0.999568
Order Item Total	0.999563	0.999563
Order Profit Per Order	0.99959	0.99959
Order Region	0.999557	0.999557
Order State	0.999518	0.999518
Order Status	0.999518	0.999518
shipping date (DateOrders)	0.999923	0.999923

A bar chart represents the number of missing values in each column of the Order table. The dataset is read into a Pandas DataFrame, and the isnull() function is used to count the number of null values in each column. A horizontal bar chart is then created to visualize the results, where each bar represents the number of null values in a particular column.

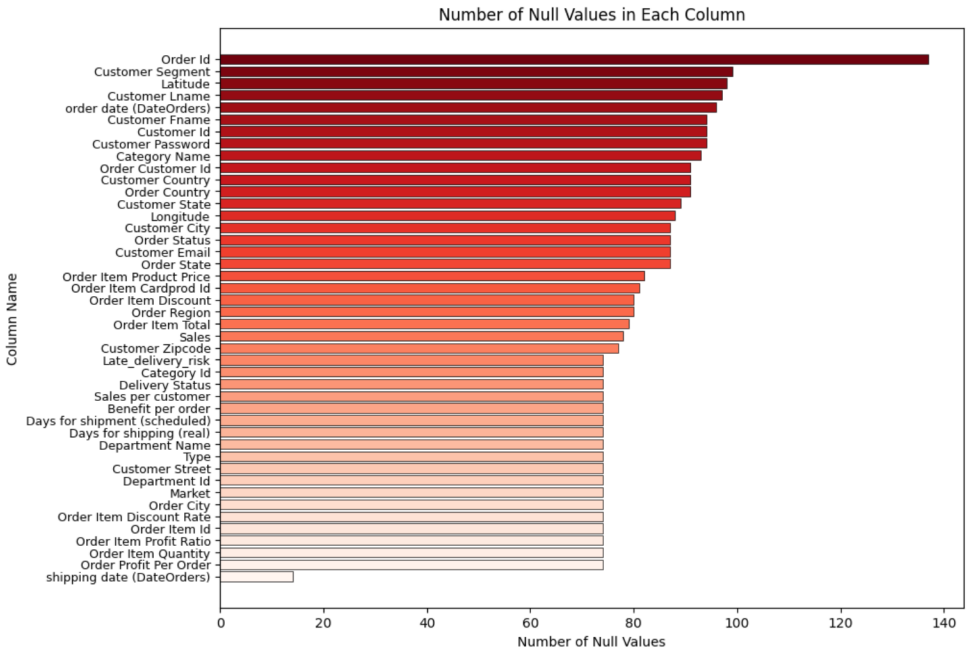
The purpose of this data exploration task is to identify columns in the dataset that have a large number of missing values. The horizontal bar chart gives an overview of the columns containing the most missing values, and subsequently helps decide whether to drop these

columns, impute missing values, or take other data cleaning measures. Additionally, this task helps assess the overall quality of the dataset, as missing values can indicate problems with data collection or entry. For example, missing values in the Order Id could indicate equipment or server malfunction while generating a number and so on.

Inferences from the chart can provide insights into the overall quality of the dataset. Each bar's height represents the number of missing values in the corresponding column. If a column has a relatively large number of missing values compared to the other columns as shown in Figure 13, it may indicate that this column is not useful for analysis and should be dropped from the dataset. Conversely, if a column has very few missing values, it may be an important feature that should be included in the analysis. Furthermore, if multiple columns have the same number of missing values, it may indicate that there is a problem with the data collection or entry process.

Figure 13

Null Value Count of All the Features



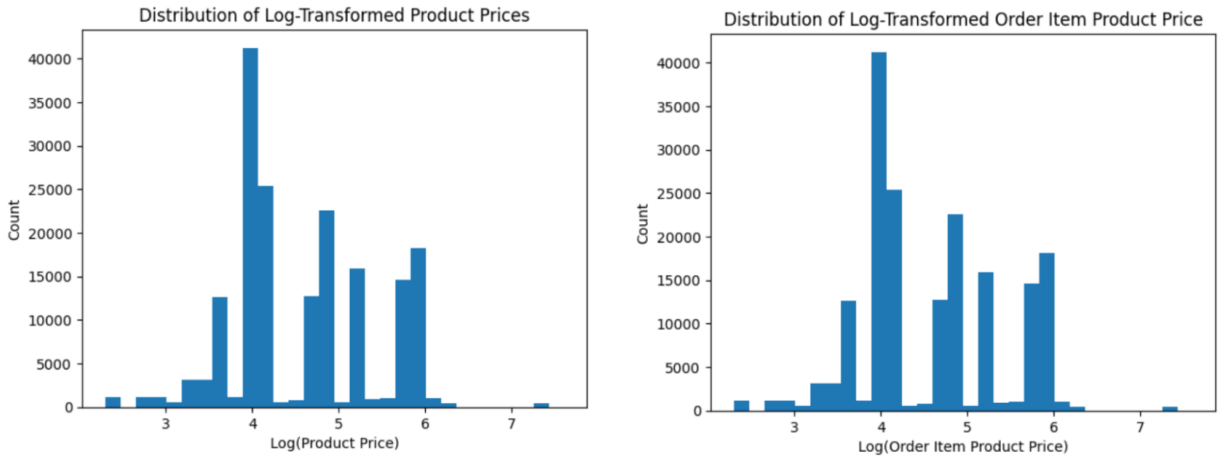
Log transformation is performed on the columns 'Product Price' and 'Order Item Product Price' of the Product and Order tables respectively. The transformed data is then plotted as histograms to visualize the distribution of the transformed values as shown in Figure 14.

The histograms generated by both code snippets show the same distribution of data, indicating that the 'Product Price' and 'Order Item Product Price' columns are identical in the two dataframes. This suggests that one of the columns is a duplicate and can be dropped to reduce redundancy and simplify the dataset.

It is important to note that such inconsistencies in data quality can arise due to several reasons, including data entry errors or data integration from multiple sources. Therefore, it is crucial to perform thorough data exploration and quality checks to ensure the accuracy and consistency of data in any research or analysis.

Figure 14

The Distribution of Log-Transformed Product Prices and Order Item Product Prices



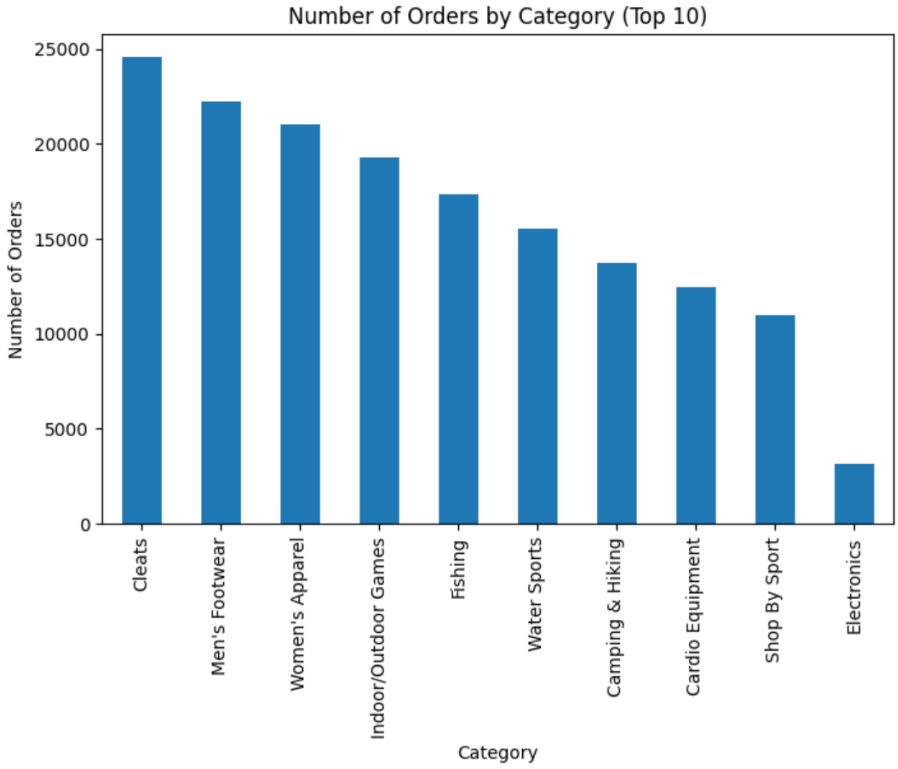
One of the tasks performed is to visualize the distribution of orders by category. For this purpose, a bar chart is created using the 'Category Name' column of the dataset. The bar chart shows the number of orders for each category in the dataset. As can be observed from the chart,

the category with the highest number of orders is 'Cleats', with approximately 24000 orders. The second-highest category is 'Men's Footwear', with a little over 22000 orders. 'Golf Bags & Carts' has the lowest number of orders.

The bar chart as shown in Figure 15 provides a clear overview of the distribution of orders across different categories. It can be a useful tool for identifying the popular and less popular categories. In this case, the highest number of orders is for 'Cleats', which could indicate that it is the most popular category among customers.

Figure 15

Order Count of All the Categories



Pie chart is used to show the proportion of orders by customer segment. The 'Customer Segment' column in the dataset is used to group the orders by customer segments. Each segment is represented by a slice of the pie chart, and the percentage of orders in each segment is shown

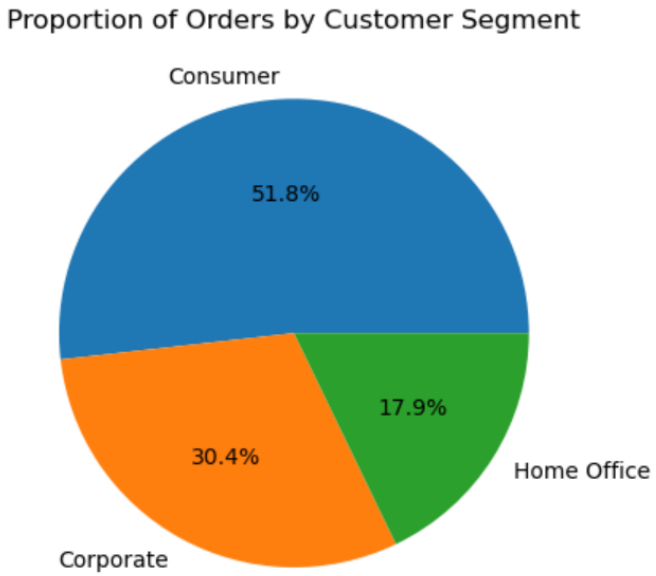
on the chart using the autopct parameter, which formats the percentage values with one decimal point.

This visualization as shown in Figure 16 provides insights into the customer segments that contribute the most to the orders in the dataset. By analyzing the chart, it can be insinuated that the majority of the orders come from the "Consumer" segment, which accounts for 51.8% of all orders. The "Corporate" segment is the second largest with 30.4% of the orders, followed by the "Home Office" segment with 17.9%.

This analysis can help businesses to understand the customer segments that contribute the most to their revenue and focus on targeting those segments for better sales performance. For example, the analysis may lead to a marketing strategy to promote products that are popular among the "Consumer" segment, which has the largest contribution to the orders.

Figure 16

Pie Chart Representation of the Order Proportion by Customer Segment

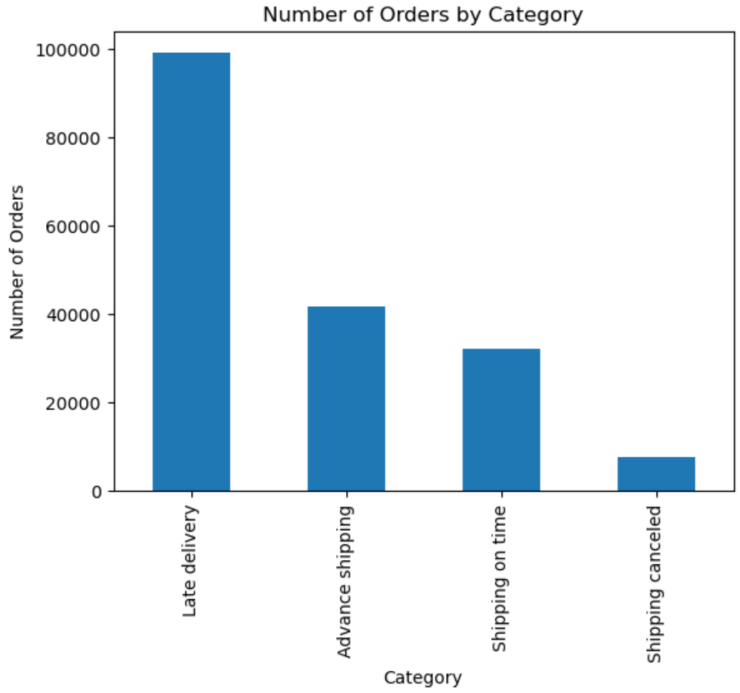


A bar chart is used to visualize the number of orders by delivery status. The x-axis in Figure 17 represents the different delivery status categories, while the y-axis shows the number of orders for each category. This visualization helps understand the distribution of delivery status categories in the dataset. By observing the chart, which delivery status categories have the highest and lowest number of orders can be inferred. It can also help detect any potential outliers or imbalances in the dataset.

Additionally, this chart can help explain if there is any relationship between the delivery status and the other variables in the dataset, such as the customer segment, category name, or sales. This visualization provides valuable insights into the distribution of delivery status in the dataset, which can help inform further analysis and decision-making.

Figure 17

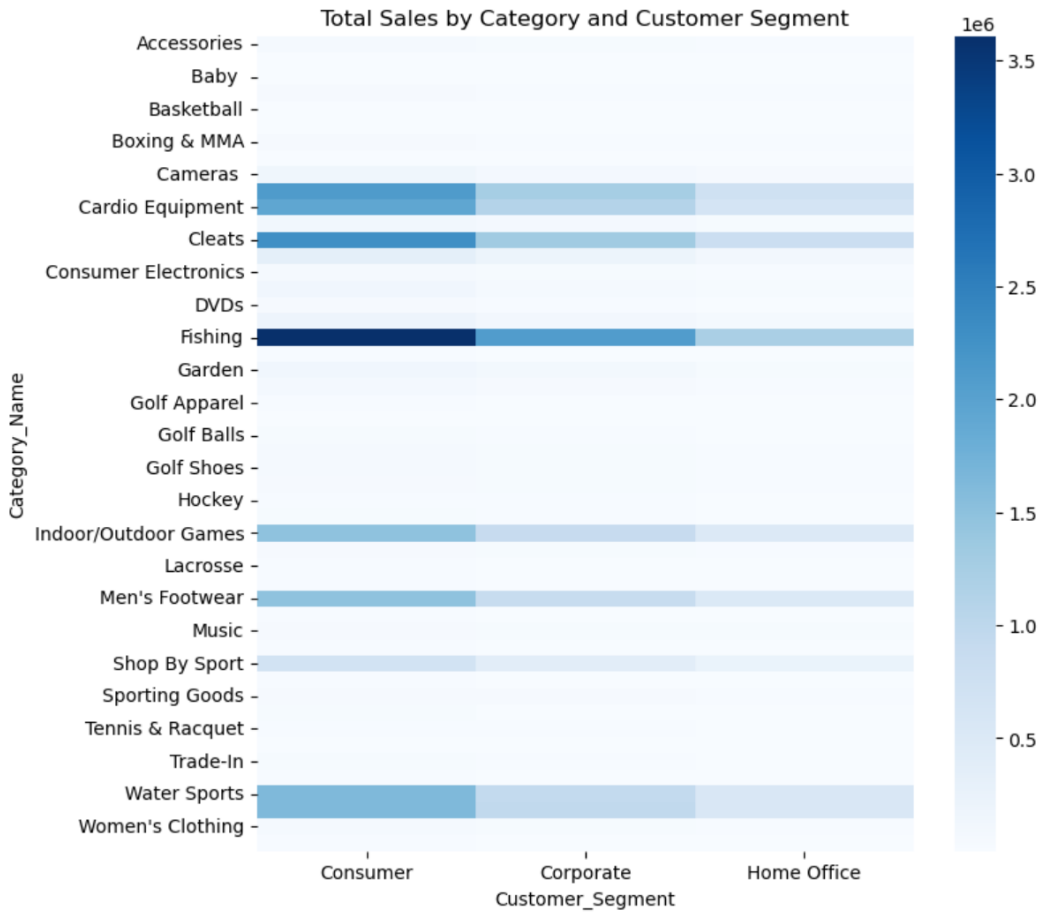
Bar Chart Representation of the Delivery Status and Number of Orders



A heatmap visualization as shown in Figure 18 is used to show the total sales by category and customer segment. It provides a visual representation of the total sales across different categories and customer segments. The color intensity of each cell indicates the magnitude of sales for that particular combination of category and customer segment. This visualization can be useful in identifying which categories and customer segments are the most profitable for the business. It can also help to identify potential areas for improvement, such as categories or customer segments with low sales that may require further attention.

Figure 18

Heatmap Representation of Sales by Category and Customer Segment



A bar plot is used to show the number of orders by customer segment and delivery status. The plot has two categorical variables on the x-axis: Customer Segment and Delivery Status. The number of orders is displayed on the y-axis. The plot is divided into multiple subplots based on the value of the 'Type' variable, which has four possible values: cash, debit, payment, and transfer. The shipping mode variable, which has four possible values: first class, same day, second class, and standard class, is represented by the color of the bars in each subplot.

The plot as shown in Figure 19 shows the number of orders for each customer segment and delivery status, which can be used to identify any patterns or trends in the data. The plot suggests that the standard class shipping mode is the most popular among all customer segments, followed by the second class mode. Additionally, it shows that the number of orders decreases from the debit type to the transfer type, indicating that customers tend to use debit cards more frequently than other payment methods.

Figure 19

Bar Plot Representation of Orders Number by Customer Segment and Delivery Status

Data Pre-processing

The purpose of machine learning models is to identify patterns and relationships in data, and if the data used to train these models contains inaccuracies, errors, or inconsistencies, the accuracy and reliability of the model's predictions will be affected. By cleaning the data before training the model, the reliability and accuracy of the model's predictions will be improved. Data cleansing is a crucial process which involves identifying and correcting errors, inconsistencies, and inaccuracies in a dataset ensuring data accuracy, reliability and consistency which ultimately enhances the quality of insights and conclusions derived from the data. The product dataset contains details about the products being sold, such as the product ID, name, category, price, and quantity available. The order dataset, on the other hand, contains details about the orders placed by customers, such as the order ID, customer ID, product ID, quantity ordered, and order date.

The two datasets are linked through the order ID, which presents as a foreign key in the order dataset, linking each order to the corresponding product being purchased. This relational structure allows for more efficient and organized data management, as it avoids redundancy and simplifies data updates and maintenance.

Handling of Incomplete and Missing Data

Incomplete and missing data are the situations in which some of the data values are either not recorded or are only partially recorded in a dataset. Incomplete data happens when only some of the variables are measured or recorded for some observations, while missing data occurs when the values of a variable are not recorded or are unknown. The absence of data can lead to biased or inaccurate analysis if not handled appropriately.

Figure 20 shows that the product dataset has a high number of missing values, as shown by the summary of missing values. The "Product Description" column, in particular, has a large

number of missing values, with 180656 observations missing. This column is not crucial to the analysis or model, so it is removed from the dataset.

The product dataset has a relatively small number of missing values in most of the columns. As the dataset is large and the number of missing values is relatively small, the rows with missing values will be deleted.

Figure 20

Missing Values in the Product Dataset

```
Order Id                          137
Product Card Id                    11
Product Category Id                26
Product Description            180656
Product Image                      50
Product Name                       56
Product Price                      29
Product Status                      0
Shipping Mode                      17
Product Price Winsorized            0
Product Price Moving Average       47
dtype: int64
```

In Figure 21 the order dataset has a significant number of missing values across many columns. To address these missing values, all the rows containing missing values are deleted. Removing these rows will enable the work with a complete and usable dataset for analysis. These consequences may include a reduction in sample size and an impact on the statistical properties of the remaining data. Therefore, before deleting all the rows with missing values, it is essential to carefully evaluate the specific characteristics of the dataset and the research question. It is crucial to consider the potential impact of removing these rows and to weigh it against the benefits of retaining them.

The "Order Zipcode" column has a very large number of missing values, 155816, which is a significant portion of the total number of rows in the dataset. As a result, dropping this column from the dataset is a logical choice, as it would reduce noise and simplify the dataset

Figure 21

Missing Values in Order Dataset

```
Type                             74
Days for shipping (real)         74
Days for shipment (scheduled)    74
Benefit per order                74
Sales per customer               74
Delivery Status                  74
Late_delivery_risk               74
Category Id                      74
Category Name                    93
Customer City                    87
Customer Country                 91
Customer Email                   87
Customer Fname                   94
Customer Id                      94
Customer Lname                   97
Customer Password                94
Customer Segment                 99
Customer State                   89
Customer Street                  74
Customer Zipcode                 77
Department Id                    74
Department Name                  74
Latitude                         98
Longitude                        88
Market                           74
Order City                       74
Order Country                    91
Order Customer Id                91
order date (DateOrders)          96
Order Id                        137
Order Item Cardprod Id           81
Order Item Discount              80
Order Item Discount Rate         74
Order Item Id                    74
Order Item Product Price         82
Order Item Profit Ratio          74
Order Item Quantity              74
Sales                            78
Order Item Total                 79
Order Profit Per Order           74
Order Region                     80
Order State                      87
Order Status                     87
Order Zipcode                155816
shipping date (DateOrders)       14
dtype: int64
```

After conducting the necessary cleaning operations on both the datasets, null values present, including columns with a significant number of null values, were eliminated. Figure 22 and Figure 23 implies that the data was thoroughly cleaned to remove missing values and irrelevant columns, resulting in an improved dataset that is now suitable for further analysis (See Appendix A).

Figure 22

Cleaned Order Data After Removing Missing Values and Irrelevant Columns

```
Type                           0
Days for shipping (real)       0
Days for shipment (scheduled)  0
Benefit per order              0
Sales per customer             0
Delivery Status                0
Late_delivery_risk             0
Category Id                    0
Category Name                  0
Customer City                  0
Customer Country               0
Customer Email                 0
Customer Fname                 0
Customer Id                    0
Customer Lname                 0
Customer Password              0
Customer Segment               0
Customer State                 0
Customer Street                0
Customer Zipcode               0
Department Id                  0
Department Name                0
Latitude                       0
Longitude                      0
Market                         0
Order City                     0
Order Country                  0
Order Customer Id              0
order date (DateOrders)        0
Order Id                       0
Order Item Cardprod Id         0
Order Item Discount            0
Order Item Discount Rate       0
Order Item Id                  0
Order Item Product Price       0
Order Item Profit Ratio        0
Order Item Quantity            0
Sales                          0
Order Item Total               0
Order Profit Per Order         0
Order Region                   0
Order State                    0
Order Status                   0
shipping date (DateOrders)     0
dtype: int64
```

Figure 23

Cleaned Product Data After Removing Missing Values and Irrelevant Columns

```
Order Id              0
Product Card Id       0
Product Category Id   0
Product Image         0
Product Name          0
Product Price         0
Product Status        0
Shipping Mode         0
dtype: float64
```

Handling of Noisy Data

Noisy data contains errors, outliers, or random fluctuations which are not representative of the true underlying data distribution. The presence of noisy data can negatively impact the performance of predictive models, which is why handling it is an important task in machine learning.

Noisy data can affect the performance of machine learning models in several ways. Firstly, it can lead to overfitting, where the model learns the noise in the data rather than the underlying pattern, resulting in poor performance on new data. Secondly, noisy data can increase the variance of the model, making it more sensitive to changes in the training data. Finally, noisy data can also reduce the interpretability of the model, making it more difficult to understand the relationships between the input variables and the output.

After performing a comprehensive exploratory data analysis (EDA), it was discovered that five features, spanning across two datasets, exhibit outliers. Figure 24 displays outliers identified in the 'Product Price' column of the Product dataset and Figure 25 shows the 'Sales per customer', 'Order Item Product Price', 'Sales', and 'Order Item Total' columns of the Order dataset. These outliers represent data points that will significantly deviate from the majority of the dataset, and may significantly influence statistical analysis and machine learning models (See Appendix A).

Figure 24

Box Plot Representation of the Presence of Noisy Data

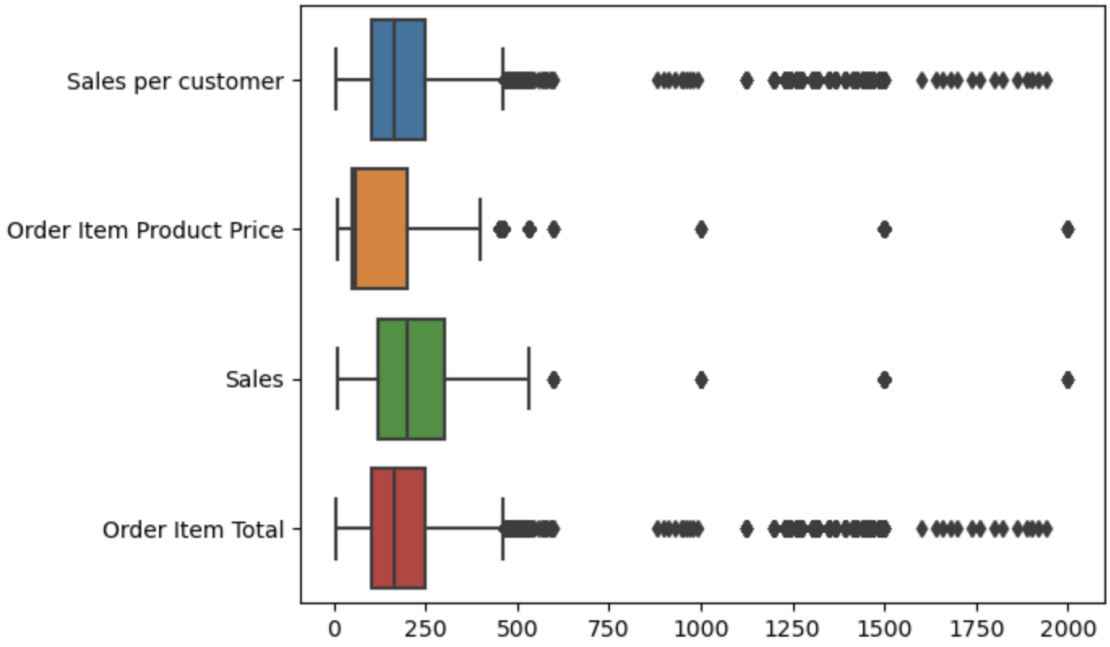
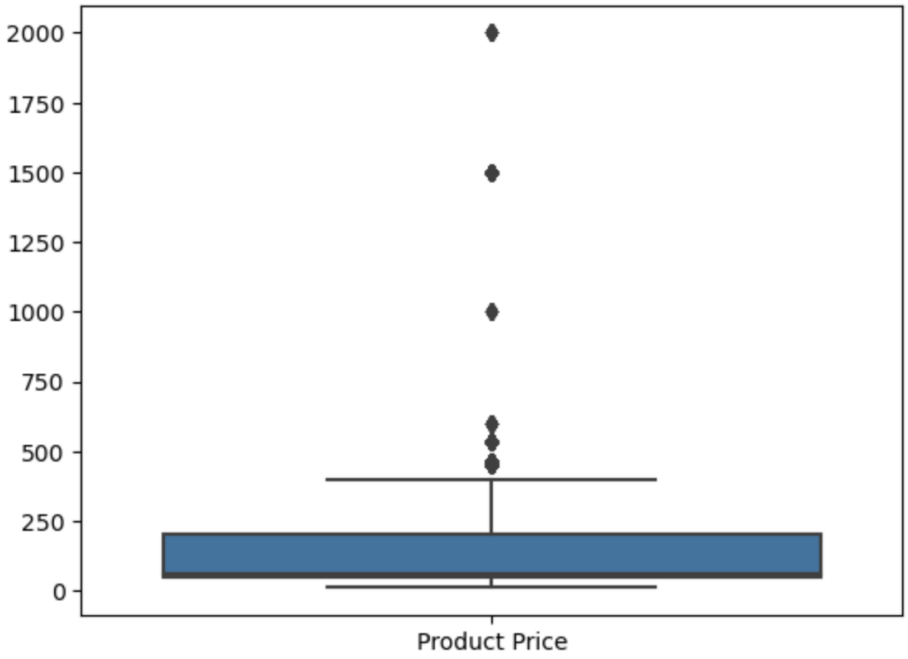


Figure 25

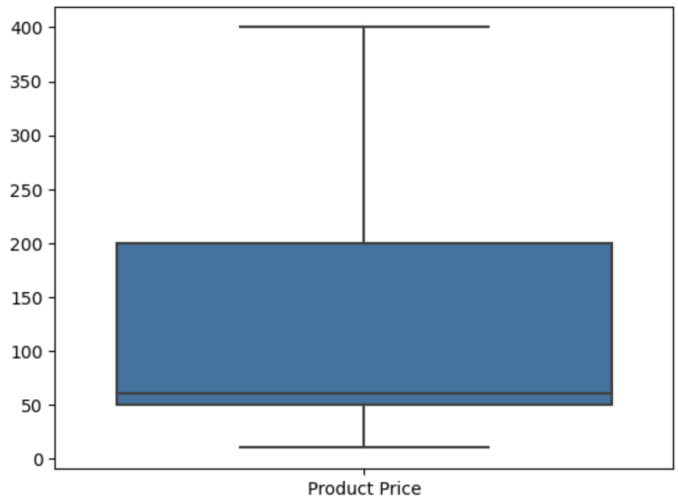
Box Plot Representation of Presence of Outliers in Product Price Value



The outliers in each column were identified and removed using the 1.5*IQR rule, and this method identifies outliers as data points that are greater than 1.5 times the interquartile range (IQR) away from the first or third quartile. After identifying and removing the outliers from each column, the data was reanalyzed to ensure that the results accurately reflected the underlying trends and patterns in the dataset as shown in Figure 26. Removing outliers will improve the accuracy of statistical analyses and the data is more representative of the underlying trends and patterns, leading to more accurate and reliable analyses.

Figure 26

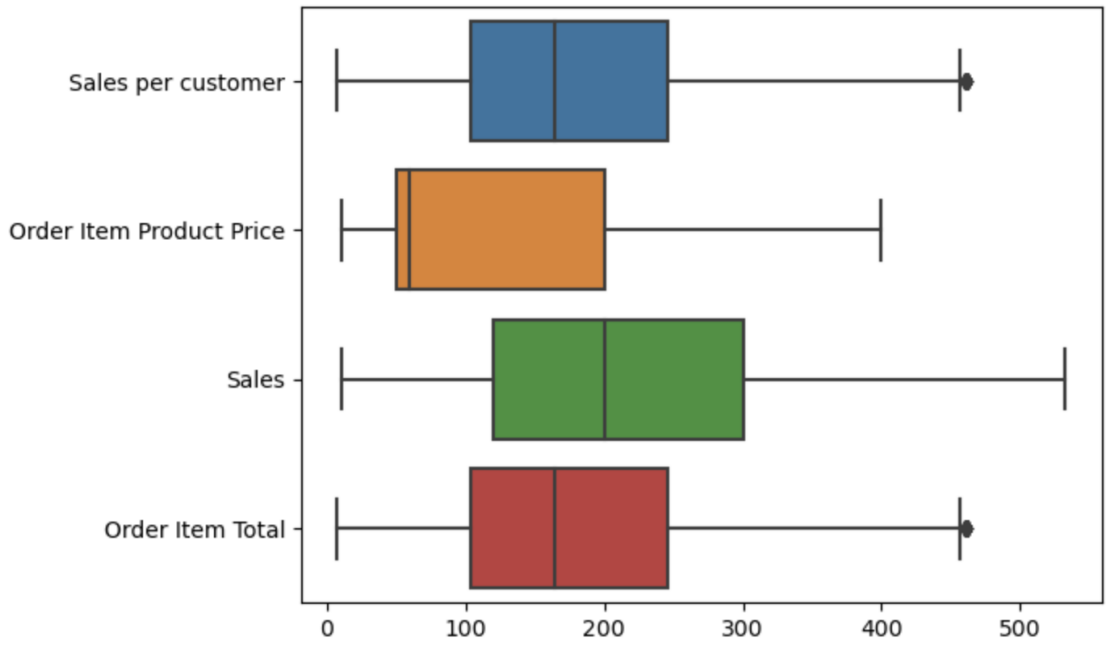
Box Plot Representation of Product Price Value After Removal of Outliers



In figure 27, there are data points just outside the maximum of the boxplot in the "Sales per Customer" and "Order Item Total" columns. These data points will be included in the analysis, to gain a complete understanding of the trends and patterns in the data and since the data points are still within the IQR range for outliers and just outside the maximum of the boxplot, they are likely to be valid data points and will provide more important insights into the data. The boxplot provides only a rough estimate of the spread of the data, and data points just outside the whiskers are valid and meaningful data points.

Figure 27

Box Plot Representation of the Features



Handling of Inconsistent Data

Inconsistent data refers to data that contains conflicting or contradictory values across different records or attributes, that can occur due to various reasons such as errors in data entry, differences in data formatting, or changes in the data over time. Handling inconsistent data is essential in data cleaning as it can adversely affect the accuracy and reliability of analyses or models.

In Figure 28, 'shipping date' and 'order date' - it contains information about the dates of a set of orders. Each row represents a unique order and displays the corresponding shipping and order dates. The format of date is 'MM/DD/YYYY HH:MM,' where HH:MM denotes the time of the day in hours and minutes. Since the time information is not relevant, it is essential to remove it and keep only the date information. The date information can be extracted by retaining only the date part of the timestamp and eliminating the time information.

The extracted date information is used for various analyses and modeling purposes, such as calculating the delivery time, understanding the seasonal trends in the order and shipping data, identifying the peak demand periods, and more.

Order Id contains floating-point numbers as identifiers which will create issues in sorting, indexing, and matching operations due to the imprecise nature of floating-point values. Machine learning models can also be affected by the use of floating-point numbers as identifiers. This is because these models rely heavily on mathematical computations that involve numeric values.

Float values are converted into integer values, as this ensures the accuracy and consistency of machine learning models. Additionally, some machine learning algorithms, such as decision trees, rely on comparing data values with each other to make decisions. These comparisons will also be impacted by the use of floating-point values since they may not always be precisely equal or ordered correctly.

Figure 28

Shipping Date, Order Date and Order Id in the Dataset

shipping date (DateOrders)	order date (DateOrders)	Order Id
2/3/2018 22:56	1/31/2018 22:56	77202.0
1/18/2018 12:27	1/13/2018 12:27	75939.0
1/17/2018 12:06	1/13/2018 12:06	75938.0
1/16/2018 11:45	1/13/2018 11:45	75937.0
1/15/2018 11:24	1/13/2018 11:24	75936.0
1/19/2018 11:03	1/13/2018 11:03	75935.0
1/15/2018 10:42	1/13/2018 10:42	75934.0
1/15/2018 10:21	1/13/2018 10:21	75933.0
1/16/2018 10:00	1/13/2018 10:00	75932.0
1/15/2018 9:39	1/13/2018 9:39	75931.0
1/19/2018 9:18	1/13/2018 9:18	75930.0
1/18/2018 8:57	1/13/2018 8:57	75929.0
1/17/2018 8:36	1/13/2018 8:36	75928.0
1/15/2018 8:15	1/13/2018 8:15	75927.0
1/15/2018 7:54	1/13/2018 7:54	75926.0

Note. The above table consists of the Shipping date, Order date details and Order Id

Figure 29 shows the time component from the "Shipping and order date" column from the order dataset is converted to the column that contains only the date information without any accompanying time information. The date format is also changed to ‘yyyy-mm-dd’ format. This modification is useful for performing analysis or visualization tasks that will focus on daily or monthly trends, as it will enable a clearer focus on the date information alone.

The Order ID feature in the Order dataset is converted from a float data type to an integer data type. This conversion ensures that the "Order ID" column contains only whole number values, without any decimal places. It is beneficial when performing certain types of analyses or when using integer-specific functions or operations.

Figure 29

Modified Order Date and Shipping Date Columns

shipping date (DateOrders)	order date (DateOrders)	Order Id
2016-06-13	2016-06-11	36146
2016-08-29	2016-08-24	41234
2016-04-19	2016-04-13	32090
2016-05-26	2016-05-22	34773
2016-05-13	2016-05-08	33824
2016-04-07	2016-04-02	31364
2016-08-20	2016-08-14	40495
2016-04-07	2016-04-02	31364
2016-05-20	2016-05-18	34506
2016-05-08	2016-05-05	33607
2016-04-24	2016-04-21	32617
2016-07-03	2016-07-01	37496
2016-04-24	2016-04-21	32617
2016-05-13	2016-05-08	33824
2016-07-29	2016-07-27	39271

Note. The above table shows the modified Shipping date and Order date details.

After cleaning, merging datasets will help identify patterns and relationships between different aspects of the supply chain as shown in Figure 30 and Figure 31. For example, merging product data with order data can provide insights into which products are in high demand and

which suppliers are most reliable. The resulting merged data is written to a new file called cleaned_supply_chain.csv as shown in Figure 32 (see Appendix A).

Figure 30

Cleaned Product Table

	Order Id	Product Card Id	Product Category Id	Product Name	Product Price	Shipping Mode
0	77202	1360	73	Smart watch	327.75	Standard Class
1	75939	1360	73	Smart watch	327.75	Standard Class
2	75938	1360	73	Smart watch	327.75	Standard Class
3	75937	1360	73	Smart watch	327.75	Standard Class
4	75936	1360	73	Smart watch	327.75	Standard Class

Note. The above table shows a cleaned product details table

Figure 31

Cleaned Order Table

	Type	Days_for_shipping_(real)	Days_for_shipment_(scheduled)	Benefit_per_order	Sales_per_customer	Delivery_Status	Late_delivery_risk
0	DEBIT	3.0	4.0	91.250000	314.640015	Advance shipping	0.0
1	TRANSFER	5.0	4.0	-249.089996	311.359985	Late delivery	1.0
2	CASH	4.0	4.0	-247.779999	309.720001	Shipping on time	0.0
3	DEBIT	3.0	4.0	22.860001	304.809998	Advance shipping	0.0
4	PAYMENT	2.0	4.0	134.210007	298.250000	Advance shipping	0.0

Note. The above table shows a cleaned order details table

Figure 32

Merged Product And Order table

	Type	Delivery_Status	Customer_City	Customer_Country	Customer_Segment	Customer_State	Customer_Street	Department_Name	Market	Order_
0	DEBIT	Advance shipping	Caguas	Puerto Rico	Consumer	PR	5365 Noble Nectar Island	Fitness	Pacific Asia	E
1	TRANSFER	Late delivery	Caguas	Puerto Rico	Consumer	PR	2679 Rustic Loop	Fitness	Pacific Asia	Bi
2	CASH	Shipping on time	San Jose	EE. UU.	Consumer	CA	8510 Round Bear Gate	Fitness	Pacific Asia	Bi
3	DEBIT	Advance shipping	Los Angeles	EE. UU.	Home Office	CA	3200 Amber Bend	Fitness	Pacific Asia	Towr
4	PAYMENT	Advance shipping	Caguas	Puerto Rico	Corporate	PR	8671 Iron Anchor Corners	Fitness	Pacific Asia	Towr

Note. The above table shows the merged product and order details

Data Transformation

Following the successful integration of the dataset, the next step is to start transforming the data. Data transformation is a crucial step in the data analysis process that involves converting raw data into a more structured format for further analysis. This involves various techniques such as normalization, scaling, and encoding to preprocess the raw data into a format that is suitable for the chosen machine learning algorithm. Data transformation enables the machine learning model to better understand the underlying patterns in the data, thereby improving its predictive power. Moreover, appropriate data transformation also ensures that the model is not biased towards certain features, leading to more accurate and reliable predictions. One common transformation technique is data normalization, which is the process of scaling data to a common range.

In the project, the data normalization process is applied to the 'Sales' and 'Order_Profit_Per_Order' columns in the merged table. The original data is transformed to a normalized form with range for Sales being [0,1] and Order_Profit_Per_Order [-1,1] in new columns Sales_Normalized and Order_Profit_Per_Order_Normalized.

The normalization of the Sales data to the [0,1] range is a common technique used to remove the impact of the absolute scale of the data. This transformation is useful when comparing data across different time periods, different countries, or different industries, where the absolute values of the Sales data may vary significantly. Normalizing the data to the same range allows comparison of the relative changes in Sales between different periods, countries or industries.

Additionally, the normalization of the Order_Profit_Per_Order data to the [-1,1] range is a different technique that is useful for data that has negative values. The normalization technique used here is known as Min-Max normalization, where the data is shifted and scaled to a common range. In this case, the data is shifted to be centered at zero (since the mean is negative) and then scaled to [-1,1] range. This transformation ensures that the negative values are retained but the data is rescaled to a common range.

Data normalization is an important process in data transformation that allows for easier comparison and analysis of data. By transforming the Sales and Order_Profit_Per_Order data to a common range, it gets easier to compare and analyze these variables in the merged table. Figure 33 shows the two columns Sales and Order_Profit_Per_Order as well as the normalized columns Sales_Normalized and Order_Profit_Per_Order_Normalized.

Figure 33

Normalized Data

tio	Sales	Order_Item_Total	Order_Profit_Per_Order	Product_Price	order_date	shipping_date	Order_Profit_Per_Order_Normalized	Sales_Normalized
.29	327.75	314.640015	91.250000	327.75	1/31/2018	2/3/2018	0.683599	0.159678
.80	327.75	311.359985	-249.089996	327.75	1/13/2018	1/18/2018	0.552366	0.159678
.80	327.75	309.720001	-247.779999	327.75	1/13/2018	1/17/2018	0.552871	0.159678
.08	327.75	304.809998	22.860001	327.75	1/13/2018	1/16/2018	0.657229	0.159678
.45	327.75	298.250000	134.210007	327.75	1/13/2018	1/15/2018	0.700165	0.159678

Feature engineering is a specific type of data transformation that involves creating new features or modifying existing ones to improve the performance of machine learning models. In the project, feature engineering is used to create a new target variable called "Delivery_Delay". This variable represents the difference between the number of days it takes to ship an order as scheduled and the actual number of days it took for the order to be delivered. By creating this new variable, one can use machine learning techniques to predict which orders are most likely to experience delays in delivery, and take steps to optimize the supply chain to reduce or eliminate these delays. One-hot encoding is performed to transform categorical data into an algorithm (see Appendix A).

The need for feature engineering in this research is driven by the importance of supply chain efficiency in maintaining customer satisfaction and loyalty. In order to optimize the supply chain, it is critical to identify areas where delays are occurring and take steps to reduce or eliminate them. By creating the Delivery_Delay variable, specific areas of the supply chain can be identified that are causing delays, and develop targeted strategies to improve them. Additionally, the predictive model that uses the Delivery_Delay variable as the target variable can be used as shown in Figure 34 to proactively identify orders that are at risk of being delayed, and take preemptive measures to ensure timely delivery.

Figure 34

Derived Target Feature from Existing Features

	Days_for_shipping_(real)	Days_for_shipment_(scheduled)	Delivery_Delay
0	3	4	1
1	5	4	-1
2	4	4	0
3	3	4	1
4	2	4	2
5	6	4	-2
6	2	1	-1
7	2	1	-1
8	3	2	-1
9	2	1	-1

Regularization

Regularization is a widely used technique in machine learning and statistical modeling to prevent overfitting and improve the overall performance of a model. Two common types of regularization are lasso regression (which uses L1 regularization) and ridge regression (which uses L2 regularization).

Both Lasso and Ridge regression methods were applied to the data.

Alpha values were chosen on a logarithmic scale, ranging from -10 to 10 using the NumPy logspace function with 5 increments. Cross-validation across 5 - folds was then performed to identify the best alpha value, based on metrics such as the highest mean cross-validation score, the least Mean Squared Error (MSE), and the best Regression Score. This approach helps to prevent overfitting and ensures that the unseen data is well generalized by the model.

The optimal alpha value for regularization in the model as shown in Figure 35 was determined based on a careful evaluation of different alpha values. While there were other alpha values with better mean cross-validation scores, it was observed that their ridge regression scores were low and MSE was high, indicating a trade-off between prediction accuracy and model consistency. Therefore, the alpha value that achieved a better ridge regression score and a lower MSE was chosen as the best alpha. The goal was to provide a balance between prediction accuracy and model stability. Hence the alpha with better MSE and ridge regression and an acceptable mean cross-validation score was considered.

Figure 35

Best Alpha and Mean Cross-validation Score for L2 Regularization

```
Best Alpha: 1.7782794100389228
Best mean cross-validation score: 0.05560898286640673
```

Figure 36 shows the ridge regression score (77%) and the MSE (0.05) for L2 regularization, and the lasso regression score (73%) and MSE (0.07) for L1 regularization. Considering the MSE and the regression scores for both the regularizations, L2 regularization was considered better fit for this dataset. Another reason being, the dataset considered has multiple predictor values with high correlations, making ridge regression a better fit compared to lasso regression. One advantage of ridge regression is that it shrinks the predictor values towards zero but does not force them to exactly zero. This means that ridge regression can retain some small non-zero coefficients, which can be beneficial in not having zero values to important input features. Figure 37 shows the dataset after regression was applied.

Figure 36

Ridge Regression Score, Lasso Regression Score and Their Respective MSEs

```
Ridge Regression Score: 0.7793787058628511
MSE: 0.05560898286640674


Lasso Regression Score: 0.7374882620232005
MSE: 0.0758477784055179
```

Figure 37

Feature and Coefficient After L2 Regularization

Order_Profit_Per_Order	Product_Price	order_date	shipping_date	Sales_Normalized	Order_Profit_Per_Order_Normalized	Feature	Coefficient
87.180000	327.75	19	18	0.159678	0.682030	Department_Id	1.022269e-03
154.860001	327.75	19	18	0.159678	0.708127	Order_Customer_Id	-1.060465e-07
82.300003	327.75	19	38	0.159678	0.680148	Order_Id	1.096593e-07
22.370001	327.75	19	34	0.159678	0.657040	Order_Item_Cardprod_Id	-3.548816e-06
17.700001	327.75	19	30	0.159678	0.655239	Order_Item_Id	-3.728817e-08
90.279999	327.75	19	26	0.159678	0.683225	Product_Card_Id	-3.195127e-06

Principal Component Analysis

Principal Component Analysis (PCA) is a commonly used technique in data analysis that can effectively reduce the no. of dimensions in the dataset while retaining most of the original information. PCA accomplishes this by transforming the original features into a new set of uncorrelated variables, known as principal components. However, to achieve optimal performance, it is recommended to standardize the data by scaling each feature to have zero mean and unit variance before applying PCA.

In addition to PCA, data transformation is an essential step in preparing data for machine learning models. One common transformation involves converting categorical data into numerical data. Label encoding is a frequently used method for this, which assigns a unique integer value to each category in a categorical variable.

To apply label encoding, the selected features are transformed into a numerical format. The resulting values of selected features are represented below. By converting categorical data into numerical data, machine learning algorithms can effectively utilize this information for further analysis and modeling.

After transforming the data into matrix format, Principal Component Analysis (PCA) is applied to derive the principal components. In this particular case, the target feature selected is Delivery_Delay, while the selected transformed features consist of factors such as Shipping Mode, Payment Type, Delivery Status and Market. Figure 38 shows the Snapshot of Dataset Before Applying PCA. Figure 39 displays the transformed values of the selected features.

Moreover, the explained variance ratio is computed for the principal components. For this dataset, the first principal component accounts for 75.03% of the variance, followed by the second principal component with 10.6%. These results demonstrate that the principal components obtained through PCA are capable of explaining a significant proportion of the variation in the original dataset.

Figure 38

Snapshot of Dataset Before Applying PCA

	Type	Delivery_Status	Customer_City	Customer_Country	Customer_Segment	Customer_State	Customer_Street	I
0	1	0	66	1	0	36	3683	
1	3	1	66	1	0	36	1400	
2	0	3	452	0	0	5	6217	
3	1	0	285	0	2	5	1803	
4	2	0	66	1	1	36	6345	
...	...	...	...	...	...	...	...	
180514	0	3	59	0	2	31	285	
180515	1	1	26	0	1	5	5261	
180516	3	1	55	0	1	7	7208	
180517	2	0	66	1	0	36	1335	
180518	2	3	66	1	0	36	5001	

Figure 39

Snapshot of Dataset After Feature Selection

	Type	Delivery_Status	Late_delivery_risk	Market	Shipping_Mode	Order_Item_Discount_Rate
0	1	0	0	3	3	0.04
1	3	1	1	3	3	0.05
2	0	3	0	3	3	0.06
3	1	0	0	3	3	0.07
4	2	0	0	3	3	0.09
...	...	...	...	...	...	...
180514	0	3	0	3	3	0.00
180515	1	1	1	3	2	0.01
180516	3	1	1	3	3	0.02
180517	2	0	0	3	3	0.03
180518	2	3	0	3	3	0.04

In the project, PCA is performed using label encoded data to reduce the dimensionality of the data and identify the underlying structure. However, in order to use the PCA results for machine learning, the cleaned version of the data has been transformed to target encoded format. The target encoded data allowed us to better capture the underlying patterns and relationships in the data, which ultimately led to more accurate predictive models.

Target encoding replaces each category value with the average target value (in this case, the average delivery delay) for that category. This technique is particularly useful when the categorical variable has a large number of unique values or when the target variable is strongly correlated with the categorical variable.

In the project, target encoding is used to encode the descriptive features that have a categorical nature. The need for target encoding in this project is driven by the importance of accurately predicting demand and optimizing the supply chain to improve efficiency. By encoding the descriptive variables using target encoding, the relationship can better be captured

between the categorical variables and the target variable (delivery delay), and use this information to predict demand more accurately as shown in Figure 40. Additionally, accurate demand forecasting can help optimize the supply chain, ensuring that the right products are available in the right quantities at the right time, and reducing the risk of stockouts or excess inventory.

Figure 40

Application of Label Encoding on Selected Features Required for PCA

	Type	Delivery_Status	Market	Shipping_Mode	Order_Item_Discount_Rate	Order_Country	Late_delivery_risk
0	DEBIT	Advance shipping	Pacific Asia	Standard Class	0.04	Indonesia	0
1	TRANSFER	Late delivery	Pacific Asia	Standard Class	0.05	India	1
2	CASH	Shipping on time	Pacific Asia	Standard Class	0.06	India	0
3	DEBIT	Advance shipping	Pacific Asia	Standard Class	0.07	Australia	0
4	PAYMENT	Advance shipping	Pacific Asia	Standard Class	0.09	Australia	0

	Type	Delivery_Status	Market	Shipping_Mode	Order_Item_Discount_Rate	Order_Country	Late_delivery_risk
0	-0.557544	1.501851	-0.569365	0.004093	-0.559178	-0.580162	0.711584
1	-0.565062	-1.618184	-0.569365	0.004093	-0.583608	-0.615513	-1.618184
2	-0.547716	0.000000	-0.569365	0.004093	-0.564662	-0.615513	0.711584
3	-0.557544	1.501851	-0.569365	0.004093	-0.551999	-0.570201	0.711584
4	-0.588928	1.501851	-0.569365	0.004093	-0.561571	-0.570201	0.711584

Data Preparation

To create precise predictive models in data analysis and machine learning, the data must be divided into training, validation, and testing sets. This helps to reduce the risk of overfitting and permits a more thorough evaluation of the model's performance.

Train Dataset

The data used to train a machine learning model is called the training set, which is a subset of the total amount of data. It is used to train the model to spot relationships and patterns in the input features and to make precise predictions about brand-new, untainted data. Figure 35 and 36 show the training sample of input and target features

Validation Dataset

A subset of the available data called the validation set is utilized to assess how well a machine learning model performed during training. It is used to track the model's performance on different sets of data and to tweak the hyperparameters to increase generalizability. Figure 35 and 36 show the validation sample of input and target features

Test Dataset

After a machine learning model has been trained and optimized on the training and validation sets, it is used with the test dataset, a subset of the available data, to assess how well it performs. It serves to quantify the model's performance in the actual world and to judge how well the model generalizes to fresh, untested data. Figure 41 and 42 show the test sample of input and target features.

The cleaned and processed data is partitioned into training, validation, and testing sets for both input and target features as shown in Figure 43. Initially, the data is split into an 85% training set and a 15% testing set. Subsequently, the 85% training set is further divided into a 70% training set and a 15% validation set. The final data partitioning results in a split of 70% training data, 15% validation data, and 15% testing data.

Figure 41

Train, Validate and Test Set for Input Features Respectively

	Type	Delivery_Status	Late_delivery_risk	Market	Shipping_Mode	Order_Item_Discount_Rate
160444	-0.565062	0.000000	0.711584	-0.569365	0.004093	-0.583608
35450	-0.547716	-1.618184	-1.618184	-0.557836	0.004093	-0.534949
173340	-0.547716	0.000000	0.711584	-0.557836	-1.990828	-0.551999
142098	-0.557544	-1.618184	-1.618184	-0.569365	-1.990828	-0.583608
119017	-0.565062	-1.618184	-1.618184	-0.557836	-1.990828	-0.587097

	Type	Delivery_Status	Late_delivery_risk	Market	Shipping_Mode	Order_Item_Discount_Rate
99076	-0.557544	1.501851	0.711584	-0.560014	0.004093	-0.545174
102906	-0.588928	-1.618184	-1.618184	-0.570843	-1.990828	-0.583608
118038	-0.565062	0.000000	0.711584	-0.570843	-1.990828	-0.551999
144205	-0.557544	1.501851	0.711584	-0.570843	0.004093	-0.551301
176931	-0.557544	-1.618184	-1.618184	-0.568859	-0.478279	-0.539635

	Type	Delivery_Status	Late_delivery_risk	Market	Shipping_Mode	Order_Item_Discount_Rate
80120	-0.565062	-1.618184	-1.618184	-0.568859	0.004093	-0.563366
19670	-0.588928	-1.618184	-1.618184	-0.557836	-1.000000	-0.545174
114887	-0.565062	1.501851	0.711584	-0.557836	0.004093	-0.545174
120110	-0.565062	-1.618184	-1.618184	-0.560014	0.004093	-0.534949
56658	-0.557544	1.501851	0.711584	-0.557836	0.004093	-0.583209

Figure 42

Train, Validate and Test Set for Output Features Respectively

	Delivery_Delay
160444	0
35450	-1
173340	0
142098	-2
119017	-3

	Delivery_Delay
99076	2
102906	-2
118038	0
144205	2
176931	-1

	Delivery_Delay
80120	-1
19670	-1
114887	2
120110	-1
56658	2

Figure 43

Shape of the Train, Validate and Test Datasets for Both Input and Output Features

```
For Input Features
Training set shape: (108311, 6)
Validation set shape: (36104, 6)
Test set shape: (36104, 6)


For Target Feature
Training set shape: (108311, 1)
Validation set shape: (36104, 1)
Test set shape: (36104, 1)
```

Data Statistics

In table 9, the data preparation results summary is described with the statistics mentioned in Figure 44 for each of the stages. These are assessed based on the data set and gaining insights into various patterns and trends, which aids in the extraction of useful information from the collected data.

Table 9

Summary of the Data Preparation Results

Stage	Methods	Statistics on the Rows & Columns
Raw data		Product Dataset : 180657 x 9 Orders Dataset : 180657 x 45
Pre-processing data		Product Dataset : 180583 x 9 Orders Dataset : 180583 x 45
Data Transformation	PCA for Feature Extraction	180583 x 6
Data Preparation	For Training For Testing For Validation	Input : 108311 x 6 Input : 36104 x 6 Input : 36104 x 6

Figure 44

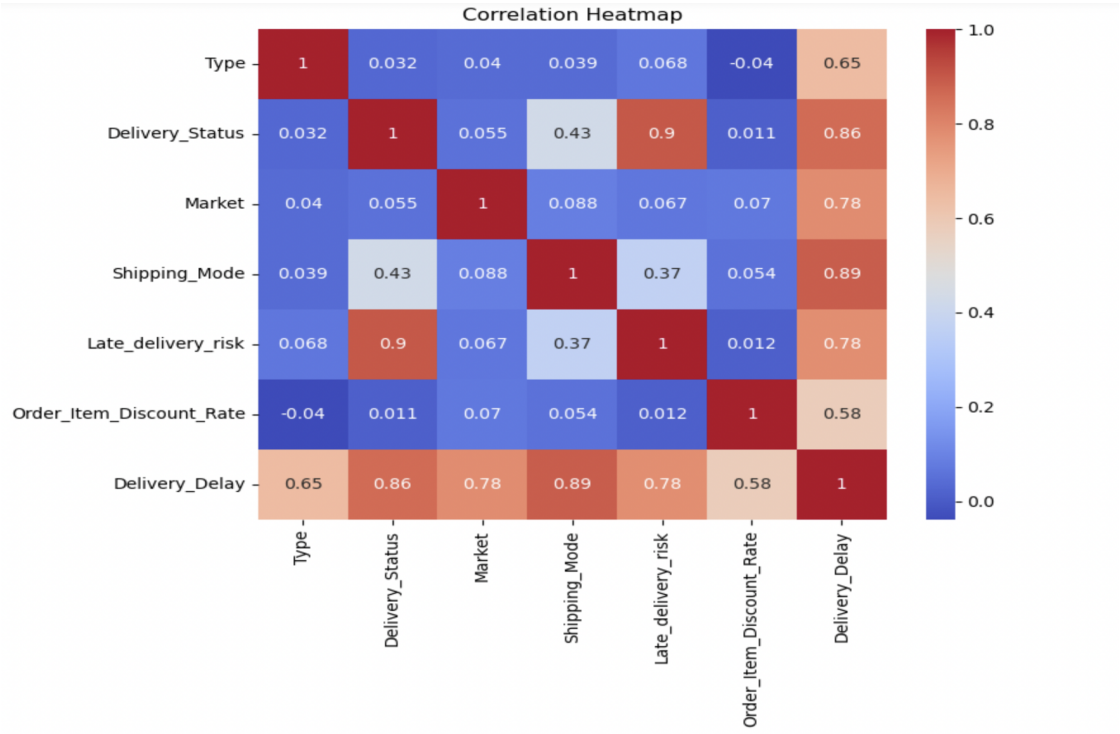
Statistical Information of the Features in the Dataset

	Type	Delivery_Status	Market	Shipping_Mode	Order_Item_Discount_Rate	Order_Country	Late_delivery_risk	Delivery_Delay
count	180519.000000	180519.000000	180519.000000	180519.000000	180519.000000	180519.000000	180519.000000	180519.000000
mean	-0.565807	-0.565807	-0.565807	-0.565807	-0.565807	-0.565783	-0.565807	-0.565807
std	0.013619	1.284273	0.005622	0.787070	0.016727	0.078787	1.159441	1.490966
min	-0.588928	-1.618184	-0.570843	-1.990828	-0.588693	-1.407708	-1.618184	-4.000000
25%	-0.565062	-1.618184	-0.570843	-1.000000	-0.583209	-0.595530	-1.618184	-1.000000
50%	-0.565062	-1.618184	-0.569365	0.004093	-0.563366	-0.575725	-1.618184	-1.000000
75%	-0.557544	0.000000	-0.557836	0.004093	-0.551999	-0.538356	0.711584	0.000000
max	-0.547716	1.501851	-0.557836	0.004093	-0.534949	0.028113	0.711584	2.000000

In Figure 45, the correlation heatmap is shown for the selected features namely, Type, Delivery Status, Market, Shipping Mode, Late Delivery Risk, Order Item Discount Rate, and Delivery Delay. In the heatmap, the correlations are shown between the features. Most of the features show that the selected features are strongly correlated with each other.

Figure 45

Correlation Heatmap of Input Features with Respect to the Target Variable



Data Analytics Results

An important aspect of the project is the visualization of the categorical features and their relationships. Using Neo4J graph, a visualization of the nodes and relationships between the categorical features for the dataset, with the main feature being Order_Country is created. Edges are added for the rest of the features including Type, Department_Name, Delivery_Status, Customer_Segment, Market, and Shipping_Mode. The resulting graph shows the feature values that had the most relationships with Order_Country in the center, while the values at the edges had fewer edges with the countries, indicating that there were fewer of those values present in the column as shown in Figure 46 and Figure 47. By analyzing the graph, one can draw several insights that align with the insights obtained from the summary statistics of the transformed and prepared data.

For instance, Late Delivery is closer to the center than Shipping on Time, and the mean of the target variable Delivery_Delay, calculated through feature engineering, was negative, indicating that more orders were not delivered on time. Similarly, Shipping Canceled is at the edges of the graph, suggesting that not many orders were canceled. Furthermore, Second Class is located at the center of the graph, followed by First Class, Standard Class, and Same Day Delivery at the edges. This could mean that customers are not willing to pay as much for Same Day Delivery, possibly due to extra charges.

Figure 46

Network Diagram for Categorical Features

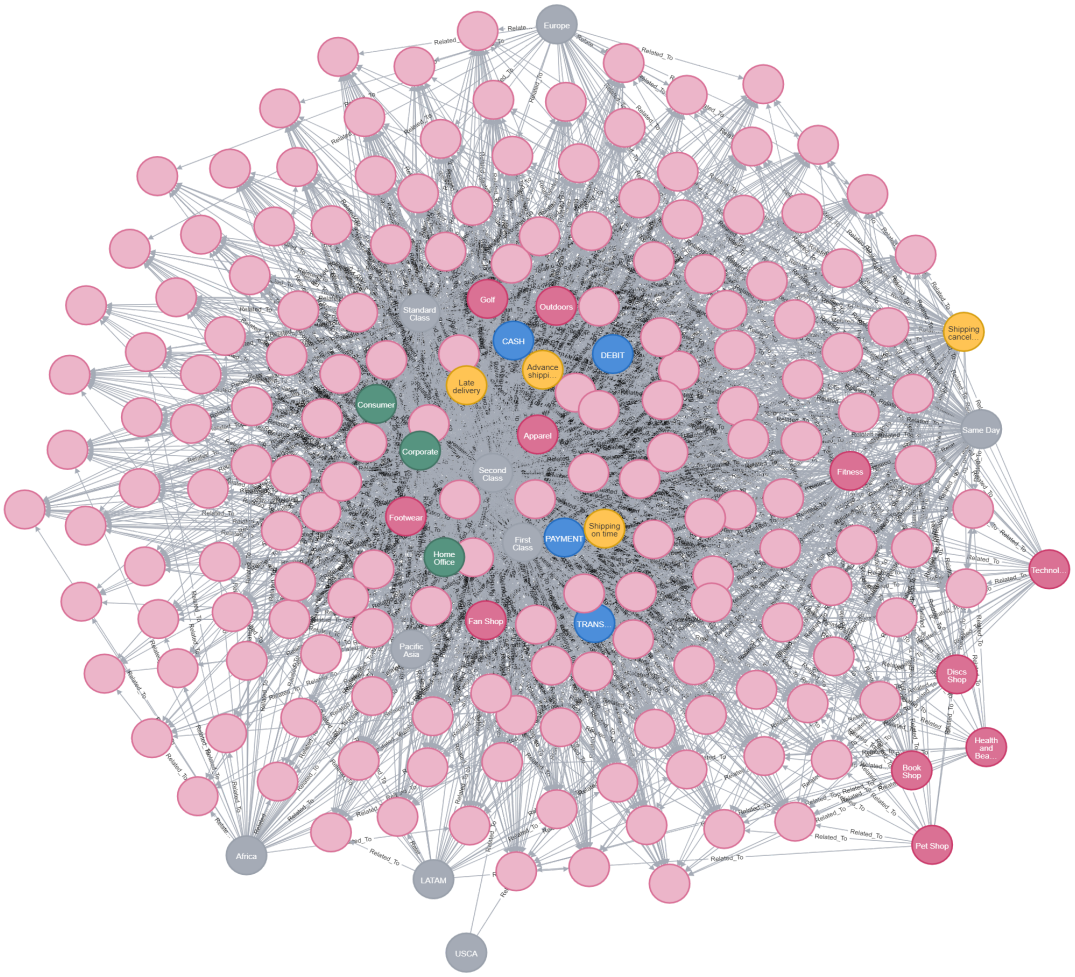
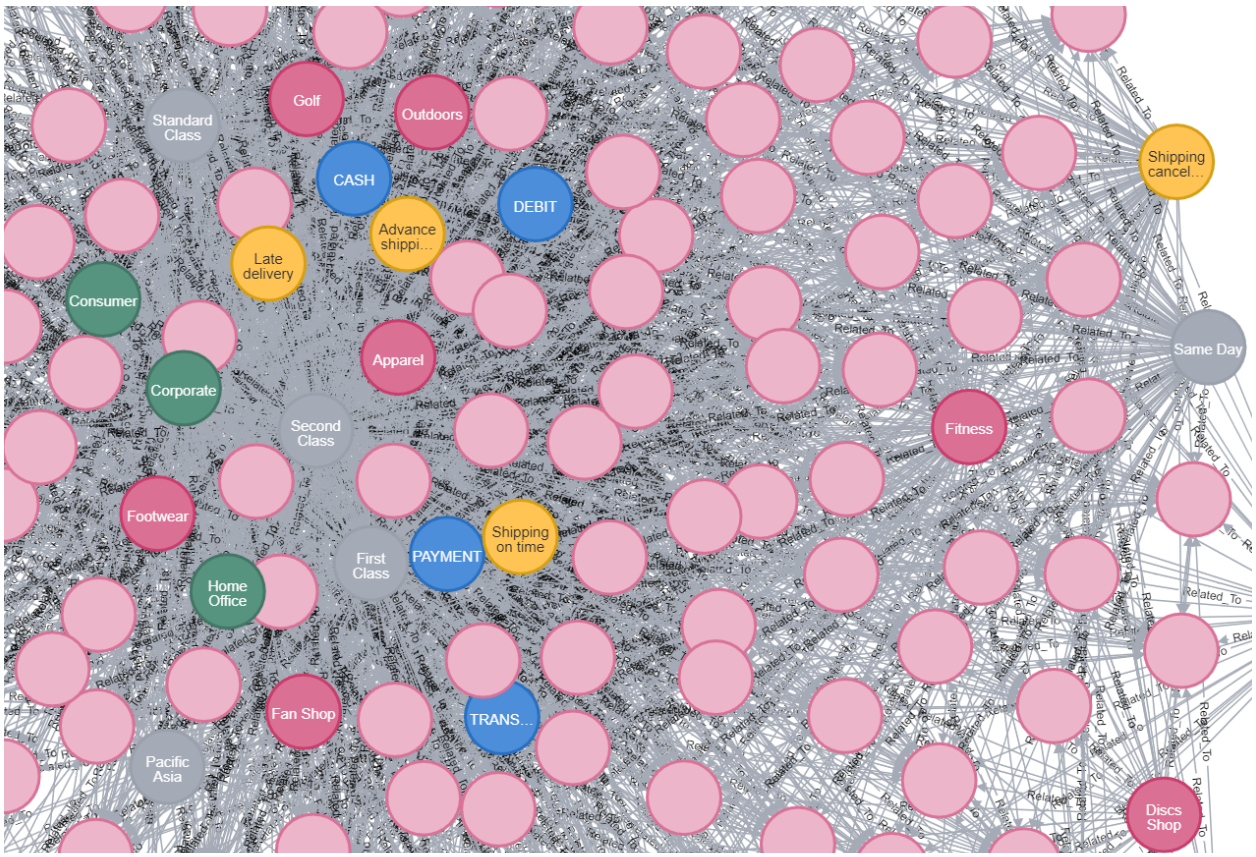


Figure 47

Closer Look of the Network Diagram for Categorical Features

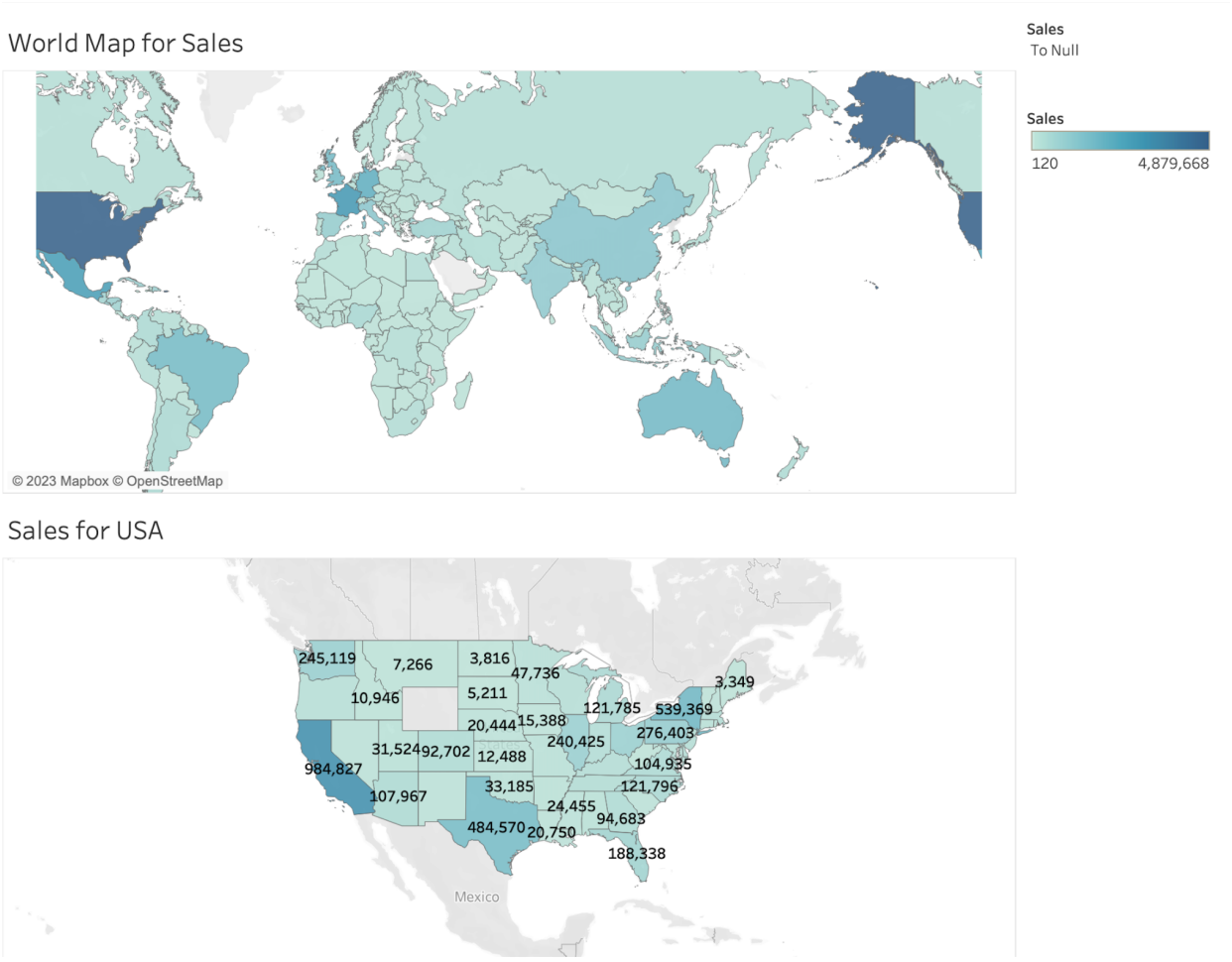


The Tableau dashboard's visualizations will provide a user-friendly and engaging way to explore Sales data. The geographical representation of the data as shown in Figure 48 will allow easy identification of regions with high and low Sales values, enabling data-driven decision-making. Dashboard consists of two visualizations: a world map visualization of Sales data and a specific view for Sales in the USA. The world map shows Sales data for different countries across the globe, with colors representing the highest and lowest levels of Sales. The visualization indicates that the USA has the highest Sales compared to other countries.

To gain a deeper understanding of the Sales data for the USA, a separate view is created specifically for Sales in the country. This view displays a map of the USA, the visualization reveals that California has the highest Sales among all states, while Wyoming has the lowest.

Figure 48

Dashboard for Sales Data



Model Development

In today's ever-competitive market, organizations strive to optimize their supply chain management to ensure timely product delivery and customer satisfaction. This research project aims to leverage predictive analytics to enhance supply chain effectiveness by focusing on forecasting shipping days, identifying bottlenecks, and improving transportation efficiency. By

examining factors such as Payment Type, Order Status, Shipping Mode, Order Quantity, and Item Discount Rate, the study aims to predict delays in order delivery.

The research utilizes two datasets, 'PRODUCT' and 'ORDER,' obtained from the Mendeley Data website, which was originally used by DataCo Global between 2015 and 2018. These datasets were merged based on Order Ids to analyze the supply chain process. Prior to analysis, the datasets underwent exploratory data analysis (EDA), including handling missing values, removing duplicates and outliers, normalizing numerical data, and performing One-Hot Encoding on categorical data.

To model the data, various machine-learning techniques were employed. Although Regularizations (Lasso and Ridge regression) and Principal Component Analysis (PCA) did not yield satisfactory results on the selected dataset, baseline models were used. Support Vector Regression (SVR), Random Forests Regressor, and XGBoost Regressor were employed based on data analysis findings. Hyperparameter Tuning was performed to optimize model performance.

The dataset was split into training (70%), validation (15%), and testing (15%) sets using historical data. Effective supply chain management is crucial for organizational success, and this project aims to provide insights and strategies to enhance customer satisfaction and profitability through the implementation of predictive analytics in supply chain management. By addressing challenges and identifying factors influencing order delivery delays, organizations can streamline their operations and achieve greater efficiency.

Model Proposals

Support Vector Regression

The Machine Learning models that were used for the supply chain dataset in the research are Random Forest Regressor, XGBoost Regressor, and SVR. The machine learning algorithm

SVR is successfully used to solve regression issues. The delays in the delivery of products are averted by taking preventative actions by using SVR, which analyzes historical data on shipping delays to find patterns and forecast them. The ability to handle both linear and nonlinear correlations between input and output variables makes it a flexible tool for the analysis of complicated supply chain data.

A kernel technique in SVR enables it to handle big datasets having feature spaces that are high-dimensional and have nonlinear connections between variables. This is essential in the field of the supply chain since there are a variety of factors that impact how quickly products are delivered, such as shipment location, quantity of stock, and modes of transportation.

The SVR model works by mapping the nonlinear features using the Equation (1) (Chicco et al., 2021).

$$f(x) = w\varphi(x) + b \tag{1}$$

Equation (2) and Equation (3) are used to decrease the function of regularized risk, and is solved to get the estimation of the coefficients w and b (Sarhani & Afia, 2014).

$$\min \tfrac{1}{2} \|w\|^2 + C\sum(\xi i + \xi i^*) \tag{2}$$

$$w^T\Phi(xi) - yi <= \varepsilon + \xi i$$

$$yi - w^T\Phi(xi) <= \varepsilon + \xi i^*$$

$$\xi i, \xi i^* >= 0,\ i = 1,....m \tag{3}$$

Here, the predicted values are indicated by f(x). w is the weight vector, b is bias term, Φ(xi) is the input xi's feature vector, αi are the Lagrange multipliers, ξi are slack variables, K(xi, xj) is the kernel function that maps input xi and xj to a higher-dimensional space and yi are the

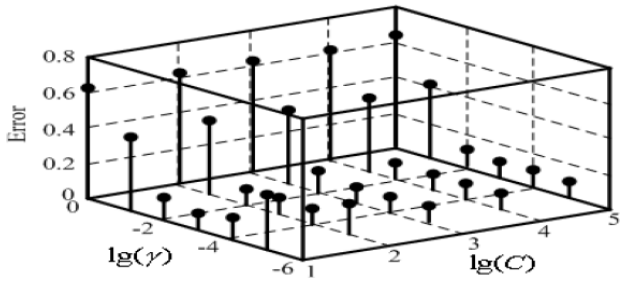
target outputs. By decreasing the regularization term and training error, the model averts underfitting and overfitting of the training data (Sarhani & Afia, 2014).

In order to estimate the learning precision, the study has predicted the highest combination that can be obtained as the most effective features in the M x N combination of C, γ, and epsilon using the grid search method that takes N values in γ and M values in C, respectively.

Error rates shown in Figure 49 illustrate the two main variables that affect the performance, which are the kernel parameter and the error penalty factor. It demonstrates how the kernel parameters and the punish-factor C affect the effectiveness of SVR. Figure 49 from Han et al., 2012, p. 2).

Figure 49

Effectiveness of Kernel Parameters and Punish Factor on SVR Model



In supply chain company operations, where there is frequently a combination of both types of data, SVR's potential to handle both categorical and continuous data is a crucial feature. Missing data, which is typical in supply chain datasets, can also be handled by SVR. Another essential component of SVR that can be adjusted to strike a balance between model complexity and generalization performance is its regularization parameter. This parameter is crucial in

preventing overfitting, which can happen when a model fits the training data too closely and becomes excessively complex, resulting in subpar generalization of new data. As a robust algorithm for predictive modeling in the supply chain sector, SVR can handle single-output and multi-output regression tasks.

Hyperparameter Tuning. Cross-validation was performed, where the dataset was separated into subgroups and one subset was used for training, one subset for testing, and the others for validating. This method prevents overfitting and offers a more precise assessment of the model's performance based on the supply chain data. In SVR with RBF kernel, hyperparameter tuning entails determining the best values for the parameters that influence the model's performance and behavior. A defined set of C, gamma, and epsilon values was used in Grid Search. The regularization parameter (C) manages the adjustment between accepting hyperplane deviations and limiting training error. The kernel coefficient (γ) decides how each training sample affects the SVR model. Epsilon (ε) establishes the size of the error-free region surrounding the predicted value using the epsilon-insensitive loss function. After the SVR model was selected with the kernel as ‘rbf’, the hyperparameter grid was built with the parameters. GridSearchCV was used for cross-validation to test various combinations of hyperparameters, chooses the best ones based on the scoring metric, and is used to execute grid search to retrieve the finest hyperparameters. After performing the iterations to find the best hyperparameter, the result was not satisfying, therefore the baseline SVR model was used.

Random Forest - Regression

According to Abouloifa and Bahaj (2022), RF is well-suited for predicting demand in the supply chain due to its ability to handle high data dimensionality and multicollinearity. RF can effectively capture complex relationships among numerous variables and factors, such as

transportation modes, distances, supplier performance, and weather conditions. It mitigates multicollinearity by using random feature selection and constructing decision trees with subsets of features. RF's ensemble approach and randomization techniques also make it insensitive to overfitting, providing robust predictions of delivery delays. This research utilizes RF regression to predict the demand of products based on other factors (Abouloifa & Bahaj, 2022).

The study by Kinadi et al. (2022) focused on predicting used car prices using RF regression, which is known for its ability to handle large amounts of data with high dimensions, including both categorical and numerical variables. The research highlighted the suitability of RF for handling mixed data types in supply chain predictions, as it can effectively analyze diverse information related to order details, transportation, inventory, and historical performance data, leading to accurate predictions of demand.

The study by Naresh et al. (2022) utilized RF regression to predict stock prices. RF's ensemble of decision trees ensures unbiased predictions, as each tree operates independently and in parallel without influencing one another. This approach allows the model to capture various aspects and patterns within the supply chain dataset, considering factors such as order details, transportation information, inventory levels, and external influences. RF's ability to handle complex and interdependent factors, along with its generalization capabilities and robustness against overfitting, make it well-suited for predicting demand in the supply chain.

The study by Albadrani et al. (2021) focused on predicting the inbound logistic process using RF and other machine learning algorithms. RF was chosen for its ability to handle large volumes of data and its interpretability. In the context of predicting demand, RF's capacity to handle complex datasets with factors such as order details, transportation information, inventory levels, and historical performance data is advantageous. Its decision tree-based ensemble

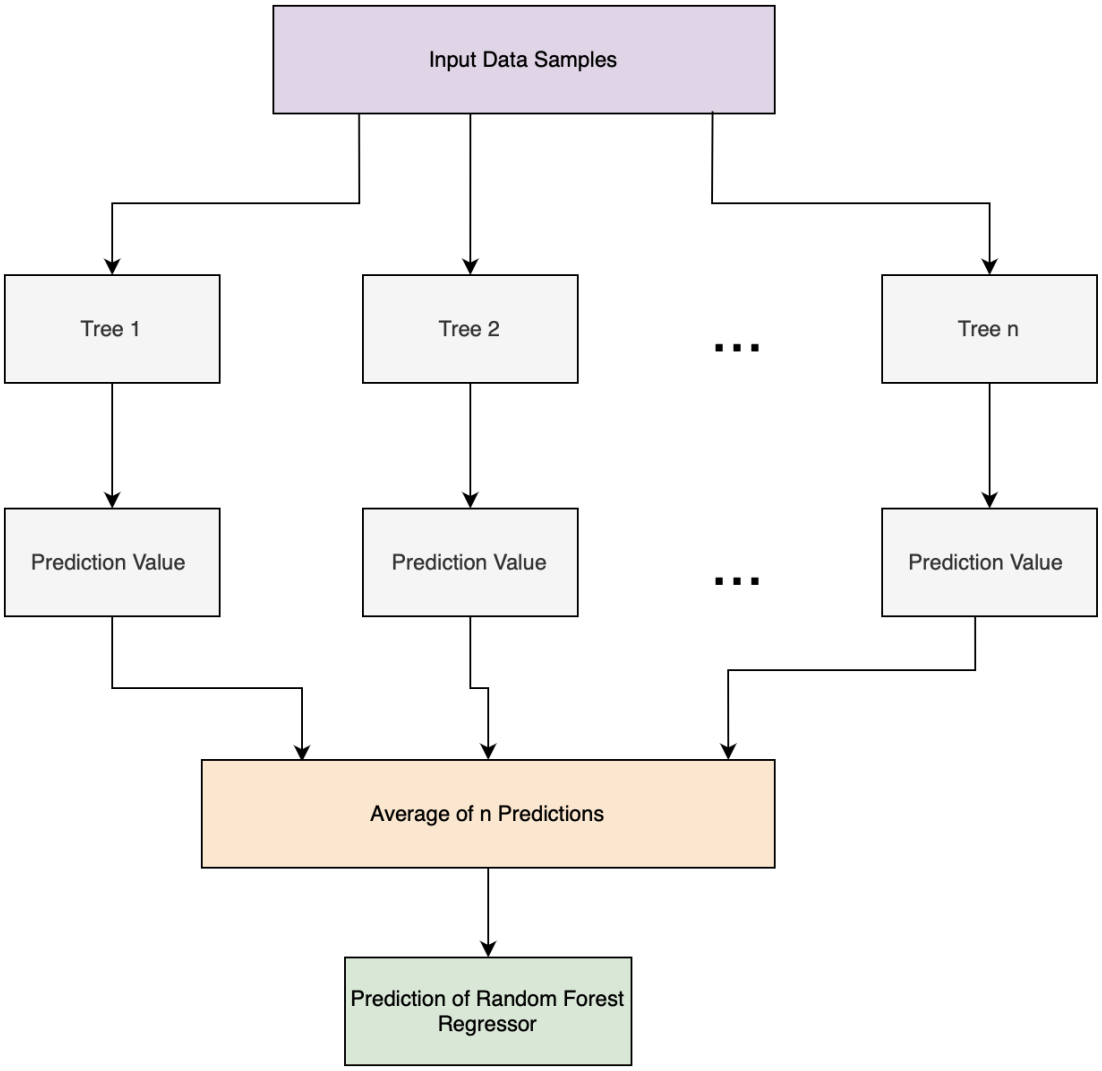
approach allows supply chain professionals to understand the relationships between variables and the importance of each factor in predicting demand. The study's use of RF in predicting the inbound logistic process demonstrates its effectiveness in capturing the intricacies and nonlinear relationships within the supply chain dataset, leading to accurate predictions of demand.

According to Cheng et al. (2019), RF is effective in predicting demand in the supply chain. The study highlighted RF's key feature of de-correlating decision trees through randomization, which improves forecasting accuracy and reduces variance. RF can effectively capture and consider various factors and their interactions, such as order details, transportation information, inventory levels, and historical performance data. RF's ability to reduce variance indicates its capacity to generalize well to unseen data, which is crucial for accurate predictions of demand in a changing supply chain environment.

According to Pillai et al. (2019), RF is a supervised learning technique that builds a forest of decision trees. Each tree is constructed by recursively splitting the data based on decision nodes until reaching the leaf nodes for final predictions. The RF combines the predictions of all trees by taking the mean, resulting in an overall forecast. Figure 50 illustrates the basic working of the RF model, with 'n' trees and the final output obtained by summing the outputs of all trees divided by the number of trees in the forest.

Figure 50

Basic Architecture of RF



The predictions of each tree in the forest are summed, and then the sum is divided by the number of trees in the forest to obtain the output predicted by the RF model as seen in equation (4).

$$final_{pred} = \frac{1}{n} \cdot \sum(t.pred(X)) \quad (4)$$

where, n is the number of trees in the forest, t.pred (X) is the prediction of each individual tree for the input X, Σ is the summation operation and $final_{pred}$ is the final output of the model.

The implementation of the RF algorithm follows a generic structure, as illustrated in Figure 51. The first step involves importing the RandomForestRegressor class from the scikit-learn library, which allows the use of the algorithm for the modeling task. An object representing the RF regression model is then instantiated and stored in a variable. This object serves as the model and will be utilized for both training and making predictions. Next, a variable is employed to define and store the hyperparameters of the model. These parameters determine the configuration of the RF, including the maximum depth of each tree (max_depth) and number of trees (n_estimators). To evaluate the model's performance, the data is split into multiple splits and folds using the repeated k fold cross-validation technique. This results in a total of (k * m) iterations, where k represents the number of splits and m represents the number of folds.

The GridSearchCV method is utilized to perform a grid search over the specified parameter grid. It fits the RandomForestRegressor model on the training data and assesses different combinations of hyperparameters using cross-validation. The best combination of hyperparameters is determined based on the provided scoring metric. A function is employed to calculate the performance of the RandomForestRegressor model. This function applies cross-validation with the specified settings of 'splits' and 'repeats' and returns the scores for each fold. These scores are then averaged to obtain a single value representing the model's performance.

Figure 51

Pseudocode of RF Regression

```
IMPORT RandomForestRegressor using Scikit Learn

STORE RandomForestRegressor model in a variable

STORE parameter of the model in a variable

STORE cross validation value of model using 'k' splits and 'm'
repeats in a variable using RepeatedKFold

CALCULATE AND STORE the value of hypertuned parameters of the
model using GridSearchCV and RepeatedKFold

Calculate and store results using cross_val_score method of
model_selection and applying the hypertuned extracted parameters by
imputing Train and Testing data

Append results in a list of existing results

Print mean absolute error, mean squared error, and R-squared score

```

Note. A pseudocode written for RF regression is shown.

Model Optimization. RF can be optimized by fine-tuning its features and parameters. There are several aspects that can be optimized to enhance the performance of the model. Few of which are listed here.

Number of Trees (n_estimators). Increasing the number of trees increases accuracy but also processing complexity. Optimizing the number of trees entails striking the correct balance between accuracy and processing efficiency.

Maximum Depth (max_depth). The RF model's capacity to capture intricate correlations in the data is influenced by the maximum depth of each decision tree. A deeper tree may overfit the training data, whereas a shallow tree may underfit and so perform poorly. By adjusting the maximum depth, the model may determine the best level of complexity that balances bias and variation.

Minimum Samples Split (min_samples_split). This parameter indicates the minimum number of samples required to split an internal node. By regulating the level of information in the trees, these factors can moderate the model's tendency to overfit. Increasing these numbers can result in a more generic model that is less prone to overfitting.

Randomness Control. Randomness is introduced into the model via features and data sampling. The sample size for each tree (bootstrap) and the number of characteristics evaluated at each split (max_features) can be optimized. Controlling these factors can assist reduce tree correlation and boost variety, resulting in improved generalization and resilience.

Overall, optimization empowers the RF to achieve better generalization, enhanced efficiency, and improved robustness, leading to more accurate predictions and valuable insights.

K-Nearest Neighbors Regression

K-Nearest Neighbors (KNN), according to the study by Xu et al. (2019), is a non-parametric and simple machine learning model used for regression tasks. KNN is used for predicting the value of a target variable by finding the k closest data points to the new observation in the training dataset. The prediction is made by averaging the k nearest values in the target feature. The number of neighbors, k, is a hyperparameter that needs to be specified before the training process begins.

To calculate the distance between the new observation and the training instances, KNN uses various distance metrics such as Euclidean, Manhattan, or Minkowski distance. Once the distances are calculated, the k nearest neighbors are selected based on their distance from the new observation.

A research paper by Ghosh (2006) emphasizes the importance of the choice of k while using KNN. A smaller value of k will result in a model with low bias and high variance, which

may lead to overfitting. Conversely, a larger value of k will result in a model with low variance and high bias, which may lead to underfitting.

To overcome the limitation of choosing the optimal value of k, various techniques such as cross-validation and grid search can be used. Cross-validation is a common technique used to estimate the performance of a model, whereas grid search is a method used to search for the optimal hyperparameters by exhaustively trying all the possible combinations of hyperparameters.

In their research paper, Chomboon et al. (2015) presented different formulas for calculating the distance metrics used in the KNN model. The formulas are conveniently listed in Table 10 (p. 281 - 282).

Table 10

Formulas to Calculate Distance Metrics

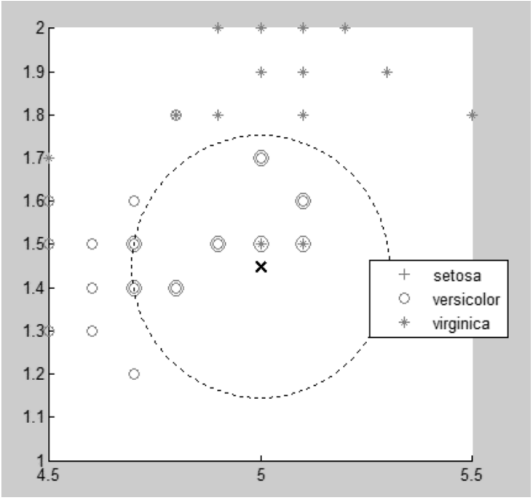
Name	Formula	Description
Euclidean Distance	$d_{st}^2 = (x_s - y_t)(x_s - y_t)'$	The metric is a straight-line distance between two points in Euclidean space.
Standardized Euclidean Distance	$d_{st}^2 = (x_s - y_t)V^{-1}(x_s - y_t)'$	Similar to Euclidean distance, but with weights for each dimension.
Mahalanobis Distance	$d_{st}^2 = (x_s - y_t)C^{-1}(x_s - y_t)'$	It is the measure of the distance between a point and a distribution of data, taking into account the covariance of the data.
City Block Distance	$d_{st} = \sum_{j=1}^{n} \lvert x_{sj} - y_{tj} \rvert$	It is the distance between two points in a grid-like path, i.e., absolute variances of their Cartesian coordinates are added.

Name	Formula	Description
Minkowski Distance	$d_{st} = \sqrt[p]{\sum_{j=1}^{n} \left\|x_{sj} - y_{tj}\right\|^p}$	Generalization of other distance metrics, including Euclidean and City Block, where the parameter p determines the weight assigned to each dimension.
Chebychev Distance	$d_{st} = max_j\{\left\|x_{sj} - y_{tj}\right\|\}$	Measures the distance between two points based on their largest absolute difference in any dimension.
Cosine Distance	$d_{st} = \left(1 - \frac{x_s y'_t}{\sqrt{(x_s x'_s)(y_t y'_t)}}\right)$	Measures the angle between two non-zero vectors in high-dimensional space, often used to determine the similarity between documents or other text data.
Hamming Distance	$d_{st} = \left(\frac{\#(x_{sj} \neq y_{tj})}{n}\right)$	Measures the number of positions at which two strings of equal length differ, often used in coding theory and digital communication.
Jaccard Distance	$d_{st} = \left(\frac{\#\left[(x_{sj} \neq y_{tj}) \cap \left((x_{sj} \neq 0) \cup (y_{tj} \neq 0)\right)\right]}{\#\left[(x_{sj} \neq 0) \cup (y_{tj} \neq 0)\right]}\right)$	Measures the dissimilarity between two sets by calculating the ratio of the size of their union and the size of their intersection.
Spearman Distance	$d_{st} = 1 - \frac{(r_s - \bar{r}_s)(r_t - \bar{r}_t)'}{\sqrt{(r_s - \bar{r}_s)(r_s - \bar{r}_s)'}\sqrt{(r_t - \bar{r}_t)(r_t - \bar{r}_t)'}}$	Measures the difference between the ranks of two sets of data, often used to evaluate correlation in non-parametric statistical tests.
Correlation Distance	$d_{st} = \left(1 - \frac{(x_s - \bar{x}_s)(y_t - \bar{y}_t)'}{\sqrt{(x_s - \bar{x}_s)(x_s - \bar{x}_s)'}\sqrt{(y_t - \bar{y}_t)(y_t - \bar{y}_t)'}}\right)$ where $\bar{x}_s = \frac{1}{n}\sum_j x_{sj}$ $\bar{y}_t = \frac{1}{n}\sum_j y_{tj}$	Measures the distance between two variables in terms of their correlation, often used in pattern recognition and machine learning applications.

Chomboon et al. (2015) then proceeded to examine various distance metrics and compared their performance by training models with the same percentage split dataset to ensure fair and accurate results. Their findings demonstrated that the Hamming and Jaccard techniques performed poorly for classification tasks, while other metrics yielded comparable results. However, the Hamming and Jaccard techniques lacked consistency when tested on multiple other datasets, suggesting that they may not be reliable for KNN models. To better understand how KNN models employ distance metrics and choose the optimal k, Figure 52 from the study can be used. The researchers used the iris dataset to conduct their test, and the figure shows a hypothetical circle enclosing a test data point, with the radius determined by the chosen distance metric. The KNN model selects neighboring points within the circle to make a prediction or classification, with k representing the number of neighboring points selected. Choosing the optimal k is important to prevent underfitting or overfitting, and techniques such as cross-validation and grid search can be used to determine the best hyperparameter value.

Figure 52

KNN Prediction with k = 8



Note. The above figure is referenced from the paper by Chomboon et al. (2015)

The K-Nearest Neighbors (KNN) model is a promising candidate for this project, given the categorical nature of all the descriptive features, including the continuous features such as order quantity and discount rate. The order quantity feature is represented as a finite integer with a maximum data point of 10, while the discount rate ranges from 0.1 to a maximum of 0.25 or 25%. These attributes are ideal for KNN because the model uses a distance metric to compare the similarities between the data points and select the k closest neighbors for prediction, which makes it well-suited for categorical and continuous variables with finite and bounded ranges.

Model Optimization. In comparison to other machine learning models, according to Triguero et al. (2019) in their research, KNN (k-nearest neighbors) is a simple and effective method for classification and regression. However, when dealing with large datasets or high-dimensional feature spaces, KNN's computational complexity can become a bottleneck. Therefore, it is essential to optimize the KNN model's parameters to achieve high performance in terms of time and resource utilization.

The process of finding the optimal parameters is called hyperparameter tuning. The hyperparameters for KNN include the number of neighbors (k), distance metric, algorithm type, etc. Adjusting these parameters can improve the model's accuracy and efficiency. For example, increasing the number of neighbors can reduce the model's tendency to overfit, while decreasing it can increase the model's sensitivity to local features. Similarly, choosing the right distance metric (such as Euclidean or Manhattan distance) can significantly impact the model's performance.

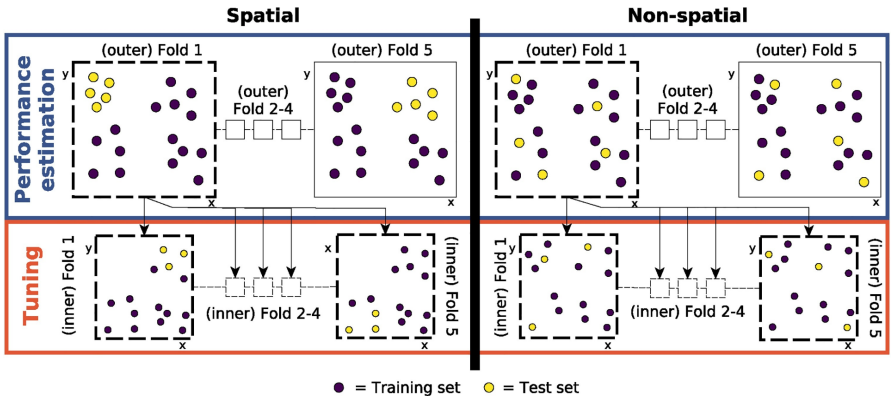
Hyperparameter tuning can also affect the cost of a project, as it can impact the resource requirement, infrastructure, and training time. Therefore, striking a balance between performance and cost is essential.

A study by Schratz et al. (2019) demonstrated how different hyperparameters affect the performance of the KNN model. They found that optimizing the value of k and selecting the appropriate distance metric improved the model's accuracy. Additionally, they showed that using a KD-tree algorithm for finding nearest neighbors can significantly reduce the model's computation time.

Figure 53 from the paper by Schratz et al. (2019) represents spatial and non-spatial nested cross-validation where the researchers used five folds of hyperparameter tuning and performance estimation. The figure displays the training and test sets used for performance estimation, with yellow and purple dots representing the training and test sets, respectively. The selection of the tuning sample relies on the performance estimation fold sample and comprises training and test sets in orange and blue, respectively. The process is repeated five times, with a model trained on the training set and evaluated on the test set for each iteration. Every partition is tested once, and hyperparameter tuning is carried out for each iteration of the performance estimation level. The use of multiple iterations helps to improve the model's performance by reducing the risk of overfitting, a common issue in machine learning.

Figure 53

Spatial and Non-Spatial Nested Cross-Validation



Note. The above figure is referenced in the paper by Schratz et al. (2019).

XGBoost Regressor

XGBoost is an optimized gradient boosting algorithm which is widely used in various fields for both classification and regression tasks. It has proven effective in applications such as fraud detection, recommendation systems, image recognition, and predicting customer churn.

The model development is divided into two parts, the first part is to extract the delivery related descriptive features with focus on feature engineering techniques like normalization to create an appropriate target variable which is used in predicting the delivery delay. In the second phase, the utilization of the feature-engineered data to devise a regression model.

In this paper, Chen and Guestrin (2016) present the motivation behind developing XGBoost and discuss the proposed techniques used in XGBoost, including a sparsity-aware algorithm, a compressed column storage format for parallel processing and reuse of pre-sorted data, a cache-aware prefetching algorithm, and a weighted quantile sketch algorithm. The article demonstrates that XGBoot performs better than others systems in terms of accuracy and speed, especially with large-scale data and resource constraints, where XGBoost achieved a significantly faster execution time of 77.04 seconds compared to 338.1 seconds for Gradient Boosting.

In this research paper, Gómez-Ríos et al. (2017) focussed on class noise and its impact on supervised learning systems. The researchers conducted experiments on 21 datasets with varying levels of manufactured class noise and used different seeds to introduce noise, resulting in multiple versions of the datasets. A 5-fold cross-validation strategy was used to assess a total of 336 datasets. According to the evaluation using the Equalized Loss of Accuracy (ELA) metric, XGBoost regularly beats AdaBoost and GBM in handling label noise, displaying superior

performance and robustness in binary and multi-class classification settings. The most successful method for reducing the disruptive effects of label noise was discovered to be XGBoost.

The research covers a variety of methods rather than focusing on a single model, including LASSO, linear regression, XGBoost, random forest, and SVR. It shows how to overcome obstacles like overfitting (which XGBoost experienced in this study) and shows how to use ensemble approaches to avoid them (Huang et al., 2020).

This study uses eXtreme Gradient Boosting (XGBoost) and Poisson regression analysis to create prediction models for pediatric acute otitis media (AOM). The study focuses on using environmental and air quality factors to forecast the development of AOM in young patients. Data was gathered between 2014 and 2019. Root-mean-square errors (RMSE) and correlation coefficients were used to assess how well the models performed. By showing stronger correlation coefficients and lower RMSE values than the Poisson regression model, the findings indicated that the XGBoost performed better (Mun & Chang, 2022).

In this research study, Lee and Mangalaraj (2022) examines a variety of urban components using an advanced type of ensemble machine learning called extreme gradient boosting, or XGBoost. The researchers looked at a wide range of factors, including components of both traditional and advanced transportation systems. Dataset was obtained through open Big Data platforms. XGBoost was chosen because of its track record of success in handling Big Data and its skill at creating models with the fewest possible faults. The results of the study are interpreted using the SHapley Additive exPlanations (SHAP) method.

The XGBoost approach employs a boosting strategy in which a number of weak learners (decision trees) are successively trained to correct the mistakes of the prior learners. Each weak learner focuses on the samples that were misclassified or have high residuals, allowing the model

to gradually improve its predictions. XGBoost incorporates several key techniques to enhance its performance, including regularization, gradient-based optimization, and parallel processing. The first step is to initialize the model with the constant prediction. The first step in most gradient boosting models is to predict a constant value. This could be the mean of the target variable in the case of regression, or the probability of the majority class in the case of classification. Equation (5) below is referred to from the paper by Chen and Guestrin (2016).

$$F0(x) = \text{argmin}_\gamma \Sigma L(yi, \gamma) \quad (5)$$

where ‘L’ is the chosen loss function, ‘yi’ are the target values and ‘γ’ is a constant. This will output a constant prediction value.

The second step is the iterative portion of the XGBoost algorithm where the model sequentially adds new decision trees to the ensemble. Each of these trees attempts to correct the errors (i.e., the residuals) made by the existing ensemble of trees. This is done for m rounds, where m is a user-specified parameter denoting the total number of trees.

The residuals represent the errors made by the current model. In the context of XGBoost, they're called "pseudo" residuals because they're computed based on the gradient of loss function with respect to the model's predictions. This is what makes XGBoost a "gradient" boosting algorithm—it's using gradient descent to minimize the loss. Mathematically, below two equations (6) and (7) below are referred to from the research by Chen and Guestrin (2016).

$$g_i = \partial L(yi, Fi\text{-}1(xi)) / \partial Fi\text{-}1(xi) \quad (6)$$

$$h_i = \partial^2 L(yi, Fi\text{-}1(xi)) / \partial Fi\text{-}1(xi)^2 \quad (7)$$

where 'yi' is the true label for the i-th occurrence, 'Fi-1(xi)' is the prediction of the i-th instance from the previous iteration, and 'i' ranges across all instances in the dataset. ‘h_i’ and ‘g_i’ are the initial and subsequent derivatives of the loss function 'L'.

The next step is to fit a new decision tree to these pseudo residuals. Below equation (8) is referred to from the research by Chen and Guestrin (2016).

$$Obj = \Sigma [g_i * f(xi) + 1/2 * h_i * f(xi)^2] + \Omega(f) \quad (8)$$

where 'g_i' and 'h_i' are the first and second order gradient statistics, 'f(xi)' is the new tree's output, and ‘Ω(f)’ is the regularization (L1 regularization (Lasso) and L2 regularization (Ridge)).

The contribution of the new tree is scaled by a learning rate ‘η’ to prevent overfitting. This is also known as "shrinkage". The i-th instance's prediction is updated. Below equation (9) is referred to from the research by Chen and Guestrin (2016).

$$Fi(x) = Fi\text{-}1(x) + \eta * f(x) \quad (9)$$

where "f(x)" is the forecast of the new tree, ‘η’ is the learning rate, and ‘Fi-1(x)’ is the forecast of the i-th instance from the previous iteration of learning.

Several methods can be employed to perform optimization of XGBoost which include gradient based optimization, cross-validation, true pruning and Grid Search. XGBoost has a lot of customizable and optimized features which can boost the accuracy. Its unique attributes, including column block structure for parallel learning, regularized learning objective, sparsity awareness, in-built cross-validation, pruning, customization flexibility, and early stopping, make it a formidable tool in the machine learning sphere.

This optimization process helps in identifying the most suitable parameter values that, when incorporated into the model, will enhance its predictive capabilities. The method employed for this process is GridSearchCV. GridSearchCV exhaustively tries out all possible parameter combinations, the optimal parameters discovered via this tuning process are then incorporated back into the training model to assess the improvement in prediction accuracy.

N_estimators which will lead to better learning performance, max_depth represents the maximum depth that each decision tree can achieve.learning_rate (eta) prevents overfitting and also determines how quickly the model learns. Subsample parameter controls the fraction of the total training set that is used for any given boosting round. A lower value can lead to underfitting, while a higher value might lead to overfitting.

The percentage of the cells that will be chosen at random for each tree or for each boosting round is called colsample_bytree. Gamma, which describes the minimal loss reduction necessary to create a new partition on a tree leaf node. The L1 and the L2 regularization factors for the weights, respectively, reg_alpha and reg_lambda, can be employed to prevent overfitting.

Model Supports

Environment, Platform

Table 11

Environment and Platforms

Platform	Version	Purposes
Jupyter Notebook (Python)	v6. 5.4	Data Preprocessing, Cleaning, Analysis, visualizations, Machine Learning Model Building.
Tableau	2023.1	Visualizing and analyzing the data.
GitHub	3.8.0	Version control and to track changes to code over time.

Tools

Table 12

Tools used

Library		Method	Usage
Scikit-Learn	sklearn.ensemble	RandomForestRegressor	Implementation of RF regression model
	sklearn.svm	SVR	Implementation of Support Vector Regression model
	sklearn.neighbors	KNeighborsRegressor	Implement KNN regression algorithm
	sklearn.model_selection	XGBRegressor	implement XGBoost model
		GridSearchCV	Used for hyperparameter tuning.
		train_test_split	Split the data for training, validation and testing
		cross_val_score	Performing cross-validation
		RepeatedKFold	For repeated k-fold cross-validation
	sklearn.preprocessing	MinMaxScaler	Data Normalization
		StandardScaler	Data Normalization
	sklearn.linear_model	Ridge	Data Regularization
	sklearn.metrics	r2_score mean_squared_error, mean_absolute_error	Used for models performance evaluation

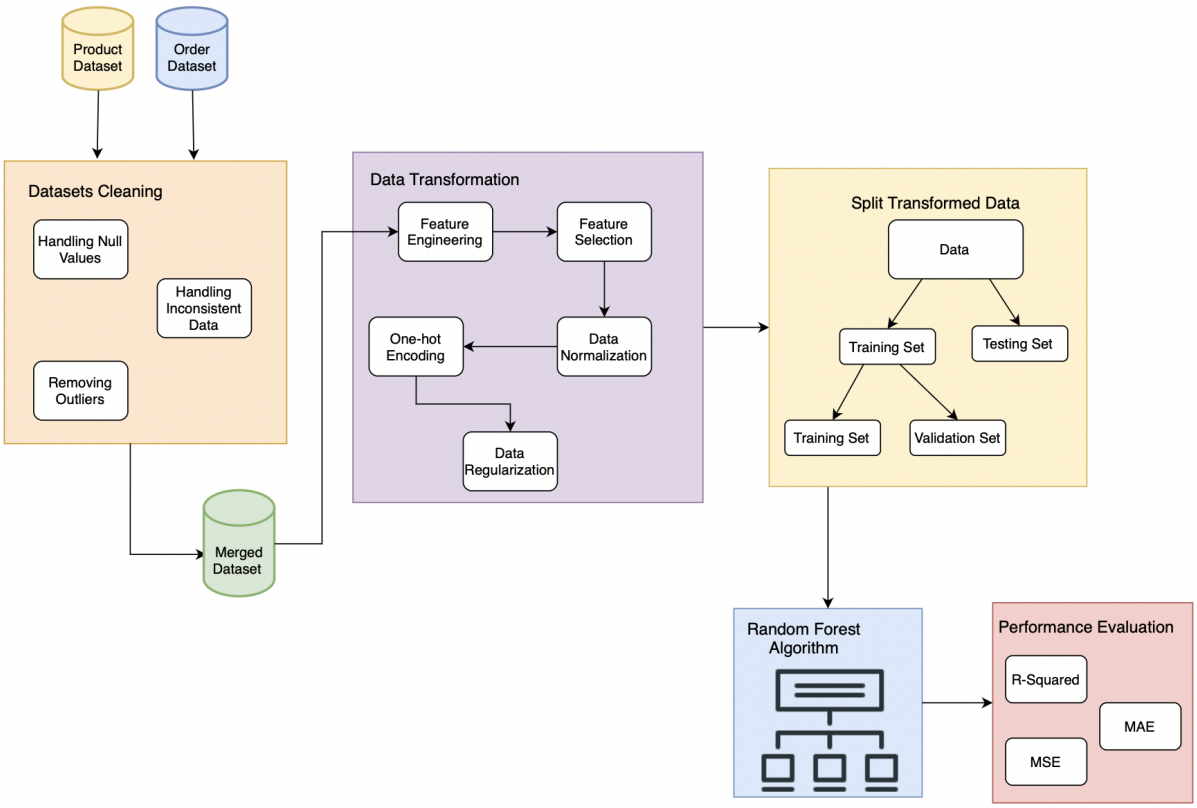
Library		Method	Usage
Pandas	dataFrame, series	drop, shape, head, info, describe, dropna, merge, iloc	Loading the data, data cleaning, and manipulating the data frame, checking data statistics before modeling
	get_dummies	get_dummies	one-hot encoding of categorical variables
Garbage collect	gc	collect	Explicitly run garbage collection to free memory
Seaborn		heatmap, histplot, scatterplot, boxplot	Used for data visualizations
Numpy	numpy.ndarray	std, min, mean, abs, sqrt, sum, max, reshape, concatenate	For numerical computations, Reshape the array to a desired shape, Combine multiple arrays into a single array
Matplotlib	pyplot	pie, title, show, figure,scatter,bar	Plotting various graphs
DateTime	datetime	to_datetime, day_name	Converting date time format, extracting information from dates
Yellowbrick	yellowbrick	regressor	Residual plot and prediction error
Keras	keras.models	Sequential	Define a sequential model architecture
	keras.layers	Dense, Dropout, Conv2D, LSTM	Add layers to the model architecture for learning
	keras.optimizers	Adam, SGD, RMSprop, etc.	Define optimization algorithms for model training

Model Architecture and Data Flow

Random Forest. There are two datasets collected from the DataCo company website. The datasets ‘Order’ and ‘Products’ are first cleaned separately, this allows for better control over data quality, consistency, and the ability to address any issues specific to individual datasets. The outliers are removed and the missing and inconsistent data is handled for each dataset. The datasets are then merged on the cleaned ‘Order_Id’ column. During the following data transformation phase, additional features are extracted from the existing merged dataset. To find the most influential traits, features with a correlation greater than 0.5 or less than -0.5 with regard to the target variable are selectively retained. Numeric features within the specified subset are normalized. For effective representation, categorical features, on the other hand, are encoded using the one-hot encoding technique. Continuing the transformation, regularization is applied to the dataset using ridge regression to prevent any overfitting, assuring the model's robustness and generalizability. Following the transformation, the dataset is partitioned into training, validation, and testing sets in order to evaluate the model's performance. The model is then trained, validated, and tested using the prepared dataset, utilizing the R-squared, MSE, and MAE metrics. These measures provide information about the model's accuracy, goodness of fit, and the ability to predict. The dataflow for this study is shown in Figure 54.

Figure 54

Dataflow for RF

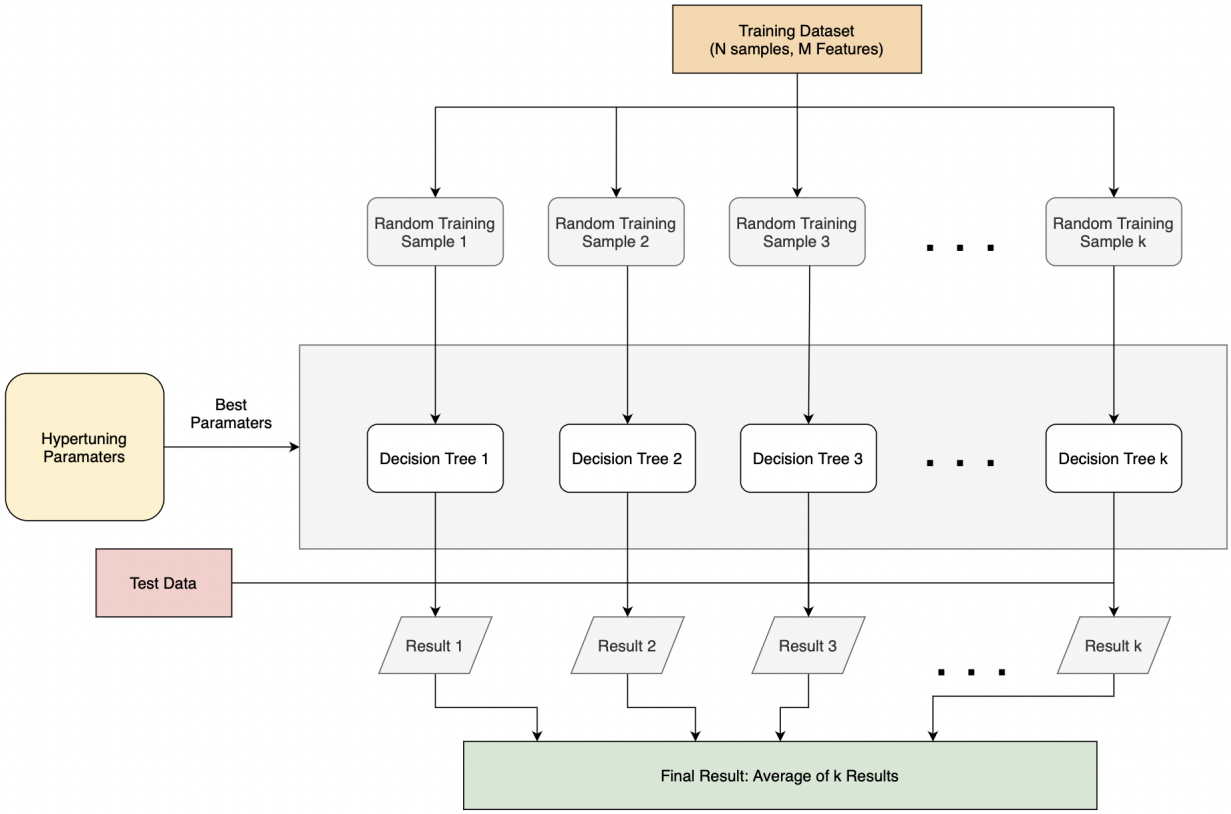


The architecture of a RF involves three main steps that are data sampling, decision tree construction, and aggregation of results. In the data sampling step, a subset of the original training dataset is randomly sampled with replacement using a process called bootstrapping. This creates diversity among the trees by selecting different subsets of data for each tree. Next, in the decision tree construction step, each tree is built independently using the selected subset of the training data. This is done by recursively splitting the data based on the feature values, selecting the best feature and split point at each node. The construction process continues until a stopping criterion is met. The best hyperparameters are passed during the decision tree construction stage. The hyperparameters, such as the maximum depth of each tree, the number of decision trees and the minimum number of samples required for splitting, are set before constructing the individual

decision trees.Finally, in the aggregation of results step, predictions are made for new data points by combining the individual predictions of each tree. For regression tasks, the predictions of all trees are averaged to obtain the final result. Figure 55 shows the architecture of the RF regression model.

Figure 55

Architecture of RF Model



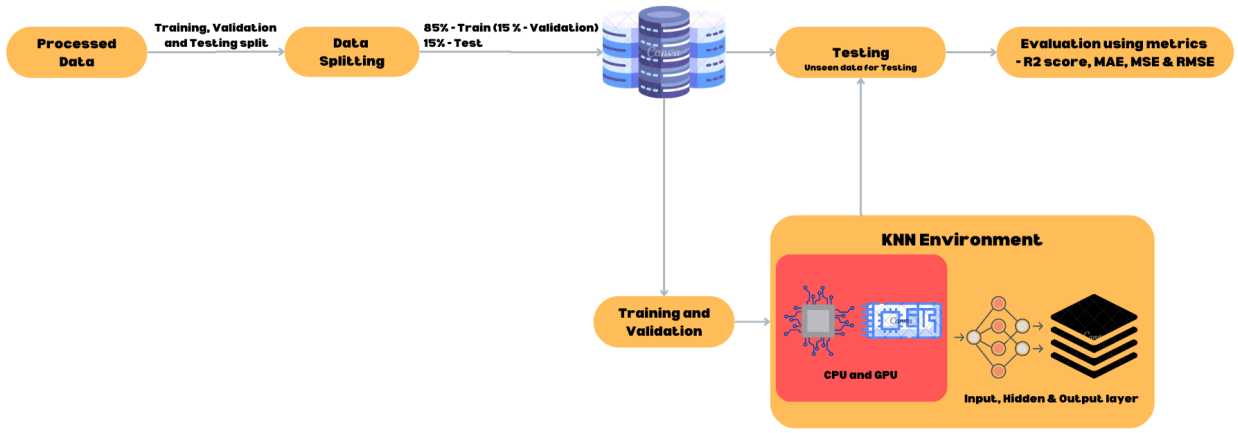
KNN. Figure 56 illustrates the data flow and architecture of the proposed KNN regression model. The data is split into training, testing, and validation sets to prevent overfitting and ensure that the model generalizes well to new data. First, the model is fitted using the training set, the validation set is used to tune the hyperparameters of the model, and using the testing set, the final performance of the model is evaluated. Prior to training the model, the data undergoes pre-processing procedures such as sparsity

removal, feature normalization, and shuffling to prevent overfitting. The training phase utilizes the GPU and multiple processors.

The dataset created during the data processing stage is split into training and testing data in the ratio of 85:15. The training data is then used to fit the KNN regressor model with five neighbors. Once the model is trained, it is used to make predictions on the test set. The predicted values are then compared against the actual values in the test set to calculate the R-squared score. Furthermore, the KNN model is also evaluated on a validation set (15% of the training data). The predicted values are then compared against the actual values in the validation set to calculate the R-squared score and other evaluation metrics for the validation set and the model's performance is assessed by testing these metrics on the unseen data.

Figure 56

KNN Model Architecture and Dataflow



SVR. The SVR-RBF model's data flow for the research begin with supply chain data collection from the data source, where the input features and associated target output were collected. The target output is the 'Delivery Delay' which is the difference between the scheduled number of days for order delivery and the actual number of days for order. The features were then normalized as part of the preprocessing of the data. Additionally, Data Wrangling and detection and removal of outliers were performed. The data was then transformed into a feature space that is high-dimensional and a decision boundary which is non-linear is

created to fit the data before the SVR model is trained on the data using an RBF kernel. The Radial Basis Function (RBF) kernel is a frequently utilized kernel function and is used by calculating the distance among the data points and mapping them to another space where the distance is estimated using a radial basis function. The RBF kernel function expression is shown in Equation (10) (Han et al., 2012).

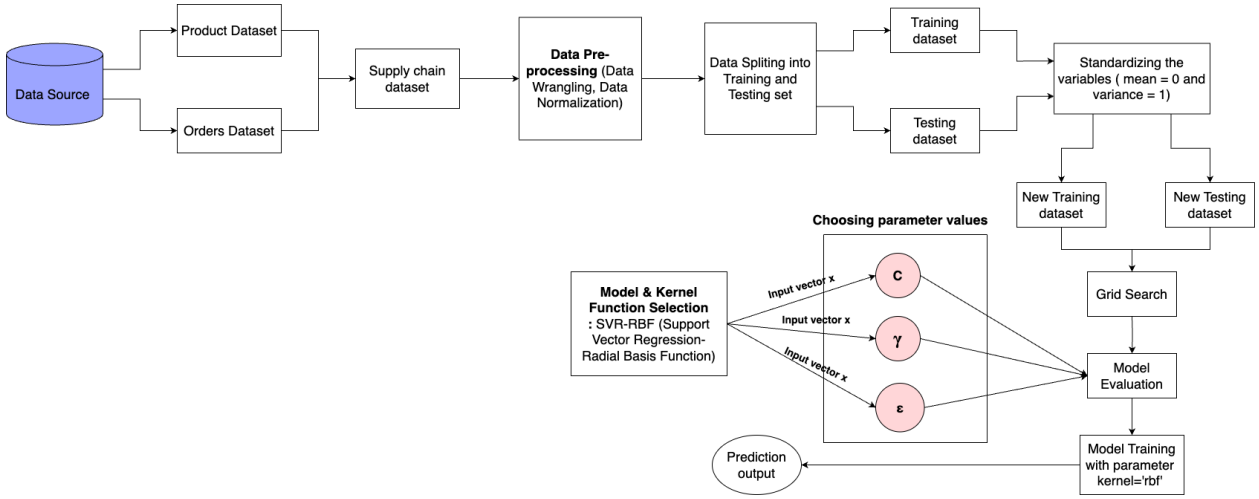
$$K(xi,xj) = \exp(-\gamma \| xi-xj \|)^2 \quad (10)$$

In a particular data subspace, C's function is to modify the learning machine's confidence interval range, and C's optimization varies depending on the data subspace. Changing the kernel parameters γ changes the mapping function, which modifies the distribution's complexity level for the sample's data subspace distribution (Han et al., 2012).

The regularization parameter C and the kernel parameter gamma, which regulates the RBF kernel's form, are both used in the SVR model. These parameters' ideal values are discovered using Grid Search and Cross-Validation. A defined set of C and gamma values is used in Grid Search, and the SVR model is trained and assessed on each set of data. To assess the SVR model's performance on several subsets of data and avoid overfitting, cross-validation was utilized. The Mean Absolute Error (MAE), Mean Squared Error (MSE), and R2 score are the assessment metrics used to evaluate the performance of the SVR model. The target output for new input features is predicted using the SVR model after it has been trained and tuned. High prediction accuracy is possible with the SVR model due to the RBF kernel's ability to capture complicated, non-linear correlations in the data. The in-detail process of the data flow of the SVR model is shown in Figure 57.

Figure 57

Data Flow of SVR Model

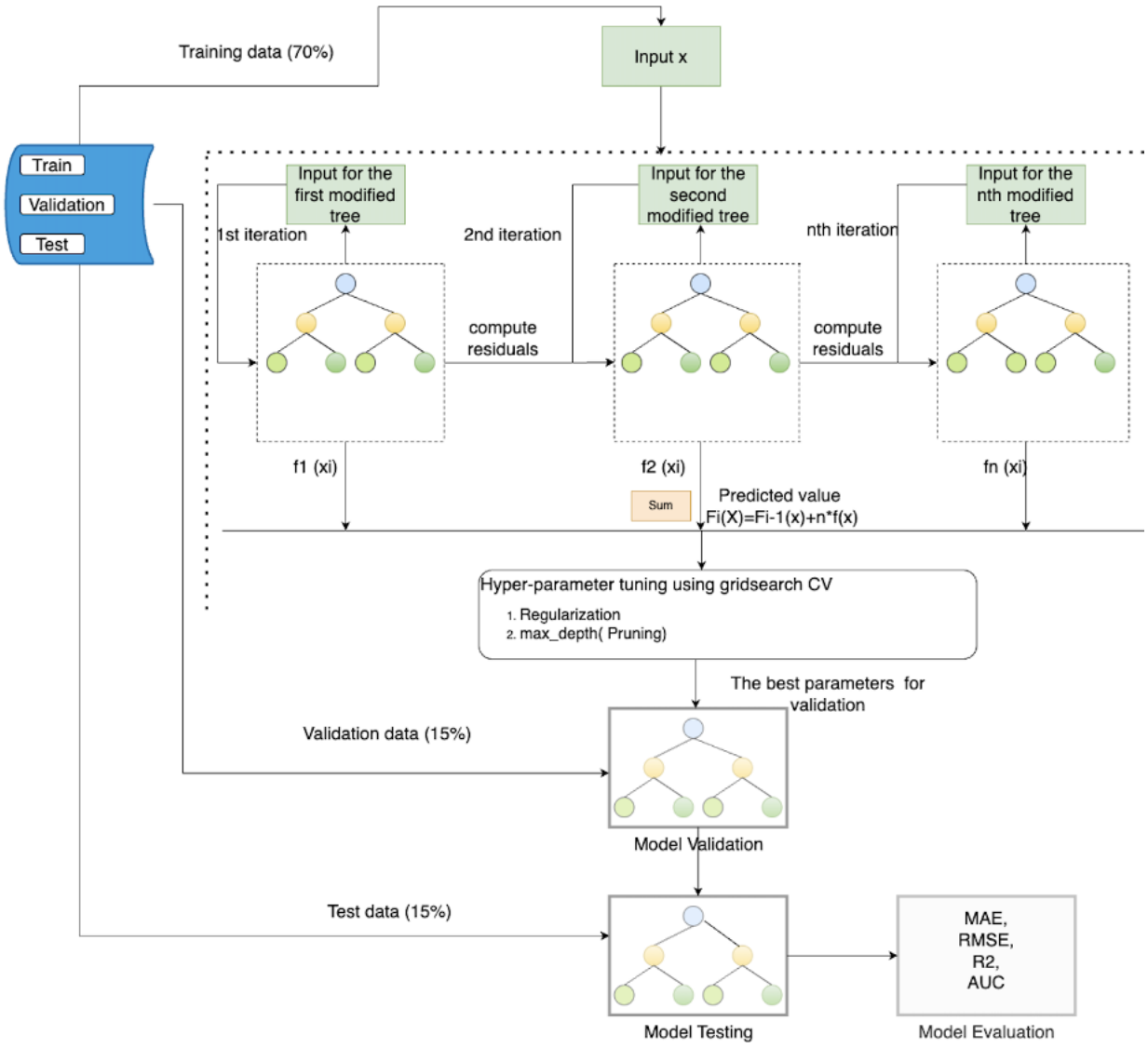


XGBoost. The implementation of XGBoost necessitates a comprehensive and well-prepared dataset, given its demand for high-quality input data. While XGBoost is adept at handling missing values and outlier data points, the effectiveness of the model can be compromised if the data is sparse or incomplete. To address this issue, data preprocessing techniques can be used to remove missing values, handle outliers and inconsistent data.

Once the data has been preprocessed, it is used to train an XGBoost model as shown in Figure 58. The initial stage of model training involves defining the model parameters such as the number of estimators, learning rate, and tree depth. This is followed by training the model on the training data (70%) and using it to make predictions on the test set (15%) and validation set (15%). The XGBoost employs cross validation within the appropriate data set assigned to it. The model's performance is evaluated using relevant evaluation metrics like Mean Squared Error, Mean Absolute Error, and the Root Mean Squared Error. The model is further fine-tuned through hyperparameter tuning techniques like grid search. Subsequently the final model is selected and validated and it is also used for making predictions on new, unknown facts.

Figure 58

Model Architecture and Data Flow



Model Evaluation Methods

Model evaluation methods are essential for assessing the performance and effectiveness of machine learning models. These methods involve metrics such as R-squared, MSE and MAE, which provide insights into the predictive capabilities of regression models.

R-Squared

R-squared calculates the regression results' approximation with real data points. The greater the R-squared value, the better the method because the variable alone explains the variation in the target variable. It has a value ranging from 0 to 1 (Kumar et al., 2022).

MAE

MAE is a statistic used to assess the magnitude of inaccuracy in a collection of anticipated and observed values. It computes the average of the absolute differences between predicted and observed values, giving a measure of the collection's overall error magnitude (Kumar et al., 2022).

MSE

MSE is an error that is used to determine how closely the line fitted resembles the data points. MSE is the summation of the square of errors caused due to the difference in the sample's predicted value and the target value (Kumar et al., 2022).

Table 13

Formulas of the Evaluation Metrics

Name	Formula	Variable
R-squared	$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i \left(a_i - \widehat{a}\right)^2}{\sum_i \left(a_i - \overline{a}\right)^2}$	SS_{RES} : Residuals sum of squares SS_{TOT} : Determination coefficient a_i : Predicted value $\widehat{a}$: Actual value $\overline{a}$: Mean of observed data
MAE	$MAE = \frac{1}{n} \sum_{i=1}^{y} \left\lvert \widehat{a}_i - a_i \right\rvert$	n : Total amount of observational data $\widehat{a}_i$: Predicted value

Name	Formula	Variable
MSE	$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(\widehat{a}_i - a_i\right)^2$	a_i : Actual value n : Total amount of observational data $\widehat{a}_i$: Predicted value a_i : Actual value

Note. The table demonstrates the formulas for the various evaluation metrics and also describes the variables of each formula.

Compared to MSE and RMSE, MAE is less sensitive to outliers in the data. This means that it can be a useful metric when there are extreme values in the dataset that might skew the results. However, it does not penalize large errors as heavily as MSE and RMSE do, which can be a drawback in some situations. R2 score can be useful for understanding how well the model fits the data overall, while MSE and RMSE can be useful for understanding the average size of the errors. MAE can be a good choice when outliers are a concern.

Model Validation and Evaluation

K-Nearest Neighbors Regression

Baseline Model. The KNN model is implemented using the KNeighborsRegressor class from the scikit-learn library. The model is first trained on the training set with 5 nearest neighbors, and the R-squared score is used as the evaluation metric. The KNN model is a non-parametric algorithm, meaning that there are no assumptions made about the underlying distribution of the data.

Before training the model, the data is split into training, validation, and test sets. The training set is used to fit the model and then evaluated on the validation and test sets. To avoid overfitting, the model is evaluated on the validation set after each epoch, and the best model is selected based on the R-squared

score. The evaluation metrics that are used to assess the performance of the KNN Regression model are R2 score, MAE, MSE, and RMSE.

The KNN model achieves an R2 score of 0.795 on the training set, indicating that the model explains 79.5% of the variability in the target variable. The R2 score on the validation set and test set are also close to the training set, with values of 0.797 and 0.795, respectively. This suggests that the model is not overfitting to the training data and will generalize well to the fed new data.

The MAE values for the KNN model are low, with values of 0.071, 0.072, and 0.071 for the training, validation, and test sets, respectively. This indicates that the model's predictions are close to the actual values on average.

The MSE values for the KNN model are also low, with values of 0.012, 0.013, and 0.013 for the training, validation, and test sets, respectively. This suggests that the model's predictions are accurate, with low variance in the error.

Lastly, the RMSE values for the KNN model are the square root of the MSE values, and they are 0.109, 0.114, and 0.114 for the training, validation, and test sets, respectively. This indicates that the model's predictions have an average error of approximately 0.11, in the same units as the target variable, the number of days.

Hyper Parameter Tuning. To further optimize the performance of the KNN model, hyperparameter tuning was conducted using cross-validation and grid search. Specifically, the number of neighbors varied from 1 to 10, and different distance metrics, such as Euclidean and Manhattan, were evaluated. However, the performance of the model did not improve significantly with these changes.

Moreover, the model was also evaluated with different combinations of features and preprocessing methods, such as PCA and feature scaling. However, these modifications did not result in any significant improvement in the model's performance either.

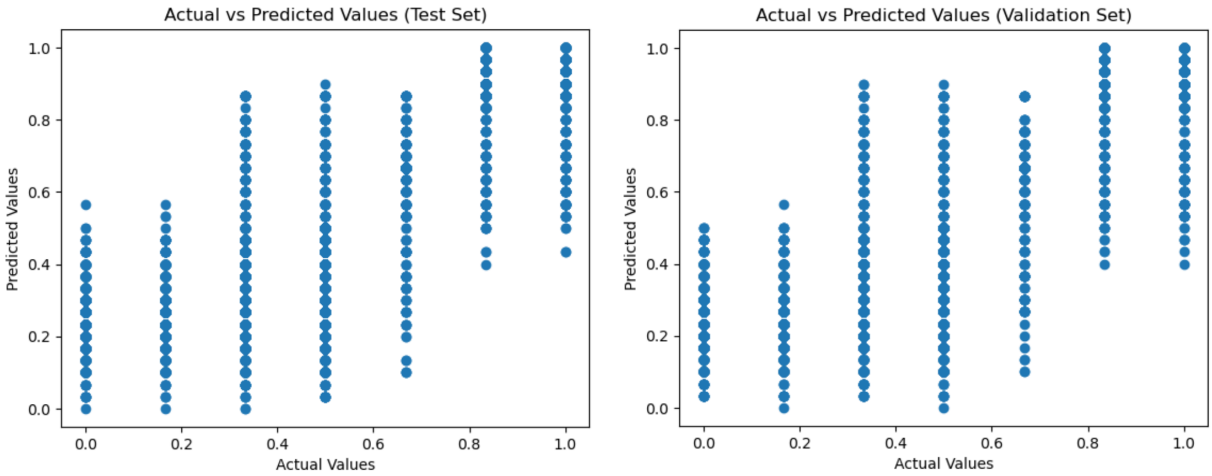
After extensive experimentation, the hyperparameters and preprocessing methods described in the previous section were found to produce the best results. Therefore, they were selected as the final

configuration for the KNN model. The evaluation metrics achieved by this final configuration were considered to be satisfactory, and any further modifications were not deemed necessary.

To visualize the performance of the model, a scatter plot of actual vs predicted values was created both for the test and validation set. From Figure 59, it seems like the majority of predicted values are close to the actual values, which is indicated by the closely packed vertical lines. However, there are a few instances where the predicted value deviates significantly from the actual value, which is indicated by the gaps between dots in those specific vertical lines. Overall, this suggests that the KNN model is performing reasonably well at predicting the target variable, but there may be some instances where the predictions are less accurate.

Figure 59

Actual vs Predicted Values - KNN Model



Random Forest

The baseline model was built using parameter ‘n_estimators’ which was valued 100. It gave a R-squared score of 0.80. Hyperparameter. Max_depth, min_samples_split and n_estimators were parameters given whose best values were found using GridSearchCV. The process took a significant amount of time to be implemented. The best parameters were ‘max_depth’: 5, ‘min_samples_split’: 2, and ‘n_estimators’: 200. Figure 60 shows the best parameters after cross

validation. The hypertuned model resulted in a R-squared score 0.82, 0.070 and 0.015 MAE and MSE respectively.

Figure 60

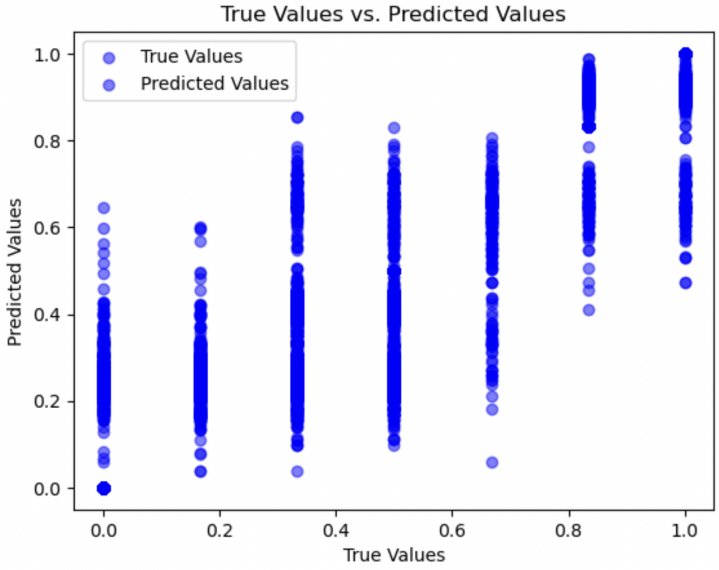
Best Parameters for the RF Model Built

```
{'max_depth': 5, 'min_samples_split': 2, 'n_estimators': 200}
```

To visualize the performance of the model, a scatter plot of true vs predicted values was created for the test set. Figure 61 shows the scatter plot.

Figure 61

RF Scatter Plot



SVR

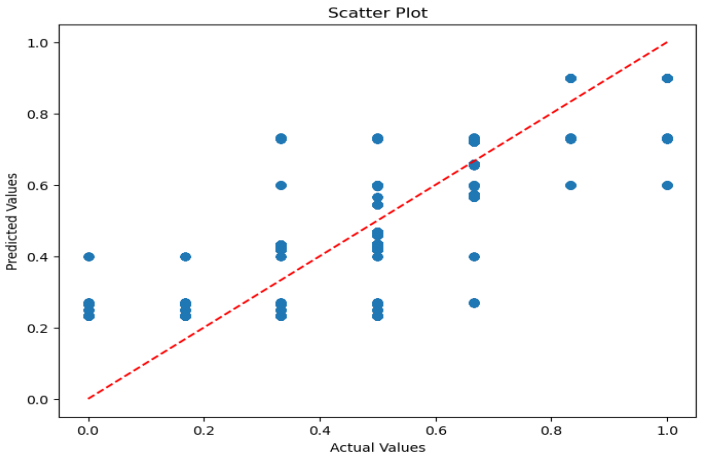
The SVR model was evaluated based on the data presented, with R-squared score ranging 0.810 MAE score of 0.088. The MSE value was 0.011. Hyper parameter tuning was performed. Cross-validation was performed, where the dataset was separated into subgroups and one subset was used for training, one subset for testing, and the others for validating. This method prevents overfitting and offers a more precise assessment of the model's performance based on the supply

chain data. In SVR with RBF kernel, hyperparameter tuning entails determining the best values for the parameters that influence the model's performance and behavior. A defined set of C, gamma, and epsilon values was used in Grid Search. The regularization parameter (C) manages the adjustment between accepting hyperplane deviations and limiting training error. The kernel coefficient (γ) decides how each training sample affects the SVR model. Epsilon (ε) establishes the size of the error-free region surrounding the predicted value using the epsilon-insensitive loss function. After the SVR model was selected with the kernel as 'rbf', the hyperparameter grid was built with the parameters. GridSearchCV was used for cross-validation to test various combinations of hyperparameters, chooses the best ones based on the scoring metric, and is used to execute grid search to retrieve the finest hyperparameters. After performing the iterations to find the best hyperparameter, the result was not satisfying, therefore the baseline SVR model was used.

Figure 62 shows the scatter plot for the actual and predicted values for the SVR model with the RBF kernel function. From the graph, an underlying pattern is seen when comparing both values.

Figure 62

Scatter Plot Distribution



XGBoost

Baseline model for XGBoost.A foundational XGBoost regressor framework is built after data splitting. With all parameters set to default values, the baseline model finishes training on the training set in under 8 minutes. The learning_rate has been set to 0.3, the max_depth is equal to 6, and the size of the subsample is fixed to 1. The n_estimators have been set to 100. The R2 score, the RMSE, and the MAE values are the assessment measures that are employed to evaluate the model's correctness. The baseline model's test dataset has an R2 value of 0.830. The results for RMSE, MSE and MAE are, respectively, 0.118, 0.014 and 0.072.

XGBoost with Hyperparameter Tuning. The collection of parameters that need to be fine-tuned and fit the training dataset are subjected to grid search. The process took more time to complete. The best-fit grid search parameters are n_estimators is equal to 300, max_depth is set to 5, learning_rate has been set to 0.05, subsample equates to 0.8, colsample_bytree = 0.8, gamma = 0.1, reg_alpha = 0.1, and reg_lambda = 0.9. The test results are obtained by fitting these parameters into the model. The grid search hyperparameter test result indicates a boost over the first search results. RMSE is 0.005, MAE is 0.055, and R2 is 0.89. Figure 63 shows the prediction error for GridSearchCV.

Figure 63

Prediction Error for GridSearchCV

```
Prediction Error Plot
```

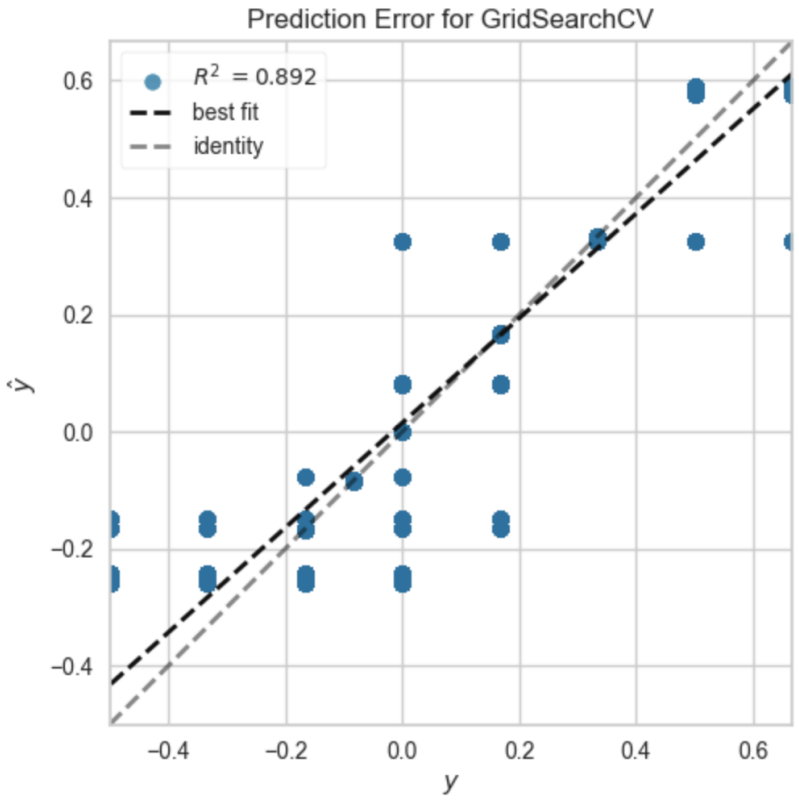


Table 14

Performance Model Comparison Based on R-Squared Score, MAE, MSE

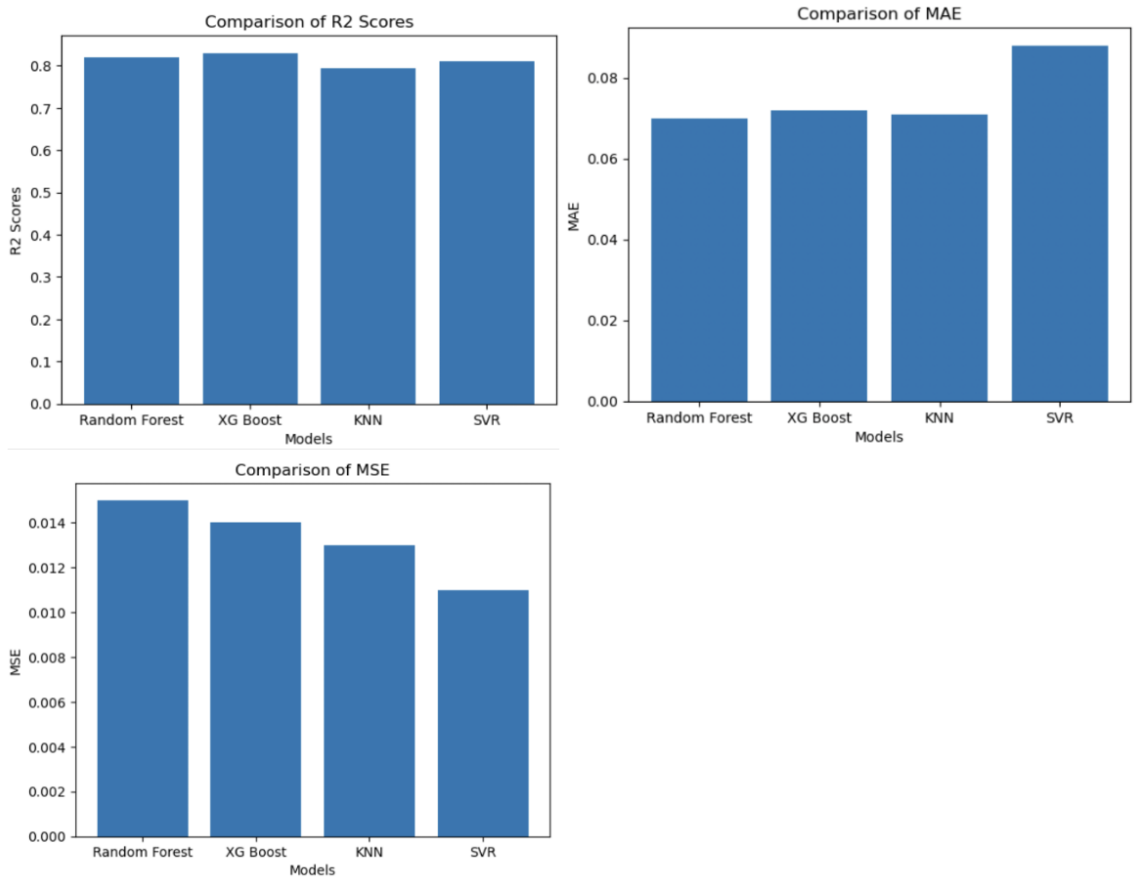
Model	R-Squared	MAE	MSE
Support Vector Regression	0.810	0.088	0.011
Random Forest Regressor	0.82	0.070	0.015
XGBoost	0.830	0.072	0.014
KNN	0.795	0.071	0.013

Note. The scores presented are for the test data.

Figure 64 shows the bar charts for the R2 score, MAE and MSE for all the models implemented.

Figure 64

Comparison Graphs of R2, MAE and MSE



Conclusions

The results indicate that all three models, Random Forest, XGBoost, and KNN, are strong performers on this regression task. They were all able to explain a similar amount of variation in the target variable, as evidenced by the similar R2 scores achieved on the test set. However, there were some differences in their performance based on other evaluation metrics. Random Forest achieved the lowest MSE on the test set, which indicates that its predictions had the smallest errors compared to the other two models. This suggests that Random Forest may be more accurate in predicting the target variable, particularly in cases where smaller errors are important.

In terms of MAE, all three models performed similarly, but Random Forest had the lowest MAE on the test set. This means that, on average, the predictions made by Random Forest were closest to the true values. This is a desirable characteristic in a regression model as it indicates that the model is making predictions that are close to the actual values. Although XGBoost had the highest MAE on the test set, it still performed very well overall. In fact, it had the highest R2 score among the three models. This suggests that XGBoost is a strong model for this regression task, particularly when the focus is on explaining as many variations in the target variable as possible.

KNN performed slightly worse than the other two models in terms of both MSE and MAE. However, its performance was still strong overall, with an R2 score of 0.795 on the test set. This suggests that KNN may be a good option for this regression task in cases where a simple, easy-to-understand model is desired.

In conclusion, all three models performed well on this regression task, with Random Forest being the top performer based on MSE and MAE. But Random Forest took 6 hours to run while XGBoost took only 10 minutes and had the highest R2 score, indicating that it is the best model for explaining variation in the target variable. KNN performed slightly worse than the other two models but still achieved a strong R2 score, indicating that it may be a good option for simple regression tasks.

Limitations

The limitations of this project in using predictive analytics techniques in the supply chain industry include the small size and lack of representativeness of the dataset used. Obtaining a larger and more diverse dataset would improve the accuracy of supply chain efficiency predictions. Additionally, the models used, namely Random Forest, XGBoost, and KNN, were only tested on static data and not real-time data. Evaluating the scalability of these models with real-time data is crucial due to the potential volume and velocity of such data. The models may face challenges with time and space complexity as data size increases, requiring proper resource allocation in production environments. Random Forest models have memory consumption issues

due to their ensemble nature, which combines multiple decision trees, and training them can be computationally intensive. Hyperparameter tuning is necessary for maximizing the performance of Random Forest models, which can further increase training time.

Future scope

There is an immense amount of potential for the research to improve and grow, including creating a website or mobile application that integrates with IoT(Internet of Things). Additional features can be added to the website such as real-time market trends. The website can be hosted with the help of AWS CloudFront for easier and faster accessibility for a wider range of audience. To improve logistics and cut down on the number of days required for product delivery, IoT sensors can offer real-time data on stock levels, delivery schedules, and product quality.

References

Aamer, A. M., Yani, L. E., & Priyatna, I. M. A. (2020). Data Analytics in the Supply Chain Management: Review of Machine Learning Applications in Demand Forecasting. *Operations and Supply Chain Management*, 1–13. https://doi.org/10.31387/oscm0440281

Abouloifa, H., & Bahaj, M. (2022). *Predicting late delivery in Supply chain 4.0 using feature selection: a machine learning model*. https://doi.org/10.1109/commnet56067.2022.9993969

Albadrani, A., Alghayadh, F., Zohdy, M., Aloufi, E., & Olawoyin, R. (2021). *Performance and Predicting of Inbound Logistics Processes Using Machine Learning*. https://doi.org/10.1109/ccwc51732.2021.9376171

Aviv, Y. (2003). A Time-Series Framework for Supply-Chain Inventory Management. *Operations Research*, *51*(2), 210–227. https://doi.org/10.1287/opre.51.2.210.12780

Belhadi, A., Kamble, S. S., Mani, V., Benkhati, I., & Touriki, F. E. (2021). An ensemble machine learning approach for forecasting credit risk of agricultural SMEs' investments in agriculture 4.0 through supply chain finance. *Annals of Operations Research*. https://doi.org/10.1007/s10479-021-04366-9

Cadavid, J., Lamouri, S., & Grabot, B. (2018). Trends in Machine Learning Applied to Demand & Sales Forecasting: A Review. HAL (Le Centre Pour La Communication Scientifique Directe).

Carbonneau, R. A., Laframboise, K., & Vahidov, R. (2008). Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, *184*(3), 1140–1154. https://doi.org/10.1016/j.ejor.2006.12.004

Chen, T., & Guestrin, C. (2016). *XGBoost*. https://doi.org/10.1145/2939672.2939785

Cheng, L., Chen, X., De Vos, J., Lai, X., & Witlox, F. (2019). Applying a random forest method approach to model travel mode choice behavior. *Travel Behaviour and Society*, *14*, 1–10. https://doi.org/10.1016/j.tbs.2018.09.002

Chicco, D., Warrens, M. J., & Jurman, G. (2021). The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ*, *7*, e623. https://doi.org/10.7717/peerj-cs.623

Chomboon, K., Chujai, P., Teerarassammee, P., Kerdprasop, K., & Kerdprasop, N. (2015). *An Empirical Study of Distance Metrics for k-Nearest Neighbor Algorithm*. https://doi.org/10.12792/iciae2015.051

Dhanush, G. A., Raj, K. K., & Kumar, P. (2021). Blockchain Aided Predictive Time Series Analysis in Supply Chain System. In *Springer eBooks* (pp. 913–925). https://doi.org/10.1007/978-981-16-0749-3_70

Filali, A., Lahmer, E. B., & Filali, S. E. (2021). *Exploring applications of Machine Learning for supply chain management*. https://doi.org/10.1109/tst52996.2021.00015

Ghosh, A. K. (2006). On optimum choice of k in nearest neighbor classification. *Computational Statistics & Data Analysis*, *50*(11), 3113–3123. https://doi.org/10.1016/j.csda.2005.06.007

Gómez-Ríos, A., Luengo, J., & Herrera, F. (2017). A Study on the Noise Label Influence in Boosting Algorithms: AdaBoost, GBM and XGBoost. In *Lecture Notes in Computer Science* (pp. 268–280). Springer Science+Business Media. https://doi.org/10.1007/978-3-319-59650-1_23

GuangHui, W. (2012). Demand Forecasting of Supply Chain Based on Support Vector Regression Method. *Procedia Engineering*, *29*, 280–284. https://doi.org/10.1016/j.proeng.2011.12.707

Han, S., Qubo, C., & Meng, H. (2012). Parameter selection in SVM with RBF kernel function. In World Automation Congress (pp. 1–4). https://ieeexplore.ieee.org/document/6321759

Huang, J., Tsai, Y., Wu, P., Lien, Y., Chien, C., Kuo, C. H., Hung, J., Chen, S., & Kuo, C. (2020). Predictive modeling of blood pressure during hemodialysis: a comparison of linear model, random forest, support vector regression, XGBoost, LASSO regression and ensemble method. *Computer Methods and Programs in Biomedicine*, *195*, 105536. https://doi.org/10.1016/j.cmpb.2020.105536

Islam, S., & Amin, S. H. (2020). Prediction of probable backorder scenarios in the supply chain using Distributed Random Forest and Gradient Boosting Machine learning techniques. *Journal of Big Data*, *7*(1). https://doi.org/10.1186/s40537-020-00345-2

Khan, M. S., Saqib, S., Alyas, T., Rehman, A. U., Saeed, Y., Zeb, A., Zareei, M., & Mohamed, E. M. (2020). Effective Demand Forecasting Model Using Business Intelligence Empowered With Machine Learning. *IEEE Access*, *8*, 116013–116023. https://doi.org/10.1109/access.2020.3003790

Kilimci, Z. H., Akyuz, A. O., Uysal, M., Akyokus, S., Uysal, M. O., Bülbül, B., & Ekmis, M. A. (2019). An Improved Demand Forecasting Model Using Deep Learning Approach and Proposed Decision Integration Strategy for Supply Chain. *Complexity*, *2019*, 1–15. https://doi.org/10.1155/2019/9067367

Kinadi, A. F., Andreswari, R., Sutoyo, E., Nugraha, R., & Kamil, A. a. B. (2022). *Used Car Price Prediction in Surabaya Using Random Forest Regressor Algorithms*. https://doi.org/10.1109/icadeis56544.2022.10037526

Kot, S., Grondys, K., & Szopa, R. (2011). Theory of inventory management based on demand forecasting. Polish Journal of Management Studies, 3(1), 147–155. http://yadda.icm.edu.pl/yadda/element/bwmeta1.element.baztech-article-BPC8-0003-0032/c/httppjms_zim_pcz_plpdfpjms3theory20of20inventory20management20based20on20demand20forecasting.pdf

Kumar, A., Mishra, S. K., & Kejriwal, A. (2022). Prediction of Happiness Score of Countries by Considering Maximum Infection Rate of People by COVID-19 using Random Forest Algorithm. In *2022 2nd International Conference on Intelligent Technologies (CONIT)*. https://doi.org/10.1109/conit55038.2022.9847791

Lee, I., & Mangalaraj, G. (2022). Big Data Analytics in Supply Chain Management: A Systematic Literature Review and Research Directions. *Big Data and Cognitive Computing*, *6*(1), 17. https://doi.org/10.3390/bdcc6010017

Mitra, A., Jain, A., Kishore, A., & Kumar, P. (2022). A Comparative Study of Demand Forecasting Models for a Multi-Channel Retail Company: A Novel Hybrid Machine

Learning Approach. *Operations Research Forum*, *3*(4). https://doi.org/10.1007/s43069-022-00166-4

Mun, S. K., & Chang, M. Y. (2022). Development of prediction models for the incidence of pediatric acute otitis media using Poisson regression analysis and XGBoost. *Environmental Science and Pollution Research*. https://doi.org/10.1007/s11356-021-17135-9

Naresh, E., Ananda, B. J., Keerthi, K. S., & Tejonidhi, M. R. (2022). Predicting the Stock Price Using Natural Language Processing and Random Forest Regressor. In *2022 IEEE International Conference on Data Science and Information System (ICDSIS)*. https://doi.org/10.1109/icdsis55133.2022.9915940

Parzen, E. (1961). An Approach to Time Series Analysis. *Annals of Mathematical Statistics*, *32*(4), 951–989. https://doi.org/10.1214/aoms/1177704840

Pillai, M. A., Ghosh, A., Joy, J., Kamal, S., Satheesh, C. C., Balakrishnan, A., & Supriya, M. H. (2019). *Acoustic Source Localization using Random Forest Regressor*. https://doi.org/10.1109/sympol48207.2019.9005303

Praveen, U., Farnaz, G., & Hatim, G. (2019). Inventory management and cost reduction of supply chain processes using AI based time-series forecasting and ANN modeling. *Procedia Manufacturing*, *38*, 256–263. https://doi.org/10.1016/j.promfg.2020.01.034

Sarhani, M., & Afia, A. E. (2014). *Intelligent system based support vector regression for supply chain demand forecasting*. https://doi.org/10.1109/icocs.2014.7060941

Schlegel, G. L. (2014). Utilizing Big Data and Predictive Analytics to Manage Supply Chain Risk. Journal of Business Forecasting, 33(4), 11.

https://www.questia.com/library/journal/1P3-3601906311/utilizing-big-data-and-predictive-analytics-to-manage

Schratz, P., Muenchow, J., Iturritxa, E., Richter, J., & Brenning, A. (2019). Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecological Modelling*, *406*, 109–120. https://doi.org/10.1016/j.ecolmodel.2019.06.002

Schröer, C., Kruse, F., & Gómez, J. M. (2021). A Systematic Literature Review on Applying CRISP-DM Process Model. *Procedia Computer Science*, *181*, 526–534. https://doi.org/10.1016/j.procs.2021.01.199

Seyedan, M., & Mafakheri, F. (2020). Predictive big data analytics for supply chain demand forecasting: methods, applications, and research opportunities. *Journal of Big Data*, *7*(1). https://doi.org/10.1186/s40537-020-00329-2

Sheremetov, L., González-Sánchez, A., López-Yáñez, I., & Levashova, T. (2013). Time Series Forecasting: Applications to the Upstream Oil and Gas Supply Chain. *IFAC Proceedings Volumes*, *46*(9), 957–962. https://doi.org/10.3182/20130619-3-ru-3018.00526

Tarallo, E. A., Akabane, G. K., Shimabukuro, C. I., Mello, J., & Amancio, D. (2019). Machine Learning in Predicting Demand for Fast-Moving Consumer Goods: An Exploratory Research. *IFAC-PapersOnLine*, *52*(13), 737–742. https://doi.org/10.1016/j.ifacol.2019.11.203

Toktay, L. B., & Wein, L. M. (2001). Analysis of a Forecasting-Production-Inventory System with Stationary Demand. *Management Science*, *47*(9), 1268–1281. https://doi.org/10.1287/mnsc.47.9.1268.9787

Triguero, I., García-Gil, D., Maillo, J., Luengo, J., García, S., & Herrera, F. (2019). Transforming big data into smart data: An insight on the use of the k‐nearest neighbors algorithm to obtain quality data. *Wiley Interdisciplinary Reviews-Data Mining and Knowledge Discovery*, *9*(2). https://doi.org/10.1002/widm.1289

Vairagade, N., Logofătu, D., Leon, F., & Muharemi, F. (2019). Demand Forecasting Using Random Forest and Artificial Neural Network for Supply Chain Management. In *Lecture Notes in Computer Science* (pp. 328–339). Springer Science+Business Media. https://doi.org/10.1007/978-3-030-28377-3_27

Xi, J., & Sha, P. B. (2014). Research on Optimization of Inventory Management Based on Demand Forecasting. *Applied Mechanics and Materials*, *687–691*, 4828–4831. https://doi.org/10.4028/www.scientific.net/amm.687-691.4828

Xu, H., Zhou, J., Asteris, P. G., Armaghani, D. J., & Tahir, M. M. (2019). Supervised Machine Learning Techniques to the Prediction of Tunnel Boring Machine Penetration Rate. *Applied Sciences*, *9*(18), 3715. https://doi.org/10.3390/app9183715

Zwißler, F., & Hermann, M. (2012). Supply Chain Risk Management in the Electronics Industry. In *InTech eBooks*. https://doi.org/10.5772/32958

Appendix A

```
merged_df = df_product.merge(df_order, on='Order ID')
merged_df
```

	Type	Delivery_Status	Customer_City	Customer_Country	Customer_Segment	Customer_State	Customer_Street	Department_Name	Market	Order
0	DEBIT	Advance shipping	Caguas	Puerto Rico	Consumer	PR	5365 Noble Nectar Island	Fitness	Pacific Asia	B
1	TRANSFER	Late delivery	Caguas	Puerto Rico	Consumer	PR	2679 Rustic Loop	Fitness	Pacific Asia	Bi
2	CASH	Shipping on time	San Jose	EE. UU.	Consumer	CA	8510 Round Bear Gate	Fitness	Pacific Asia	Bi
3	DEBIT	Advance shipping	Los Angeles	EE. UU.	Home Office	CA	3200 Amber Bend	Fitness	Pacific Asia	Towr
4	PAYMENT	Advance shipping	Caguas	Puerto Rico	Corporate	PR	8671 Iron Anchor Corners	Fitness	Pacific Asia	Towr
...	...	...	...	...	...	...	...	...	...	
180514	CASH	Shipping on time	Brooklyn	EE. UU.	Home Office	NY	1322 Broad Glade	Fan Shop	Pacific Asia	Sha
180515	DEBIT	Late delivery	Bakersfield	EE. UU.	Corporate	CA	7330 Broad Apple Moor	Fan Shop	Pacific Asia	Hir
180516	TRANSFER	Late delivery	Bristol	EE. UU.	Corporate	CT	97 Burning Landing	Fan Shop	Pacific Asia	Ade
180517	PAYMENT	Advance shipping	Caguas	Puerto Rico	Consumer	PR	2585 Silent Autumn Landing	Fan Shop	Pacific Asia	Ade
180518	PAYMENT	Shipping on time	Caguas	Puerto Rico	Consumer	PR	697 Little Meadow	Fan Shop	Pacific Asia	Nag

Figure A1. The datasets, ‘Order’ and ‘Product’ are merged on the ‘Order ID’ feature.

```
[ ] df.dropna(inplace=True)
```

```
[ ] df
```

	Type	Days for shipping (real)	Days for shipment (scheduled)	Benefit per order	Sales per customer	Delivery Status	Late_delivery_risk	Category Id	Category Name	Customer City	...	Order Item Profit Ratio	Order Item Quantity
55	PAYMENT	2.0	2.0	22.410000	74.680000	Shipping on time	0.0	13.0	Electronics	Caguas	...	0.30	2.0
56	PAYMENT	5.0	2.0	25.240000	90.150002	Late delivery	1.0	12.0	Boxing & MMA	Caguas	...	0.28	2.0
57	PAYMENT	6.0	2.0	30.570000	117.580002	Late delivery	1.0	17.0	Cleats	Caguas	...	0.26	2.0
58	PAYMENT	4.0	2.0	46.070000	95.980003	Late delivery	1.0	17.0	Cleats	Caguas	...	0.48	2.0
183	TRANSFER	5.0	4.0	28.850000	128.220001	Shipping canceled	0.0	13.0	Electronics	Freeport	...	0.23	3.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
179402	TRANSFER	5.0	4.0	-178.440002	101.970001	Late delivery	1.0	44.0	Hunting & Shooting	Caguas	...	-1.75	4.0

Figure A2. Dropped null values in Order table

```
[ ] del df['Order Zipcode']

df.isnull().sum()

Type                           0
Days for shipping (real)       0
Days for shipment (scheduled)  0
Benefit per order              0
Sales per customer             0
Delivery Status                0
Late_delivery_risk             0
Category Id                    0
Category Name                  0
Customer City                  0
Customer Country               0
Customer Email                 0
Customer Fname                 0
Customer Id                    0
Customer Lname                 0
Customer Password              0
Customer Segment               0
Customer State                 0
Customer Street                0
Customer Zipcode               0
Department Id                  0
Department Name                0
Latitude                       0
Longitude                      0
```

Figure A3. Dropped Order Zipcode column

```
Q1 = df200['Product Price'].quantile(0.25)
Q3 = df200['Product Price'].quantile(0.75)
IQR = Q3 - Q1
outlier_threshold = 1.5 * IQR
```

```
def remove_outliers(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    outlier_threshold = 1.5 * IQR
    outliers = (column < (Q1 - outlier_threshold)) | (column > (Q3 + outlier_threshold))
    return column[~outliers]
```

```
# Remove outliers from the 'Order Item Total' column
df200['Order Item Total'] = remove_outliers(df200['Order Item Total'])
# Remove outliers from the 'Sales' column
df200['Sales'] = remove_outliers(df200['Sales'])
# Remove outliers from the 'Order Item Product Price' column
df200['Order Item Product Price'] = remove_outliers(df200['Order Item Product Price'])
# Remove outliers from the 'Sales per customer' column
df200['Sales per customer'] = remove_outliers(df200['Sales per customer'])
```

```
sns.boxplot(data=df200[['Sales per customer', 'Order Item Product Price', 'Sales', 'Order Item Total']],
            orient='h')
```

```
<AxesSubplot:>
```

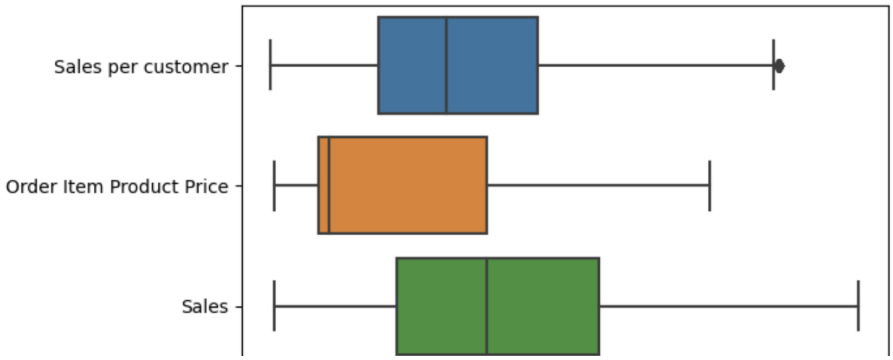


Figure A4. Removing the outliers

```
df['order_date'] = pd.to_datetime(df['order_date'], format='%Y/%m/%d')
df['shipping_date'] = pd.to_datetime(df['shipping_date'], format='%Y/%m/%d')

# calculate order processing days
df['order_processing_days'] = (df['shipping_date'] - df['order_date']).dt.days

# print DataFrame
df
```

	Type	Delivery_Status	Customer_City	Customer_Country	Customer_Segment	Customer_State	Customer_Street	Department_Name	Market	Order
0	DEBIT	Advance shipping	Caguas	Puerto Rico	Consumer	PR	5365 Noble Nectar Island	Fitness	Pacific Asia	B
1	TRANSFER	Late delivery	Caguas	Puerto Rico	Consumer	PR	2679 Rustic Loop	Fitness	Pacific Asia	Bi
2	CASH	Shipping on time	San Jose	EE. UU.	Consumer	CA	8510 Round Bear Gate	Fitness	Pacific Asia	Bi
3	DEBIT	Advance shipping	Los Angeles	EE. UU.	Home Office	CA	3200 Amber Bend	Fitness	Pacific Asia	Towr
4	PAYMENT	Advance shipping	Caguas	Puerto Rico	Corporate	PR	8671 Iron Anchor Corners	Fitness	Pacific Asia	Towr
...	...	...	...	...	...	...	...	...	...	
180514	CASH	Shipping on time	Brooklyn	EE. UU.	Home Office	NY	1322 Broad Glade	Fan Shop	Pacific Asia	Sha
180515	DEBIT	Late delivery	Bakersfield	EE. UU.	Corporate	CA	7330 Broad Apple Moor	Fan Shop	Pacific Asia	Hir
180516	TRANSFER	Late delivery	Bristol	EE. UU.	Corporate	CT	97 Burning Landing	Fan Shop	Pacific Asia	Ade

Figure A5. Calculating the order processing date

```
df['Delivery_Delay'] = df['Days_for_shipment_(scheduled)'] - df['Days_for_shipping_(real)']
```

```
df
```

	Type	Delivery_Status	Customer_City	Customer_Country	Customer_Segment	Customer_State	Customer_Street	Department_Name	Market	Order
0	DEBIT	Advance shipping	Caguas	Puerto Rico	Consumer	PR	5365 Noble Nectar Island	Fitness	Pacific Asia	B
1	TRANSFER	Late delivery	Caguas	Puerto Rico	Consumer	PR	2679 Rustic Loop	Fitness	Pacific Asia	Bi
2	CASH	Shipping on time	San Jose	EE. UU.	Consumer	CA	8510 Round Bear Gate	Fitness	Pacific Asia	Bi
3	DEBIT	Advance shipping	Los Angeles	EE. UU.	Home Office	CA	3200 Amber Bend	Fitness	Pacific Asia	Towr
4	PAYMENT	Advance shipping	Caguas	Puerto Rico	Corporate	PR	8671 Iron Anchor Corners	Fitness	Pacific Asia	Towr
...	...	...	...	...	...	...	...	...	...	
180514	CASH	Shipping on time	Brooklyn	EE. UU.	Home Office	NY	1322 Broad Glade	Fan Shop	Pacific Asia	Sha
180515	DEBIT	Late delivery	Bakersfield	EE. UU.	Corporate	CA	7330 Broad Apple Moor	Fan Shop	Pacific Asia	Hir
180516	TRANSFER	Late delivery	Bristol	EE. UU.	Corporate	CT	97 Burning Landing	Fan Shop	Pacific Asia	Ade
180517	PAYMENT	Advance shipping	Caguas	Puerto Rico	Consumer	PR	2585 Silent Autumn Landing	Fan Shop	Pacific Asia	Ade
180518	PAYMENT	Shipping on time	Caguas	Puerto Rico	Consumer	PR	697 Little Meadow	Fan Shop	Pacific Asia	Nag

Figure A6. Calculating the delivery delay

```
df = pd.get_dummies(df, columns=['Order_Status'], prefix='Order_Status')
```

```
df = pd.get_dummies(df, columns=['Product_Name'], prefix='Product_Name')
```

```
df = pd.get_dummies(df, columns=['Shipping_Mode'], prefix='Shipping_Mode')
```

```
df = pd.get_dummies(df, columns=['Category_Name'], prefix='Category_Name')
```

```
df
```

	Customer_City	Customer_Country	Customer_State	Customer_Street	Market	Order_City	Order_Country	Order_Region	Order_State	Category_Id	...	C
0	Caguas	Puerto Rico	PR	5365 Noble Nectar Island	Pacific Asia	Bekasi	Indonesia	Southeast Asia	Java Occidental	73	...	
1	Caguas	Puerto Rico	PR	2679 Rustic Loop	Pacific Asia	Bikaner	India	South Asia	Rajastán	73	...	
2	San Jose	EE. UU.	CA	8510 Round Bear Gate	Pacific Asia	Bikaner	India	South Asia	Rajastán	73	...	
3	Los Angeles	EE. UU.	CA	3200 Amber Bend	Pacific Asia	Townsville	Australia	Oceania	Queensland	73	...	
4	Caguas	Puerto Rico	PR	8671 Iron Anchor Corners	Pacific Asia	Townsville	Australia	Oceania	Queensland	73	...	
...	...	...	...	...	...	...	...	...	...	...	...	
180514	Brooklyn	EE. UU.	NY	1322 Broad Glade	Pacific Asia	Shanghái	China	Eastern Asia	Shanghái	45	...	
180515	Bakersfield	EE. UU.	CA	7330 Broad Apple Moor	Pacific Asia	Hirakata	Japón	Eastern Asia	Osaka	45	...	
180516	Bristol	EE. UU.	CT	97 Burning Landing	Pacific Asia	Adelaide	Australia	Oceania	Australia del Sur	45	...	
180517	Caguas	Puerto Rico	PR	2585 Silent Autumn Landing	Pacific Asia	Adelaide	Australia	Oceania	Australia del Sur	45	...	
180518	Caguas	Puerto Rico	PR	697 Little Meadow	Pacific Asia	Nagercoil	India	South Asia	Tamil Nadu	45	...	

180519 rows × 239 columns

Figure A7. One-hot encoding

```python
from sklearn.preprocessing import MinMaxScaler

# create a scaler object for Sales normalization
sales_scaler = MinMaxScaler(feature_range=(0, 1))

# fit and transform the Sales column
sel_df['Delivery_Delay'] = sales_scaler.fit_transform(sel_df[['Delivery_Delay']])
sel_df['Order_Item_Quantity'] = sales_scaler.fit_transform(sel_df[['Order_Item_Quantity']])
sel_df['Order_Item_Discount_Rate'] = sales_scaler.fit_transform(sel_df[['Order_Item_Discount_Rate']])

```

```python
sel_df
```

	Order_Item_Discount_Rate	Order_Item_Quantity	Delivery_Delay	Delivery_Status_Advance shipping	Delivery_Status_Late delivery	Delivery_Status_Shipping canceled	Delivery_Statu
0	0.16	0.0	0.833333	1	0	0	
1	0.20	0.0	0.500000	0	1	0	
2	0.24	0.0	0.666667	0	0	0	
3	0.28	0.0	0.833333	1	0	0	
4	0.36	0.0	1.000000	1	0	0	
...	...	...	...	...	...	...	
180514	0.00	0.0	0.666667	0	0	0	
180515	0.04	0.0	0.500000	0	1	0	
180516	0.08	0.0	0.500000	0	1	0	
180517	0.12	0.0	0.833333	1	0	0	
180518	0.16	0.0	0.666667	0	0	0	

Figure A8. Data normalization

```python
from sklearn.linear_model import Ridge
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_absolute_error

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Scale the input features using StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Create and train the Ridge regression model
ridge = Ridge(alpha=1.0)  # You can adjust the regularization strength by changing the alpha value
ridge.fit(X_train_scaled, y_train)

# Evaluate the model on the test set
score = ridge.score(X_test_scaled, y_test)
print("R^2 Score: {:.2f}".format(score))

# Predict on the test set
y_pred = ridge.predict(X_test_scaled)

# Calculate Mean Absolute Error (MAE)
mae = mean_absolute_error(y_test, y_pred)
print("MAE: {:.2f}".format(mae))

```

```
R^2 Score: 0.81
MAE: 0.08
```

Figure A9. L2 Regularization

```python
from sklearn.model_selection import train_test_split

# Split the data into training and test sets (85% train, 15% test)
X_train, X_test, y_train, y_test = train_test_split(changed_data, y, test_size=0.15, random_state=42)

# Split the training data into training and validation sets (70% train, 15% validation)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.1765, random_state=42)


# Print the shape of each dataset
print('For Input Features')
print('Training set shape:', X_train.shape)
print('Validation set shape:', X_val.shape)
print('Test set shape:', X_test.shape)
```

```
For Input Features
Training set shape: (126358, 25)
Validation set shape: (27083, 25)
Test set shape: (27078, 25)
```

Figure A10. Data split to train, validate and test.

```python
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import GridSearchCV, cross_val_score, RepeatedKFold
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import numpy as np

# Create a Random Forest Regressor model
model = RandomForestRegressor()

start_time = time.time()
rf.fit(X_train, y_train)
end_time = time.time()



# Define the parameter grid for hyperparameter tuning
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 5, 10],
    'min_samples_split': [2, 5, 10]
}

# Define the cross-validation strategy
cv = RepeatedKFold(n_splits=10, n_repeats=4)

# Perform hyperparameter tuning using GridSearchCV
grid_search = GridSearchCV(model, param_grid, scoring='neg_mean_squared_error', cv=cv)
grid_search.fit(X_train, y_train)

# Get the best hyperparameters
best_params = grid_search.best_params_

# Create a new model with the best hyperparameters
model = RandomForestRegressor(**best_params)
model.fit(X_train, y_train)

# Calculate and store results using cross-validation
results = cross_val_score(model, X, y, scoring='neg_mean_squared_error', cv=cv)
rmse_scores = np.sqrt(-results)

# Print mean absolute error, mean squared error, and R-squared score
print("Mean Absolute Error:", np.mean(rmse_scores))
print("Mean Squared Error:", mean_squared_error(y_test, y_pred))
print("R-squared Score:", r2_score(y_test, y_pred))
```

Figure A11. Random forest model implemented

```python
import xgboost as xgb


xgb_model = xgb.XGBRegressor(objective='reg:squarederror', random_state=42)

# Fit the model on the training set
xgb_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = xgb_model.predict(X_test)

# Calculate the R-squared score
r2 = r2_score(y_test, y_pred)

# Print the R-squared score
print("R-squared score:", r2)
print("MAE:", mean_absolute_error(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
```

Figure A12. XGBoost implementation

```python
from sklearn.neighbors import KNeighborsRegressor


# Create a KNN regressor object
knn_model = KNeighborsRegressor(n_neighbors=5)

# Fit the model on the training set
knn_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = knn_model.predict(X_test)

# Calculate the R-squared score
r2 = r2_score(y_test, y_pred)

# Print the R-squared score
print("R-squared score:", r2)
print("MAE:", mean_absolute_error(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
```

Figure A13. KNN implementation

```python
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error
from sklearn.svm import SVR
# Create an SVR object
svr_model = SVR(kernel='rbf')

# Fit the model on the training set
svr_model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = svr_model.predict(X_test)

# Calculate the R-squared score
r2 = r2_score(y_test, y_pred)

# Print the R-squared score
print("R-squared score:", r2)
print("MAE:", mean_absolute_error(y_test, y_pred))
print("MSE:", mean_squared_error(y_test, y_pred))
```

Figure A14. SVR implementation

Appendix B

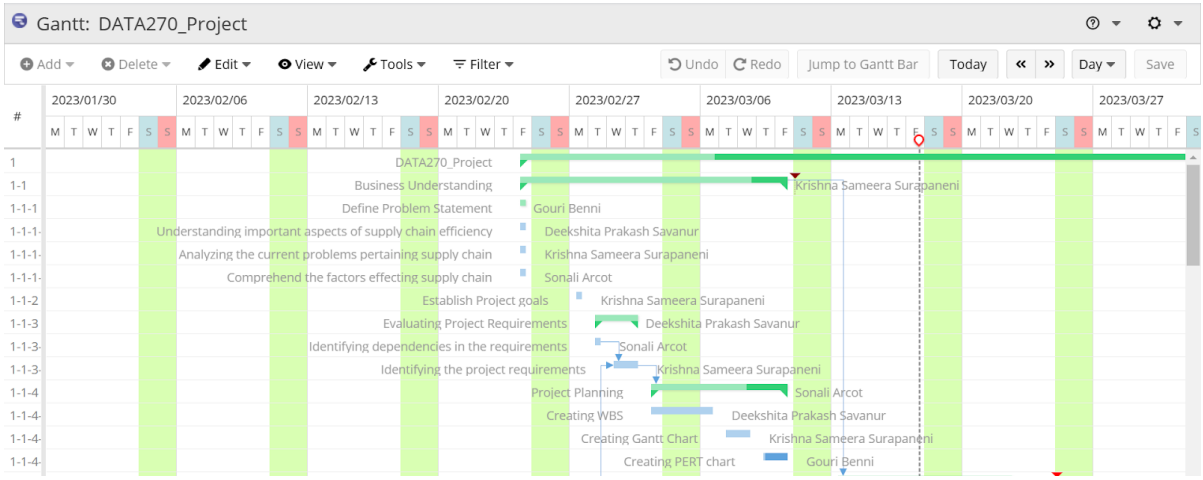


Figure B1. Gantt Chart explaining Business Understanding.

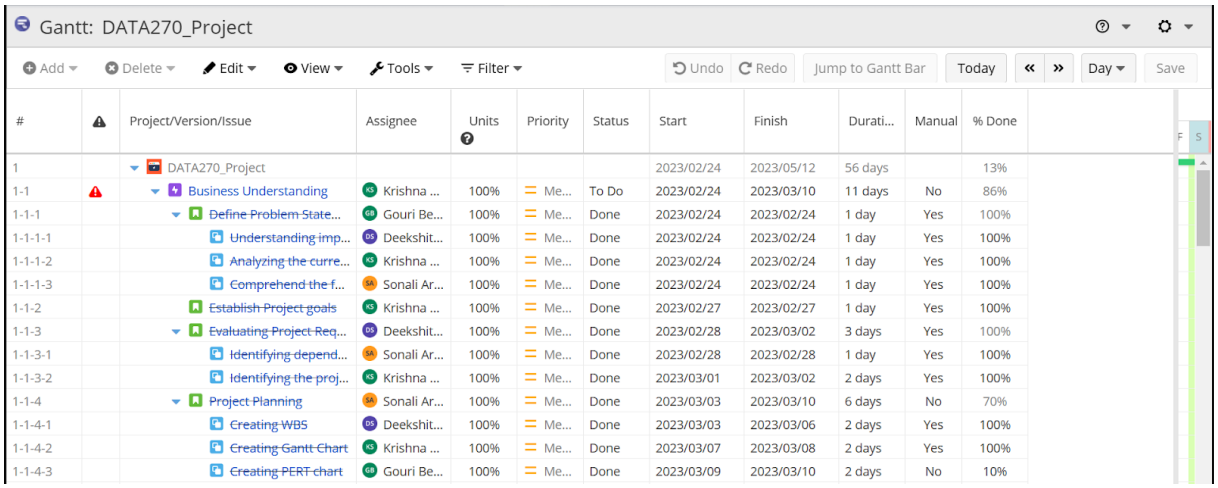


Figure B2. Effort Estimate under Business Understanding.

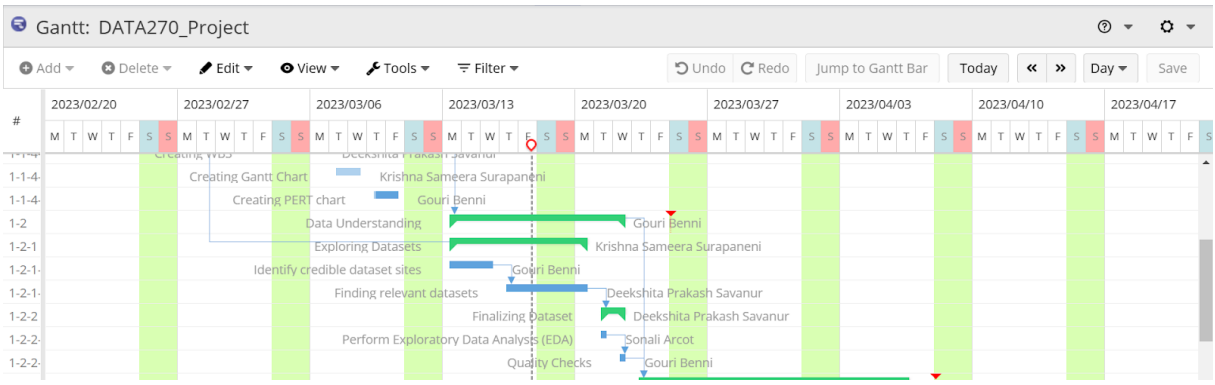


Figure B3. Gantt Chart explaining Data Understanding.

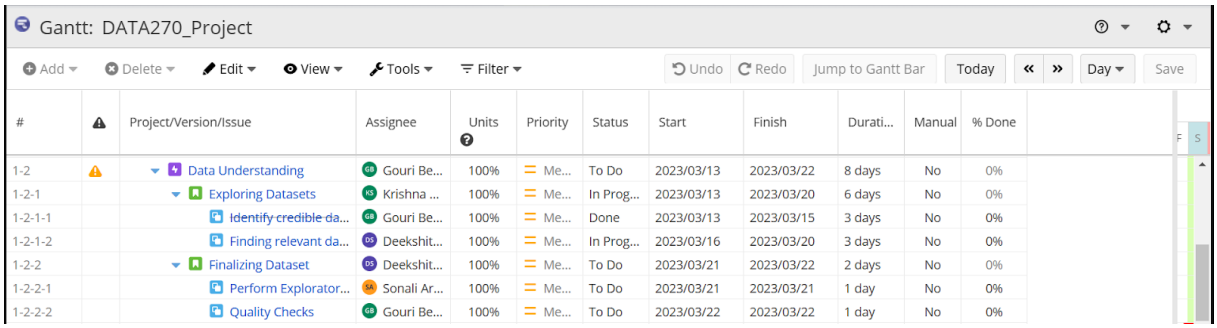


Figure B4. Effort Estimate under Data Understanding.

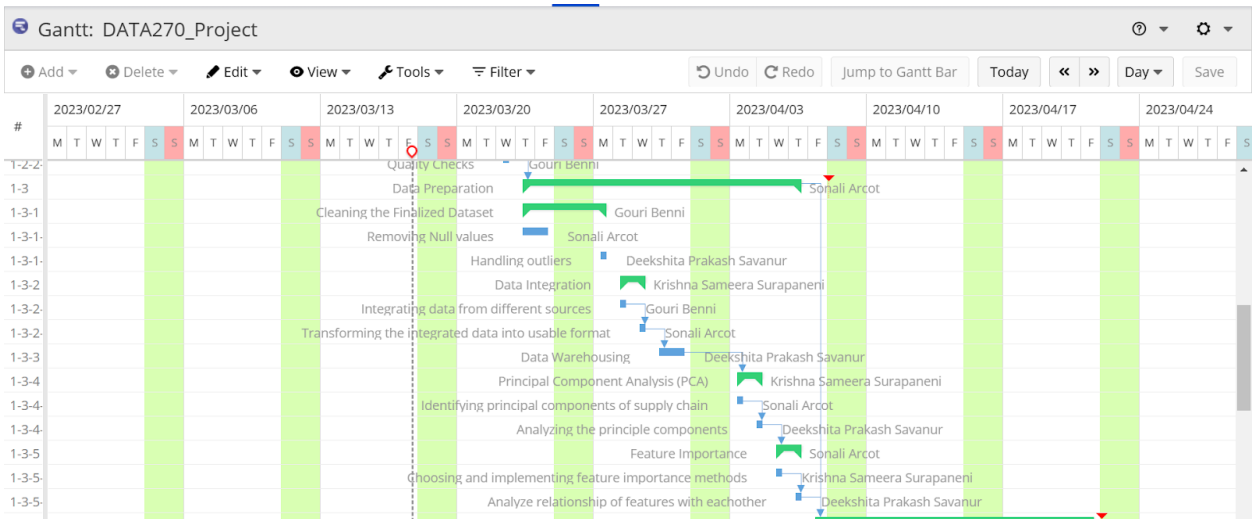


Figure B5. Gantt Chart explaining Data Preparation.

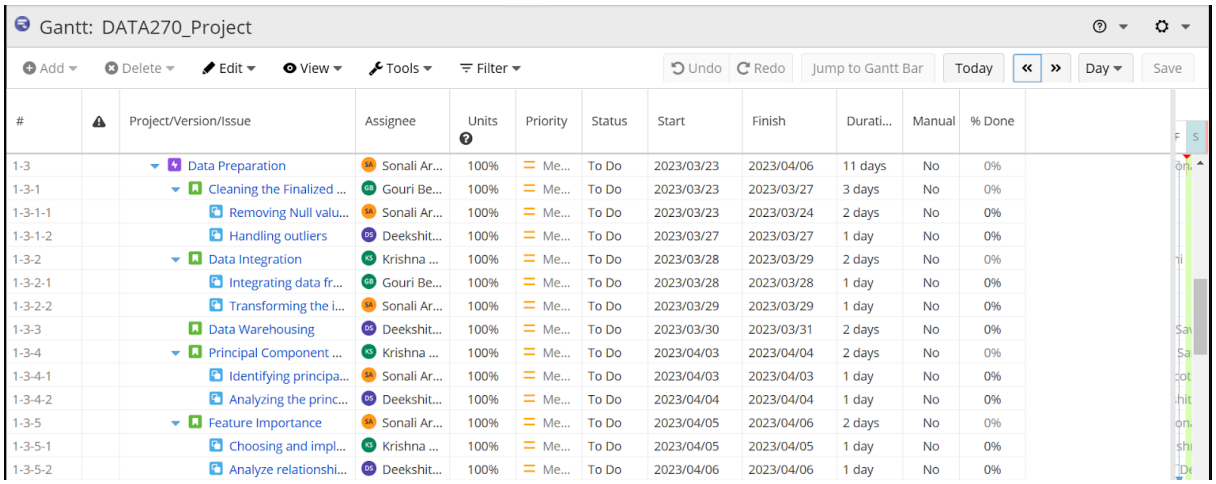


Figure B6. Effort Estimate of Data Preparation.

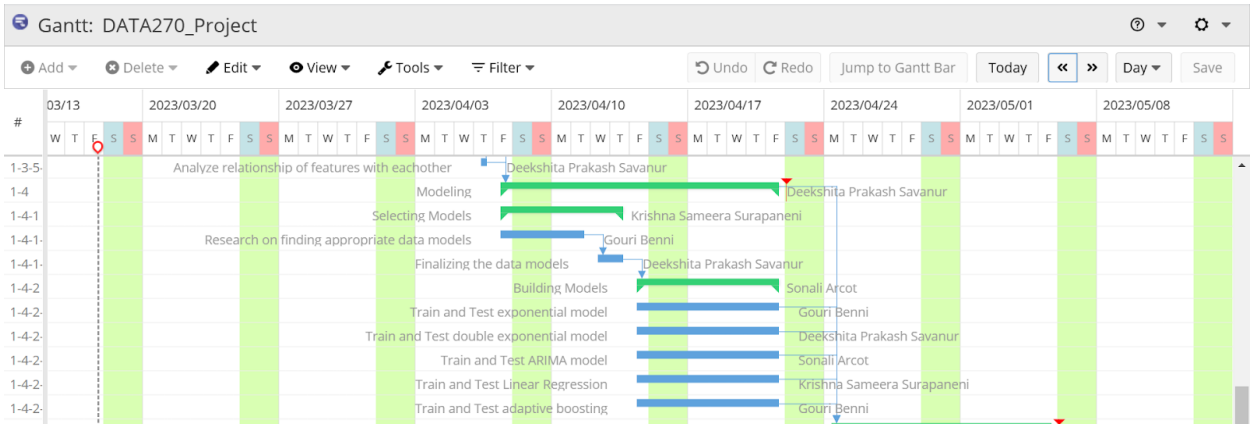


Figure B7. Gantt Chart explaining Data Modeling.

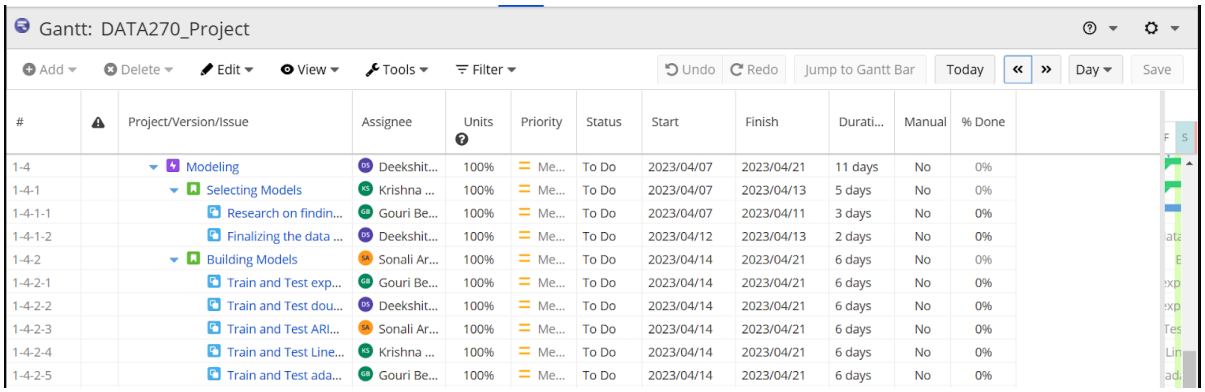


Figure B8. Effort Estimate of Data Modeling.

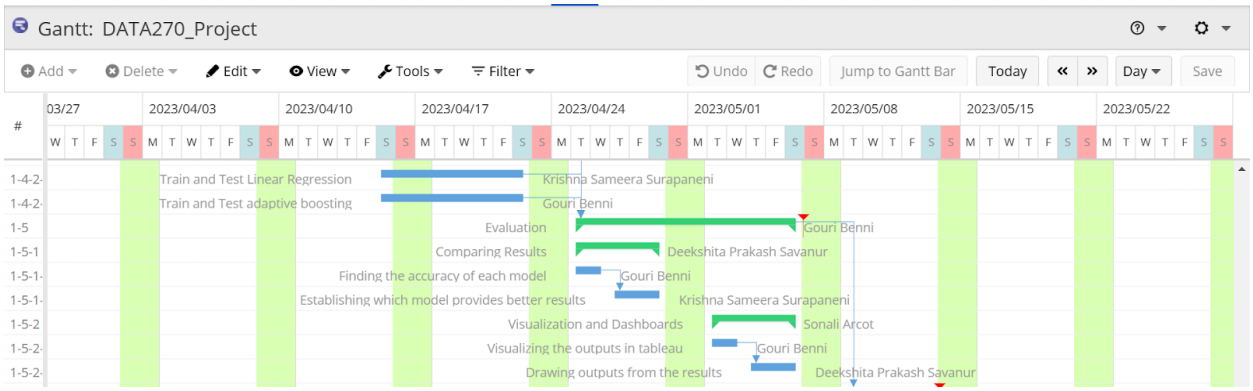
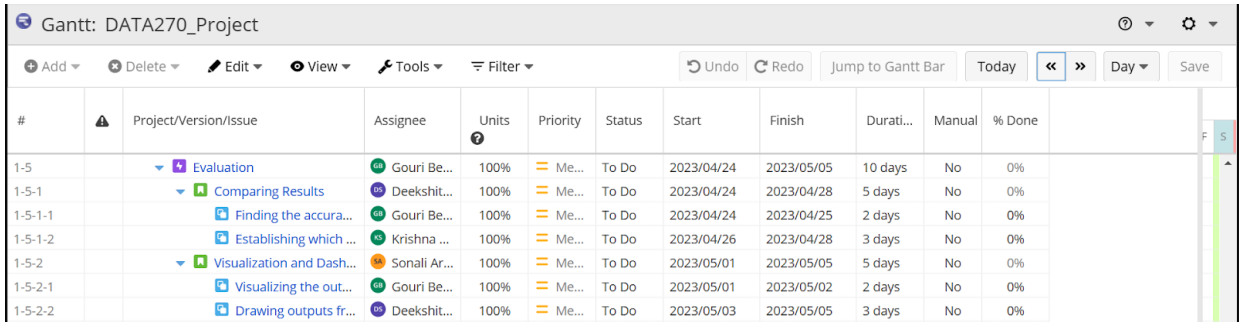


Figure B9. Gantt Chart explaining Data Evaluation.



#	Project/Version/Issue	Assignee	Units	Priority	Status	Start	Finish	Durati...	Manual	% Done
1-5	Evaluation	Gouri Be...	100%	Me...	To Do	2023/04/24	2023/05/05	10 days	No	0%
1-5-1	Comparing Results	Deekshit...	100%	Me...	To Do	2023/04/24	2023/04/28	5 days	No	0%
1-5-1-1	Finding the accura...	Gouri Be...	100%	Me...	To Do	2023/04/24	2023/04/25	2 days	No	0%
1-5-1-2	Establishing which ...	Krishna ...	100%	Me...	To Do	2023/04/26	2023/04/28	3 days	No	0%
1-5-2	Visualization and Dash...	Sonali Ar...	100%	Me...	To Do	2023/05/01	2023/05/05	5 days	No	0%
1-5-2-1	Visualizing the out...	Gouri Be...	100%	Me...	To Do	2023/05/01	2023/05/02	2 days	No	0%
1-5-2-2	Drawing outputs fr...	Deekshit...	100%	Me...	To Do	2023/05/03	2023/05/05	3 days	No	0%

Figure B10. Effort Estimate of Data Evaluation.

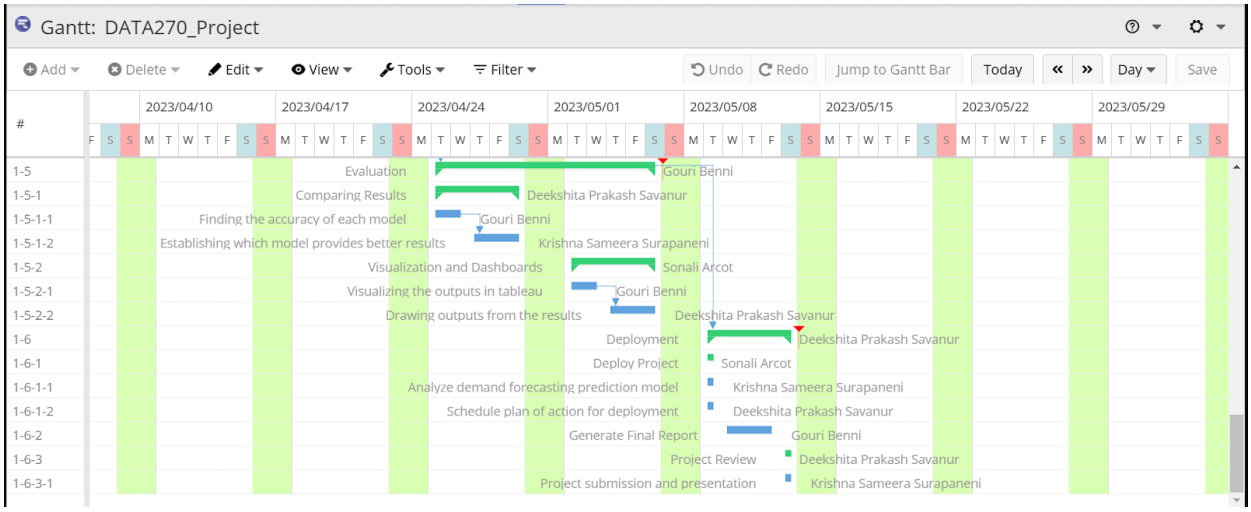
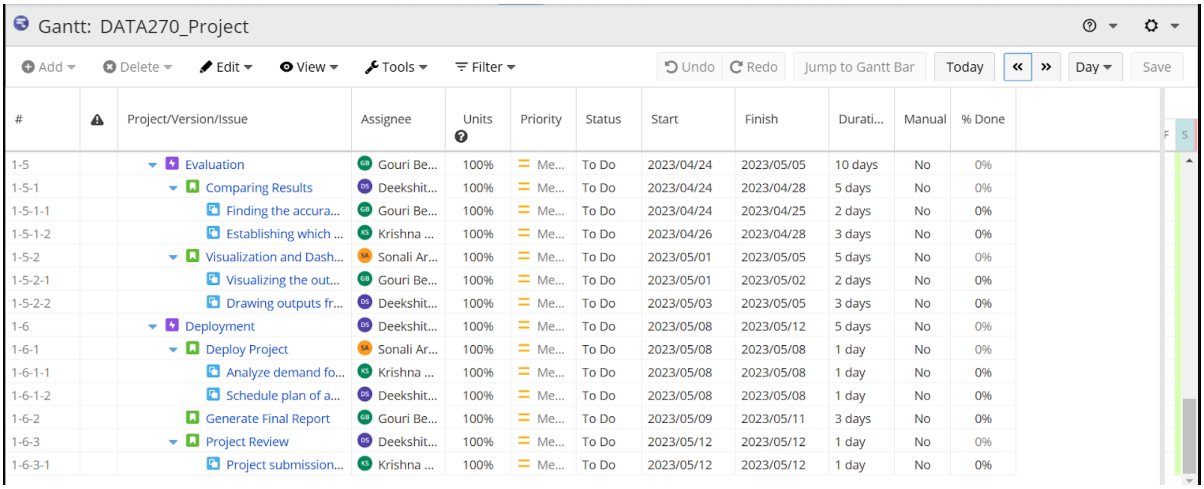


Figure B11. Gantt Chart explaining Model Deployment.



#	Project/Version/Issue	Assignee	Units	Priority	Status	Start	Finish	Durati...	Manual	% Done
1-5	Evaluation	Gouri Be...	100%	Me...	To Do	2023/04/24	2023/05/05	10 days	No	0%
1-5-1	Comparing Results	Deekshit...	100%	Me...	To Do	2023/04/24	2023/04/28	5 days	No	0%
1-5-1-1	Finding the accura...	Gouri Be...	100%	Me...	To Do	2023/04/24	2023/04/25	2 days	No	0%
1-5-1-2	Establishing which ...	Krishna ...	100%	Me...	To Do	2023/04/26	2023/04/28	3 days	No	0%
1-5-2	Visualization and Dash...	Sonali Ar...	100%	Me...	To Do	2023/05/01	2023/05/05	5 days	No	0%
1-5-2-1	Visualizing the out...	Gouri Be...	100%	Me...	To Do	2023/05/01	2023/05/02	2 days	No	0%
1-5-2-2	Drawing outputs fr...	Deekshit...	100%	Me...	To Do	2023/05/03	2023/05/05	3 days	No	0%
1-6	Deployment	Deekshit...	100%	Me...	To Do	2023/05/08	2023/05/12	5 days	No	0%
1-6-1	Deploy Project	Sonali Ar...	100%	Me...	To Do	2023/05/08	2023/05/08	1 day	No	0%
1-6-1-1	Analyze demand fo...	Krishna ...	100%	Me...	To Do	2023/05/08	2023/05/08	1 day	No	0%
1-6-1-2	Schedule plan of a...	Deekshit...	100%	Me...	To Do	2023/05/08	2023/05/08	1 day	No	0%
1-6-2	Generate Final Report	Gouri Be...	100%	Me...	To Do	2023/05/09	2023/05/11	3 days	No	0%
1-6-3	Project Review	Deekshit...	100%	Me...	To Do	2023/05/12	2023/05/12	1 day	No	0%
1-6-3-1	Project submission...	Krishna ...	100%	Me...	To Do	2023/05/12	2023/05/12	1 day	No	0%

Figure B12. Effort Estimate of Model Deployment.