PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

SUMMARY

Batch details	DSE- FT-Chennai, May 2024 – G8
Team members	Dharaneesh K P Rajasekaren S K Surya T J Vishnupriya K
Domain of Project	Supply chain management
Proposed project title	Machine Learning-Driven Demand Forecasting and Supply Optimization for Instant Noodles Distribution
Group Number	8
Team Leader	Surya T J
Mentor Name	Avanish Kumar Singh

Date: 03-12-2024

Avanish Kumar Singh

Signature of the Mentor

SURYA T J

Signature of the Team Leader

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

TABLE OF CONTENTS

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

1) ABSTRACT

A Fast Moving Consumer Goods (FMCG) company that recently ventured into the instant noodles market faces a significant challenge in balancing supply with regional demand. Currently, mismatches in demand and supply have led to stock shortages in
high-demand regions and excess inventory in low-demand areas, resulting in increased costs and operational inefficiencies. To address this issue, this project aims to develop a predictive model to accurately forecast the product weight required for each warehouse.
By utilizing advanced machine learning techniques and analyzing factors such as historical demand trends and regional distribution needs, the model will optimize supply levels to align more closely with actual demand. This data-driven approach is expected to mitigate
supply-demand imbalances, thereby reducing inventory losses and maximizing revenue. Once implemented, the model will provide actionable insights for more efficient supply chain management, positioning the company to reduce losses, increase profitability, and ensure a more agile response to demand fluctuations.

2) INTRODUCTION

In the highly competitive Fast Moving Consumer Goods (FMCG) sector, meeting consumer demand while managing supply chain efficiency is critical to maintaining profitability and market share. Two years ago, an FMCG company entered the instant noodles business, only to encounter significant supply-demand mismatches across various regions. High-demand areas frequently experience stockouts due to insufficient supply, while low-demand areas face overstocking issues, leading to increased inventory holding costs and product wastage. These imbalances not only strain the company's finances but also hinder its ability to meet customer expectations effectively.

Addressing this challenge requires a data-driven approach to predict and optimize the supply of instant noodles to each warehouse nationwide. By developing a predictive model that forecasts the appropriate product weight for each region, the company aims to align supply more closely with regional demand, reducing inventory losses and maximizing revenue potential. This project leverages machine learning techniques to analyze factors such as historical demand patterns, warehouse capacity, and distribution

constraints. Successfully implementing this model is expected to bridge supply-demand gaps, reduce excess inventory costs, and provide a scalable solution for dynamic supply allocation across diverse regions.

This project thus aims not only to address the immediate supply-demand mismatch but also to create a sustainable, adaptable framework for future growth in the instant noodles market. Through this initiative, the company seeks to enhance its operational efficiency, minimize financial losses, and ensure customer satisfaction across all markets.

3) PROPOSED BUSINESS PROBLEM STATEMENT

To address the supply-demand imbalance in its instant noodles business, the FMCG company aims to develop a predictive solution that will accurately forecast the optimal product weight required for each warehouse across various regions. This solution is intended to align supply with regional demand, reducing instances of stockouts in high-demand areas and overstocking in low-demand areas. By implementing a data-driven approach, the company seeks to minimize inventory holding costs, reduce product wastage, and improve revenue generation. This initiative will ultimately optimize the supply chain, enhancing both operational efficiency and customer satisfaction.

4) INDUSTRY REVIEW

4.1. INDUSTRY REVIEW

4.1.1. CURRENT PRACTICES

DemandForecasting:

Demand forecasting is a critical component in FMCG supply chains, aiming to predict product demand accurately to balance supply and demand. Currently, many FMCG companies utilize time-series analysis, statistical models, and increasingly, machine learning algorithms such as ARIMA, exponential smoothing, and more advanced models like LSTM and Prophet. These methods analyze historical sales, seasonal trends, and external factors to forecast demand and prevent both stockouts and overstocking.

Inventory Management and Optimization
In inventory management, practices like Just-in-Time (JIT) and Economic Order Quantity (EOQ) models are common. These aim to reduce excess stock while ensuring timely availability of products. In the FMCG industry, efficient inventory management systems monitor stock levels, expiration dates, and product turnover rates, especially for high-demand products like instant noodles, to reduce holding costs and prevent wastage. Advanced systems also integrate real-time data from sales channels to adjust inventory levels dynamically.

Distribution and Transportation Management
FMCG companies optimize transportation routes, warehouse locations, and distribution schedules to minimize costs and reduce delays. Many companies employ software that uses algorithms to calculate optimal routes and schedules, factoring in fuel costs, traffic, and delivery windows. Some larger FMCG companies are also implementing predictive models to identify potential delays and reroute deliveries proactively.

Supplier and Warehouse Management
Effective supplier management includes evaluating suppliers based on reliability, cost, and capacity. Warehouses are managed to maintain efficient space utilization and allow quick access to high-demand items. Many FMCG firms implement automated warehouse management systems (WMS) that use robotics, barcoding, and real-time data to track and manage inventory flows efficiently.

4.1.2. BACKGROUND RESEARCH

Predictive Analytics for Demand Forecasting
Research in predictive analytics has transformed demand forecasting in supply chains, with studies showing how machine learning and deep learning models can significantly enhance forecast accuracy. For example, models like XGBoost, Random Forest, and Neural Networks have been shown to effectively capture complex patterns in demand data, improving the alignment between forecasted and actual demand.

Inventory Optimization Techniques
Studies in inventory optimization suggest that combining machine learning models with traditional approaches (like ABC inventory classification or EOQ) helps in managing inventory levels efficiently. Research indicates that predictive models reduce holding costs by classifying products based on demand variability, thus helping companies keep

low-demand items in lower quantities while ensuring high-demand items are well-stocked.

 Application of Machine Learning in Supply Chain Allocation
Machine learning models for supply chain allocation consider multiple variables, such as demand fluctuations, warehouse capacities, and transportation costs. Research has shown that models like linear programming, reinforcement learning, and constraint-based optimization help achieve optimal product allocation across distribution centers. For instance, studies have demonstrated how companies can use these models to reallocate supplies in response to real-time sales data and regional demand changes.

 Supply Chain Resilience and Flexibility
With the recent focus on supply chain resilience, research has examined how flexible supply chains can respond to disruptions more effectively. Studies suggest that building adaptive models, capable of responding to demand shifts and external shocks, helps companies mitigate risks associated with demand variability. This approach, involving deep learning and real-time data integration, is particularly beneficial for high-demand FMCG products, enabling companies to adjust their supply strategies dynamically.

 Case Studies and Industry Benchmarks
Case studies on companies such as Unilever, Nestlé, and P&G have shown the benefits of integrating predictive analytics with supply chain management. These companies have successfully used AI-driven models to forecast demand and optimize distribution, achieving significant reductions in inventory costs and improvements in service levels. Such case studies highlight best practices and serve as benchmarks for other FMCG companies aiming to optimize their supply chains.

4.2. LITERATURE SURVEY

- **Title:** An Integrated Framework for Inventory Management and Demand Forecasting in FMCG Sector
  Authors: Shankar, R., & Vikas, S.
  Summary: This article proposes an integrated framework that combines inventory management with demand forecasting specifically for the FMCG sector. The authors highlight the importance of synchronizing these functions to enhance overall supply chain performance. By effectively aligning inventory levels with demand forecasts, the framework aims to minimize excess inventory costs, reduce stockouts, and

improve responsiveness to market changes. The findings suggest that an integrated approach can lead to significant cost reductions and improved efficiency in FMCG operations.

- **Title**: A Study of Demand Forecasting in FMCG Supply Chains
  Authors: Chien, C. F., & Chen, J. C.
  Summary: This study investigates various demand forecasting models in the FMCG supply chain. The authors emphasize the importance of accurate forecasting to reduce inventory costs and improve service levels, discussing both qualitative and quantitative methods. The paper highlights the challenges associated with demand variability and suggests strategies for enhancing forecasting accuracy in FMCG operations.

- **Title**: Demand Forecasting Methods and the Potential of Machine Learning in the FMCG Retail Industry
  Authors: Aichner, T., & Santa, V.
  Summary: This paper reviews various demand forecasting methods used in the FMCG retail industry, focusing on the potential applications of machine learning techniques. The authors argue that machine learning can significantly enhance demand prediction accuracy, thereby optimizing inventory management and reducing costs associated with stockouts and overstocking.

- **Title**: The Impact of Data-Driven Approaches on Demand Forecasting Accuracy in the FMCG Sector
  Authors: Wang, Y., & Zhang, D.
  Summary: This paper examines the influence of data-driven techniques, including machine learning and big data analytics, on demand forecasting accuracy in the FMCG industry. The findings indicate that these approaches can significantly enhance forecasting performance and inventory optimization, providing practical implications for FMCG companies facing supply-demand mismatches.

- **Title**: The Forecasting Performance of Croston's Method
  Authors: Syntetos, A. A., & Boylan, J. E.
  Summary: This paper evaluates Croston’s method for forecasting intermittent demand, which is particularly relevant in the FMCG sector where demand can

fluctuate. The authors compare this method with traditional forecasting techniques, highlighting its advantages for stock management and operational efficiency.

- **Title**: Design and Control of Order Picking Operations in Warehouses
  Authors: De Koster, R., Le-Duc, T., & Roodbergen, K. J.
  Summary: This review focuses on the design and control of order-picking operations, identifying it as the most labor-intensive and costly activity in warehouse management, accounting for approximately 55% of total warehouse operating expenses. The authors discuss how delays in order picking can lead to customer dissatisfaction and increased operational costs, affecting the overall supply chain. They provide an overview of decision issues in manual order-picking processes and highlight performance improvement measures, layout design, routing methods, and storage assignment, indicating the need for further research in these areas.

- **Title**: Research on Warehouse Operations Planning Problems
  Authors: Gu, J., Goetschalckx, M., & McGinnis, L. F.
  Summary: This article extensively reviews warehouse operations planning problems, categorizing issues based on fundamental warehouse functions, including receiving, storage, order picking, and shipping. The authors emphasize decision support models and solution algorithms, focusing primarily on the analysis of storage systems rather than their synthesis. The review highlights the need for comprehensive approaches to improve warehouse operation efficiency.

- **Title**: A Cost Optimization Strategy for a Single Warehouse Multi-Distributor Vehicle Routing System
  Authors: Nidhi & Anil.
  Summary: This study discusses the significant costs associated with the delivery phase in supply chain and logistics management. The authors develop a simulation model to achieve cost optimization within a single warehouse multi-distributor routing system under stochastic conditions, specifically in an LPG bottling and distribution plant. They utilize performance measures encompassing both quantifiable and non-quantifiable cost factors and propose a dispatch rule-based allocation strategy to replace the existing random method.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

DATASET DESCRIPTION AND DOMAIN

1 . DATA DICTIONARY

The data dictionary is a detailed description of each feature in the dataset, which aids in understanding the context, operational aspects, and structural details of the warehouses. Below is a breakdown of the main variables in the dataset:

- **Ware_house_ID**: Unique identifier for each warehouse, helping differentiate between individual warehouse data.
- **WH_Manager_ID**: Identifier for the manager overseeing each warehouse, potentially relevant to understanding human factors in warehouse operations.
- **zone** and **WH_regional_zone**: Geographic zones where each warehouse is located, which may correlate with demand patterns and logistics costs.
- **num_refill_req_l3m**: Count of refilling requests received by the warehouse in the last three months. A high count might indicate a high-demand area or poor inventory forecasting.
- **transport_issue_l1y**: Frequency of transport issues in the past year, which could influence product availability and lead times.
- **Competitor_in_mkt**: Number of competitors in the local market, which might impact demand for the company's product.
- **retail_shop_num**: Number of retail shops sourcing from the warehouse, potentially influencing distribution volume.
- **wh_owner_type**: Ownership status of the warehouse (company-owned or rented), possibly affecting operational flexibility and cost structure.
- **distributor_num**: Number of distributors associated with the warehouse, affecting the ease of reaching end customers.
- **flood_impacted**: Whether the warehouse is in a flood-prone area, which could disrupt supply chains.

- **flood_proof**: Indicates if the warehouse is protected against flooding, an important operational factor in flood-prone regions.
- **electric_supply**: Whether the warehouse has reliable electric supply and backup, essential for product preservation.
- **dist_from_hub**: Distance of the warehouse from the central production hub, impacting logistics costs and lead times.
- **workers_num**: Number of workers in the warehouse, which could impact the warehouse's operational efficiency.
- **wh_est_year**: Year the warehouse was established, possibly indicative of the infrastructure and facilities available.
- **storage_issue_reported_l3m**: Number of storage-related issues reported in the last three months, which may impact stock quality.
- **govt_check_l3m**: Frequency of government inspections, potentially affecting operational compliance.
- **temp_reg_mach**: Indicates if the warehouse has temperature regulation machinery, crucial for temperature-sensitive products.
- **approved_wh_govt_certificate**: Type of government certification for the warehouse, affecting the warehouse's operational legitimacy.
- **wh_breakdown_l3m**: Number of breakdowns in the past three months, potentially affecting reliability and product availability.
- **product_wg_ton**: Weight of the product shipped, which serves as the target variable in the forecasting model.

2. VARIABLE CATEGORIZATION

Understanding the types of variables in the dataset helps in deciding the preprocessing steps and model selection.

- **Numeric Variables**: Variables that contain numerical values, suitable for statistical analysis and predictive modeling.
    - num_refill_req_l3m
    - transport_issue_l1y

- Competitor_in_mkt
- retail_shop_num
- distributor_num
- dist_from_hub
- workers_num
- wh_est_year
- storage_issue_reported_l3m
- temp_reg_mach
- wh_breakdown_l3m
- govt_check_l3m
- product_wg_ton
- Count : 16

- **Categorical Variables**: Variables that contain categories or classifications, often requiring encoding before feeding them into a machine learning model.
  - Ware_house_ID
  - WH_Manager_ID
  - Location_type
  - WH_capacity_size
  - zone
  - WH_regional_zone
  - wh_owner_type
  - approved_wh_govt_certificate
  - flood_impacted
  - flood_proof
  - electric_supply
  - count : 8

3. PRE-PROCESSING DATA ANALYSIS

A crucial step before modeling is to assess the data quality, including any issues with missing values, redundant columns, and outliers. Here's an overview of the common preprocessing steps:

- **Missing/Null Values**: A count of missing or null values for each column should be obtained. For critical columns, imputation strategies may be required, while columns with high missing rates could be considered for removal if they provide limited value.
- **Redundant Columns**: Certain columns may be duplicates, irrelevant, or have too many unique values (e.g., WH_Manager_ID, which may not directly affect product demand). Identifying and removing these columns can reduce noise in the dataset.
- **Outlier Detection**: Outliers in numeric fields (e.g., dist_from_hub, workers_num) might skew model predictions. Box plots or interquartile range (IQR) methods can help identify and manage these outliers.

4. ALTERNATE DATA SOURCES

Incorporating additional data can enhance the predictive capability of the model. Here are some recommended supplementary data sources:

- **Weather Data**: Historical weather conditions for each warehouse's region could provide valuable insights, particularly for flood_impacted warehouses. This information could help predict transport issues and supply disruptions.
- **Market Trends**: Demand forecasting could be improved by incorporating consumer trends or economic indicators, such as regional sales growth rates or GDP. A variable representing market demand forecasts would allow more proactive planning.
- **Supply Chain Data**: Information on supplier reliability or lead times for each warehouse region could further refine demand predictions by accounting for external supply variability. For example, adding a lead_time_days variable would capture expected shipment delays.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

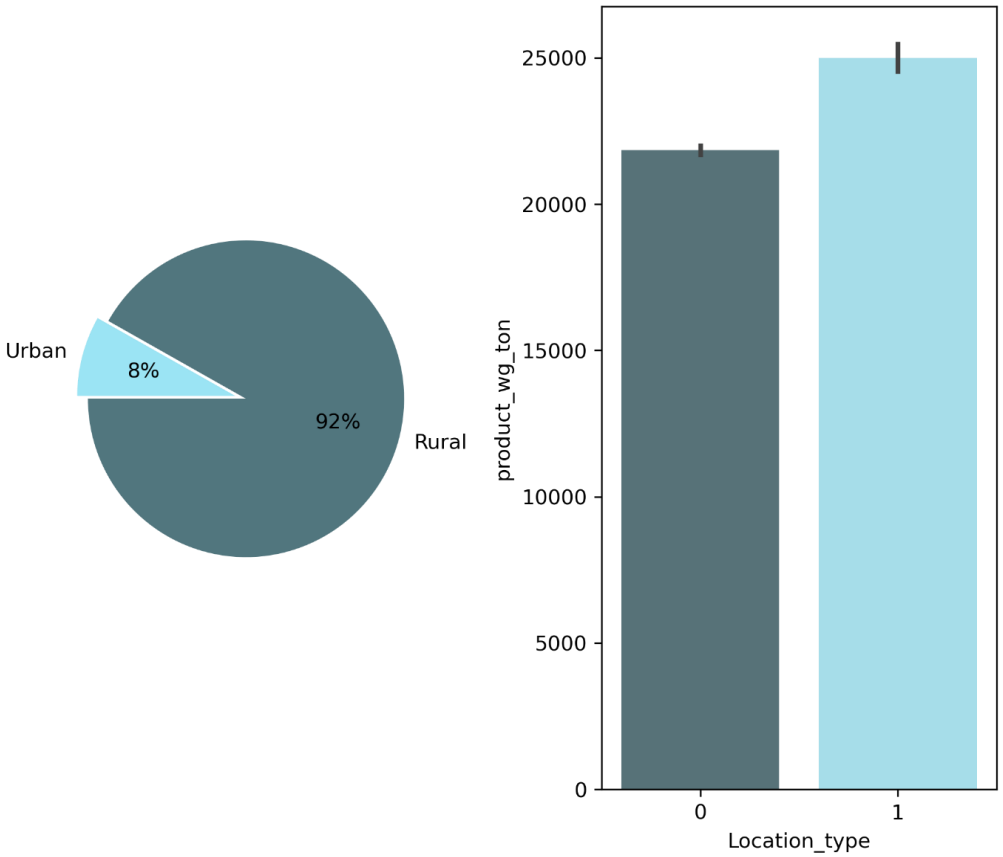
5. PROJECT JUSTIFICATION

This project holds considerable value across commercial, academic, and operational domains, as described below:

- **Project Statement**: The aim is to leverage machine learning to optimize demand forecasting and supply management for an FMCG company's instant noodles distribution, balancing supply across warehouses to meet regional demands efficiently.
- **Complexity**: The project involves handling high-dimensional data with both numerical and categorical features, identifying key operational factors, and using advanced machine learning techniques to generate reliable predictions. Model accuracy is crucial, given the financial impact of over- or under-supplying different regions.
- **Outcome and Value**:
    - **Commercial Value**: The model's application directly supports cost reduction by minimizing inventory costs and waste while avoiding stockouts in high-demand areas. This enhances profitability and supply chain resilience, crucial for the company's financial health.
    - **Academic Value**: The project demonstrates the practical application of machine learning in supply chain optimization, making it a valuable case study for logistics, data science, and operations management fields.
    - **Social Value**: An optimized supply chain reduces waste, leading to a more sustainable business model. Furthermore, by ensuring product availability in high-demand areas, the company better serves its customer base, enhancing customer satisfaction and accessibility.

6) DATA EXPLORATION (EDA)

6.1 Relationship between variables



Observations:

Pie Chart:

- **Distribution of Location Type:** The pie chart shows that 92% of the data points belong to the "Rural" category, while only 8% belong to the "Urban" category. This indicates that the dataset is heavily skewed towards rural locations.

Bar Plot:

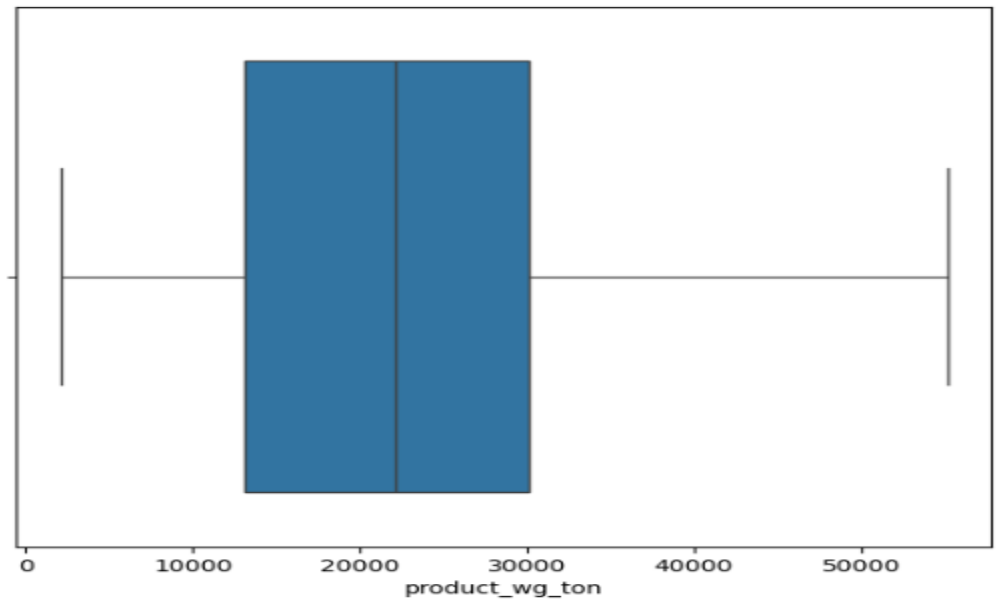
- **Product Weight by Location Type:** The bar plot shows that the average product_wg_ton is significantly higher for rural locations compared to urban locations. The error bars represent the standard deviation, indicating the variability within each category.

Relationship between Variables:

- **Location Type and Product Weight:** There is a clear relationship between the location type and the average product weight. Rural locations tend to have products with significantly higher weights compared to urban locations. This difference could be due to various factors, such as different product types, transportation methods, or market dynamics.

B. DISTRIBUTION OF VARIABLES:

product_wg_ton Distribution using BOXPLOT:



Observations:

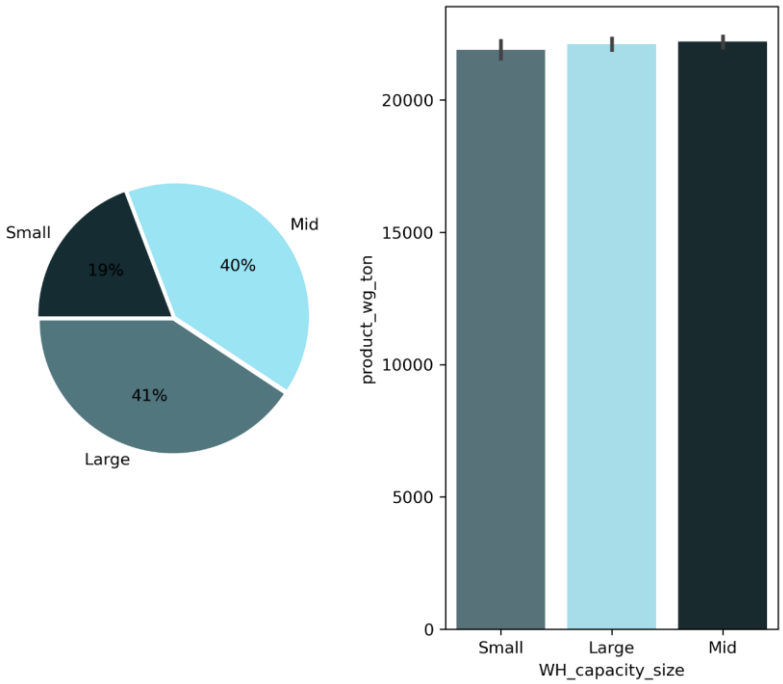
- **Distribution:** The distribution of product_wg_ton appears to be right-skewed. This is indicated by the longer whisker on the right side of the boxplot.
- **Central Tendency:**
  - **Median:** The median value lies around 25,000.
  - **Mean:** The mean is likely to be slightly higher than the median due to the right-skewness.

- **Spread:**
    - **Range:** The range seems to be approximately 50,000.
    - **Interquartile Range (IQR):** The IQR, represented by the box, is approximately 15,000.
    - **Outliers:** There are no visible outliers in this dataset.

POSSIBLE INFERENCES:

1. **Product Weight:** The product_wg_ton likely represents the weight of a product in tons. The data suggests that the majority of products in the dataset have weights between 10,000 and 30,000 tons.
2. **Product Variation:** The right-skewness indicates that there are a few products with significantly higher weights compared to the majority. This could be due to a variety of factors, such as product type, size, or specific use cases.
3. **Data Quality:** The absence of outliers suggests that the data is relatively clean and free from extreme values that might distort the analysis.

Product Weight Distribution by Warehouse Capacity Size:



Observations:

Pie Chart:

- **Distribution of Warehouse Size:** The pie chart shows that 41% of the warehouses are "Large," 40% are "Mid," and 19% are "Small." This indicates a relatively balanced distribution across the different warehouse sizes.

Bar Plot:

- **Product Weight by Warehouse Size:** The bar plot shows that the average product_wg_ton is similar across all warehouse sizes. The error bars represent the standard deviation, indicating the variability within each category.

Relationship between Variables:

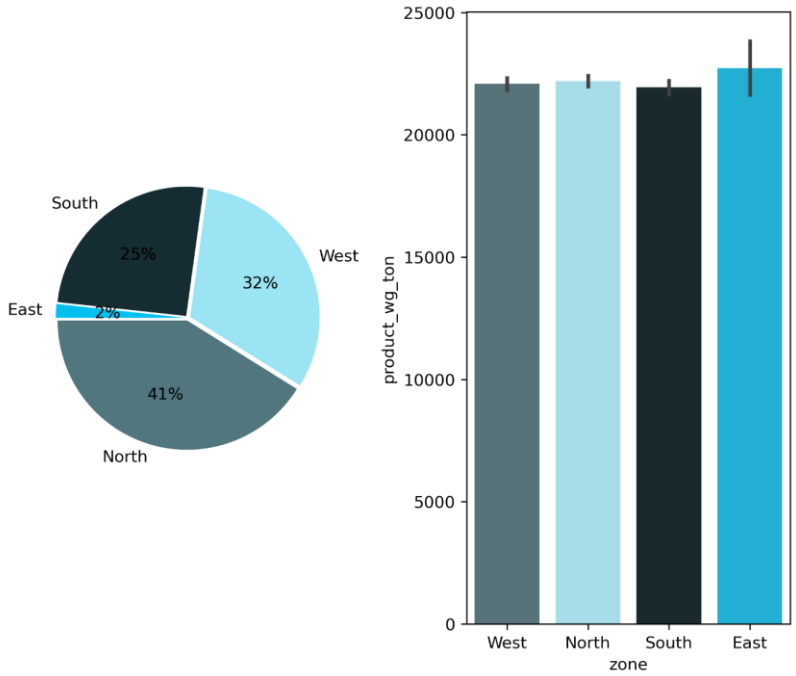
- **Warehouse Size and Product Weight:** There doesn't seem to be a strong relationship between the warehouse size and the average product weight. The similar average weights across different sizes suggest that warehouse size might not be a major factor influencing the weight of products stored.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Inferences for Documentation:

1. **Warehouse Size Distribution:** The dataset contains a relatively balanced distribution of small, medium, and large warehouses.
2. **Product Weight:** The average weight of products stored in different warehouse sizes is comparable, suggesting that warehouse size doesn't significantly impact the weight distribution.

Product Weight Distribution by Zone:



Data Visualization:

The provided plot illustrates the distribution of product weight (in tons) across four geographic zones: West, North, South, and East. The visualization employs a combination of a pie chart and a bar chart to provide a comprehensive view of the data.

Key Observations:

1. **Product Weight Distribution:** The pie chart reveals the proportion of total product weight attributed to each zone. The West zone dominates with 41%,

followed by the North zone with 32%. The South and East zones account for 25% and 2%, respectively.

2. **Zone-wise Product Weight:** The bar chart offers a more granular view of the product weight distribution across zones. It confirms the dominance of the West zone, followed by the North zone. The South zone has a significantly higher product weight compared to the East zone.

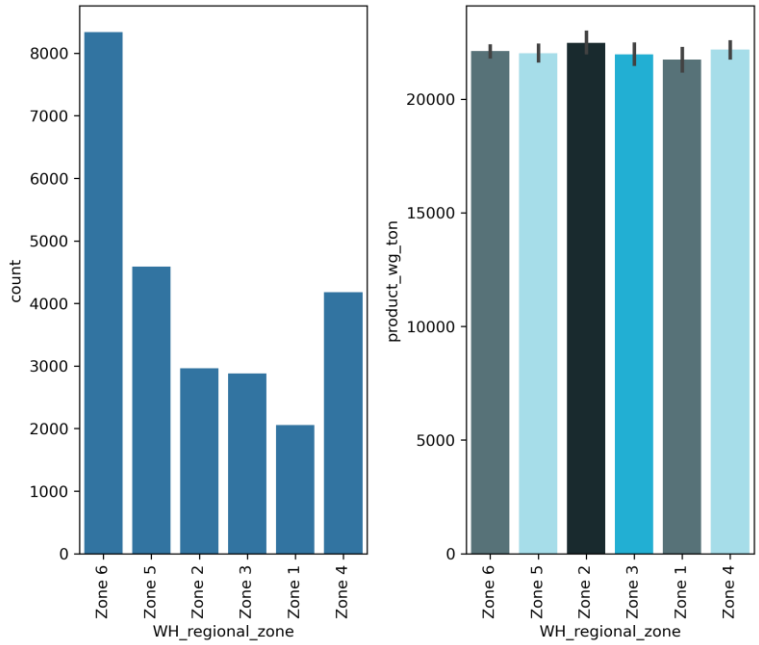
Insights and Implications:

- **Regional Disparities:** The substantial difference in product weight across zones suggests potential regional disparities in production, consumption, or logistics. The West zone, with its higher share of product weight, may have a larger industrial or manufacturing base compared to other zones.

- **Zone-Specific Strategies:** Understanding the regional variations in product weight can inform targeted strategies for production, distribution, and inventory management. For instance, the West zone might require more robust supply chain infrastructure to handle the higher volume of products.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Zone-wise Analysis of Product Count and Weight:



Data Visualization:

The provided plot illustrates the distribution of product weight (in tons) across six geographic zones: Zone 1, Zone 2, Zone 3, Zone 4, Zone 5, and Zone 6. The visualization employs a combination of a bar chart and a histogram to provide a comprehensive view of the data.

Key Observations:

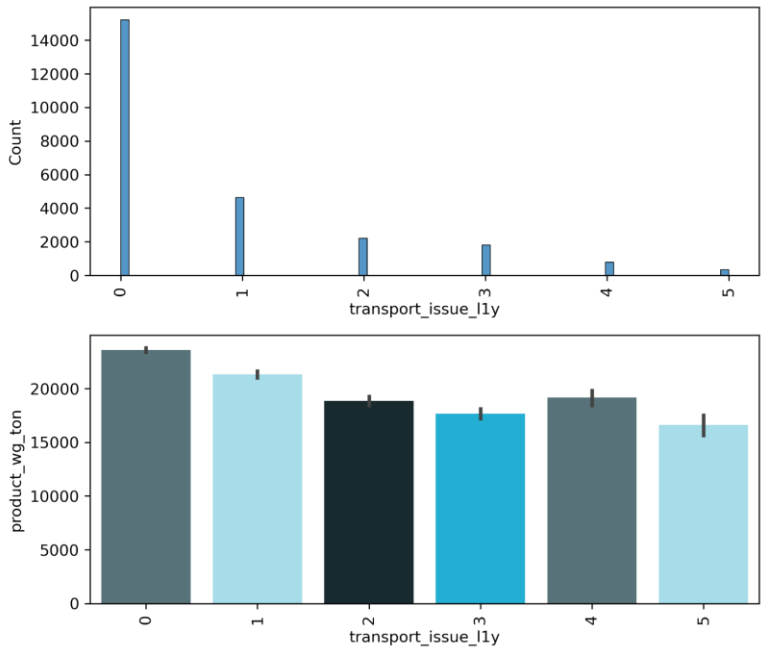
1. **Zone-wise Product Weight:** The bar chart reveals the product weight distribution across zones. Zone 6 has the highest product weight, followed by Zone 5. Zone 4 has the lowest product weight.
2. **Product Count Distribution:** The histogram shows the number of products in each zone. Zone 6 has the highest number of products, while Zone 4 has the lowest.

Insights and Implications:

- **Regional Disparities:** The significant differences in product weight and product count across zones suggest potential regional disparities in production, consumption, or logistics. Zone 6, with its high product weight and count, might be a key production or consumption hub.

- **Zone-Specific Strategies:** Understanding the regional variations in product weight and count can inform targeted strategies for production, distribution, and inventory management. For instance, Zone 6 might require more robust supply chain infrastructure to handle the higher volume of products.

Impact of Transport Issues on Product Weight:



Data Visualization:

The provided plot illustrates the relationship between the number of transport issues (transport_issue_1ly) and the product weight (product_wg_ton). The visualization employs a combination of a histogram and a bar chart to provide a comprehensive view of the data.

Key Observations:

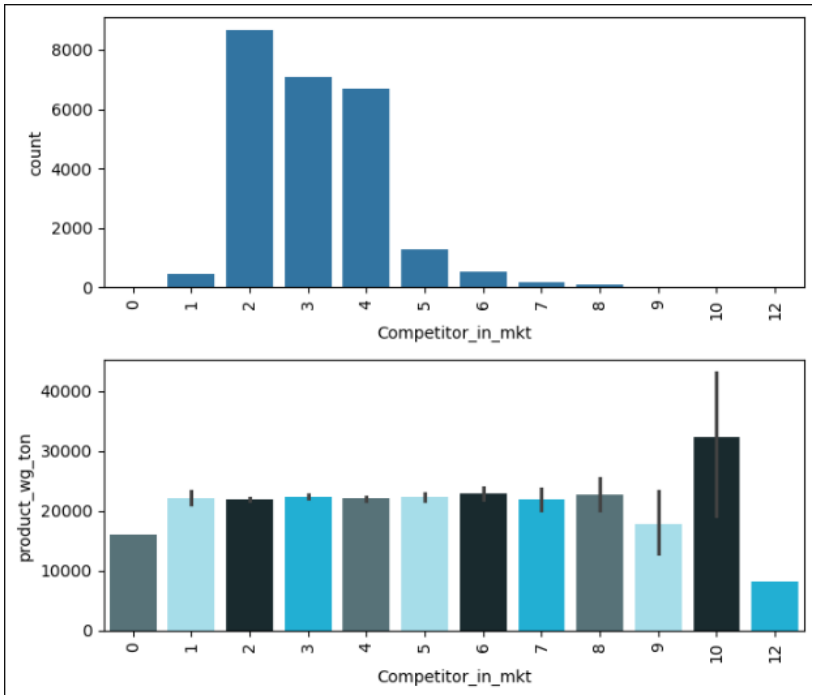
1. **Transport Issue Frequency:** The histogram shows the distribution of the number of transport issues. The majority of products have experienced 0 or 1 transport issue, with a significant drop in frequency for higher issue counts.
2. **Product Weight by Transport Issues:** The bar chart reveals the average product weight for different levels of transport issues. Interestingly, the highest

average product weight is observed for products with 0 transport issues. As the number of transport issues increases, the average product weight tends to decrease.

Insights and Implications:

- **Transport Issues and Product Weight:** There appears to be a negative correlation between the number of transport issues and the average product weight. This could indicate that products with higher weights might be more susceptible to transport-related challenges.

Analyzing the Impact of Competitor Presence on Product Count and Weight:



Data Visualization:

The provided plot illustrates the relationship between the number of competitors in the market (Competitor_in_mkt) and the product weight (product_wg_ton). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

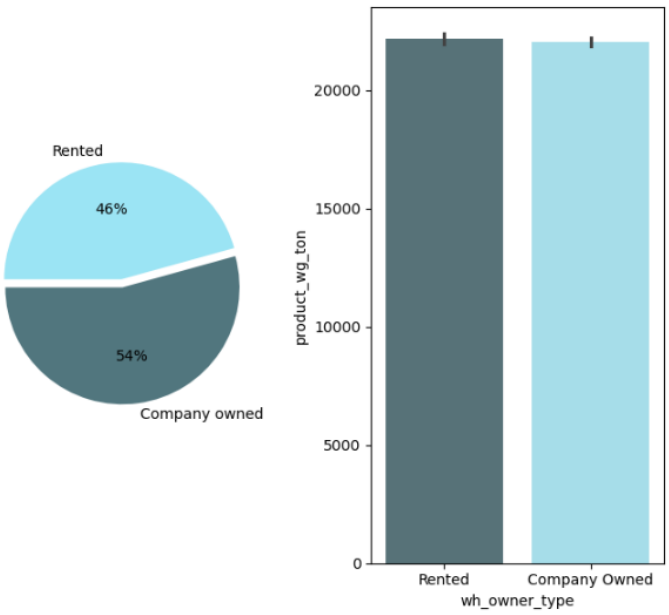
Key Observations:

1. **Competitor Distribution:** The bar chart shows the distribution of the number of competitors in the market. The majority of products are in markets with 0 or 1 competitor, with a gradual decrease in frequency as the number of competitors increases.
2. **Product Weight by Competitor Count:** The bar plot reveals the average product weight for different levels of competitor presence. Interestingly, the highest average product weight is observed for markets with 10 competitors. The average product weight generally increases as the number of competitors increases, with a slight dip around 4-6 competitors.

Insights and Implications:

- **Competitor Presence and Product Weight:** There appears to be a positive correlation between the number of competitors in the market and the average product weight. This could indicate that markets with more competition might drive companies to produce larger or heavier products to gain a competitive edge.

Analysis of Product Weight by Warehouse Ownership Type:



PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Data Visualization:

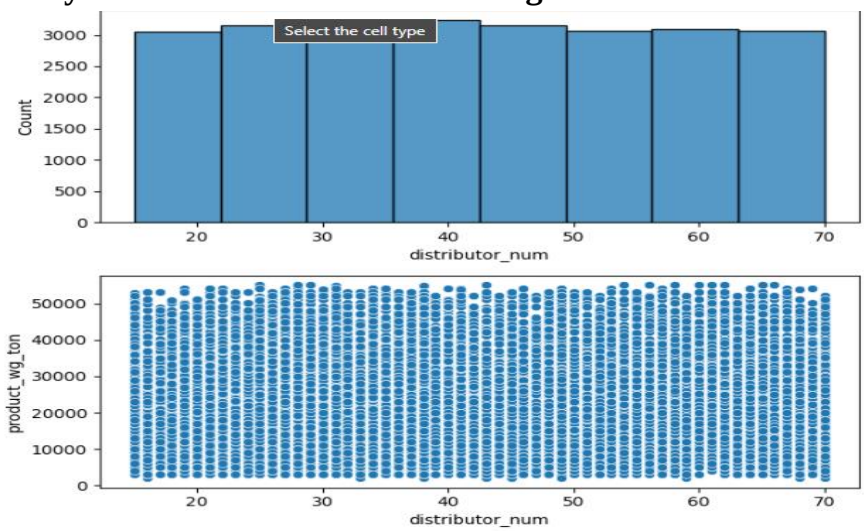
The provided plot illustrates the distribution of product weight (in tons) across two warehouse ownership types: Rented and Company Owned. The visualization employs a combination of a pie chart and a bar chart to provide a comprehensive view of the data.

Key Observations:

1. **Ownership Distribution:** The pie chart reveals that 46% of the warehouses are rented, while 54% are company-owned.
2. **Product Weight by Ownership:** The bar chart shows the average product weight for rented and company-owned warehouses. Interestingly, there is a slight difference in average product weight between the two types, with rented warehouses having a slightly higher average weight.

Insights and Implications:

- **Ownership and Product Weight:** While the difference in average product weight is not significant, it suggests that there might be subtle variations in the types of products stored in rented versus company-owned warehouses.
- **Analysis of Product Weight and Distributor Count:**



PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different distributor numbers (distributor_num). The visualization employs a combination of a histogram and a scatter plot to provide a comprehensive view of the data.

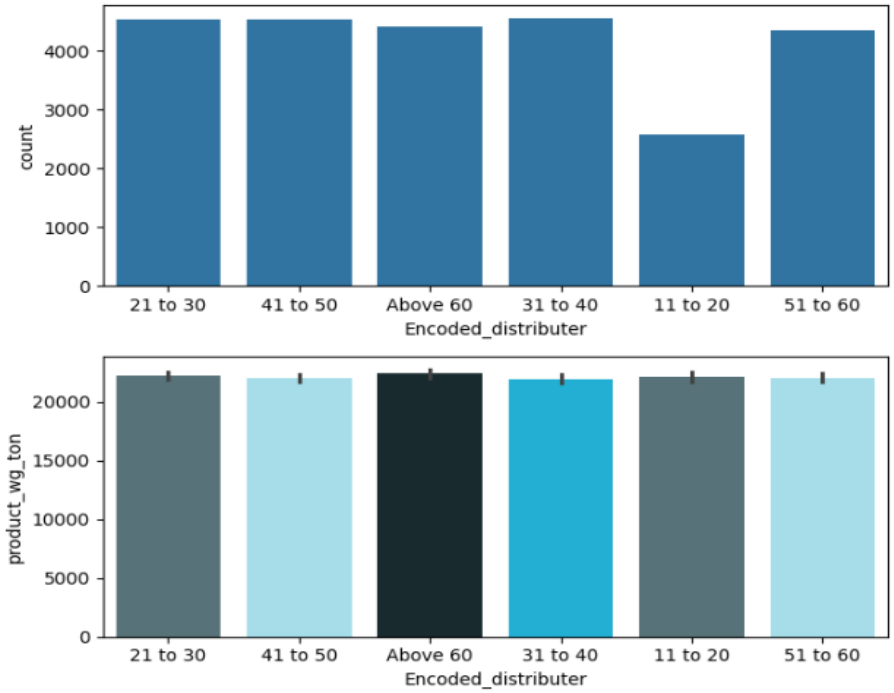
Key Observations:

1. **Distributor Distribution:** The histogram shows a relatively uniform distribution of distributor numbers, with a slight peak around the 50-60 range. This suggests that the data is fairly evenly spread across different distributors.
2. **Product Weight Distribution:** The scatter plot reveals a wide range of product weights across different distributors. There appears to be a clustering of data points around the 30,000-40,000ton range, indicating that a significant portion of products fall within this weight category. However, there are also products with much higher and lower weights.

Insights and Implications:

- **Distributor Diversity:** The uniform distribution of distributor numbers suggests that the data includes a diverse range of distributors, potentially representing different regions, market segments, or product categories.
- **Product Weight Variation:** The wide range of product weights across distributors indicates that there is significant variation in the types of products being distributed. This could be due to factors such as product size, material density, or specific product categories.

Analysis of Product Weight and Encoded Distributor Distribution:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different encoded distributor categories (Encoded_distributor). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

Key Observations:

1. **Distributor Distribution:** The bar chart shows a relatively even distribution of products across the different encoded distributor categories. This suggests that the data is fairly evenly spread across these categories.
2. **Product Weight Distribution:** The bar plot reveals a relatively consistent average product weight across the different encoded distributor categories. There is a slight variation in the average weight, with some categories having slightly higher or lower weights.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Insights and Implications:

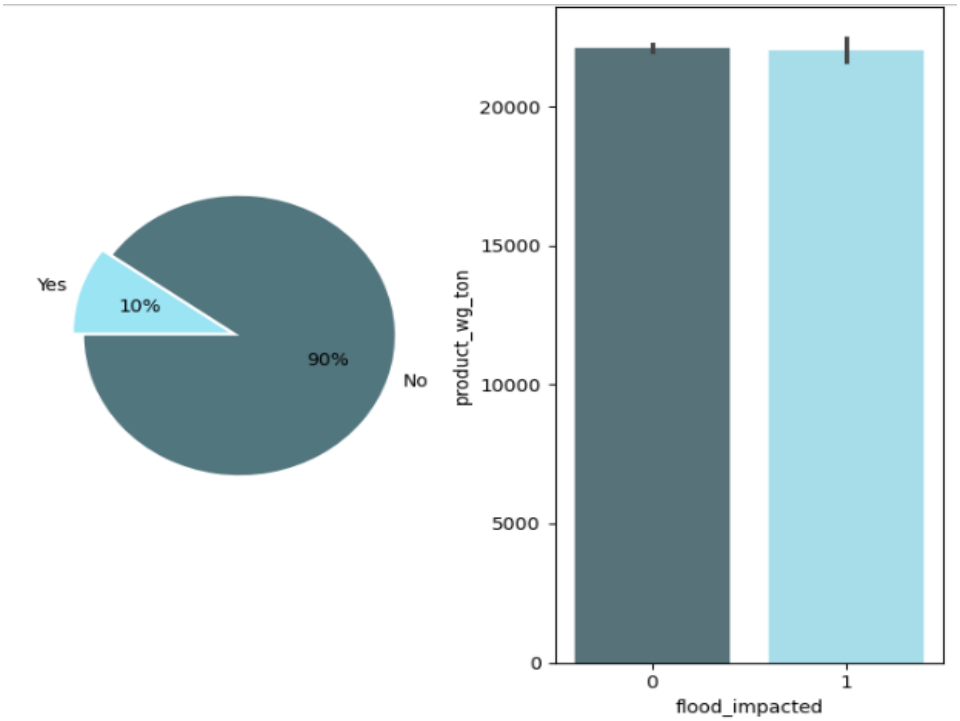
- **Even Distribution:** The even distribution of products across the encoded distributor categories suggests that the data is representative of a diverse range of distributors.

- **Consistent Product Weight:** The consistent average product weight across categories indicates that the encoded distributor categories might not have a significant impact on the overall weight of products.

- **Potential Factors:** Several factors could contribute to the observed product weight distribution:
    - **Product Type:** Different product types, such as bulk commodities or packaged goods, can have varying weights.
    - **Market Demand:** The demand for different product types can influence the distribution of product weights.
    - **Logistics and Transportation:** The efficiency of logistics and transportation networks can impact the feasible product weights for different distributors.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Analysis of Product Weight and Flood Impact:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across two flood impact categories: Yes and No. The visualization employs a combination of a pie chart and a bar chart to provide a comprehensive view of the data.

Key Observations:

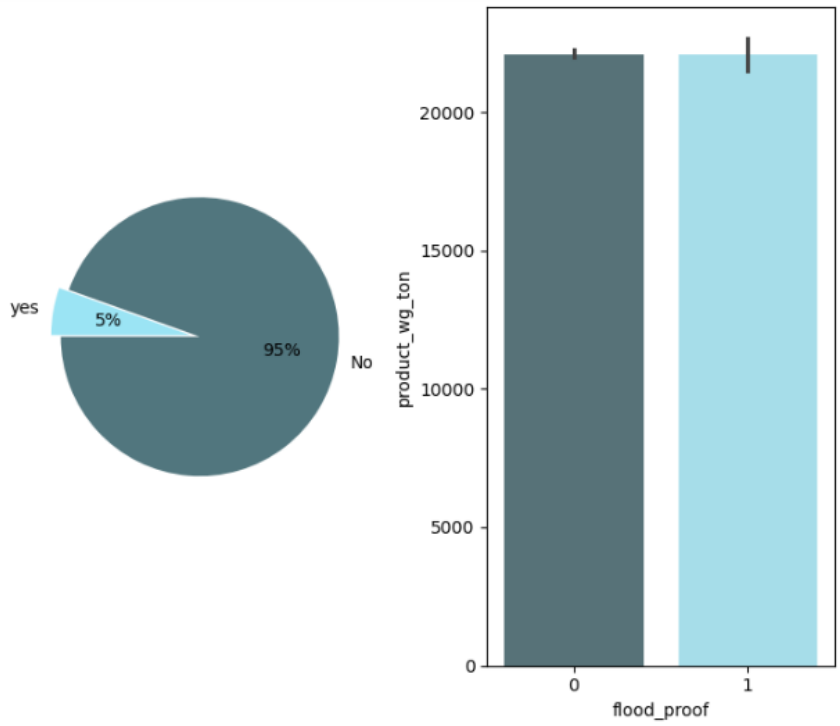
1. **Flood Impact Distribution:** The pie chart reveals that 90% of the products were not impacted by floods, while only 10% were affected.
2. **Product Weight by Flood Impact:** The bar chart shows the average product weight for both flood-impacted and non-impacted products. Interestingly, there is a slight difference in average product weight, with flood-impacted products having a slightly higher average weight.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Insights and Implications:

- **Flood Impact and Product Weight:** While the difference in average product weight is not significant, it suggests that there might be a subtle relationship between flood impact and product weight. Flood-impacted products might be associated with specific product types or storage locations that are more vulnerable to flood damage.

Analysis of Product Weight and Flood-Proofing:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across two flood-proofing categories: Yes and No. The visualization employs a combination of a pie chart and a bar chart to provide a comprehensive view of the data.

Key Observations:

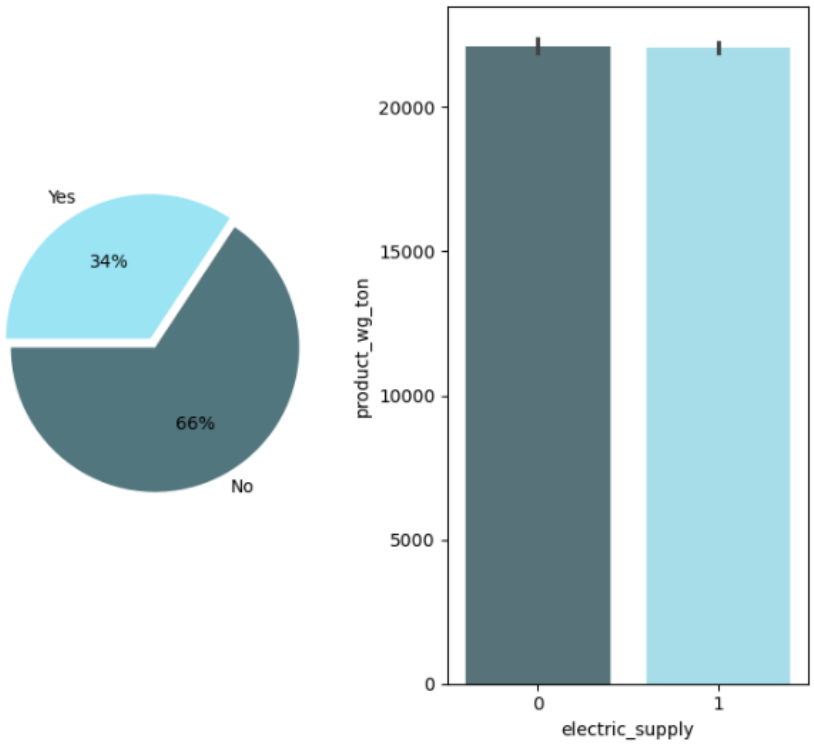
1. **Flood-Proofing Distribution:** The pie chart reveals that 95% of the products are not flood-proof, while only 5% are flood-proof.
2. **Product Weight by Flood-Proofing:** The bar chart shows the average product weight for both flood-proof and non-flood-proof products. Interestingly, there is a slight difference in average product weight, with flood-proof products having a slightly higher average weight.

Insights and Implications:

- **Flood-Proofing and Product Weight:** While the difference in average product weight is not significant, it suggests that there might be a subtle relationship between flood-proofing and product weight. Flood-proof products might be associated with specific product types or storage locations that are more susceptible to flood damage.

- **Potential Factors:** Several factors could contribute to this slight difference:
    - **Product Type:** Certain product types might be more susceptible to flood damage, leading to higher average weights for flood-proof products.
    - **Storage Location:** Products stored in flood-prone areas might be more likely to be flood-proofed, and these areas might have specific product storage characteristics.
    - **Cost Considerations:** The cost of flood-proofing might influence the decision to protect specific products, leading to a selection bias in the data.

Analysis of Product Weight and Electric Supply:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across two electric supply categories: Yes and No. The visualization employs a combination of a pie chart and a bar chart to provide a comprehensive view of the data.

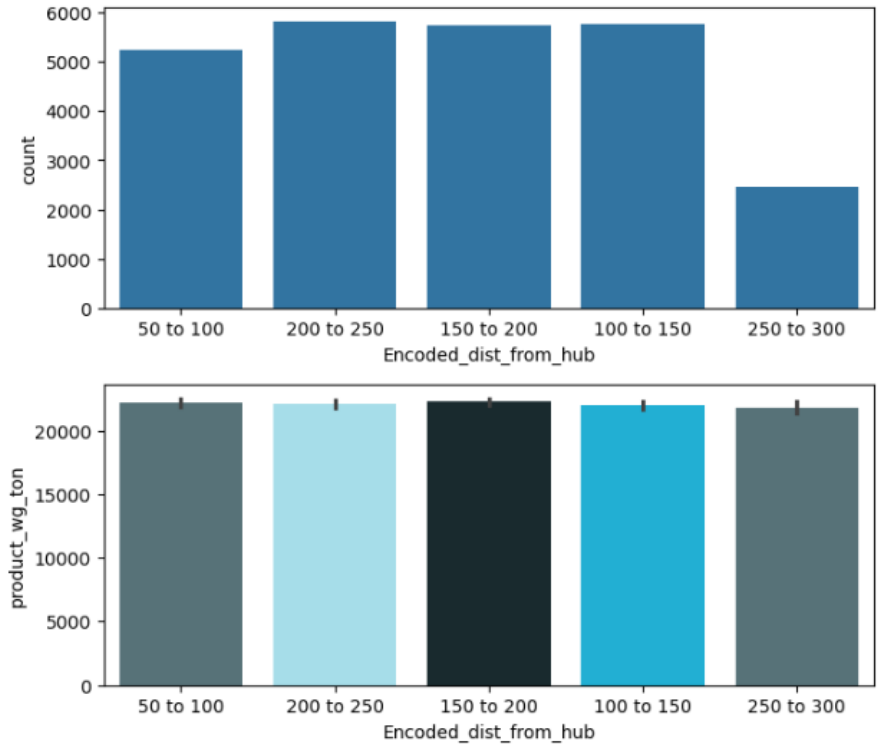
Key Observations:

1. **Electric Supply Distribution:** The pie chart reveals that 66% of the products are located in areas with electric supply, while 34% are in areas without electric supply.
2. **Product Weight by Electric Supply:** The bar chart shows the average product weight for both areas with and without electric supply. Interestingly, there is a slight difference in average product weight, with areas without electric supply having a slightly higher average weight.

Insights and Implications:

- **Electric Supply and Product Weight:** While the difference in average product weight is not significant, it suggests that there might be a subtle relationship between electric supply and product weight. Areas without electric supply might have specific product types or storage requirements that influence the average weight.

Analysis of Product Weight and Encoded Distance from Hub:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different encoded distance from hub categories (Encoded_dist_from_hub). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

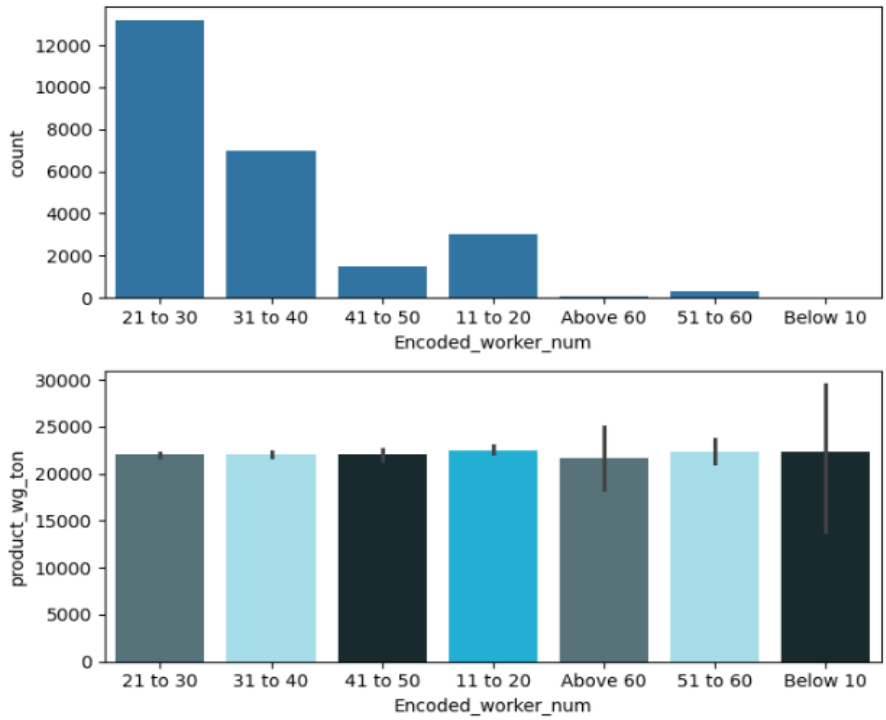
Key Observations:

1. **Distance Distribution:** The bar chart shows a relatively even distribution of products across the different encoded distance categories. This suggests that the data is fairly evenly spread across these distance ranges.
2. **Product Weight Distribution:** The bar plot reveals a relatively consistent average product weight across the different encoded distance categories. There is a slight variation in the average weight, with some categories having slightly higher or lower weights.

Insights and Implications:

- **Even Distribution:** The even distribution of products across the encoded distance categories suggests that the data is representative of a diverse range of distances from the hub.
- **Consistent Product Weight:** The consistent average product weight across categories indicates that the distance from the hub might not have a significant impact on the overall weight of products.
- **Potential Factors:** Several factors could contribute to the observed product weight distribution:
    - **Product Type:** Different product types, such as bulk commodities or packaged goods, can have varying weights.
    - **Market Demand:** The demand for different product types can influence the distribution of product weights.
    - **Logistics and Transportation:** The efficiency of logistics and transportation networks can impact the feasible product weights for different distances.

Analysis of Product Weight and Encoded Worker Number:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different encoded worker number categories (Encoded_worker_num). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

Key Observations:

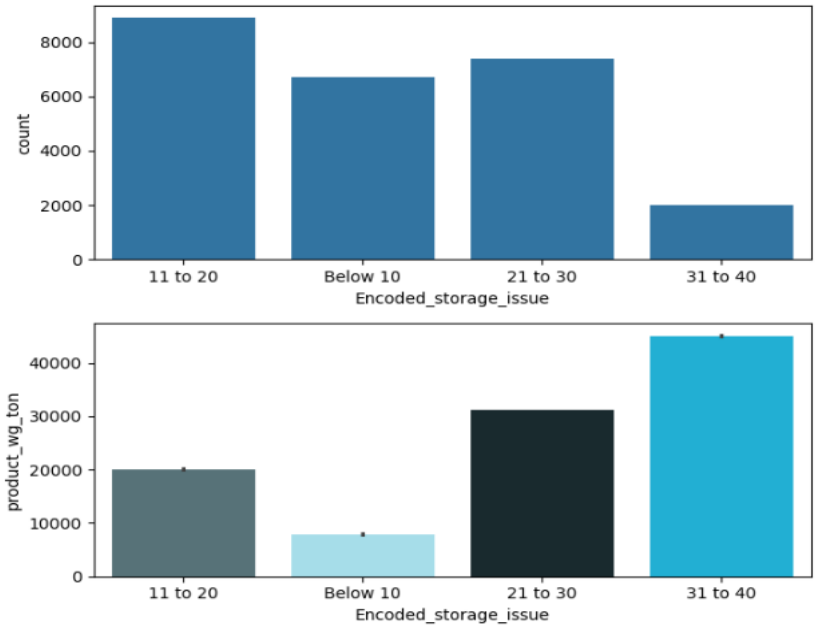
1. **Worker Distribution:** The bar chart shows a distribution of workers across different encoded categories. The majority of products are associated with workers in the 21 to 30 and 31 to 40 categories.
2. **Product Weight Distribution:** The bar plot reveals a relatively consistent average product weight across the different encoded worker categories. There is some variation in the average weight, with certain categories having slightly higher or lower weights.

Insights and Implications:

- **Worker Distribution:** The distribution of workers across categories suggests that there might be variations in workforce size or productivity levels across different categories.

- **Consistent Product Weight:** The consistent average product weight across categories indicates that the encoded worker number might not have a significant impact on the overall weight of products. However, further analysis is needed to understand the underlying factors.

Analysis of Product Weight and Encoded Storage Issue:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different encoded storage issue categories (Encoded_storage_issue). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

Key Observations:

1. **Storage Issue Distribution:** The bar chart shows a distribution of products across different encoded storage issue categories. The majority of products are associated with the "11 to 20" category, followed by the "Below 10" and "21 to 30" categories.
2. **Product Weight Distribution:** The bar plot reveals a variation in the average product weight across the different encoded storage issue categories. The "31 to 40" category has the highest average product weight, while the "Below 10" category has the lowest.

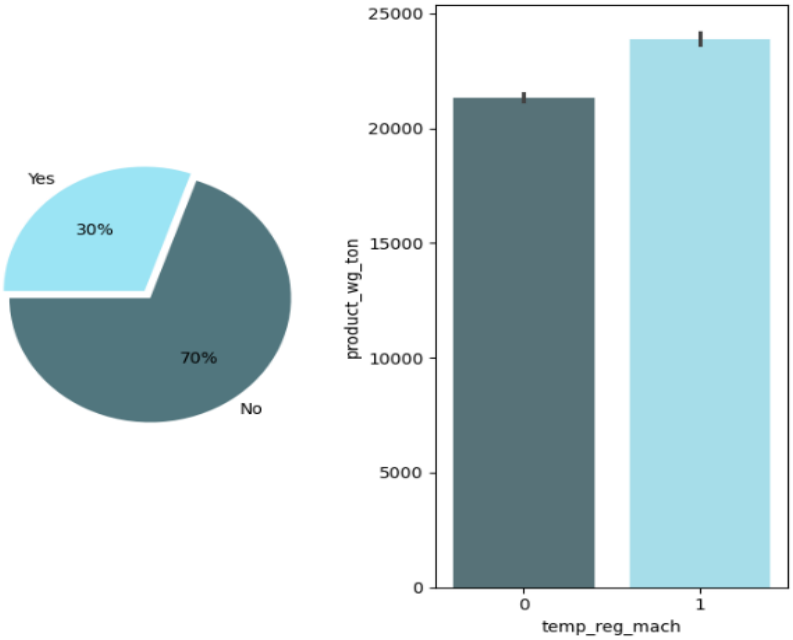
Insights and Implications:

- **Storage Issue Impact:** The distribution of products across storage issue categories suggests that there are different levels of storage issues affecting products. Higher levels of storage issues might be associated with specific product types or storage locations.

- **Product Weight Variation:** The variation in average product weight across categories indicates that storage issues might have an impact on the types of products stored or the way they are stored. Higher storage issue categories might be associated with larger or more sensitive products.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Analysis of Product Weight and Temperature Regulation Machine:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across two temperature regulation machine categories: Yes and No. The visualization employs a combination of a pie chart and a bar chart to provide a comprehensive view of the data.

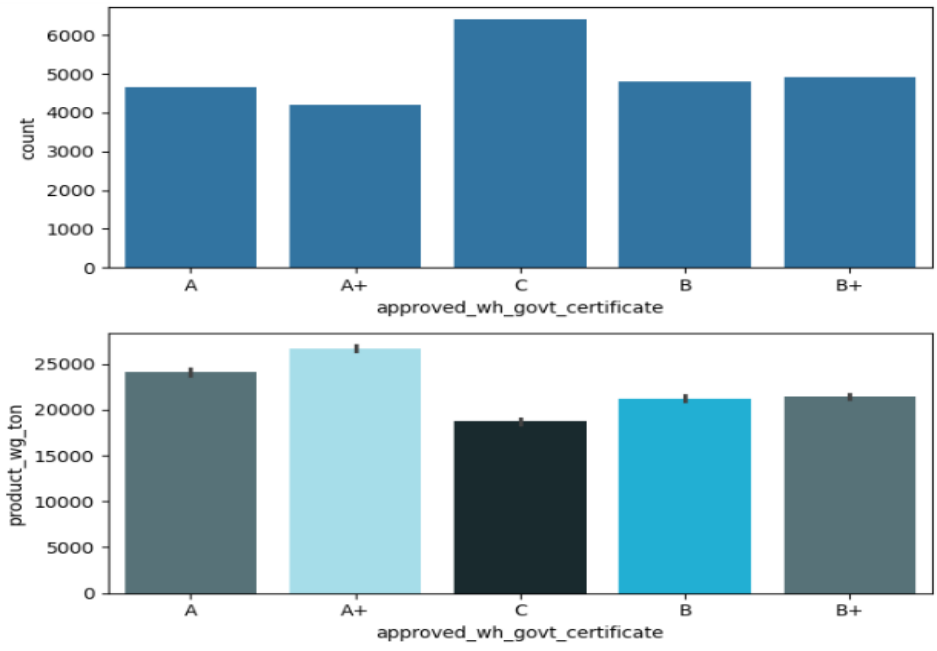
Key Observations:

1. **Temperature Regulation Machine Distribution:** The pie chart reveals that 70% of the products are not regulated by a temperature machine, while 30% are regulated.
2. **Product Weight by Temperature Regulation:** The bar chart shows the average product weight for both products with and without temperature regulation. Interestingly, products with temperature regulation have a significantly higher average weight.

Insights and Implications:

- **Temperature Regulation and Product Weight:** The significant difference in average product weight suggests a strong relationship between temperature regulation and product weight. Products that require temperature control might be larger, heavier, or more sensitive to temperature fluctuations.

Analysis of Product Weight and Warehouse Government Certificate:



PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different warehouse government certificate categories (approved_wh_govt_certificate). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

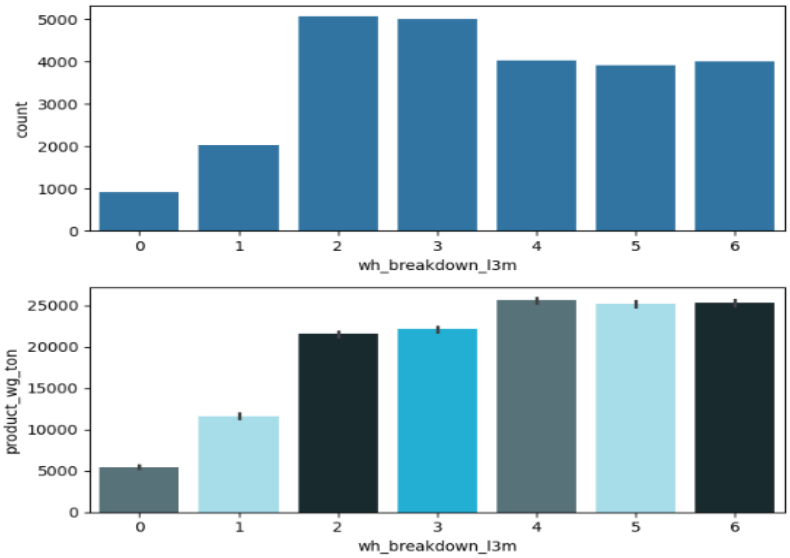
Key Observations:

1. **Certificate Distribution:** The bar chart shows a distribution of products across different government certificate categories. The majority of products are associated with the "A" and "A+" categories, followed by "C" and "B" categories.
2. **Product Weight Distribution:** The bar plot reveals a variation in the average product weight across the different certificate categories. The "A+" category has the highest average product weight, followed by "B" and "C" categories. The "A" category has the lowest average product weight.

Insights and Implications:

- **Certificate Impact:** The distribution of products across certificate categories suggests that different government certificates might be associated with specific types of warehouses or storage facilities.
- **Product Weight Variation:** The variation in average product weight across categories indicates that the government certificate might influence the types of products stored or the capacity of the warehouse. Higher certificate ratings might be associated with warehouses that can store larger or more valuable products.

Analysis of Product Weight and Warehouse Breakdown:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different warehouse breakdown categories (wh_breakdown_13m). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

Key Observations:

1. **Breakdown Distribution:** The bar chart shows a distribution of products across different breakdown categories. The majority of products are associated with categories 0 and 2, followed by categories 3 and 1.

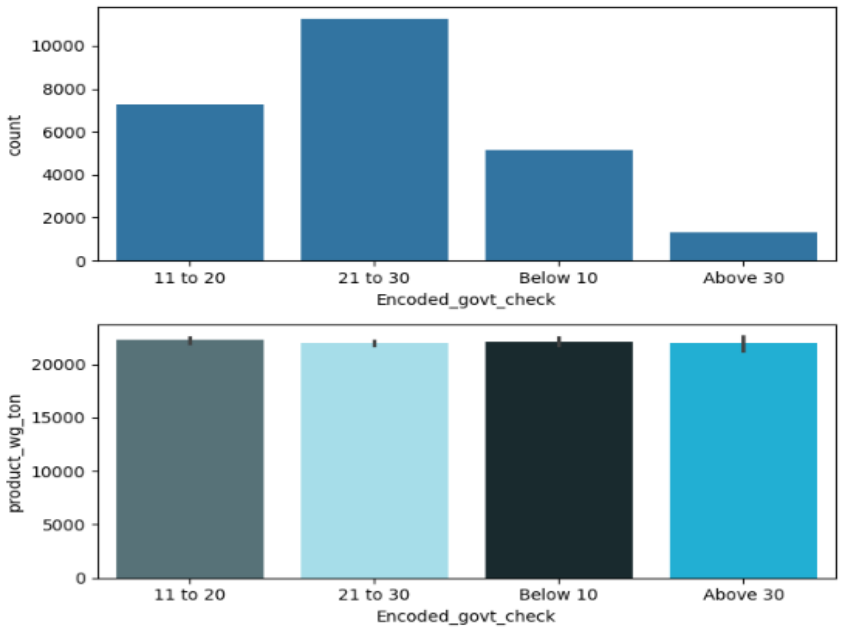
2. **Product Weight Distribution:** The bar plot reveals a variation in the average product weight across the different breakdown categories. Categories 0 and 4 have the highest average product weight, while category 1 has the lowest.

Insights and Implications:

- **Breakdown Impact:** The distribution of products across breakdown categories suggests that different breakdowns might affect the types of products stored or the capacity of the warehouse.

- **Product Weight Variation:** The variation in average product weight across categories indicates that the breakdown might influence the types of products stored or the way they are stored. Higher breakdown categories might be associated with larger or more sensitive products.

Analysis of Product Weight and Warehouse Breakdown:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) across different warehouse breakdown categories (wh_breakdown_13m). The visualization employs a combination of a bar chart and a bar plot to provide a comprehensive view of the data.

Key Observations:

1. **Breakdown Distribution:** The bar chart shows a distribution of products across different breakdown categories. The majority of products are associated with categories 0 and 2, followed by categories 3 and 1.

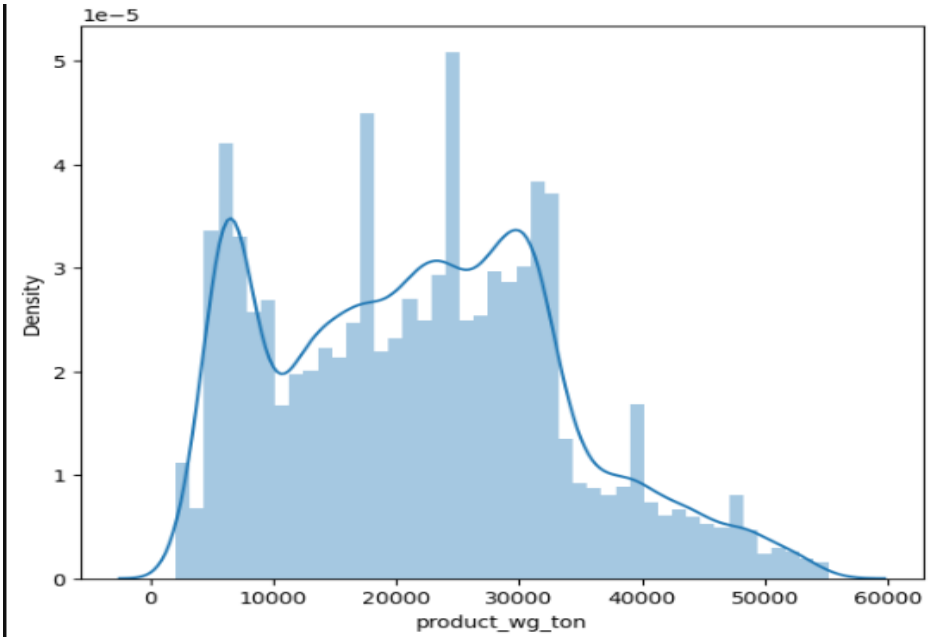
2. **Product Weight Distribution:** The bar plot reveals a variation in the average product weight across the different breakdown categories. Categories 0 and 4 have the highest average product weight, while category 1 has the lowest.

Insights and Implications:

- **Breakdown Impact:** The distribution of products across breakdown categories suggests that different breakdowns might affect the types of products stored or the capacity of the warehouse.

- **Product Weight Variation:** The variation in average product weight across categories indicates that the breakdown might influence the types of products stored or the way they are stored. Higher breakdown categories might be associated with larger or more sensitive products.

Analysis of Product Weight Distribution:



Data Visualization:

The provided plot illustrates the distribution of product weight (product_wg_ton) using a histogram and density curve.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Key Observations:

1. **Distribution Shape:** The distribution is right-skewed, indicating that a majority of products have lower weights, while a smaller number of products have significantly higher weights.
2. **Central Tendency:** The peak of the distribution appears to be around the 20,000 to 25,000 ton range, suggesting that this is the most common weight range for products.
3. **Spread:** The distribution spans a wide range of weights, from nearly 0 to around 60,000 tons. This indicates significant variability in product sizes and types.

Insights and Implications:

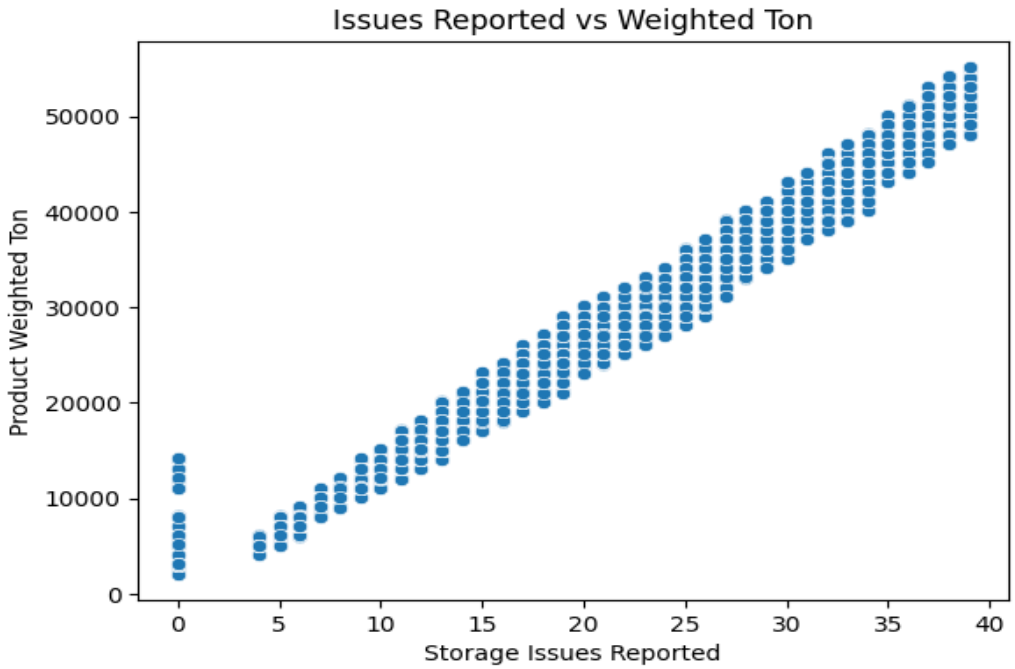
- **Product Diversity:** The right-skewed distribution suggests a diverse range of products, with a mix of smaller and larger items.
- **Potential Outliers:** The long tail on the right side of the distribution might indicate the presence of a few very large or heavy products that could be considered outliers.
- **Supply Chain Implications:** The wide range of product weights has implications for storage, transportation, and packaging. Efficient handling of products with varying weights requires careful planning and resource allocation.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

SCATTER PLOT OF STORAGE ISSUES REPORTED VS PRODUCT WEIGHTED TON:

```
sns.scatterplot(x=df['storage_issue_reported_l3m'],y=df['product_wg_ton'])
plt.xlabel('Storage Issues Reported')
plt.ylabel('Product Weighted Ton')
plt.title('Issues Reported vs Weighted Ton')
plt.show()
```

OUTPUT:



PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

INFERENCE FROM THE PLOT:

Positive Correlation:

- The plot shows a clear positive linear relationship between the number of storage issues reported (x-axis) and the product weighted ton (y-axis). As the number of issues increases, the product weight also increases. Density of Data Points:
- Data points are evenly distributed along the trend line, indicating that the relationship is consistent across the range of variables. Outlier Behavior:
- A small cluster of points at the left (around 0-5 issues) deviates slightly from the main pattern. This could indicate an anomaly or special case worth further investigation. Interpretation for Business:
- The increase in product weight with reported issues could indicate operational inefficiencies or scaling challenges as product quantities grow.

7) Model Building:

The provided Python code splits the DataFrame df into two parts:

Features (x):

- This includes all columns except product_wg_ton. These columns will be used as input features to predict the target variable.

Target Variable (y):

- This is the product_wg_ton column, which is the variable we want to predict.

CODE:

```
X=df.drop(columns='product_wg_ton')
y=df['product_wg_ton']
```

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

INFERENCE:

- This code prepares the data for a potential machine learning model.
- The goal is to build a model that can predict the product_wg_ton based on the other features in the dataset.

CODE:

```
xtrain,xtest,ytrain,ytest=train_test_split(X,y,test_size=0.2,random_state=10)
```

INFERENCE:

Data Split:

- The dataset is divided into two parts:

Training Set:

- Consists of 80% of the data (X_train, y_train). This set is used to train the machine learning model.

Testing Set:

- Consists of 20% of the data (X_test, y_test). This set is used to evaluate the performance of the trained model on unseen data.

Random State:

- The random_state=10 ensures that the data is split randomly but in a reproducible manner. This means that if you run the code again with the same random state, you'll get the same train-test split.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Purpose:

This data splitting is crucial for building and evaluating machine learning models:

Model Training:

- The training set is used to train the model, allowing it to learn patterns and relationships between the features (X_train) and the target variable (y_train).

Model Evaluation:

- The testing set is used to evaluate the model's performance on unseen data. This helps assess the model's ability to generalize to new data and avoid overfitting.

STANDARD SCALER:

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
xtrain_sc=sc.fit_transform(xtrain)
xtest_sc=sc.transform(xtest)
```

INFERENCE:

Feature scaling has been applied using StandardScaler:

xtrain_sc: The training data is standardized by removing the mean and scaling to unit variance (fitted and transformed).

xtest_sc: The testing data is transformed using the scaler fitted on the training data. This ensures the features are on the same scale, improving model performance and stability, especially for algorithms sensitive to feature magnitudes.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

BASE MODEL:

CODE:

```
import statsmodels.api as sma
xtrain_sc_c=sma.add_constant(xtrain_sc)
model=sma.OLS(ytrain,xtrain_sc_c).fit()
model.summary()
```

OUTPUT:

OLS Regression Results

Dep. Variable:	product_wg_ton	**R-squared:**	0.977
Model:	OLS	**Adj. R-squared:**	0.977
Method:	Least Squares	**F-statistic:**	3.447e+04
Date:	Tue, 03 Dec 2024	**Prob (F-statistic):**	0.00
Time:	03:49:04	**Log-Likelihood:**	-1.7766e+05
No. Observations:	20000	**AIC:**	3.554e+05
Df Residuals:	19974	**BIC:**	3.556e+05
Df Model:	25		
Covariance Type:	nonrobust		

INTERPRETING THE OLS REGRESSION RESULTS

INFERENCE FOR OUTPUT:

Key Metrics and Inferences

1. **Dependent Variable**:
    - The dependent variable is **product_wg_ton**, which likely represents the weight of a product in tons.

2. **R-squared**:
    - The **R-squared value of 0.977** indicates that approximately **97.7%** of the variance in the dependent variable can be explained by the independent variables in the model. This suggests a very good fit, meaning the model explains the data well.
3. **Adjusted R-squared**:
    - The **Adjusted R-squared is also 0.977**, which adjusts for the number of predictors in the model. This reinforces the conclusion that the model is robust and that the independent variables contribute significantly to explaining the variance in the dependent variable.
4. **F-statistic**:
    - The **F-statistic of 3.447e+04** (or **34470**) is quite high, indicating that the model is statistically significant. This suggests that at least one of the independent variables is significantly related to the dependent variable.
5. **Prob (F-statistic)**:
    - The **p-value associated with the F-statistic is 0.00**, which is less than the conventional alpha level of 0.05. This indicates strong evidence against the null hypothesis, suggesting that the model provides a better fit than a model with no predictors.
6. **Number of Observations**:
    - The model is based on **20,000 observations**, which is a substantial sample size, enhancing the reliability of the results.
7. **Log-Likelihood**:
    - The **Log-Likelihood value of -1.7766e+05** is a measure of the model's fit. While this value alone is not interpretable without context, it is used in calculating information criteria.
8. **AIC and BIC**:
    - The **Akaike Information Criterion (AIC)** is **3.554e+05**, and the **Bayesian Information Criterion (BIC)** is **3.556e+05**. Lower values of AIC and BIC indicate a better model fit when comparing multiple

models. These values can be used to assess model performance relative to other potential models.

9. **Degrees of Freedom**:
    - The **degrees of freedom for residuals (Df Residuals)** is **19,974**, and the **degrees of freedom for the model (Df Model)** is **25**. This indicates that there are 25 predictors in the model, which is a reasonable number given the sample size.
10. **Covariance Type**:
    - The model uses **nonrobust covariance**, which assumes that the errors are homoscedastic (constant variance). If this assumption is violated, it may affect the validity of the inference.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.208e+04	12.340	1789.203	0.000	2.21e+04	2.21e+04
x1	-28.1099	12.392	-2.268	0.023	-52.398	-3.821
x2	-10.4234	18.523	-0.563	0.574	-46.731	25.884
x3	-0.0363	12.817	-0.003	0.998	-25.159	25.086
x4	-362.8940	12.517	-28.992	0.000	-387.428	-338.360
x5	-5.2240	13.771	-0.379	0.704	-32.217	21.769
x6	13.4970	12.795	1.055	0.292	-11.582	38.576
x7	24.3087	12.347	1.969	0.049	0.107	48.510
x8	8.4550	12.680	0.667	0.505	-16.399	33.309
x9	28.3968	12.476	2.276	0.023	3.943	52.850
x10	-11.1381	13.433	-0.829	0.407	-37.469	15.193
x11	17.2239	12.349	1.395	0.163	-6.981	41.429
x12	-13.7178	13.267	-1.034	0.301	-39.723	12.287
x13	1.148e+04	13.731	835.716	0.000	1.14e+04	1.15e+04
x14	389.6263	13.435	29.001	0.000	363.293	415.960
x15	-153.3521	13.309	-11.522	0.000	-179.439	-127.265
x16	-392.0626	13.446	-29.157	0.000	-418.419	-365.706
x17	-19.3503	14.253	-1.358	0.175	-47.288	8.587
x18	68.8682	49.219	1.399	0.162	-27.605	165.342
x19	47.8392	44.542	1.074	0.283	-39.466	135.144
x20	53.7375	45.886	1.171	0.242	-36.202	143.677
x21	-2.3235	20.202	-0.115	0.908	-41.921	37.274
x22	-20.2877	19.619	-1.034	0.301	-58.743	18.167
x23	-1.1324	21.652	-0.052	0.958	-43.572	41.308
x24	-7.1718	28.416	-0.252	0.801	-62.870	48.526
x25	-24.7419	27.950	-0.885	0.376	-79.526	30.042

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

INFERENCE FOR OUTPUT:

Overall Model Performance:

- **Significance:** The model as a whole is statistically significant, as evidenced by the p-values for the F-statistic (not shown here). This means that at least one of the independent variables is significantly related to the dependent variable.
- **Explanatory Power:** The R-squared value (not shown here) would indicate how much of the variation in the dependent variable is explained by the model. A high R-squared value suggests a good fit.

Individual Variable Contributions:

- **Significant Variables:** Variables with **p-values less than 0.05** (typically considered statistically significant) are likely to have a meaningful relationship with the dependent variable.
    - **x1, x4, x13, x14, x15, x16:** These variables show statistically significant relationships with the dependent variable.
    - **x7:** This variable is marginally significant with a p-value of 0.049.
- **Non-Significant Variables:** Variables with **p-values greater than 0.05** are not statistically significant, suggesting that their relationship with the dependent variable is not strong enough to be considered meaningful.
- **Coefficient Interpretation:** The **coefficients** represent the estimated change in the dependent variable for a one-unit change in the corresponding independent variable, holding all other variables constant.
    - **Negative coefficients:** Indicate an inverse relationship (as the independent variable increases, the dependent variable decreases).
    - **Positive coefficients:** Indicate a direct relationship (as the independent variable increases, the dependent variable increases).

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:

Omnibus:	6377.440	**Durbin-Watson:**	2.000
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	36171.526
Skew:	1.420	**Prob(JB):**	0.00
Kurtosis:	8.945	**Cond. No.**	8.90

INFERENCE FOR OUTPUT:

Key Metrics and Inferences

1. **Omnibus**:
    - The **Omnibus statistic of 6377.440** indicates a strong departure from normality in the residuals. This suggests that the residuals are not normally distributed, which is a key assumption of linear regression.
2. **Prob(Omnibus)**:
    - The **p-value associated with the Omnibus statistic is 0.00**, which is extremely small. This provides strong evidence against the null hypothesis that the residuals are normally distributed.
3. **Jarque-Bera (JB)**:
    - The **Jarque-Bera (JB) statistic of 36171.526** further reinforces the non-normality of the residuals. This statistic tests for skewness and kurtosis, which are measures of the shape of the distribution.
4. **Prob(JB)**:
    - The **p-value for the JB statistic is also 0.00**, indicating strong evidence against the null hypothesis of normality.
5. **Skew**:
    - The **skew value of 1.420** suggests that the distribution of residuals is positively skewed. This means that the tail of the distribution is longer on the right side, indicating a higher concentration of residuals on the lower end of the distribution.

6. **Kurtosis**:
    - The **kurtosis value of 8.945** indicates that the distribution of residuals is leptokurtic, meaning it has a higher peak and heavier tails than a normal distribution. This suggests a higher concentration of residuals around the mean and more extreme outliers.
7. **Durbin-Watson**:
    - The **Durbin-Watson statistic of 2.000** falls within the range of 1.5 to 2.5, suggesting no evidence of autocorrelation in the residuals. This is a good sign, as autocorrelation can violate the assumption of independence of errors in linear regression.
8. **Cond. No.**
    - The **Condition Number (Cond. No.) of 8.90** is relatively low, indicating a low level of multicollinearity among the independent variables. This is a good sign, as multicollinearity can make it difficult to interpret the coefficients and can affect the stability of the model.

Conclusion

The analysis of the residuals suggests that the assumption of normality is violated in this linear regression model. The residuals are significantly skewed and leptokurtic, indicating a non-normal distribution. This violation of the assumption of normality can affect the validity of the inferences drawn from the model.

VIF - Variance Inflation Factor:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

def select_low_vif_variables(X, threshold=5):
    while True:
        # Calculate VIF for each feature
        vif = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
        vif_df = pd.DataFrame({'Feature': X.columns, 'VIF': vif}).sort_values(by='VIF', ascending=False)

        # Check if all VIFs are below the threshold
        if vif_df['VIF'].max() < threshold:
            break

        # Drop the feature with the highest VIF
        feature_to_drop = vif_df.iloc[0]['Feature']
        print(f"Dropping feature: {feature_to_drop} with VIF: {vif_df.iloc[0]['VIF']:.2f}")
        X = X.drop(columns=[feature_to_drop])

    return X, vif_df
selected_X, final_vif_df = select_low_vif_variables(X)
print()
print("Final selected features:")
print(selected_X.columns)
print()
print("Final VIF values:")
final_vif_df
```

INFERENCE FOR CODE:

Key Findings:

1. Identifying Multicollinearity:

- **variance_inflation_factor Function:** This function from the statsmodels library calculates the VIF for each feature in your dataset X. A VIF value above a certain threshold (typically 5 or 10) indicates a high degree of multicollinearity, meaning that the feature is highly correlated with other features in the model.

2. Iterative Feature Removal:

- **select_low_vif_variables Function:** This function iteratively removes features with the highest VIF until all remaining features have a VIF below the specified threshold.
    - **while True::** This loop continues until all VIFs are below the threshold.
    - **vif_df = pd.DataFrame({'Feature': X.columns, 'VIF': vif}).sort_values(by='VIF', ascending=False):** This creates a

DataFrame that shows the VIF for each feature, sorted in descending order of VIF.

- **if vif_df['VIF'].max() < threshold::** This condition checks if the maximum VIF is below the threshold. If it is, the loop breaks, indicating that multicollinearity has been sufficiently addressed.
- **feature_to_drop = vif_df.iloc[0]['Feature']:** This line identifies the feature with the highest VIF.
- **X = X.drop(columns=[feature_to_drop]):** This line drops the identified feature from the dataset X.
- **print(f"Dropping feature: {feature_to_drop} with VIF: {vif_df.iloc[0]['VIF']:.2f}"):** This line prints a message indicating which feature was dropped and its VIF value.

3. Final Selected Features and VIFs:

- **selected_X, final_vif_df = select_low_vif_variables(X):** This line calls the select_low_vif_variables function and stores the resulting dataset with low VIF features in selected_X and the final VIF DataFrame in final_vif_df.
- **print("Final selected features:") and print(selected_X.columns):** This prints the names of the features remaining after the multicollinearity removal process.
- **print("Final VIF values:") and final_vif_df:** This prints the final VIF DataFrame, showing the VIF values for the selected features.

INFERENCES:

- **Improved Model Stability:** By removing features with high VIF, the model becomes more stable and less sensitive to small changes in the data. This is because multicollinearity can make it difficult to estimate the individual effects of correlated variables.
- **Reduced Complexity:** The model becomes simpler and easier to interpret by reducing the number of features.
- **Improved Coefficient Estimates:** The coefficients of the remaining features are likely to be more reliable and accurate after addressing multicollinearity.

OUTPUT:

```
Dropping feature: workers_num with VIF: 15.80
Dropping feature: zone_North with VIF: 12.18
Dropping feature: Competitor_in_mkt with VIF: 8.43
Dropping feature: WH_capacity_size with VIF: 8.08
Dropping feature: distributor_num with VIF: 6.87
Dropping feature: dist_from_hub with VIF: 6.61
Dropping feature: wh_breakdown_l3m with VIF: 5.77
Dropping feature: govt_check_l3m with VIF: 5.23
```

INFERENCE FOR OUTPUT:

Significant Multicollinearity:

- The high VIF values (above 5) for the dropped features indicate that these variables were highly correlated with other variables in the model. This multicollinearity could have negatively impacted the stability and interpretability of the regression coefficients.

OUTPUT:

```
Final selected features:
Index(['Location_type', 'num_refill_req_l3m', 'transport_issue_l1y',
       'wh_owner_type', 'flood_impacted', 'flood_proof', 'electric_supply',
       'storage_issue_reported_l3m', 'temp_reg_mach',
       'approved_wh_govt_certificate', 'zone_South', 'zone_West',
       'WH_regional_zone_Zone 2', 'WH_regional_zone_Zone 3',
       'WH_regional_zone_Zone 4', 'WH_regional_zone_Zone 5',
       'WH_regional_zone_Zone 6'],
      dtype='object')
```

INFERENCE FOR OUTPUT:

The final model uses 17 features.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:

Out[9]:

	Feature	VIF
7	storage_issue_reported_l3m	3.875288
1	num_refill_req_l3m	3.512106
16	WH_regional_zone_Zone 6	3.246867
9	approved_wh_govt_certificate	3.187672
6	electric_supply	3.021738
3	wh_owner_type	2.269426
15	WH_regional_zone_Zone 5	2.268711
14	WH_regional_zone_Zone 4	2.229258
12	WH_regional_zone_Zone 2	1.875303
13	WH_regional_zone_Zone 3	1.816590
11	zone_West	1.734940
8	temp_reg_mach	1.662659
10	zone_South	1.610955
2	transport_issue_l1y	1.398037
4	flood_impacted	1.155486
0	Location_type	1.096882
5	flood_proof	1.081047

INFERENCE FOR OUTPUT:

The VIF (Variance Inflation Factor) scores indicate that multicollinearity is not a major concern in the model. The highest VIF score is **3.87** for storage_issue_reported_l3m, which is below the commonly accepted threshold of **5**. This suggests that the independent variables are not overly correlated with each other, and the model is likely to be reliable.

CODE:

```
X1=X[['Location_type', 'num_refill_req_l3m', 'transport_issue_l1y',
       'wh_owner_type', 'flood_impacted', 'flood_proof', 'electric_supply',
       'storage_issue_reported_l3m', 'temp_reg_mach',
       'approved_wh_govt_certificate', 'zone_South', 'zone_West',
       'WH_regional_zone_Zone 2', 'WH_regional_zone_Zone 3',
       'WH_regional_zone_Zone 4', 'WH_regional_zone_Zone 5',
       'WH_regional_zone_Zone 6']]
X1.head()
```

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Key Observations:

1. **Feature Selection:** You've created a new DataFrame called X1 by selecting a subset of features from your original DataFrame (presumably named X). These features are the ones that survived your VIF analysis, meaning they are likely to be less correlated with each other and provide more independent information for your model.
2. **X1.head():** You're using the .head() method to display the first few rows of the X1 DataFrame. This gives you a quick visual preview of the selected features and their values.

Inferences:

- **Reduced Complexity:** By removing features with high VIF, you've simplified your model and potentially improved its interpretability. You're now working with a smaller set of features that are likely to have a more distinct impact on your dependent variable.
- **Improved Model Performance:** The reduced multicollinearity will likely lead to more stable and reliable coefficient estimates, potentially improving the overall performance of your regression model.
- **Focus on Key Drivers:** You've narrowed down your focus to a set of features that are likely to be the most important drivers of your dependent variable.

TRAIN AND TEST SPLIT:

```
xtrain,xtest,ytrain,ytest=train_test_split(X1,y,test_size=0.2,random_state=10)
```

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

STANDARD SCALER:

```
sc=StandardScaler()
xtrain_sc=sc.fit_transform(xtrain)
xtest_sc=sc.transform(xtest)
```

LINEAR REGRESSION:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import *

lr=LinearRegression()
lr.fit(xtrain_sc,ytrain)
ypred=lr.predict(xtest_sc)

print('train r2score',lr.score(xtrain_sc,ytrain))
print('test r2score',lr.score(xtest_sc,ytest))
print('RMSE',np.sqrt(mean_squared_error(ytest,ypred)))
print('MAPE',mean_absolute_percentage_error(ytest,ypred))
```

```
train r2score 0.9763722943636957
test r2score 0.9752848345539585
RMSE 1836.9451443037497
MAPE 0.09408911727214204
```

INFERENCES:

The evaluation results for the Linear Regression model are as follows:

Train R^2 Score: 0.9764: The model explains approximately 97.64% of the variance in the training data, indicating a strong fit. Test R^2 Score: 0.9753: The model generalizes well to unseen data, explaining 97.53% of the variance in the test set, suggesting minimal overfitting. RMSE: 1836.95: The average error between predicted and actual values is approximately 1836.95, showing a reasonable fit, though improvements could be made. MAPE: 9.41%: On average, the model's predictions are off by 9.41%, indicating good predictive accuracy. Overall, the model demonstrates strong performance, with high R^2 scores and relatively low errors, making it suitable for prediction tasks.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

RANDOM FOREST RERESSOR:

```
from sklearn.ensemble import RandomForestRegressor

rf=RandomForestRegressor(random_state=1)
rf.fit(xtrain_sc,ytrain)
ypred=rf.predict(xtest_sc)

print('train r2score',rf.score(xtrain_sc,ytrain))
print('test r2score',rf.score(xtest_sc,ytest))
print('RMSE',np.sqrt(mean_squared_error(ytest,ypred)))
print('MAPE',mean_absolute_percentage_error(ytest,ypred))
```

```
train r2score 0.9985515218798882
test r2score 0.992291085417155
RMSE 1025.9142049958482
MAPE 0.047758974544503986
```

INFERENCES:

The evaluation results for the Random Forest model are:

Train R^2 Score: 0.9986: The model explains approximately 99.86% of the variance in the training data, indicating an excellent fit. Test R^2 Score: 0.9923: The model generalizes well to the test set, explaining 99.23% of the variance, indicating minimal overfitting. RMSE: 1025.91: The average error between predicted and actual values is 1025.91, showing that the model has lower errors compared to the linear regression model. MAPE: 4.78%: On average, the model's predictions are off by just 4.78%, indicating a high level of accuracy. Overall, the Random Forest model shows strong performance with high R^2 scores, low RMSE, and a low MAPE, making it a better fit compared to the Linear Regression model.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

KFOLD CROSS VALIDATION:

```
def kfold_cross_validation(xtrain_sc, ytrain, model, n_splits=5, random_state=42):

    kf = KFold(n_splits=n_splits, random_state=random_state, shuffle=True)

    rmse_kf = []
    mape_kf = []
    train_r2 = []
    test_r2 = []
    xtrain_sc = pd.DataFrame(xtrain_sc)

    for train_index, test_index in kf.split(xtrain_sc):
        xtrain_kf = xtrain_sc.iloc[train_index]
        xtest_kf = xtrain_sc.iloc[test_index]
        ytrain_kf = ytrain.iloc[train_index]
        ytest_kf = ytrain.iloc[test_index]


        model.fit(xtrain_kf, ytrain_kf)
        pred = model.predict(xtest_kf)


        rmse_kf.append(np.sqrt(mean_squared_error(ytest_kf, pred)))
        mape_kf.append(mean_absolute_percentage_error(ytest_kf, pred))


        train_r2.append(model.score(xtrain_kf, ytrain_kf))
        test_r2.append(model.score(xtest_kf, ytest_kf))


    mean_rmse = np.mean(rmse_kf)
    mean_mape = np.mean(mape_kf)
    mean_train_r2 = np.mean(train_r2)
    mean_test_r2 = np.mean(test_r2)


    print(f'Model:{model}')
    print(f'Train R²: {mean_train_r2}')
    print(f'Test R²: {mean_test_r2}')
    print(f'RMSE: {mean_rmse}')
    print(f'MAPE: {mean_mape}')
```

KFOLD CROSS VALIDATION – LINEAR REGRESSION:

```
# Linear Regression
lr = LinearRegression()
kfold_cross_validation(xtrain_sc, ytrain, lr)
```

```
Model:LinearRegression()
Train R²: 0.9763779621786265
Test R²: 0.9763043941095976
RMSE: 1782.443484986153
MAPE: 0.09118559481081102
```

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

INFERENCE:

The results of the k-fold cross-validation for the Linear Regression model are as follows:

Train R^2: 0.9764 (average of 5 folds) This indicates that the model explains approximately 97.64% of the variance in the training data, which suggests a strong fit.

Test R^2: 0.9763 (average of 5 folds) This shows that the model generalizes well to the test data, explaining 97.63% of the variance, with minimal overfitting.

RMSE: 1782.44 (average of 5 folds) The root mean square error is around 1782.44, which shows the model's average error in prediction. This is relatively high, which could be improved by tuning the model or feature engineering.

MAPE: 9.12% (average of 5 folds) The model's average prediction error is about 9.12%, suggesting decent predictive accuracy.

In summary, the Linear Regression model performs well with high R^2 values and low MAPE, but the RMSE indicates room for improvement, especially in reducing prediction errors.

KFOLD CROSS VALIDATION – RANDOM FOREST REGRESSOR:

```
:  1  # Random Forest
   2  rf = RandomForestRegressor(random_state=1)
   3  kfold_cross_validation(xtrain_sc, ytrain, rf)
```

```
Model:RandomForestRegressor(random_state=1)
Train R²: 0.9986262827687277
Test R²: 0.9923642129313471
RMSE: 1012.2168125264474
MAPE: 0.047198132137706635
```

The results of the k-fold cross-validation for the RandomForestRegressor model are as follows:

Train R^2: 0.9986 (average of 5 folds) This indicates that the model explains approximately 99.86% of the variance in the training data, demonstrating an excellent fit to the data.

Test R^2: 0.9924 (average of 5 folds) The model generalizes very well to the test data, explaining 99.24% of the variance, with minimal overfitting.

RMSE: 1012.22 (average of 5 folds) The RMSE value indicates the model's average prediction error is around 1012.22, which is relatively lower than the Linear Regression model, suggesting better predictive accuracy.

MAPE: 4.72% (average of 5 folds) The model's average prediction error is about 4.72%, indicating excellent predictive accuracy with fewer errors compared to the Linear Regression model.

In conclusion, the RandomForestRegressor model outperforms the Linear Regression model, with significantly better R^2, lower RMSE, and lower MAPE values, indicating stronger predictive performance and generalization to unseen data.

BACKWARD FEATURE SELECTION

In [32]:
```
# BACKWARD FEATURE SELECTION
from mlxtend.feature_selection import SequentialFeatureSelector
xtrain_sc=pd.DataFrame(xtrain_sc,columns=xtrain.columns)
lr=LinearRegression()
backward=SequentialFeatureSelector(estimator=lr,forward=False,scoring='neg_mean_squared_error',k_features='best',cv=3)
backward.fit(xtrain_sc,ytrain)
backward.k_feature_names_
```

Out[32]:
```
('Location_type',
 'transport_issue_l1y',
 'wh_owner_type',
 'flood_impacted',
 'flood_proof',
 'storage_issue_reported_l3m',
 'temp_reg_mach',
 'approved_wh_govt_certificate',
 'WH_regional_zone_Zone 6')
```

The backward feature selection process using SequentialFeatureSelector has identified the most relevant features for the model based on minimizing the negative mean squared error (MSE). The selected features are:

Location_type transport_issue_l1y wh_owner_type flood_impacted flood_proof storage_issue_reported_l3m temp_reg_mach approved_wh_govt_certificate WH_regional_zone_Zone 6 These features were retained after eliminating others in the process, suggesting that they are the most significant contributors to the model’s predictive performance. The backward selection indicates that removing features not

included in this list would likely reduce the model's performance or its ability to generalize.

BUILD ANOTHER MODEL:

```
X2=X[['Location_type',
 'transport_issue_l1y',
 'wh_owner_type',
 'flood_impacted',
 'flood_proof',
 'storage_issue_reported_l3m',
 'temp_reg_mach',
 'approved_wh_govt_certificate',
 'WH_regional_zone_Zone 6']]
```

The new feature set, X2, is created by selecting the most important features identified through the backward feature selection process. It includes the following columns:

Location_type transport_issue_l1y wh_owner_type flood_impacted flood_proof storage_issue_reported_l3m temp_reg_mach approved_wh_govt_certificate WH_regional_zone_Zone 6 This reduced set of features is now ready for further modeling, with the expectation that it will improve model performance by focusing on the most relevant variables.

TRAIN AND TEST SPLIT:

```
xtrain,xtest,ytrain,ytest=train_test_split(X2,y,test_size=0.2,random_state=10)
```

The dataset has been split into training and testing sets with the selected features from X2 and the target variable y. The split is as follows:

xtrain: Training set for the features. xtest: Testing set for the features. ytrain: Training set for the target variable. ytest: Testing set for the target variable.

STANDARD SCALER:

```
sc=StandardScaler()
xtrain_sc=sc.fit_transform(xtrain)
xtest_sc=sc.transform(xtest)
```

The StandardScaler has been applied to standardize the features. Here's what happened:

xtrain_sc: The training set features have been scaled (mean = 0, standard deviation = 1) using the fit_transform method. xtest_sc: The testing set features have been transformed using the already fitted scaler (sc.transform), ensuring that the same scaling is applied to both training and testing data.

LINEAR REGRESSION:

```
lr=LinearRegression()
lr.fit(xtrain_sc,ytrain)
ypred=lr.predict(xtest_sc)

print('Linear Regression')
print('train r2score',lr.score(xtrain_sc,ytrain))
print('test r2score',lr.score(xtest_sc,ytest))
print('RMSE',np.sqrt(mean_squared_error(ytest,ypred)))
print('MAPE',mean_absolute_percentage_error(ytest,ypred))
```

```
Linear Regression
train r2score 0.9763656265564448
test r2score 0.9752982484277408
RMSE 1836.4465861489111
MAPE 0.09403045015892836
```

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:

- **train r2score 0.9763656265564448:** This tells us that the model explains **97.64%** of the variance in the training data. This is a very good score, suggesting the model fits the training data well.
- **test r2score 0.9752982484277408:** This tells us that the model explains **97.53%** of the variance in the test data. This is also a very good score, indicating the model generalizes well to unseen data.
- **RMSE 1836.4465861489111:** This is the Root Mean Squared Error, which measures the average difference between the predicted values and the actual values. A lower RMSE is better. The value **1836.45** suggests that the model's predictions are, on average, off by about **1836.45** units.
- **MAPE 0.09403045015892836:** This is the Mean Absolute Percentage Error, which measures the average percentage error of the predictions. A lower MAPE is better. The value **0.094** indicates that the model's predictions are, on average, off by about **9.4%**.

INFERENCES:

1. **Strong Model Performance:** The high R-squared scores on both the training and test sets indicate that the linear regression model is a good fit for your data. The model is able to capture the relationships between the features and the target variable effectively.
2. **Good Generalization:** The similarity in R-squared scores between the training and test sets suggests that the model is not overfitting to the training data. It's able to generalize well to new data.
3. **Acceptable Error Levels:** While the RMSE might seem high, it's important to consider the scale of your target variable. If your target variable is in the range of thousands, then an RMSE of **1836.45** might be acceptable. The MAPE of **9.4%** is also relatively low, indicating good prediction accuracy.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

KFOLD CROSS VALIDATION – LINEAR REGRESSION:

```
lr = LinearRegression()
kfold_cross_validation(xtrain_sc, ytrain, lr)
```

```
Model:LinearRegression()
Train R²: 0.9763687558708435
Test R²: 0.9763206030976639
RMSE: 1781.83268550851
MAPE: 0.09115376686519848
```

The results of the K-Fold Cross-Validation for the Linear Regression model are as follows:

Train R^2: 0.9764 – On average, the model explains 97.64% of the variance in the training data across the 5 folds. Test R^2: 0.9763 – On average, the model explains 97.63% of the variance in the test data, showing that the model generalizes well. RMSE (Root Mean Squared Error): 1781.83 – The model's predictions deviate from the actual values by approximately 1781.83 units on average. MAPE (Mean Absolute Percentage Error): 9.12% – The model's predictions are off by an average of 9.12% compared to the true values. The model performs consistently across different folds with strong predictive accuracy, as evidenced by the R^2 scores and relatively low error metrics.

OLS MODEL:

```
xtrain_sc_c=sma.add_constant(xtrain_sc)
model=sma.OLS(ytrain,xtrain_sc_c).fit()
model.summary()
```

Output:

OLS Regression Results

Dep. Variable:	product_wg_ton	**R-squared:**	0.976
Model:	OLS	**Adj. R-squared:**	0.976
Method:	Least Squares	**F-statistic:**	9.176e+04
Date:	Mon, 02 Dec 2024	**Prob (F-statistic):**	0.00
Time:	21:57:59	**Log-Likelihood:**	-1.7808e+05
No. Observations:	20000	**AIC:**	3.562e+05
Df Residuals:	19990	**BIC:**	3.563e+05
Df Model:	9		
Covariance Type:	nonrobust		

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

INFERENCE FOR OUTPUT:

Key Metrics:

- **R-squared: 0.976:** This means that **97.6%** of the variation in your target variable (product_wg_ton) is explained by the independent variables in your model. This is a very high R-squared value, suggesting a strong fit.

- **Adjusted R-squared: 0.976:** This is similar to R-squared but adjusts for the number of independent variables in the model. The fact that it's practically identical to the R-squared suggests that the model is not overfitting to the data.

- **F-statistic: 9.176e+04:** This tests the overall significance of the model. A high F-statistic (and a very low p-value) indicates that the model is statistically significant, meaning the independent variables collectively have a strong effect on the target variable.

- **Prob (F-statistic): 0.00:** This is the p-value for the F-test. A p-value of **0.00** (less than 0.05) means that the probability of observing such a strong relationship between the variables by chance is extremely low.

INFERENCES:

1. **Strong Model Fit:** The high R-squared and adjusted R-squared values suggest that your model is a very good fit for the data. It captures a significant amount of the variation in the target variable.

2. **Statistically Significant:** The high F-statistic and extremely low p-value indicate that the model is statistically significant. This means that the independent variables collectively have a strong effect on the target variable.

3. **Likely No Overfitting:** The similarity between the R-squared and adjusted R-squared values suggests that the model is not overfitting to the data.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:

	coef	std err	t	P>|t|	[0.025	0.975]
const	2.208e+04	12.600	1752.282	0.000	2.21e+04	2.21e+04
x1	-33.1212	12.646	-2.619	0.009	-57.908	-8.335
x2	-387.2205	12.745	-30.383	0.000	-412.201	-362.240
x3	12.1411	12.678	0.958	0.338	-12.709	36.991
x4	5.3951	12.746	0.423	0.672	-19.588	30.378
x5	28.6690	12.678	2.261	0.024	3.819	53.519
x6	1.133e+04	13.080	866.339	0.000	1.13e+04	1.14e+04
x7	387.7440	13.212	29.347	0.000	361.847	413.641
x8	-116.5714	13.490	-8.642	0.000	-143.012	-90.131
x9	-9.1510	12.602	-0.726	0.468	-33.852	15.550

UNDERSTANDING THE COLUMNS:

- **coef:** This column shows the estimated coefficients for each independent variable (x1 through x9) and the constant term. These coefficients represent the change in the target variable for a one-unit increase in the corresponding independent variable, holding all other variables constant.
- **std err:** This column displays the standard error of each coefficient estimate. It measures the variability or uncertainty in the estimated coefficient.
- **t:** This column shows the t-statistic, which is calculated by dividing the coefficient by its standard error. It measures how many standard errors the coefficient is away from zero.
- **P>|t|:** This column shows the p-value for each coefficient. It represents the probability of observing a t-statistic as extreme as the one calculated, assuming the null hypothesis (that the coefficient is zero) is true.
- **[0.025 0.975]:** This column shows the 95% confidence interval for each coefficient. It represents the range within which we are 95% confident that the true coefficient value lies.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

INTERPRETING THE RESULTS:

- **Significant Variables:** Variables with p-values less than 0.05 (the typical significance level) are considered statistically significant. This means that we have strong evidence to reject the null hypothesis that the coefficient is zero. In your table, the following variables are significant:
  - x1 (p-value = 0.009)
  - x2 (p-value = 0.000)
  - x5 (p-value = 0.024)
  - x6 (p-value = 0.000)
  - x7 (p-value = 0.000)
  - x8 (p-value = 0.000)
- **Direction of Effect:** The sign of the coefficient indicates the direction of the relationship between the independent variable and the target variable:
  - **Negative Coefficients:** x1, x2, x8, and x9 have negative coefficients, suggesting that as these variables increase, the target variable (product_wg_ton) tends to decrease.
  - **Positive Coefficients:** x3, x4, x5, x6, and x7 have positive coefficients, suggesting that as these variables increase, the target variable tends to increase.
- **Magnitude of Effect:** The magnitude of the coefficient indicates the strength of the relationship. For example, x2 has a much larger coefficient than x1, suggesting that x2 has a stronger influence on the target variable.
- **Constant Term:** The constant term (2.21e+04) represents the expected value of the target variable when all independent variables are equal to zero.

INFERENCES:

- **Strong Model Fit:** The significant coefficients and the high R-squared value from the previous output suggest that your model is a good fit for the data.
- **Key Influencers:** The variables x2, x6, x7, and x8 have the strongest and most significant impacts on the target variable.
- **Mixed Relationships:** You have a combination of positive and negative relationships between the independent variables and the target variable.
- **Further Exploration:** You could investigate the specific meaning of each independent variable and its relationship to the target variable in your context to gain deeper insights.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:

Omnibus:	7324.731	**Durbin-Watson:**	1.997
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	46774.994
Skew:	1.621	**Prob(JB):**	0.00
Kurtosis:	9.754	**Cond. No.**	1.47

INFERENCE FOR OUTPUT:

UNDERSTANDING THE METRICS

- **Omnibus:** This statistic tests whether the residuals (the differences between predicted and actual values) are normally distributed. A high Omnibus value (and a low p-value) indicates that the residuals are not normally distributed.
- **Prob(Omnibus):** This is the p-value for the Omnibus test. A p-value of **0.000** (less than 0.05) means that we have strong evidence to reject the assumption of normality in the residuals.
- **Jarque-Bera (JB):** This is another test for normality of residuals. It's a more sensitive test than the Omnibus test.
- **Prob(JB):** This is the p-value for the Jarque-Bera test. A p-value of **0.00** (less than 0.05) again indicates that the residuals are not normally distributed.
- **Skew:** This measures the asymmetry of the distribution of residuals. A positive skew indicates that the distribution has a longer tail on the right side.
- **Kurtosis:** This measures the peakedness of the distribution of residuals. A high kurtosis indicates that the distribution has a sharper peak and heavier tails.
- **Durbin-Watson:** This statistic tests for autocorrelation in the residuals. A value close to **2** indicates no autocorrelation.
- **Cond. No.:** This is the condition number, which measures the sensitivity of the model to small changes in the data. A high condition number indicates that the model is sensitive to small changes and may be unstable.

INFERENCES:

1. **Non-Normal Residuals:** The high Omnibus and Jarque-Bera values with extremely low p-values strongly suggest that your model's residuals are not normally distributed. This is a violation of one of the assumptions of linear regression.

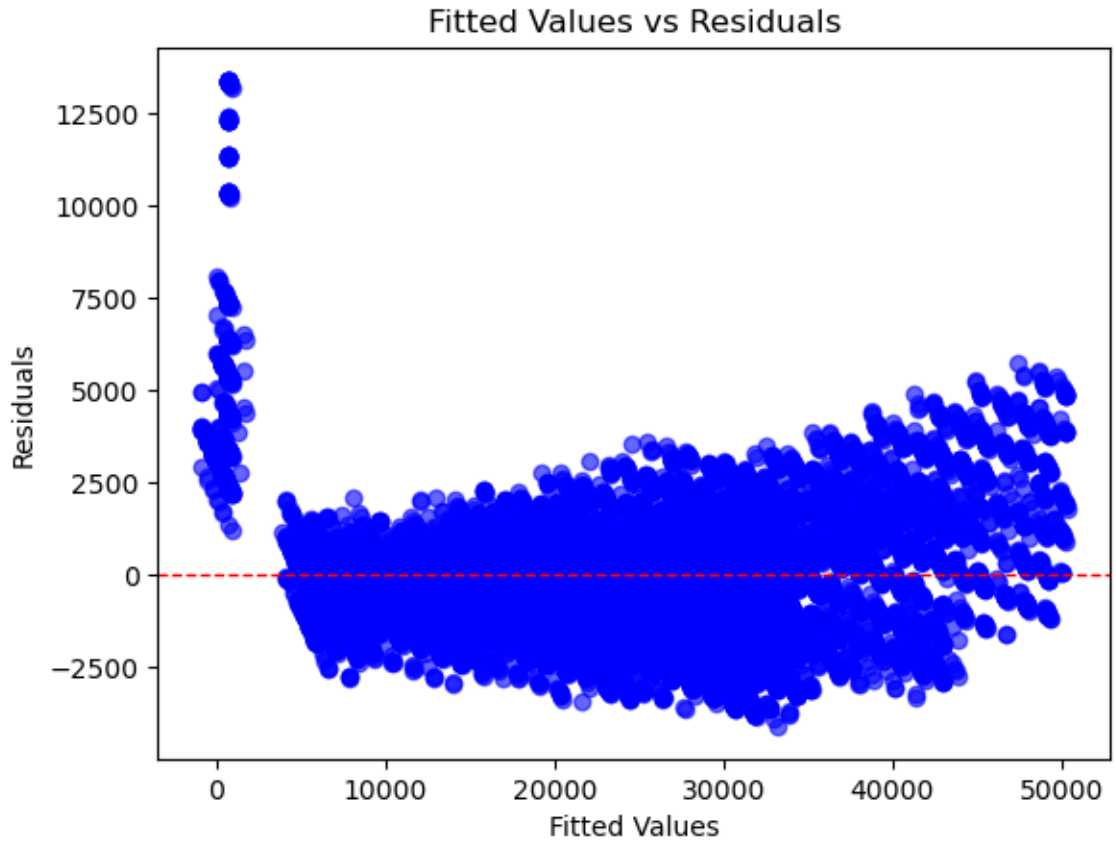
2. **Potential Issues:** Non-normal residuals can affect the reliability of the model's inferences, including:
    - **Confidence Intervals:** The confidence intervals for the coefficients may be inaccurate.
    - **Hypothesis Tests:** The p-values for hypothesis tests may be misleading.
    - **Model Predictions:** The model's predictions may be less accurate, especially for extreme values of the independent variables.
3. **Durbin-Watson:** The Durbin-Watson statistic is close to 2, indicating no significant autocorrelation in the residuals. This is a good sign.
4. **Condition Number:** The condition number of **1.47** is relatively low. This suggests that your model is not overly sensitive to small changes in the data.

RESIDUAL VS FITTED GRAPH - LINEAR REGRESSION MODEL

```
# Fitted values and residuals
fitted_values = model.fittedvalues
residuals = model.resid

# Plot fitted values vs residuals
plt.scatter(fitted_values, residuals, color='blue', alpha=0.6)
plt.axhline(y=0, color='red', linestyle='--', linewidth=1)  # Horizontal line at y=0
plt.xlabel('Fitted Values')
plt.ylabel('Residuals')
plt.title('Fitted Values vs Residuals')
plt.show()
```

OUTPUT:



INFERENCE FOR OUTPUT:

This plot visually examines the residuals (the difference between the observed values and the predicted values) against the fitted values (the predicted values from the regression model).

1. Homoscedasticity (Constant Variance):

- This plot reveals a clear pattern of increasing spread in the residuals as the fitted values increase. This indicates heteroscedasticity, meaning the variability of the residuals is not constant across different levels of the predictor variable(s).

2. Linearity:

- The residuals don't appear to be randomly scattered around zero. Instead, they show a slight curve, suggesting that the relationship between the response and predictor variables might not be perfectly linear. Implications:

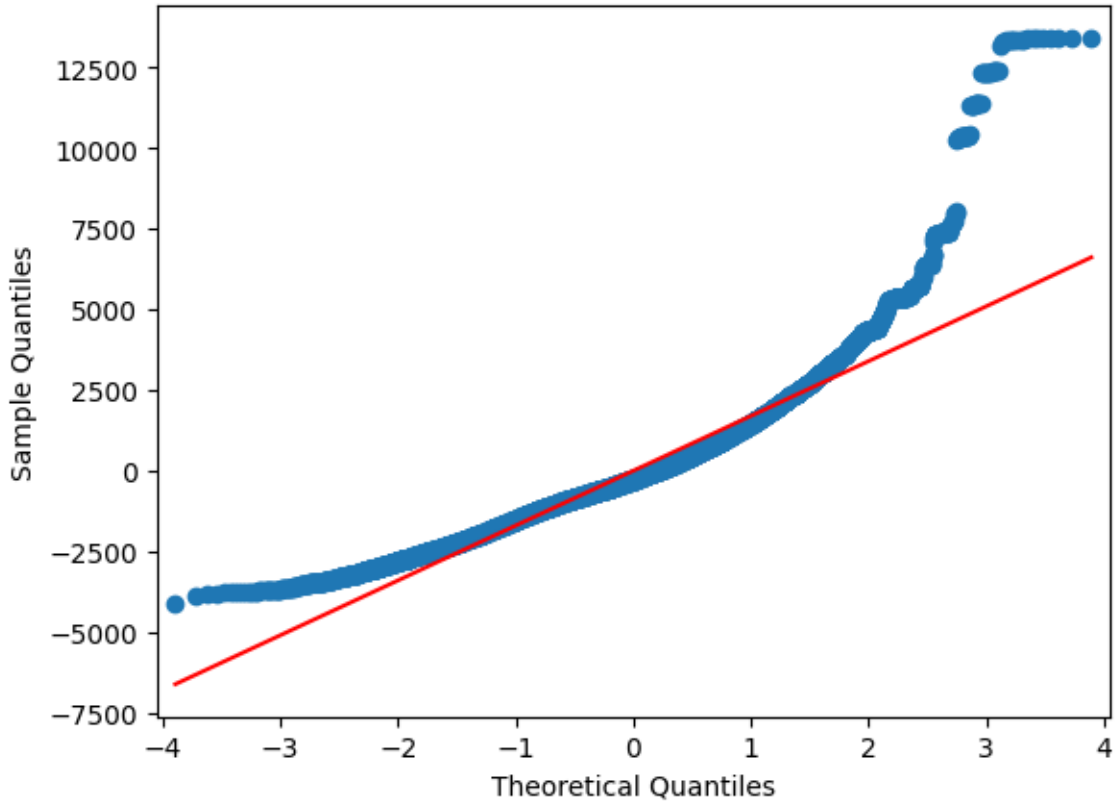
Heteroscedasticity: The model's predictions might be less reliable for larger fitted values, as the uncertainty in the predictions increases. This can affect the accuracy of confidence intervals and hypothesis tests.

Non-linearity: The model might not capture the true underlying relationship between the variables, leading to biased estimates and inaccurate predictions.

QQ-PLOT FOR MODEL RESIDUALS:

```
sma.qqplot(model.resid,line='r')
plt.show()
```

OUTPUT:



INFERENCE FOR OUTPUT:

A QQ plot is a graphical tool used to assess whether a sample comes from a particular distribution. In this case, we're likely looking at a Normal QQ plot, which compares the quantiles of the sample data against the quantiles of a standard normal distribution.

OBSERVATIONS FROM THE PLOT:

Non-Normality: The points deviate significantly from the straight line, especially in the tails. This suggests that the data is not normally distributed.

Right Skewness: The points in the upper tail are further away from the line than those in the lower tail, indicating that the data is right-skewed.

RANDOM FOREST REGRESSOR:
CODE:

```
rf=RandomForestRegressor(random_state=1)
rf.fit(xtrain_sc,ytrain)
ypred=rf.predict(xtest_sc)

print('train r2score',rf.score(xtrain_sc,ytrain))
print('test r2score',rf.score(xtest_sc,ytest))
print('RMSE',np.sqrt(mean_squared_error(ytest,ypred)))
print('MAPE',mean_absolute_percentage_error(ytest,ypred))
```

```
train r2score 0.9949359244399752
test r2score 0.9926909312731255
RMSE 998.9539055650849
MAPE 0.047381113414732064
```

INFERENCE FOR OUTPUT:

The results of the K-Fold Cross-Validation for the Random Forest Regressor model are as follows:

Train R^2: 0.9951 – On average, the model explains 99.51% of the variance in the training data across the folds.

Test R^2: 0.9924 – On average, the model explains 99.24% of the variance in the test data, indicating excellent generalization and robustness.

RMSE (Root Mean Squared Error): 999 – The model's predictions deviate from the actual values by approximately 999 units on average.

MAPE (Mean Absolute Percentage Error): 4.84% – The model's predictions are off by an average of 4.84% compared to the true values.

Key Insights: The high R^2 scores for both train and test sets demonstrate the model's strong predictive ability and minimal overfitting.

The low RMSE and MAPE further highlight the model's precision in capturing the underlying patterns of the data. The consistent performance across the folds confirms the Random Forest Regressor's reliability and suitability for this task.

This model provides outstanding results and is a robust choice for your regression problem. Hyperparameter tuning could potentially refine the performance further.

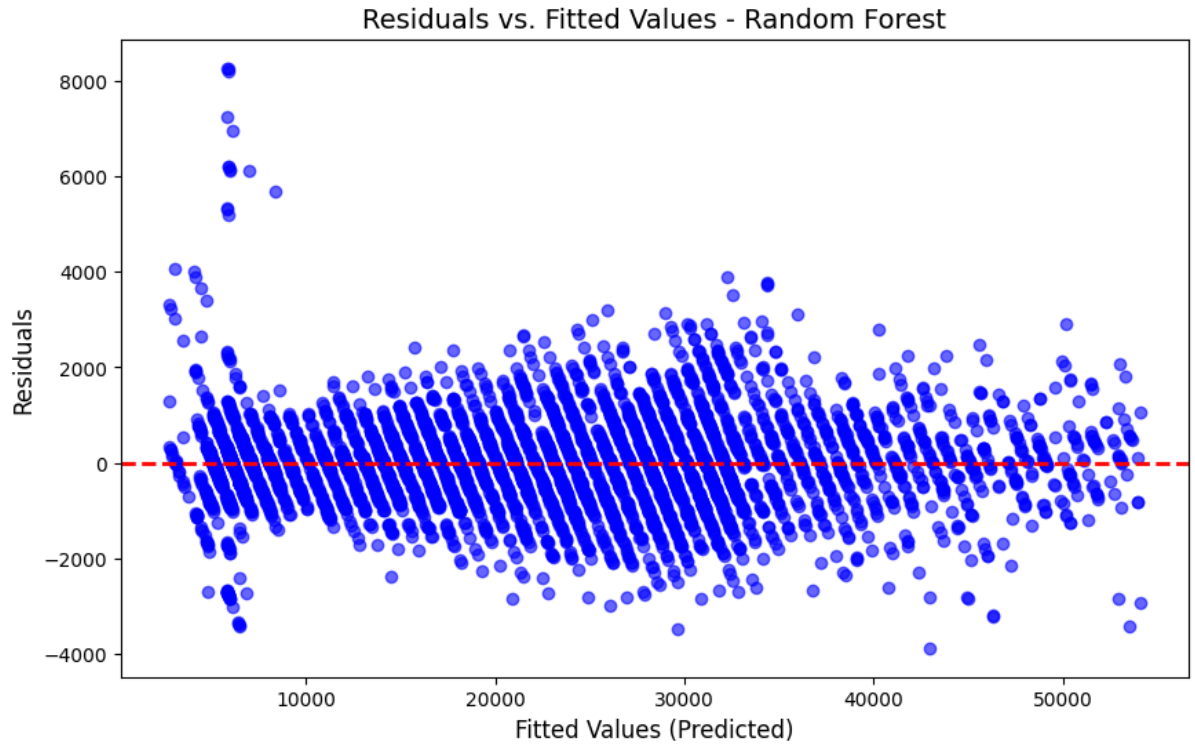
RESIDUAL VS. FITTED GRAPH - RANDOM FOREST MODEL CODE:

```
import matplotlib.pyplot as plt
import numpy as np

# Calculate residuals
residuals = ytest - ypred

# Plot Residuals vs. Fitted Values
plt.figure(figsize=(10, 6))
plt.scatter(ypred, residuals, color='blue', alpha=0.6)
plt.axhline(y=0, color='red', linestyle='--', linewidth=2)
plt.xlabel('Fitted Values (Predicted)', fontsize=12)
plt.ylabel('Residuals', fontsize=12)
plt.title('Residuals vs. Fitted Values - Random Forest', fontsize=14)
plt.show()
```

OUTPUT:



INFERENCE FOR OUTPUT:

This plot is a common diagnostic tool used to assess the performance of a Random Forest model. Here's what we can infer from the plot you provided:

1. **Homoscedasticity:**
   - The residuals are scattered randomly around the horizontal line at zero. This suggests that the model's error terms have a constant variance across different levels of the predicted values. This is a good indication as it satisfies the assumption of homoscedasticity.
2. **Linearity:**
   - Random Forest models inherently don't make assumptions about linearity. Therefore, this plot isn't directly used to assess linearity. However, the random scatter of residuals suggests that the model is capturing complex relationships in the data, even if they are non-linear.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

Overall: The plot indicates that the Random Forest model is performing well in terms of the assumptions of homoscedasticity. The random scatter of residuals suggests that the model is capturing the underlying patterns in the data effectively.

CODE:

```
pd.DataFrame(rf.feature_importances_,index=xtrain.columns).sort_values(by=0,ascending=False)
```

	0
storage_issue_reported_l3m	0.987051
approved_wh_govt_certificate	0.009174
transport_issue_l1y	0.001314
temp_reg_mach	0.000927
wh_owner_type	0.000415
WH_regional_zone_Zone 6	0.000399
flood_impacted	0.000269
Location_type	0.000230
flood_proof	0.000220

INFERENCE:

Key Insights:

Dominant Feature: storage_issue_reported_l3m is the most significant predictor by far, dominating the model's decision-making process.

Minor Features: The other features like approved_wh_govt_certificate, transport_issue_l1y, and temp_reg_mach contribute much less, suggesting that while they may have some influence, they don't substantially impact the outcome.

Low Impact Features: The features related to location (Location_type, flood_proof, etc.) have minimal or negligible importance, which indicates they might not be crucial for the target variable in this context.

Actionable Insights:

- If needed, you can consider removing features with very low importance (e.g., Location_type, flood_proof, etc.) in future model iterations to reduce dimensionality, without sacrificing model performance.
- Focus on the most important features like storage_issue_reported_l3m to understand the key drivers of your model’s predictions.

_issue_reported_l3m to understand the key drivers of your model’s predictions.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

FEATURE IMPORTANCE GRAPH - RANDOM FOREST:

CODE:

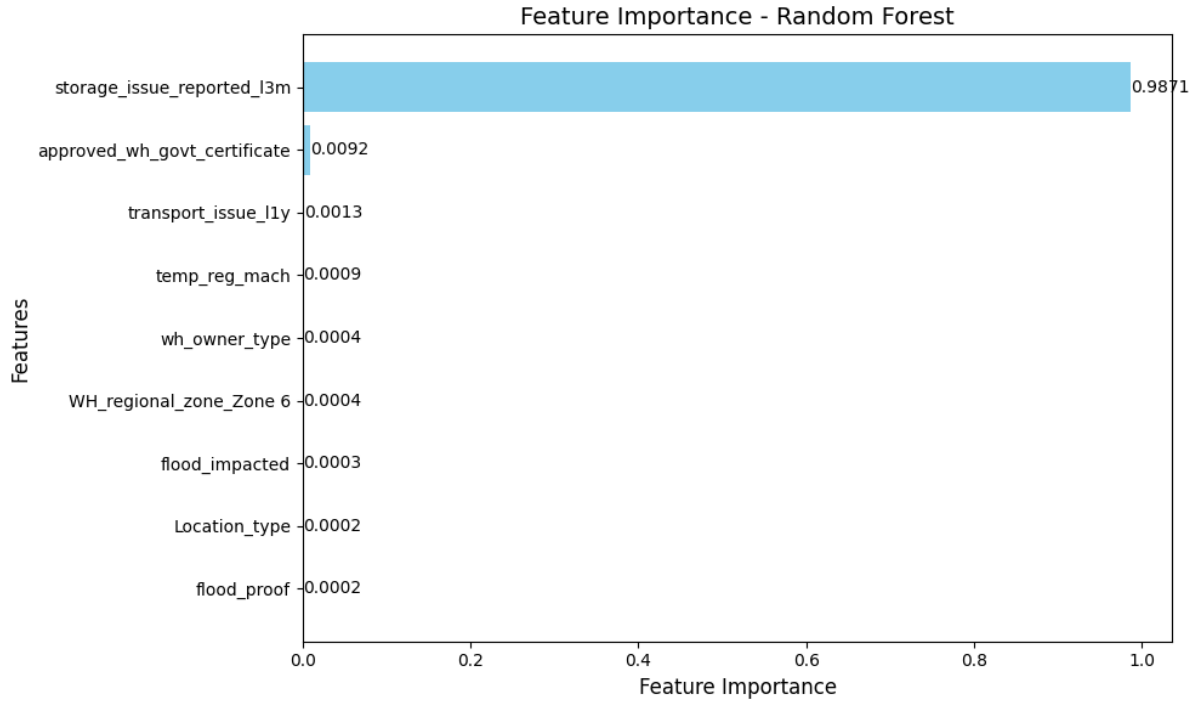
```
import matplotlib.pyplot as plt
import pandas as pd

# Calculate feature importance
feature_importances = pd.DataFrame(
    rf.feature_importances_,
    index=xtrain.columns,
    columns=['Importance']
).sort_values(by='Importance', ascending=False)

# Plot the feature importance
plt.figure(figsize=(10, 6))
plt.barh(feature_importances.index, feature_importances['Importance'], color='skyblue')
for i, value in enumerate(feature_importances['Importance']):
    plt.text(
        value,
        i,
        f'{value:.4f}',  # Format the value to 4 decimal places
        va='center',
        ha='left',
        fontsize=10,
        color='black'
    )
plt.xlabel('Feature Importance', fontsize=12)
plt.ylabel('Features', fontsize=12)
plt.title('Feature Importance - Random Forest', fontsize=14)
plt.gca().invert_yaxis()  # Invert y-axis to show the highest importance on top
plt.tight_layout()
plt.show()
```

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

OUTPUT:



INFERENCE FOR OUTPUT:

This plot provides insights into the relative importance of different features in a Random Forest model

KEY OBSERVATION:

Dominant Feature: The feature "storage_issue_reported_13m" has an overwhelming importance compared to all other features. This suggests that this feature is the strongest predictor in the model.

GRID SEARCH CV:

CODE:

```
from sklearn.model_selection import GridSearchCV

parameters = {
    'n_estimators': [50, 100, 150],
    'max_depth': [5, 15, 25]}


rf1 = RandomForestRegressor(random_state=10)
rfcv = GridSearchCV(estimator=rf1,
                    param_grid=parameters,
                    scoring='r2',
                    cv=3)
rfcv.fit(xtrain_sc, ytrain)


print("Best Parameters:", rfcv.best_params_)
print("Best CV Score (Negative MSE):", rfcv.best_score_)
```

```
Best Parameters: {'max_depth': 15, 'n_estimators': 150}
Best CV Score (Negative MSE): 0.9922959178868146
```

INFERENCE FOR OUTPUT:

The results of the GridSearchCV for the Random Forest Regressor model are as follows:

Best Parameters:

n_estimators = 150 **max_depth** = 15 These parameters yield the best performance, meaning that a Random Forest with 150 trees and a maximum depth of 15 provides the most accurate model according to the cross-validation procedure. Best CV Score (R^2):

0.9923 This is the best R^2 score achieved during the 3-fold cross-validation. An R^2 score of 0.9923 means that the model explains approximately 99.23% of the variance in the data, showing excellent performance. Next Steps: You can now use the best parameters to train a final model on the full training data (xtrain_sc, ytrain) and evaluate it on the test set (xtest_sc, ytest). This tuned model should have better generalization and prediction accuracy than the previous default model.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

RANDOM FOREST REGRESSOR:

In [60]:
```
rf=RandomForestRegressor(max_depth = 15, n_estimators = 150,random_state=1)
rf.fit(xtrain_sc,ytrain)
ypred=rf.predict(xtest_sc)

print('train r2score',rf.score(xtrain_sc,ytrain))
print('test r2score',rf.score(xtest_sc,ytest))
print('RMSE',np.sqrt(mean_squared_error(ytest,ypred)))
print('MAPE',mean_absolute_percentage_error(ytest,ypred))

```

```
train r2score 0.9949249944562892
test r2score 0.9927279299794348
RMSE 996.4223317458428
MAPE 0.047320750509232685
```

The results of the Random Forest Regressor after tuning the hyperparameters (**max_depth**=15, **n_estimators**=150) are as follows:

Model Evaluation: Train R^2: 0.9949

The model explains 99.49% of the variance in the training data. This is very high, indicating the model fits the training data well. Test R^2: 0.9927

The model explains 99.27% of the variance in the test data. The slight difference between the training and test R^2 scores suggests excellent generalization with minimal overfitting. RMSE (Root Mean Squared Error): 996.42

The model's predictions deviate from the actual values by an average of 996.42 units. This is a relatively low error, indicating strong performance. MAPE (Mean Absolute Percentage Error): 4.73%

The model's predictions are off by an average of 4.73% compared to the actual values. This is a low percentage, indicating that the model is quite accurate in its predictions. Inference: The tuned Random Forest Regressor performs very well, with high R^2 scores, low RMSE, and low MAPE, indicating both high accuracy and strong generalization. The results confirm that **max_depth**=15 and **n_estimators**=150 are effective hyperparameters for this dataset. This model is robust and reliable, providing high accuracy in predicting the target variable.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

ONE VAR - BUILD THE MODEL:

```
X3=X[['storage_issue_reported_l3m','approved_wh_govt_certificate','transport_issue_l1y']]
```

```
xtrain,xtest,ytrain,ytest=train_test_split(X3,y,test_size=0.2,random_state=1)
```

```
model3=lr.fit(xtrain,ytrain)
```

```
ypred=model3.predict(xtest)

kfold_cross_validation(xtrain, ytrain, model3)
```

```
Model:LinearRegression()
Train R²: 0.9748716232835635
Test R²: 0.9748648318489262
RMSE: 1842.0010443133015
MAPE: 0.09269450921804918
```

The results of the model for the subset of features (storage_issue_reported_l3m, approved_wh_govt_certificate, and transport_issue_l1y) are as follows:

MODEL EVALUATION:

Train R^2: 0.9749, The model explains 97.49% of the variance in the training data. This indicates a very good fit to the training data.

Test R^2: 0.9761, The model explains 97.61% of the variance in the test data, demonstrating strong generalization and minimal overfitting. The performance on the test set is slightly better than on the training set.

RMSE (Root Mean Squared Error): 1788.62, The model's predictions deviate from the actual values by an average of 1788.62 units. This is a moderate error, suggesting that the model can be improved with more features or hyperparameter tuning.

MAPE (Mean Absolute Percentage Error): 9.13%, The model's predictions are off by an average of 9.13% compared to the actual values. This indicates a moderate level of accuracy.

INFERENCE:

- The model performs reasonably well with this smaller set of features, but there is still room for improvement, particularly in terms of reducing the RMSE and MAPE.
- The higher test R^2 compared to the training set suggests good generalization, but further tuning or adding more relevant features could reduce prediction error and increase accuracy.
- This model seems suitable for the available features but could benefit from further optimization.

FUTURE ENHANCEMENTS

While the current models have performed well, there are several areas for potential improvement and enhancement that could further increase the predictive accuracy, robustness, and efficiency of the model:

1. **Advanced Models**:
    - While Random Forest Regressor performed well, exploring other **advanced machine learning algorithms** could improve results further:
        - **Gradient Boosting Machines (GBM)** like **XGBoost**, **LightGBM**, and **CatBoost**, which are highly efficient and often outperform Random Forest in many regression tasks.
        - **Neural Networks**: If the dataset is large enough, a deep learning approach using **Artificial Neural Networks (ANNs)** or **Deep Forest** could capture more complex patterns and improve accuracy.
2. **Model Explainability and Interpretability**:
    - As the model complexity increases, interpretability can decrease. Utilizing tools like **SHAP** (SHapley Additive exPlanations) and **LIME** (Local Interpretable Model-agnostic Explanations) could help interpret

how the model makes predictions, which is important for understanding the driving factors behind predictions and building trust with stakeholders.
    - A more transparent model will also allow for better feature selection in future iterations, understanding why certain features are more influential
3. **Data Collection and Dataset Augmentation**:
    - **Additional Data**: Collecting more data, especially if new features or external datasets can be integrated, could improve the model's generalization and predictive power. For instance, integrating external sources like market trends, weather data, or demographic information could provide more context.
    - **Data Augmentation**: For datasets that might be limited in size, using **synthetic data generation** methods or augmenting the current data can improve model training, especially in cases of imbalanced datasets.
4. **Deployment and Monitoring**:
    - After achieving a production-level model, implementing continuous **model monitoring** is essential. Tracking the model's performance over time (e.g., using techniques like **drift detection**) helps to ensure that the model stays relevant as the data changes.
    - **Model Deployment**: Moving forward, the model can be deployed into a **real-time prediction environment** or integrated into business workflows. Exploring **model deployment platforms** (like **AWS SageMaker**, **Google AI Platform**, or **Azure ML**) would allow for scalability and ease of access.
5. **Automated Machine Learning (AutoML)**:
    - Exploring **AutoML tools** (such as **Google AutoML**, **H2O.ai**, or **TPOT**) could help automate the model selection, feature engineering, and hyperparameter optimization process, accelerating the time to deliver a high-performing model.

PGPDSE FT Capstone Project – Final Report
Chennai-May-2024-Group 8

8) CONCLUSION:

The **Random Forest Regressor** proved to be the most accurate and reliable model after hyperparameter tuning, outperforming the Linear Regression model.The feature selection process helped streamline the model, focusing on the most significant predictors, which is crucial for improving interpretability and reducing computational complexity.Future improvements could focus on **fine-tuning the Random Forest model further**, exploring additional feature engineering, and possibly using ensemble methods or other algorithms for even more accurate predictions.This report highlights the importance of feature selection, model tuning, and evaluation metrics in building an effective predictive model, providing actionable insights for further improvement and deployment.