Dec 2024

Optimizing Supply Chain Efficiency with Geospatial Data

Presented to
Savvy Barnes

Presented by
Mahnoor Sheikh, Andrew John, AB Basit

TABLE OF CONTENTS

Problem Statement

Supply Chain Management (SCM) encompasses the management of the flow of goods, from the procurement of raw materials to the final delivery to consumers. All SCM models share common goals: streamlining operations, improving customer satisfaction, and lowering operational expenses to maximize profits. A key component of SCM, logistics and transport management, plays a direct role in achieving these objectives. However, frequent disruptions in this process cost organizations their revenue and trust, and society its timely access to essential needs. Persistent disruptions impacting the flow of goods also stifle economic activity.

Season, mode of delivery and type of product are critical factors of logistic and transport management, making last-mile delivery the most vulnerable stage. Adverse weather conditions like snow, storms, or heavy rains can significantly hamper operations, leading to delayed or failed deliveries. With approximately 25-30% of failed deliveries across industries being associated with weather-related issues, addressing these challenges is crucial to meet the goals of the SCM process. One proposed solution is integrating the use of geospatial data to determine optimal delivery time and mode. A demand forecasting model that accurately predicts future demand based on weather and satellite imagery. Visualizations help companies understand demand fluctuations driven by environmental factors. Insights that companies can use to optimize their supply chain and inventory management strategies.

Data Overview

Dataset Details:

- Total Rows: 9,900
- Total Columns: 90 (filtered to 20 relevant columns).

Selected Columns:

1. **Shipment Details:**
   - Days for shipment (scheduled)
   - Late_delivery_risk
   - Order Item Quantity
2. **Categorical Variables:**
   - Category Name, Customer City, Department Name
   - weather_preciptype, weather_conditions, weather_description
3. **Weather Metrics:**
   - weather_tempmax, weather_tempmin, weather_temp
   - weather_precip, weather_snow, weather_snowdepth
   - weather_windspeed, weather_cloudcover, weather_visibility, weather_severerisk
4. **Date Information:**
   - shipping_date_day_month

Data Types:

- Numerical Columns: 13
- Categorical Columns: 7

Methodology - Initial Data Analysis (IDA)

Numerical Summary:

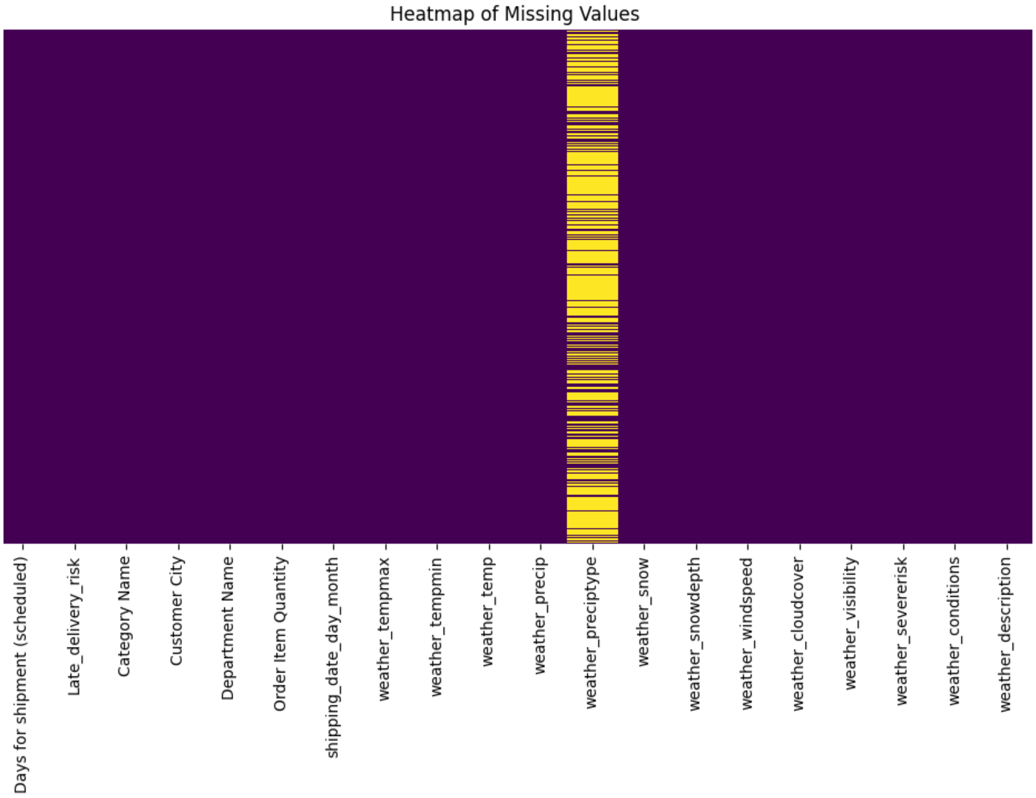
Feature	Mean	Std. Dev.	Min	25%	50%	75%	Max
Days for shipment (scheduled	2.92	1.40	0.00	2.00	4.00	4.00	4.00
Late_delivery_risk	0.53	0.50	0.00	0.00	1.00	1.00	1.00
Weather Temp Max	22.77	8.80	-8.10	17.20	23.00	28.90	47.60
Weather Precipitation	2.46	7.96	0.00	0.00	0.00	0.30	112.81

Key Observations:

- Deliveries are typically scheduled within 2–4 days.
- Late_delivery_risk is balanced across the dataset (53% risk).
- Weather metrics show varied distributions, with some outliers in precipitation and snow depth.

Missing Values:

- Missing values in weather_preciptype (68.34% missingness) were filled with 'Unknown'.

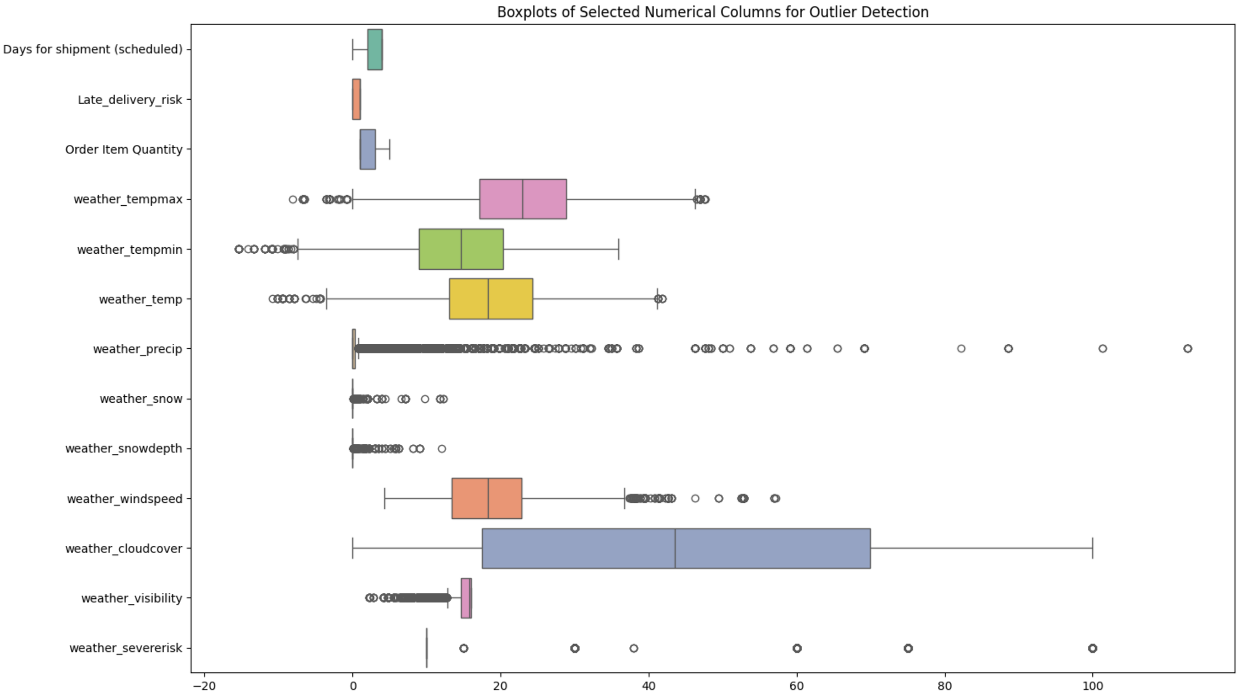


Data Visualizations

Univariate Analysis:

1. **Numerical Data:**

- Histograms show varied distributions for weather metrics like temperature and precipitation.
- Boxplots reveal outliers in weather_precip and weather_snow.

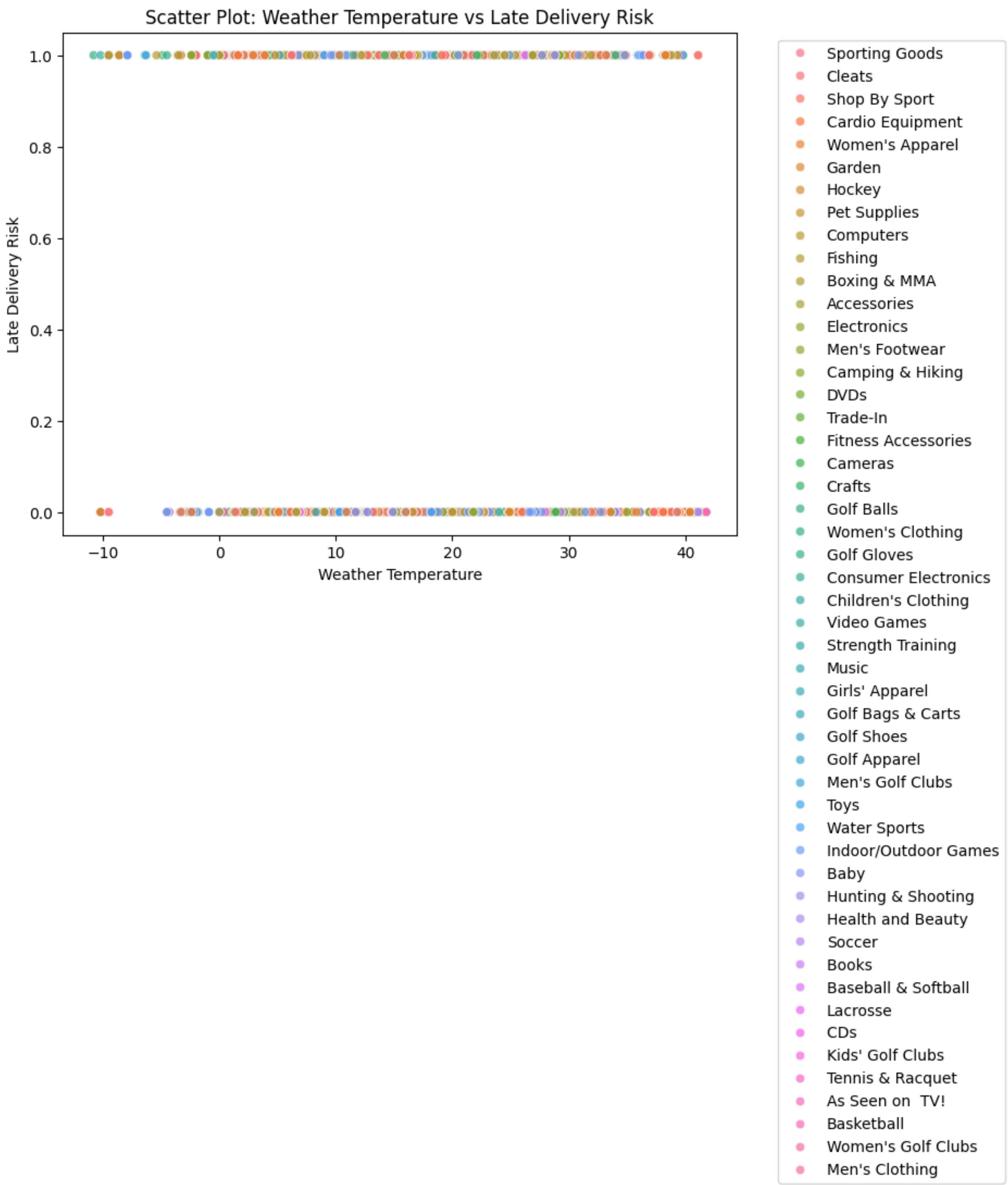


2. **Categorical Data:**

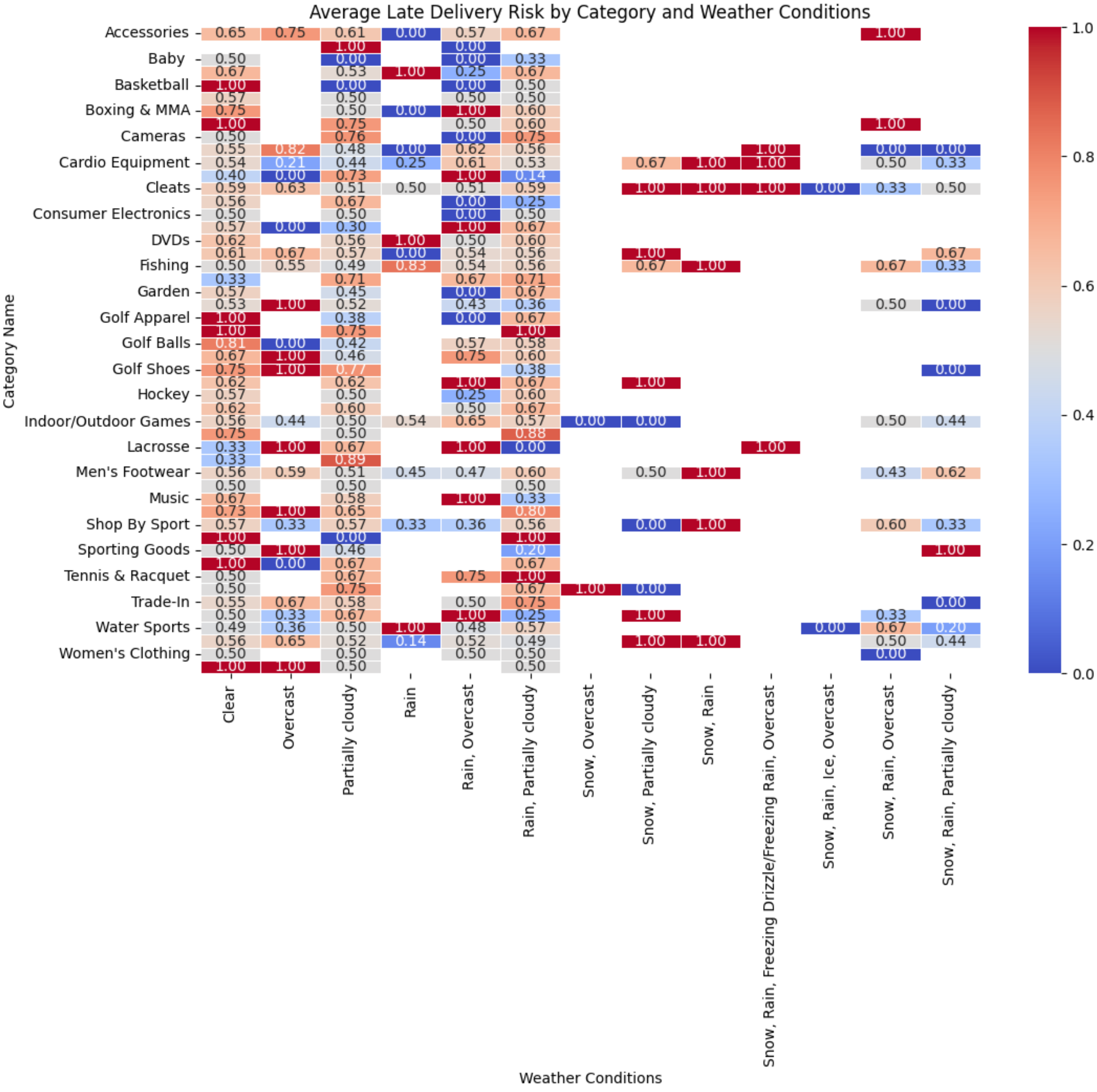
- Most frequent product category: “Sporting Goods”.
- Predominant weather condition: “Partially Cloudy”.

Multivariate Analysis:

1. **Scatter Plot:** No significant correlation between weather_temp and Late_delivery_risk.



2. **Heatmap:** Strong correlations among weather metrics like weather_temp, weather_tempmax, and weather_tempmin.



Key Insights:

Shipment Trends:

- Delivery risk is not significantly influenced by temperature alone.
- Most deliveries are on-time, but late deliveries still account for 53%.

Weather Influence:

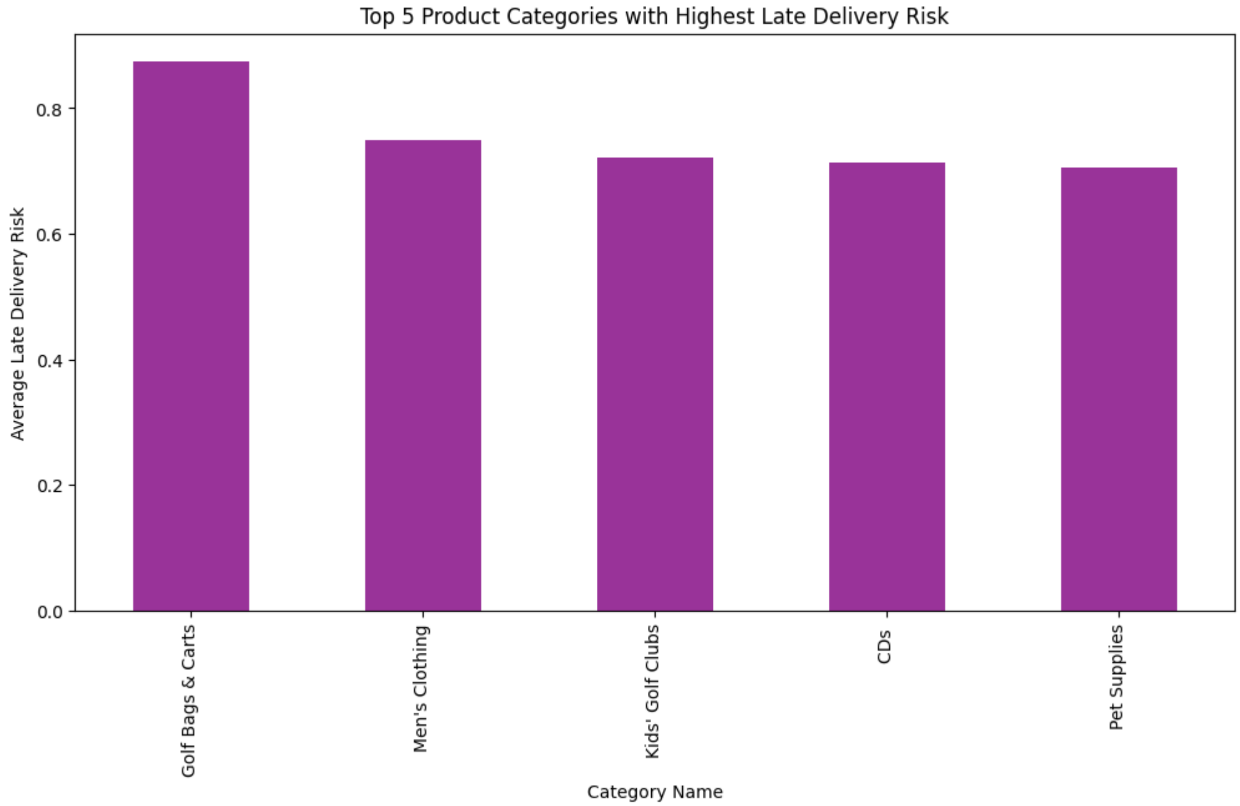
- Severe weather events increase delivery risks.
- Precipitation and snow depth rarely reach extreme values, reducing their overall impact.

Geographic Impact:

- Cities with logistical challenges show higher late delivery risks.

Product Categories:

- Categories like “Sporting Goods” and “Cleats” exhibit higher delivery risks.



Additional Analysis and Insights:

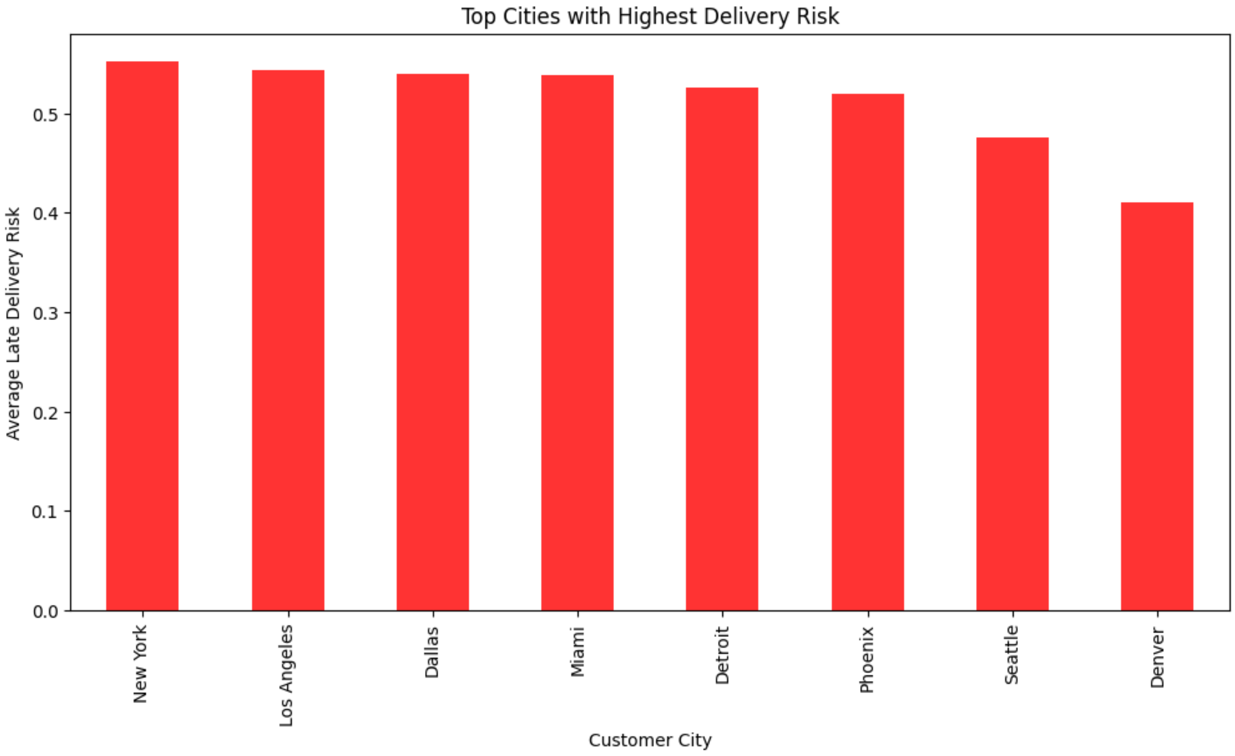
1. Weather Features (Wind Speed and Visibility)

- **Wind Speed:**
  - The highest average wind speeds are observed in weather conditions like "Rain, Overcast" and "Partially Cloudy," indicating these conditions might cause logistical challenges.

- **Visibility:**
  - Visibility remains consistently high (~16 km) across most weather conditions, except for specific conditions like "Rain, Overcast," where it decreases slightly.

2. Customer Demographics and Delivery Success

- **Top Cities with Delivery Risks:**
  - The cities with the highest delivery risks include locations with logistical or weather-related challenges. These cities should be prioritized for operational improvements.



Predictive Modeling

Our objective is to develop a Last-Mile Delivery strategy that accounts for potential weather disruptions to enhance the rate of successful deliveries.

To achieve this, we leverage advanced statistical and machine learning techniques to predict the likelihood of late delivery. By building this predictive model, we can anticipate whether a delivery will succeed on a specific day and proactively take measures if it is likely to fail, thereby reducing operational costs.

I. Feature Engineering

Encoding

Since we have 6 categorical features of nominal nature in our chosen columns, we will have to proceed with encoding to make them compatible with further analytical techniques. First we assign numeric labels using Label Encoder and then use Binary Encoding to transform our features. Repetitive columns are then dropped from the dataset to make it cleaner. This way the bias (higher labels given more weightage) introduced by label encoding is removed. Binary Encoding can be more efficient than One-Hot Encoding since it generates fewer features, and our data is already of a high dimension with variables having many categories.

Handling datetime64 feature 'shipping_date_day_month'

The variable 'shipping_date_day_month' cannot be directly handled by the models, hence we break it up into two new columns: 'shipping_day' and 'shipping_month'. Since we are using data for a single year, the year part of the values is not significant and hence ignored.

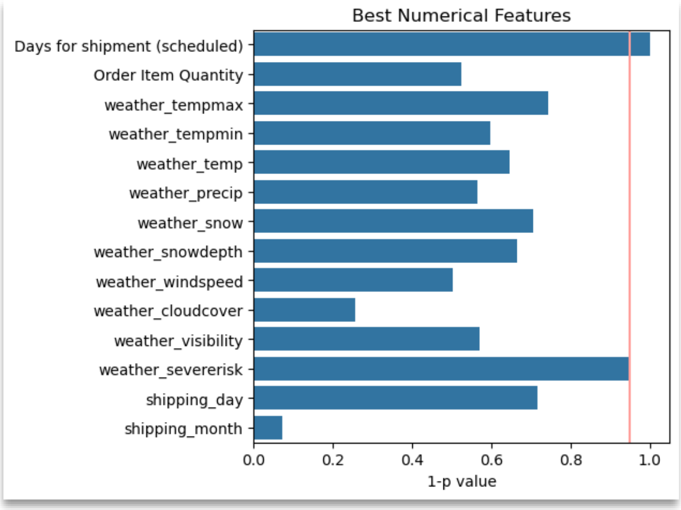
Identifying the most relevant features ensures that our model focuses on the most impactful predictors of our target variable: 'Late_delivery_risk'. For this, we execute two statistical tests.

T-test for Numerical Features

We use a T-test to determine if the mean values of numerical features significantly differ between successful (Late_delivery_risk = 0) and failed (Late_delivery_risk = 1) classes on a 95% confidence interval. This is calculated as follows:

$$t = \frac{\bar{S} - \bar{F}}{\sqrt{\frac{s_S^2}{n_S} + \frac{s_F^2}{n_F}}}$$

$\bar{S}$ = mean of successful deliveries; $\bar{F}$ = mean of failed deliveries; s_S^2, s_F^2 = variances of both classes; n_S, n_F = sample sizes of both classes



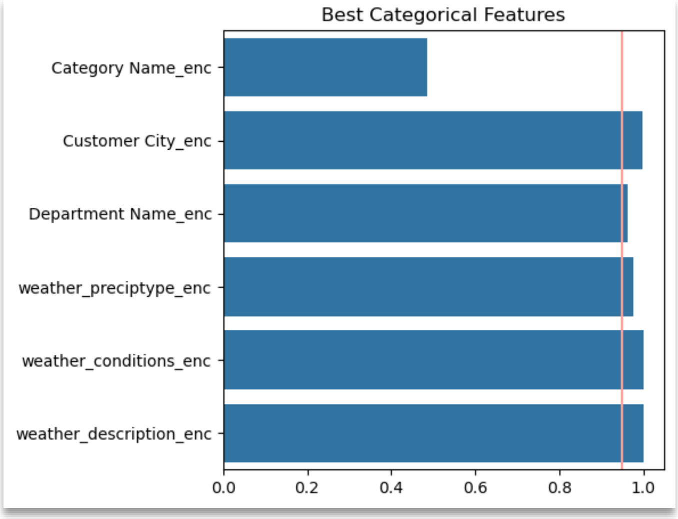
For the 95% confidence interval, if the t-statistic for a feature corresponds to a p-value < 0.05, we conclude the feature significantly impacts Late_delivery_risk. From the bar chart, we can conclude ‘Days for shipment (scheduled)’ and ‘weather_severerisk’ to strongly influence the difference between the classes and ‘shipping_month’ to be the least valuable predictor for it.

Chi-Square test for Categorical Features

We use a chi-square test to determine how independent the categorical features are of the target variable on a 95% confidence interval. This is calculated as follows:

$$X^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

O_i = observed frequency; E_i = expected frequency (if independence assumed)

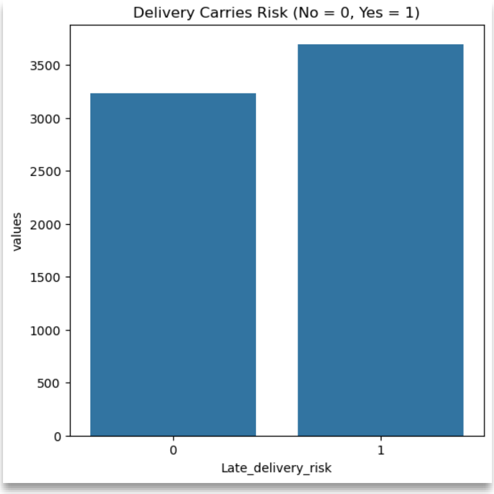


For the 95% confidence interval, a high chi-square X^2 value for a feature with p-value < 0.05 suggests that the feature is strongly associated with Late_delivery_risk. From the bar chart, we can conclude all categorical variables, except ‘Category Name_enc’, to be strongly associated with the target variable.

The dataset is then split into training (70%) and testing (30%) subsets to ensure robust model evaluation.

II. Data Preprocessing

Handling Class Imbalance



The plot illustrates the distribution of classes in the training set. The majority class "failed deliveries" makes up approximately 53% samples, whereas the minority class "successful deliveries" has 47% samples. Data is approximately balanced. Hence, implementation of resampling techniques for the training set are not required.

Z-Score Standardization

Numerical features in the training set were standardized using StandardScaler to ensure all numerical features have a comparable scale. This prevents models like Logistic Regression from being biased by features with larger magnitudes and helps satisfy the normality assumption required by the Gaussian Naïve Bayes model.

The StandardScaler class within the sklearn.preprocessing module employs Z-score standardization to transform the data. This transforms each feature to have a mean of 0 and a standard deviation of 1, ensuring a uniform scale and normal distribution. This is calculated as follows:

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

X = Original value; μ = feature mean; σ = feature standard deviation

III. Modeling and Results

Evaluation Metrics

Before diving into the models used, we lay out the accuracy metrics employed for each model’s evaluation. Results have been interpreted using a confusion matrix to calculate classification metrics (Accuracy, Precision, Recall, F1-score), Cross-Validation Accuracy, and Learning Curve. The computations for each are outlined as follows:

1. *Confusion Matrix* provides counts for each of the following:

- *True Positives (TP)*: Correctly predicted failed deliveries.
- *False Positives (FP)*: Incorrectly predicted failed deliveries.
- *True Negatives (TN)*: Correctly predicted successful deliveries.
- *False Negatives (FN)*: Incorrectly predicted successful deliveries.

2. *Accuracy* = $\frac{TP+TN}{TP+TN+FP+FN}$

3. *Precision* = $\frac{TP}{TP+FP}$

4. *Recall* = $\frac{TP}{TP+FN}$

5. *F1-Score* = 2 * $\frac{Precision * Recall}{Precision + Recall}$

6. *Cross-Validation Accuracy* is computed by splitting the training and test data into 5 (chosen by us) equal subsets (folds), then training the model on 4 randomly chosen subsets to test on the remaining test set. For each simulation, the accuracy is calculated and finally averaged to provide the Cross-Validation Accuracy with the confidence interval.

7. *Learning Curve* displays the training and validation (accuracy on unseen test data) accuracy as a function of the training set size.

<u>Logistic Regression</u>

Logistic Regression extends linear regression by applying the sigmoid function to transform linear predictions into probability values bounded between 0 and 1, instead of outputting a real-valued number. Graphically, it fits a straight line in a multi-dimensional space with each feature having an axis. The line divides the space into two regions for each class of the binary target variable. The closer a data point is to the line, the less confident is a prediction. This methodology makes Logistic Regression well suited for binary classification tasks. To illustrate:

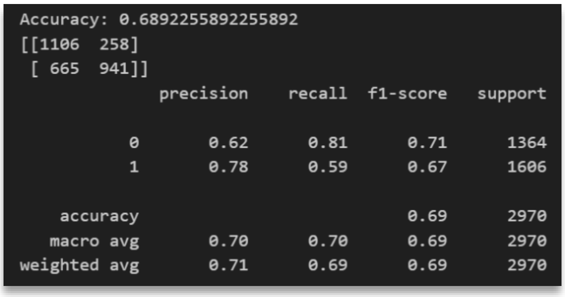
Linear Regression: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$

Logistic Regression: $P(Y = 1|X) = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\cdots+\beta_n X_n)}}$

Y = Target variable; X_1 *to* X_n *= Features in data;* β_0 *= Intercept term;* β_1 *to* β_n *= Coefficients for features*

Here we use Logistic Regression to model the probability of the class 'Late_delivery_risk == 1' (failed delivery). To prevent overfitting, we apply a technique called L2 regularization. This discourages the model from assigning very large weights to any single feature.

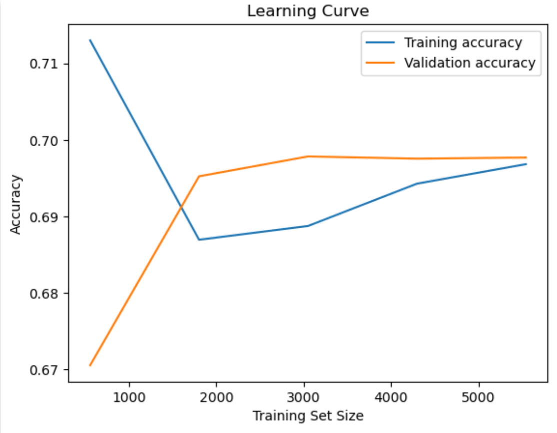
Results:



```
Accuracy: 0.6892255892255892
[[1106  258]
 [ 665  941]]
              precision    recall  f1-score   support

           0       0.62      0.81      0.71      1364
           1       0.78      0.59      0.67      1606

    accuracy                           0.69      2970
   macro avg       0.70      0.70      0.69      2970
weighted avg       0.71      0.69      0.69      2970
```



Model accurately predicted 69% observations overall. It performs better at identifying class 0 (successful deliveries), as recall is higher compared to class 1 (failed deliveries). Recall=0.59 for class 1 is relatively low since the model misses 41% of actual failed deliveries. However, Precision=0.78 is higher for class 1 indicating fewer successful deliveries are incorrectly labeled as failed. The F1-score for both classes is balanced but low, indicating need for a more robust model.

The cross-validation accuracy is *0.6976911976911977 ± 0.012946867564881948*, indicating the model correctly predicts approximately 70% outcomes on average, and ± 1.29% standard deviation suggests that the model's performance is relatively stable across different subsets.

Finally, the training and testing accuracy converges as the traning set size increases, suggesting a good balance bias-variance-tradeoff for larger datasets.

Naïve Bayes Classifier

Naïve Bayes algorithm employs Bayes Theorem to compute the probability of a class given feature data as evidence. This can be expressed as:

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

C: Class C (failed or successful deliveries); X = feature set

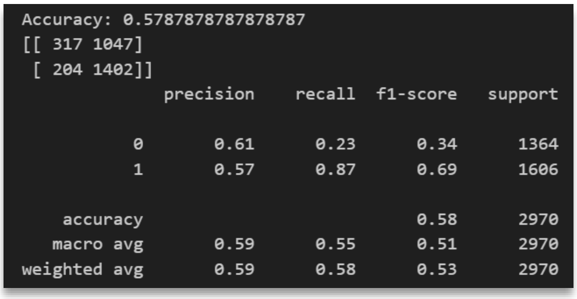
*A*ll features are assumed to be conditionally independent given the class, i.e.

$$P(X|C) = P(X_1, X_2, \dots, X_n|C) = P(X_1|C)P(X_2|C) \dots P(X_n|C)$$

Here we use Gaussian Naïve Bayes model for continuous features and Multinomial Naïve Bayes to model categorical features. The former assumes that the features follow a Gaussian distribution for each class (i.e. normally distributed), making it useful for continuous data. It calculates the posterior probability $P(C|X)$ for each class of the target variable and outputs the class with the highest probability. The latter assumes features follow a Multinomial distribution and is useful for features containing categorical data. The probability computation follows a similar approach to the Gaussian model.

An ensemble method is employed to merge the predictions of both models into a final prediction. Both prediction values are averaged, with the model assigning more weight to consistent predictions from both classifiers.

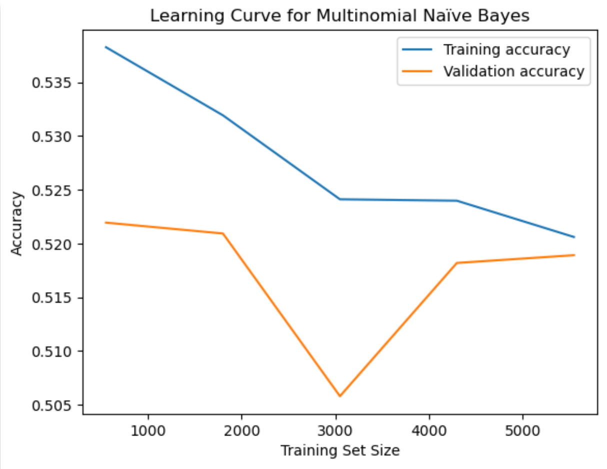
Results:

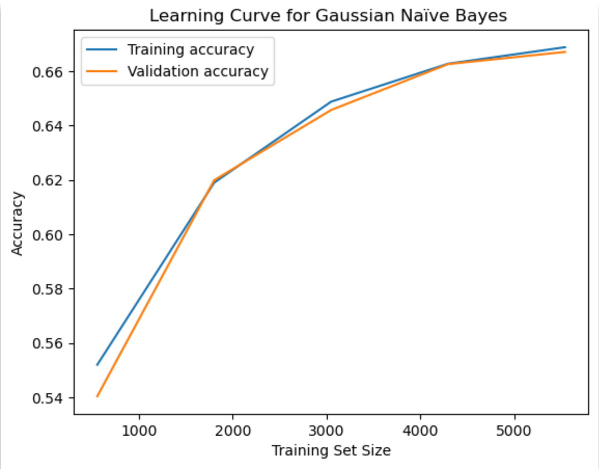


```
Accuracy: 0.5787878787878787
[[ 317 1047]
 [ 204 1402]]
              precision    recall  f1-score   support

           0       0.61      0.23      0.34      1364
           1       0.57      0.87      0.69      1606

    accuracy                           0.58      2970
   macro avg       0.59      0.55      0.51      2970
weighted avg       0.59      0.58      0.53      2970
```





Overall, the model performs poorly compared to Logistic Regression by predicting 58% of total observations accurately. In comparison to the prior model, it performs better at identifying failed deliveries due to its high Recall=0.87 for class 1, but performs poorly for successfull deliveries with extremely low recall=0.23 and F1-score=0.34 for Class 0. Precision=0.57 also greatly reduces for class 1 indicating more successful deliveries are incorrectly labeled as failed in comparison to Logistic Regression. The F1-score for both classes differs greatly, indicating the model to be biased towards class 1.

The cross-validation accuracy for Multinomial NB is *0.5189033189033189 ± 0.013956144202741912* and for Gaussian NB is *0.667099567099567 ± 0.03554473139713673*. So the Gaussian model for conitnuous features performs better than the Multonimal model and both models' performance is relatively stable across different subsets.

Finally, the training and testing accuracy for Multinomial NB converges as the training set size increases, suggesting a good balance bias-variance-tradeoff for larger datasets. Whereas, for the

Gaussian Naïve Bayes model, both validation and traning accuracies are comparable throughout irrepective of the dataset size.

Confidence Intervals and Bootstrapping

The Confidence Interval (CI) and Bootstrapping techniques are statistical methods used to estimate the uncertainty or variability around a population parameter (in this case, the mean of "Days for Shipment (Scheduled)"). Here's why we used both methods:

- ***Confidence Interval (CI):***

A confidence interval provides a range of values that is likely to contain the true population parameter (e.g., the true average days for shipment).

a. <u>Purpose of Confidence Interval</u>:

- Estimate Precision: It gives us an idea of how precise our estimate of the population mean is. The narrower the interval, the more precise our estimate.
- Decision Making: When making business or operational decisions, we often need to know not just the estimate (e.g., the average shipment days), but also how much uncertainty is involved in that estimate. A CI helps quantify that uncertainty.
- Assess Reliability: By calculating a 95% CI, we can be confident that if we took many samples, 95% of those samples' confidence intervals would contain the true population mean.

b. <u>Example Use Case</u>:

- If the CI for the "Days for Shipment (Scheduled)" is narrow and tightly clustered around a value (e.g., 2.89 - 2.94 days), it suggests that the shipping time is quite consistent across all shipments. This consistency is valuable for logistics planning.

c. <u>The confidence interval for the mean is calculated as:</u>

The confidence interval for the mean is calculated as:

$$\mathbf{CI = \bar{x} \pm t * (s / \sqrt{n})}$$

- where:
  - **$\bar{x}$:** Sample mean
  - **t:** Critical value from the t-distribution for the desired confidence level (e.g., 0.975 for 95%)
  - **s:** Sample standard deviation
  - **n:** Sample size

a. Bootstrapping:

Bootstrapping is a resampling method that allows us to estimate the distribution of a statistic (e.g., mean, median, variance) without making assumptions about the data's underlying distribution. In this case, we used bootstrapping to estimate the mean of "Days for Shipment (Scheduled)" and calculate its bootstrap confidence interval.

a. Purpose of Bootstrapping:

- No Distribution Assumptions: Unlike traditional methods (such as t-tests or normality-based confidence intervals), bootstrapping doesn't assume the data follows a specific distribution (like the normal distribution). This makes it useful when we can't guarantee the data is normally distributed or when the sample size is small.
- Assess Variability: Bootstrapping helps us understand the variability in the estimate (in this case, the mean) by simulating the sampling process thousands of times. The distribution of bootstrapped sample means provides an empirical way to quantify this variability.
- Alternative to Parametric Methods: If the data doesn't meet the assumptions of traditional methods (e.g., normality), bootstrapping offers a more flexible alternative. It can be applied in scenarios where parametric methods (like calculating a CI using the t-distribution) might not be valid.

b. Example Use Case:

• For this dataset, bootstrapping helps us validate the estimate of the average shipment time (mean) without relying on assumptions about the underlying distribution. It also provides a bootstrap confidence interval to understand the range of plausible values for the population mean.

c. For Bootstrapping, using the percentile method:

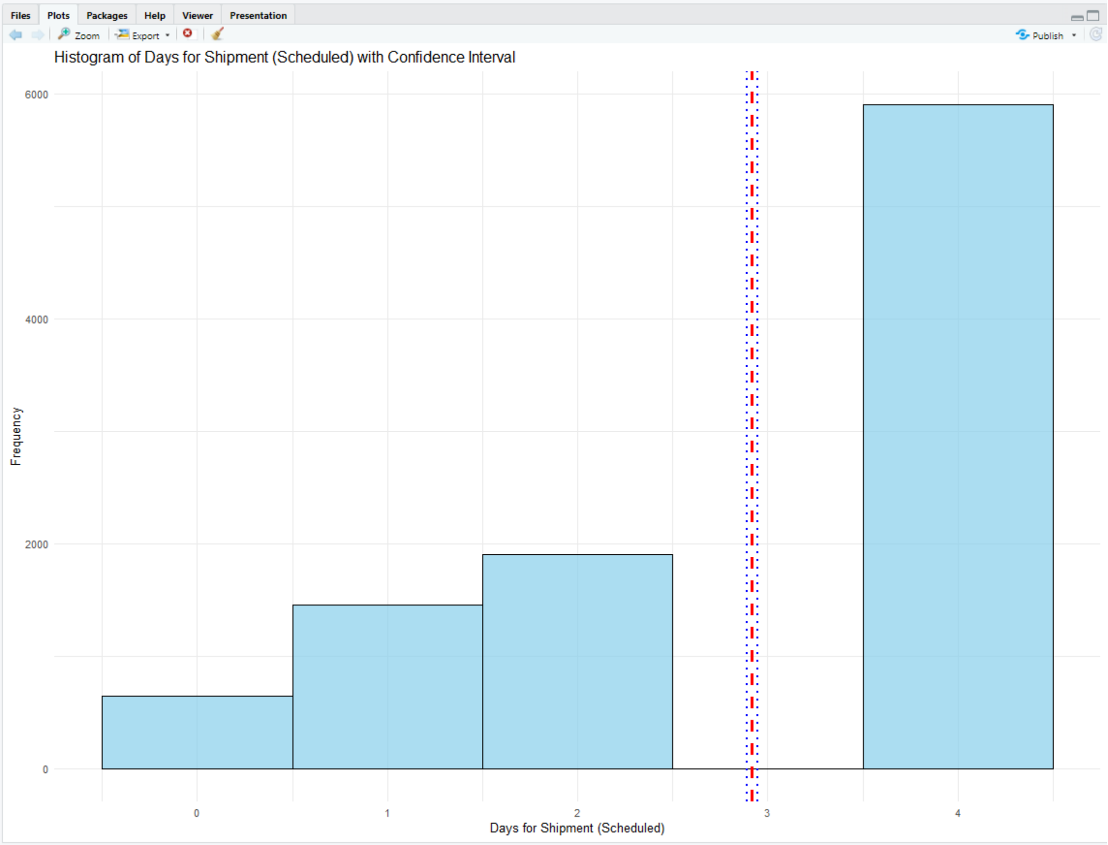
- Generate R bootstrap samples (with replacement) from the original dataset.
- Compute the sample statistic (e.g., mean) for each bootstrap sample: θ*̂_b, b = 1, 2, ..., R
  - Where b is the bootstrap sample index, and θ*̂_b represents the computed statistic for the b-th sample.
- Sort the bootstrap replicates θ*̂_b in ascending order.
- The 100(1 - α)% bootstrap confidence interval is given by: CI = [θ*̂_(⌊R * α/2⌋), θ*̂_(⌊R * (1 - α/2)⌋)]
  - Where:
  - α: Significance level (e.g., 0.05 for a 95% CI).
  - ⌊R * α/2⌋ and ⌊R * (1 - α/2)⌋: Indices of the lower and upper percentiles in the sorted bootstrap replicates.

Traditional CI (based on the t-distribution) assumes normality of the data, but bootstrapping is non-parametric and doesn't require such assumptions. Using both methods gives us a more comprehensive understanding of the uncertainty around the mean. The CI provides a straightforward statistical interval based on a well-known theory, while bootstrapping offers a more flexible, empirical approach.

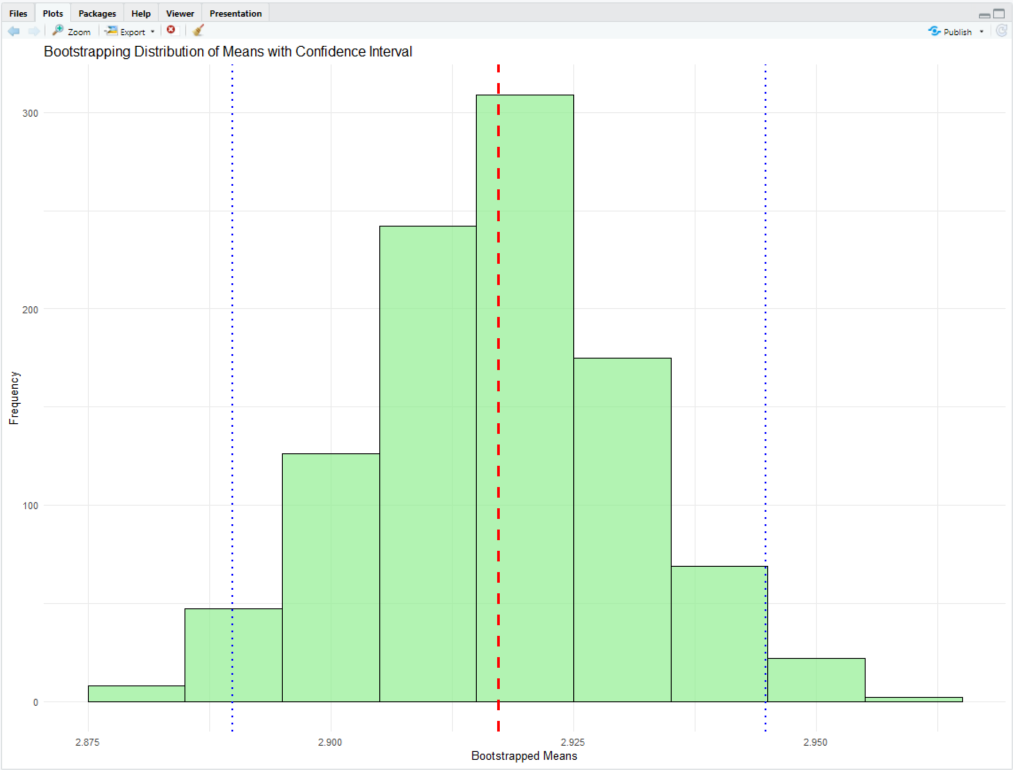
Verification: If the results from both methods align closely, it strengthens the reliability of the findings (i.e., both the traditional and bootstrapped confidence intervals suggest that the true mean is around 2.92 days).

CI helps to quantify the uncertainty of our mean estimate and is useful for decision-making in a business context. Meanwhile bootstrapping provides a more robust, assumption-free method for estimating the confidence interval, especially useful when the normality assumption doesn't hold. By combining these two methods, we ensure that our estimate of the average shipment time is reliable, accurate, and not overly sensitive to underlying distributional assumptions.

Based on our analysis, the histogram shows the distribution of "Days for Shipment (Scheduled)," with most orders scheduled for 4 days, indicating a right-skewed distribution. The red dashed line represents the mean (~2.92 days), and the blue dotted lines mark the 95% confidence interval (2.89 to 2.94), indicating a precise estimate of the average shipment days. Despite the skewness, the average is close to 3 days, suggesting that while most orders are scheduled for 4 days, there are fewer orders with shorter durations.


Files
Plots
Packages
Help
Viewer
Presentation
Zoom
Export
Publish
Histogram of Days for Shipment (Scheduled) with Confidence Interval
6000
4000
2000
0
Frequency
0
1
2
3
4
Days for Shipment (Scheduled)

The bootstrapped distribution of sample means for "Days for Shipment (Scheduled)" based on 1,000 resamples. The distribution is approximately normal, centered around the mean (2.92), with a narrow 95% confidence interval (2.89 to 2.94), indicating high precision. The red dashed line marks the mean of the bootstrapped sample means, reinforcing the stability of the estimate. The narrow confidence interval and the normal shape of the distribution highlight the reliability of bootstrapping as a method for estimating population means and their uncertainty.



Result and Conclusion

In conclusion, we used predictive modeling techniques to estimate the likelihood of delivery failures, enabling proactive measures to minimize operational costs. Among the models tested, Logistic Regression emerged as the better option due to its balance between recall and precision and its stability across datasets of varying sizes. However, it misses some late deliveries that Naïve Bayes can address, highlighting the need to explore more advanced machine learning and statistical models for improved results.

Geospatial data has transformative potential for inventory forecasting and last-mile delivery optimization, as demonstrated by industry leaders like Amazon, Walmart, and DHL. In this project, we incorporated a basic approach to addressing weather-related challenges in the supply chain process, showcasing its practical value.

Refining and expanding this project could yield actionable insights that businesses can adapt for their unique supply chain needs, offering a scalable and impactful solution for enhancing supply chain management.

Concepts covered as part of STT810 in this Project

We have covered 7 concepts from our class, namely -

a. **Correlation** & **Covariance** in the form of Correlation and Covariance Matrix for our EDA.
b. **Logistic Regression** & **Naïve Bayes Classification** in Data Modelling to predict feasibility of
c. **Bootstrapping** & **Confidence Intervals (CI)** were used in this analysis to evaluate the reliability and precision of the average shipment days estimate
d. ***dplyr*** implementation in our R code.

References

- Supply Chain Dataset - Constante, Fabian; Silva, Fernando; Pereira, António (2019), "DataCo SMART SUPPLY CHAIN FOR BIG DATA ANALYSIS", Mendeley Data, V5, doi: 10.17632/8gx2fvg2k6.5
- Weather Dataset - https://www.visualcrossing.com/
- https://en.wikipedia.org/wiki/Chi-squared_test
- https://www.ncei.noaa.gov/news/noaa-data-helps-retail-and-manufacturing-business-minimize-impacts-weather-and-climate
- https://parcelindustry.com/article-6377-The-Impact-of-Weather-on-the-Supply-Chain.html
- https://en.wikipedia.org/wiki/Supply_chain
- Assistance from https://chatgpt.com/ and https://claude.ai/new was taken for comprehension of concepts and theory.