Report: Order Delivery Time Prediction¶

Include your visualisations, analysis, results, insights, and outcomes. Explain your methodology and approach to the tasks. Add your conclusions to the sections.

1. Data Understanding

1.1. Loading the data

The dataset is loaded into a DataFrame containing detailed records of food delivery orders. Each row represents a single order and includes information related to order timing, store and order characteristics, item pricing, and delivery logistics. This data will serve as the foundation for further analysis and modeling tasks, such as predicting delivery times or understanding factors affecting delivery efficiency..

Loading the dataset:

```
# Importing the file porter_data_1.csv
df = pd.read_csv("porter_data_1.csv")
df.head(5)
```

This results in 1.76 lakh (175,777) entries, which is approximately 0.18 million.

2. Data Preprocessing and Feature Engineering

2.1. Fixing the Data Types

2.1.1. Convert date and time fields to appropriate data type

To ensure accurate time-based calculations and enable time series analysis, the **created_at** and **actual_delivery_time** columns are converted from string (object) type to Python datetime objects using pandas.to_datetime(). This conversion allows for efficient operations such as computing time differences, filtering by date ranges, extracting specific time components (e.g., hour, day), and sorting chronologically.

2.1.2. Convert categorical fields to appropriate data type

To optimize memory usage and improve performance during analysis and modeling, categorical fields such as **store_primary_category** and **order_protocol** are explicitly converted to the category data type using pandas.

2.2. Feature Engineering

2.2.1. Calculate the time taken using the features `actual_delivery_time` and `created_at`

To derive the total time taken for each order, the difference between **actual_delivery_time** and **created_at** is computed. The result, originally in timedelta format, is converted into minutes by extracting the total seconds and dividing by 60. This new feature, **time_taken_minutes**, provides a quantitative measure of delivery duration that can be used for exploratory analysis and predictive modeling.

2.2.2. Extract the hour at which the order was placed and which day of the week it was. Drop the unnecessary columns.

2.2.2.1. Extracting Temporal Features

1. The hour of the day when each order was placed is extracted using .dt.hour.
2. The day of the week is derived using .dt.dayofweek where 0 = Monday and 6 = Sunday.
3. A new categorical binary feature, isWeekend, is created to indicate whether the order was placed on a weekend (1) or not (0), using a simple lambda function.

2.2.2.2. Convert categorical fields to appropriate data type

The **day_of_week**, **isWeekend**, and **market_id** fields are explicitly converted to the category data type.

2.2.2.3. Dropping Unnecessary Columns

The **created_at** and **actual_delivery_time** columns are dropped after extracting the necessary temporal features.
The **store_primary_category** column is also dropped due to its high cardinality (73 unique values). Retaining it would lead to:

1. Increased dimensionality from one-hot encoding
2. Greater model complexity
3. Risk of overfitting and multicollinearity

2.2.2.4. Handling Negative Values in Numeric Fields

Negative values in features like **min_item_price**, **total_onshift_dashers**, **total_busy_dashers**, and **total_outstanding_orders** are not logically valid. The percentage of such entries is extremely low (less than **0.03%** for any column), so those rows are removed to ensure data quality.

2.3. Creating training and validation sets

2.3.1. Define target and input features

To prepare the dataset for modeling, the target variable and input features are explicitly separated:

1. **Target (y)**:
   The variable to be predicted is time_taken_minutes, representing the total time taken for an order from placement to actual delivery.
2. **Input Features (X)**:
   All remaining columns are used as predictors. The time_taken_minutes column is dropped from the feature set.

2.3.2. Split the data into training and test sets

To evaluate model performance effectively, the dataset is split into training and testing subsets:

1. **Purpose:** Separating the data ensures that the model is trained on one portion of the data (X_train, y_train) and tested on unseen data (X_test, y_test), simulating real-world scenarios.

2. **Split Strategy:**
   a. 80% of the data is used for training the model.
   b. 20% is reserved for testing.
   c. The split is randomized using random_state=42 to ensure reproducibility.

3. **Validation:** The data has been successfully split into training and testing sets with the following dimensions:

   Training Features (X_train): 140,549 samples and 14 features.
   Testing Features (X_test): 35,138 samples and 14 features.
   Training Target (y_train): 140,549 samples.
   Testing Target (y_test): 35,138 samples.

3. Exploratory Data Analysis on Training Data

3.1. Feature Distributions

The features in the dataset are distributed into numerical and categorical columns as follows:

Numerical Features:
These features represent measurable quantities and continuous values related to the order and delivery process:

1. total_items: Total number of items in the order
2. subtotal: Final order price (before taxes/tip)
3. num_distinct_items: Number of unique items in the order
4. min_item_price: Price of the least expensive item
5. max_item_price: Price of the most expensive item
6. total_onshift_dashers: Number of delivery agents available at the time
7. total_busy_dashers: Number of agents currently occupied
8. total_outstanding_orders: Number of orders pending at that time
9. distance: Distance from the restaurant to the customer

Categorical Features:
These are non-numeric, discrete variables that describe categories or classifications:

1. market_id: Identifier for the market/region
2. order_protocol: Method through which the order was placed
3. day_of_week: Day the order was placed
4. isWeekend: Whether the order was placed on a weekend (Yes/No)
5. hour: Hour of the day the order was placed (0 to 23)

3.1.1. Plot distributions for numerical columns in the training set to understand their spread and any skewness

To understand the spread, scale, and potential skewness of the numerical features, we plotted histograms for each column in the training dataset. This helps in identifying:

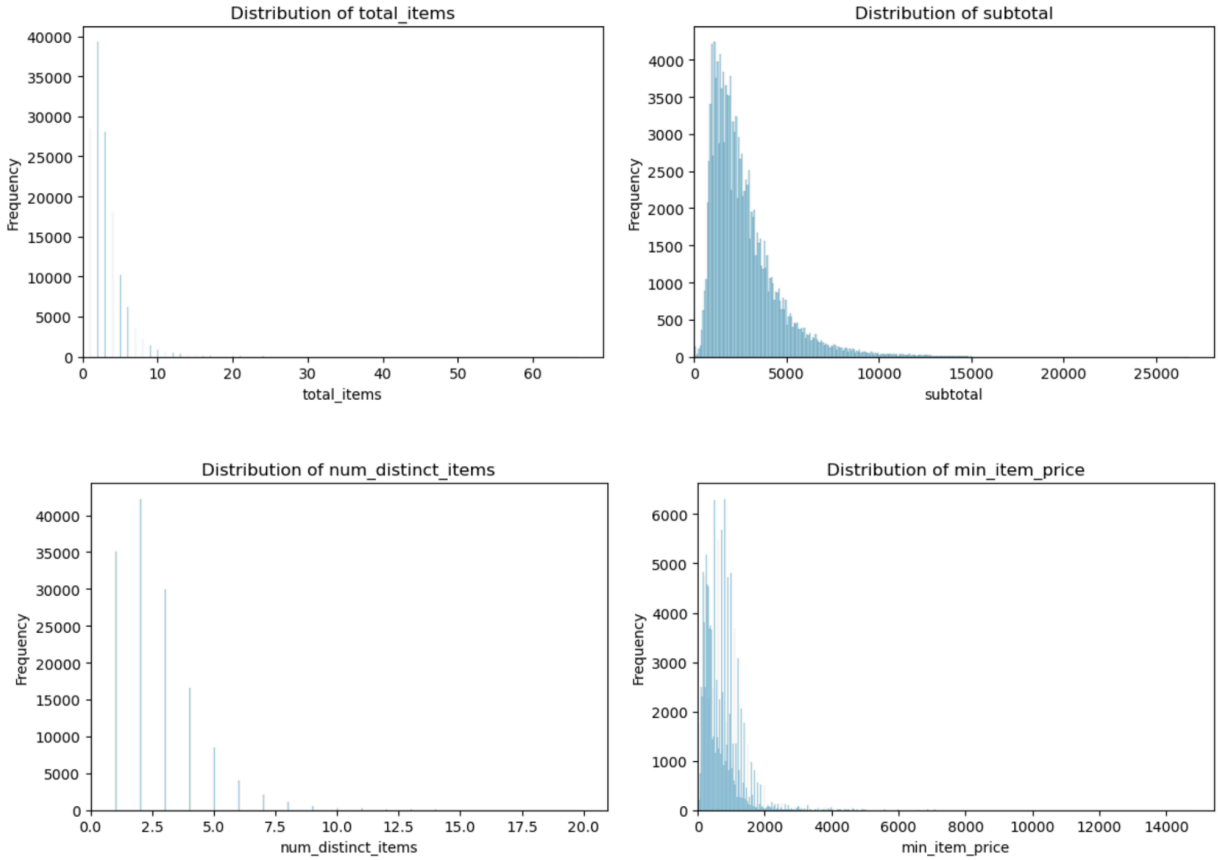
1. Outliers or extreme values
2. Right or left skewness
3. Feature scaling needs

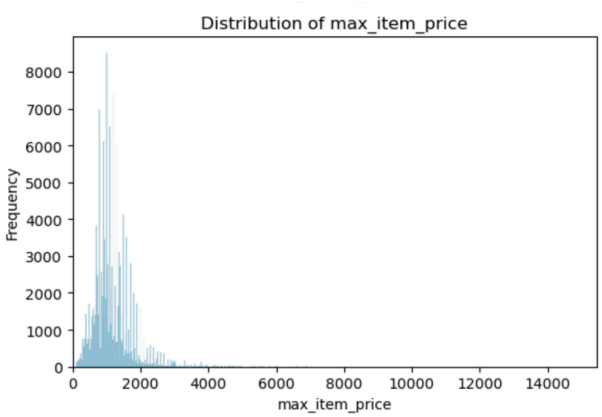
Each subplot displays the frequency distribution of a numerical variable with the x-axis limited to non-negative values (as negative values were already removed earlier). This visualization aids

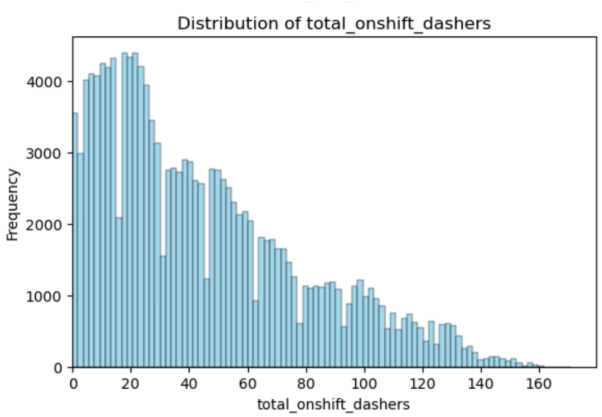
in deciding if transformations (e.g., log, normalization) are needed for specific features during preprocessing.

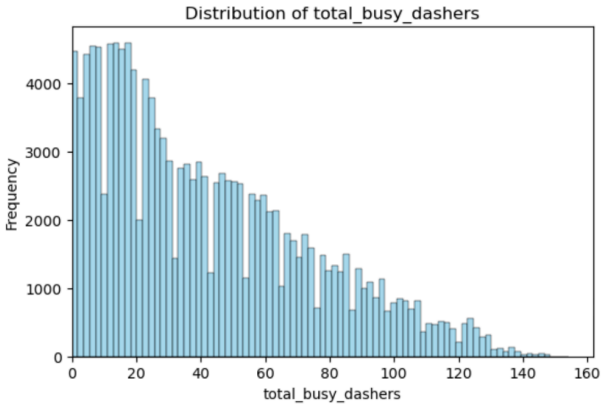
To understand the spread, skewness, and frequency distribution of key numerical variables, the following features were plotted using subplots:

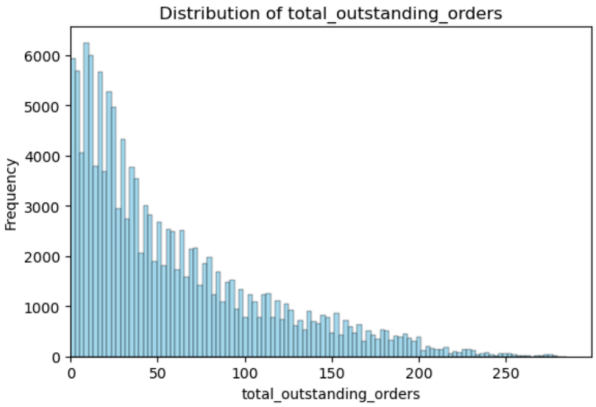
Feature Name	**Description**
total_items	Total number of items in the order.
subtotal	Final price of the order (in cents).
num_distinct_items	Number of distinct items in the order.
min_item_price	Price of the cheapest item in the order (in cents).
max_item_price	Price of the most expensive item in the order (in cents).
total_onshift_dashers	Number of delivery partners on duty when the order was placed.
total_busy_dashers	Number of delivery partners already occupied with other orders.
total_outstanding_orders	Number of orders pending fulfillment at the time of the order.
distance	Total distance from the restaurant to the customer (in miles).

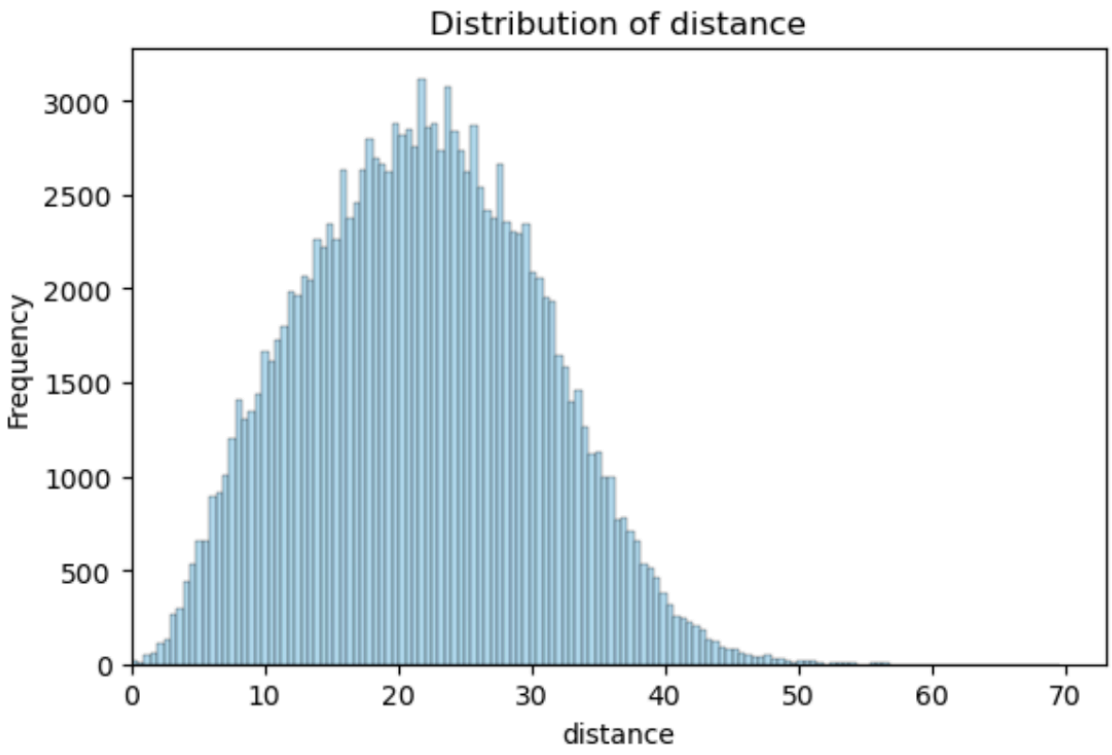



Distribution of max_item_price
Frequency
max_item_price
0
1000
2000
3000
4000
5000
6000
7000
8000
0
2000
4000
6000
8000
10000
12000
14000


Distribution of total_onshift_dashers
Frequency
total_onshift_dashers
0
1000
2000
3000
4000
0
20
40
60
80
100
120
140
160


Distribution of total_busy_dashers
Frequency
total_busy_dashers
0
1000
2000
3000
4000
0
20
40
60
80
100
120
140
160


Distribution of total_outstanding_orders
Frequency
total_outstanding_orders
0
1000
2000
3000
4000
5000
6000
0
50
100
150
200
250

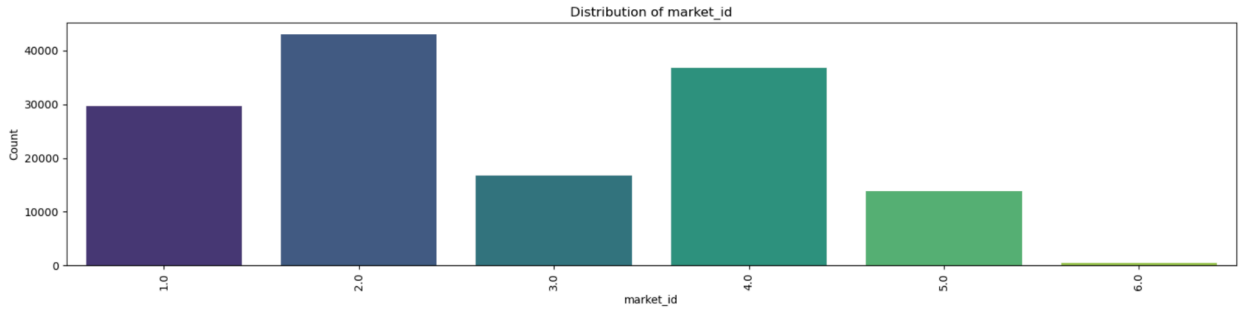


3.1.2. Check the distribution of categorical features

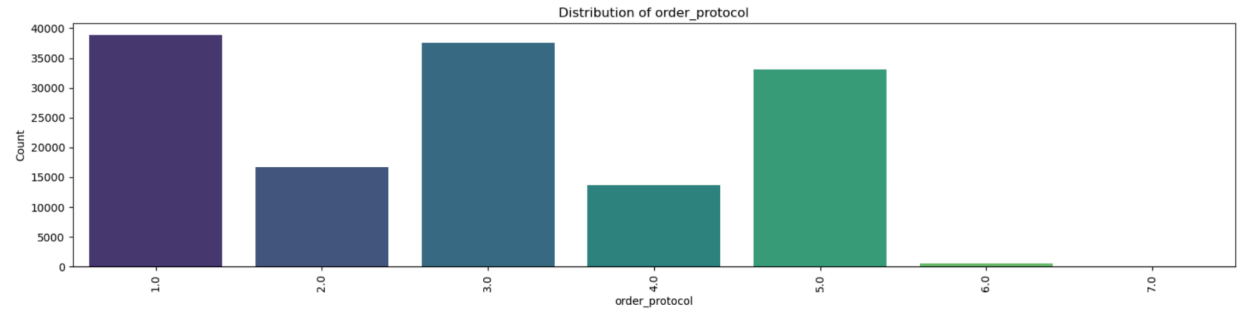
To understand the frequency and spread of different categorical variables, we plotted count plots for the following features:

Feature Name	Description
market_id	Encodes the geographical market/region where the order originated.
order_protocol	Denotes how the order was placed (e.g., app, phone call, etc.).
day_of_week	Day the order was placed (0 = Monday, ..., 6 = Sunday).
isWeekend	Indicates whether the order was placed on a weekend (1) or not (0).
hour	Hour of the day the order was placed (0 to 23).

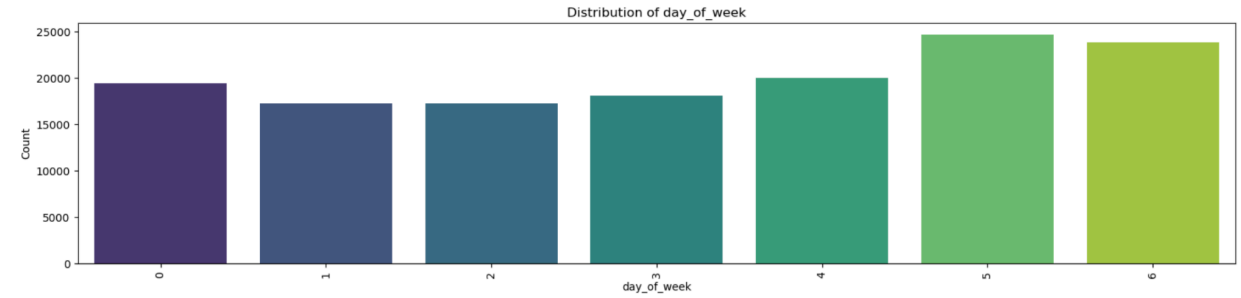
Distribution of market_id:



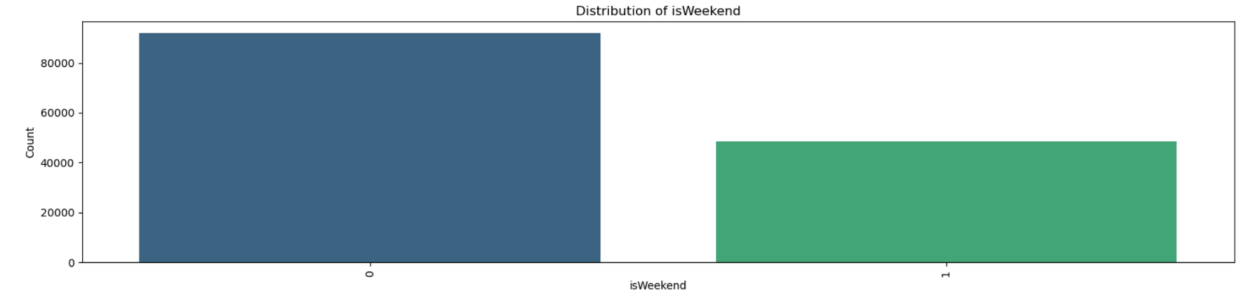
Distribution of order_protocol:



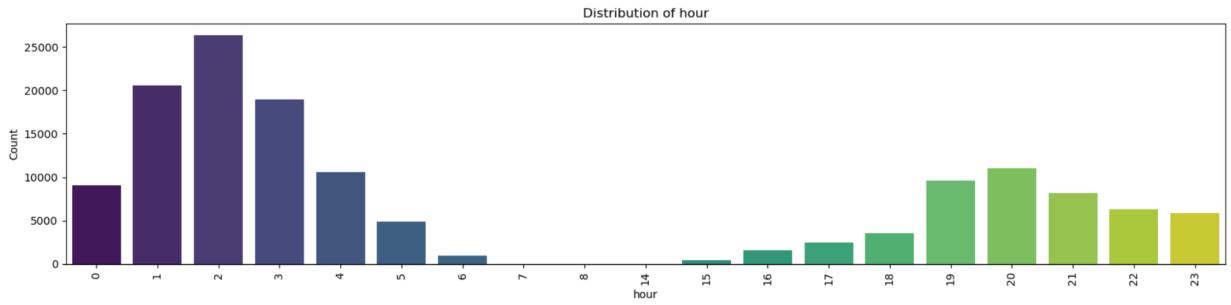
Distribution of day_of_week:



Distribution of isWeekend:



Distribution of hour:



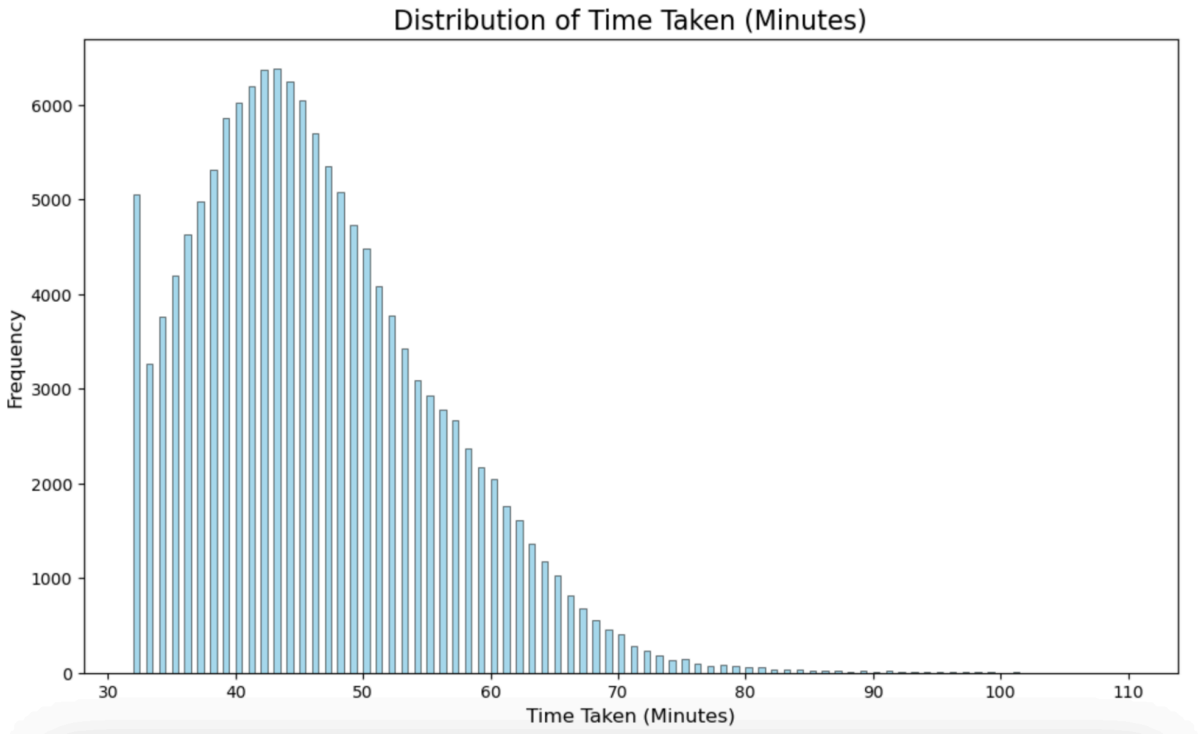
Observations:

1. **market_id:** Imbalanced distribution; IDs 2.0 and 4.0 are most common, while 6.0 is rare.
2. **order_protocol:** Protocols 1.0 and 3.0 dominate; 6.0 and 7.0 are least used.
3. **isWeekend:** More activity on weekdays than weekends.
4. **hour:** Activity peaks during late night (0–4 AM) and evening (7–10 PM); low during daytime (6 AM–2 PM).

After a detailed analysis, it was observed that the **day_of_week** feature had minimal impact on delivery time variation. Since the **isWeekend** feature already captures the key difference between weekdays and weekends, retaining **day_of_week** would unnecessarily introduce six additional dummy variables (for the seven days of the week), thereby increasing model complexity without contributing significant predictive value. Hence, it was removed from the model.

3.1.3. Visualise the distribution of the target variable to understand its spread and any skewness

The histogram below visualizes the distribution of the target variable **time_taken_minutes**, which represents the total time taken for an order from placement to delivery.



Observations:

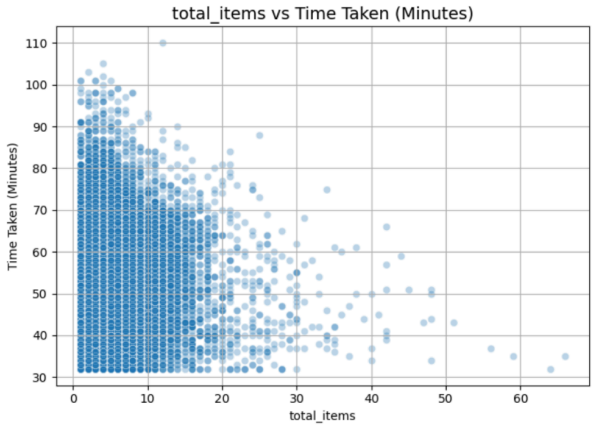
1. The distribution is right-skewed, meaning most orders are completed in a shorter time, while fewer take longer.
2. The majority of orders are delivered within 35 to 55 minutes, with a peak around 45 minutes.
3. A long tail is observed toward the right, indicating the presence of outliers or extreme cases where delivery takes significantly longer (up to ~100 minutes).

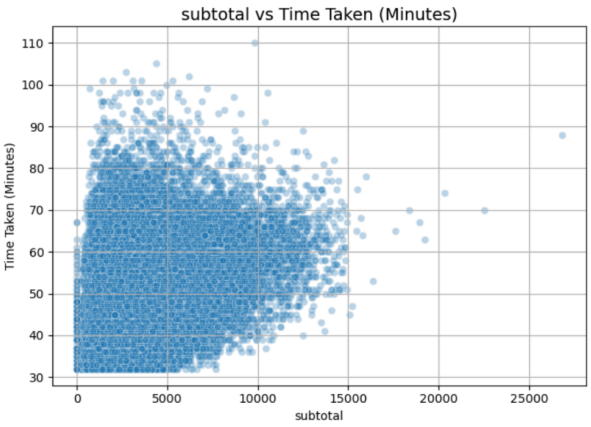
3.2. Relationships Between Features

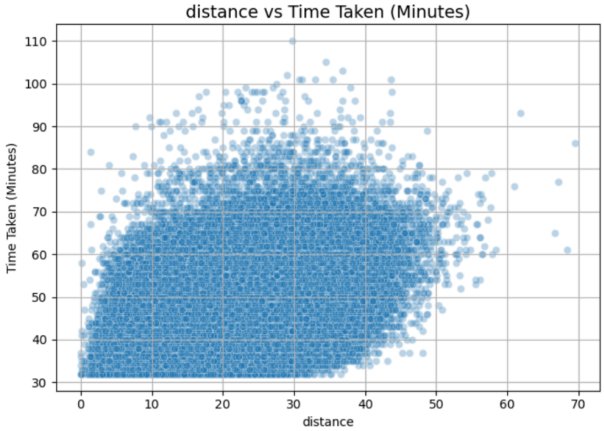
3.2.1. Scatter plots for important numerical and categorical features to observe how they relate to `time_taken`

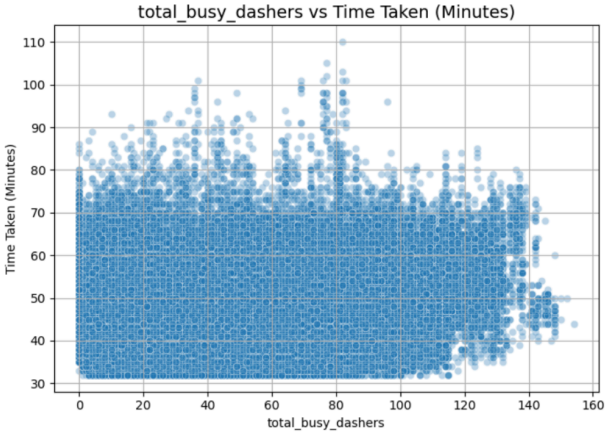
Numerical Features:

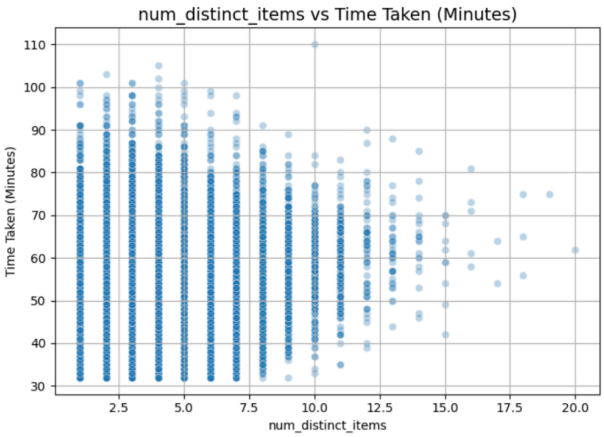
To analyze the relationship between key numerical predictors and the target variable (Time Taken (Minutes)), we created scatter plots for the following important numerical features: **total_items, subtotal, distance, total_busy_dashers, num_distinct_items, min_item_price, max_item_price, total_onshift_dashers, and total_outstanding_orders**. These plots help us visually assess the trends, variance, and potential correlations between each feature and the delivery time. Such insights are valuable in identifying features that may significantly influence the model's performance.

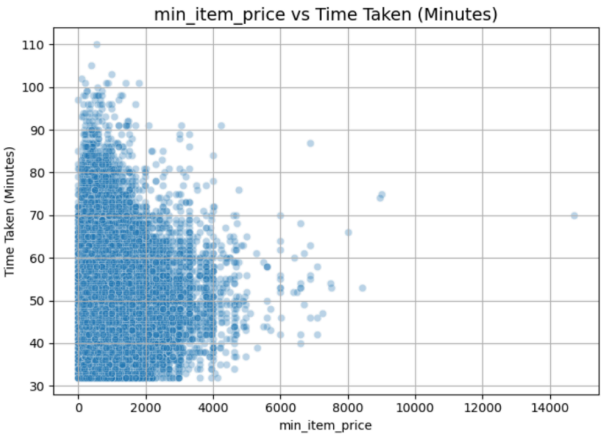

total_items vs Time Taken (Minutes)
Time Taken (Minutes)
total_items

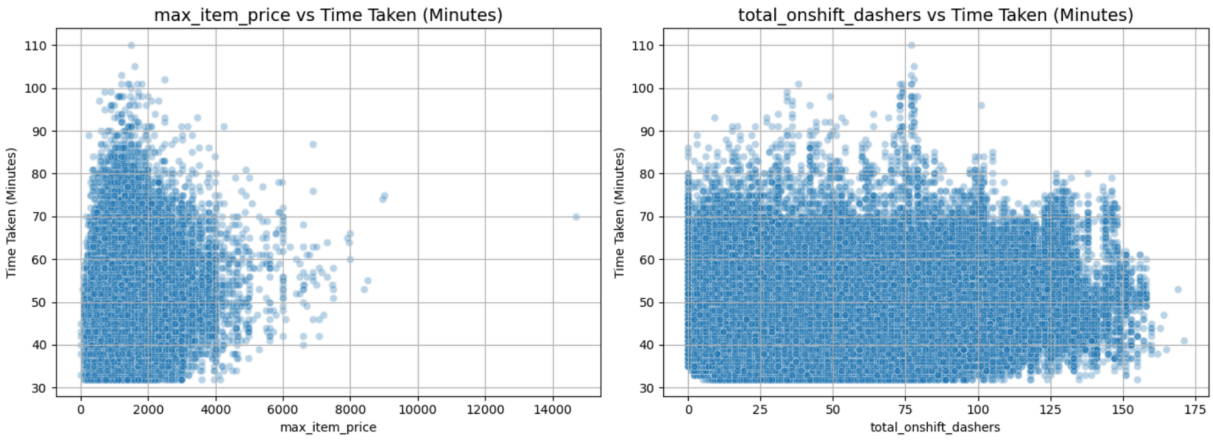

subtotal vs Time Taken (Minutes)
Time Taken (Minutes)
subtotal

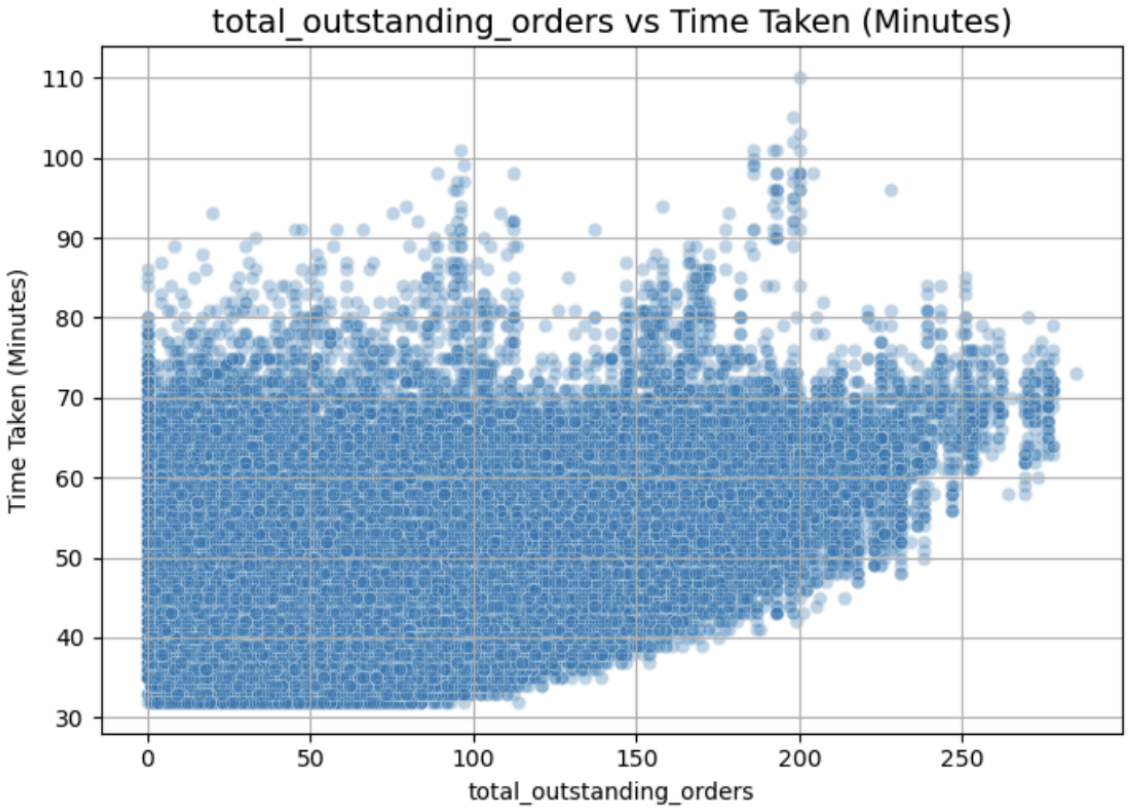

distance vs Time Taken (Minutes)
Time Taken (Minutes)
distance


total_busy_dashers vs Time Taken (Minutes)
Time Taken (Minutes)
total_busy_dashers


num_distinct_items vs Time Taken (Minutes)
Time Taken (Minutes)
num_distinct_items


min_item_price vs Time Taken (Minutes)
Time Taken (Minutes)
min_item_price





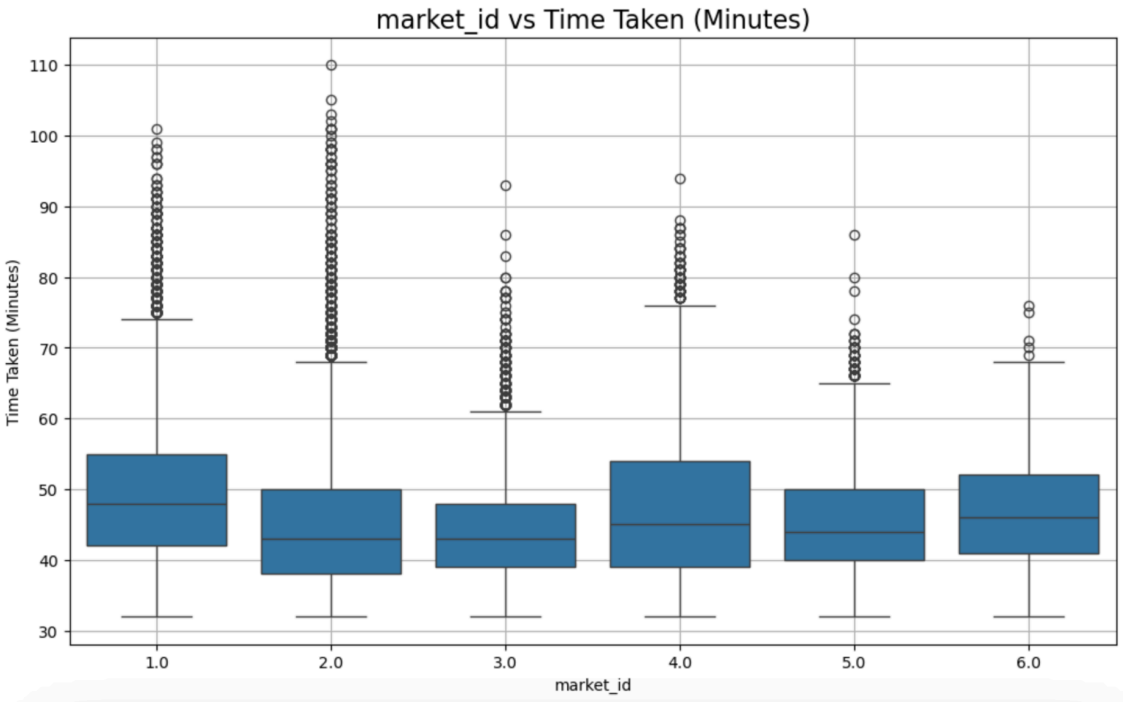
Observations:

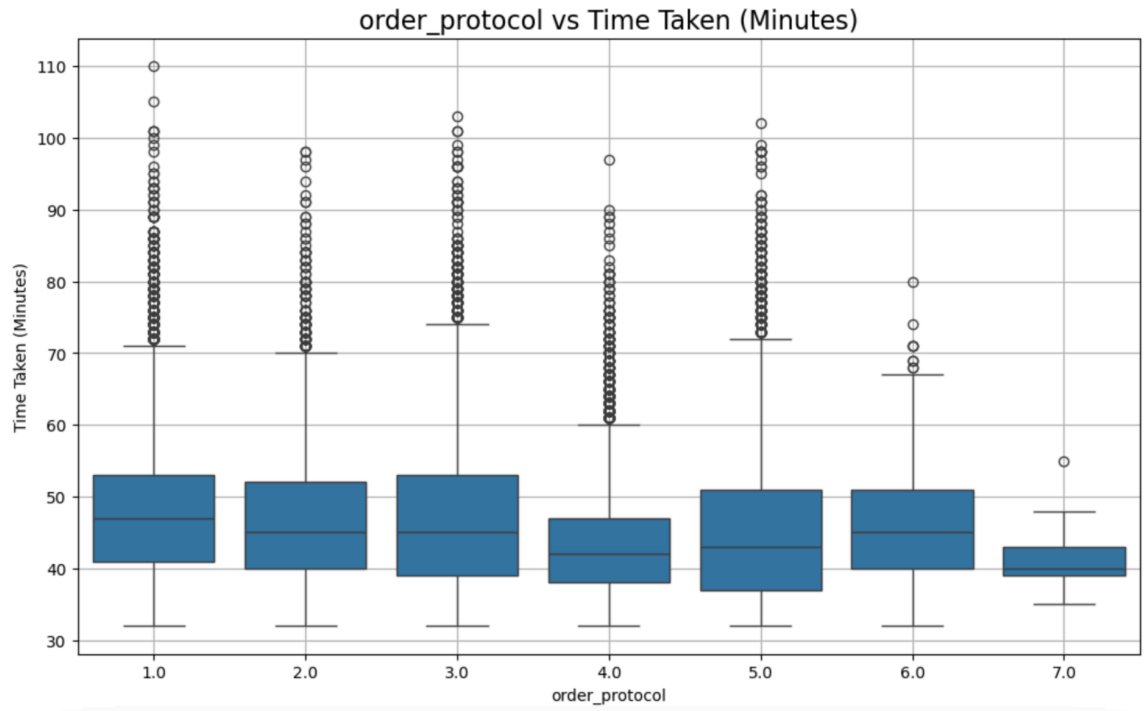
1. The scatter plot for **total_items** vs **Time Taken (Minutes)** shows a slight downward trend, suggesting that as the number of items increases, the delivery time tends to slightly decrease or remain stable. Most orders contain fewer than 20 items, with delivery times typically ranging between 35 to 90 minutes. A few outliers are observed for orders with a high item count.

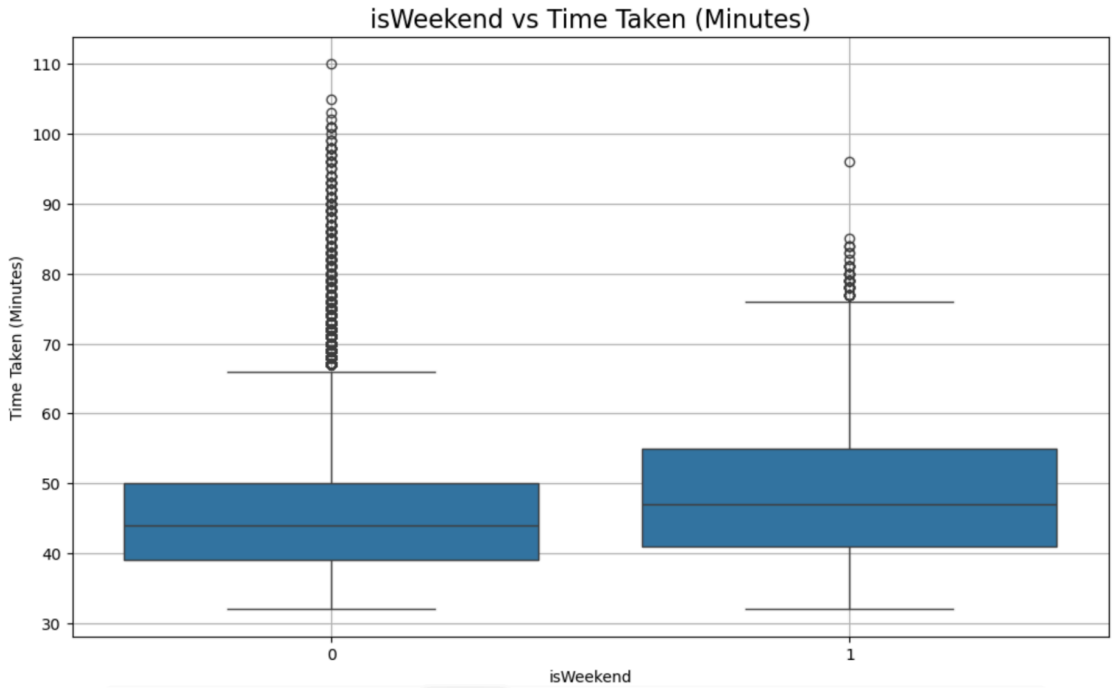
2. The scatter plot of **subtotal** vs **Time Taken (Minutes)** shows that most orders with subtotals below 10,000 are delivered in 40–80 minutes. Time taken remains fairly consistent as subtotal increases, with a few high-value outliers showing more variation. No strong linear relationship is observed.
3. The scatter plot of **distance** vs **Time Taken (Minutes)** shows a moderate positive trend: as distance increases, the time taken also generally increases. Most deliveries are within 10 to 40 km and take between 40 to 80 minutes. A few outliers exist beyond 50 km with higher delivery times.
4. The scatter plot of **total_busy_dashers** vs **Time Taken (Minutes)** shows no clear linear relationship. Most deliveries occur when there are between 20 and 120 busy dashers, with delivery times commonly ranging from 40 to 80 minutes. Outliers exist across the range, indicating some variation in delivery time regardless of dasher availability.
5. The scatter plot of **num_distinct_items** vs **Time Taken (Minutes)** shows that delivery time stays relatively consistent regardless of the number of distinct items, mostly ranging between 40 to 80 minutes, indicating no strong correlation.
6. The scatter plot of **min_item_price** vs **Time Taken (Minutes)** reveals that while higher minimum item prices tend to have slightly faster deliveries, the overall relationship is weak, with most deliveries clustered below a price of 2000 and within the 40 to 80-minute range.
7. The scatter plot of **max_item_price** vs **Time Taken (Minutes)** shows that higher maximum item prices slightly correlate with faster delivery times, but the overall relationship remains weak, with most deliveries concentrated below 4000 in price.
8. The scatter plot of **total_onshift_dashers** vs **Time Taken (Minutes)** indicates a mild inverse relationship—delivery times tend to decrease as the number of on-shift dashers increases, suggesting that greater dasher availability may reduce wait times.
9. The scatter plot of **total_outstanding_orders** vs **Time Taken (Minutes)** shows a positive correlation—delivery times tend to increase as the number of outstanding orders rises, indicating that higher demand may lead to longer wait times.

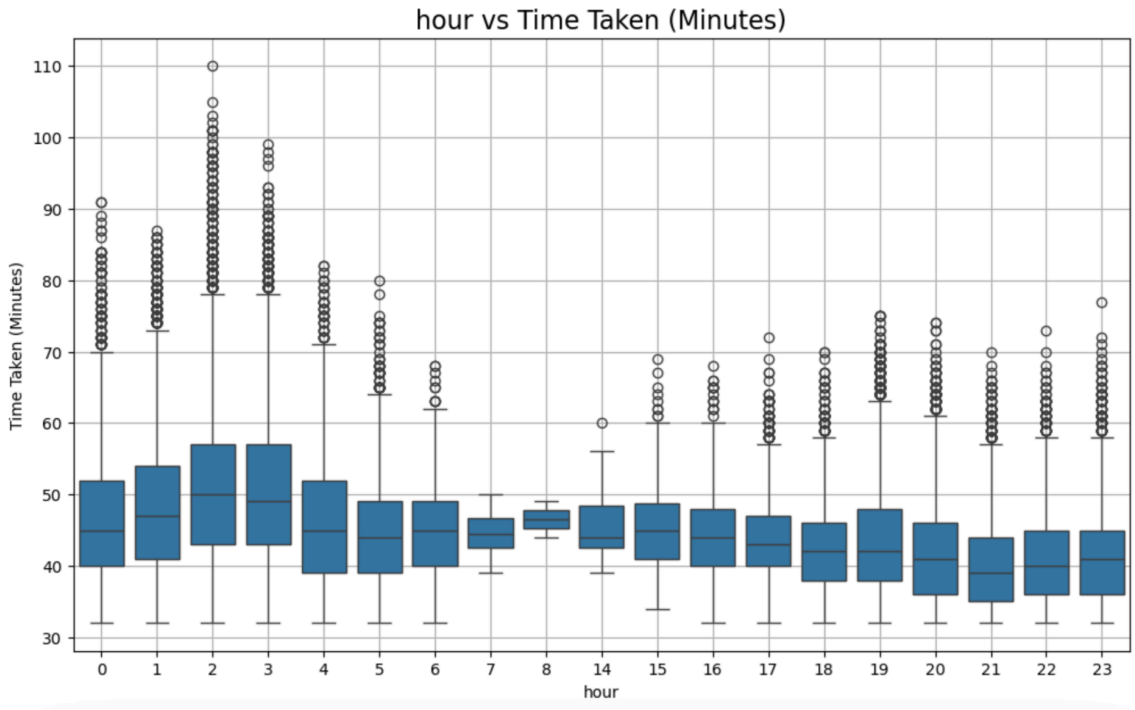
Categorical Features:

To understand the distribution and potential impact of categorical variables on the target variable (Time Taken (Minutes)), we plotted count plots for the following features: **market_id**, **order_protocol**, **day_of_week**, **isWeekend**, and **hour**. These visualizations help us explore the frequency of each category, detect any imbalance or dominant classes, and identify patterns that might affect delivery time. Such exploratory analysis is crucial for informed feature encoding and model development.


market_id vs Time Taken (Minutes)
Time Taken (Minutes)
110
100
90
80
70
60
50
40
30
1.0
2.0
3.0
4.0
5.0
6.0
market_id


order_protocol vs Time Taken (Minutes)
Time Taken (Minutes)
110
100
90
80
70
60
50
40
30
1.0
2.0
3.0
4.0
5.0
6.0
7.0
order_protocol





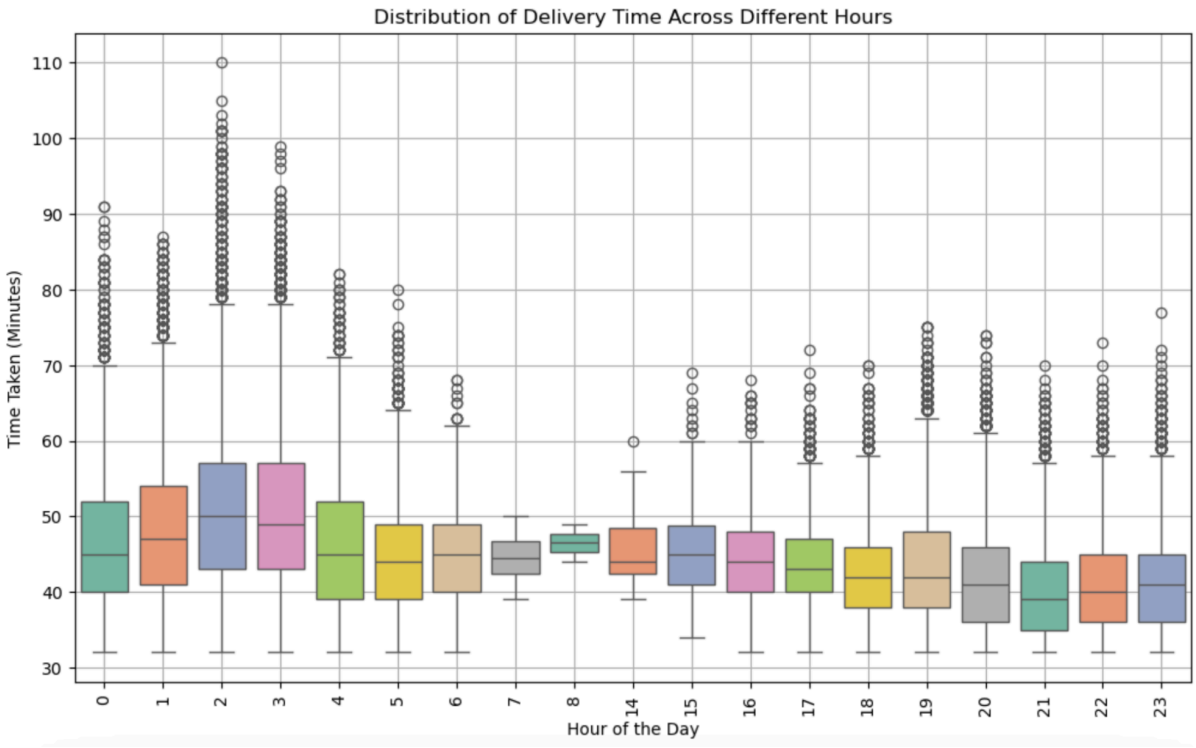
Observations:

1. market_id: The box plot shows that market_id has the shortest and most consistent delivery times, while Markets 1 and 4 have longer and more variable times. All markets have some outliers with high delivery times.

2. order_protocol: The box plot shows that Order Protocol 7 has the lowest and most consistent delivery time, while Protocols 1, 2, and 3 have higher medians and more outliers, indicating longer and more variable delivery durations.
3. isWeekend: The box plot shows that Time Taken is slightly higher on weekends (isWeekend = 1) compared to weekdays, with a higher median and more spread, indicating longer and more variable delivery times during weekends.
4. hour: The box plot shows that Time Taken is higher during early morning hours (0–5 AM), with more variability and outliers, while delivery time is shorter and more consistent during daytime hours (8 AM–8 PM).

Show the distribution of time_taken for different hours

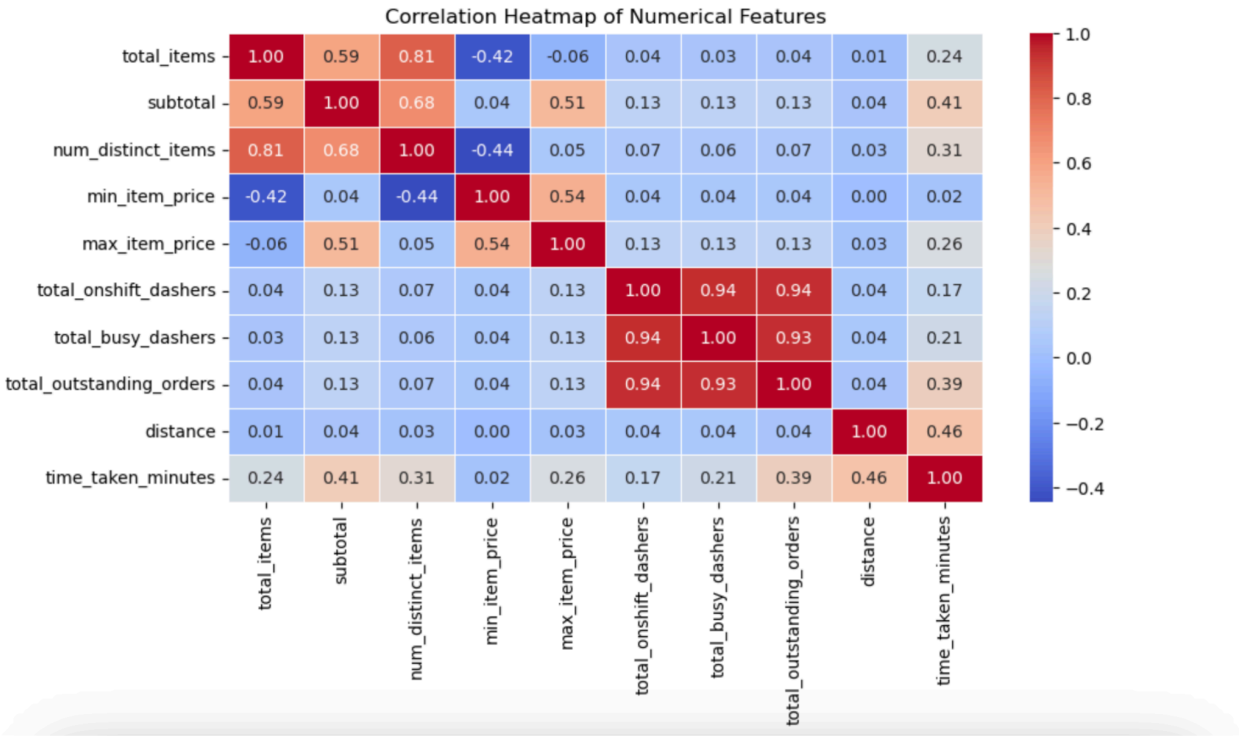
To explore how delivery time varies throughout the day, we plotted a box plot showing the distribution of Time Taken (Minutes) across different hour values (0–23). The plot reveals that delivery times tend to be higher and more variable during late night and early morning hours (0–3 AM), likely due to limited delivery personnel or low order volumes. As the day progresses, the delivery time generally stabilizes, with relatively lower median times observed between 8 AM and 6 PM. This visualization helps in identifying peak and off-peak delivery hours, which could be important for demand forecasting and resource allocation.



3.3. Correlation Analysis

3.3.1. Check correlations between numerical features to identify which variables are strongly related to `time_taken`. Plot a heatmap to display correlations

The correlation heatmap visualizes the linear relationships between numerical features in the dataset, including the target variable time_taken_minutes. The color intensity represents the strength and direction of correlation: red for positive, blue for negative, and white for near-zero correlation.



Observations

1. **Strongest correlations with time_taken_minutes:**
    a. Distance (0.46): Most positively correlated feature — longer distances generally result in higher delivery times.
    b. Subtotal (0.41) and total_outstanding_orders (0.39) also show moderate positive correlations, indicating larger orders or higher order load may increase delivery time.
    c. num_distinct_items (0.31) and max_item_price (0.26) show weak to moderate positive correlation with delivery time.
2. **Weak or negligible correlation:**

    a. Features like min_item_price, total_onshift_dashers, and total_busy_dashers have very low correlations (close to 0), suggesting little direct impact on delivery time.
3. **High inter-feature correlation (non-target):**
    a. total_onshift_dashers, total_busy_dashers, and total_outstanding_orders are strongly correlated with each other (~0.93–0.94), possibly due to operational dependencies.

3.3.2. Drop the columns with weak correlations with the target variable

To improve the quality of the model, numerical features with weak correlations to the target variable **time_taken_minutes** were identified and removed. Correlation coefficients were calculated to measure the linear relationship between each feature and the target. The five features with the lowest absolute correlation values were considered uninformative for predicting delivery time and were subsequently dropped from the training dataset. This step helps streamline the dataset, reduce noise, and focus the model on the most relevant inputs.

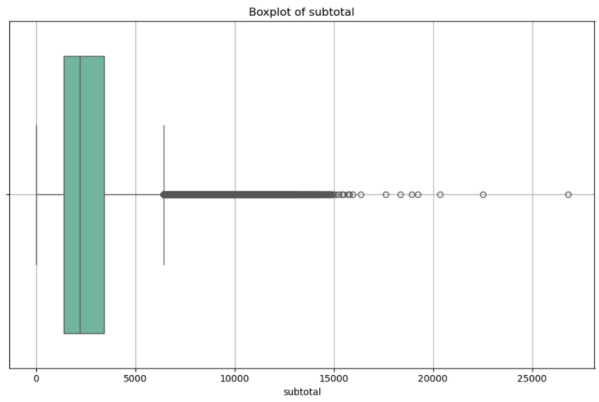
Features Removed

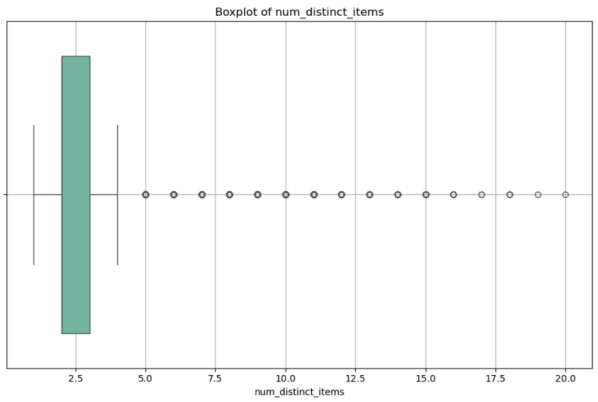
Feature	Correlation with time_taken_minutes
min_item_price	0.02
total_onshift_dashers	0.17
total_busy_dashers	0.21
total_items	0.24
max_item_price	0.26

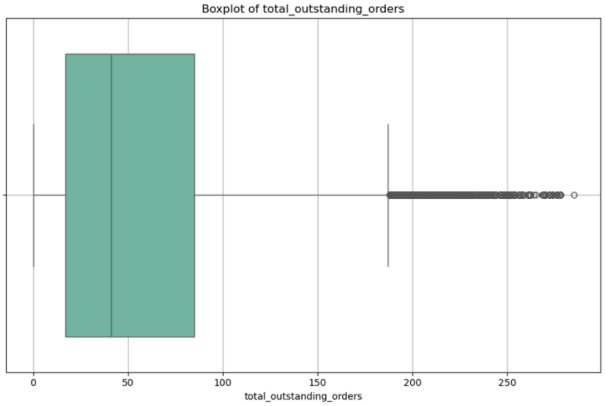
3.4. Handling the Outliers

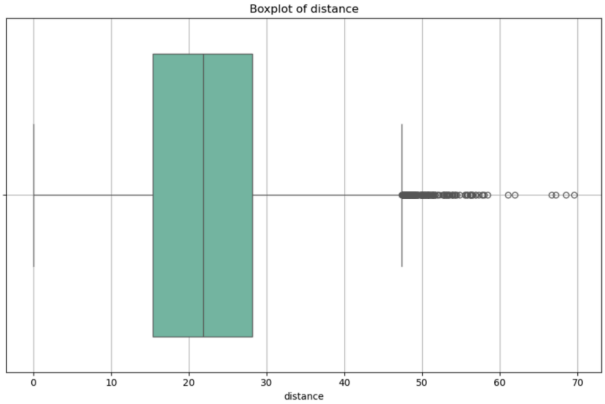
3.4.1. Visualise potential outliers for the target variable and other numerical features using boxplots

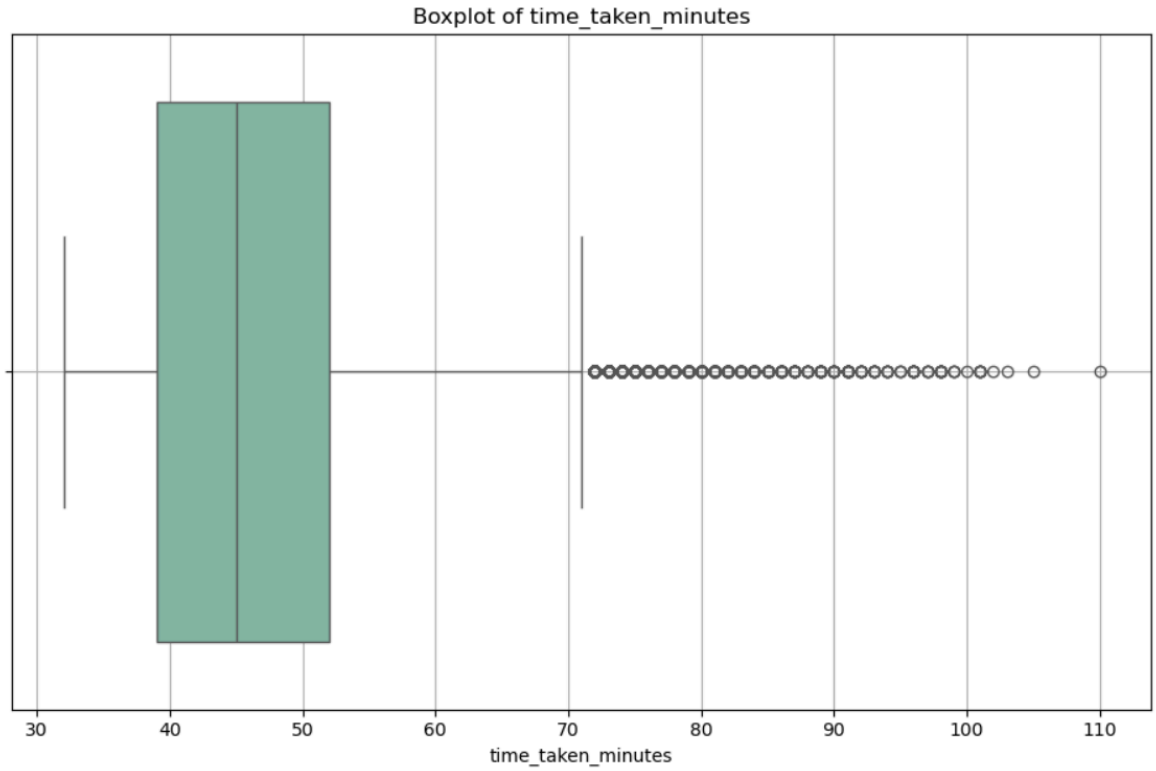
Boxplots were generated for the target variable **time_taken_minutes** and all remaining numerical features to visually assess the presence of outliers. These plots provide a summary of the data distribution, highlighting the median, interquartile range, and any extreme values (potential outliers) beyond the whiskers.
From the visualizations, it is evident that several features, including time_taken_minutes, exhibit outliers—values that fall significantly outside the normal range. Identifying these points is crucial for downstream preprocessing, as they may impact model performance if left unaddressed.


Boxplot of subtotal
0
5000
10000
15000
20000
25000
subtotal


Boxplot of num_distinct_items
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
num_distinct_items


Boxplot of total_outstanding_orders
0
50
100
150
200
250
total_outstanding_orders


Boxplot of distance
0
10
20
30
40
50
60
70
distance



Observations:

1. **time_taken_minutes**: Slight right skew with outliers above ~71 minutes; most values between 40–70.
2. **subtotal**: Highly right-skewed with many large outliers; most data clustered at lower values.
3. **num_distinct_items**: Low variability; outliers above ~5 items indicate rare, complex orders.
4. **total_outstanding_orders**: Strong positive skew; outliers above ~175, most values under 100.
5. **distance**: Right-skewed with outliers beyond ~48 units; most deliveries within 10–30 units.

3.4.2. Handle outliers present in all columns

Outliers were detected in five key numerical features using the Interquartile Range (IQR) method. The percentages of outliers in each column are:

1. subtotal: 4.56% outliers
2. num_distinct_items: 11.92% outliers
3. total_outstanding_orders: 2.96% outliers

4. distance: 0.18% outliers
5. time_taken_minutes: 1.02% outliers

While most columns have less than 5% outliers, num_distinct_items has a slightly higher rate at nearly 12%.

The following methods were applied to handle outliers in the columns mentioned above:

1. **distance**: Removed outliers using IQR (few outliers, safe to drop).
2. **total_outstanding_orders**: Capped at 95th percentile to reduce high-end impact.
3. **subtotal**: Capped at 95th percentile to control extreme values.
4. **num_distinct_items**: Capped using IQR to handle discrete outliers.

Although **time_taken_minutes** contains a small proportion of outliers (**1.02%**), these were **not capped or removed**. In regression tasks, especially with real-world delivery data, such outliers often represent genuine long delivery times rather than data errors.
Modifying the target variable can:

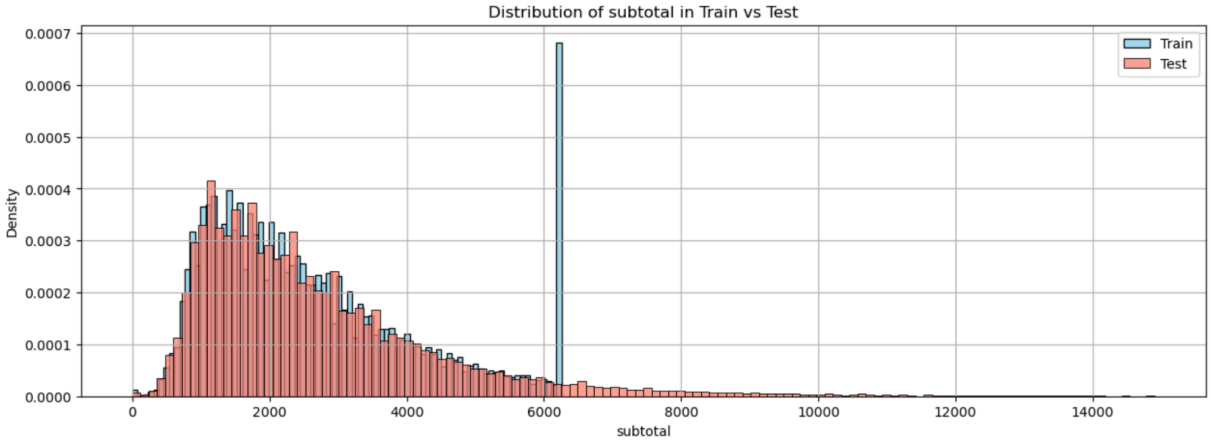
1. Distort meaningful variation.
2. Reduce the model's ability to predict rare but important cases.

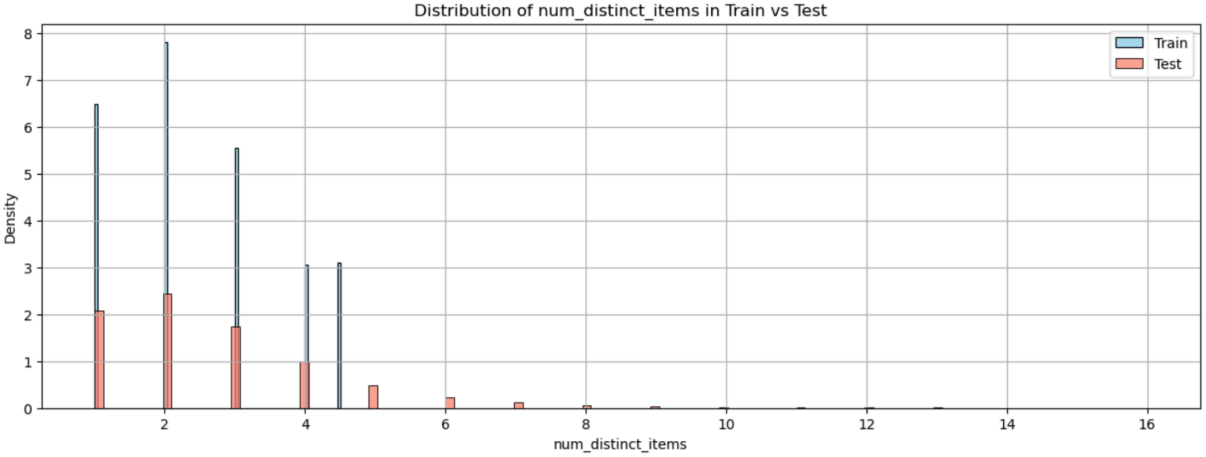
Hence, the target variable was **left unchanged**, unless further analysis reveals these outliers are due to anomalies or data quality issues.

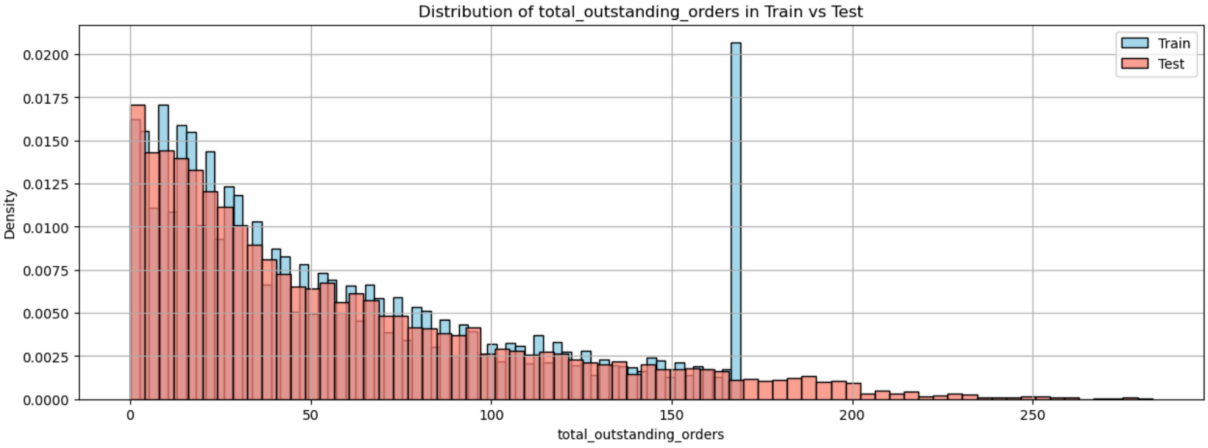
4. Exploratory Data Analysis on Validation Data

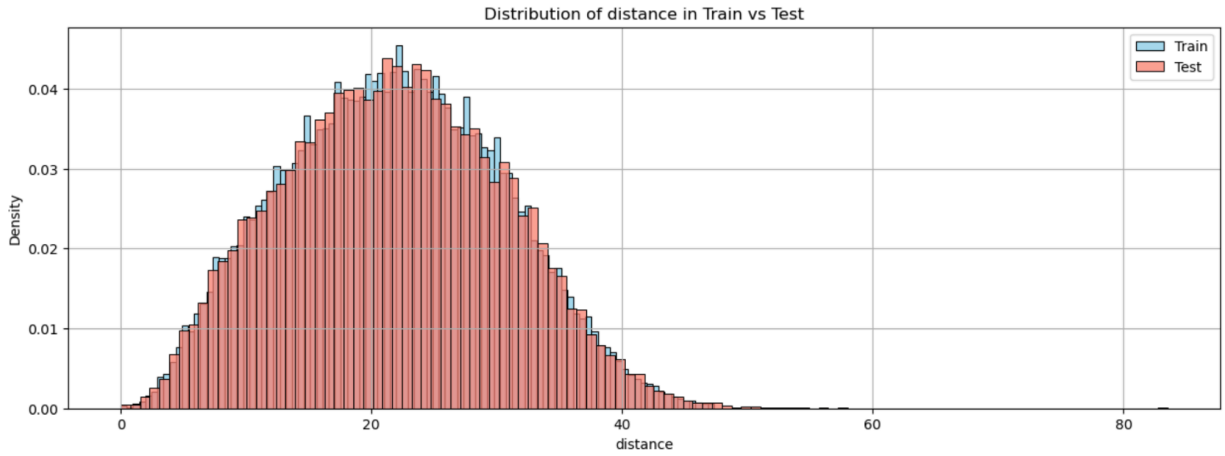
Optionally, perform EDA on test data to see if the distribution match with the training data

To ensure model generalization and avoid training–testing mismatch, exploratory data analysis (EDA) was performed to compare the distributions of numerical features between the training and test datasets.

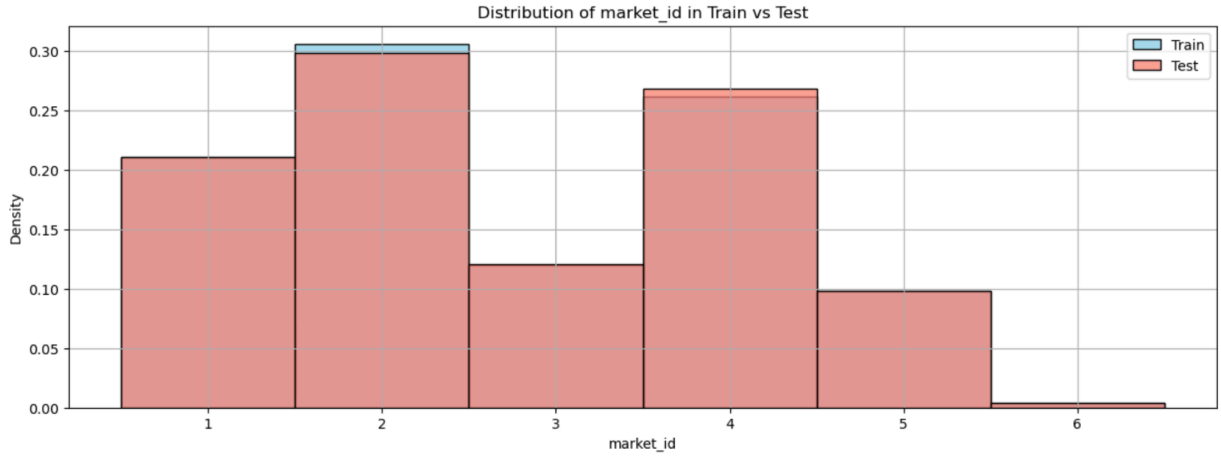

Distribution of subtotal in Train vs Test
Train
Test
Density
0.0007
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
2000
4000
6000
8000
10000
12000
14000
subtotal

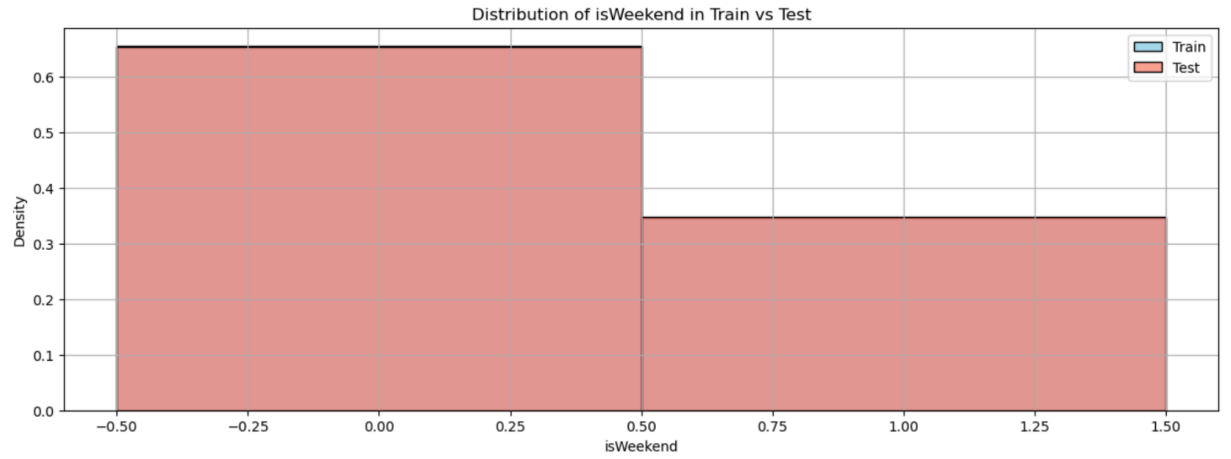

Distribution of num_distinct_items in Train vs Test
Train
Test
Density
8
7
6
5
4
3
2
1
0
2
4
6
8
10
12
14
16
num_distinct_items

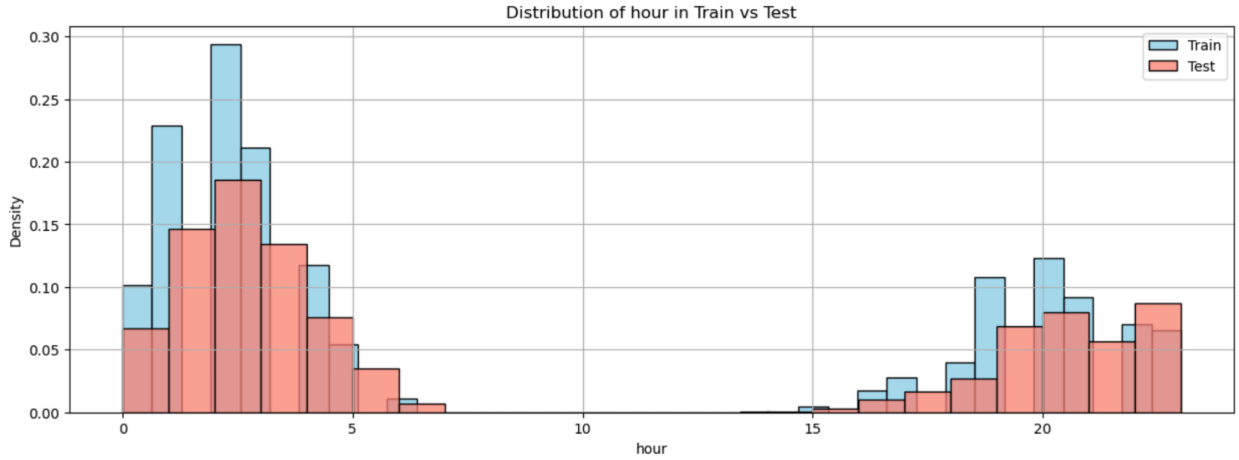

Distribution of total_outstanding_orders in Train vs Test
Train
Test
Density
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
50
100
150
200
250
total_outstanding_orders



To verify consistency between training and test data, categorical feature distributions were visually compared using count plots.





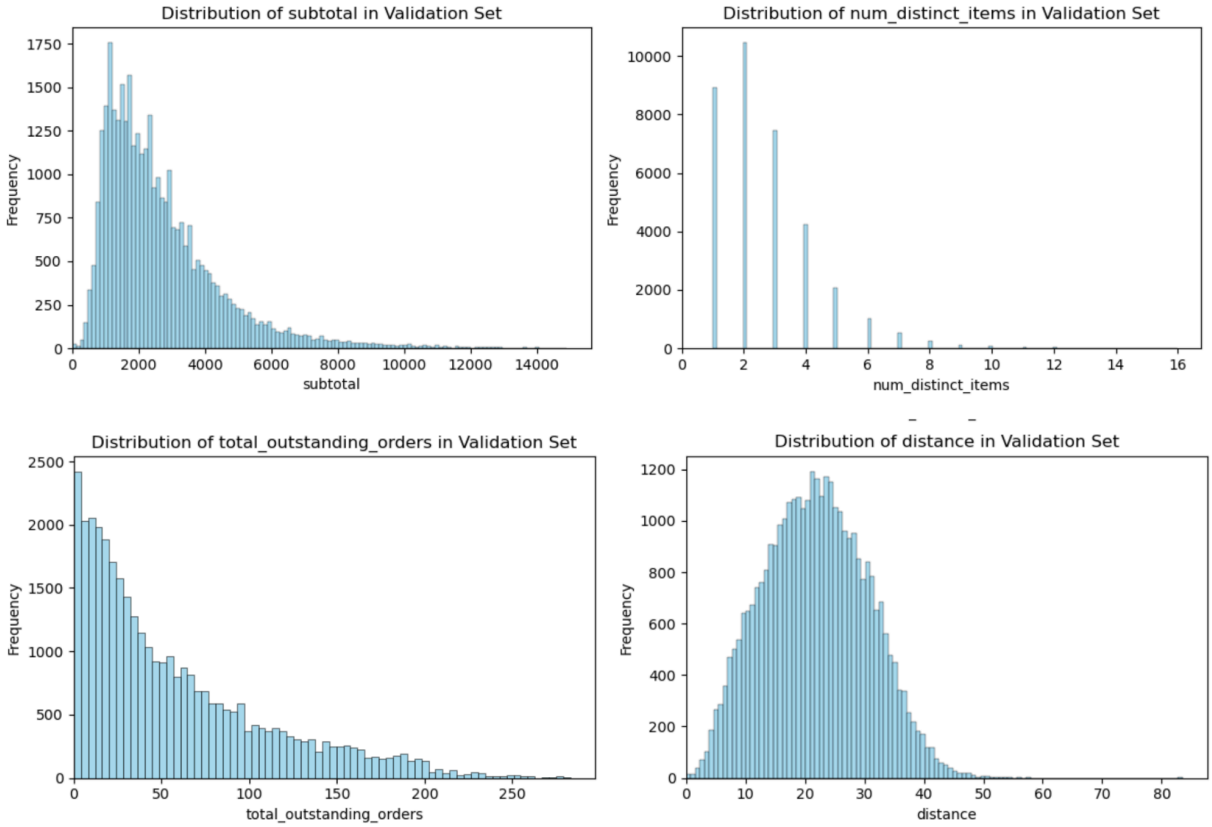


Observations:

1. Train and Test subtotal distributions are right-skewed. Train has a spike around 6300, likely a fixed value. Outliers are capped at the 95th percentile to reduce skew and improve model performance.
2. The num_distinct_items distribution is right-skewed with most values concentrated between 1 and 5. Extreme values beyond this range were capped using IQR-based methods to reduce their influence.
3. The total_outstanding_orders variable is right-skewed with a noticeable spike around 170 in the training data, indicating a potential outlier cluster. Outliers were capped using IQR-based thresholds to align the distribution more closely with the test set.
4. The distance feature follows a near-normal distribution and shows excellent alignment between training and test sets, indicating good data consistency without outliers or shifts.

4.1 Feature Distributions

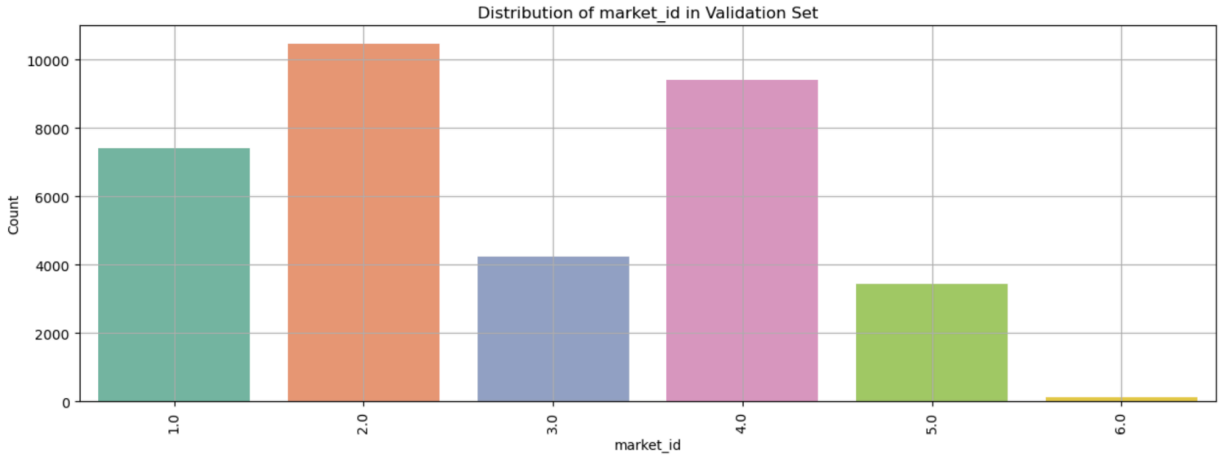
4.1.1. Plot distributions for numerical columns in the validation set to understand their spread and any skewness

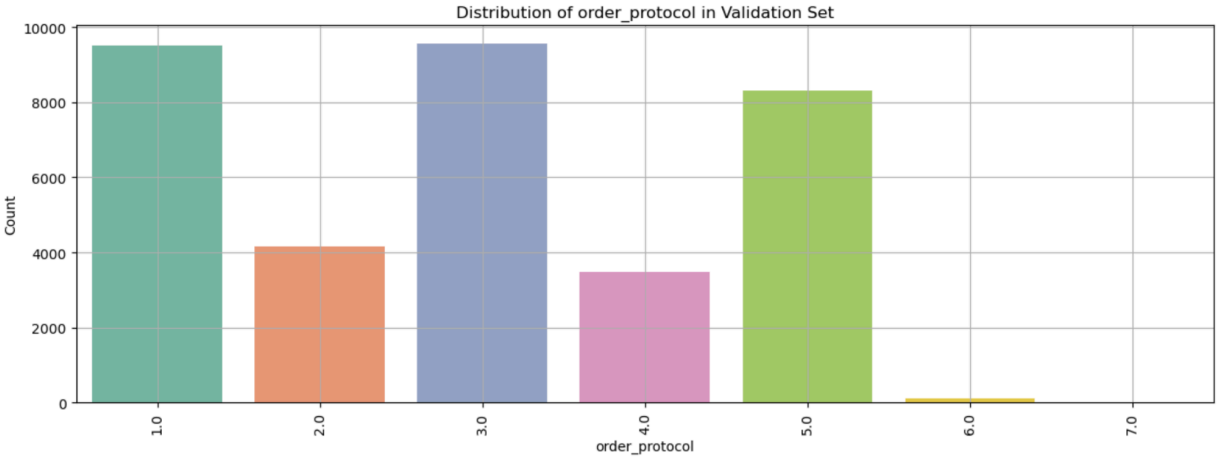


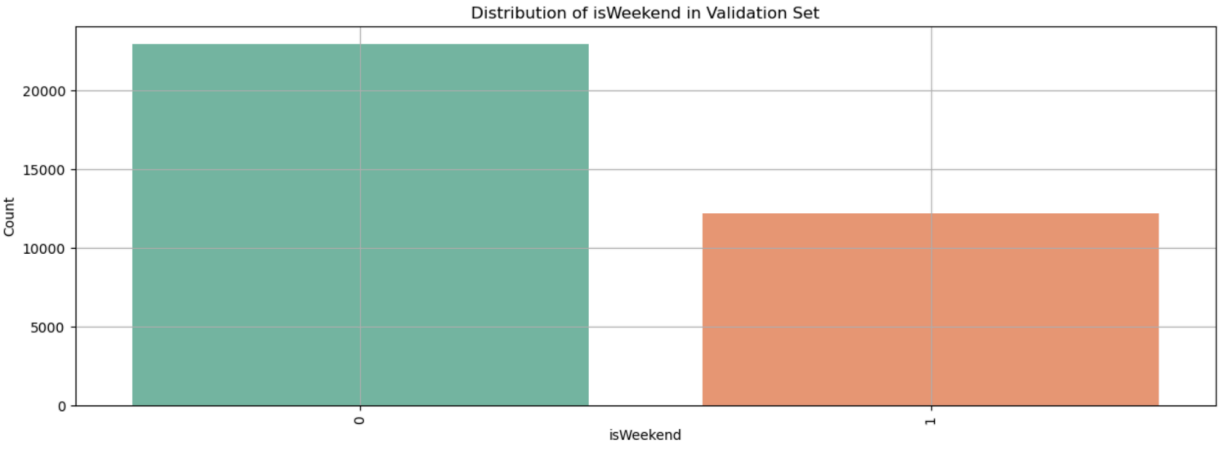
Observations:

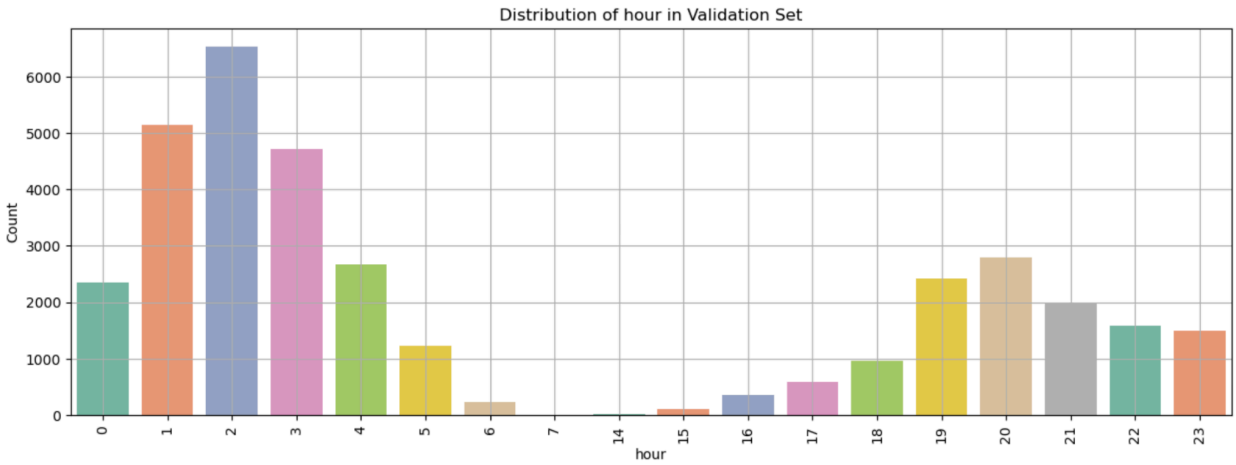
1. **Subtotal Distribution:** Most orders have a subtotal between ₹1000 and ₹3000, with frequency gradually tapering off beyond ₹4000.
2. **Number of Distinct Items:** The majority of orders contain 1–3 distinct items, and very few orders include more than 6 items.
3. **Total Outstanding Orders:** The majority of entries have fewer than 50 outstanding orders, with frequency sharply declining as the count increases.
4. **Distance:** Most deliveries occur within a 10–35 km range, with a peak around 20–25 km and a gradual drop beyond 40 km.

4.1.2. Check the distribution of categorical features


Distribution of market_id in Validation Set
Count
0
2000
4000
6000
8000
10000
1.0
2.0
3.0
4.0
5.0
6.0
market_id


Distribution of order_protocol in Validation Set
Count
0
2000
4000
6000
8000
10000
1.0
2.0
3.0
4.0
5.0
6.0
7.0
order_protocol

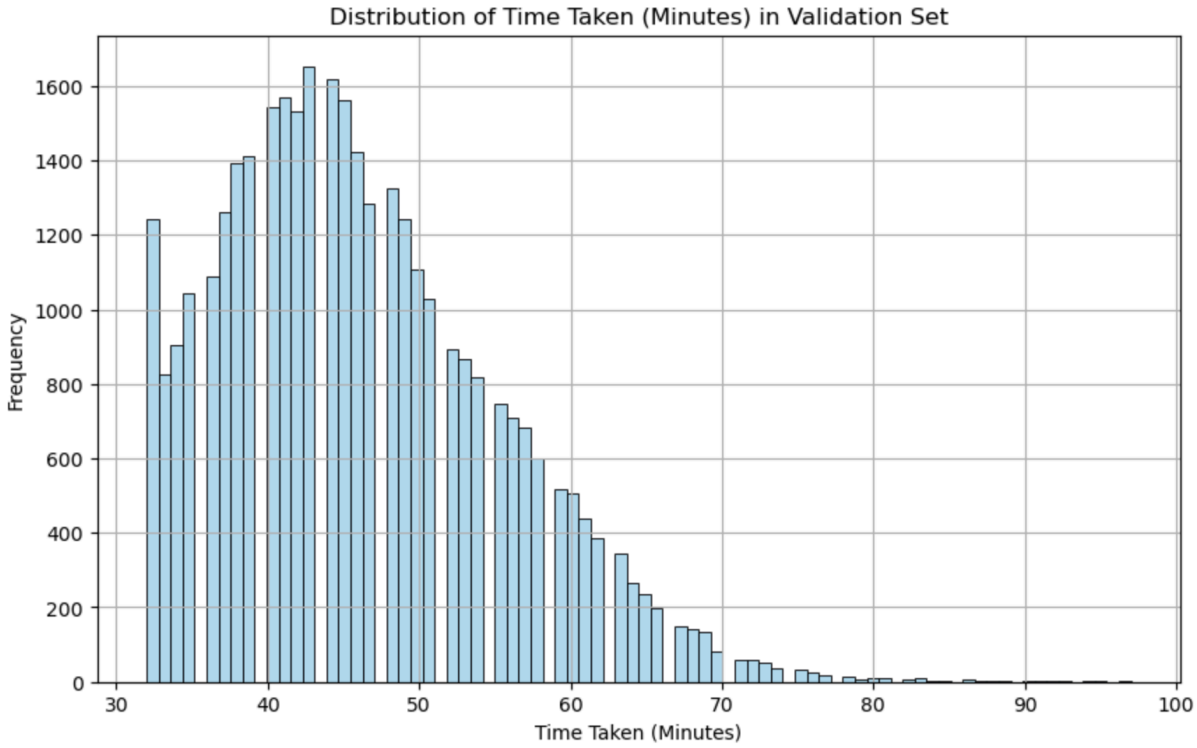

Distribution of isWeekend in Validation Set
Count
0
5000
10000
15000
20000
0
1
isWeekend



Observations:

1. **market_id:** The **market_id** distribution in the validation set is highly imbalanced. Markets 2, 4, and 1 dominate the data, while market 6 is severely underrepresented. This imbalance may lead to biased model performance, favoring the larger markets.
2. **order_protocol:** The **order_protocol** distribution in the validation set is uneven, with protocols 1, 3, and 5 being the most frequent. Protocol 6 and especially 7 are severely underrepresented, which may lead to poor model performance for those categories.
3. **isWeekend**: The majority of the orders in the validation set occurred on weekdays (isWeekend = 0), nearly double compared to weekends (isWeekend = 1), indicating higher order activity during weekdays.
4. **hours**: Orders in the validation set peak during early morning hours, especially at 2 AM, with another smaller peak around 8–10 PM. Activity drops sharply after 3 AM and remains low until the evening.

4.1.3. Visualise the distribution of the target variable to understand its spread and any skewness

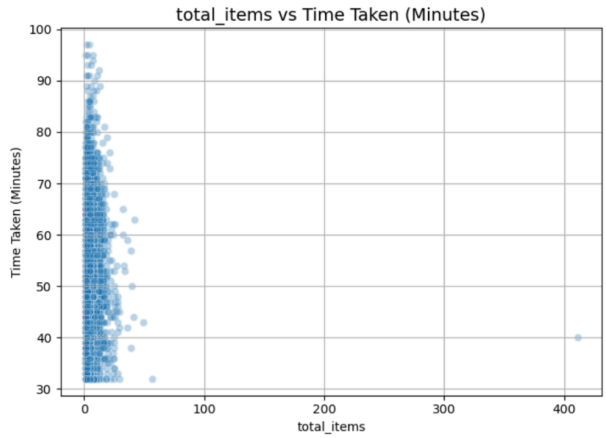


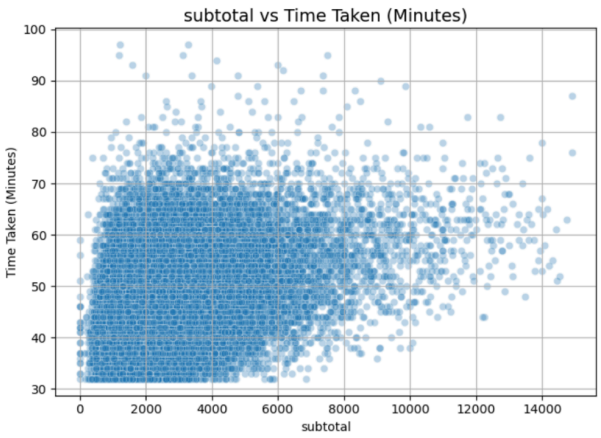
Observations:

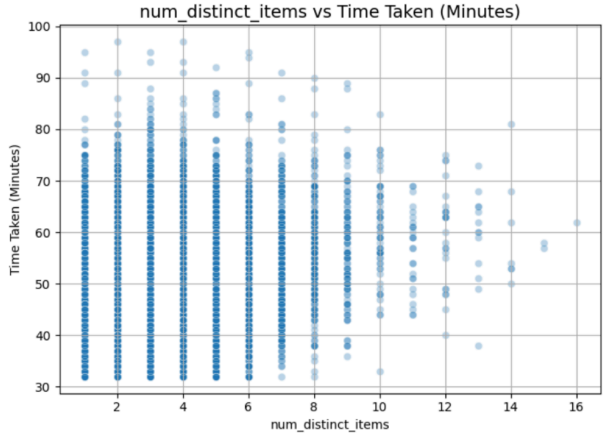
The delivery time distribution is right-skewed, with most deliveries occurring between 30–60 minutes and peaking around 45 minutes. Few deliveries exceed 70 minutes, suggesting possible outliers or delays. The distribution is asymmetric with a longer right tail.

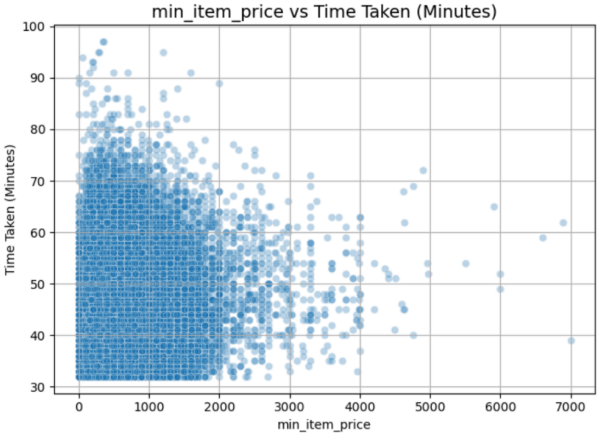
4.2. Relationships Between Features

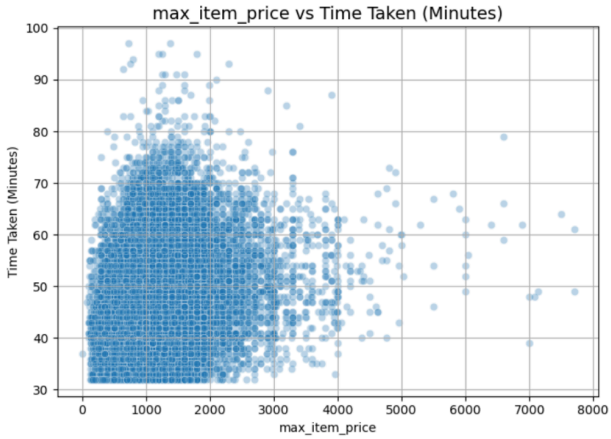
Scatter plots for numerical features to observe how they relate to each other, especially to `time_taken`

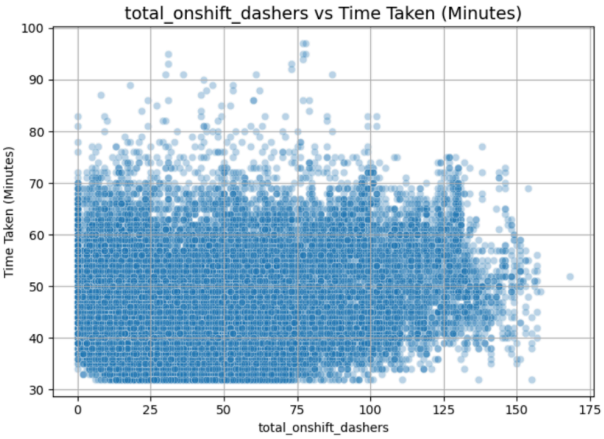

total_items vs Time Taken (Minutes)
Time Taken (Minutes)
total_items

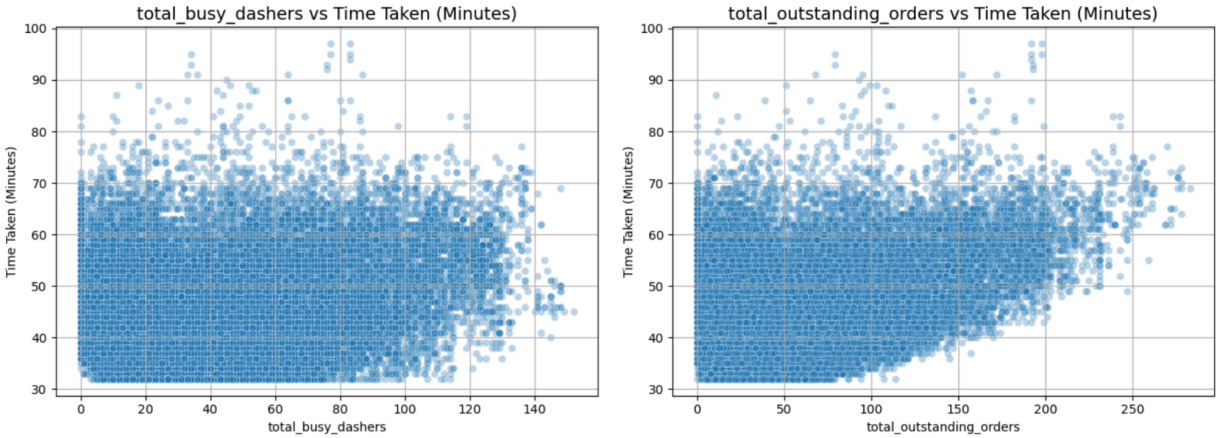

subtotal vs Time Taken (Minutes)
Time Taken (Minutes)
subtotal

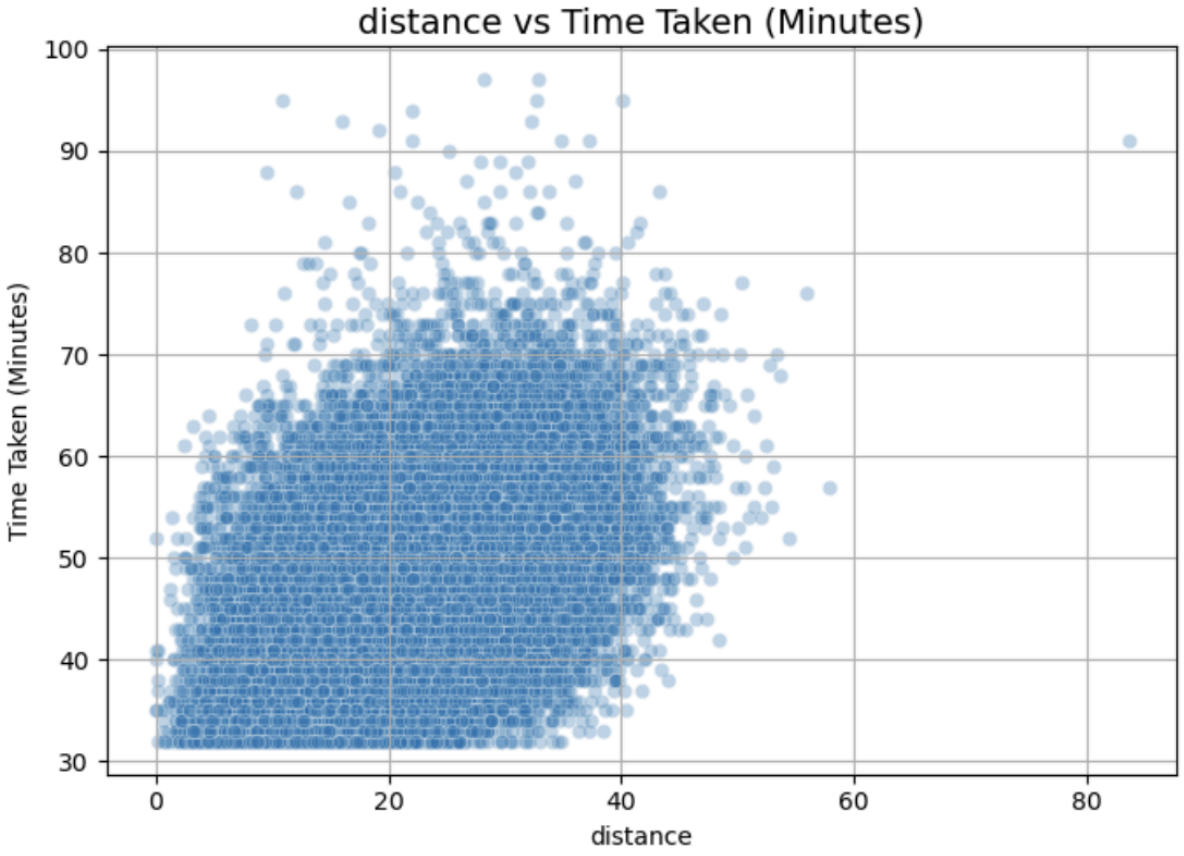

num_distinct_items vs Time Taken (Minutes)
Time Taken (Minutes)
num_distinct_items


min_item_price vs Time Taken (Minutes)
Time Taken (Minutes)
min_item_price


max_item_price vs Time Taken (Minutes)
Time Taken (Minutes)
max_item_price


total_onshift_dashers vs Time Taken (Minutes)
Time Taken (Minutes)
total_onshift_dashers





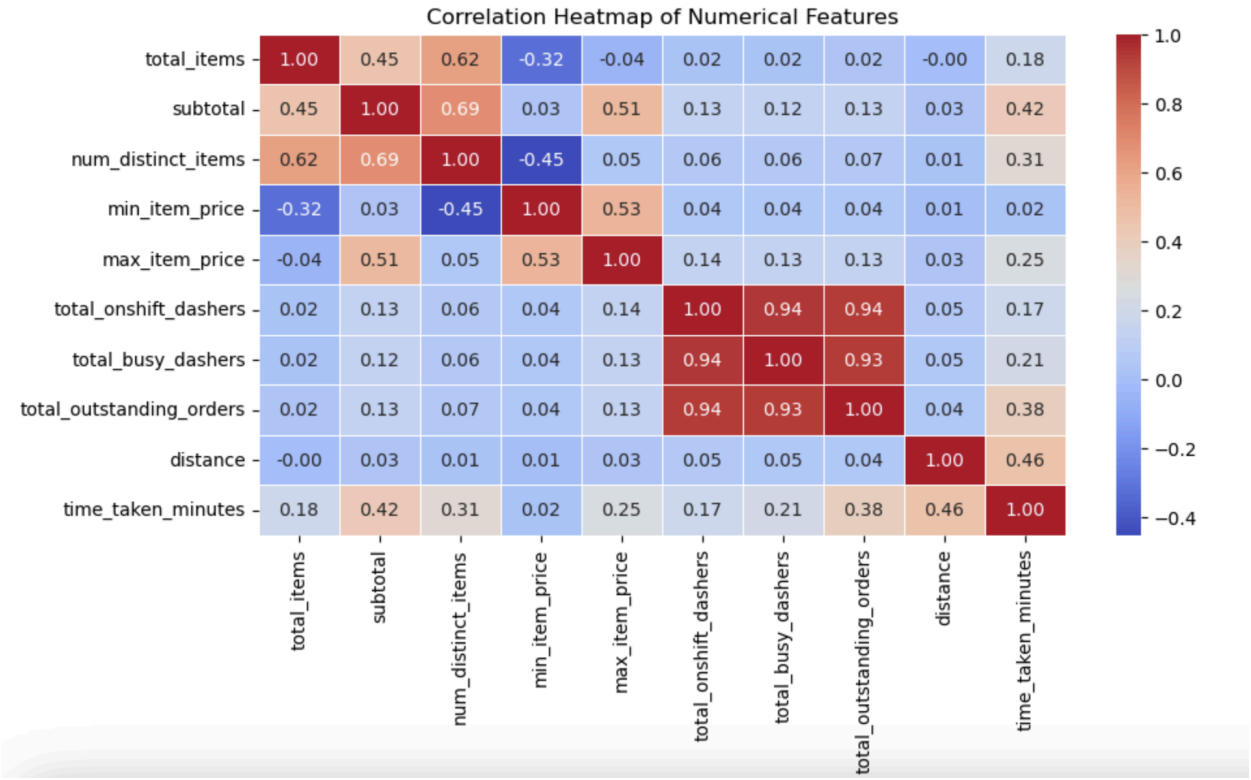
Observations:

1. **Total Items vs Time Taken**
   The scatter plot shows that most deliveries involve fewer than 50 items, with delivery times typically ranging from 35 to 80 minutes. There is no strong correlation between the number of items and time taken, though some high-item outliers are visible.
2. **Subtotal vs Time Taken**
   Deliveries mostly cluster around subtotals between ₹1,000 and ₹6,000, with time durations

typically between 35 and 75 minutes. There's no clear linear relationship, though higher subtotals show slightly more variation in delivery time.

3. **Distinct Items vs Time Taken**
   Most orders contain between 1 and 10 distinct items, with delivery times generally between 35 and 75 minutes. The time taken remains stable across varying item counts, indicating no significant correlation.
4. **Minimum Item Price vs Time Taken**
   Most orders have a minimum item price below ₹2,000, with delivery times concentrated between 35 and 75 minutes. There's no strong correlation, though higher minimum prices may be linked to slightly shorter delivery times, possibly due to fewer or premium items.
5. **Maximum Item Price vs Time Taken**
   Orders typically have a maximum item price under ₹3,000, with delivery times mostly between 40 and 75 minutes. No strong correlation is observed, though high-value items show a mild tendency toward shorter or average delivery durations.
6. **On-Shift Dashers vs Time Taken**
   Data points are spread across various dasher counts, with delivery times mainly between 40 and 70 minutes. A slight downward trend suggests that more available dashers may lead to marginally quicker deliveries, though the effect is weak.
7. **Busy Dashers vs Time Taken**
   Delivery times largely range between 40 and 70 minutes. While no clear correlation is evident, slightly longer times may occur when fewer dashers are active, though the distribution is widely scattered.
8. **Outstanding Orders vs Time Taken**
   A modest upward trend indicates that higher outstanding order volumes may slightly increase delivery times. Most data clusters around 0–150 outstanding orders and 40–70 minute delivery times.
9. **Distance vs Time Taken**
   There is a clear positive correlation between distance and delivery time—longer distances tend to result in longer delivery durations.

4.3. Drop the columns with weak correlations with the target variable



The correlation heatmap was used to identify features weakly correlated with delivery time. Based on this, five features — **min_item_price, total_onshift_dashers, total_items, total_busy_dashers, and max_item_price** — were dropped from the test dataset to improve model efficiency.

5. Model Building

5.1. Feature Scaling

The training and test datasets were combined to ensure consistent one-hot encoding for categorical variables (**market_id**, **order_protocol**, and **hour**). After creating dummy variables, the data was split back into **X_train** and **X_test**.

```
# Concatenate X_train and X_test
X_combined = pd.concat([X_train, X_test], axis=0)

# Create dummies
X_combined = pd.get_dummies(X_combined, columns=['market_id', 'order_protocol', 'hour'], drop_first=True, dtype=int)

# Now split back
X_train = X_combined.iloc[:len(X_train)]
X_test = X_combined.iloc[len(X_train):]
```

Apply scaling to the numerical columns:
Numerical features in the training and test sets were standardized using StandardScaler to normalize the data. The scaled values were then converted back into DataFrames with the original column names for further analysis or modeling.

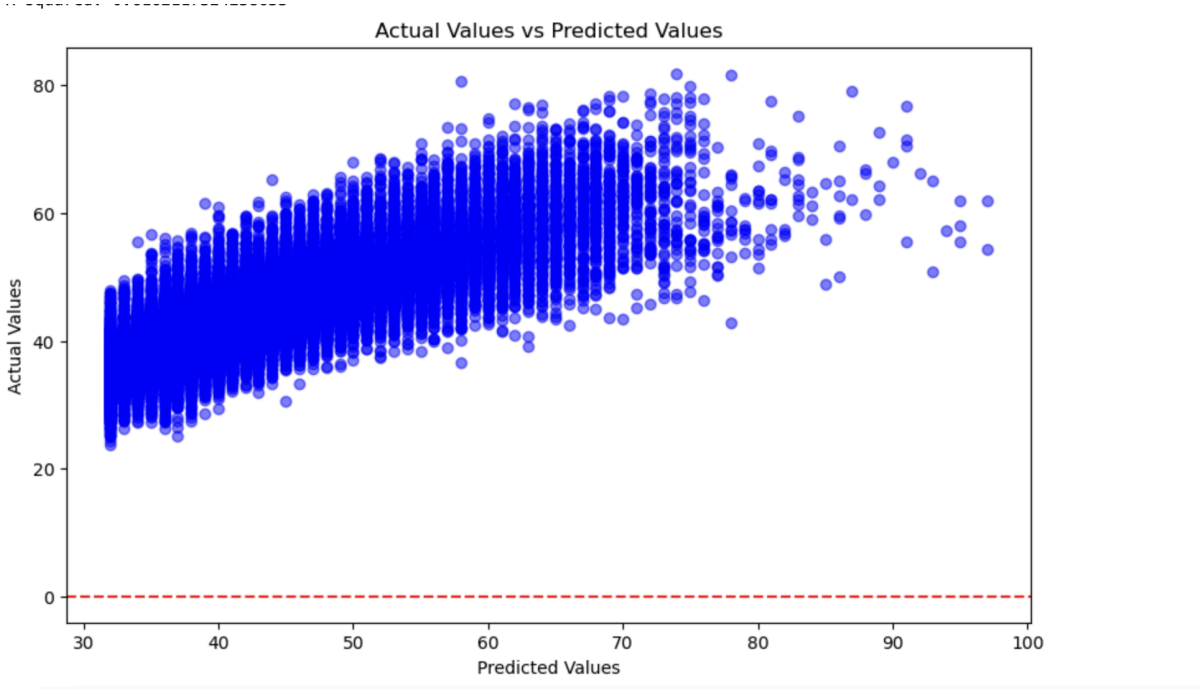
```
# Apply scaling to the numerical columns
scaler = StandardScaler()
X_train_scalar = scaler.fit_transform(X_train)
X_test_scalar = scaler.transform(X_test)

X_train_df=pd.DataFrame(X_train_scalar,columns=X_train.columns)
X_test_df=pd.DataFrame(X_test_scalar,columns=X_test.columns)
```

5.2. Build a linear regression model

To build a Linear Regression model using OLS (Ordinary Least Squares), the following structured steps should be followed, each with a clear purpose:

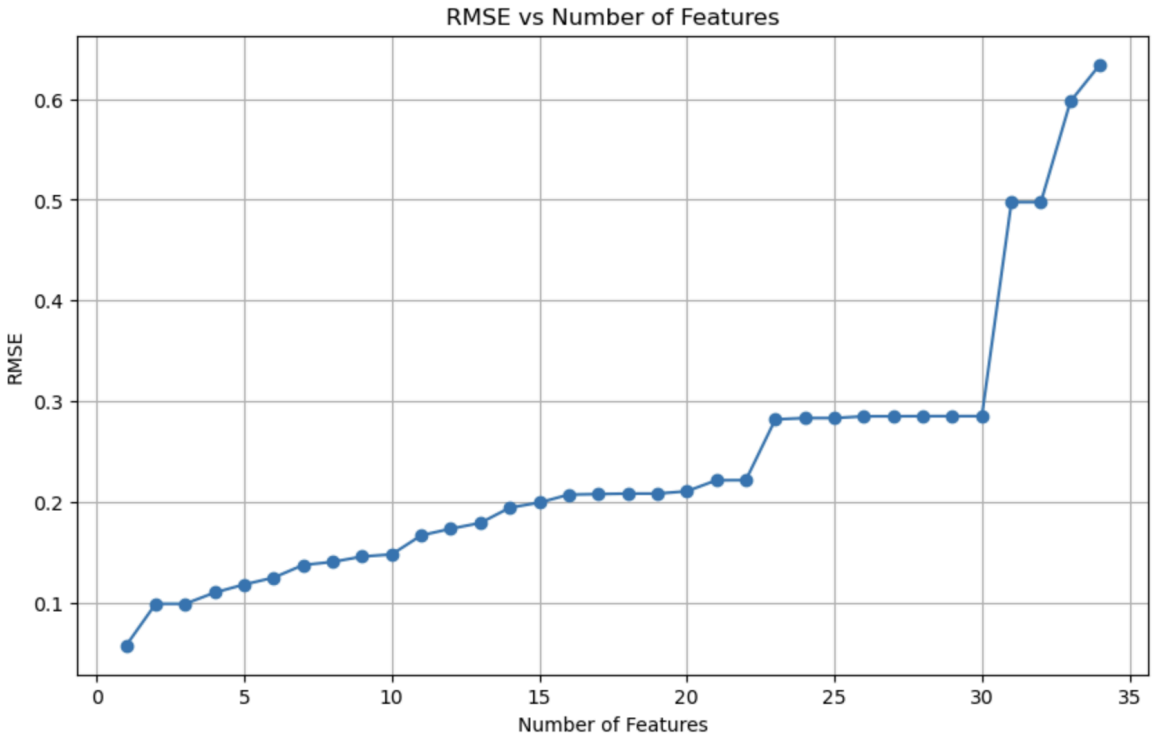
1. **Added a Constant Term**
   A constant (intercept) term was added to the training and testing feature sets using sm.add_constant() to include the bias term in the regression equation.
2. **Initialized and Trained the Model**
   An OLS model was initialized using sm.OLS() and trained on the training data using the .fit() method to estimate regression coefficients.
3. **Checked the Model Summary**
   The model summary was reviewed using results.summary(), which provided key statistics such as coefficients, R-squared, p-values, and standard errors.
4. **Made Predictions**
   Predictions were made on the test data using the trained model and results.predict() after adding the constant term to X_test.
5. **Evaluated the Model**
   Computed evaluation metrics to assess model performance on test data:
    a. **Mean Absolute Error (MAE):** 4.27 minutes
    b. **Mean Squared Error (MSE):** 30.99
    c. **Root Mean Squared Error (RMSE):** 5.57 minutes
    d. **R-squared:** 0.62



The scatter plot demonstrates a positive correlation between the predicted and actual values, indicating that the model captures the overall trend reasonably well. Most data points align along a diagonal pattern, suggesting good predictive performance. However, there is noticeable dispersion around the ideal line, especially at higher predicted values, implying increasing prediction errors as delivery time increases.

5.3. Build the model and fit RFE to select the most important features

A Recursive Feature Elimination (RFE) process was implemented by looping through models with 1 to 34 features. At each step, a linear regression model was trained, and performance was evaluated using RMSE and R^2 on the test set. The performance significantly improved as more features were added, with 31 to 34 features showing the best results.



Based on the RFE evaluation, the top 31 features mentioned below were selected for building the final model. These features were identified using rfe_final.get_support() and used to subset the training and test datasets.

A final linear regression model was trained using Recursive Feature Elimination (RFE) with 31 selected features. When evaluated on the test set, the model achieved a Root Mean Squared Error (RMSE) of 6.51 and an R^2 score of 0.4976, indicating a moderate level of predictive performance.

Selected Features: num_distinct_items, distance, isWeekend, market_id_2.0, market_id_3.0, market_id_4.0, market_id_5.0, market_id_6.0, order_protocol_2.0, order_protocol_3.0, order_protocol_4.0, order_protocol_5.0, order_protocol_6.0, order_protocol_7.0, hour_1, hour_2, hour_3, hour_5, hour_6, hour_7, hour_8, hour_14, hour_15, hour_16, hour_17, hour_18, hour_19, hour_20, hour_21, hour_22, hour_23

An Ordinary Least Squares (OLS) model was fitted using statsmodels to analyze feature significance. The summary output included coefficients, p-values, and R^2, helping assess the individual importance of each feature.

Variance Inflation Factor (VIF) was calculated for all selected features to check for multicollinearity. All VIF values were low (well below the threshold of 5), indicating no significant multicollinearity among predictors.

To further improve model interpretability and potentially enhance performance, feature selection was refined based on p-values and Variance Inflation Factor (VIF):

1. **Multicollinearity Check**:
   The initial VIF values for all selected features were between **1.0 and 1.9**, indicating **low multicollinearity** and confirming that the model does not suffer from redundancy among independent variables.
2. **Statistical Significance (p-value) Filtering**:
   Variables with **p-values greater than 0.05** were considered statistically insignificant and thus removed in a stepwise fashion to improve model clarity:
    a. **hour_8** war removed due to high p-value.
    b. Then, **order_protocol_7.0** was removed due to high p-value.
    c. After refitting the model, **hour_7** was removed next.
    d. Then, **hour_14** was dropped.
    e. Finally, **hour_15** was removed based on its persistently high p-value.
3. After each removal, the model was refitted using **Ordinary Least Squares (OLS)**, and both **summary statistics** and updated **VIF values** were reviewed to ensure:
    a. Remaining variables were statistically significant (p-value < 0.05),
    b. Multicollinearity remained within acceptable limits (VIF < 5).

OLS Regression Results:

```
==============================================================================
Dep. Variable:                      y   R-squared:                       0.488
Model:                            OLS   Adj. R-squared:                  0.488
Method:                 Least Squares   F-statistic:                     5151.
Date:                Fri, 02 May 2025   Prob (F-statistic):               0.00
Time:                        19:06:03   Log-Likelihood:            -4.6678e+05
No. Observations:              140549   AIC:                         9.336e+05
Df Residuals:                  140522   BIC:                         9.339e+05
Df Model:                          26
Covariance Type:            nonrobust
==============================================================================
                         coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const                 35.0154      0.086    407.210      0.000      34.847      35.184
num_distinct_items     1.7267      0.015    113.696      0.000       1.697       1.756
distance               0.4909      0.002    238.451      0.000       0.487       0.495
isWeekend              2.4458      0.038     64.542      0.000       2.371       2.520
market_id_2.0         -4.8317      0.052    -93.514      0.000      -4.933      -4.730
market_id_3.0         -5.3849      0.065    -82.966      0.000      -5.512      -5.258
market_id_4.0         -2.9313      0.054    -54.452      0.000      -3.037      -2.826
market_id_5.0         -4.6441      0.070    -66.809      0.000      -4.780      -4.508
market_id_6.0         -2.8556      0.299     -9.548      0.000      -3.442      -2.269
order_protocol_2.0    -0.7855      0.063    -12.418      0.000      -0.910      -0.662
order_protocol_3.0    -1.4299      0.051    -28.290      0.000      -1.529      -1.331
order_protocol_4.0    -3.3318      0.067    -49.859      0.000      -3.463      -3.201
order_protocol_5.0    -2.7607      0.052    -53.406      0.000      -2.862      -2.659
order_protocol_6.0    -2.0824      0.286     -7.293      0.000      -2.642      -1.523
hour_1                 0.9533      0.067     14.310      0.000       0.823       1.084
hour_2                 4.1272      0.063     65.674      0.000       4.004       4.250
hour_3                 4.0390      0.068     59.531      0.000       3.906       4.172
hour_5                -1.1220      0.107    -10.456      0.000      -1.332      -0.912
hour_6                -1.1526      0.220     -5.245      0.000      -1.583      -0.722
hour_16               -1.4065      0.175     -8.026      0.000      -1.750      -1.063
hour_17               -2.0071      0.143    -14.009      0.000      -2.288      -1.726
hour_18               -3.3788      0.122    -27.672      0.000      -3.618      -3.139
hour_19               -1.8592      0.083    -22.376      0.000      -2.022      -1.696
hour_20               -3.7201      0.079    -46.845      0.000      -3.876      -3.564
hour_21               -5.2843      0.088    -60.183      0.000      -5.456      -5.112
hour_22               -4.8724      0.097    -50.261      0.000      -5.062      -4.682
hour_23               -4.5545      0.100    -45.681      0.000      -4.750      -4.359
```

Final VIF Summary:

	Feature	VIF
0	const	23.143563
1	num_distinct_items	1.033225
2	distance	1.010166
3	isWeekend	1.015876
4	market_id_2.0	1.774031
5	market_id_3.0	1.387215
6	market_id_4.0	1.753298
7	market_id_5.0	1.339148
8	market_id_6.0	1.014244
9	order_protocol_2.0	1.313258
10	order_protocol_3.0	1.565437

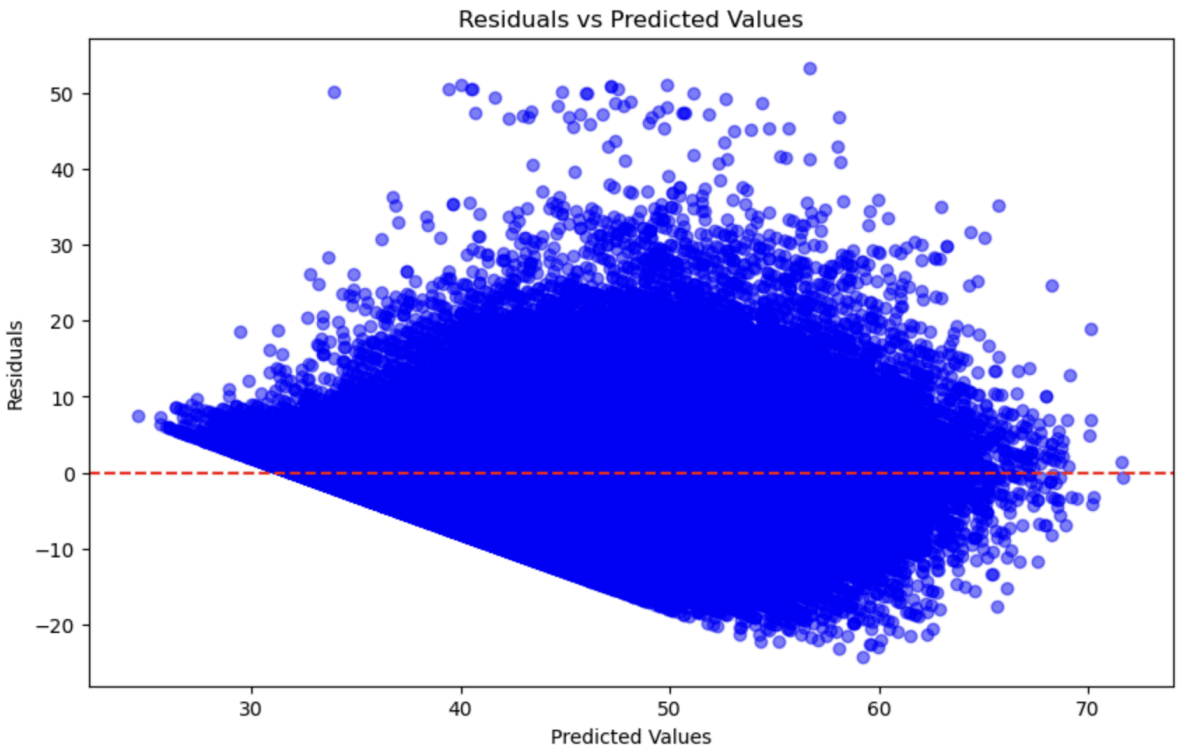
11	order_protocol_4.0	1.233249
12	order_protocol_5.0	1.505542
13	order_protocol_6.0	1.014580
14	hour_1	1.735631
15	hour_2	1.883992
16	hour_3	1.682716
17	hour_5	1.200668
18	hour_6	1.048501
19	hour_16	1.068603
20	hour_17	1.105779
21	hour_18	1.148572
22	hour_19	1.382199
23	hour_20	1.432978
24	hour_21	1.330261
25	hour_22	1.256920
26	hour_23	1.239496

6. Results and Inference

6.1. Perform Residual Analysis

1. **Residual Analysis** was conducted to assess the assumptions of linear regression and validate model fit:

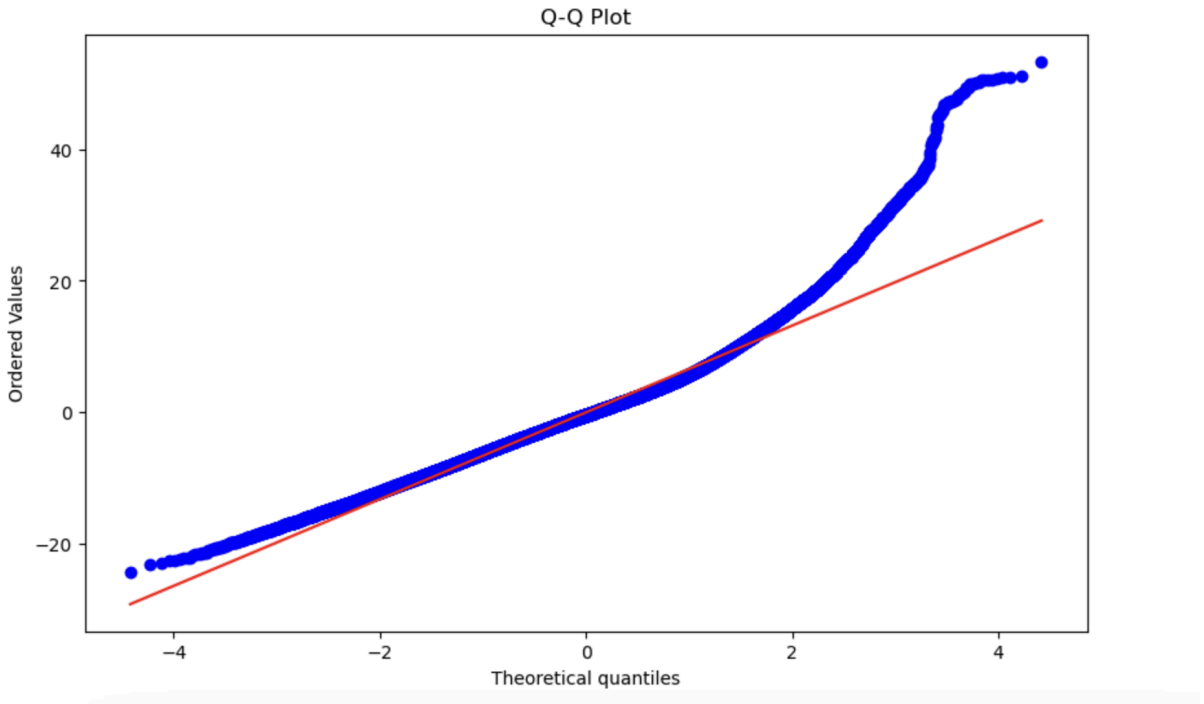
   Residuals vs Predicted Values Plot:



Observations:

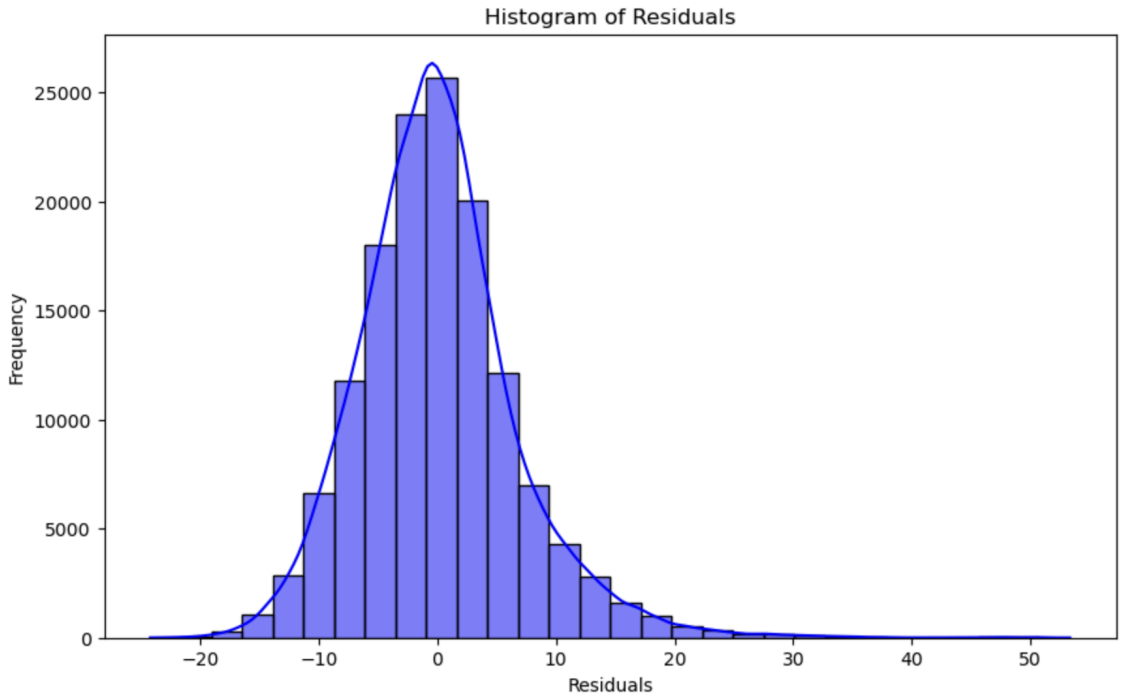
1. **Funnel-Shaped Pattern:**
    a. The spread of residuals increases as the predicted values increase.
    b. This indicates **heteroscedasticity**, i.e., the variance of errors is not constant — violating a key assumption of linear regression.
2. **Residuals Centered Around Zero:**
    a. Despite the varying spread, most residuals are symmetrically distributed around the zero line, which supports the **linearity** of the model.
3. **No Clear Outliers:**
    a. While there are a few large residuals, the plot doesn't show extreme outliers that would heavily distort the model.
4. **Possible Model Misspecification:**
    a. The pattern in the spread suggests that some non-linear relationships may exist that the linear model isn't capturing well.
5. **Implication for Model Validity:**
    a. The violation of the constant variance assumption (homoscedasticity) could lead to **inefficient estimates** and **biased statistical inference** (e.g., p-values, confidence intervals).

2. **Q-Q Plot**:



The Q-Q plot shows significant deviation from the reference line, especially at the tails, indicating that the residuals are not normally distributed and have heavy tails (presence of outliers or skewness).

3. Histogram of Residuals:
The histogram confirmed the **near-normal distribution** of residuals, reinforcing the Q-Q plot observations.



The histogram of residuals is approximately bell-shaped but right-skewed, indicating that while the residuals are roughly centered around zero, they deviate from perfect normality due to the presence of positive outliers.

Model Evaluation Summary (R^2 Score)
Training $R^2 = 0.49$: The model explains 49% of the variance in delivery time on the training data.
Testing $R^2 = 0.50$: The model explains 50% of the variance in delivery time on the unseen test data.

6.2. Perform Coefficient Analysis

A coefficient analysis was performed to evaluate how different features influence the target variable, **time_taken_minutes**, by examining both **scaled** and **unscaled** coefficients:

1. **Scaled Coefficients**:
   These reflect the standardized effect of each feature, allowing for comparison of their relative importance regardless of differing units or scales. They are useful for identifying which features have the strongest influence in the model overall.
2. **Unscaled Coefficients**:
   These provide the real-world interpretation of the model, indicating the actual change in delivery time (in minutes) associated with a one-unit change in each feature. This allows for practical understanding of the feature impact.

By comparing the scaled and unscaled coefficients, the analysis revealed which features were not only statistically significant but also practically impactful.

Key Insights

The most influential features affecting delivery time included:

1. **num_distinct_items** — A higher number of distinct items in an order was associated with increased delivery time.
2. **isWeekend** — Deliveries made during weekends tended to take longer.
3. Specific **hour** indicators — Orders placed during certain hours significantly affected delivery times.
4. **market_id** categories — Different market locations had varying impacts on the time taken for delivery.

	Scaled Coefficients	Unscaled Coefficients
const	35.041134	35.041134
num_distinct_items	1.725757	1.725757
distance	0.488789	0.488789
isWeekend	2.449446	2.449446
market_id_2.0	-4.806472	-4.806472
market_id_3.0	-5.377090	-5.377090
market_id_4.0	-2.908616	-2.908616
market_id_5.0	-4.624064	-4.624064
market_id_6.0	-2.838215	-2.838215
order_protocol_2.0	-0.786162	-0.786162
order_protocol_3.0	-1.435369	-1.435369
order_protocol_4.0	-3.327144	-3.327144
order_protocol_5.0	-2.761628	-2.761628
order_protocol_6.0	-2.120450	-2.120450
hour_1	0.948358	0.948358
hour_2	4.127595	4.127595
hour_3	4.039311	4.039311
hour_5	-1.124125	-1.124125
hour_6	-1.155930	-1.155930
hour_16	-1.403666	-1.403666
hour_17	-2.008445	-2.008445
hour_18	-3.378594	-3.378594
hour_19	-1.861073	-1.861073
hour_20	-3.714967	-3.714967
hour_21	-5.282746	-5.282746
hour_22	-4.874367	-4.874367
hour_23	-4.561419	-4.561419

We further calculated the "Effect of 1 Unit Change" by dividing the scaled coefficients by the standard deviation of each corresponding numeric feature. This quantifies how much delivery time changes with a one-unit increase in the original feature.

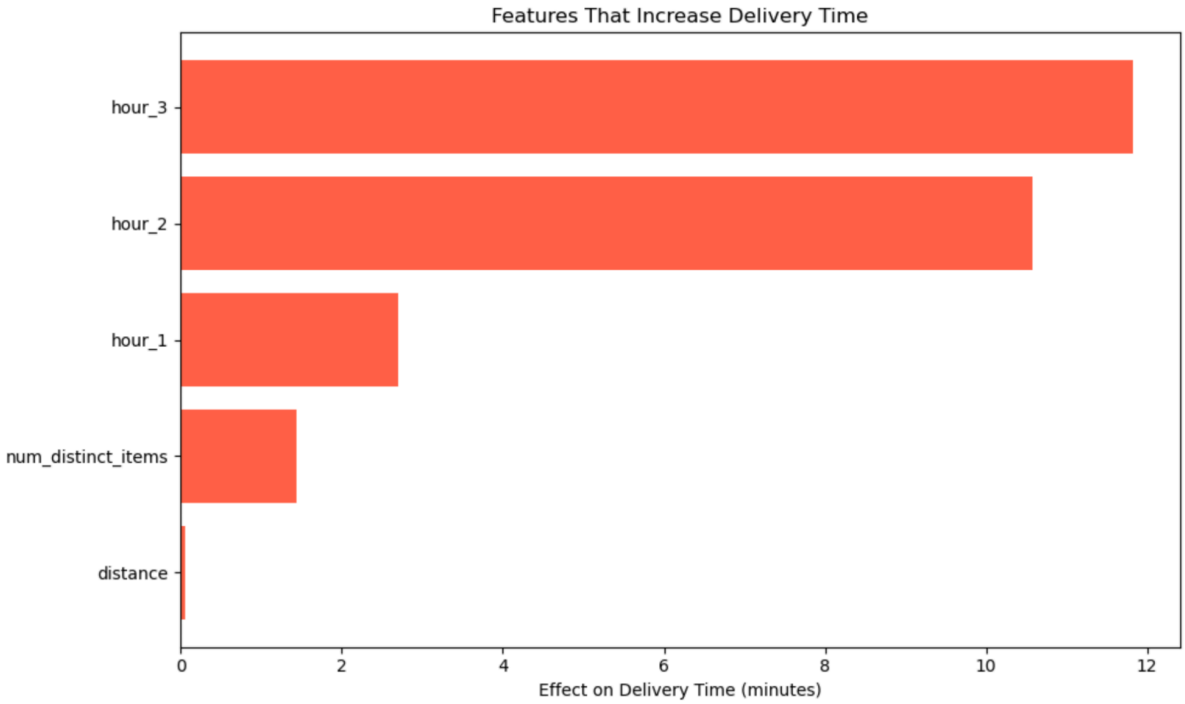
Features Increasing Delivery Time (Calculated Based on One-Unit Change Impact):

This analysis identified the features whose **one-unit increase led to longer delivery times**, based on the "Effect of 1 Unit Change" derived from a trained linear regression model.

Key features that significantly increased delivery time:

1. **hour_3**: +11.82 minutes
2. **hour_2**: +10.57 minutes
3. **hour_1**: +2.68 minutes
4. **num_distinct_items**: +1.44 minutes
5. **distance**: +0.06 minutes

These findings indicated that **early morning hours (1 AM to 3 AM)** were strongly associated with increased delivery durations—likely due to operational inefficiencies, lower staffing, or reduced road activity. Additionally, a higher number of distinct items and longer travel distances also contributed to longer delivery times.



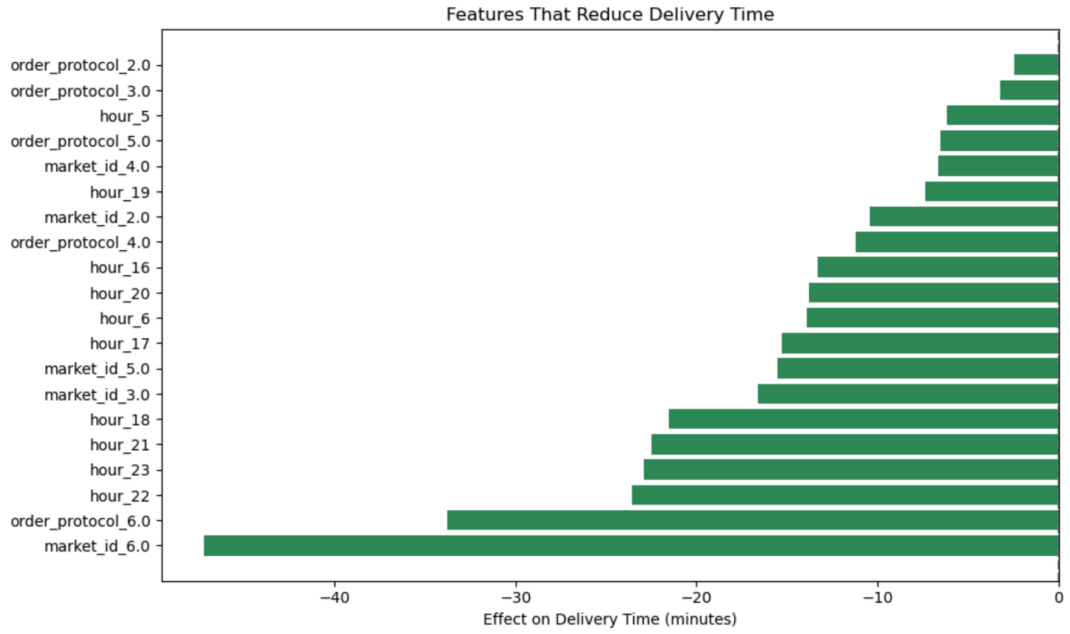
Features decreasing Delivery Time (Calculated Based on One-Unit Change Impact):

This analysis identified the features whose one-unit increase led to shorter delivery times, based on the "Effect of 1 Unit Change" derived from the trained linear regression model.

Key features that significantly reduced delivery time:

1. **market_id_6.0**: Reduces delivery time by approximately **47.21 minutes**.
2. **order_protocol_6.0**: Reduces delivery time by approximately **33.75 minutes**.
3. **hour_22**: Reduces delivery time by approximately **23.58 minutes** when orders are placed at 10 PM.
4. **hour_23**: Reduces delivery time by approximately **22.88 minutes** at 11 PM.

5. **hour_21**: Reduces delivery time by approximately **22.49 minutes** at 9 PM.



Conclusion:
While scaled and unscaled coefficients vary in magnitude, they convey the same underlying relationships between features and the target variable.
For practical, business-oriented interpretation, it's recommended to focus on **unscaled coefficients** or the **effect of a 1-unit change**, as these reflect real-world impacts in original units and offer clearer guidance for decision-making.