Comparative Analysis of Machine Learning Models for Delivery Time Prediction

Group 6

Sai Soumya Aloor
Sri Charan Desetty
Venkata Naga Anirudh Chaganti
Anantha Prahlada Kurudi
Hemanth Varma Pericherla

Table of Contents

1. Introduction

As part of our final capstone project for this course, our group undertakes the comprehensive application of the entire data science workflow that we have learned throughout the curriculum. This includes critical steps such as data sourcing, cleaning, preprocessing, exploratory data analysis (EDA), modeling, evaluation, and result interpretation.

Our goal is to apply these concepts to a real-world scenario in a way that demonstrates not only our technical proficiency but also our ability to solve practical problems using data driven decision making.

We decide to focus our project on analyzing and predicting food delivery times, a highly relevant and impactful issue in today's fast paced and logistics driven economy. Timely deliveries are a key performance metric for food delivery platforms and logistics providers, directly influencing customer satisfaction, operational efficiency, and overall business outcomes.

By working on this problem, we aim to extract actionable insights from the data and build robust predictive models that could, in theory, help businesses optimize their delivery operations. This project allows us to bring together everything we have learned and apply it to a practical challenge that has significant realworld implications.

2. Topic and Dataset for the Project

We select our dataset from Kaggle, titled "Food Delivery Time: A Multi Factor Dataset." This dataset is an excellent fit for our analysis for several key reasons:

1. Relevant Variables

The dataset contains several factors that directly influence delivery time, such as:
Traffic conditions
Weather descriptions
Delivery person ratings
Delivery distances
These variables are essential for understanding and modeling the dynamics of food delivery performance.

2. Real World Application

The dataset closely mirrors real-world delivery scenarios, making it highly applicable to industries like logistics, food delivery, and ecommerce. This makes our findings potentially valuable for optimizing real operational systems.

3. Variety of Data Types

It includes both numerical features (e.g., temperature, delivery distance, delivery time) and categorical features (e.g., type of vehicle, type of order, weather conditions). This diversity allows us to explore a wide range of data preprocessing and modeling techniques.

4. Predictive Modeling Potential

Since the target variable delivery time is continuous, we can apply various regression algorithms and explore time series patterns. This gives us the opportunity to build predictive models that can estimate delivery times under different conditions.

3. Data collection, preprocessing and data cleanup

For our project, we utilize the "Food Delivery Time: A multifactor Dataset" sourced from Kaggle. This dataset is chosen because it presents a well-rounded mix of features that influence food delivery times, making it highly suitable for a predictive modeling task. It simulates a real-world logistics scenario by incorporating delivery personnel characteristics, environmental conditions, and situational factors such as traffic and weather elements that are often overlooked in basic delivery prediction models used by platforms like Zomato, Blinkit, or Swiggy.

Below is a table of all the attributes:

Column Name	**Description**
ID	A unique identifier for each delivery.
Delivery_person_ID	A unique identifier assigned to each delivery person for tracking purposes.
Delivery_person_Age	Age of the delivery person.
Delivery_person_Ratings	Customer ratings of the delivery person.
Restaurant_latitude	Geographical latitude coordinate of the restaurant's location.
Restaurant_longitude	Geographical longitude coordinate of the restaurant's location.
Delivery_location_latitude	Latitude coordinate of the delivery location where the order is delivered.
Delivery_location_longitude	Longitude coordinate of the delivery location for the order.
Type_of_order	Category of food ordered (e.g., meal, snacks, drinks, buffet) to analyze preparation times.
Type_of_vehicle	The vehicle used for delivery (e.g., scooter, motor cycle , cycle ,ev scooter), which affects speed and travel time.
Temperature	Ambient temperature during the delivery time, potentially impacting delivery efficiency.
Humidity	Level of atmospheric moisture during delivery, affecting conditions for travel.
Precipitation	Amount of rainfall or snow, indicating weather disruptions during delivery.
Weather_description	Textual description of the weather (e.g., sunny, cloudy, stormy) for context in travel conditions.
Traffic_Level	Severity of traffic congestion during the delivery (e.g., low, medium, high).
Distance (km)	The calculated distance between the restaurant and the delivery location in kilometers.
TARGET	The target variable representing the delivery time in minutes for model predictions.

3.1 Data Collection:

In this part we will focus on importing the data set in csv and applying basic starts to explore the raw data. We named out raw dataset as data [Data frame in pandas]

1) **.info():** We applied this to understand the datatype of each attribute.

```
Data columns (total 18 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   ID                          10000 non-null  object
 1   Delivery_person_ID          10000 non-null  object
 2   Delivery_person_Age         10000 non-null  float64
 3   Delivery_person_Ratings     10000 non-null  float64
 4   Restaurant_latitude         10000 non-null  float64
 5   Restaurant_longitude        10000 non-null  float64
 6   Delivery_location_latitude  10000 non-null  float64
 7   Delivery_location_longitude 10000 non-null  float64
 8   Type_of_order               10000 non-null  object
 9   Type_of_vehicle             10000 non-null  object
 10  temperature                 9995 non-null   float64
 11  humidity                    9995 non-null   float64
 12  precipitation               9995 non-null   float64
 13  weather_description         9995 non-null   object
 14  Unnamed: 14                 0 non-null      float64
 15  Traffic_Level               9085 non-null   object
 16  Distance (km)               9080 non-null   float64
 17  TARGET                      9459 non-null   object
dtypes: float64(11), object(7)
memory usage: 1.4+ MB
None
```

Inference:

1. **Numerical columns**: 11 columns are of type float64, indicating continuous or numeric data such as age, ratings, weather conditions, and distance

2. **Categorical columns**: 7 columns are of type object, representing identifiers and categorical variables like Type_of_order, Type_of_vehicle, and Traffic_Level.

2) **.describe():** We applied this to understand the stats of each attribute.

```
       Delivery_person_Age  Delivery_person_Ratings  Restaurant_latitude  \
count         10000.000000             10000.000000         10000.000000
mean             29.522000                 4.629370            16.893418
std               5.700348                 0.322941             8.330948
min              15.000000                 1.000000           -30.902872
25%              25.000000                 4.500000            12.913041
50%              29.000000                 4.700000            18.546258
75%              34.000000                 4.800000            22.727021
max              50.000000                 6.000000            30.914057

       Restaurant_longitude  Delivery_location_latitude  \
count          10000.000000                10000.000000
mean              70.177749                   17.412655
std               23.203352                    7.336846
min              -88.352885                    0.010000
25%               73.170937                   12.983959
50%               75.902847                   18.626216
75%               78.047717                   22.785089
max               88.433452                   31.054057

       Delivery_location_longitude  temperature     humidity  precipitation  \
count                 10000.000000  9995.000000  9995.000000    9995.000000
mean                     70.880072    22.936907    66.164882       0.016233
std                      21.174585     3.379448    15.602939       0.074911
min                       0.010000     6.770000    27.000000       0.000000
...
25%          NaN      7.620000
50%          NaN     13.400000
75%          NaN     19.610000
max          NaN     59.840000
```

Inference:

1. Age & Ratings: Most delivery persons are aged 25–34 with ratings between 4.5–5. Some outliers exist.

2. Weather: Avg temp ~23°C, humidity ~66%, and low precipitation overall.

3. Location: Most lat/long values are normal, but some are outliers.

4. Distance : Typical delivery is around 13 km, max ~60 km.

3) **.value_columns() :** We applied this to understand what is the number of unique values in each attribute.

```
data['Delivery_person_ID'].value_counts()
✓ 0.0s

SURRES16DEL01        22
CHENRES01DEL02       22
RANCHIRES18DEL01     22
COIMBRES06DEL01      22
COIMBRES03DEL02      20
                     ..
DEHRES16DEL01         1
AURGRES13DEL03        1
KNPRES08DEL03         1
BHPRES17DEL01         1
ALHRES13DEL01         1
Name: Delivery_person_ID, Length: 1285, dtype: int64
```

```
data['ID'].value_counts()
✓ 0.0s

6.00E+02    2
6.00E+03    2
BEF 1.00    2
9.00E+02    2
5.00E+09    2
           ..
4481        1
81AE        1
B900        1
9417        1
3FB2        1
Name: ID, Length: 9995, dtype: int64
```

```
data['Type_of_order'].value_counts()
✓ 0.0s

Snack      2551
Meal       2530
Drinks     2507
Buffet     2412
Name: Type_of_order, dtype: int64
```

```
data['Type_of_vehicle'].value_counts()
✓ 0.0s

motorcycle          5862
scooter             3304
electric_scooter     814
bicycle               20
Name: Type_of_vehicle, dtype: int64
```

```
data['weather_description'].value_counts()
✓ 0.0s

clear sky           3260
haze                2406
mist                1751
broken clouds        721
light rain           536
smoke                501
scattered clouds     422
overcast clouds      308
fog                   49
few clouds            40
moderate rain          1
Name: weather_description, dtype: int64
```

```
data['TARGET'].value_counts()
✓ 0.0s

#VALUE!        419
33.36666667     12
29.88333333     11
30.53333333     11
34.8            11
               ...
9.583333333      1
63.2             1
99.85            1
37.11666667      1
51.06666667      1
Name: TARGET, Length: 3389, dtype: int64
```

Inference :

1. Top Delivery Persons : Some IDs appear up to 22 times, indicating frequent assignments.

2. Order IDs : Mostly unique, but a few duplicates and anomalies like scientific notation or malformed IDs (e.g., "BEF 1.00").

3. Order Type : Fairly balanced — Snacks (2551), Meals (2530), Drinks (2507), Buffet (2412).

4. Vehicle Type : Motorcycles dominate (5862), followed by scooters; very few bicycles.

5. Weather : Mostly clear sky, haze, and mist. Rare cases of rain or fog.

6. Traffic Levels : Majority face high to moderate traffic; very low traffic is least common.

7. TARGET Issues : 419 values are invalid (`#VALUE!`). The rest vary widely from ~9 to 99 minutes, with many repeated durations.

3.2 Preprocessing and Data cleanup:

1. In this first we will focus on clearing on NaN values. We found many NaN value using the function data.isna().sum().
2. To Drop these NaN value we used data .dropna(inplace=True)

```
ID                            1
Delivery_person_ID            1
Delivery_person_Age           1
Delivery_person_Ratings       1
Restaurant_latitude           1
Restaurant_longitude          1
Delivery_location_latitude    1
Delivery_location_longitude   1
Type_of_order                 1
Type_of_vehicle               1
temperature                   6
humidity                      6
precipitation                 6
weather_description           6
Unnamed: 14               10001
Traffic_Level               916
Distance (km)               921
TARGET                      542
dtype: int64
```

Dropping NaN values

.dropna(inplace=True)

```
ID                             0
Delivery_person_ID             0
Delivery_person_Age            0
Delivery_person_Ratings        0
Restaurant_latitude            0
Restaurant_longitude           0
Delivery_location_latitude     0
Delivery_location_longitude    0
Type_of_order                  0
Type_of_vehicle                0
temperature                    0
humidity                       0
precipitation                  0
weather_description            0
Traffic_Level                  0
Distance (km)                  0
TARGET                         0
dtype: int64
```

3. Next step we have taken to calculate the distance between the restraint and the delivery address using the longitude and latitude.

temperature	humidity	precipitation	weather_description	Traffic_Level	Distance (km)	TARGET	Calculated_Distance
19.50	93.0	0.0	mist	Very High	37.17	85.26666667	20.183530
20.45	91.0	0.0	mist	Low	3.34	28.58333333	1.552758
23.86	78.0	0.0	mist	Moderate	10.05	35.18333333	7.790401
26.55	87.0	0.0	mist	High	9.89	43.45	6.210138
21.43	65.0	0.0	broken clouds	Moderate	11.30	30.6	4.610365
...	...	...	...	...	...	...	...
28.03	57.0	0.0	smoke	Low	3.78	18.2	1.529877
23.96	64.0	0.0	haze	High	18.92	32.61666667	13.631344
22.94	60.0	0.0	haze	Low	2.64	12.01666667	1.536621
23.72	31.0	0.0	clear sky	Very High	28.80	51.06666667	20.851557
28.01	57.0	0.0	smoke	High	17.63	43.8	13.771133

4. Next, we have removed 5 columns and created a new Data frame as df. Since "ID","Delivery_person_ID", "Restaurant_latitude", "Restaurant_longitude", "Delivery_location_latitude", “Delivery_location_longitude" are no longer needed.
5. We observed that 5 columns as objects: "Type_of_order", "Type_of_vehicle", "weather_description", "Traffic_Level", “TARGET" are Object Datatype.
6. We converted them to numerical or structured format using onehot and ordinal encoding. values in TARGET column must be converted to numeric as the values are in numeric values we did it by using (df['TARGET'] = pd.to_numeric(df['TARGET'], errors= ‘coerce')

```
Data columns (total 12 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Delivery_person_Age     9035 non-null   float64
 1   Delivery_person_Ratings 9035 non-null   float64
 2   Type_of_order           9035 non-null   object
 3   Type_of_vehicle         9035 non-null   object
 4   temperature             9035 non-null   float64
 5   humidity                9035 non-null   float64
 6   precipitation           9035 non-null   float64
 7   weather_description     9035 non-null   object
 8   Traffic_Level           9035 non-null   object
 9   Distance (km)           9035 non-null   float64
 10  TARGET                  9035 non-null   object
 11  Calculated_Distance     9035 non-null   float64
dtypes: float64(7), object(5)
```

→

```
Data columns (total 24 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   Delivery_person_Age                 9035 non-null   float64
 1   Delivery_person_Ratings             9035 non-null   float64
 2   temperature                         9035 non-null   float64
 3   humidity                            9035 non-null   float64
 4   precipitation                       9035 non-null   float64
 5   Traffic_Level                       9035 non-null   int64
 6   Distance (km)                       9035 non-null   float64
 7   TARGET                              9035 non-null   float64
 8   Calculated_Distance                 9035 non-null   float64
 9   Type_of_order_Drinks                9035 non-null   uint8
 10  Type_of_order_Meal                  9035 non-null   uint8
 11  Type_of_order_Snack                 9035 non-null   uint8
 12  Type_of_vehicle_electric_scooter    9035 non-null   uint8
 13  Type_of_vehicle_motorcycle          9035 non-null   uint8
 14  Type_of_vehicle_scooter             9035 non-null   uint8
 15  weather_description_clear sky       9035 non-null   uint8
 16  weather_description_few clouds      9035 non-null   uint8
 17  weather_description_fog             9035 non-null   uint8
 18  weather_description_haze            9035 non-null   uint8
 19  weather_description_mist            9035 non-null   uint8
...
 22  weather_description_scattered clouds  9035 non-null   uint8
 23  weather_description_smoke           9035 non-null   uint8
dtypes: float64(8), int64(1), uint8(15)
```

7. After Preprocessing and cleaning of data we get the below with No NaN values and all the attributes containing numeric values

```
Delivery_person_Age                     0
Delivery_person_Ratings                 0
temperature                             0
humidity                                0
precipitation                           0
Traffic_Level                           0
Distance (km)                           0
TARGET                                  0
Calculated_Distance                     0
Type_of_order_Drinks                    0
Type_of_order_Meal                      0
Type_of_order_Snack                     0
Type_of_vehicle_electric_scooter        0
Type_of_vehicle_motorcycle              0
Type_of_vehicle_scooter                 0
weather_description_clear sky           0
weather_description_few clouds          0
weather_description_fog                 0
weather_description_haze                0
weather_description_mist                0
weather_description_moderate rain       0
weather_description_overcast clouds     0
weather_description_scattered clouds    0
weather_description_smoke               0
dtype: int64
```

4. Summarize the Data and construct data visualizations.

4.1 Summarizing the Data:

1. First, we will summarize the data using df.descirbe() below are the inferences we did on
2. Delivery Person Ratings: Delivery person ratings have a mean of 4.63, suggesting generally high customer satisfaction. Ratings are tightly clustered around the higher end of the scale, with the 25th percentile being 4.5 and the 75th at 4.8.
3. Weather Data Temperature: The mean temperature is around 22.6°C, with a range from 6.77°C to 29.05°C. Humidity: The mean humidity level is 64.6%, with values ranging from 27% to 94%. This suggests some variability in weather conditions.
4. Precipitation: Most days have no precipitation (mean close to zero), but there are occasional higher values (max = 1.46).
5. Traffic Level: Traffic levels have a mean of 2.36 (out of 4), indicating moderate traffic conditions overall, but with significant variability (ranging from 0 to 4).
6. Delivery Distance: The average delivery distance is 14.28 km, with a wide range (from 1.55 km to 59.84 km). This suggests that some deliveries are considerably farther than others.
7. Order Type: The distribution of orders shows that the "Meal" and "Snack" categories are more common, with both having a mean close to 0.25, indicating a fairly even distribution of these order types. The "Drinks" category has a slightly lower mean (0.25), suggesting it's slightly less common than Meals and Snacks.
8. Type of Vehicle: Motorcycle is the most common mode of delivery (mean = 0.59), followed by scooter (mean = 0.33). Electric scooters are less common (mean = 0.08).
9. Weather Descriptions : "Clear sky" is the most frequent weather description (mean = 0.36), followed by "fog" and "haze" with values around 0.26 and 0.26, respectively. "Moderate rain," "scattered clouds," and "overcast clouds" are much less frequent. "Smoke" is also relatively rare, with a mean of 0.05.

Summary of the Data:

• The dataset provides useful details on various factors affecting delivery times and conditions, such as delivery person attributes, traffic levels, weather conditions, and order types.

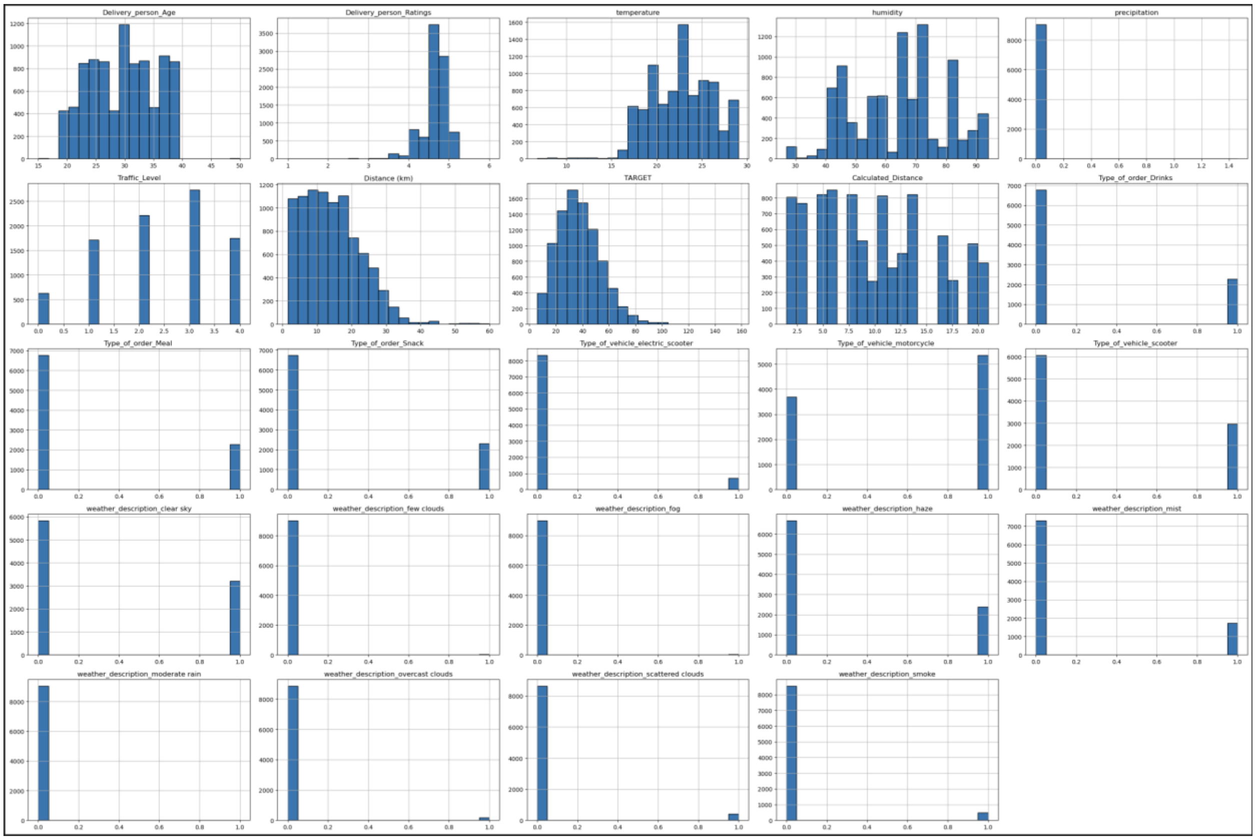
• The delivery persons are relatively young and highly rated. Orders are more commonly meals and snacks, while motorcycles are the most frequently used vehicles.

• The weather data indicates moderate temperature and humidity, with clear skies being the most common weather condition.

• The dataset's wide range of delivery distances and traffic levels highlights the variability in delivery conditions.

• This summary provides an overview of the data's structure and key variables, which could guide further analysis or model building.

4.2 Data Visualization
4.2.1 Univariate



Visualization and its key insights:

1. Delivery_person_Age: Most delivery persons are aged between 25 and 34, with a mean age of 29.5. The distribution is symmetric with a few outliers (ages between 11.5 and 47.5).

2. Delivery_person_Ratings: Ratings are concentrated between 4.5 and 5.0, with a peak at the higher end. The distribution is skewed left, indicating more higher ratings and some outliers.

3. Temperature: Temperatures mostly range from 19°C to 25°C, with a mean of 22.6°C. The distribution is roughly normal, with some outliers in the higher range.

4. Humidity: Humidity values range from 27% to 94%, with a peak around 65%. It has a slightly negative skew, indicating lower values are more frequent.

5. Precipitation: Most values are 0 (no precipitation), with a very small proportion showing higher values. The data is highly skewed to the right.

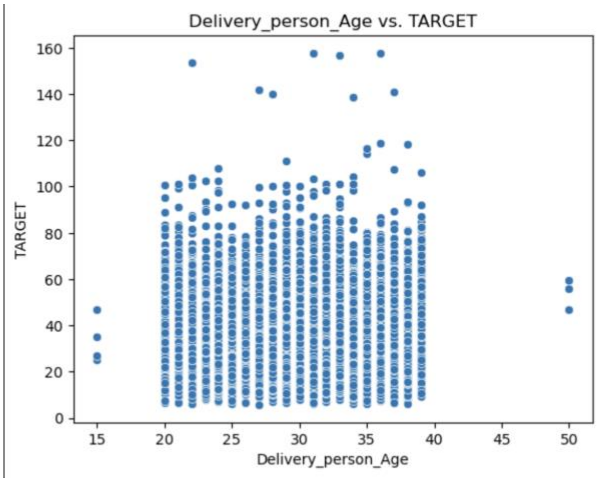
6. Traffic_Level: Traffic levels mostly range from 1 to 3, with a peak at 2 (moderate traffic). The distribution is slightly left skewed.

7. Distance (km): Distances mostly range from 1.5 km to 19.6 km, with a mean of 14.3 km. The distribution is slightly right skewed, with some long-distance deliveries as outliers.

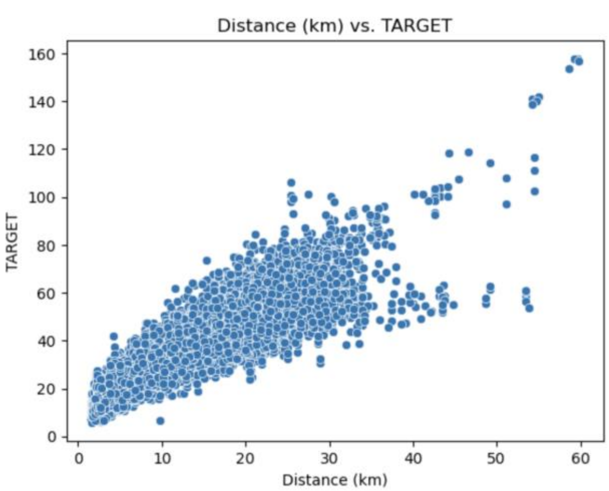
8. TARGET: Delivery times (TARGET) range from 5.8 to 157.75 minutes, with a mean of 37.65 minutes. It has a right skewed distribution, indicating most deliveries take less time but some outliers take much longer.

9. Calculated Distance: The calculated distances range from 1.47 km to 20.97 km, with a mean of 9.7 km. The distribution is normal with no significant outliers.

4.2.2 Bivariate Visualization and its key insights:

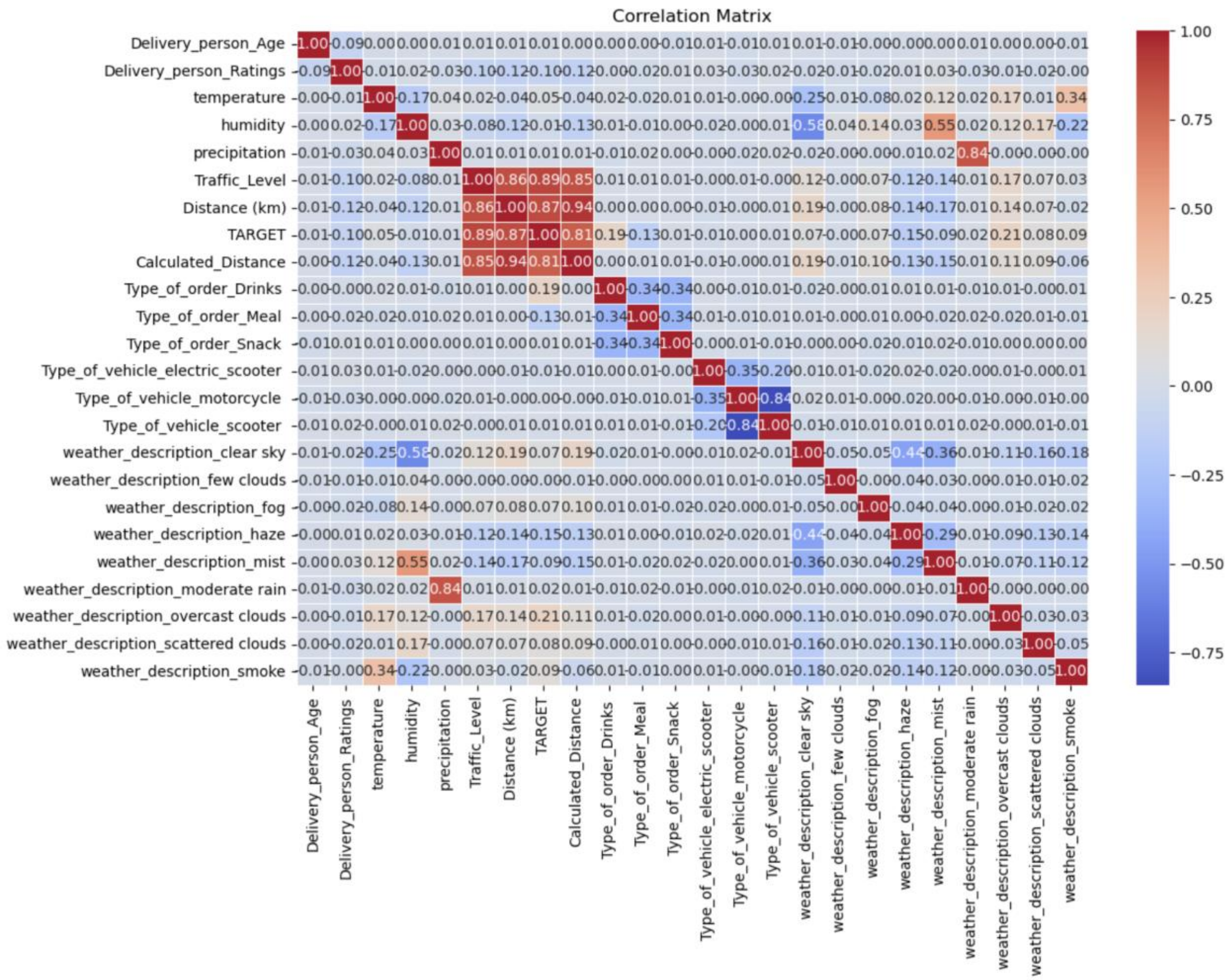


1. Delivery_person_Age: Weak correlation (0.01) with \`TARGET\`. Regression shows it's not a significant predictor (Pvalue = 0.583), with a negligible effect on \`TARGET\` (coefficient = 0.0168).



2. Distance (km): Strong correlation (0.87) with \`TARGET\`. Regression indicates a significant relationship (Pvalue = 0.000), with each additional kilometer increasing \`TARGET\` by 1.72 units.

4.2.3 Correlation Matrix (Heatmap):

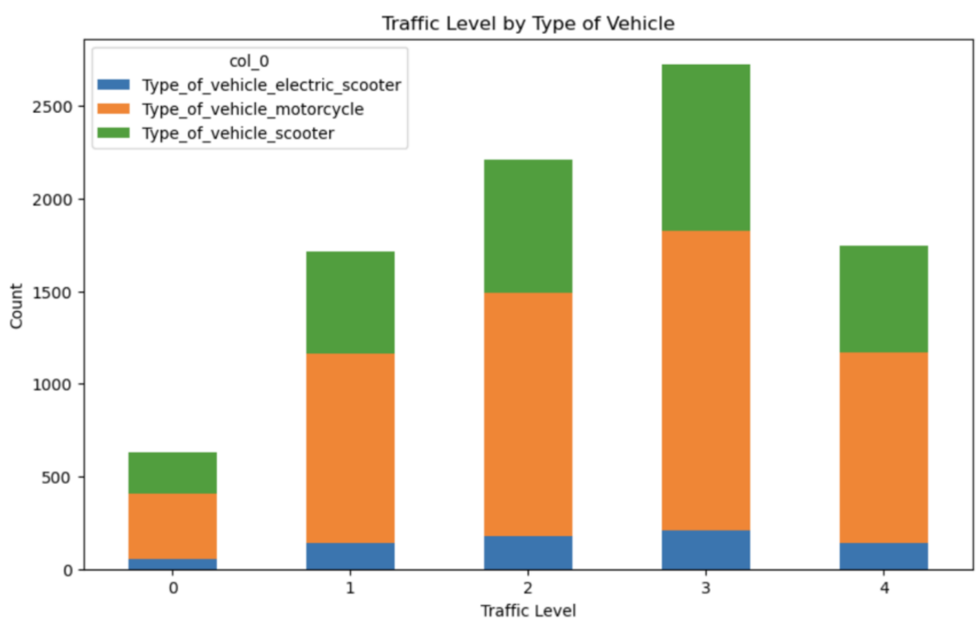


1. Delivery_person_Age: Slight negative correlation with delivery ratings, indicating older or younger delivery persons may have slightly lower ratings. Mild relationships with Traffic_Level and Distance, suggesting age influences delivery conditions to some extent.
2. Delivery_person_Ratings: Negative correlation with age, which could imply younger delivery persons tend to receive higher ratings. Slight correlations with Traffic_Level and Distance, suggesting performance may be affected by traffic conditions.
3. Temperature: Strong positive correlation with Humidity, and mild correlation with Traffic_Level. Negative correlation with clear skies and certain weather conditions, which may impact delivery performance.
4. Precipitation: Strong positive correlation with Moderate Rain, and links to other weather types, indicating it affects delivery performance.

5. Traffic_Level: Strongly correlated with Distance (km), suggesting longer distances usually experience more traffic. Affects TARGET, which may indicate longer trips or delays.
6. Distance (km): Correlates positively with Traffic_Level and Calculated Distance, impacting delivery time or efficiency.

7. TARGET (likely delivery time/efficiency): Correlates with Distance and Traffic_Level, showing that delivery times increase with longer distances and higher traffic. Weather conditions, especially fog and haze, also impact this outcome.
8. Calculated Distance: Positive correlation with Distance and Traffic_Level, suggesting longer and more complicated deliveries take more time.
9. Type_of_order (Drinks, Meal, Snack): Meals show a positive correlation with TARGET, potentially indicating more complex deliveries. Drinks and Snacks have weak correlations, suggesting they don't significantly impact delivery outcomes.
10. Type_of_vehicle (Electric scooter, Motorcycle, Scooter): Electric scooters and Motorcycles have negative correlations with each other, suggesting a preference for certain vehicles. Scooters show a positive correlation with Distance, indicating they are used for longer deliveries.
11. Weather Descriptions (Clear Sky, Few Clouds, Fog, Haze, Mist, etc.): Clear Sky and Few Clouds tend to have mild to negative correlations with TARGET and other features like Traffic_Level. Fog and Haze have strong negative correlations with TARGET, likely indicating delays due to poor visibility.Mist also correlates with longer delivery times or difficulty.

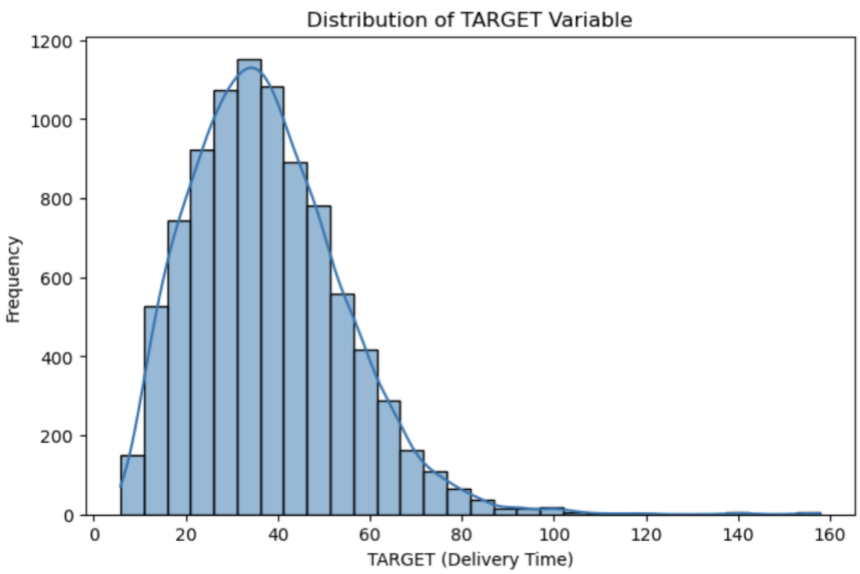
4.2.4 Categorical Visualizations:



5. Various methods, tools to analyze the data to develop data models.

5.1 Classification Analysis:

- Since TARGET is continuous, we eliminated classification methods (e.g., Logistic Regression, Naive Bayes, Decision Trees), which are best suited for categorical targets.
- TARGET is continuous float64, confirming that classification is not suitable.
- Mean: ~37.65 min, Median: ~35.98 min, Std Dev: ~16.55 min.
- Right skewed distribution with most deliveries between 2050 min




```
count    9035.000000
mean       37.653929
std        16.555688
min         5.800000
25%        25.566667
50%        35.983333
75%        47.633333
max       157.750000
Name: TARGET, dtype: float64
```

Graph Analysis:

- The histogram shows a peak around **3040 min**, suggesting common delivery times.
- The right skew confirms that longer deliveries are less frequent.
- Classification models (e.g., Logistic Regression, Naïve Bayes, Decision Trees) are designed for categorical targets (e.g., "Late" vs. "Ontime"), not for predicting continuous values.
- Our goal is to predict exact delivery times, which requires regression models instead of assigning labels.
- Regression models (Linear, SVR, KNN) are more appropriate for prediction.

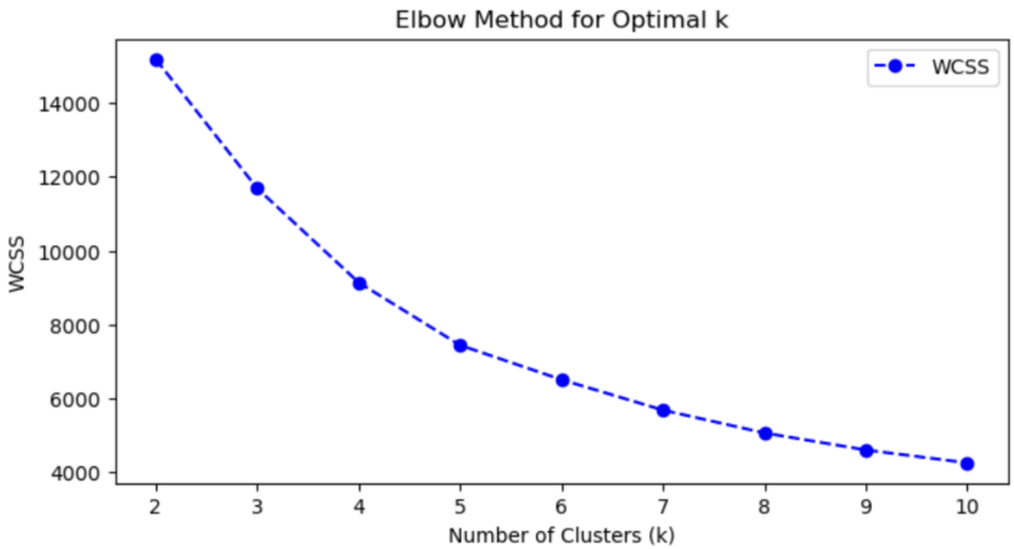
5.2 Clustering Analysis:

1. To identify patterns based on features like Temperature, Traffic_Level and Distance.
2. We want to use below models.

   i) K Means and

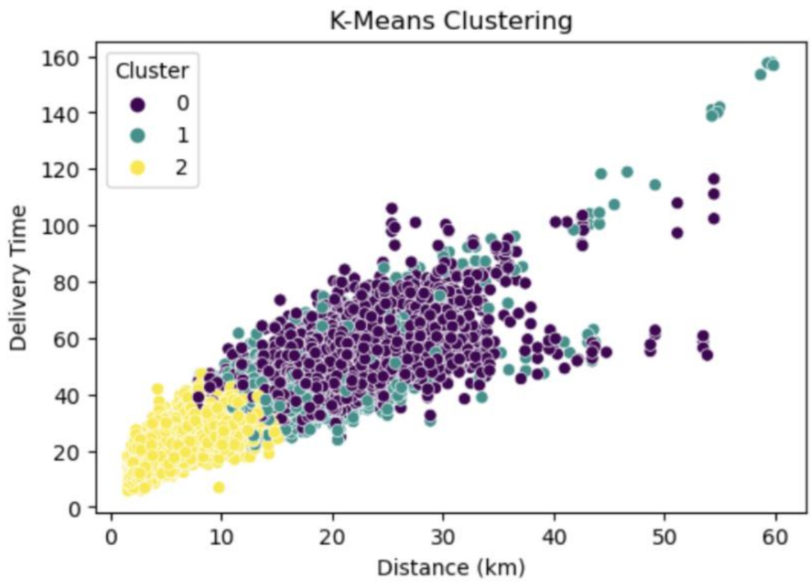
   ii) DBSCAN for grouping similar deliveries and detecting outliers or patterns.

3. Elbow Method (WCSS) : We calculated the within Cluster Sum of Squares (WCSS) for cluster values from k=2 to k=10 to identify the optimal number of clusters where the WCSS starts to level off, indicating a good balance between compactness and complexity.
4. Silhouette Score: For each `k`, we also computed the silhouette score to evaluate the quality of clustering higher scores indicate better-defined clusters. This metric complements the Elbow Method by measuring how well separated the clusters are.

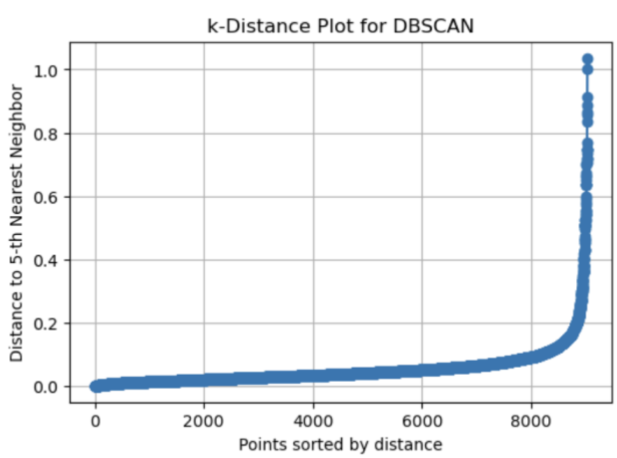


5. Elbow Method Plot: The WCSS curve shows a noticeable "elbow" at k=4, suggesting that 4 clusters provide a good tradeoff between compactness and simplicity.
6. Silhouette Score Plot: The silhouette score peaks at k=2, indicating that while 2 cluster are most well separated, additional clusters (like k=4) still offer reasonable structure with better segmentation.
7. KMeans clustering is applied using three clusters, selected based on the Elbow Method. The resulting cluster labels are stored in a new column to track which group each data point belongs to. DBSCAN is also used on the same dataset with `eps=0.5` and `min_samples=5` to form density-based clusters.
8. A scatter plot is generated to visualize the KMeans clusters, illustrating how delivery distance and delivery time are grouped across the identified clusters. The plot reveals noticeable patterns and separations between

the groups. Finally, the clustering quality is evaluated using the silhouette score, which returns a value of 0.3429, indicating a moderately strong cluster formation.



9. KMeans (k=3) gives a moderate silhouette score of 0.3429, indicating overlapping clusters and assuming spherical shapes.
10. DBSCAN, in contrast, handles irregular shapes and identifies outliers effectively without needing to predefine the number of clusters.
11. Given the noisy and realworld nature of the dataset, DBSCAN performs better for uncovering natural groupings in delivery patterns.
12. Let's Tune DBSCAN (ε and min_samples) using a kdistance plot.




```
K-Means Silhouette Score: 0.3429592230715477
DBSCAN Silhouette Score: 0.20209118615100838
```

13. KMeans performs is better overall with a Silhouette Score of 0.3429, indicating clearer and more well separated clusters than DBSCAN.
14. DBSCAN scores lower with a Silhouette Score of 0.2021, but it identifies outliers (cluster label `1`), which KMeans ignores.

15. KMeans forms 3 main clusters, assuming spherical shapes and requiring predefined `k`, which works well for structured, dense data.
16. DBSCAN forms 11 clusters including many small ones, making interpretation harder, but it handles irregular shapes and doesn't need predefined k.
17. KMeans is more suitable for well-defined segmentation when data is evenly distributed.
18. DBSCAN is more suitable when detecting anomalies or noise points is important.
19. DBSCAN tuning (via `ε` and `min_samples`, using kdistance plots) is essential to improve its clustering performance.

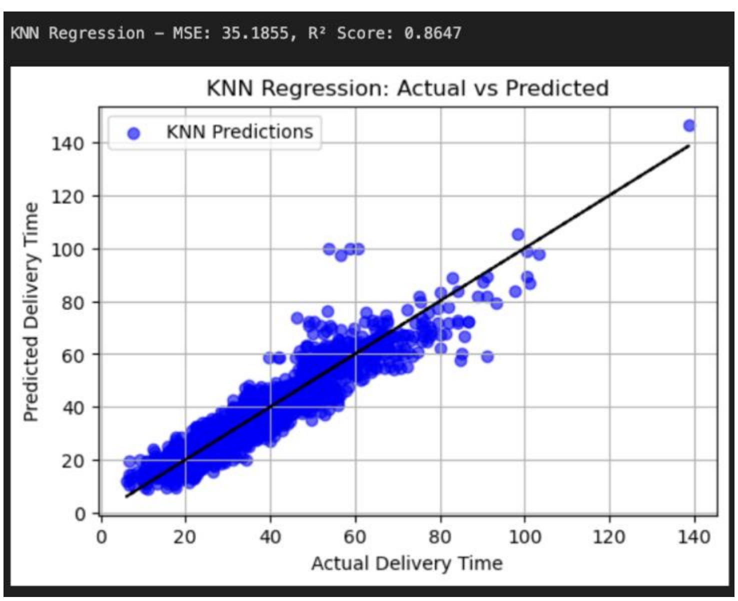
5.3 Forecast and Time Series analysis:

- For Predicting Target (Delivery Time).

1. Feature Selection & Data Splitting: We select `temperature`, `Traffic_Level`, and `Distance (km)` as features, with `TARGET` as the delivery time. We split the data into 80% training and 20% testing using `train_test_split`.
2. Feature Scaling: We applied Standard Scaler to standardize the features. We fit the scaler on the training data and transform both training and test sets to maintain consistency.

5.3.1 KNearest Neighbor Regression (KNN):

- `n_neighbors=5` to predict delivery time.
- Evaluate the model using Mean Squared Error (MSE) and R^2 Score.
- MSE = 35.1855 indicates the average squared error between predicted and actual values.
- $R^2 = 0.8647$ shows the model explains 86.47% of the variance in delivery time, which reflects good predictive power.

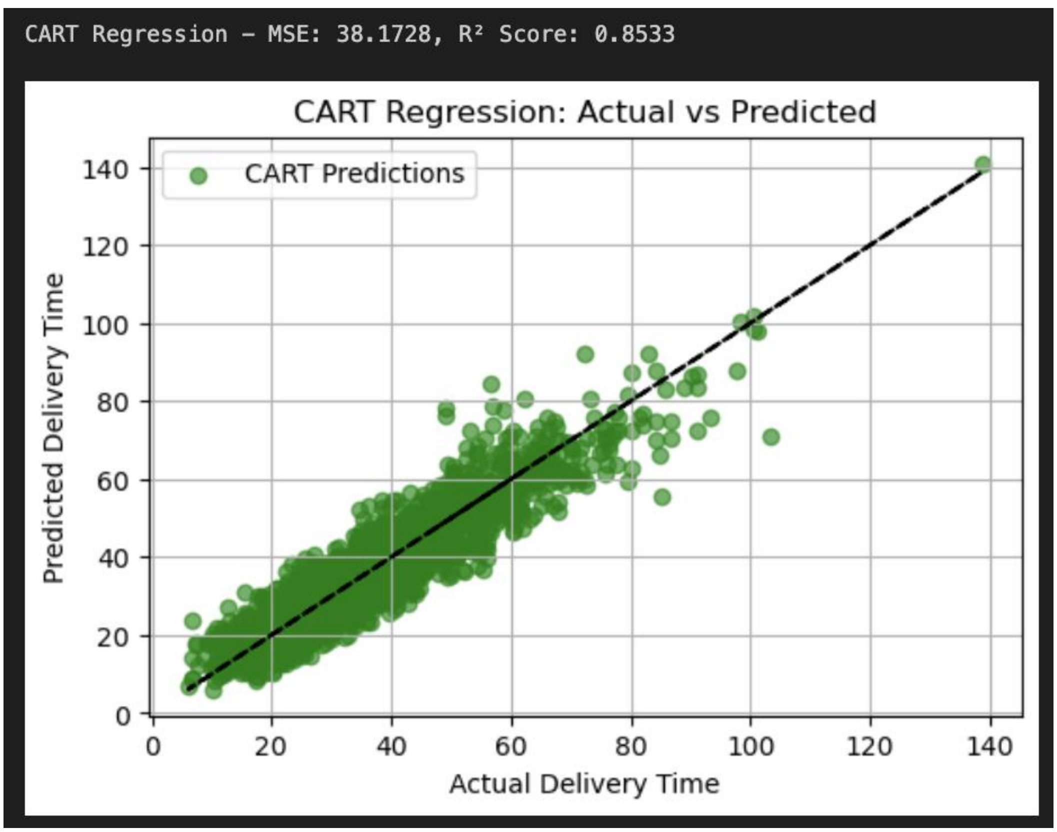


Graph Interpretation (Actual vs Predicted)

- The scatter plot shows actual delivery time (xaxis) vs. predicted delivery time (yaxis).
- Blue dots represent the KNN predictions.
- The black dashed line represents a perfect fit (ideal prediction line).
- Most points lie close to the line, indicating that our model performs well, though a few outliers are present at higher delivery times.

5.3.2 Classification and Regression Trees (CART):

- We used a Decision Tree Regressor with random_state=42 to predict delivery time.
  We evaluate the model using Mean Squared Error (MSE) and R^2 Score.
- MSE = 38.1728 indicates the average squared difference between predicted and actual delivery times.
- $R^2 = 0.8533$ shows the model explains 85.33% of the variance in delivery time, which also reflects strong predictive power, slightly below KNN.



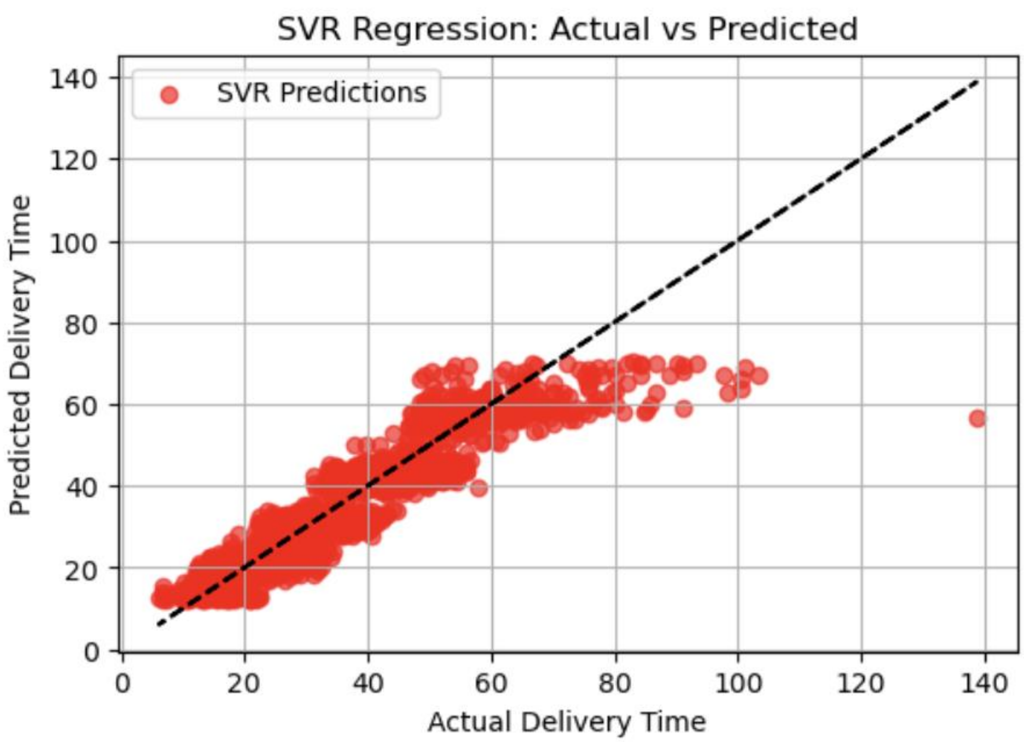
Graph Interpretation (Actual vs Predicted)

- The scatter plot shows **actual delivery time** (xaxis) vs. **predicted delivery time** (y-axis).
- **Green dots** represent the CART model predictions.
- The **black dashed line** represents the ideal perfect prediction line.

- Most predictions align closely with this line, but the spread is slightly wider compared to KNN, suggesting slightly lower accuracy in some cases.

5.3.3 Support Vector Regression (SVR):

- We used SVR with an RBF kernel to predict delivery time, training it on scaled data.
  We evaluate the model using Mean Squared Error (MSE) and R^2 Score.
- MSE = 39.9387 indicates the average squared error between predicted and actual delivery times.
- $R^2 = 0.8465$ shows the model explains 84.65% of the variance, slightly less than KNN and CART but still strong.

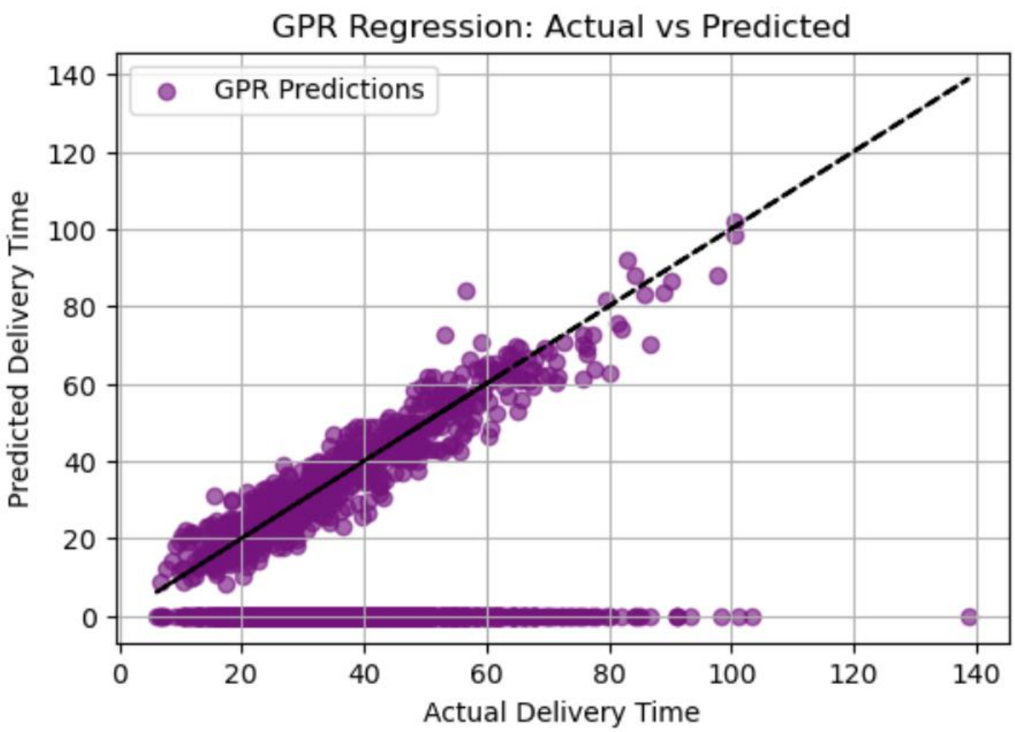


Graph Interpretation (Actual vs Predicted)

- The scatter plot shows actual delivery time (xaxis) vs. predicted delivery time (yaxis).
- Red dots represent SVR predictions.
- The black dashed line represents perfect predictions.
- Most points are near the line, but compared to KNN and CART, SVR predictions show a bit more spread, especially at higher delivery times.

5.3.4 Gaussian Process Regression (GPR):

- GPR with an RBF kernel is used to predict delivery time, trained on **scaled data**. We assess the model using **Mean Squared Error (MSE)** and **R^2 Score**.
- **MSE = 1069.0182** indicates a high average squared error, suggesting poor predictive accuracy.
- **R^2 = 3.1094** implies the model performs significantly worse than a simple meanbased prediction.



Graph Interpretation (Actual vs Predicted)

- The plot compares actual delivery time (xaxis) vs. predicted delivery time (yaxis).
- **Purple dots** show the GPR predictions.
- The **black dashed line** is the perfect prediction line.
- Predictions are scattered and often far from the ideal line, especially for higher delivery times, indicating weak model performance.

	Model	MAE	MSE	RMSE	R² Score
0	KNN Regression	4.388297	35.185505	5.931737	0.864743
1	CART Regression	4.749953	38.172757	6.178411	0.853260
2	SVR	14.969733	325.813190	18.050296	-0.252462
3	GPR	37.500507	1666.426312	40.821885	-5.405925

KNN Regression performed the best

- Lowest MAE (4.39) → Smallest absolute errors.
- Lowest MSE (35.19) & RMSE (5.93)→ Smallest squared errors.
- Highest R^2 Score (0.86) → Explains 86% of variance, meaning strong predictive power.

CART Regression is slightly worse than KNN

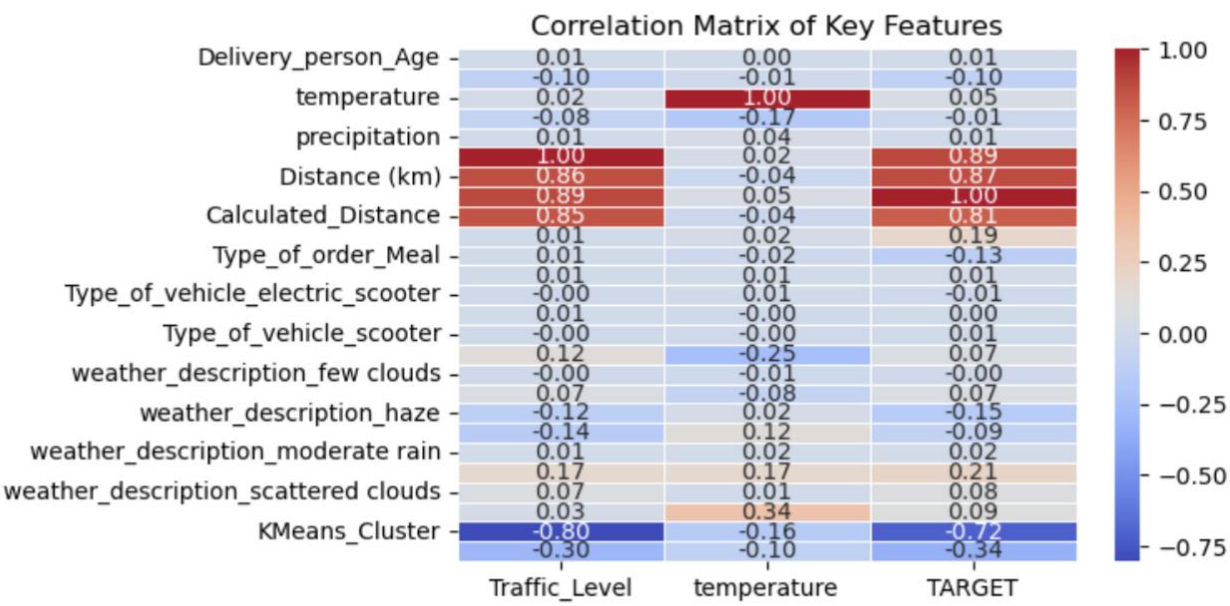
- RMSE is slightly higher (6.18), but still a decent model.

SVR and GPR performed very poorly

- SVR has a negative R^2 Score (0.25) → Means it performs worse than a simple mean prediction.
- GPR is the worst (5.41 R^2, 1666.43 MSE) → Huge prediction errors, likely due to overfitting or inappropriate kernel selection.
- KNN Regression is the best model for predicting TARGET. SVR and GPR should be eliminated or optimized.

5.4 Association Analysis Methods (Market Basket Analysis)

- To find relationships between features:
- We want to use: Correlation Matrix to analyze how features like Traffic_Level and Temperature affect TARGET.
- We want to eliminate: NLP and Apriori, as they are not suitable for our dataset



Inference from the Correlation Matrix:

1. Traffic_Level vs. TARGET (0.12)

- A weak negative correlation suggests that as `Traffic_Level` increases, `TARGET` (Delivery Time) slightly decreases.
- This is unexpected—higher traffic is usually expected to increase delivery time. You might want to check data consistency.

2. Temperature vs. TARGET (0.10)

- A weak positive correlation means that higher `Temperature` might slightly increase `TARGET`, but the effect is minimal.
- This suggests weather conditions don't strongly affect delivery time in this dataset.

3. Distance (km) vs. TARGET (0.87)

- A strong positive correlation shows that longer distances significantly increase delivery time, which is expected.

4. KMeans_Cluster vs. Traffic_Level (0.80)

- Very strong negative correlation implies that the clustering algorithm has grouped data where higher traffic levels are in one cluster and lower in another.
- This indicates that `KMeans_Cluster` effectively separates traffic levels.

5. KMeans_Cluster vs. TARGET (0.34)

- A moderate negative correlation suggests that deliveries in certain clusters (possibly high traffic areas) tend to have lower `TARGET` values.
- This contradicts intuition, as one would expect higher traffic to delay deliveries.

Final Thoughts:

- The weak correlations between `Traffic_Level`, `Temperature`, and `TARGET` suggest other features (like `Distance (km)`) play a more significant role in determining delivery time.
- There might be underlying patterns in `KMeans_Cluster` that require further investigation.
- You might want to check for data inconsistencies or other influential factors.

6. Conclusion

- The objective of this project was to predict delivery time using various machine learning regression models. We implemented and evaluated four models: KNearest Neighbors (KNN), Decision Tree Regressor (CART), Support Vector Regressor (SVR), and Gaussian Process Regressor (GPR).
- Each model was assessed using two key metrics: Mean Squared Error (MSE) to measure prediction accuracy, and R^2 Score to evaluate how well the model explains the variance in delivery time.
- Among all models, the KNN regressor performed the best. It had the lowest MSE (35.18), indicating minimal prediction error, and the highest R^2 score (0.8647), showing it explained about 86.5% of the variance in delivery time.
- The CART and SVR models also gave good results, with slightly higher MSEs and slightly lower R^2 scores compared to KNN. They are still viable options but did not outperform KNN in this case.
- The GPR model performed poorly, with an extremely high MSE (1069.02) and a negative R^2 score (–3.11). This suggests it failed to capture the relationship in the data and is not suitable for this prediction task.
- In summary, KNN was the most effective and reliable model for delivery time prediction in this dataset. The comparison emphasizes the importance of testing multiple models and metrics to make an informed choice.