

Data Analysis & Machine Learning

By Shrouk Hassan



DATA ANALYSIS

Food Delivery Time Prediction and Analysis

- **Potential issue: Improving delivery time.**
- **The target column for forecasting is: Delivery_Time_min** (delivery time).
- **The goal of forecasting is:** to understand how factors (such as weather, distance, vehicle type) affect delivery time.

Data Description

1 Order_ID Unique identifier for each order.	2 Distance_km The delivery distance in kilometers.	3 Weather Weather conditions during the delivery, including Clear, Rainy, Snowy, Foggy,	4 Traffic_Level Traffic conditions categorized as Low, Medium, or High.	5 Time_of_Day The time when the delivery took place, categorized as Morning, Afternoon,
6 Vehicle_Type Type of vehicle used for delivery, including Bike, Scooter, and Car.	7 Preparation_Time_min The time required to prepare the order, measured in minutes.	8 Courier_Experience_yrs Experience of the courier in years.	Delivery_Time_min: The total delivery time in minutes (target variable).	

Extract ideas:

- How weather affects delivery time.
- Traffic impact on delivery times.
- Effect of Courier experience on performance.



Importing Necessary Modules and Data Acquisition

```
▶ # importing necessary modules
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

[25] df=pd.read_csv('/content/Food_Delivery_Times (1).csv')
df.head()
```

	Order_ID	Distance_km	Weather	Traffic_Level	Time_of_Day	Vehicle_Type	Preparation_Time_min	Courier_Experience_yrs	Delivery_Time_min
0	522	7.93	Windy	Low	Afternoon	Scooter	12	1.0	43
1	738	16.42	Clear	Medium	Evening	Bike	20	2.0	84
2	741	9.52	Foggy	Low	Night	Scooter	28	1.0	59
3	661	7.44	Rainy	Medium	Afternoon	Scooter	5	1.0	37
4	412	19.03	Clear	Low	Morning	Bike	16	5.0	68

Understanding, Describing The Data and Checking for Null Values

The dataset contains 9 columns and 1000 entries, with some columns missing values such as:

- Weather: Weather conditions (30 missing values).
- Traffic_Level: Traffic conditions (30 missing values).
- Time_of_Day: Time of the day for delivery (30 missing values).
- Courier_Experience_yrs: Courier's experience in years (30 missing values).

```
[26] df.shape
→ (1000, 9)

[27] df.info()
→ <class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 9 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Order_ID         1000 non-null   int64  
 1   Distance_km     1000 non-null   float64
 2   Weather          970 non-null   object  
 3   Traffic_Level    970 non-null   object  
 4   Time_of_Day      970 non-null   object  
 5   Vehicle_Type     1000 non-null   object  
 6   Preparation_Time_min 1000 non-null   int64  
 7   Courier_Experience_yrs 970 non-null   float64
 8   Delivery_Time_min 1000 non-null   int64  
dtypes: float64(2), int64(3), object(4)
memory usage: 70.4+ KB
```

```
[29] df.isna().sum()
→ 0
Order_ID           0
Distance_km        0
Weather            30
Traffic_Level      30
Time_of_Day         30
Vehicle_Type        0
Preparation_Time_min 0
Courier_Experience_yrs 30
Delivery_Time_min   0
dtype: int64
```

Data Pre-Processing

Data Cleaning

- Handle Missing Values by Fill with the mode at Weather,Traffic_Level,Time_of_Day (Categorical).
 - Handle Missing Values by Fill with the mean at Courier_Experience_yrs (Numeric).

```
[30] df['Weather'] = df['Weather'].fillna(df['Weather'].mode()[0])
    df['Traffic_Level'] = df['Traffic_Level'].fillna(df['Traffic_Level'].mode()[0])
    df['Time_of_Day'] = df['Time_of_Day'].fillna(df['Time_of_Day'].mode()[0])

[31] x= df['Courier_Experience_yrs'].mean()
    df['Courier_Experience_yrs'].fillna(x,inplace=True)
```

check again null value and duplicate values

- no null values and no duplicate values.

[32] df.isnull().sum()	
Order_ID	0
Distance_km	0
Weather	0
Traffic_Level	0
Time_of_Day	0
Vehicle_Type	0
Preparation_Time_min	0
Courier_Experience_yrs	0
Delivery_Time_min	0

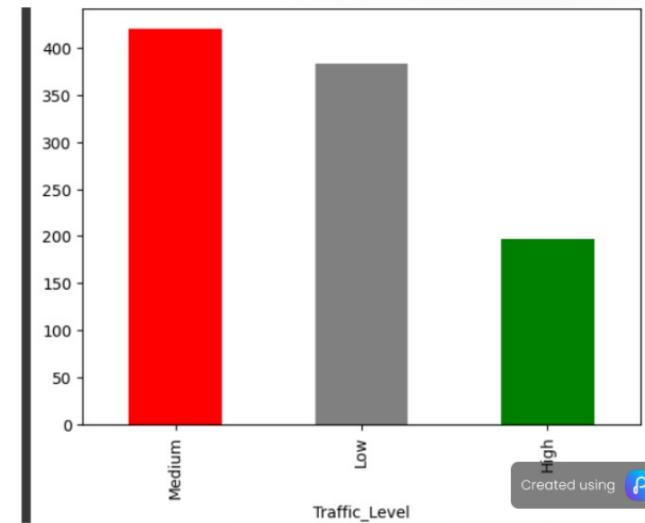
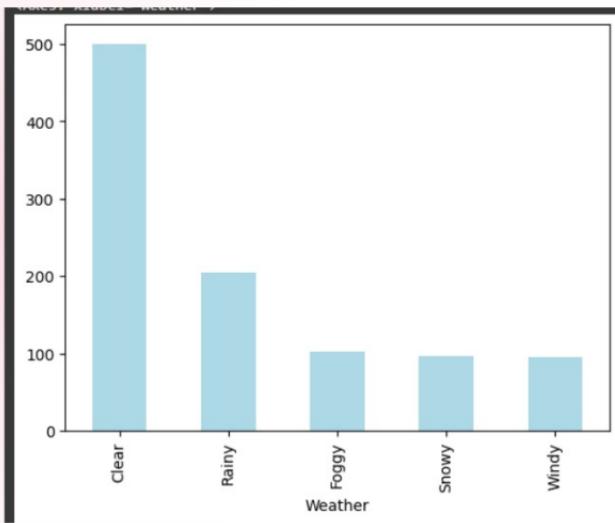
dtype: int64

[33] df.duplicated().sum()	
	0

Exploratory data analysis (EDA)

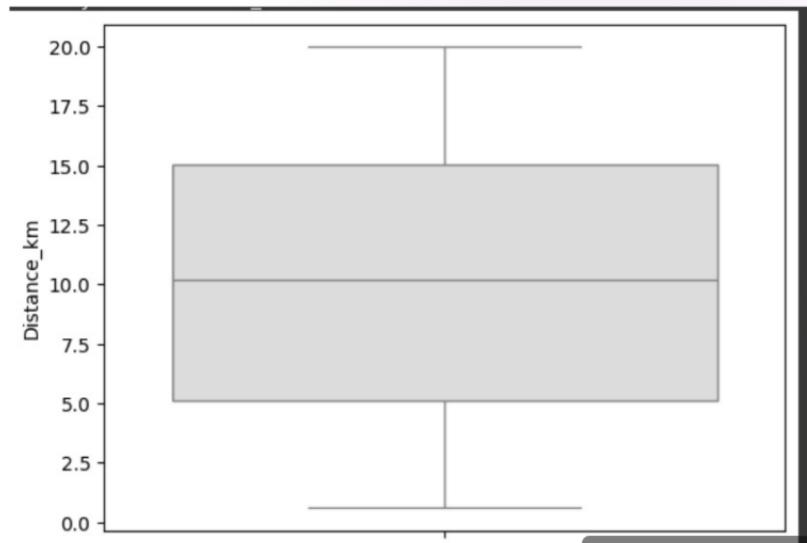
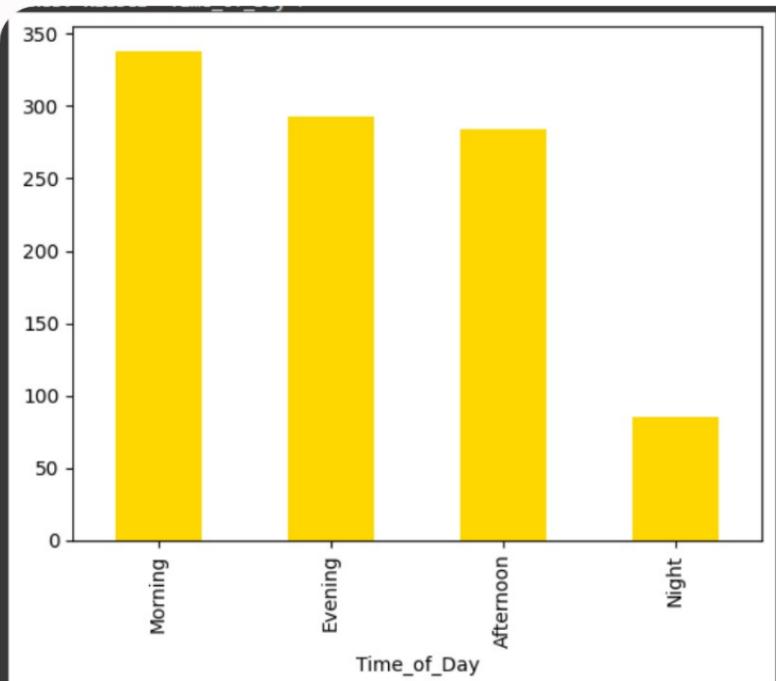
(univariate)

- Most of the cases are clear weather.(Weather)
- Data is distributed between low and medium congestion, with low congestion being the most common.(Traffic_Level)



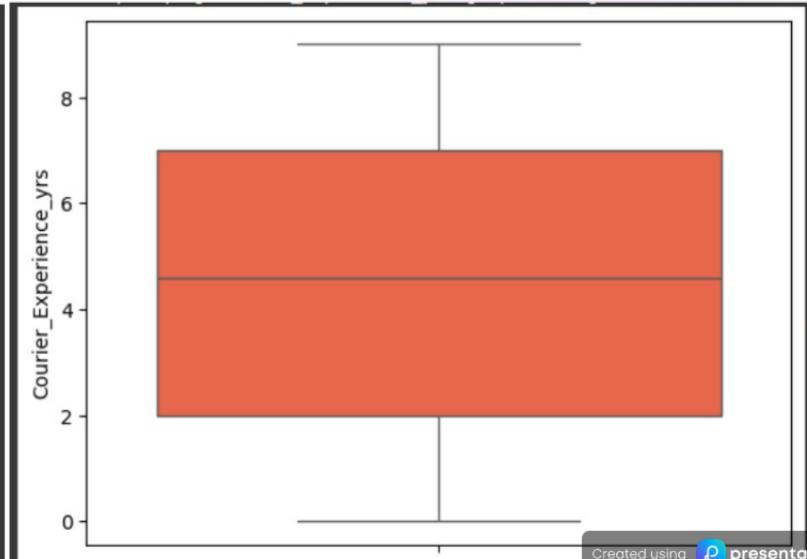
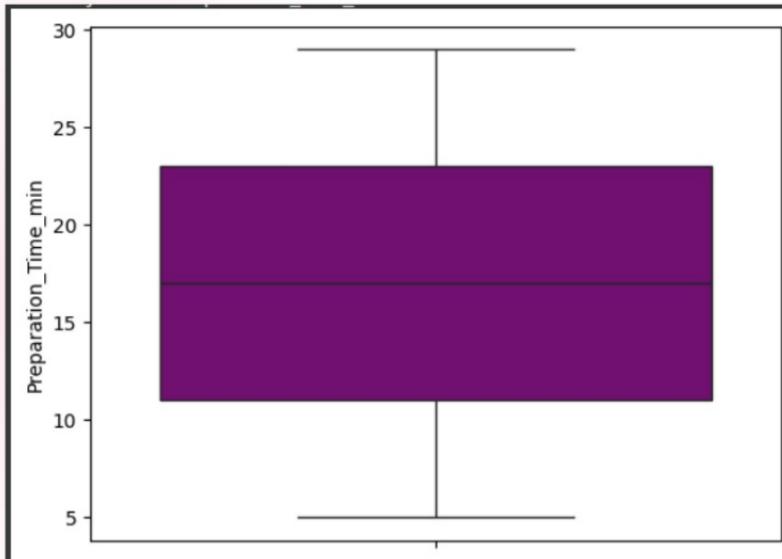
EDA (univariant)

- Orders are distributed almost evenly over the different periods except for the night.
- The distribution shows most of the requests within a range of 5-15 km.



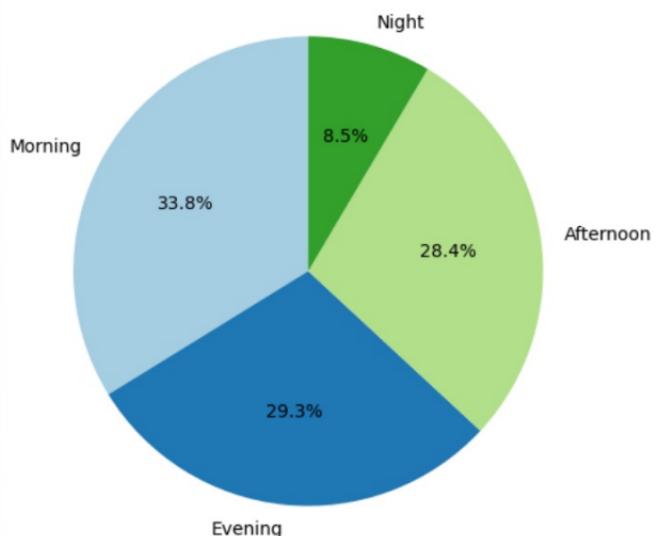
EDA (univariant)

- Preparation time varies greatly, with many concentrating in under 20 minutes.
 - Most drivers have less than 5 years experience.

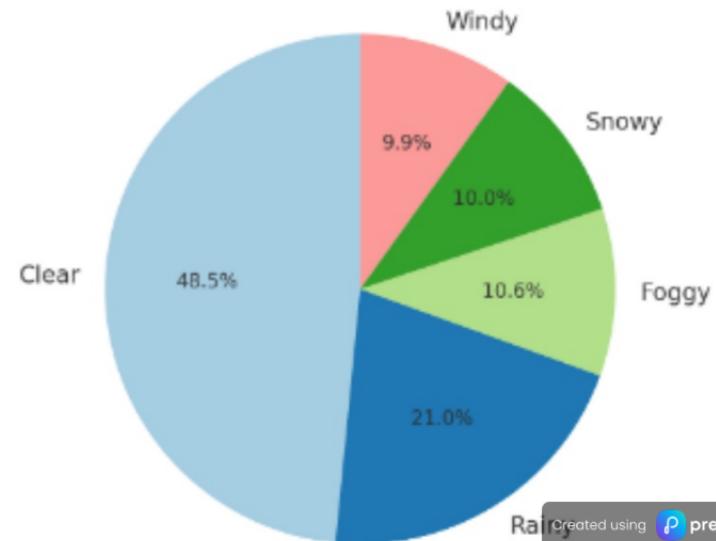


EDA (univariant)

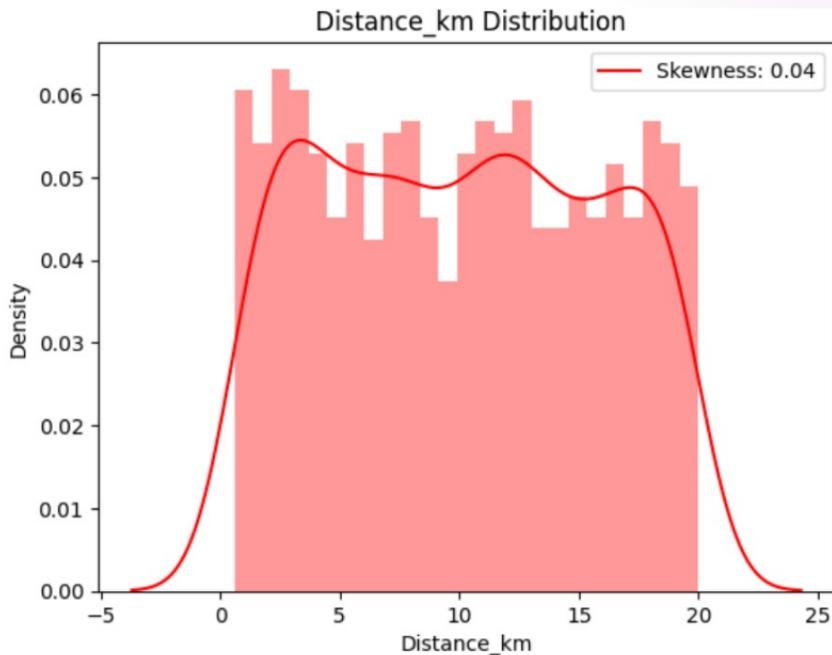
Time_of_Day Distribution



Weather Distribution

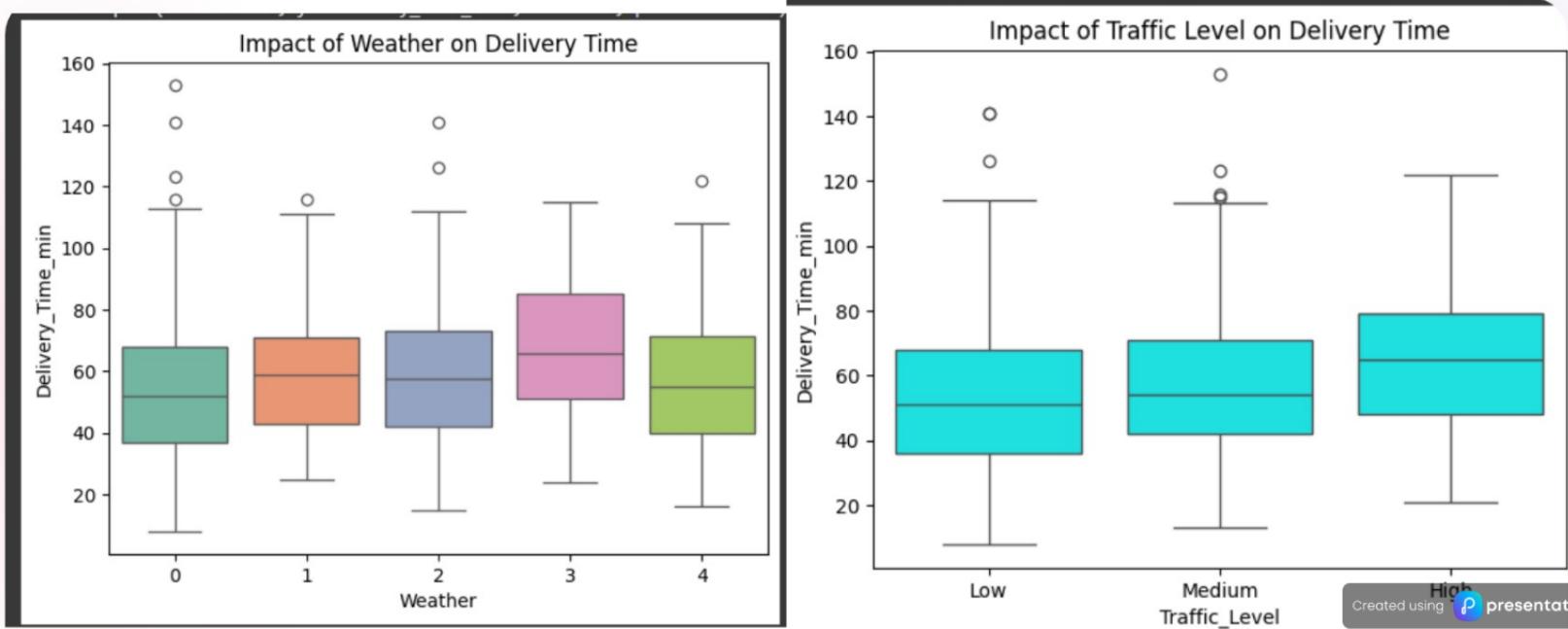


EDA (univariant)



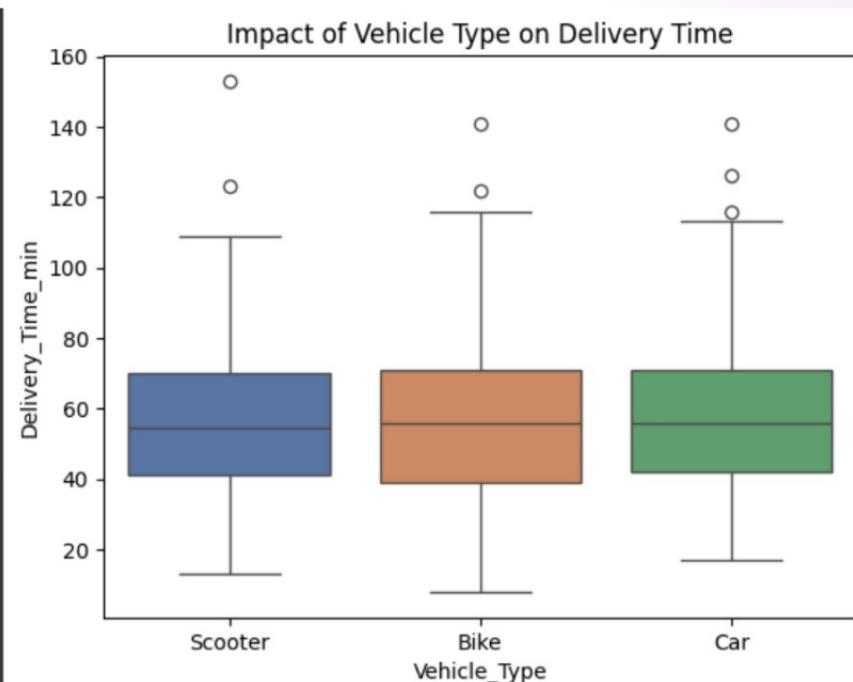
Bivariant

- Weather has an inverse relationship with delivery time.
- When traffic level is high it affects the delivery time.

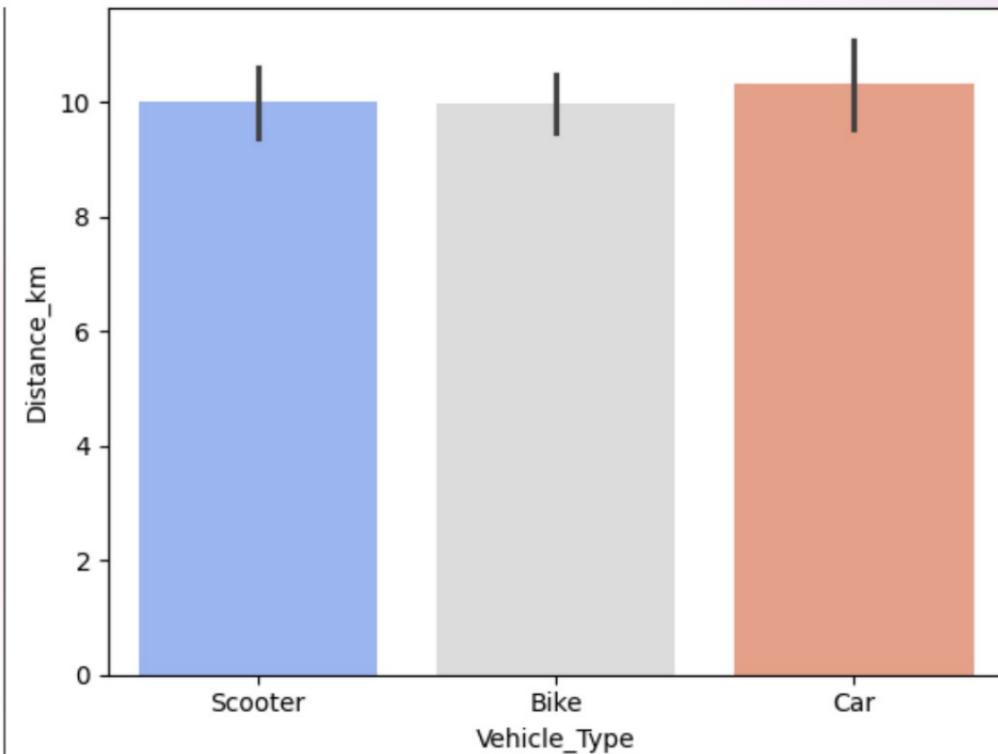


Bivariant

- Scooter is faster in delivery.



Bivariant



MACHINE LEARNING



machine learning

- Recover libraries
- Converting categorical variables to numbers
- Identifying independent variables and target variable

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
df['Weather'] = df['Weather'].astype('category').cat.codes
df['Traffic_Level'] = df['Traffic_Level'].astype('category').cat.codes
df['Vehicle_Type'] = df['Vehicle_Type'].astype('category').cat.codes
x = df[['Weather', 'Traffic_Level', 'Vehicle_Type', 'Distance_km']] #
y = df['Delivery_Time_min'] # المتغير المستهدف
```

machine learning

- Splitting the data into a training and test set
 - Predicting values for the test set
 - Displaying results
- Creating a linear regression model
 - Evaluating performance
 - Displaying coefficients
- Training the model

```
# تقسيم البيانات إلى مجموعة تدريب واختبار
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# إنشاء نموذج الانحدار الخطي
model = LinearRegression()

# تدريب النموذج
model.fit(X_train, y_train)

# توقع القيم لمجموعة الاختبار
y_pred = model.predict(X_test)

# تقييم الأداء
mse = mean_squared_error(y_test, y_pred) # متوسط مربع الخطأ
r2 = r2_score(y_test, y_pred) # معامل التحديد

# عرض النتائج
print("Mean Squared Error (MSE):", mse)
print("R-squared (R2):", r2)

# عرض المعاملات
coefficients = pd.DataFrame({'Feature': X.columns, 'Coefficient': model.coef_})
print(coefficients)
```

machine learning

- The model is relatively good, explaining 66.44% of the variance in delivery times.
 - Distance_km is the variable that most directly affects delivery time.
 - Weather has a medium effect.



Mean Squared Error (MSE): 150.4112197528693
R-squared (R2): 0.6644306832428608

	Feature	Coefficient
0	Weather	1.276292
1	Traffic_Level	-1.633788
2	Vehicle_Type	-0.550671
3	Distance_km	2.992732



The Ends

Dataset From Kaggle