```
Start coding or generate with AI.
```

Step 1: Import Libraries

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np
```

Step 2: Upload the Dataset

```
from google.colab import files

# Upload your CSV file manually
uploaded = files.upload()

# Load the dataset
df = pd.read_csv(list(uploaded.keys())[0])
df.head()
```

Choose files archive (2).zip
archive (2).zip(application/x-zip-compressed) - 11861 bytes, last modified: 21/09/2025 - 100% done

```
Saving archive (2).zip to archive (2).zip
```

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

Next steps: Generate code with df New interactive sheet

Step 3: Explore the Dataset

```
# Check dataset info and missing values
print(df.info())
print(df.isnull().sum())

# Drop missing values (simple method)
df = df.dropna()
```

```
<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
None
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
```

Step 4: Prepare Features and Target

```
# Target column
y = df['Delivery_Time_min']
X = df.drop('Delivery_Time_min', axis=1)

# Encode categorical variables if any
X = pd.get_dummies(X, drop_first=True)

# Split dataset into training and testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
print(df.columns)
```

```
Index(['Order_ID', 'Distance_km', 'Weather', 'Traffic_Level', 'Time_of_Day',
       'Vehicle_Type', 'Preparation_Time_min', 'Courier_Experience_yrs',
       'Delivery_Time_min'],
      dtype='object')
```

Step 5: Train Linear Regression Model

```
model = LinearRegression()
model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

Step 6: Make Predictions

```
y_pred = model.predict(X_test)
```

Step 7: Evaluate Model

```
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"R2 Score: {r2}")
```

```
RMSE: 8.276334838498395
R2 Score: 0.8324092509049891
```

UPLOADIG DATA SET

Step 1: Upload the CSV File

```
from google.colab import files
import pandas as pd

# This will open a file browser to select your CSV
uploaded = files.upload()
```

Choose files archive (2).zip
archive (2).zip(application/x-zip-compressed) - 11861 bytes, last modified: 21/09/2025 - 100% done

```
Saving archive (2).zip to archive (2) (1).zip
```

Step 2: Load the Uploaded CSV

```
# Load the uploaded CSV into a DataFrame
df = pd.read_csv(list(uploaded.keys())[0])

# Preview the dataset
df.head()
```

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

Next steps: Generate code with df New interactive sheet

Step 3: Continue With Your Code

```
# Example: check column names
print(df.columns)

# Drop missing values (optional)
df = df.dropna()

# Target column (replace with the actual column name in your CSV)
y = df['Delivery_Time_min']
X = df.drop('Delivery_Time_min', axis=1)

# Encode categorical variables
X = pd.get_dummies(X, drop_first=True)

# Split dataset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
Index(['Order_ID', 'Distance_km', 'Weather', 'Traffic_Level', 'Time_of_Day',
       'Vehicle_Type', 'Preparation_Time_min', 'Courier_Experience_yrs',
       'Delivery_Time_min'],
      dtype='object')
```

Make Predictions

```
y_pred = model.predict(X_test)
```

Evaluate Model

```
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print(f"RMSE: {rmse}")
print(f"R2 Score: {r2}")
```

```
RMSE: 8.276334838498395
R2 Score: 0.8324092509049891
```

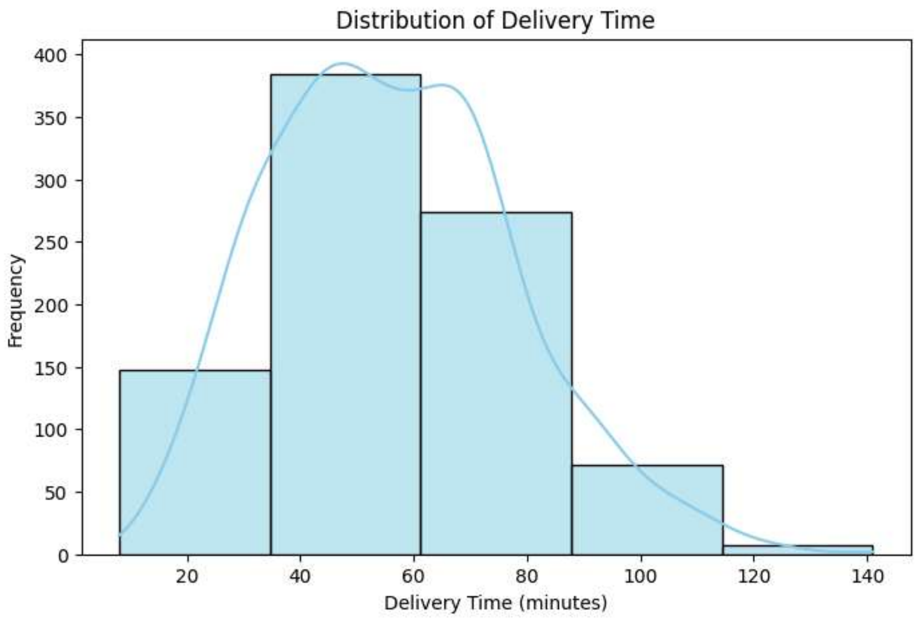
** Visualizations**

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

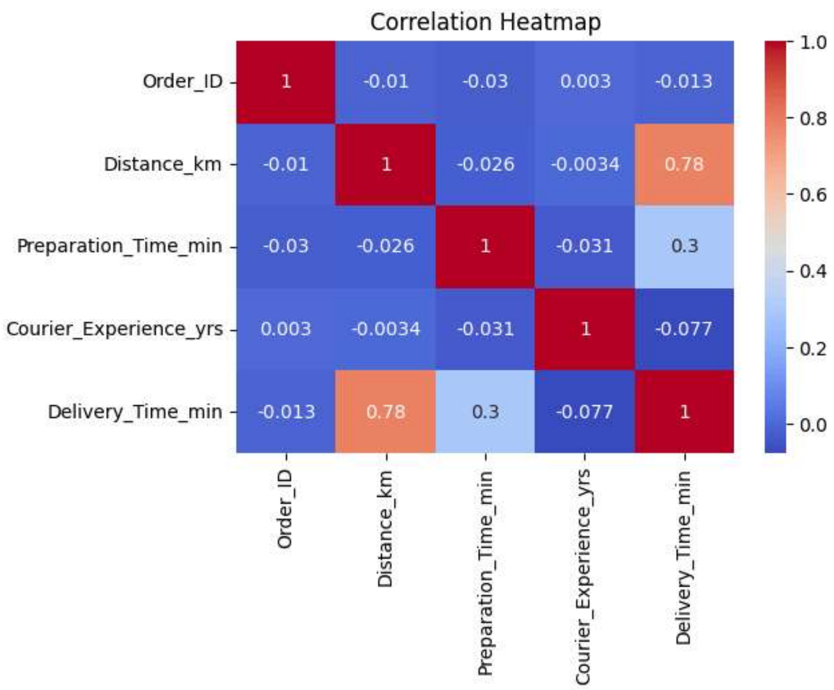
1️⃣ Distribution of Delivery Time

```
plt.figure(figsize=(8,5))
sns.histplot(df['Delivery_Time_min'], bins=5, kde=True, color='skyblue')
plt.title('Distribution of Delivery Time')
plt.xlabel('Delivery Time (minutes)')
plt.ylabel('Frequency')
plt.show()
```



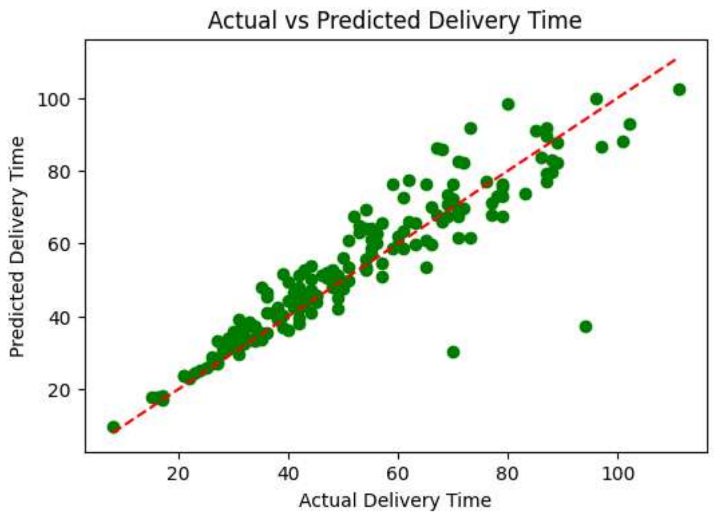
2️⃣ Correlation Heatmap

```
import matplotlib.pyplot as plt
import seaborn as sns

# Make sure you only use numeric columns for correlation
numeric_df = df.select_dtypes(include=['int64', 'float64'])

plt.figure(figsize=(6,4))  # width=6, height=4
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



Actual vs Predicted Delivery Time

```
plt.figure(figsize=(6,4))
plt.scatter(y_test, y_pred, color='green')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
plt.xlabel('Actual Delivery Time')
plt.ylabel('Predicted Delivery Time')
plt.title('Actual vs Predicted Delivery Time')
plt.show()
```



Predict a New Order

```
['Courier_Experience_yrs', 'Distance_km', 'Order_ID', 'Preparation_Time_min', 'Time_of_Day_Evening', ...]
```

```
['Courier_Experience_yrs',
 'Distance_km',
 'Order_ID',
 'Preparation_Time_min',
 'Time_of_Day_Evening',
 Ellipsis]
```

```
['Distance', 'Traffic', 'Weather_Rainy']
```

```
['Distance', 'Traffic', 'Weather_Rainy']
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files
```

```
# Upload your CSV file
uploaded = files.upload()

# Load the dataset
df = pd.read_csv(list(uploaded.keys())[0])
df.head()
```

Choose files archive (2).zip

archive (2).zip(application/x-zip-compressed) - 11861 bytes, last modified: 21/09/2025 - 100% done

```
Saving archive (2).zip to archive (2) (3).zip
```

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

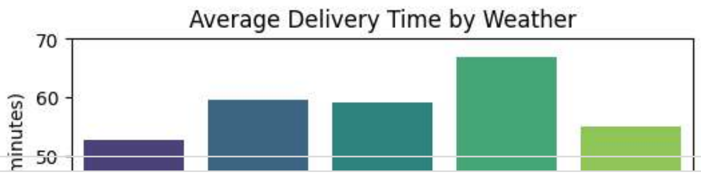
Next steps: Generate code with df New interactive sheet

```
# Drop missing values if any
df = df.dropna()

# Check column names
print(df.columns)

# Make sure target column is 'Delivery_Time_min'
```

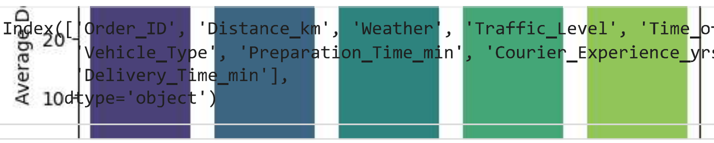
```
Index(['Order_ID', 'Distance_km', 'Weather', 'Traffic_Level', 'Time_of_Day',
       'Vehicle_Type', 'Preparation_Time_min', 'Courier_Experience_yrs',
       'Delivery_Time_min'],
      dtype='object')
```

```
# Average delivery time by Weather
avg_delivery = df.groupby('Weather')['Delivery_Time_min'].mean().reset_index()

plt.figure(figsize=(6,4))
sns.barplot(x='Weather', y='Delivery_Time_min', data=avg_delivery, palette='viridis')
plt.title('Average Delivery Time by Weather')
plt.xlabel('Weather')
plt.ylabel('Average Delivery Time (minutes)')
plt.show()
```

```
/tmp/ipython-input-605935706.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  sns.barplot(x='Weather', y='Delivery_Time_min', data=avg_delivery, palette='viridis')
```



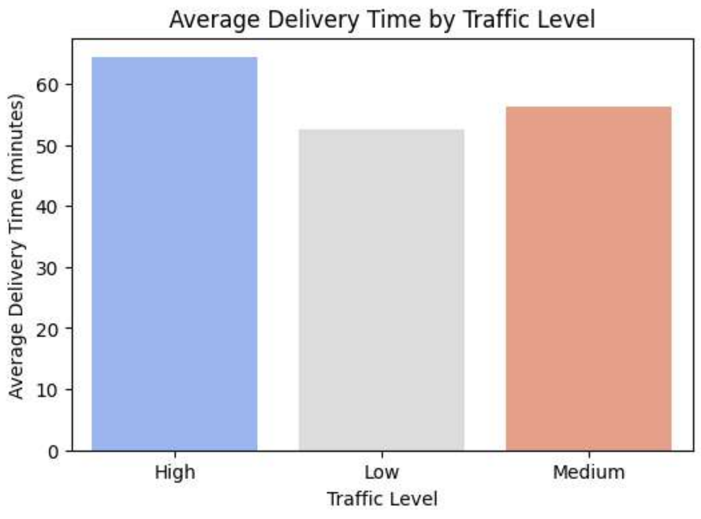
```
print(df.columns)
```



```
Index(['Order_ID', 'Distance_km', 'Weather', 'Traffic_Level', 'Time_of_Day',
       'Vehicle_Type', 'Preparation_Time_min', 'Courier_Experience_yrs',
       'Delivery_Time_min'],
      dtype='object')
```

```
# Average delivery time by Traffic
avg_delivery_traffic = df.groupby('Traffic_Level')['Delivery_Time_min'].mean().reset_index()

plt.figure(figsize=(6,4))
sns.barplot(x='Traffic_Level', y='Delivery_Time_min', data=avg_delivery_traffic, palette='coolwarm')
plt.title('Average Delivery Time by Traffic Level')
plt.xlabel('Traffic Level')
plt.ylabel('Average Delivery Time (minutes)')
plt.show()
```

```
/tmp/ipython-input-4037448811.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `l

  sns.barplot(x='Traffic_Level', y='Delivery_Time_min', data=avg_delivery_traffic, palette='coolwarm')
```


Average Delivery Time by Traffic Level
Average Delivery Time (minutes)
High
Low
Medium
Traffic Level