```
1  import pandas as pd
2  df = pd.read_csv("Zomato Dataset.csv")
3  df.head()
```

	ID	Delivery_person_ID	Delivery_person_Age	Delivery_person_Ratings	Restaurant_latitude	Restaurant_longitude	Deliver
0	0xcdcd	DEHRES17DEL01	36.0	4.2	30.327968	78.046106	
1	0xd987	KOCRES16DEL01	21.0	4.7	10.003064	76.307589	
2	0x2784	PUNERES13DEL03	23.0	4.7	18.562450	73.916619	
3	0xc8b6	LUDHRES15DEL02	34.0	4.3	30.899584	75.809346	
4	0xdb64	KNPRES14DEL02	24.0	4.7	26.463504	80.372929	

Next steps: Generate code with df | View recommended plots

```
1  from sklearn.model_selection import train_test_split
2  from sklearn.preprocessing import LabelEncoder
3  from sklearn.ensemble import HistGradientBoostingRegressor
4  from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error, mean_absolute_percentage_error
```

```
1 # Separate the independent and dependent variables
2 X = df[['Restaurant_latitude', 'Delivery_person_Age', 'Delivery_person_Ratings','Restaurant_longitude', 'Delivery_location_latit
3 y = df['Time_taken (min)']
```

```
# Create label encoders for the nominal variables
label_encoders = {
    'City': LabelEncoder(),
    'Festival': LabelEncoder(),
    'Type_of_vehicle': LabelEncoder(),
    'Type_of_order': LabelEncoder(),
    'Road_traffic_density': LabelEncoder(),
    'Weather_conditions': LabelEncoder()
}
# Encode the nominal variables
for variable in label_encoders.keys():
    X[variable] = label_encoders[variable].fit_transform(X[variable])
```

Show hidden output

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

```
# Create a HistGradientBoostingRegressor object
model = HistGradientBoostingRegressor()
```

```
# Train the model using the training data
model.fit(X_train, y_train)
```

▾ HistGradientBoostingRegressor

HistGradientBoostingRegressor()

```
# Make predictions on the test data
y_pred = model.predict(X_test)
```

```
# Calculate the mean squared error and R^2 score
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

```
  # Print the model fit indices
  print("Mean Squared Error:", mse)
  print("Mean Absolute Error:", mae)
  print("R Squared:", r2)
  print("Mean Absolute Percentage Error:", mape)
```

```
Mean Squared Error: 21.722572823378236
Mean Absolute Error: 3.6831371650509013
R Squared: 0.7568322282407817
Mean Absolute Percentage Error: 0.16063275205011818
```