Food Delivery Time Prediction

Introduction :

- Food delivery is a crucial part of modern life, especially with the growth of online food ordering platforms.
- Accurately predicting delivery time improves **customer satisfaction**, **operational efficiency**, and **resource management**.
- This project uses **machine learning** to predict how long a food delivery will take based on multiple real-world conditions.
- The system analyzes several factors like **distance**, **traffic**, **weather**, and **vehicle type** to estimate delivery time.
- This report explains the full pipeline, from data preparation to model development and results.

Libraries Used :

This project uses the following Python libraries:

- **pandas** – To load, clean, and analyze data
- **numpy** – For numerical operations
- **matplotlib & seaborn** – For visualizing trends and patterns
- **scikit-learn** – For encoding, model building, evaluation, and preprocessing
- **streamlit** – For building a user-friendly web interface

Process Overview :

1. Data Loading and Preprocessing

- The dataset comes from `Food_Delivery_Times.csv`, which includes records like distance, traffic, weather, and preparation time.
- Columns with missing values were cleaned or transformed.
- Delivery time was used as the **target variable**.
- Distance and time-related data were normalized or scaled to bring all values to the same scale.
- Categorical features (like weather, traffic level, etc.) were **One-Hot Encoded** to convert text into machine-readable numbers.

2. Explanation of Features

Each delivery record contains the following information:

Feature	Description
`Distance_km`	Distance between restaurant and customer in kilometers
`Weather`	Weather condition during delivery (e.g. Sunny, Rainy)
`Traffic_Level`	Traffic at the time of delivery (Low, Medium, High, Jam)

`Time_of_Day`	Whether delivery occurred during morning, afternoon, etc.
`Vehicle_Type`	Type of vehicle used (Bike, Car, Scooter, etc.)
`Preparation_Time_min`	Time taken to prepare food before delivery
`Courier_Experience_yrs`	Years of experience the delivery person has
`Delivery_Time_min`	(Target) Total time taken to complete the delivery

---

3. Visualizations

The following insights were visualized during **EDA (Exploratory Data Analysis):**

- Histogram of delivery times – showed the most common delivery durations.
- Correlation heatmap – showed strong relationships between distance, preparation time, and delivery time.
- Boxplots – revealed how delivery time varies across traffic levels and weather conditions.
- Count plots – showed which weather/traffic conditions were most common.

---

4. Model Training and Prediction

- Features (`X`) and target (`y`) were separated.
- Categorical features were encoded using `OneHotEncoder`.
- The dataset was split into **training (80%)** and **testing (20%)** sets.
- Multiple models were trained, including:
  - **Linear Regression**
  - **Random Forest Regressor**
  - **Support Vector Regressor (SVR)**
  - **K-Nearest Neighbors (KNN)**

---

5. Evaluation Results

Model	MAE (Mean Absolute Error)
Linear Regression	**5.32** minutes
Random Forest	6.81 minutes
SVR	7.40 minutes
KNN	7.65 minutes

✅ The **Linear Regression model** gave the lowest error, meaning it predicted delivery time most accurately among the tested models.

---

6. User Interface (Streamlit App)

- A simple UI was built using Streamlit.
- Users can input the following:
  - Distance in km

- Weather condition
- Traffic level
- Time of day
- Vehicle type
- Food preparation time
- Courier's experience

- After input, the app returns:
  - **Estimated delivery time in minutes**
  - The result is shown with a success message

Conclusion :

This project built a working machine learning model that predicts food delivery times using real-world features such as traffic, weather, and courier experience. The model can help food delivery platforms:

- Set more accurate ETAs (Estimated Time of Arrival)
- Improve delivery route planning
- Enhance customer trust with timely deliveries

The best-performing model (Linear Regression) achieved a **Mean Absolute Error of just 5.32 minutes**, making it a reliable choice for practical use.

Output Image :