### **Food Delivery Time Prediction**

**Objective** The goal is to predict whether food delivery will be fast or delayed based on features like customer location, restaurant location, weather conditions, traffic conditions, and more. This task is a **binary classification problem** where the model will predict delivery status: "Fast" or "Delayed."

#### **Phase 1: Data Preprocessing**

* **Data Import and Cleaning:**
  + Load the dataset ([Food\_Delivery\_Time\_Prediction.csv](https://drive.google.com/file/d/1gAJGXBob3zRpJkRWQhvf3a94PS6NhGjx/view?usp=sharing)).
  + Handle missing values through imputation.
  + Encode categorical features (e.g., weather, traffic, and vehicle type) using LabelEncoder.
  + Normalize continuous features such as distance and delivery time.
* **Feature Engineering:**
  + Calculate the geographic distance between the customer and restaurant using latitude and longitude (Haversine formula).
  + Create binary categories based on delivery time (e.g., 1 for delayed and 0 for fast).

#### **Phase 2: Classification using Naive Bayes, K-Nearest Neighbors, and Decision Tree**

1. **Naive Bayes Classifier**:  
   * Apply the **Gaussian Naive Bayes** classifier, which is suitable for continuous features, to predict the binary class of delivery status (fast or delayed).
   * **Evaluation Metrics:** Accuracy, Confusion Matrix, Precision, Recall, F1-score.
2. **K-Nearest Neighbors (KNN)**:  
   * Use the **KNN classifier** for the binary classification of fast vs. delayed deliveries.
   * **Hyperparameter Tuning:** Find the optimal value for the number of neighbors (K) using cross-validation.
   * **Evaluation Metrics:** Accuracy, Confusion Matrix, Precision, Recall, F1-score.
3. **Decision Tree**:  
   * Train a **Decision Tree classifier** to model the classification of delivery times.
   * **Hyperparameter Tuning:** Prune the tree to avoid overfitting using max\_depth and min\_samples\_split.
   * **Evaluation Metrics:** Accuracy, Confusion Matrix, Precision, Recall, F1-score.

#### **Phase 3: Reporting and Insights**

* **Model Comparison:**
  + Compare the performance of Naive Bayes, KNN, and Decision Tree classifiers using metrics such as accuracy, precision, recall, and F1-score.
  + Visualize the confusion matrix and ROC curves to analyze the classification results.
* **Actionable Insights:**
  + Identify the strengths and weaknesses of each model.
  + Recommend the best classifier based on the task requirements (e.g., accuracy, interpretability).

## **Final Deliverables**

1. **Jupyter Notebook (.ipynb)** containing the entire code and analysis.
2. **Data Visualizations** in image format or embedded in the notebook.
3. **Final Report** summarizing key findings, model evaluations, and actionable recommendations.