Shipment Prediction System

1. Project Overview

The Shipment Prediction System is a machine learning tool designed to predict whether a shipment will be "Delayed" or "On Time". It uses various shipment attributes, including origin, destination, vehicle type, weather conditions, and distance, to make predictions about the delivery status.

2. Technologies Used

- **Python**: Programming language for the project.
- **Pandas**: Data manipulation and analysis.
- **Scikit-learn**: For building and training machine learning models (SVM).
- **XGBoost**: An alternative to SVM, for better handling of complex datasets.
- **Flask/FastAPI** (Optional): For deploying the prediction as a web service.

3. Steps

- **Data Preprocessing:**
    - **Drop irrelevant columns (e.g.,Shipment ID).**
    - **Create new features like `year of delivery`.**
    - **Encode categorical variables (e.g., Origin, Destination) into numerical values using one-hot encoding.**
- **Model Selection:**
    - **The system uses ADA Boost for prediction after comparing performance of models like XG Boost, Random Forest, Decision Trees, Logistic Regression, Naves Bayes, SVC etc over key metrics recall and precision owing to bias and variance tradeoff of ADA Boost, also XG Boost and logistic regression we the 2nd best option.**
    - **The model predicts whether the shipment is "Delayed" or "On Time" based on input features.**
- **Prediction:**
    - **Users can input shipment details (e.g., origin, destination, weather conditions) into the system, and it will predict if the shipment will be delayed or not.**

4. Core Functions

- **predict_shipment_status(shipment_details):**

    - Input: List of shipment details (e.g., ['SHIP000000', 'Jaipur', 'Mumbai', '2023-04-26', 'Trailer', 1603, 'Rain', 'Light']).
    - Output: Prediction of shipment status ("Delayed" or "On Time").

Image Caption Generator Web Application

Objective

This project is designed to generate a description of an image uploaded by the user. The web application uses a pre-trained image captioning model to analyze the content of the image and provide a natural language description.

Technology Stack

- Backend: Python
- Machine Learning Model: BLIP (Bootstrapping Language-Image Pretraining) model from Hugging Face
- Libraries:
    - `transformers`: For utilizing the BLIP image captioning model.
    - `PIL`: For image handling and processing.
    - `IPython.display`: For displaying images inline.
    - `google.colab.files`: For handling file uploads in Colab.

Functionality

1. Image Upload: The user can upload an image file using the Colab file upload interface.
2. Image Captioning: Once the image is uploaded, it is passed through the BLIP model, which generates a textual description based on the image content.
3. Display Results: After processing, the generated description is displayed, and the uploaded image is shown to the user.

Workflow

1. **The user uploads an image.**
2. **The image is processed using the pre-trained BLIP model.**
3. **A description of the image is generated and displayed.**
4. **The image is also shown alongside the description.**

Output

The application generates a description for the uploaded image, and the description is printed along with the image being displayed.