EcoPack-AI Project Documentation

1. Project Overview

EcoPack-AI is an AI-powered full-stack web platform designed to recommend optimal packaging materials based on product attributes, sustainability parameters, and industry standards. The system addresses the reliance on non-biodegradable materials by providing intelligent decision support for eco-friendly alternatives.

+1

- **Live Demo:** https://ecopack-ai-ojwg.onrender.com/
- **Repository:** GitHub Link

2. Implementation by Milestone

Milestone 1: Data Collection & Management

Focus: Establishing the data foundation, including material attributes and product categories.

- **Data Source & Structure:**
    - The system utilizes a structured dataset containing material types (e.g., Corrugated Box, Paper) and their properties.
    - **Implemented File:** cleaned_data.csv and materials_database.joblib serve as the cleaned and serialized knowledge base for the application.
- **Data Processing:**
    - **Cleaning:** Implemented via data_cleaning.ipynb, which handles missing values and normalizes numerical features.
    - **Feature Engineering:** categorical material properties were encoded, and specific indices for CO2 and Cost efficiency were derived to train the models.
    - **Artifacts:** Label encoders (co2_label_encoder.joblib, cost_label_encoder.joblib) are generated to map categorical data for the ML models.

Milestone 2: AI & ML Model Development

Focus: Developing and training machine learning models for cost and CO2 prediction.

- **Predictive Models:**
    - **CO2 Emission Model:** Uses **XGBoost Regressor** to predict the environmental footprint of packaging materials.
        - *Artifact:* co2_prediction_model.joblib

- **Cost Prediction Model:** Uses **Random Forest Regressor** to estimate financial implications.
    - *Artifact:* cost_prediction_model.joblib

- **Model Integration:**
    - The application utilizes a ModelLoader class to initialize these models along with their respective feature order files (co2_feature_order.joblib, cost_feature_order.joblib) to ensure consistent input formatting during inference.

Milestone 3: Backend API & Frontend UI Integration

Focus: Connecting the AI models to a Flask backend and developing the user interface.

- **Backend Architecture (Flask):**
    - **API Framework:** The core logic is built on Flask (app.py), serving as the bridge between the UI and the ML models.
    - **Key Endpoints:**
        - /recommend (POST): Accepts product weight, volume, distance, and mode. Returns a ranked list of materials with predicted CO2 and Cost.
        - /health (GET): System status check to ensure models are loaded correctly.
        - /model-info (GET): Returns metadata about the trained models, including versioning and error metrics (RMSE/MAE).
- **Recommendation Logic:**
    - The system calculates a **Combined Score** based on user preference (Eco-Friendly vs. Cost-Effective).
    - *Eco-Mode:* 80% weight on CO2 reduction, 20% on Cost.
    - *Cost-Mode:* 80% weight on Cost, 20% on CO2.
- **Frontend UI:**
    - **Interface:** Built using HTML/CSS/Bootstrap, accessible via the root route /.
    - **Features:** Interactive forms for "Shipping Mode" (Air, Road, Rail, Sea) and optimization goals, displaying results in a ranked table.

Milestone 4: Deployment & Analytics

Focus: System deployment, documentation, and business intelligence features.

- **Deployment:**
    - The application is deployed on **Render**, making the sustainable packaging recommendation system publicly accessible.

- **Business Intelligence & Reporting:**
    - While a full standalone dashboard is planned, the current system provides real-time analytics via the /model-info endpoint, exposing model performance metrics (RMSE, MAE) and training dates.
    - The system outputs specific "Predicted CO2" and "Predicted Cost" values for every recommendation, enabling immediate quantitative comparison for the user.

3. Technical Architecture Summary

- **Language:** Python 3.8+
- **Framework:** Flask
- **ML Libraries:** Scikit-learn, XGBoost, Pandas, NumPy, Joblib
- **Frontend:** HTML5, CSS3, JavaScript
- **Infrastructure:** Render (Cloud Platform)

4. How to Use

1. **Input:** Enter shipment weight (kg), volume (m^3), and distance (km).
2. **Select Mode:** Choose transport method (Road, Air, Sea, Rail).
3. **Optimize:** Select "Eco-Friendly" to prioritize carbon reduction.
4. **Result:** View top 5 sustainable material recommendations with detailed CO2 and cost impact analysis.