Frontend: Automated Review Rating System

Frontend Overview: Streamlit Application

The frontend is implemented using **Streamlit**, providing an interactive web interface for predicting product review ratings. Users can input any textual review, and the app compares predictions from two trained machine learning models:

- **Model A:** Trained on a balanced dataset
- **Model B:** Trained on an imbalanced dataset

Predictions are displayed side by side along with confidence scores, enabling easy comparison of model behavior.

Key Components

1. Model & Vectorizer Loading:

Pre-trained Logistic Regression models (`Model_Balanced.pkl` and `Model_Imbalanced.pkl`) and their corresponding TF-IDF vectorizers (`tfidf_Balanced.pkl` and `tfidf_Imbalanced.pkl`) are loaded using `joblib`. Models are ready to perform inference on new reviews.

2. Text Preprocessing:

Function `preprocess_review()` handles text cleaning:

- Converts text to lowercase.
- Removes URLs, HTML tags, special characters, and extra whitespace.
- Lemmatizes words using SpaCy.
- Removes stopwords to retain meaningful words.

This ensures that input text aligns with the format used during model training.

3. Prediction Function:

Function `predict_review()` takes a raw review, preprocesses it, applies TF-IDF vectorization, and generates the predicted rating and confidence score from the selected model.

4. User Interface:

- Page title and subtitle with custom CSS styling.
- Input area for typing or pasting a review.
- Predict button triggers rating prediction for both models.
- Results displayed in cards with distinct borders and color themes:
  - Balanced Model: Green border
  - Imbalanced Model: Orange border
- Confidence is visualized using Streamlit progress bars.

5. Additional Features:

- Spinner animation provides feedback while processing.
- Footer credits the use of Streamlit and SpaCy.

Workflow

1. User enters a review in the text area.
2. Upon clicking `Predict Rating` button:
    - The review is preprocessed.
    - Predictions are generated using both balanced and imbalanced models.
    - Results are displayed side by side:
        - Predicted rating (1–5 stars)
        - Model confidence
3. Users can visually compare how different training data distributions affect predictions.

Technologies Used

- **Streamlit:** Frontend web application framework for Python.
- **SpaCy:** NLP preprocessing (lemmatization, stopword removal).
- **Joblib:** Model and vectorizer serialization/deserialization.
- **Python Libraries:** `re` for regex cleaning, `numpy` for numeric operations, `time` for spinner simulation.

Python Code Used

```
import streamlit as st
import joblib
import re
import spacy
import numpy as np
import time

# --- Load NLP Model ---
nlp = spacy.load("en_core_web_sm")

# --- Load Models and Vectorizers ---
model_dir = "../models"
model_A = joblib.load(f"{model_dir}/Model_Balanced.pkl")
model_B = joblib.load(f"{model_dir}/Model_Imbalanced.pkl")
vectorizer_A = joblib.load(f"{model_dir}/tfidf_Balanced.pkl")
vectorizer_B = joblib.load(f"{model_dir}/tfidf_Imbalanced.pkl")

# --- Helper Functions ---
```

```
def preprocess_review(text):
    """Clean, remove stopwords, and lemmatize text."""
    text = text.lower()
    text = re.sub(r"http\S+|www\S+|https\S+", "", text)
    text = re.sub(r"<.*?>", "", text)
    text = re.sub(r"[^a-zA-Z0-9\s]", "", text)
    text = re.sub(r"\s+", " ", text).strip()
    doc = nlp(text)
    return " ".join([token.lemma_ for token in doc if token.text not in
        nlp.Defaults.stop_words])

def predict_review(review, model, vectorizer):
    processed = preprocess_review(review)
    tfidf_review = vectorizer.transform([processed])
    prediction = model.predict(tfidf_review)[0]
    confidence = model.predict_proba(tfidf_review).max()
    return prediction, confidence

# --- Streamlit Page Setup ---
st.set_page_config(page_title="Review Rating Predictor", page_icon="",
    layout="wide")

# --- Custom CSS for Styling ---
st.markdown("""
    <style>
        body {
            background-color: #f8f9fa;
        }
        .main-title {
            text-align: center;
            color: #2E8B57;
            font-size: 40px;
            font-weight: 700;
            margin-bottom: 10px;
        }
        .sub-title {
            text-align: center;
            color: #555;
            font-size: 18px;
            margin-bottom: 40px;
        }
        .card {
            padding: 20px;
            border-radius: 15px;
            box-shadow: 0 4px 10px rgba(0,0,0,0.1);
            background-color: white;
            transition: transform 0.3s ease;
        }
        .card:hover {
            transform: scale(1.03);
        }
```

```
        .balanced {
            border-left: 8px solid #4CAF50;
        }
        .imbalanced {
            border-left: 8px solid #FF9800;
        }
        .confidence-bar {
            height: 10px;
            border-radius: 10px;
            margin-top: 5px;
        }
        .footer {
            text-align:center;
            color:gray;
            margin-top:50px;
            font-size:14px;
        }
    </style>
""", unsafe_allow_html=True)

# --- Title Section ---
st.markdown("<div class='main-title'> Review Rating Predictor</div>",
    unsafe_allow_html=True)
st.markdown("<div class='sub-title'>Compare predictions from Balanced
    vs. Imbalanced Models</div>", unsafe_allow_html=True)

# --- Input Section ---
review_input = st.text_area(" Enter a product review below:",
    placeholder="Type your review here...")

if st.button(" Predict Rating"):
    if review_input.strip() == "":
        st.warning("Please enter a review first.")
    else:
        with st.spinner(". Loading Analyzing review....."):
            time.sleep(1.2)
            pred_A, conf_A = predict_review(review_input, model_A,
                vectorizer_A)
            pred_B, conf_B = predict_review(review_input, model_B,
                vectorizer_B)

        st.success(" Prediction completed successfully")

        # --- Display Results Side by Side ---
        col1, col2 = st.columns(2, gap="large")

        with col1:
            st.markdown("<div class='card balanced'>", unsafe_allow_html=
                True)
            st.markdown("###  Model A (Balanced Data)")
            st.write(f"**Predicted Rating:**  {pred_A}")
```

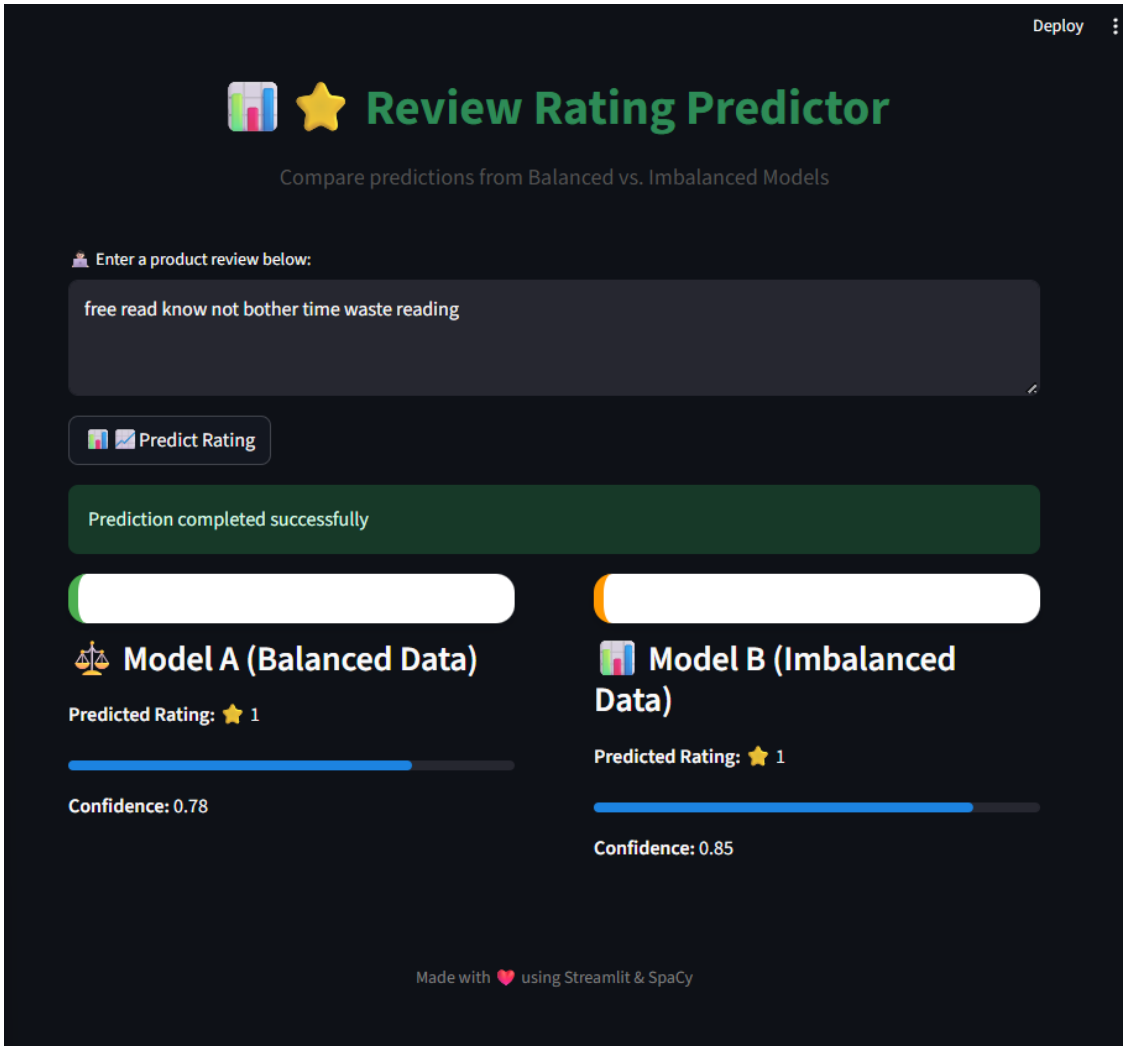
```
        st.progress(float(conf_A))
        st.write(f"**Confidence:** {conf_A:.2f}")
        st.markdown("</div>", unsafe_allow_html=True)

    with col2:
        st.markdown("<div class='card imbalanced'>", unsafe_allow_html
            =True)
        st.markdown("### Model B (Imbalanced Data)")
        st.write(f"**Predicted Rating:** {pred_B}")
        st.progress(float(conf_B))
        st.write(f"**Confidence:** {conf_B:.2f}")
        st.markdown("</div>", unsafe_allow_html=True)



st.markdown("<div class='footer'>Made with using Streamlit & SpaCy</
    div>", unsafe_allow_html=True)
```

Outputs:


Deploy
Review Rating Predictor
Compare predictions from Balanced vs. Imbalanced Models
Enter a product review below:
free read know not bother time waste reading
Predict Rating
Prediction completed successfully
Model A (Balanced Data)
Predicted Rating: 1
Confidence: 0.78
Model B (Imbalanced Data)
Predicted Rating: 1
Confidence: 0.85
Made with using Streamlit & SpaCy