**Mini Project 1: Building a Sentiment Analysis System for Movie Reviews**

**Introduction:**

This mini project requires you to implement a sentiment analysis system similar to Chapter 8 of your book. You will apply the skills and techniques covered in Chapters 1 to 6 to complete this task. Additionally, you will integrate the **CodeCarbon** library to measure the carbon footprint of your model, discuss the ethical implications of your AI system, and explore the deployment of your model on embedded systems with limited compute and memory resources.

The machine learning model to be used in this project is **Logistic Regression**. Logistic Regression is a well-suited algorithm for sentiment classification tasks and provides interpretable predictions. You will train, evaluate, and deploy this model throughout the project.

You will also be required to build a **GitHub repository** with a **detailed README** that includes project documentation, licensing information, and best practices for responsible code release and maintenance. The final deliverable includes a GitHub repository with proper documentation and licensing.

**Tasks and Goals:**

**1. Data Preparation**

**Goal:** Load and inspect the IMDb dataset containing movie reviews labeled with positive and negative sentiments. (<https://ai.stanford.edu/%7Eamaas/data/sentiment/>)  
**Task:** Read the dataset, store the reviews and their associated sentiments, and explore the dataset to understand its structure.

**2. Text Preprocessing**

**Goal:** Clean and preprocess the text data to remove noise and prepare it for analysis.  
**Task:** Remove unnecessary characters (e.g., HTML tags, punctuation), convert text to lowercase, and process words by removing stop words and stemming/lemmatizing them.

**3. Feature Extraction**

**Goal:** Transform the cleaned text into numerical features for machine learning.  
**Task:** Use a vectorization technique such as TF-IDF to convert the text into a numerical matrix that captures the importance of each word in the dataset.

**4. Model Training**

**Goal:** Train a machine learning model to classify reviews based on their sentiment.  
**Task:** Split the dataset into training and testing sets, train a Logistic Regression model, and evaluate its performance on the test data.

**5. Model Evaluation**

**Goal:** Assess the performance of your model using appropriate metrics.  
**Task:** Evaluate precision, recall, and F1-score of the Logistic Regression model. Use these metrics to identify the strengths and weaknesses of your system. Visualize the Confusion Matrix to better understand how well the model classifies positive and negative reviews. Additionally, test the model with a new review, preprocess it, make a prediction, and display the result. Example: test it with a new review such as:

"*The movie had great visuals, but the storyline was dull and predictable.*" The expected output might be: Negative Sentiment.

**6. Hyperparameter Tuning**

**Goal:** Optimize your Logistic Regression model by tuning its hyperparameters.  
**Task:** Use an optimization method to find the best parameters for your model and improve its accuracy.

**7. Learning Curve Analysis**

**Goal:** Diagnose your model's performance by plotting learning curves.  
**Task:** Analyze training and validation performance as a function of the training set size to identify underfitting or overfitting issues.

**8. Carbon Footprint Analysis with CodeCarbon**

**Goal:** Quantify and discuss the carbon footprint of your model using the CodeCarbon methodology.  
**Task:** Install and integrate CodeCarbon into your project. Track and report carbon emissions of your model. You can find needed information here: <https://mlco2.github.io/codecarbon/> & <https://codecarbon.io/>

**9. Ethical Considerations and Explainability**

**Goal:** Discuss the ethics in using and deploying your AI-based solution by investigating and implementing suitable explainability methods.  
**Task:** Understanding how a machine learning model makes predictions is crucial for ensuring transparency, fairness, and accountability in AI deployment. One of the widely used techniques for model explainability is SHAP (SHapley Additive exPlanations), which helps determine how much each feature (word) contributes to a prediction.

In this task, you will use SHAP to analyze the impact of individual words on sentiment classification. This will allow you to visualize which words increase or decrease the probability of a positive or negative sentiment prediction. Additionally, discuss key aspects such as potential biases in the model, fairness in outcomes, and accountability in AI decision-making. You can find more information here: <https://shap.readthedocs.io/en/latest/generated/shap.Explanation.html>

**10. Deployment Considerations for Embedded Systems**

**Goal:** Optimize and convert the trained logistic regression model for deployment on embedded systems like Arduino  
**Task:** To deploy the trained logistic regression model on a resource-constrained embedded system like an Arduino, we must optimize and convert the model into a format suitable for execution in an environment with limited memory and processing power. Since embedded systems do not support direct execution of machine learning models trained in Python, we extract the model’s learned parameters—namely, the weights and bias—after training. These parameters are then quantized to fixed-point integers to eliminate the need for floating-point calculations, which are inefficient on microcontrollers.

Once quantization is applied, we generate a C++ .h header file containing the model’s coefficients and bias, formatted in a way that allows direct use within an Arduino sketch. The final model is optimized to perform inference using integer arithmetic, making it both lightweight and efficient for deployment on microcontrollers. You can find more information here: <https://medium.com/@thommaskevin/tinyml-binomial-logistic-regression-0fdbf00e6765>

**GitHub Repository and Documentation Requirements:**

You are required to create a **GitHub repository** for your project and include the following:

* **A detailed README** that includes:
  + Project overview and objectives
  + Instructions for dataset download and preprocessing
  + Model training and evaluation steps
  + Carbon footprint analysis results
  + Ethical considerations and explainability methods
  + Embedded system deployment insights
* **A section on code release responsibilities:**
  + Define your responsibility regarding source code release.
  + Justify the inclusion of an appropriate open-source license (e.g., MIT, Apache, GPL) and discuss its implications.
  + Compare different code release practices and how they impact transparency, collaboration, and software maintenance.
  + Outline a bug reporting and fixing protocol (e.g., issue tracking, pull requests).