**Name :** Ayan Das

**Date :** 05/06/2024

Summary of Each Python File

1. MultinomialNB.py

This file implements a Multinomial Naive Bayes Classifier, a common probabilistic method in Natural Language Processing (NLP). The classifier handles text data using a bag-of-words model and applies Add-one (Laplace) smoothing to address zero probabilities in categorical data.

- **fit:** Trains the model with feature vectors and labels, calculating essential probabilities for classification.

- **predict\_log\_proba:** Computes log probabilities of each class for given input samples based on learned model parameters.

- **predict:** Determines the most likely class label for each input sample based on the log probabilities.

- **create\_vocab:** Generates a sorted list of unique words from documents, creating a vocabulary.

- **vectorize\_docs**: Converts text documents into numerical format using the vocabulary, creating feature vectors.

2. main.py

This central driver script manages data processing, model training, and evaluation for both a small corpus of labeled movie reviews and a large real-world dataset (IMDB reviews). It uses functions and classes from `MultinomialNB.py` and `DataPreprocessing.py` to ensure each data handling stage is seamlessly executed.

**Loading and Preprocessing Data:**

- Loads the IMDB dataset.

- Applies text preprocessing to normalize the data by removing HTML tags, converting text to lowercase, removing punctuation, and filtering out stop-words.

**Vocabulary Creation and Data Vectorization:**

- Creates a vocabulary from the training set's processed text.

- Saves this vocabulary to a text file.

- Vectorizes both the training and test data using the created vocabulary.

**Model Training and Evaluation:**

- Trains the Multinomial Naive Bayes classifier with a small corpus and extends training to the first 25,000 IMDb reviews.

- Evaluates the classifier on the remaining IMDb reviews, calculating accuracy, precision, recall, and F1-score.

**Prediction and Output:**

- Makes predictions on new, unseen text ("fast couple shoot fly").

- Outputs the predicted class and log probabilities for each class.

Performance Metrics Reporting:

- Reports statistical measures such as accuracy, precision, recall, and F1 score.

3. DataPreprocessing.py

This file handles all preprocessing tasks for text data, ensuring it is suitably formatted for the classifier.

- **preprocess\_text:** Cleans the text by removing HTML tags, converting text to lowercase, removing punctuation, and filtering out stop-words.

- **create\_vocab:** Creates a vocabulary list from processed text.

- **save\_vocabulary:** Saves the generated vocabulary to a text file for future reference.

**Output Explanation and Performance Evaluation for main.py:**

- **Accuracy:** 85.25% of the predictions were correct, indicating high effectiveness in sentiment determination.

- **Precision:** 86.88% precision suggests high reliability in predicting positive reviews.

- **Recall:** 83.11% recall shows the classifier's capability in identifying most positive reviews correctly.

- **F1 Score:** An F1 score of approximately 84.95% indicates a balanced performance between precision and recall.