**SUMMARY OF THE ADVANCE MADE – CARLOS AND LEO 11/07/2024**

**Import Libraries:**

Essential libraries for data manipulation, text processing, model building, and evaluation are imported, including pandas, scikit-learn, nltk, and matplotlib.

**Download NLTK Resources:**

Necessary resources like stopwords and the WordNet lemmatizer are downloaded from the NLTK library.

**Load the Dataset:**

The dataset is loaded from a CSV file located in the same directory.

**Initial Exploration of the Dataset:**

Basic information about the dataset info, description, shape, label distribution, unique values, and text samples was printed.

**Cleaning Process:**

A function to clean the text was defined, the cleaning made was the following:

**Remove URLs:** Deletes any URLs present in the text.

**Remove Mentions and Hashtags:** Cleans out any Twitter-style mentions and hashtags.

**Remove Special Characters and Digits:** Keeps only alphabetical characters and spaces.

**Convert to Lowercase:** Converts all text to lowercase to ensure uniformity.

**Remove Extra Spaces:** Trims leading and trailing spaces.

**Remove Stop Words:** Eliminates common English stop words using NLTK’s stopwords list.

**Lemmatization:** Converts words to their base form using the WordNet lemmatizer.

**Sentiment Features calculated:**

Resulting in two new columns for the DF, sentiment\_polarity and sentiment\_subjectivity.

**LDA for dominant topics in the data:**

This distribution can help you understand the main themes in your corpus and their relative importance or frequency. It could also be used as a feature in further analysis or classification tasks. Output files here are a new df.csv and a lda\_visualization dashboard to understand the Intertopic Distance

**Select Features and Target:**

The text data (features) and labels (target) were separated.

**Split the Dataset:**

The dataset is split into training and testing sets using an 80-20 split.

**Apply Text Cleaning:**

The text cleaning function was applied to both the training and testing sets to preprocess the text data.

**Word Frequency Analysis:**

**Tokenization and Frequency Counting:**

All cleaned text from the training set is concatenated into a single string and split into individual words. The frequency of each word is then counted using Counter from the collections module.

**Display and Plot Top Words:**

The top 40 most common words and their frequencies are printed. Optionally, a horizontal bar chart is plotted to visualize these frequencies.

**Define Pipeline:**

A scikit-learn pipeline was created with a TF-IDF vectorizer and a Logistic Regression model.

**TF-IDF Vectorizer:**

Converts the cleaned text data into numerical features using Term Frequency-Inverse Document Frequency (TF-IDF) method.

**Logistic Regression:**

A simple yet effective classification model.

**Define Parameter Grid:**

A grid of hyperparameters for both the TF-IDF vectorizer and Logistic Regression model is defined to be used for hyperparameter tuning.

**Grid Search with Cross-Validation:**

Grid search with 5-fold cross-validation is performed to find the best combination of hyperparameters that maximizes the AUC score.

**Evaluate Model:**

The best model from the grid search is used to make predictions on the test set. The AUC score is calculated to evaluate the model’s performance. We got 96.90%

**Display Best Parameters:**

The best combination of hyperparameters found during the grid search is printed.