COMP 550 Assignment – 1

Vraj Patel McGill ID: 261022581

- **Problem setup:**
  The primary objective of this assignment is to develop and evaluate Natural Language processing models for classifying real facts about animals from fake facts. For this I have used various preprocessing techniques, including lemmatization, stemming, and N-grams. The text data undergoes feature extraction to convert it into numerical representations, which are then used by linear classifiers to categorize the sentences into two distinct labels.

- **Dataset generation and experimental procedure:**
  A dataset comprising 240 facts and fake facts about different animals was generated using ChatGPT. Real facts were labeled as '1,' while fake facts were assigned '0.' Text preprocessing techniques included Lemmatization, Stemming, converting text to lowercase, punctuation removal, white space trimming, and stop words elimination. These steps aimed to enhance feature extraction and data quality. Feature extraction employed the **CountVectorizer** from scikit-learn, transforming text into tokens and creating a sparse matrix for model training. Three linear classifiers, Naive Bayes, Logistic Regression, and Support Vector Machines, were employed for classification. The dataset was split into an 80% training set and a 20% test set using train-test split from scikit-learn, with accuracy measured on the test data. In generating the dataset, ChatGPT was used to create animal-related facts and fake facts. Additionally, N-gram models, specifically unigrams and bigrams, was conducted during the preprocessing stage.

- **Range of parameters setting that tried:**
  During the experimentation phase, I systematically varied the number of features in the CountVectorizer to assess its impact on model performance. The parameters ranged from 200 to 1300, with increments of 100. Notably, the optimal parameter settings were found to be 800 for stemming and 1200 for bigram feature extraction, as they resulted in the best model performance. Regarding feature extraction, I explored three distinct combinations:
  - **Removing Special Characters:** Special characters were removed from the text data to ensure that the models focus on the linguistic content rather than symbols or punctuation.
  - **Removing Numbers:** Numerical values were eliminated from the text to simplify and standardize the textual content.
  - **White Space Trimming and Stopwords Removal:** White spaces were trimmed to enhance data cleanliness, and common stopwords were removed to remove the less informative words.

- **Results and conclusions:**

Preprocessing and classifier	**Accuracy**
Naïve Bayes (lemmatization + Unigrams only + stop words + No white space)	83.3 %
Naïve Bayes (stemming + Unigrams only + stop words + No white space)	83.3 %
Naïve Bayes (Bi-grams only + stop words + No white space)	58.3 %
SVM (lemmatization + Unigrams only + stop words + No white space)	62.5 %
SVM (stemming + Unigrams only + stop words + No white space)	62.5 %
SVM (Bi-grams only + stop words + No white space)	50.0 %
Logistic Regression (lemmatization + Unigrams only + stop words + No white space)	77.0 %

Logistic Regression (stemming + Unigrams only + stop words + No white space)	77.0 %
Logistic Regression (Bi-grams only + stop words + No white space)	58.3 %

Table 1

The results presented in the table 1, provide an overview of the performance of various preprocessing techniques in combination with different classifiers, along with their corresponding accuracies. Notably, models incorporating lemmatization and stemming, particularly when using unigrams, consistently outperform those relying solely on bigrams. This observation underscores the value of lemmatization and stemming in transforming text into its base form. In contrast, models using only bigrams without the aid of lemmatization and stemming struggle to extract vital information effectively. Furthermore, the vocabulary size of datasets subjected to lemmatization and stemming tends to be smaller than those without these preprocessing steps, highlighting the efficiency of these techniques in text normalization. When comparing the performance of different linear classifiers, Naive Bayes consistently emerges as the top performer. This outcome can be attributed to the classifier's adeptness at handling high-dimensional data in text classification tasks. Given the use of N-grams as a preprocessing technique and CountVectorizer for feature extraction, Naive Bayes excels, particularly when word frequencies play a critical role in the classification process. It is worth noting that SVM, while renowned for its performance on large datasets, lags behind other classifiers in these experiments. This is likely due to the relatively smaller dataset size in this context.
In summary, the combination of preprocessing techniques, feature extraction methods, and classifier selection significantly influences the effectiveness of text classification. Lemmatization and stemming, in conjunction with Naive Bayes, stand out as a powerful combination for this particular task.

- **Limitations of the study:**
  This study has limitations from the modest dataset size of 240 instances, which may not fully capture real-world complexity. In addition, it only focuses on textual characteristics for classification does not incorporate contextual information.