**EDS 6397 – NLP**

**Assignment 2 – Text Classification**

**Varun Vaddi - #2347481**

**Dataset:**

Downloaded the Dataset from Kaggle – **‘moviereviews.tsv’**, which contains two columns i.e., ‘**label’** & ‘**review’**. Where ‘label’ column contains only two values – ‘**pos’** & ‘**neg’**.

A close-up of a document

Description automatically generated

**Data Cleaning:**

Read the **tsv** file using **read\_table()** of pandas & observed that it has 2000 rows and 2 columns, **removed 35** **reviews** for being NULL, using dropna().

A screen shot of a computer

Description automatically generated

Then encoded ‘**label’** values ‘**neg’** & ‘**pos’** to ‘**0’** & ‘**1’** respectively using **map** function.

A screenshot of a movie

Description automatically generated

**Data preprocessing:**

Written a function to perform Lemmatization, removing stop words & to handle logical negation.

A computer screen shot of text

Description automatically generated

**Train & test the model:**

Split the data into 80% Training dataset and 20% testing dataset. Then ran the model using Multinomial Naïve Bayes algorithm to Train the model, with random state = 42.

A screen shot of a computer program

Description automatically generated

**Comparison table:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Scenario** | **Precision (Negative)** | **Precision (Positive)** | **Recall (Negative)** | **Recall (Positive)** | **F1-Score (Negative)** | **F1-Score (Positive)** | **Accuracy** |
| **No Lemmatization,  with Stop Word Removal,  No Logical Negation** | 0.78 | 0.83 | 0.85 | 0.75 | 0.82 | 0.79 | 0.8 |
| **With Lemmatization,  No Stop Word Removal,  No Logical Negation** | 0.77 | 0.81 | 0.84 | 0.74 | 0.8 | 0.77 | 0.79 |
| **With Lemmatization,  with Stop Word Removal,  No Logical Negation** | 0.77 | 0.81 | 0.84 | 0.74 | 0.8 | 0.77 | 0.79 |
| **With Lemmatization,  with Stop Word Removal,  Handling Logical Negation** | 0.78 | 0.81 | 0.84 | 0.74 | 0.8 | 0.78 | 0.79 |

**Conclusion:**

1. **Overall Performance**: Scenario 1 achieved the highest accuracy **(80%)** and demonstrated the best performance in terms of F1-scores for both classes.
2. **Impact of Stop Word Removal**: Stop word removal appears to have a **NO effect** on the overall results in Scenarios 3 and 4, maintaining **similar scores** compared to Scenario 2.
3. **Impact of Lemmatization**: The presence of lemmatization **did not improve accuracy** in Scenarios 2, 3, or 4, and, in return it resulted in **slightly lower F1-scores**.
4. **Impact of Logical Negation**: Adding logical negation handling in Scenario 4 did not provide a **significant improvement** over Scenario 3, suggesting that the model may still struggle with identifying the negation effectively.
5. The results across Scenarios 2, 3, and 4 were quite **consistent**, with only **minor variations** in precision and F1-scores, indicating that while lemmatization and stop word removal have their merits, they did not drastically alter the model's performance in this case.

**Q/A:**

1. A Naive Bayes classifier handles misspelled words as distinct tokens. Using the training data, the model determines the probabilities of these tokens depending on how frequently they occur. A misspelled word can have a big impact on the classification result if it occurs frequently enough. On the other hand, if the misspelled term is less common, it might not significantly affect the model's overall performance, if the bulk of the words have the correct spelling.
2. Yes, misspelled words and typos can affect the Naive Bayes classifier. Every misspelled word can change the frequency distribution used for categorization because the model is based on word probabilities. This might cause misclassifications, if the misspelled word is like another word wit a different meaning.
3. Yes, we have a couple of methods to fix spelling errors:
   1. Run Spell-check as part of pre-processing steps
   2. Usage of Feedback loop
4. Yes, fixing spelling errors will be beneficial as:
   1. It improves overall Accuracy
   2. Reduces the risk of misclassification due to unknown tokens.
5. Using n-grams could potentially yield better results when compared to Naïve bayes classifier particularly when it comes to interpreting the word relationships and their context.
   1. When n-grams are used in model, the model becomes less sensitive to misspellings as n-grams will provide good information for classification, even though individual words are incorrectly spelled.
   2. High contextual information due to presence of n-grams, as they provide relationships between adjacent words, which will definitely help in case of words that conveys different meanings based on its surrounding words.