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**'.

moviereviews

label	review
neg	how do films like mouse hunt get into theatres ? isn't there a law or something ? this diabolical load of claptrap from steven speilberg's dreamworks studio is hollywood family fare at its deadly worst . mouse hunt takes the bare threads of a plot and tries to prop it up with overacting and flat-out stupid slapstick that makes comedies like jingle all the way look decent by comparison . writer adam rifkin and director gore verbinski are the names chiefly responsible for this swill . the plot , for what its worth , concerns two brothers (nathan lane and an appalling lee evens) who inherit a poorly run string factory and a seemingly worthless house from their eccentric father . deciding to check out the long-abandoned house , they soon learn that it's worth a fortune and set about selling it in auction to the highest bidder . but battling them at every turn is a very smart mouse , happy with his run-down little abode and wanting it to stay that way . the story alternates between unfunny scenes of the brothers bickering over what to do with their inheritance and endless action sequences as the two take on their increasingly determined furry foe . whatever promise the film starts with soon deteriorates into boring dialogue , terrible overacting , and increasingly uninspired slapstick that becomes all sound and fury , signifying nothing . the script becomes so unspeakably bad that the best line poor lee evens can utter after another run in with the rodent is : " i hate that mouse " . oh cringe ! this is home alone all over again , and ten times worse . one touching scene early on is worth mentioning . we follow the mouse through a maze of walls and pipes until he arrives at his makeshift abode somewhere in a wall . he jumps into a tiny bed , pulls up a makeshift sheet and snuggles up to sleep , seemingly happy and just wanting to be left alone . it's a magical little moment in an otherwise soulless film . a message to speilberg : if you want dreamworks to be associated with some kind of artistic credibility , then either give all concerned in mouse hunt a swift kick up the arse or hire yourself some decent writers and directors this kind of rubbish will just not do at all .

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().

```
[3]: data.shape
[3]: (2000, 2)
```

From the above output, we can observe that the data contains 2000 rows and 2 columns

Data Cleanup & 'label' column Mapping

```
[4]: # Remove rows with missing reviews
     data = data.dropna(subset=['review'])

     # Convert labels to numerical values (0 for negative, 1 for positive)
     data['label'] = data['label'].map({'neg': 0, 'pos': 1})
     data.shape
[4]: (1965, 2)
```

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

	label	review
0	0	how do films like mouse hunt get into theatres...
1	0	some talented actresses are blessed with a dem...
2	1	this has been an extraordinary year for austra...
3	1	according to hollywood movies made in last few...
4	0	my first press screening of 1998 and already i...

Data preprocessing:

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

```
# Token processing
tokens = []
for token in doc:
    if token.is_punct or token.is_space:
        continue
    if remove_stop_words and token.is_stop:
        continue
    if lemmatize_words:
        token = token.lemma_
    else:
        token = token.text
    tokens.append(token)

# Join tokens into a single string
processed_text = ' '.join(tokens)

if remove_stop_words:
    # Remove stop words - from, of, in, he/she
    stop_words = set(stopwords.words('english'))
    tokens = word_tokenize(text)
    processed_text = ' '.join(token for token in tokens if token.lower() not in stop_words)

if lemmatize_words:
    # lemmatize
    processed_text = ' '.join(token.lemma_ for token in doc)

if handle_logical_negation and lemmatize_words:
    # Replace "not" and "n't" with a placeholder for negation
    processed_text = re.sub(r'\bnot\b|\b(?!\bnot\b)\w+\'?nt\b', 'NEG', processed_text)
```

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.

```
def run_scenario(lemmatize_words, remove_stop_words, handle_logical_negation):
    # Preprocess reviews
    data['processed_review'] = data['review'].apply(
        lambda x: preprocess_text(x, lemmatize_words, remove_stop_words, handle_logical_negation)
    )

    # Split data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(
        data['processed_review'], data['label'], test_size=0.2, random_state=42
    )

    # Train and evaluate the model
    train_and_evaluate(X_train, y_train, X_test, y_test)
```

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:
    a. Run Spell-check as part of pre-processing steps
    b. Usage of Feedback loop

4. Yes, fixing spelling errors will be beneficial as:
    a. It improves overall Accuracy
    b. 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.
    a. 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.
    b. 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.