Data Mining (DSC550-T301_2245_1)

Assignement Week 3;

Author: Zemelak Goraga;

Date: 03/30/2024

In [ ]:
```
pip install pandas textblob nltk scikit-learn
```

In [1]:
```
# import required libraries
import pandas as pd
import zipfile
from textblob import TextBlob
from sklearn.metrics import accuracy_score
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
import string
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
# Load the dataset into a DataFrame, skipping lines with errors
train_data = pd.read_csv('labeledTrainData.tsv', error_bad_lines=False, sep='\t')

# Display the first few rows of the DataFrame
print(train_data.head())
```

```
       id  sentiment                                             review
0  5814_8          1  With all this stuff going down at the moment w...
1  2381_9          1  \The Classic War of the Worlds\" by Timothy Hi...
2  7759_3          0  The film starts with a manager (Nicholas Bell)...
3  3630_4          0  It must be assumed that those who praised this...
4  9495_8          1  Superbly trashy and wondrously unpretentious 8...
```

In [3]:
```
# 2. Counting positive and negative reviews
positive_reviews = train_data[train_data['sentiment'] == 1]
negative_reviews = train_data[train_data['sentiment'] == 0]
print("Number of positive reviews:", len(positive_reviews))
print("Number of negative reviews:", len(negative_reviews))
```

```
Number of positive reviews: 12500
Number of negative reviews: 12500
```

In [4]:
```
# 3. Using TextBlob to classify each movie review
def classify_sentiment_textblob(review):
    analysis = TextBlob(review)
    if analysis.sentiment.polarity >= 0:
        return 1
    else:
        return 0

train_data['predicted_sentiment_textblob'] = train_data['review'].apply(classify_se
```

In [5]:
```
# 4. Checking accuracy of the TextBlob model
textblob_accuracy = accuracy_score(train_data['sentiment'], train_data['predicted_s
print("Accuracy of TextBlob model:", textblob_accuracy)
```

```
Accuracy of TextBlob model: 0.68524
```

In [6]:
```
# 5. Using VADER sentiment analyzer
from nltk.sentiment.vader import SentimentIntensityAnalyzer

import nltk
nltk.download('vader_lexicon')

def classify_sentiment_vader(review):
    analyzer = SentimentIntensityAnalyzer()
    compound_score = analyzer.polarity_scores(review)['compound']
    if compound_score >= 0:
        return 1
    else:
        return 0

train_data['predicted_sentiment_vader'] = train_data['review'].apply(classify_senti

vader_accuracy = accuracy_score(train_data['sentiment'], train_data['predicted_sent
print("Accuracy of VADER model:", vader_accuracy)
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\MariaStella\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
Accuracy of VADER model: 0.69356
```

In [7]:
```
# Part 2
# Prepping Text for a Custom Model
# 2.1 Convert all text to lowercase letters
train_data['review'] = train_data['review'].apply(lambda x: x.lower())
```

In [8]:
```
# 2.2 Remove punctuation and special characters from the text
train_data['review'] = train_data['review'].apply(lambda x: x.translate(str.maketra
```

In [9]:
```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\MariaStella\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[9]:
```
True
```

In [10]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\MariaStella\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[10]:
```
True
```

In [11]:
```
# 2.3 Remove stop words
stop_words = set(stopwords.words('english'))
train_data['review'] = train_data['review'].apply(lambda x: ' '.join([word for word
```

In [12]:
```
# 2.4 Apply NLTK's PorterStemmer
stemmer = PorterStemmer()
train_data['review'] = train_data['review'].apply(lambda x: ' '.join([stemmer.stem(
```

In [13]:
```
# 2.4.1 Create a bag-of-words matrix
count_vectorizer = CountVectorizer()
bag_of_words_matrix = count_vectorizer.fit_transform(train_data['review'])
```

In [14]:
```
# 2.4.2 Display dimensions of bag-of-words matrix
print("Dimensions of bag-of-words matrix:", bag_of_words_matrix.shape)
```

```
Dimensions of bag-of-words matrix: (25000, 92345)
```

In [15]:
```
# 2.4.3 Create a term frequency-inverse document frequency (tf-idf) matrix
tfidf_transformer = TfidfTransformer()
tfidf_matrix = tfidf_transformer.fit_transform(bag_of_words_matrix)
```

In [16]:
```
# 2.4.4 Display dimensions of tf-idf matrix
print("Dimensions of tf-idf matrix:", tfidf_matrix.shape)
```

```
Dimensions of tf-idf matrix: (25000, 92345)
```

In [ ]:

Short Report:

Topic: Sentiment Analysis of Movie Reviews

Summary: This report presents the results of a sentiment analysis conducted on a dataset of movie reviews obtained from Kaggle. The analysis aimed to classify the sentiment of the reviews as either positive or negative using two different prebuilt sentiment analysis tools, namely TextBlob and VADER. Additionally, the text data was preprocessed to prepare it for custom model building, including steps such as lowercase conversion, punctuation removal, stop word removal, and stemming. The report concludes with a discussion on the results, implications, and potential future directions.

Introduction: In today's digital age, online reviews play a crucial role in shaping consumer perceptions and decision-making processes. Sentiment analysis, a subfield of natural language processing (NLP), offers a systematic approach to extract and quantify sentiments expressed in textual data. Understanding the sentiment of movie reviews can provide valuable insights into audience reactions, critical acclaim, and commercial success.

Statement of the Problem: The objective of this study is to perform sentiment analysis on a dataset of movie reviews and evaluate the effectiveness of two prebuilt sentiment analysis tools, TextBlob and VADER. Additionally, the text data will be preprocessed to prepare it for custom model building, enabling further analysis and exploration.

Methodology: Data Collection: The movie review dataset was obtained from Kaggle's "Bag of Words Meets Bags of Popcorn" competition using the Kaggle API. Data Preprocessing: Conversion of text to lowercase. Removal of punctuation and special characters. Elimination of stop words. Application of NLTK's PorterStemmer for word stemming. Sentiment Analysis: Utilization of TextBlob and VADER for sentiment classification. TextBlob classified reviews with polarity scores >= 0 as positive and < 0 as negative. VADER assigned positive sentiment to reviews with compound scores >= 0 and negative sentiment otherwise. Model Evaluation: Calculation of accuracy scores for TextBlob and VADER models. Custom Model Preprocessing: Creation of bag-of-words and tf-idf matrices from preprocessed text data.

Results: Data Overview: The dataset consists of movie reviews with 12,500 positive and 12,500 negative reviews. There are a total of 25,000 reviews in the dataset, evenly split between positive and negative sentiments (12,500 each).

Sentiment Analysis Results: TextBlob Model Accuracy: 68.524% VADER Model Accuracy: 69.356% Custom Model Preprocessing:

The preprocessing steps included converting text to lowercase, removing punctuation and special characters, eliminating stop words, and applying stemming using NLTK's PorterStemmer.

The dimensions of the bag-of-words matrix and the tf-idf matrix are both (25,000, 92,345), indicating 25,000 documents (reviews) and 92,345 unique words in the vocabulary. These results provide insights into the performance of sentiment analysis models and the preprocessing steps necessary for text data preparation. Further analysis and model refinement can be conducted based on these findings to improve sentiment classification accuracy and explore additional features for enhancing model performance.

Discussion and conclusions:

Model Performance:

Both TextBlob and VADER sentiment analysis models demonstrated reasonable accuracy in classifying movie review sentiments, with TextBlob achieving an accuracy of 68.524% and VADER achieving 69.356%.

While VADER marginally outperformed TextBlob in this dataset, both models showed effectiveness in sentiment classification, indicating their utility in analyzing movie reviews. Data Distribution:

The dataset comprises 25,000 movie reviews, evenly split between positive and negative sentiments, with 12,500 reviews in each category. This balanced distribution ensures that the dataset is representative and can provide reliable insights into sentiment patterns. Preprocessing Importance:

Preprocessing steps such as converting text to lowercase, removing punctuation, eliminating stop words, and applying stemming were crucial in preparing the text data for sentiment analysis. These steps helped in standardizing the input data and reducing noise, thereby improving the performance of sentiment analysis models. Feature Representation:

The bag-of-words and tf-idf matrices, with dimensions (25,000, 92,345), represented the preprocessed text data and provided a rich feature space for modeling. These matrices captured the frequency and importance of words in each document, enabling effective sentiment analysis. Future Directions:

Further exploration could involve refining sentiment analysis models to improve accuracy and account for nuanced expressions in movie reviews.

Additionally, incorporating additional features such as review length, word frequency, or genre-specific analysis could enhance sentiment classification accuracy and provide deeper insights into audience reactions. Deployment of sentiment analysis models in real-world

applications, such as recommendation systems or market research, could offer valuable insights for stakeholders in the movie industry.

In conclusion, the sentiment analysis of movie reviews using TextBlob, VADER, and custom preprocessing techniques proved effective in classifying sentiments and extracting meaningful insights. These findings lay the groundwork for further research and application of sentiment analysis techniques in understanding audience reactions and enhancing decision-making processes in the movie industry.

Way Forward:

Moving forward, it is imperative to focus on refining sentiment analysis models like TextBlob and VADER to enhance accuracy and robustness, while also exploring additional features such as review length and sentiment intensity to enrich the analysis process. Custom model development using machine learning algorithms and deployment in real-world applications like recommendation systems and market research can provide valuable insights for stakeholders in the movie industry. Continuous learning, adaptation, and collaboration with domain experts will be key in driving iterative improvements and staying abreast of advancements in natural language processing and sentiment analysis techniques, ultimately empowering stakeholders to make informed decisions and better understand audience sentiments.

In [ ]: