Artificial Intelligence

(UCS411)

LAB PROJECT SUBMISSION

Sentiment Analysis
Or
Opinion Mining

Submitted To:

Ms. Navpreet Kaur

Submitted By:

Uday Sharma (102103381)

Somya Mathur (102103385)

Simarjeet Singh (102153026)

Harsh Goel (102153042)

Problem Statement:

To segregate reviews and other text blocks based upon their opinion and Sentiments.

Description of Problem:

Sentiment analysis is the process of classifying whether a block of text is positive, negative, or, neutral. The goal which Sentiment analysis tries to gain is to be analysed people's opinions in a way that can help businesses expand. It focuses not only on polarity (positive, negative & neutral) but also on emotions (happy, sad, angry, etc.). It uses various Natural Language Processing algorithms such as Rule-based, Automatic, and Hybrid.

Challenges Faced:

Misspelled or misused words can create problems for text analysis. Autocorrect and grammar correction applications can handle common mistakes, but don't always understand the writer's intention.

With spoken language, mispronunciations, different accents, stutters, etc., can be difficult for a machine to understand. However, as language databases grow and smart assistants are trained by their individual users, these issues can be minimized.

Libraries Used:

1. Pyplot is an API (Application Programming Interface) for Python's matplotlib that effectively makes matplotlib a viable open source alternative to MATLAB. Matplotlib is a library for data visualization, typically in the form of plots, graphs and charts.

2. Natural language processing (NLP) refers to the branch of computer science—and more specifically, the branch of artificial intelligence or AI—concerned with giving computers the ability to understand text and spoken words in much the same way human beings can.
   NLP combines computational linguistics—rule-based modelling of human language—with statistical, machine learning, and deep learning models. Together, these technologies enable computers to process human language in the form of text or voice data and to 'understand' its full meaning, complete with the speaker or writer's intent and sentiment.

3. The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language.

```
import matplotlib.pyplot as plt
from nltk.corpus import stopwords
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
```

a. NLTK Library

Steps Involved:

1. Opening Files

Open the file to be analysed. Convert all the text to lower case and remove punctuations.

```
text = open('read.txt', encoding='utf-8').read()
lower_case = text.lower()
cleaned_text = lower_case.translate(str.maketrans('', '', string.punctuation))
```

---

2. . Sentence Tokenization

Sentence tokenization (also called **sentence segmentation**) is the problem of **dividing a string** of written language **into** its component **sentences**. The idea here looks very simple. In English and some other languages, we can split apart the sentences whenever we see a punctuation mark.

```
tokenized_words = word_tokenize(cleaned_text, "english")
```

---

3. Remove the Stop Words

Stop words are words which are **filtered out** before or after processing of text. When applying machine learning to text, these words can add a lot of **noise**. That's why we want to remove these **irrelevant words**.

Stop words **usually** refer to the **most common words** such as "**and**", "**the**", "**a**" in a language, but there is **no single universal list** of stopwords. The list of the stop words can change depending on your application.

```
final_words = []
for word in tokenized_words:
    if word not in stopwords.words('english'):
        final_words.append(word)
```

---

4. Lemmatization

Lemmatization is a text pre-processing technique used in natural language processing (NLP) models to break a word down to its root meaning to identify similarities. For example, a lemmatization algorithm would reduce the word *better* to its root word, or lemme, *good*.

```
lemma_words = []
for word in final_words:
    word = WordNetLemmatizer().lemmatize(word)
    lemma_words.append(word)
```

---

5. Emotion List

Open the ‘emotions.txt’ file containing a list of emotions and their lemmas as read only. Any word or its lemma found in text block given appends it’s emotion to a list being maintained. Also a counter keeps track of repetitive emotions.

```
emotion_list = []
with open('emotions.txt', 'r') as file:
    for line in file:
        clear_line = line.replace("\n", '').replace(",", '').replace("'", '').strip()
        word, emotion = clear_line.split(':')

        if word in lemma_words:
            emotion_list.append(emotion)
```

---

6. Calculating Sentiment

Based Upon the Count calculated above, the text block is categorized as Positive or Negative or Neutral Sentiment.

```
def sentiment_analyse(sentiment_text):
    score = SentimentIntensityAnalyzer().polarity_scores(sentiment_text)
    print(score)
    if score['neg'] > score['pos']:
        print("Negative Sentiment")
    elif score['neg'] < score['pos']:
        print("Positive Sentiment")
    else:
        print("Neutral Sentiment")
```

---

7. Printing Graph

Pyplot Library plots graph between Emotion and their count so as to increase readeblity of user to categorized text.

```
sentiment_analyse(cleaned_text)

fig, ax1 = plt.subplots()
ax1.bar(w.keys(), w.values())
fig.autofmt_xdate()
plt.savefig('graph.png')
plt.show()
```

Result of Sample Read File:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
PS C:\Users\somya\Desktop\AI Project> python -u "c:\Users\somya\Desktop\AI Project\Sentiment_Analysis.py"
[' happy', ' happy', ' attached', ' happy', ' attracted', ' alone', ' free', ' hated', ' happy', ' entitled', ' happy', ' loved', ' hat
ed', ' entitled']
Counter({' happy': 5, ' hated': 2, ' entitled': 2, ' attached': 1, ' attracted': 1, ' alone': 1, ' free': 1, ' loved': 1})
{'neg': 0.091, 'neu': 0.747, 'pos': 0.162, 'compound': 0.9996}
Positive Sentiment
```