*Sentiment analysis*

Natural Language Processing Dr. Esraa Afify

Eng. Mostafa Gamal

Members: -

Amr Tamer 205046

Nabil Tamer 205079

Moaz Gaafar 205087

Abstract: -

The abstract summarizes the sentiment analysis NLP project using Python. It provides a concise overview of the project, including its purpose, methodology, and key findings. The abstract briefly explains the significance of sentiment analysis and NLP in various domains, such as social media monitoring, customer feedback analysis, and market research. It also highlights the specific approach taken in the project and the results obtained. The abstract aims to provide a high - level understanding of the project to the readers.

Introduction: -

The introduction section provides an overview of the sentiment analysis NLP project using Python. It introduces the concept of sentiment analysis and its importance in understanding people's opinions and emotions from textual data. The section also highlights the relevance of natural language processing (NLP) techniques in sentiment analysis and discusses the goals and objectives of the project.

Sentiment analysis is a classification task in the area of natural language processing. Sometimes called ‘opinion mining,’ sentiment analysis models transform the opinions found in written language or speech data into actionable insights. For many developers new to machine learning, it is one of the first tasks that they try to solve in the area of NLP. This is because it is conceptually simple and useful, and classical and deep learning solutions already exist.

Sentiment analysis involves determining whether the author or speaker’s feelings are positive, neutral, or negative about a given topic. For instance, you would like to gain a deeper insight into customer sentiment, so you begin looking at customer feedback under purchased products or comments under your company’s post on any social media platform. You would like to know if the customer is pleased with your services, neutral, or if he/she has any complaints, meaning whether the customer has a neutral, positive, or negative sentiment regarding your products, services or actions.

Emotion detection assigns independent emotional values, rather than discrete, numerical values. It leaves more room for interpretation, and accounts for more complex customer responses compared to a scale from negative to positive. For instance, even though sadness and anger are negative emotions, they do have different connotations; so, distinguishing between them can give more precise information about the customer's interactions with your products and a better insight into areas for improvement.

we can use it to examine whether your customer’s feedback in online reviews about your products or services is positive, negative, or neutral. You can also rate this feedback using a grading system, you can investigate their opinions about particular aspects of your products or services, and you can infer their intentions or emotions.

Methodology: -

Importing Libraries: The code begins by importing necessary libraries such as string, Counter from collections, matplotlib.pyplot, and various modules from the Natural Language Toolkit (nltk).

Downloading NLTK Resources: The code uses nltk.download() to download required resources such as tokenizers, stopwords, WordNet, and the VADER sentiment lexicon.

Reading and Cleaning the Text: The code reads the contents of a file named 'read2.txt' and stores it in the text variable.

It then converts the text to lowercase and removes punctuation using the translate() method.

Tokenization and Stopword Removal: The code tokenizes the cleaned text into individual words using the word\_tokenize() function. It removes stopwords (commonly occurring words with little semantic meaning) using the stopwords.words('english') list from NLTK.

Lemmatization: The code performs lemmatization on the remaining words using the WordNetLemmatizer from NLTK, which reduces words to their base form (e.g., plural to singular, different verb tenses to base form).

Identifying Emotions: The code reads a file named 'Feelings.txt' which contains a list of words associated with specific emotions. It matches the lemmatized words with the words in the file and appends the corresponding emotion to the Feels\_list.

Sentiment Analysis: The code defines a function sentiment\_analyse() that uses the VADER (Valence Aware Dictionary and sentiment Reasoner) sentiment analyzer from NLTK to analyze the sentiment of the cleaned text.

It calculates a sentiment score and prints the sentiment as "Negative Sentiment," "Positive Sentiment," or "Neutral Sentiment" based on the score.

Counting Emotions: The code uses the Counter() function to count the occurrences of each emotion in the Feels\_list and stores the results in the (w) variable.

Visualization: The code plots a bar graph using matplotlib.pyplot to visualize the frequency of different emotions. The graph is saved as 'graph.png' and displayed using plt.show().