CELEBAL TECHNOLOGIES

Galgotias University

Task 3: Twitter Sentiment Analysis

Code by Ansh Shankar

Question: Sentiment Analysis using NLP Solution:

1. Introduction:

In the age of social media and online communication, Twitter has emerged as a powerful platform for people to share their thoughts, opinions, and emotions. The vast amount of textual data generated on Twitter presents a unique opportunity to gain valuable insights into public sentiment and attitudes. Sentiment analysis, a subfield of natural language processing, aims to automatically determine the sentiment expressed in a piece of text, such as whether it conveys positive, negative, or neutral emotions.

In this project, I dived into the fascinating world of sentiment analysis by analyzing a dataset of tweets and building a predictive model to classify them as either positive or negative. My objective is to develop a robust system that can accurately discern the sentiment behind each tweet, allowing us to understand public perception and emotional reactions on Twitter.

2. Dataset:

In this sentiment analysis project, I explored the "Twitter Sentiment Analysis" dataset from Kaggle, comprising over 31,000 labeled tweets expressing positive or negative sentiments. Utilizing a systematic methodology, preprocessed the data, extract features using TF-IDF representation, and train a Naive Bayes classifier. The model's performance is evaluated using various metrics, showcasing its ability to accurately classify sentiments. The project's insights have significant implications for understanding public opinion and emotions on Twitter, with potential applications in brand sentiment analysis and real-time social media monitoring. The extensive dataset's relevance and impact propel our exploration of sentiment analysis techniques in the realm of natural language processing.

3. Methodology:

a. Exploring the dataset:

The dataset used for this sentiment analysis project is rich in content, containing an impressive count of over 29,000 positive tweets and more than 3,000 negative tweets.

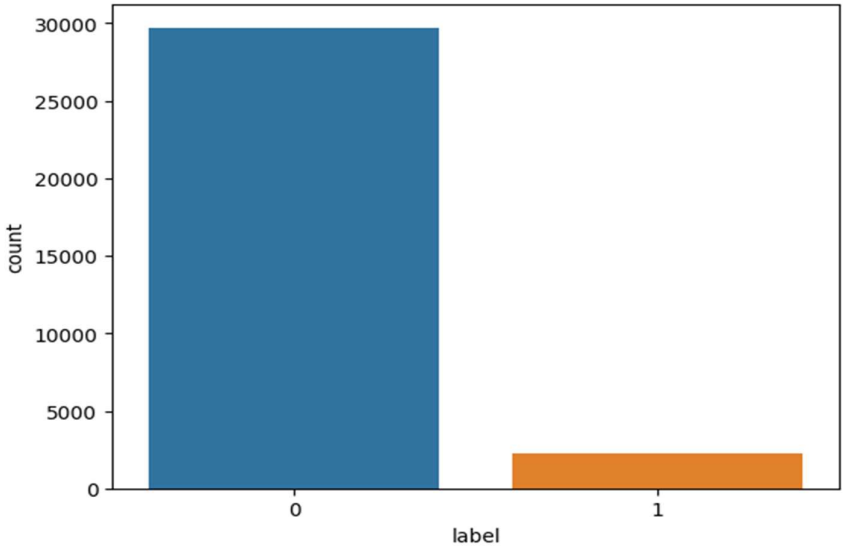

Figure 1: Split of tweets in data

For the positive word cloud image, we visualize a captivating mosaic of optimistic and uplifting words, resonating with sentiments of joy, satisfaction, and enthusiasm. The word cloud beautifully captures the frequency and importance of positive expressions, offering a glimpse into the prevailing optimism within the tweets.

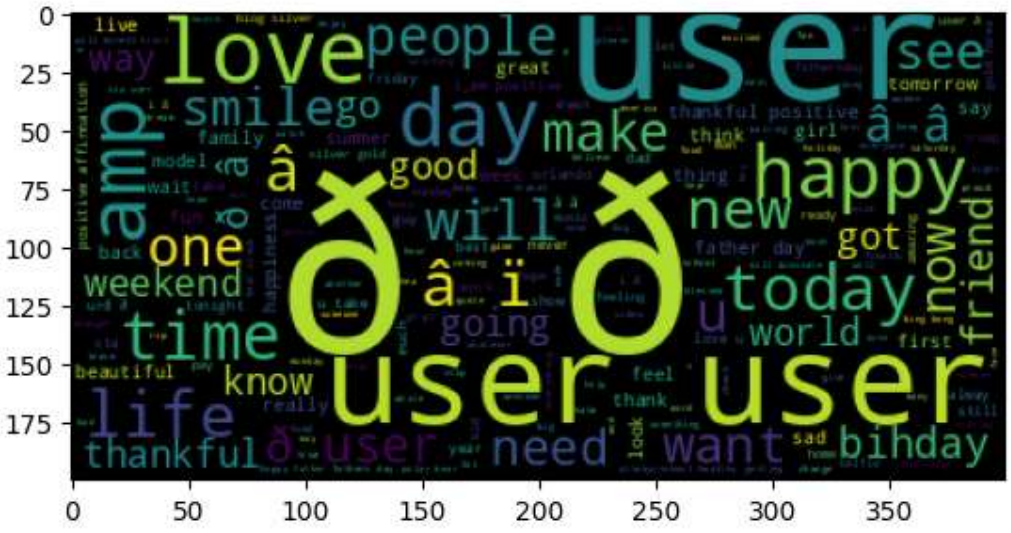

Figure 2: Positive Wordcloud

On the other hand, the negative word cloud paints a contrasting picture, showcasing an array of poignant and emotive words, reflecting sentiments of dissatisfaction, concern, and disappointment. The cloud of negative expressions provides valuable insights into the prevailing discontent and emotional distress present in the dataset.

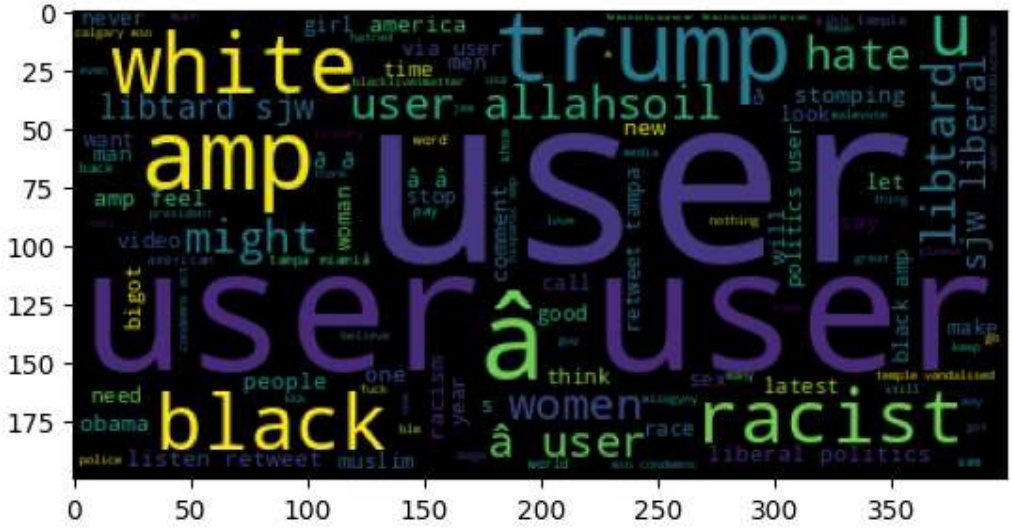

Figure 3: Negative wordcloud

b. Preprocessing Techniques:

We initiate our analysis by preparing the raw data. This preprocessing phase includes tasks such as removing punctuation, converting text to lowercase, and eliminating common stopwords. These steps enable us to focus on the essential words in the tweets, disregarding noise and irrelevant information.

```
def message_cleaning(message):
    test_punc_removed = [char for char in message if char not in string.punctuation]
    test_punc_removed = ''.join(test_punc_removed)
    test_punc_st_removed = []
    for char in test_punc_removed.split():
        if char.lower() not in st:
            test_punc_st_removed.append(char)
    test_punc_st_removed = " ".join(test_punc_st_removed)
    return test_punc_st_removed
```

Figure 4: Preprocessing data

c. Feature Extraction:

To transform the textual data into a format suitable for machine learning algorithms, we utilize the widely acclaimed Term Frequency-Inverse Document Frequency (TF-IDF) representation. By adopting TF-IDF, each tweet is represented as a numerical vector, capturing the significance of words in the context of the entire dataset.

TF-IDF vectorizer object was created with a maximum of 5000 features (words) and uses English stop words to preprocess the text. The fit_transform method is then applied to calculate the TF-IDF values for the tweet data, and the result is stored in the variable tweets_tfidfvectorizer as a dense NumPy array. This TF-IDF representation allows the tweets to be converted into numerical vectors, making it easier for further analysis and machine learning tasks.

d. **Model Selection and Training Strategies**:

For sentiment classification, we selected the Multinomial Naive Bayes classifier due to its simplicity and effectiveness with text data. This classifier is particularly suitable for tasks involving discrete features like word counts, making it a strong choice for text classification tasks. The dataset was divided into training and testing sets, with 70% of the data used for training and 30% for testing. This partitioning allowed us to train the Naive Bayes classifier on one segment of data and assess its performance on unseen data during evaluation. The training process involved fitting the classifier to the training set, and then we evaluated its performance on the test set to gauge its accuracy and effectiveness for sentiment analysis.

4. Results:

The model achieved an impressive accuracy of 95%. The classification matrix reveals valuable insights into its performance. For class 0, the model displayed high precision (95%) and recall (100%), resulting in a robust F1-score of 0.97. The support for class 0 was 8905 samples. However, for class 1, the model's precision was 93%, indicating a reasonable proportion of true positive predictions, but the recall was relatively low at 32%. Consequently, the F1-score for class 1 was 0.48. The support for class 1 was 684 samples. The macro-averaged F1-score across all classes was 0.73, with a weighted average of 0.94. Overall, the model performed well in identifying negative instances (class 0) but struggled in accurately classifying positive instances (class 1).

	precision	recall	f1-score	support
0	0.95	1.00	0.97	8905
1	0.93	0.32	0.48	684
accuracy			0.95	9589
macro avg	0.94	0.66	0.73	9589
weighted avg	0.95	0.95	0.94	9589

Figure 5: Classification matrix

The model correctly predicted 8889 samples as belonging to the negative class (True Negatives) and 221 samples as belonging to the positive class (True Positives). This indicates instances where the model accurately identified both the negative and positive cases.

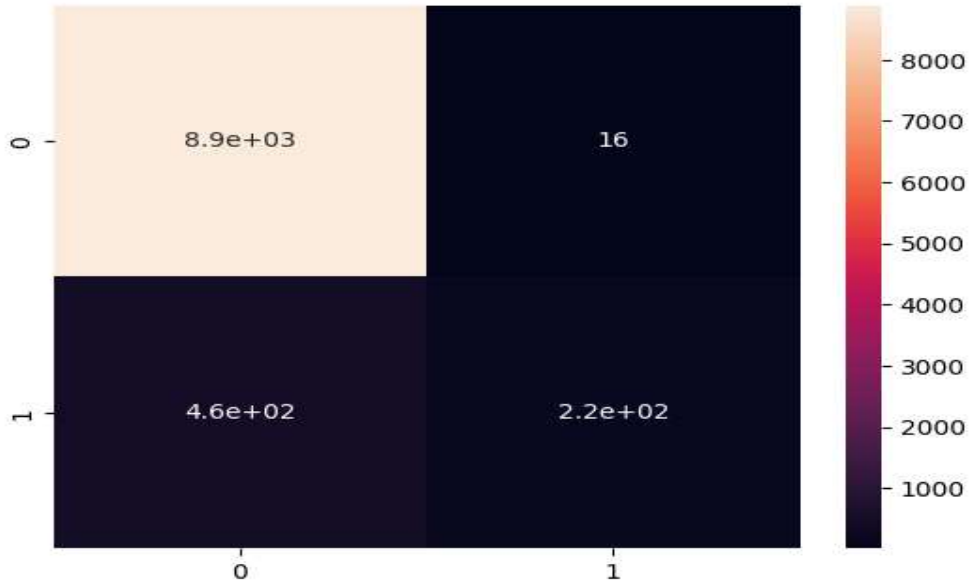

Figure 6: Confusion matrix

5. Discussion:

The results of the project are encouraging, with the chosen Naive Bayes classifier achieving an accuracy of 95%. This indicates that the model can effectively classify sentiment in the given dataset. The high precision (95%) and recall (100%) for class 0 demonstrate the model's proficiency in identifying negative sentiments. However, the relatively low recall (32%) for class 1, representing positive sentiments, indicates that the model struggles in accurately classifying positive instances. This could be due to class imbalance or the complexity of identifying positive sentiment from the text data.

Challenges Faced:

During the assignment, several challenges were encountered. One major challenge was dealing with class imbalance, where the positive class had significantly fewer samples compared to the negative class. This imbalance could lead to biased predictions and affect the model's performance. Additionally, the choice of stop words and the maximum number of features for the TF-IDF vectorization could have influenced the model's accuracy and generalization capabilities. Fine-tuning these hyperparameters required careful consideration to optimize the model's performance.

Proposed Improvements and Future Directions:

To address the challenge of class imbalance, employing techniques such as oversampling the positive class (e.g., SMOTE) or using different evaluation metrics (e.g., area under the ROC curve) can lead to a more comprehensive analysis of the model's performance. Additionally, experimenting with other advanced classifiers, such as Support Vector Machines (SVM) or deep learning models like LSTM or BERT, may enhance the accuracy and generalization capabilities, particularly for the positive sentiment classification.

6. Conclusion:

In conclusion, this project successfully performed sentiment analysis on tweets using a Naive Bayes classifier. The accuracy achieved was 95%. The report discussed the dataset, methodology, results, model strengths and weaknesses, challenges faced, and proposed future improvements. Sentiment analysis can be a valuable tool in understanding public opinion, and with further refinement, the model's accuracy and performance could be improved to better serve real-world applications.

Appendix:

Source code: https://github.com/anshshankar/Celebal-task/tree/main/Task-3

Video demonstration: Folder - Google Drive