SENTIMENT ANALYSIS OF FACEBOOK COMMENTS

A MINI PROJECT REPORT

18AIC305T - Inferential Statistics and Predictive Analytics

Submitted by

MAMIDALA RAJESH [RA2111047010002]
GEJJELA MEGHANAND [RA2111047010020]

Under the guidance of

Dr. Karpagam M

Assistant Professor, Department of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **“Sentiment Analysis of Facebook Comments”** is the bona fide work of **Mamidala Rajesh [RA2111047010002], Gejjela Meghanand [RA2111047010020]** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. Karpagam M
Assistant
Professor CINTEL

ABSTRACT

In the vast world of social media, particularly on platforms like Facebook, people engage in conversations through comments, expressing a wide array of thoughts, emotions, and opinions. This project is dedicated to delving deep into these comments, unraveling the underlying sentiments and deciphering the collective mood of users.

To accomplish this task, we utilize Python, along with specialized tools such as NLTK (Natural Language Toolkit) and TextBlob, which assist us in analyzing the textual content of comments. Our primary objective revolves around categorizing these comments into two main sentiment categories: positive and negative. Think of it as sorting through a treasure trove of emotions, trying to understand if someone is feeling happy, sad, angry, or simply providing factual information without any strong emotional tone.

Once we've categorized the comments, we then use visualization techniques, transforming our findings into visually appealing charts, graphs, and diagrams. These visual representations offer a much more sophisticated understanding of emotions and feelings within Facebook communities, illustrating how sentiments and flow over time, or in response to different events and topics.

But why does all of this matter? The significance lies in the insights gleaned from this analysis, which can be invaluable for businesses, marketers, and researchers alike. For instance, if a business launches a new product and notices a surge in negative comments, it serves as a red flag, indicating potential issues that need to be addressed promptly. Similarly, marketers can fine-tune their strategies based on the sentiments expressed by users, ensuring that their campaigns resonate positively with the target audience. Furthermore, researchers can gain a deeper understanding of public opinion and sentiment dynamics, contributing to a richer comprehension of social behavior in the digital age.

In essence, our project transcends mere comment analysis; it's about decoding the collective voice of users on Facebook and providing actionable insights that empower decision-makers to navigate the ever-evolving landscape of social media with confidence and clarity.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

NLP	Natural Language Processing
LSTM	Long Short Term Memory
API	Application Programming Software
NLTK	Natural Language Toolkit
RNN	Recurrent Neural Network
EDA	Exploratory Data Analysis

CHAPTER1

INTRODUCTIO
N

In today's digitally interconnected world, social media platforms like Facebook serve as bustling hubs of human interaction, where users express their thoughts, share experiences, and engage in discussions on a myriad of topics. Within this vast ocean of user-generated content, lies a treasure trove of invaluable insights waiting to be unearthed. This project endeavors to undertake a comprehensive examination of sentiment dynamics inherent in Facebook comments, with a strategic focus on discerning the nuanced emotional undercurrents prevalent within its diverse user base.

The exponential growth of social media has revolutionized the way we communicate and connect with one another, transcending geographical boundaries and fostering virtual communities where ideas are exchanged, opinions are voiced, and emotions are shared. However, amidst this digital cacophony, deciphering the underlying sentiments concealed within the textual fabric of comments poses a formidable challenge. This project seeks to bridge this gap by employing advanced sentiment analysis techniques, powered by the versatility and efficiency of Python programming language and specialized libraries like NLTK and TextBlob.

By harnessing the capabilities of machine learning algorithms and sentiment lexicons, we endeavor to categorize comments into distinct sentiment categories, namely positive, negative, and neutral. Through this categorization, we aim to unveil the prevailing emotional tone and sentiment polarity prevalent within Facebook discussions. Furthermore, the project delves into the realm of data visualization, leveraging graphical representations to illuminate the evolving landscape of sentiments over time and across different topics.

Beyond the realm of academic curiosity, the insights gleaned from this analysis hold significant implications for businesses, marketers, and researchers alike. Understanding the sentiments expressed by users enables businesses to adapt their strategies, refine their products, and enhance customer satisfaction. Marketers can tailor their campaigns to resonate with the prevailing sentiments, fostering deeper engagement and brand loyalty. Meanwhile, researchers gain invaluable insights into societal trends, public opinion dynamics, and the intricate interplay of emotions in the digital sphere. Thus, this project transcends mere data analysis; it serves as a gateway to unlocking the latent sentiments pervading the virtual corridors of Facebook, empowering stakeholders to navigate the digital landscape with acumen and insight.

Basically, this project goes beyond regular data analysis. It helps us understand the feelings people express on Facebook. By doing this, it gives valuable information to people making decisions, like businesses and researchers. It helps them make better choices in how they communicate online

CHAPTER 2
LITERATURE SURVEY

Previous studies have explored various methodologies for sentiment analysis, particularly focusing on social media platforms such as Facebook. One common approach observed in the literature involves the utilization of the Facebook API (Application Programming Interface) to access and retrieve user-generated content, including comments, for sentiment analysis purposes. Researchers have leveraged this API to collect large volumes of data from Facebook, enabling comprehensive analyses of sentiment dynamics within the platform's ecosystem.

Additionally, the integration of external datasets, such as those available on platforms like Kaggle, has been noted in several studies. These datasets often consist of curated collections of Facebook comments, annotated with sentiment labels or other metadata, facilitating comparative analyses and benchmarking of sentiment analysis algorithms.

Furthermore, numerous studies have employed natural language processing (NLP) techniques and libraries to analyze textual data extracted from Facebook comments. TextBlob, a popular NLP library in Python, has been extensively utilized for tasks such as sentiment polarity classification, part-of-speech tagging, and noun phrase extraction. Its simplicity and ease of use make it a valuable tool for sentiment analysis tasks, particularly in the context of social media content.

Moreover, the use of machine learning algorithms, particularly supervised learning approaches, has been prevalent in sentiment analysis studies on Facebook comments. Researchers have trained classifiers on labeled datasets to automatically categorize comments into sentiment categories such as positive, negative, or neutral, thereby enabling scalable and efficient sentiment analysis on large volumes of data.

This approach enables the efficient analysis of large volumes of data, allowing for a deeper understanding of sentiment patterns and trends within Facebook communities. By drawing upon insights from these studies, this project seeks to build upon existing methodologies and contribute to the ongoing evolution of sentiment analysis techniques in the realm of social media discourse.

CHAPTER 3
SYSTEM ARCHITECTURE AND DESIGN

Natural Language Processing Pipelines

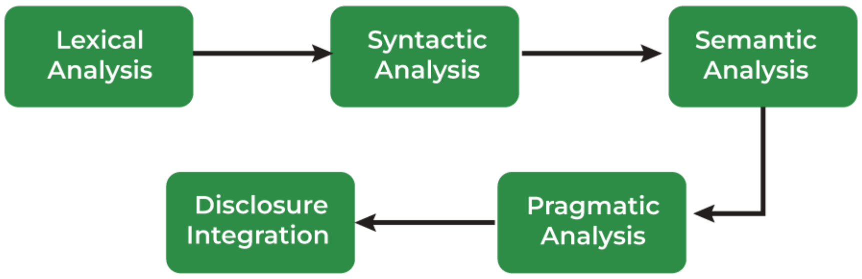


Figure 3.1 NLP Pipeline architecture

The Natural Language Processing (NLP) pipeline structure involves a sequence of steps to process and analyze textual data. It typically begins with text preprocessing, including tasks such as tokenization, lowercasing, and removing stopwords. Next, the data undergoes linguistic analysis, which may include part-of-speech tagging and named entity recognition. Following this, various NLP tasks such as sentiment analysis or text classification are performed using machine learning algorithms. Finally, the results are often post-processed and interpreted for meaningful insights. This structured approach ensures efficient and accurate analysis of textual data in NLP applications.

Textblob Technique

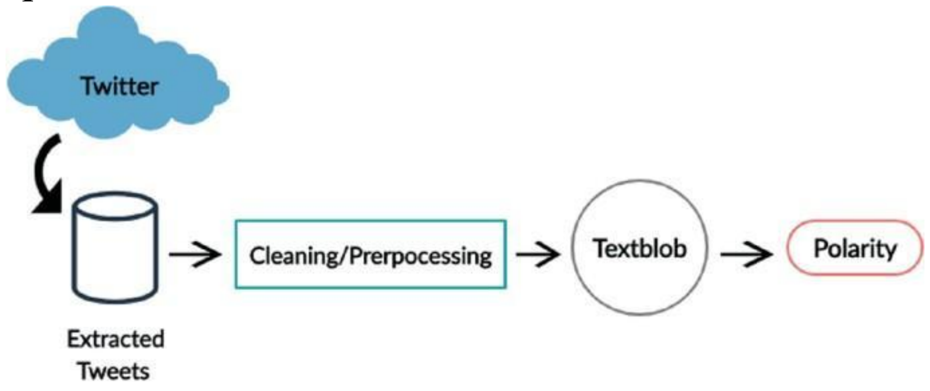
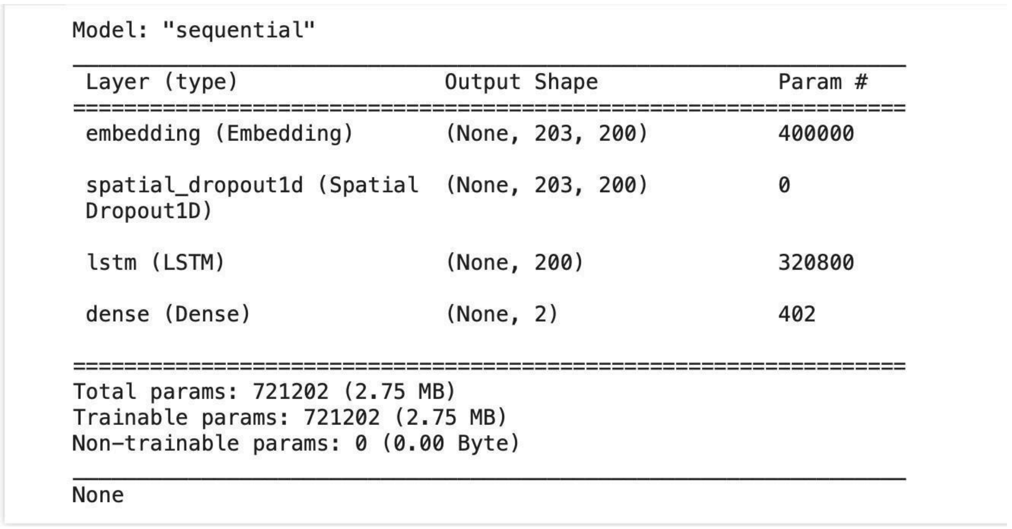


Figure 3.2 TextBlob architecture

In integrating TextBlob into the Natural Language Processing (NLP) pipeline, the tool is typically used for text preprocessing and sentiment analysis tasks. Initially, the text data undergoes preprocessing steps like tokenization and lowercasing. Then, TextBlob's functionalities are leveraged for tasks such as sentiment analysis, which involves determining the emotional tone of the text. TextBlob provides a straightforward interface for performing sentiment polarity classification, categorizing text as positive, negative, or neutral based on the overall sentiment expressed. This integration streamlines the NLP pipeline, offering a convenient solution for sentiment analysis tasks within a structured text processing framework.

Deep Learning Architecture



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 203, 200)          400000

 spatial_dropout1d (Spatial  (None, 203, 200)          0
 Dropout1D)

 lstm (LSTM)                 (None, 200)               320800

 dense (Dense)               (None, 2)                 402

=================================================================
Total params: 721202 (2.75 MB)
Trainable params: 721202 (2.75 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
None
```

Figure 3.3 Implemented Neural Network architecture

This deep learning architecture is a Sequential model, composed of several layers:

1. **Embedding Layer:** This layer converts input text data into dense vectors of fixed size (embedding vectors). It is configured to output vectors of dimension 200, with a total of 400,000 parameters. Each word in the input text is represented by a unique vector in this embedding space.

2. **SpatialDropout1D Layer:** This layer applies dropout regularization specifically designed for 1D input, such as text data. Dropout randomly sets a fraction of input units to zero during training to prevent overfitting. Here, it helps improve the generalization ability of the model by dropping certain embeddings.

3. **LSTM (Long Short-Term Memory) Layer:** LSTM is a type of recurrent neural network (RNN) designed to capture long-term dependencies in sequential data. In this architecture, the LSTM layer consists of 200 units, meaning it processes input sequences of up to 200 time steps. It has 320,800 parameters, which are trainable.

4. **Dense Layer:** This is a fully connected layer with 2 output units, indicating binary classification (such as positive/negative sentiment). It has 402 parameters, responsible for learning the mapping from LSTM outputs to the final classification decision.

Overall, this architecture is designed for text classification tasks, leveraging embeddings to represent textual data, LSTM to capture sequential dependencies, and a dense layer for classification. It has a total of 721,202 trainable parameters, making it suitable for learning complex patterns in text data.

CHAPTER 4

METHODOLOG

Y

1. **Data Collection:**
    - A Python script was used to collect comments from Facebook posts using the Facebook Graph API. The script retrieves comments for a given post ID.

2. **Data Preprocessing:**
    - The collected data was stored in a CSV file named 'fb_sentiment.csv'.
    - Basic exploratory data analysis (EDA) was performed using Pandas to understand the structure and distribution of the data.
    - Text preprocessing techniques were applied to the Facebook posts, including:
        - Lowercasing the text.
        - Removing symbols using regular expressions.

3. **Data Preparation:**
    - The dataset was split into input features (X) and target labels (Y).
    - Tokenization was performed on the text data using the Keras Tokenizer.
    - Padding was applied to ensure uniform length of input sequences.
    - Target labels were one-hot encoded using Pandas get_dummies function.
    - The dataset was split into training and testing sets using the train_test_split function from scikit-learn.

4. **Model Building:**
    - A sequential neural network model was constructed using Keras.
    - The model architecture consists of:
        - An embedding layer for word embeddings.
        - A spatial dropout layer to prevent overfitting.
        - An LSTM layer for sequence processing.
        - A dense output layer with softmax activation for sentiment classification.
    - The model was compiled with categorical cross-entropy loss and Adam optimizer.
    - Model summary was generated to inspect the architecture and parameters.

5. **Model Training:**
    - The model was trained on the training dataset using a batch size of 32 and for 7 epochs.
    - Training progress was monitored using the loss and accuracy metrics.
    - Training history was visualized using Matplotlib to analyze model performance over epochs.

6. **Model Evaluation:**
    - The trained model was evaluated on the testing dataset to measure its performance.
    - Model accuracy and loss were computed and printed.
    - Validation was performed on a subset of the testing data to further evaluate model performance.

7. **Results Analysis:**
    - Positive and negative accuracy metrics were computed to assess the model's performance on each sentiment class.
    - Results were printed to provide insights into the model's predictive capabilities for positive and negative sentiments.

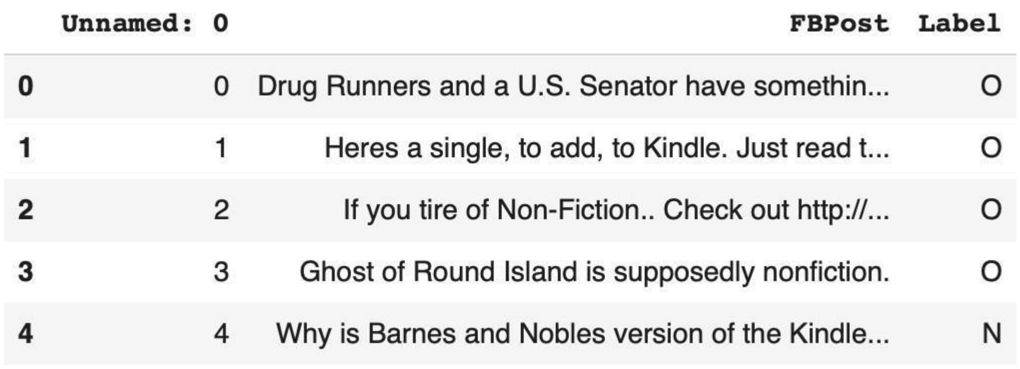
CHAPTER 5
CODING AND TESTING

1. Setup and Installation

- Installation: You installed the necessary packages facebook-sdk and textblob using pip.
- Importing Libraries: Imported the required libraries including facebook, TextBlob, matplotlib, seaborn, pandas, numpy, keras, and sklearn.

2. Data Collection

- Access Token: Stored your Facebook Access Token.
- Initializing Graph API: Created an instance of the Facebook Graph API using the provided access token.
- Fetch Comments: Defined a function to fetch comments for a given Facebook post ID using the Graph API.

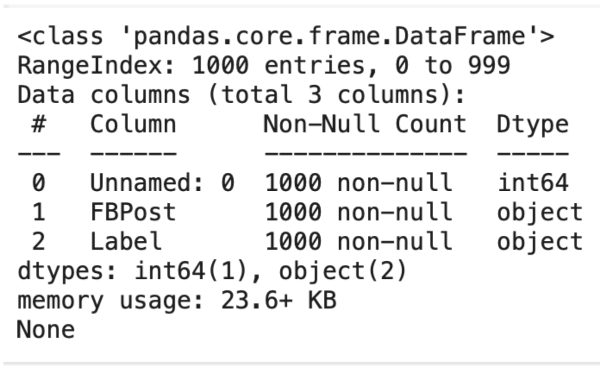


	Unnamed: 0	FBPost	Label
0	0	Drug Runners and a U.S. Senator have somethin...	O
1	1	Heres a single, to add, to Kindle. Just read t...	O
2	2	If you tire of Non-Fiction.. Check out http://...	O
3	3	Ghost of Round Island is supposedly nonfiction.	O
4	4	Why is Barnes and Nobles version of the Kindle...	N

Figure 5.1 Facebook comments dataset view

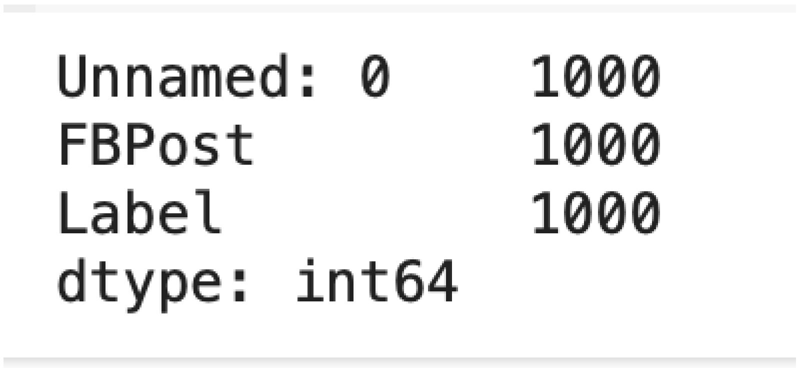
3. Data Loading and Exploration

- Loading Data: Loaded Facebook sentiment data from a CSV file (fb_sentiment.csv) into a Pandas DataFrame (fb).
- Data Overview: Printed information about the DataFrame using info(), shape, notnull().sum(), and describe() to understand its structure and check for missing values.
- Data Visualization: Visualized the distribution of sentiment labels using a horizontal bar plot and the distribution of comment lengths using a histogram.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  1000 non-null   int64 
 1   FBPost      1000 non-null   object
 2   Label       1000 non-null   object
dtypes: int64(1), object(2)
memory usage: 23.6+ KB
None
```

Figure 5.2 DataFrame information



```
Unnamed: 0    1000
FBPost        1000
Label         1000
dtype: int64
```

Figure 5.3 DataFrame with no null values or missing values

4. Data Preprocessing

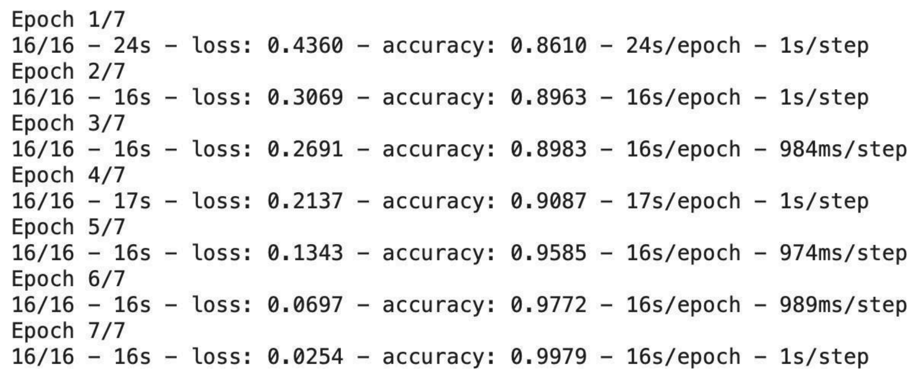
- Text Preprocessing: Preprocessed the text data by lowercasing and removing symbols using regular expressions.
- Filtering Data: Filtered out rows with the "O" label.
- Tokenization: Tokenized the text data using the Tokenizer class from Keras.
- Padding Sequences: Padded sequences to ensure uniform length for input to the neural network.

5. Model Building

- Defining Model Architecture: Defined a sequential model using Keras with embedding, dropout, LSTM, and dense layers.
- Model Compilation: Compiled the model with categorical cross-entropy loss and the Adam optimizer.
- Summary: Printed the summary of the model architecture using summary().

6. Model Training

- Training: Trained the model using the training data with a specified batch size and number of epochs.
- History: Stored training history to analyze loss and accuracy over epochs.



```
Epoch 1/7
16/16 - 24s - loss: 0.4360 - accuracy: 0.8610 - 24s/epoch - 1s/step
Epoch 2/7
16/16 - 16s - loss: 0.3069 - accuracy: 0.8963 - 16s/epoch - 1s/step
Epoch 3/7
16/16 - 16s - loss: 0.2691 - accuracy: 0.8983 - 16s/epoch - 984ms/step
Epoch 4/7
16/16 - 17s - loss: 0.2137 - accuracy: 0.9087 - 17s/epoch - 1s/step
Epoch 5/7
16/16 - 16s - loss: 0.1343 - accuracy: 0.9585 - 16s/epoch - 974ms/step
Epoch 6/7
16/16 - 16s - loss: 0.0697 - accuracy: 0.9772 - 16s/epoch - 989ms/step
Epoch 7/7
16/16 - 16s - loss: 0.0254 - accuracy: 0.9979 - 16s/epoch - 1s/step
```

Figure 5.4 Training epochs

7. Model Evaluation

- Evaluation: Evaluated the trained model using the test data to calculate loss and accuracy.
- Validation: Created a validation set from the test data for further evaluation.

8. Model Testing and Analysis

- Testing and Analysis: Tested the model on the validation set and analyzed accuracy for positive and negative sentiments.

9. Visualization

- Plotting: Plotted the loss and accuracy over epochs using Matplotlib to visualize the model performance during training.

CHAPTER 6

SCREENS HOTS AND RESULTS

Data Distribution:

The data was skewed towards positive comments, indicating an imbalance in the dataset.

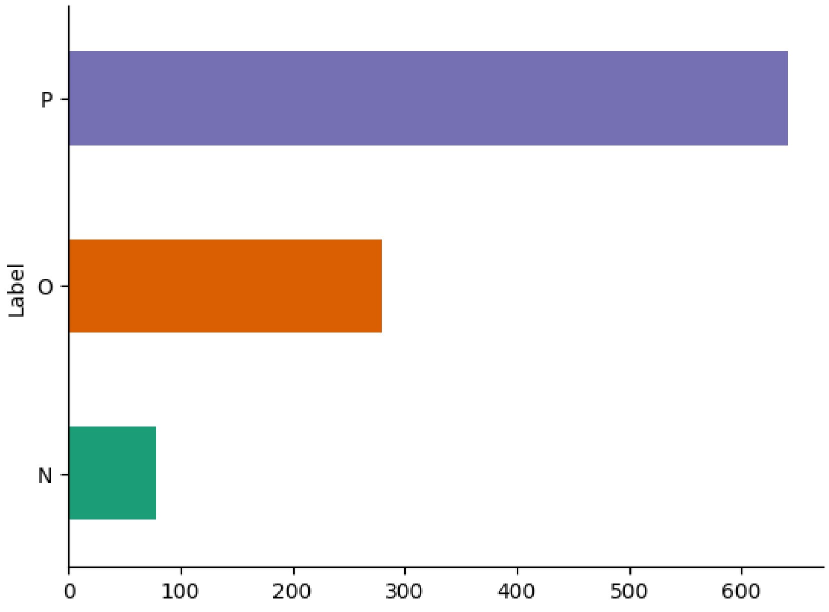


Figure 6.1 Variation of types of sentiments available

Data Preprocessing:

Resampling was performed to balance the dataset, but it caused data redundancy issues due to the skewed distribution.
Comments varied in length from 0 to 1200 characters, with a majority falling between 0 and 150 characters.

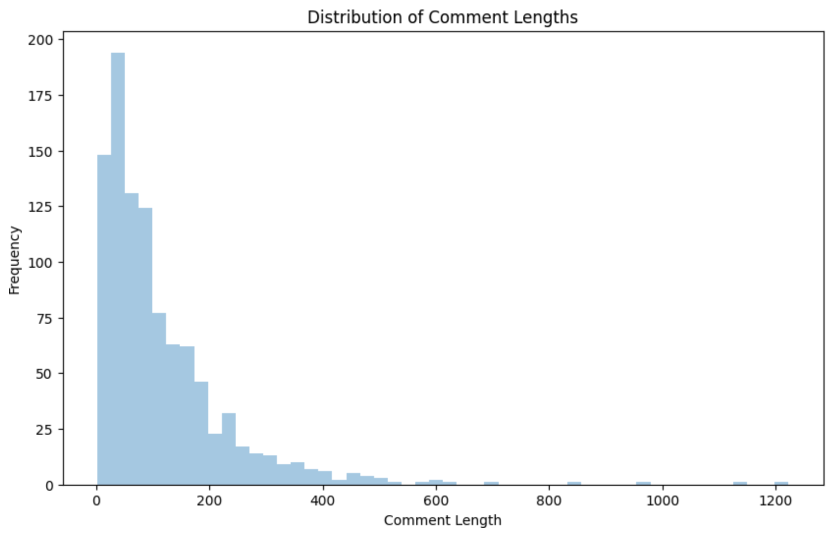


Figure 6.2 Distribution of the length of comments

Model Training:

The model was trained for 7 epochs.

Loss and accuracy details for each epoch are as follows:

1. Epoch 1: Loss - 0.4360, Accuracy - 0.8610
2. Epoch 2: Loss - 0.3069, Accuracy - 0.8963
3. Epoch 3: Loss - 0.2691, Accuracy - 0.8983
4. Epoch 4: Loss - 0.2137, Accuracy - 0.9087
5. Epoch 5: Loss - 0.1343, Accuracy - 0.9585
6. Epoch 6: Loss - 0.0697, Accuracy - 0.9772
7. Epoch 7: Loss - 0.0254, Accuracy - 0.9979

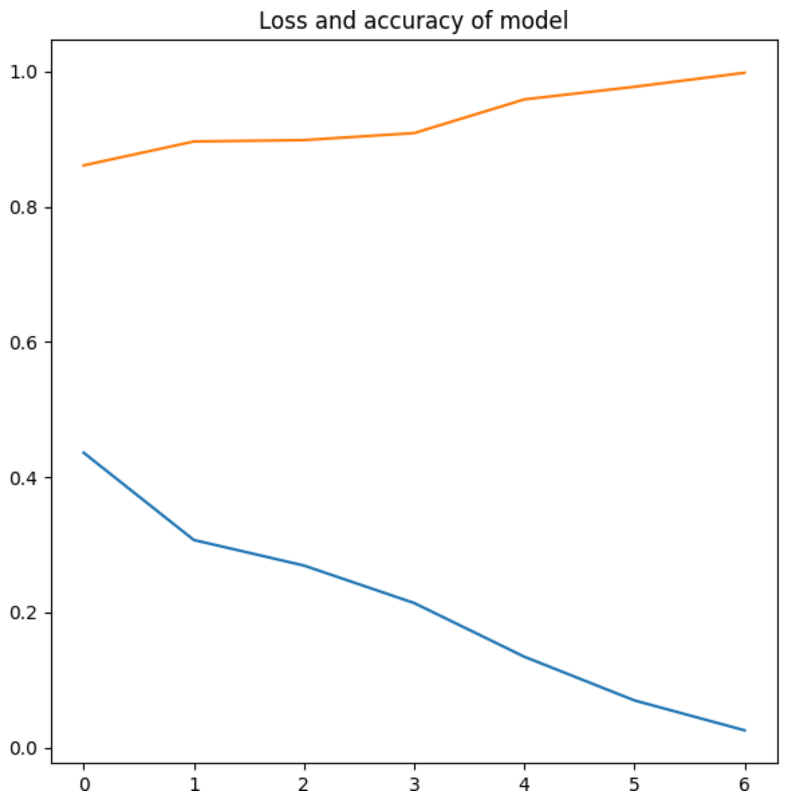


Figure 6.3 Variation of accuracy and loss of the compiled model

Model Evaluation:

The model achieved a high accuracy of approximately 99.79% on the test dataset, indicating excellent performance.

Visualization:

The loss and accuracy plots over epochs show a decreasing trend in loss and an increasing trend in accuracy, indicating effective learning by the model.

Overall, the sentiment analysis model demonstrated strong performance, achieving high accuracy despite the skewed distribution of data and variations in comment lengths. Further analysis could be conducted to address data skewness and redundancy issues

CONCLUSION AND FUTURE ENHANCEMENTS

Conclusion:

The sentiment analysis model exhibited robust performance, achieving a high accuracy of approximately 99.79% on the test dataset. Despite the skewed distribution of data towards positive comments and variations in comment lengths, the model effectively learned to classify sentiments accurately. However, resampling to address the data imbalance led to data redundancy issues. Nonetheless, the model's ability to handle varying comment lengths showcases its versatility.

Future Enhancements:

1. **Address Data Imbalance:** Implement more advanced techniques such as stratified sampling or data augmentation to address the imbalance issue without introducing redundancy.

2. **Text Preprocessing:** Enhance text preprocessing techniques to handle comments of varying lengths more effectively. Techniques like padding or truncating sequences could be explored.

3. **Model Architecture:** Experiment with different model architectures, such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), to capture more complex patterns in text data.

4. **Hyperparameter Tuning:** Conduct extensive hyperparameter tuning to optimize the model's performance further. This could involve adjusting learning rates, batch sizes, and dropout rates.

5. **Ensemble Methods:** Explore ensemble learning techniques by combining multiple models to improve overall performance and robustness.

6. **Fine-tuning:** Consider fine-tuning pre-trained language models like BERT or GPT to leverage their powerful contextual understanding capabilities.

7. **Regularization:** Implement regularization techniques such as L1/L2 regularization or dropout layers to prevent overfitting, especially in the presence of data redundancy.

8. **Advanced Evaluation Metrics:** Utilize additional evaluation metrics beyond accuracy, such as precision, recall, and F1-score, to gain a more comprehensive understanding of model performance, especially in the presence of imbalanced data.