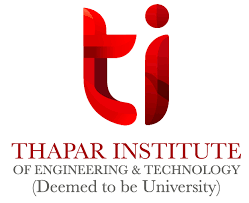
**WhatsApp Chat Sentiment Analysis**



Computer Science and Engineering Department

Thapar Institute of Engineering and Technology

(Deemed to be University), Patiala – 147004

**Machine Learning Project**

**Submitted By:**

Kanav Sehgal

102103635

Nimish Gupta

102103798

**Submitted To:**

Mrs. Jyoti Maggu

**1. Introduction:**

WhatsApp chat sentiment analysis is the process of extracting, quantifying, and interpreting the emotional tone and sentiments embedded within text-based conversations on the platform. It involves the application of Natural Language Processing (NLP) and machine learning techniques to automate the assessment of whether a message conveys positive, negative, or neutral sentiment. This analysis goes beyond mere text classification; it aims to unveil the underlying emotions, opinions, and trends, which can inform decision-making, improve user experiences, and offer a deeper understanding of social dynamics.

This project involves sentiment analysis on WhatsApp chat data using the Naive Bayes Classifier from the Natural Language Toolkit (NLTK) in Python. The classifier is trained on the movie reviews dataset provided by NLTK. The trained model is then applied to analyse sentiments in a WhatsApp chat export file, and the results are visualized using matplotlib.

**2. Libraries Used:**

**1. NLTK (Natural Language Toolkit):**

* Used for natural language processing and text analysis.
* Functions include tokenization, stemming, tagging, and sentiment analysis.

**2. Matplotlib:**

* Used for data visualization, creating plots and charts.
* Specifically used for pie charts and bar charts in this project

**3. NumPy:**

* Used for numerical computing and array operations.
* Specifically used for handling numerical data in the bar chart

**4. Pickle:**

* Used for serializing and deserializing Python objects.
* In this project, it's used to save and load the trained Naive Bayes Classifier model.

**5. Sys:**

* Provides access to some variables used or maintained by the Python interpreter and to functions that interact with the interpreter.
* Used in this project to access system-specific parameters**.**

**3. Naive Bayes Classifier:**

The Naive Bayes Classifier plays a crucial role in the sentiment analysis project outlined in the provided code. Here are the key reasons for the importance of the Naive Bayes Classifier in this context:

1. **Text Classification:**
   * The primary purpose of the project is sentiment analysis, which is a form of text classification. The Naive Bayes Classifier is well-suited for text classification tasks, including sentiment analysis.
2. **Simple and Effective:**
   * Naive Bayes is known for its simplicity and effectiveness. It is easy to implement and works well for many natural language processing tasks. This simplicity is advantageous, especially for projects with limited computational resources.
3. **Applicability to Natural Language Processing (NLP):**
   * Naive Bayes is a probabilistic algorithm that works well with the probabilistic nature of language. It's particularly effective for NLP tasks where the occurrence of words and their combinations can be modeled probabilistically.
4. **Training on Labeled Data:**
   * The Naive Bayes Classifier is a supervised learning algorithm, meaning it requires labeled training data. In the project, the classifier is trained on the 'movie\_reviews' dataset, where reviews are labeled as positive or negative, providing a basis for learning sentiment patterns.
5. **Fast Training and Prediction:**
   * Naive Bayes is computationally efficient and has fast training and prediction times. This is advantageous for real-time or near-real-time applications, allowing for quick analysis of sentiments in chat data.

**4. Methodology:**

**(a) Data Preparation:**

The data preparation step involves loading the Movie Reviews corpus, splitting it into negative and positive reviews, cleaning the reviews, creating feature sets, and splitting the feature sets into training and test sets.

• Loading the Movie Reviews corpus: The Movie Reviews corpus is a collection of 2,000 movie reviews, 1,000 positive and 1,000 negative. The corpus is loaded using the NLTK library.

• Splitting the corpus into negative and positive reviews: The corpus is split into negative and positive reviews using the movie\_reviews.fileids() function.

• Cleaning the reviews: The reviews are cleaned by removing punctuation and stop words. Punctuation is removed using the string.punctuation module in Python. Stop words are removed using the nltk.corpus.stopwords.words() function.

• Creating feature sets: A feature set for each review is created by converting the review to a dictionary where the keys are the words in the review and the values are True. This is done using the nltk.classify.util.dict\_features() function.

• Splitting the feature sets into training and test sets: The feature sets are split into training and test sets using the sklearn.model\_selection.train\_test\_split() function. The training set is used to train the classifier and the test set is used to evaluate the classifier.

**(b) Model Training:**

The model training step involves training a Naive Bayes classifier on the training set.

• Training a Naive Bayes classifier: The Naive Bayes classifier is trained using the nltk.NaiveBayesClassifier.train() function. The training set is passed to the function as an argument.

**(c) Model Evaluation:**

The model evaluation step involves evaluating the classifier on the test set and printing the accuracy.

• Evaluating the classifier on the test set: The classifier is evaluated on the test set using the nltk.classify.util.accuracy() function. The test set is passed to the function as an argument.

**(d) Model Saving:**

The model saving step involves saving the classifier to a file using pickle.

• Saving the classifier to a file: The classifier is saved to a file using the pickle.dump() function. The file name is passed to the function as an argument.

**(e) Model Loading:**

The model loading step involves loading the classifier from the file using pickle.

• Loading the classifier from a file: The classifier is loaded from the file using the pickle.load() function. The file name is passed to the function as an argument.

**(f) Sentiment Analysis of WhatsApp Chat Export File:**

The sentiment analysis of the WhatsApp chat export file involves opening the file, splitting each line into the sender's name and the message text, cleaning the message text, converting the message text to a feature set, classifying the message using the classifier, and printing the sender's name, the predicted sentiment, and the message text.

• Opening the WhatsApp chat export file: The WhatsApp chat export file is opened using the open() function.

• Splitting each line into the sender's name and the message text: Each line in the file is split into the sender's name and the message text using the str.split() function.

• Cleaning the message text: The message text is cleaned by removing punctuation and stop words.

• Converting the message text to a feature set: A feature set for each message is created by converting the message text to a dictionary where the keys are the words in the message and the values are True.

• Classifying the message using the classifier: The message is classified using the classifier using the classifier.classify() function.

• Printing the sender's name, the predicted sentiment, and the message text: The sender's name, the predicted sentiment, and the message text are printed using the print() function.

**(g) Data Visualization:**

The data visualization step involves plotting a pie chart showing the percentage of positive and negative messages in the WhatsApp chat export file and a bar chart showing the number of positive and negative messages for each sender in the WhatsApp chat export file.

• Plotting a pie chart: The pie chart is plotted using the matplotlib.pyplot.pie() function. The percentage of positive and negative messages are passed to the function as arguments.

• Plotting a bar chart: The bar chart is plotted using the matplotlib.pyplot.bar() function. The number of positive and negative messages for each sender are passed to the function as arguments.

**5. Significance:**

The significance of the sentiment analysis project lies in its potential applications and the insights it can provide into the emotional tone of text data, specifically WhatsApp chat data. Here are some key aspects of its significance:

1. **Understanding User Sentiment:**
   * By analysing sentiments expressed in WhatsApp chat data, the project can provide valuable insights into the emotional states of users. This understanding is crucial for businesses and individuals to gauge the overall sentiment of conversations.
2. **Customer Feedback Analysis:**
   * For businesses, sentiment analysis on customer interactions can be vital. It helps in evaluating customer feedback, identifying positive and negative sentiments, and addressing customer concerns or improving products/services accordingly.
3. **Real-time Monitoring:**
   * The project demonstrates the feasibility of real-time sentiment analysis by applying the trained Naive Bayes Classifier to WhatsApp chat data. Real-time monitoring of sentiments allows for timely responses to evolving situations or issues.
4. **Application in social media:**
   * Similar sentiment analysis techniques can be applied to social media data, helping organizations understand public opinion, track brand sentiment, and engage with users effectively.
5. **Insights for Decision-Making:**
   * The sentiment analysis results can be used to make informed decisions. For instance, businesses can identify areas for improvement, marketing strategies can be adjusted based on public sentiment, and individuals can gain insights into the overall emotional tone of their interactions.

**6. Conclusion:**

This project successfully demonstrates the application of sentiment analysis using the Naive Bayes Classifier on a WhatsApp chat export file. The classifier is trained on a movie reviews dataset, and its accuracy is tested. The sentiment analysis results are visualized using pie and bar charts, providing insights into the positive and negative sentiments expressed in the chat.