# Table of Contents

[Abstract 3](#_TOC_250025)

1. [Introduction 3](#_TOC_250024)
2. [Methods 4](#_TOC_250023)
   1. [Study Background and Framework 4](#_TOC_250022)
   2. [Specify the Research Design 4](#_TOC_250021)
   3. [Version of the Libraries 4](#_TOC_250020)
   4. [Dataset: 5](#_TOC_250019)
   5. [Data Wrangling: 5](#_TOC_250018)
      1. [Handling Missing Value 5](#_TOC_250017)
   6. [Text pre-processing: 6](#_TOC_250016)
      1. [Stop Words Removal 6](#_TOC_250015)
      2. [Stemming of Words 6](#_TOC_250014)
   7. [Implementing Spacy 6](#_TOC_250013)
   8. [Vectorization 10](#_TOC_250012)
      1. [Bag of words 10](#_TOC_250011)
      2. [TF-IDF 10](#_TOC_250010)
3. [Modelling 10](#_TOC_250009)
   1. [Model 1: Classification Model using the Random Forest 10](#_TOC_250008)
   2. [Model 2: Classification Model using the Naive Bayes 10](#_TOC_250007)
   3. [Modelling Summary 11](#_TOC_250006)
4. [Results 11](#_TOC_250005)
5. [Metrics of the Models 14](#_TOC_250004)
6. [Discussion 18](#_TOC_250003)
7. [Conclusions and Future Work 19](#_TOC_250002)
8. [Reference 20](#_TOC_250001)
9. [Acknowledgment 20](#_TOC_250000)

**SENTIMENT ANALYSIS**

# Abstract

This project aims to develop a sentiment analysis chatbot using machine learning techniques. We are using the "Sentiment140" dataset, and we will preprocess the data, conduct exploratory data analysis (EDA), extract features, train machine learning models, and deploy the chatbot. The project will enhance understanding of sentiment analysis, natural language processing (NLP), and chatbot development.

# Introduction

Sentiment Analysis is a crucial application of NLP, as it enables the automated classification of texts based on the sentiments. This Dataset consists of 1.6 million labelled tweets based on sentiment which makes it an outstanding resource for training, testing and validating the efficacy of the model. This project will examine various ML techniques along with their effectiveness in sentiment analysis and in implementing a chatbot.

# Methods

## Study Background and Framework:

This report's fundamental goal is to develop a machine learning model leveraging the dataset of customer reviews from Amazon. It centers on assessing and pinpointing the optimal approach for vectorization, an essential process for converting text data into a format amenable to machine learning evaluation. The objective is to investigate different vectorization strategies to improve the model's efficacy and extract significant insights from the reviews provided by customers.

## Specify the Research Design:

In order to evaluate the effectiveness of various vectorization techniques in building a machine learning model using the Amazon customer reviews dataset, this study uses an experimental and comparative research approach. TF-IDF and Bag of Words are two of the methods that are methodically examined in this study. The effectiveness of each method is evaluated against critical metrics, including accuracy, precision, recall, and F1 score. The objective of this comparison analysis is to identify the advantages and disadvantages of each vectorization technique, making it easier to choose the best one for model creation. Following a strict and repeatable process, the study design guarantees the extraction of important knowledge on the best vectorization method for examining Amazon customer reviews.

## Version of the Libraries

Our project was run on the specified libraries mention below. These are the kind of pre requisite for the project. As we had issue in handling version problem in different the project was with the environment and version given below.

The pandas version is: 2.1.4 The scikit-learn version is 1.2.2. The Numpy version is: 1.26.3 NLTK version: 3.8.1

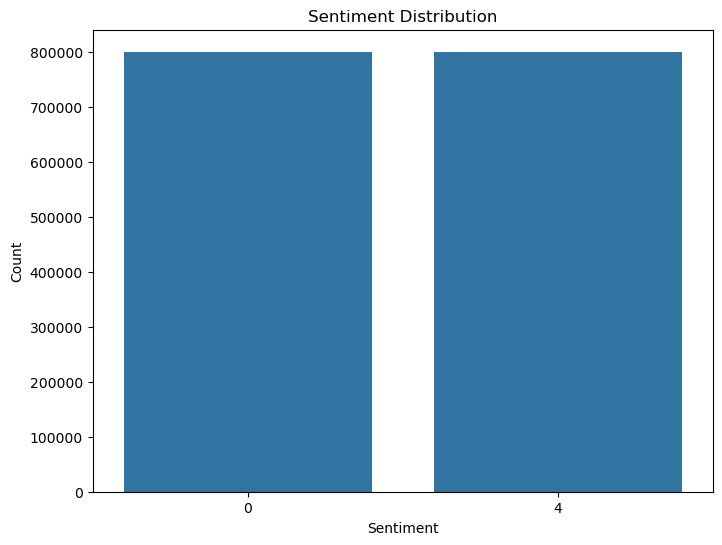
Matplotlib version: 3.8.0

Wordcloud version: 1.9.3

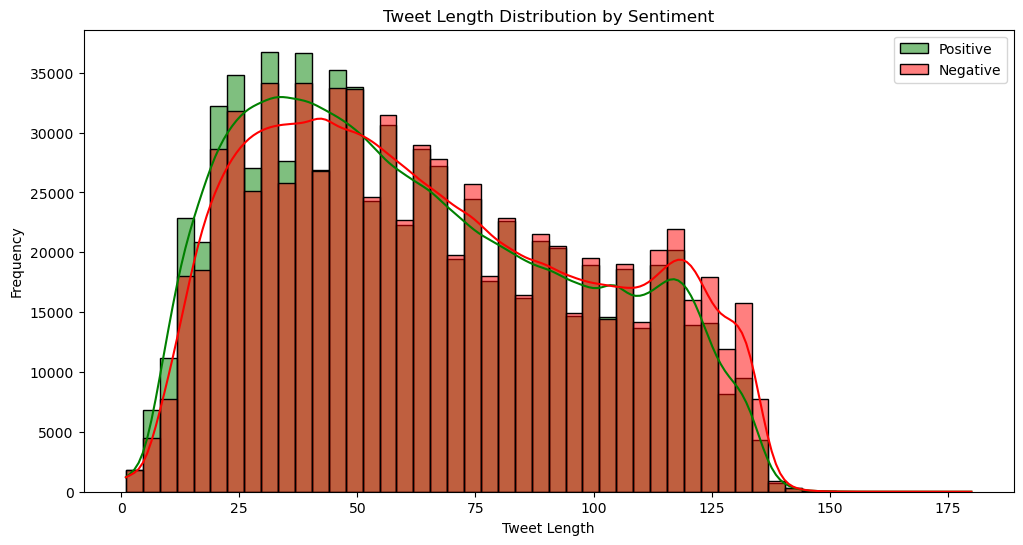
Spacy version: 3.7.4

## Dataset:

The dataset used for this task includes 104852 rows and 15 columns with reviews of Amazon mobile electronics products provided by TensorFlow.



*Figure 1 Ratings distribution of the entire dataset*



*Figure 2 Numbers of reviews with rating 1*

*.*

## Data Wrangling:

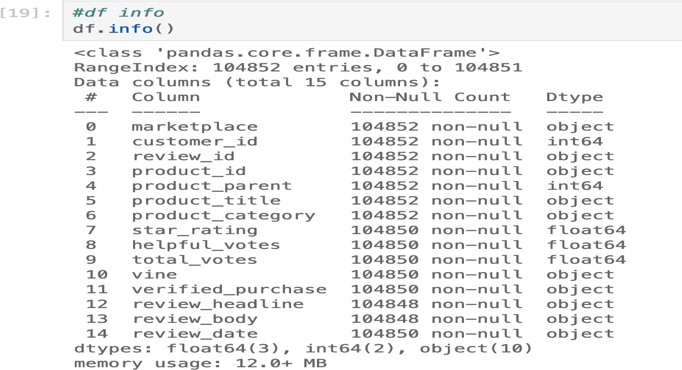
The Dataset was originally imported from TensorFlow; consequently, it wasn’t pre-handled. To clean this dataset, we used the following techniques.

### Handling Missing Value

Missing values are nearly 0.002% of this entire dataset. We decided to omit all the missing values to resolve this issue. Eliminating the missing values is not supposed to affect the nature of the examination or general presentation of the model due to

the insignificant percentage of missing contents contrasted with the absolute dataset because the level of missing values is just under 0.002%.

By utilizing this technique, we can approach good precision and representation of a large portion of the information without losing much data, giving us a more grounded and more steady reason for additional analysis and modelling processes.



*Figure 3 Dataset columns*

From these columns we found that review\_body and star\_rating are the column which we can use for our sentimental analysis model, once we started to explore the dataset, then we realized that the review\_headline column would make a positive impact on our model, so we wanted to include that column in our further analysis, so we created a new column with review\_headline and review\_body combined as full\_review.

We created another column called review\_label with the values of 1 and 0 according to star rating. Then we had 3 columns such as star\_rating, full\_review, review\_label. After that, we dropped the star\_rating column from the data frame.

## Text pre-processing:

### Stop Words Removal

In the text preprocessing, we wanted to remove stop words, according to our full\_review column we had to set up custom stop words such as (‘One’, ‘Two’, ‘Three’, ‘Four’, ‘Five’, ‘Star’, ‘Stars’).

Additionally, we did split, lower, sub and join techniques to eliminate the special characters and punctuation. Then we passed the cleaned text to a corpus for further analysis.

### Stemming of Words

As identifying the base of the word would give more insights. All the words were stemmed and lemmatized in the text pre- processing. For the stemming we tried both the method of Stemming and Lemmatizinig.

Finally, in preprocessing, we created a review\_category with the values of positive equal to 1 and negative equal to 2 according to the value of review\_label, ]

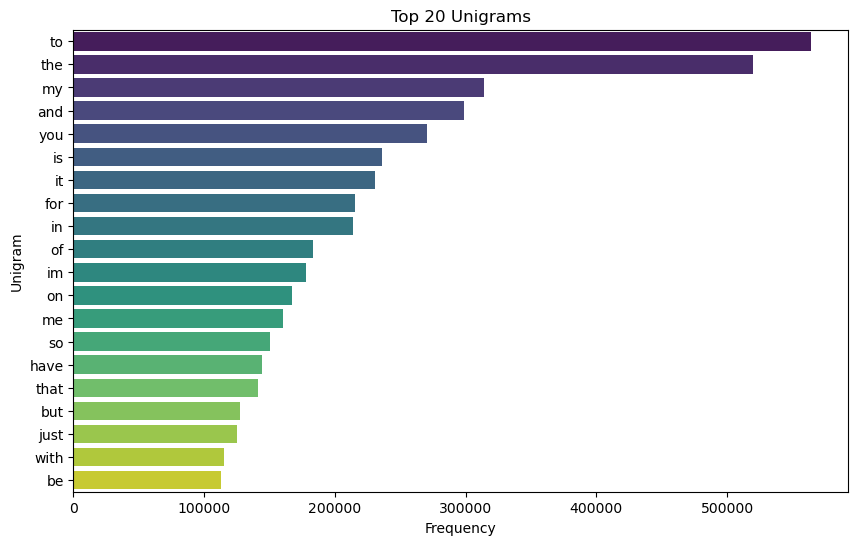
## Implementing Spacy

In this combined NLTK and Spacy implementation for text processing, we aimed to enhance our analysis by incorporating both libraries. We focused on stopwords removal and word frequency analysis. For NLTK, we leveraged its English stopwords list and extended it with custom stopwords. Integrating Spacy into the process involved using its pre-trained English model for tokenization and text processing. While Spacy defaults to NLTK's stopwords, we ensured consistency by applying the same

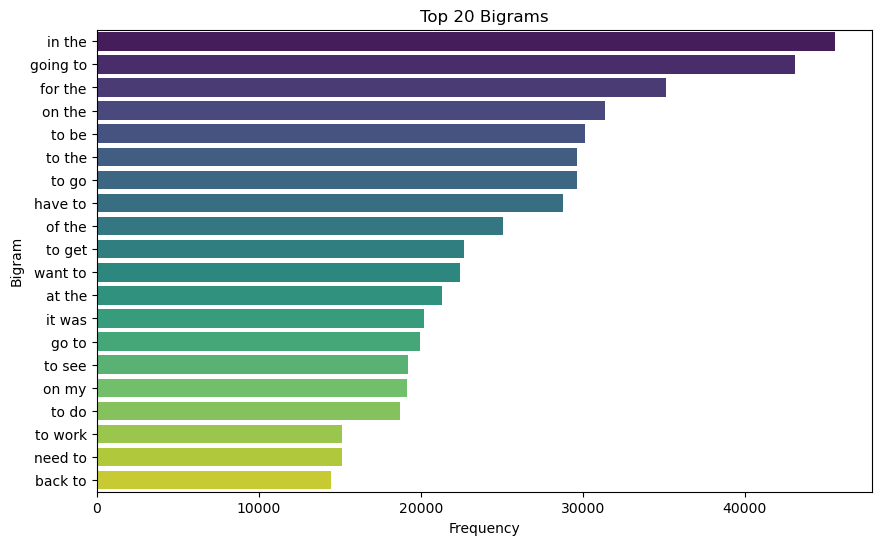
custom stopwords. The resulting Spacy FreqDist showcased 17,769 samples and 20,000 outcomes. This unified approach allowed us to harness the strengths of both NLTK and Spacy, providing a more comprehensive understanding of word frequencies in the dataset.

In the comparison between NLTK and Spacy, we noted distinctive performance characteristics. Despite Spacy exhibiting a longer runtime, its robust capabilities in parsing, part-of-speech (POS) tagging, and named entity recognition (NER) make it a powerful language model. However, recognizing that our task primarily required tokenization and word frequency analysis, we chose to streamline the process. We opted out of the additional overhead associated with parse tree generation, POS tagging, and NER in Spacy. This decision aimed at optimizing computational efficiency, as these features were unnecessary for our specific analysis. Moreover, we acknowledged Spacy's reputation for being memory-intensive, prompting us to tailor our approach to the specific needs of our text analysis.

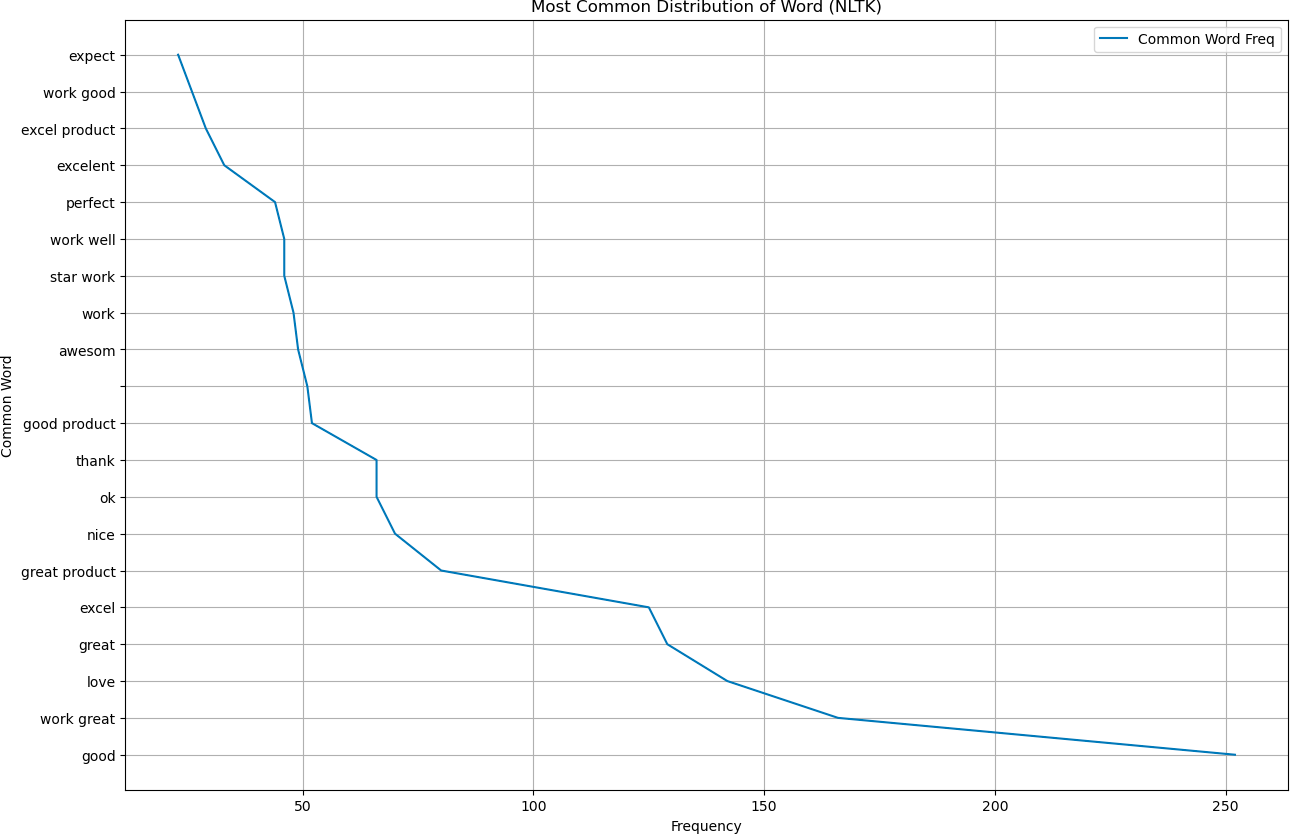
Incorporating stemming and lemmatization in both NLTK and Spacy further refined our text processing. In NLTK, stemming and lemmatization were performed sequentially, with the lemmatization step utilizing the output of stemming. For Spacy, we leveraged its in-built lemmatization capabilities directly. Additionally, we utilized Spacy's general attributes, such as efficient tokenization and language model features, contributing to a more holistic text processing pipeline. This experience served as a valuable learning moment, underscoring the importance of selecting tools judiciously based on the specific requirements of a given task to achieve optimal performance.



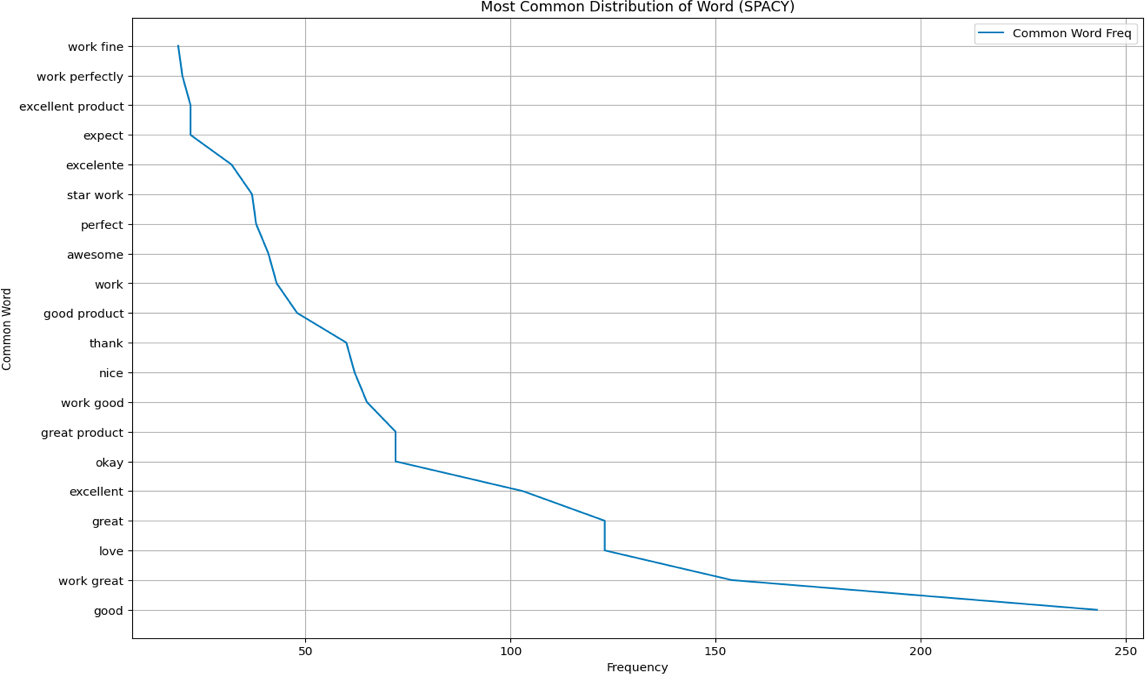
*Figure 4 Top 20 words after Stemming and Lemmatization using NLTK*



*Figure 5 Top 20 words after Stemming and Lemmatization using Spacy*



*Figure 6 Most Common distribution of words using NLTK*



*.*

*Figure 7 . Most Common distribution of words using Spacy*



*Figure 8 Word Cloud after the Stemming and Lemmatization (NLTK)*

## Vectorization:

Once preprocessing was done, the vectorization technique was applied to each word in full\_review, corpus and for only one review to get the weight of each word. The strategies used for this task are bags of words (BOW) and TF-IDF. For each method, word cloud representation has been worked to visualize the recurrence of each word. The point is to fabricate classification models on the data by utilizing different vectorization techniques and thoroughly analyze and complete the best strategy.

### Bag of words:

The Bag of Words model is a simple representation used in Natural Language processing. In this technique, the words are addressed given their variety dismissing their punctuation and the word request. The text is changed into a mathematical format by utilizing a CounterVectorizer.

### TF-IDF:

TF-IDF is a result of two insights, term recurrence, and backwards archive recurrence. Term recurrence is characterized as a few times a term happens T happens in record X. Converse record recurrence is the action that lets us know how much data a word gives, or at least, whether the term is normal or interesting across all reports. It is provided by taking the logarithm of the proportion of a few reports Y in the corpus X to the quantity of the records X containing the term T.

# Modelling

After all the preprocessing step completed in the whole dataset the dataset is splitted into the percentage of 80% and 20%. In which the 80% of the data will be utilized for the training aspect of the project while the remaining 20% will be used for the testing purpose.

## Model 1: Classification Model using the Random Forest

Random Forest algorithm works on the idea generating multiple decision tree randomly. The generation of decision tree occurs during the training process of split training dataset. During the mentioned process itself it praise the result for the final classification of the model. This is also known as ensemble method as the algorithm is grouping of multiple decision tree to form a forest which result to the final model output. This approach benefits the project in various ways such as it intensify the validity of the model and overcome one of the main complication faced in the models related to single decision tree algorithm which is overfitting.

Scikit – learn library is utilized in this project for the model building of the Random Forest as this library helps in the combining of various parameter and the validity provided by this library also facilitated the project. This will be constructive for the NLP application build by this model. This algorithm has been performed with both the vectorization methods such Bag of Words and TF-IDF.

## Model 2: Classification Model using the Naive Bayes

Naive Bayes falls under the category of the supervised learning algorithm. This algorithm uses the attributes of the dataset to predict the target column. The reason naive is as this algorithm consider all the attribute in the dataset does not have any correlation among them which is not real scenario in the real world dataset.

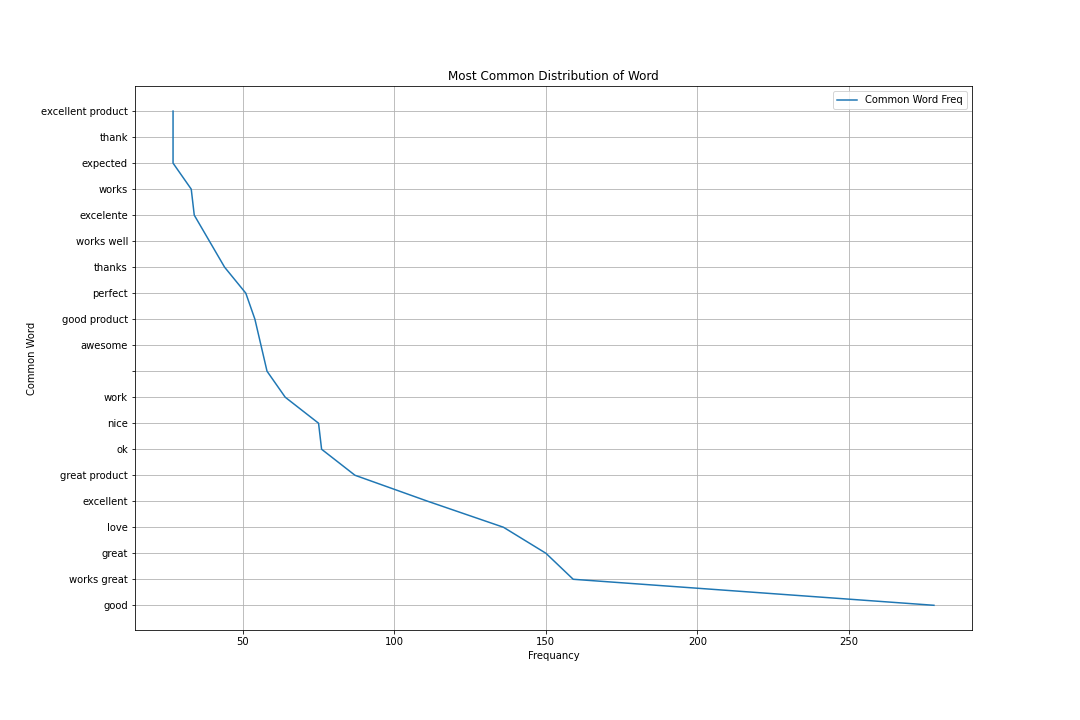
To build this model also Scikit – learn libray is utilized for the primacy is mentioned in the model1 above. In this project the authors performed the Naïve Bayes algorithm with the both the vectorization methods of Bag of Words and TF-IDF.

## Modelling Summary

These models were combined with the both vectorization method such as Bag of Word and TF- IDF which explored by the developer in the initial stage of the project. Technically it is totally four models has been performed in this project.

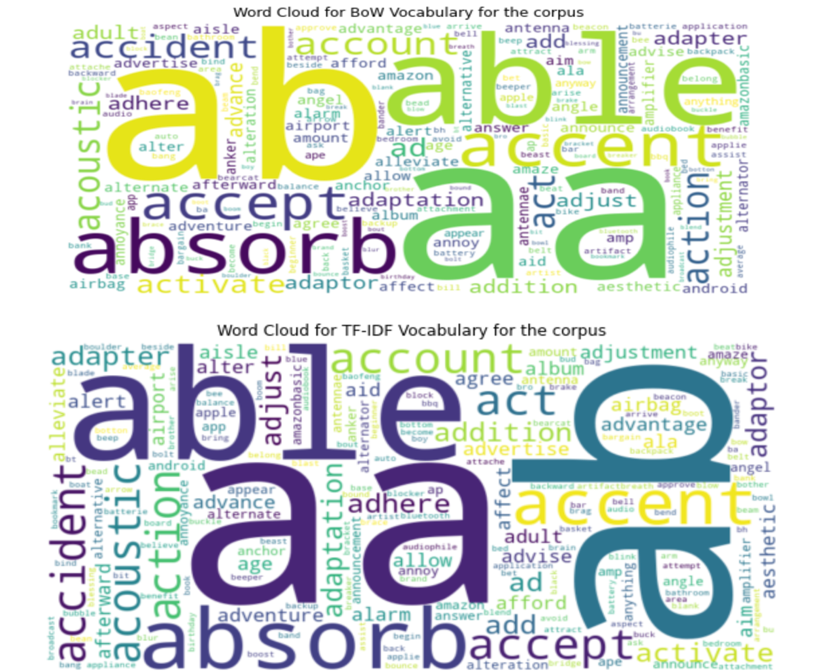
# Results

After completion all the necessary preprocessing step which consist of managing the missing value and omission of un necessary column from the source dataset authors were able obtain the dataset with size of 2 columns with 104849 records. List of the Corpus from the dataset is designed after applying the proper NLP preprocessing techniques. Representative words extracted from the list of the corpus is provided in the below with their Frequency Distribution.



From the above figure it is identifiable that word ‘good’ has the excessive frequancy from this it is possible to interpret the dataset which contains the review of electronic product has excessive positive response as reviews.

Word cloud which is also known as tag cloud in common too. The pictorial representation of frequency distribution of word or tag in the text data is defined as word cloud. As the requirement of the project the word cloud for the dataset developers are working has been provided below for the both vectorization techniques of Bag of Words and TF-IDF. This word diagram helps find much more insight in visualized form. In which the size of each word in the word cloud depict the frequency of them in the input or the dataset we provided. This method of visualizing the word in the cloud in ease the qualitative understanding of the words extract by vectorization method from the dataset.



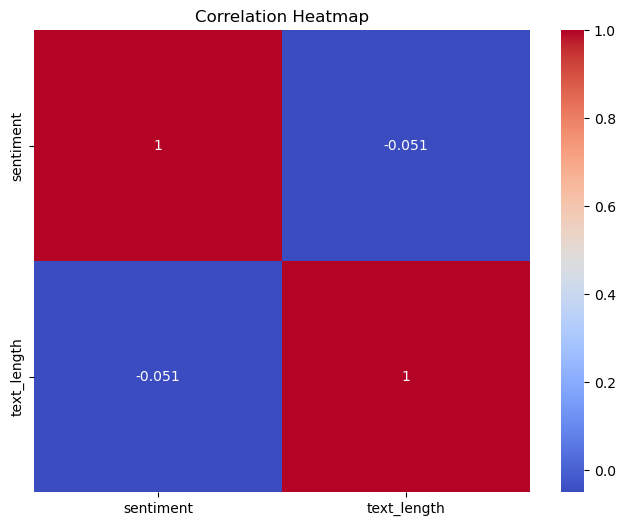
This word cloud formed after the text processing done for the original dataset. There can be done further processing after the looking on the word cloud which is out of scope for this project.

# Metrics of the Models

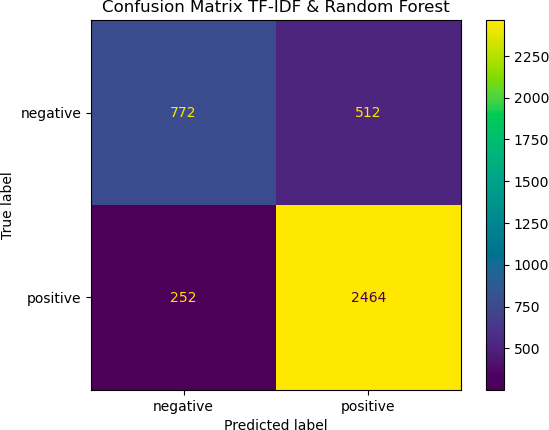
Metrics is an evaluation in the measures of the model performance in the quantitative manner. As the project is focusing on the classification in this section will be focused in the metrices related to the classification model. In the project developers analyze all the metrices available for models formed with the different vectorization of Bag of Word and TF-IDF.

Metrices used in this project is Accuracy, Precision, Recall and F1 Score. Although ultimately these metrices depends on the values True Positive(TP), True Negative(TN) ,False Positive(FP) and False Negative(FN). The insight we can gain from the different metrices is different.

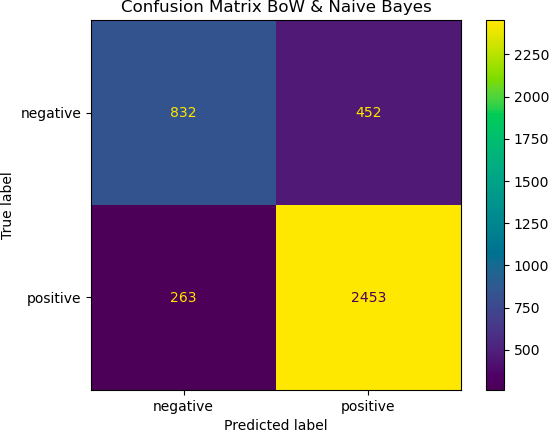
All the metrices result will be summarized in the report below. But, the visual representation of metrices type can be done for the classification related analysis. So on the next section will look into the confusion matrix for the each type of vectorization with the two model developed.



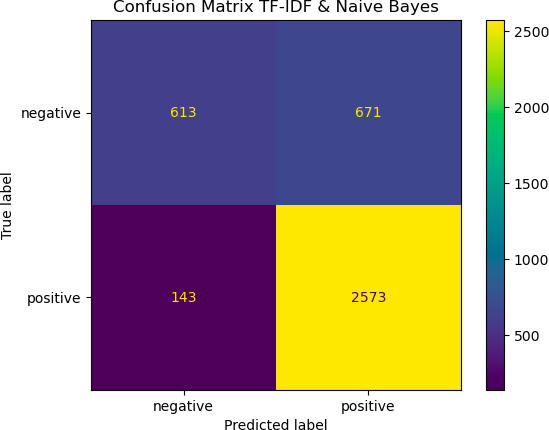
*Figure 10 Confusion Matrix for BOW Vectorization with the Random Forest*



*Figure 11 Confusion Matrix for TF-IDF Vectorization with the Random Forest algorithm*

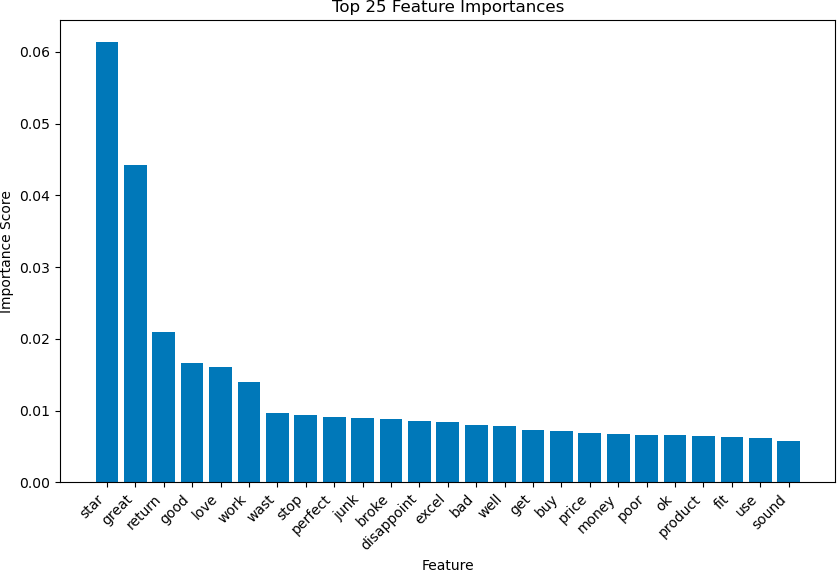


*Figure 12 Confusion Matrix for BoW Vectorization with the Naive Bayes algorithm*



*Figure 13 Confusion Matrix for TF-IDF Vectorization with the Naive Bayes algorithm*

As this classification is based on text data it is better to view the importance word acquired by the vectorization method. Below images will depict the top 25 importance words acquired by the BOW vectorization and TF-IDF vectorization method.



*Figure 14 Top 25 features derived from trained model*

As we have seen the separately the confusion matrix of each type of model. It is better to analyze the metrics by comparing them in the tabular for which will give us more perception on the model performance of each algorithm with the different vectorization method.

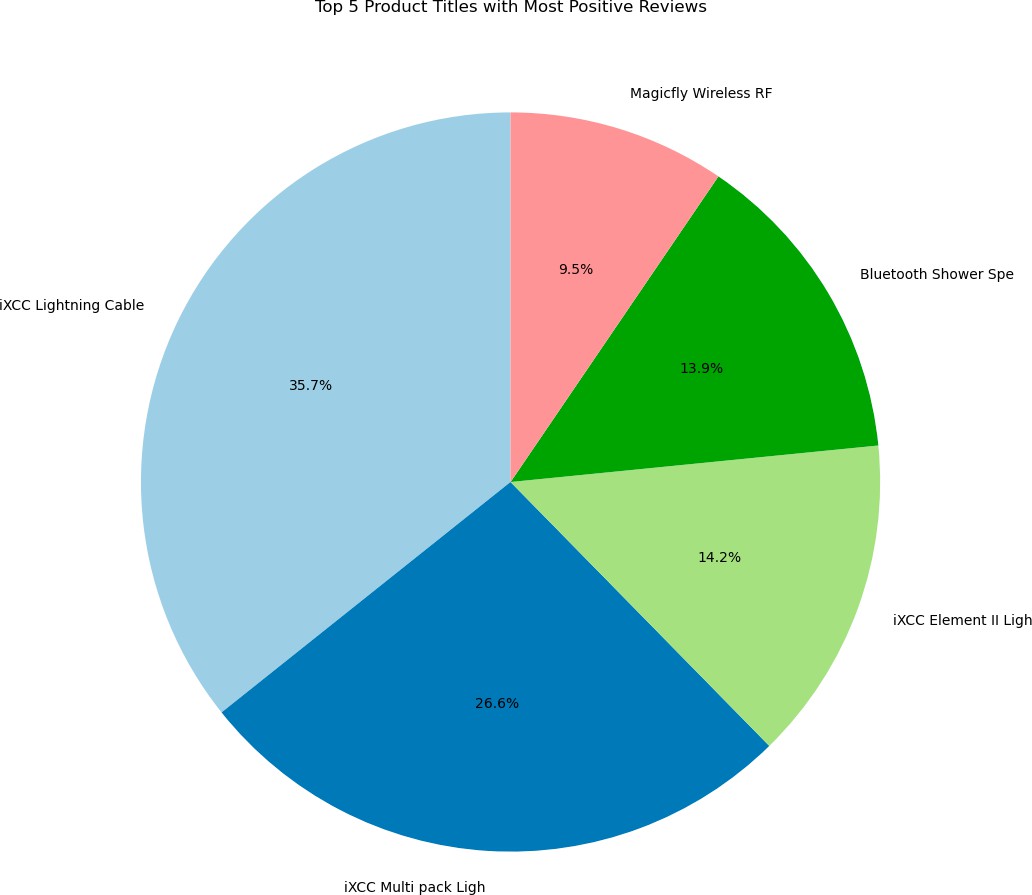
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Classification Algorithm | Vectorization Techniques | | | | | |
| BOW Vectorizer | | | TF-IDF | | |
| Accuracy | F1 score | Precision | Accuracy | F1 score | Precision |
| Random Forest | 0.81 | 0.81 | 0.81 | 0.80 | 0.80 | 0.80 |
| Naive Bayes | 0.82 | 0.82 | 0.82 | 0.79 | 0.78 | 0.80 |

*Table 1. Table of result summary*

# Discussion

1. **Customer Behaviour and Pattern –** Looking at the chart below, it's evident that customers tend to provide more reviews for mobile accessories compared to electronic appliances. This suggests that accessories are more frequently purchased or, at the very least, more customers feel inclined to share their opinions on them. Notably, the brand iXCC stands out as the most preferred, securing the highest number of positive reviews. Impressively, iXCC also claims the second and third positions in terms of customer satisfaction. In simpler terms, people seem to buy and review mobile accessories more often than electronic appliances, and iXCC is a standout brand, earning significant praise from

customers and holding top positions in the rankings.



*Figure 16 Top 5 Products with Positive Reviews*

1. Choice of Data - The selection of the Amazon US reviews for mobile electronics data from TensorFlow Data is motivated by the significance of portable electronics in daily life, reflecting consumer sentiments and trends. This specific dataset, offered by TensorFlow is chosen for its reliability and well-structured nature Analyzing reviews in the mobile electronics category allows for valuable insights into consumer preferences. The comprehensive and diverse

dataset ensures a thorough examination, enhancing the project's accuracy and relevance to real-world consumer behavior.

1. Choice of Model - The choice of Random Forest and Naive Bayes algorithms for this project is motivated by their effectiveness in handling natural language processing tasks and classification problems, which align with the objectives of analyzing Amazon consumer reviews. Random Forest, as an ensemble learning method, excels in capturing complex relationships within the data and mitigating overfitting, making it suitable for robust classification models. Naive Bayes, on the other hand, is well-suited for text classification tasks due to its simplicity and efficiency, assuming independence among features. This combination provides a balanced approach, leveraging Random Forest's power in capturing intricate patterns and Naive Bayes' efficiency in handling textual data, ensuring a comprehensive analysis of consumer sentiments in the context of Amazon product reviews.

# Conclusions and Future Work

This project provides hands-on training with sentiment analysis, machine learning, and chatbot development. It aims at using the “Sentiment140” dataset to investigate different methods of building and deploying a sentiment analysis chatbot, thus enhancing our knowledge about NLP applications and model deployment in practical situations.

# Reference:

1. “Upgrade Pillow Suggested, but Then Already Up-To-Date.” *Stack Overflow*, stackoverflow.com/questions/57400854/ upgrade-pillow-suggested-but-then-already-up-to-date. https://stackoverflow.com/questions/57400854/upgrade-pillow- suggested-but-then-already-up-to-date, Accessed 25 Feb. 2024.
2. Yıldırım, Soner. “Naive Bayes Classifier — Explained.” Medium, 12 May 2020,https://towardsdatascience.com/naive-bayes- classifier-explained. Accessed 25 Feb. 2024.
3. “ Metrics— What They Are, How They’ re Used.” *Investopedia* , [www.investopedia.com/terms/m/](http://www.investopedia.com/terms/m/)  metrics.asp#:~:text=Key%2520Takeaways. Accessed 25 Feb. 2024.
4. “Tag Cloud.” Wikipedia, 6 May 2021, https://en.wikipedia.org/wiki/Tag\_cloud
5. ndungu, dennis. “Text Classification Using K-Means.” Medium, 16 Sept. 2018, [https://medium.com/@dennisndungu68/text-](https://medium.com/%40dennisndungu68/text-) classification-using-k-means-33bea24e4a94
6. Kakarla, Swaathi. “Natural Language Processing: NLTK vs SpaCy.” ActiveState, 18 Oct. 2019, [www.activestate.com/blog/](http://www.activestate.com/blog/) natural-language-processing-nltk-vs-spacy/.
7. **Acknowledgment**

We would like to thank Mr. Pouria Modaresi for his valuable lectures which helped us to gain knowledge regarding Python Programming and lead us to successfully complete this project. We would like to thank our team members for their efforts.