

Sentimental analysis of food review contents to improve food  
culture and the quality of virtual content using natural  
language processing and machine learning

by

Ripa Sarkar  
18201083

Md. Mehedi Hassan  
19101570

Farin Beante Azad  
19101598

Md. Nibras Hossin  
20301463

A thesis submitted to the Department of Computer Science and Engineering  
in partial fulfillment of the requirements for the degree of  
B.Sc. in Computer Science

Department of Computer Science and Engineering  
Brac University  
September 2022

© 2022. Brac University  
All rights reserved.

# Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

## Student's Full Name & Signature:



---

Ripa Sarkar  
18201083



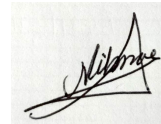
---

Farin Beante Azad  
19101598



---

Md. Mehedi Hassan  
19101570



---

Md. Nibras Hossin  
20301463

# Approval

The thesis/project titled “Sentimental analysis of food contents to improve food culture and virtual content quality using machine learning” submitted by

1. Ripa Sarkar (18201083)
2. Md. Mehedi Hassan (19101570)
3. Farin Beante Azad (19101598)
4. Md. Nibras Hossin (20301463)

Of Spring, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 19, 2023.

## Examining Committee:

Supervisor:  
(Member)



---

Arif Shakil [ARF]

Lecturer

Department of Computer Science and Engineering  
BRAC University

Program Coordinator:  
(Member)

---

Md. Golam Rabiul Alam

Associate Professor

Department of Computer Science and Engineering  
Brac University

Head of Department:  
(Chair)

---

Ms. Sadia Hamid Kazi

Chairperson

Department of Computer Science and Engineering  
Brac University

# Abstract

Internet-based applications like social media platforms and blogs have exploded in popularity, and with them have come reviews and commentary on people's daily lives. Unfortunately, the vast majority of these evaluations and commentary are critical. even in material devoted to reviewing food, like food blogs, vlogs and cooking videos. Data collection and analysis based on people's subjective feelings about a certain topic, product, subject, or service is known as sentiment analysis. By using techniques from natural language processing and text mining, sentiment analysis is able to recognize and extract empathetic details from written content. In this study, we'll go through a high-level introduction to the process for doing so, as well as the uses of sentiment analysis. After that, it analyzes the methods in order to weigh their merits and drawbacks via a series of comparisons and assessments. Several classifiers (Logistic Regression, Multinomial Naive Bayes, K-Nearest Neighbors, Decision Tree, Random Forest, AdaBoost, and SVM) are used to divide the sentiment into one of three categories, like positive, negative, or sarcastic.

**Keywords:** Facebook, YouTube, Positive, Negative, Sarcastic, Logistic Regression, K-Nearest Neighbor, Multinomial Naive Bayes, Random Forest, Decision Tree Classifier, Support Vector Machine, Adaboost Classifier, Gaussian Naive Bayes.

## **Dedication**

This paper is dedicated to all online food reviewers in an effort to help them become better at what they do as well as to restaurants to raise the standard of food quality in our country.

## **Acknowledgement**

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.

Secondly, to our co-advisor Mr. Arif Shakil sir for his kind support and advice in our work. He helped us whenever we needed help.

And finally to our parents without their throughout support it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

# Table of Contents

Declaration	i
Approval	ii
Abstract	iii
Dedication	iv
Acknowledgment	v
Table of Contents	vi
List of Figures	ix
Nomenclature	x
<b>1 Introduction</b>	<b>2</b>
1.1 Positive Comment . . . . .	3
1.2 Negative Comment . . . . .	4
1.3 Sarcastic Comment . . . . .	5
1.4 Problem Statement . . . . .	7
1.5 Research Motivation . . . . .	7
1.6 Research Objectives . . . . .	8
<b>2 Literature Review</b>	<b>9</b>
<b>3 Dataset</b>	<b>15</b>
3.1 Dataset Type . . . . .	15
3.2 Language of Dataset . . . . .	15
3.3 Dataset Collection . . . . .	15
3.4 Data Analysis . . . . .	17
3.5 Data Annotation . . . . .	20
3.6 Biasness . . . . .	21
3.7 Data Pre-Processing . . . . .	21
3.8 Pre-possessing Technique . . . . .	23
3.8.1 Bag of Words (BOW) . . . . .	23
3.8.2 Term Frequency Inverse Document Frequency (TF-IDF) . . . . .	23
3.8.3 Comparison between the Bag of Words and TF-IDF Tokenizer . . . . .	24

<b>4</b>	<b>Methodology</b>	<b>26</b>
4.1	Classifiers . . . . .	28
4.1.1	GaussianNB . . . . .	28
4.1.2	Decision Tree Classifier (DT) . . . . .	29
4.1.3	AdaBoost . . . . .	30
4.1.4	Logistic Regression (LR) . . . . .	32
4.1.5	Random Forest Classifier (RF) . . . . .	34
4.1.6	Multinomial Naive Bayes (MNB) . . . . .	35
4.1.7	Support Vector Machine (SVM) . . . . .	35
4.1.8	k-Nearest Neighbors (KNN) . . . . .	37
4.2	Features . . . . .	39
4.2.1	N-Gram . . . . .	39
4.2.2	Unigram Feature . . . . .	40
4.2.3	Bigram Feature . . . . .	40
4.2.4	Trigram Feature . . . . .	40
4.2.5	Comparison between the N-gram Features . . . . .	40
<b>5</b>	<b>Result Analysis</b>	<b>42</b>
5.1	Precision, Recall and F1 Score . . . . .	42
5.2	Result of English Dataset . . . . .	43
5.2.1	Bag of Word tokenized English Dataset . . . . .	44
5.2.2	TF-IDF tokenized English Dataset . . . . .	45
5.2.3	Score Generation . . . . .	47
5.2.4	Prediction . . . . .	47
5.3	Result of Bangla Dataset . . . . .	47
5.3.1	Dataset Distribution . . . . .	48
5.3.2	Data Statistics . . . . .	48
5.3.3	Length Frequency Distribution . . . . .	49
5.3.4	Bag of Word tokenizer . . . . .	49
5.3.5	TFIDF tokenizer . . . . .	56
5.3.6	Prediction . . . . .	62
5.4	Result of Banglish Dataset . . . . .	63
5.4.1	Bag Of Word tokenized Banglish Dataset . . . . .	63
5.4.2	TFIDF tokenized Banglish Dataset . . . . .	65
5.4.3	Score Generation . . . . .	66
5.4.4	Prediction . . . . .	67
5.5	Confusion Matrix . . . . .	67
5.5.1	Confusion Matrix of English Dataset for Bag of Words (BOW) . . . . .	68
5.5.2	Confusion Matrix of English Dataset for Term Frequency Inverse Document Frequency (TF-IDF) . . . . .	69
5.5.3	Confusion Matrix of Banglish Dataset for Bag of Words (BOW) . . . . .	70
5.5.4	Confusion Matrix of Banglish Dataset for Term Frequency Inverse Document Frequency (TF-IDF) . . . . .	71
5.6	Result Comparison . . . . .	71
5.7	Further Discussion . . . . .	72
5.7.1	Suggested Improvements . . . . .	76
<b>6</b>	<b>Conclusion</b>	<b>78</b>





# List of Figures

1.1	Positive Comment Case Ratio [23]	3
1.2	Negative Comment Case Ratio [25]	4
1.3	Sarcastic Comment Case Ratio 1 [21]	5
1.4	Sarcastic Comment Case Ratio 2 [22]	6
3.1	Filtered Dataset of Original Dataset	17
3.2	Filtered Dataset of Original Dataset	17
3.3	Bangla Dataset of Filtered Dataset	18
3.4	Bangla Dataset of Filtered Dataset	18
3.5	English Dataset of Filtered Dataset	19
3.6	English Dataset of Filtered Dataset	19
3.7	Banglish Dataset of Filtered Dataset	19
3.8	Banglish Dataset of Filtered Dataset	20
3.9	Data Pre-Processing	22
4.1	Methodology	27
4.2	Illustration of how a Gaussian Naive Bayes (GNB) classifier works[24]	28
4.3	Decision Tree Classifier [26]	30
4.4	AdaBoost Working Procedure [27]	31
4.5	Multinomial Logistic Regression Classifier [28]	32
4.6	Example of “k” models for k class	32
4.7	Choosing Decisions Using Random Forest	34
4.8	Multinomial Naive Bayes Classifier[29]	35
4.9	Graph of SVM [30]	36
4.10	Graph of KNN [31]	38
4.11	KNN Scenario-1 [31]	38
4.12	KNN Scenario-2 [31]	39
4.13	KNN Scenario-3 [31]	39
4.14	N-Gram [32]	40
5.1	Dataset Distribution	48
5.2	Data Statistics	48
5.3	Length Frequency Distribution	49
5.4	Unigram Feature	50
5.5	Bigram Feature	52
5.6	Trigram Feature	54
5.7	Unigram Feature	56
5.8	Bigram Feature	58
5.9	Trigram Feature	60

5.10	Confusion Matrix . . . . .	67
5.11	Confusion Matrices of Multi-Class Dataset of English . . . . .	68
5.12	Confusion Matrices of Multi-Class Dataset of English . . . . .	68
5.13	Confusion Matrices of Multi-Class Dataset of English . . . . .	69
5.14	Confusion Matrices of Multi-Class Dataset of English . . . . .	69
5.15	Confusion Matrices of Multi-Class Dataset of Banglish . . . . .	70
5.16	Confusion Matrices of Multi-Class Dataset of Banglish . . . . .	70
5.17	Confusion Matrices of Multi-Class Dataset of Banglish . . . . .	71
5.18	Confusion Matrices of Multi-Class Dataset of Banglish . . . . .	71

# Nomenclature

$\mu$	<i>Mean</i>
$N$	<i>Number of Rows</i>
$w$	<i>Importance Of Each Piece Of Information</i>
$\alpha$	<i>Contribution Of A Single Stump</i>
$\sigma$	<i>Standard Variance</i>
$SN$	<i>Social Network</i>
$TP$	<i>True Positive</i>
$TN$	<i>True Negative</i>
$FP$	<i>False Positive</i>
$FN$	<i>False Negative</i>
$LR$	<i>Logistic Regression</i>
$DT$	<i>Decission Tree</i>
$RF$	<i>Random Tree</i>
$ML$	<i>Machine Learning</i>
$SVM$	<i>Support Vector Machine</i>
$MNB$	<i>Multinomial Naive Bayes</i>
$KNN$	<i>K – Nearest Neighbour</i>
$RBF$	<i>Radial Basis Function Kernel</i>
$NLP$	<i>Natural Language Processing</i>
$NLTK$	<i>Natural Language Toolkit</i>
$FMGC$	<i>Fast Moving Consumer Goods</i>
$BOW$	<i>Bag Of Words</i>
$TFIDF$	<i>Term Frequency–Inverse Document Frequency</i>

# Chapter 1

## Introduction

As a sharing medium, social media has risen in popularity during the preceding decade. People share almost everything, even their daily lives, on social media. Blogging and vlogging about meal reviews has recently become popular on social media. Food critics and reviewers concentrate on restaurants and street cuisine. In addition, they evaluate the cuisine, atmosphere, cleanliness, and meal prices of various restaurants, as well as the quality of street food. Later, the meal reviewers publish the blog or vlog through social media and provide recommendations based on their own personal experiences. Finally, audience members provide remarks and are often seen applauding, condemning, or expressing varying perspectives. These evaluations and opinions have a substantial influence on restaurants, as consumers prefer to frequent establishments with a high ranking offered by reputable reviewers and customers. Moreover, the restaurant industry has developed and witnessed rapid expansion, making it one of the most competitive businesses in the world (Kristanti et al., 2012). In response to the rising demand for restaurants, a growing number of entrepreneurs are launching new eateries. Their company may flourish if they give excellent service and their cuisine is amazing. Proper advertising and evaluation are crucial for maximizing earnings. Despite the fact that customer satisfaction is the most important factor for a business to be well-known, it is essential for a business to be a well-known location.

Online reviews have an instant influence on eateries, for better or ill. Positive ratings may instantly increase revenues, whilst bad evaluations can permanently deter customers. The restaurant might make changes to its service, food, and interior design based on what customers say and ask for.

People go to different locations to taste new foods and spend time with their loved ones. It has become a fashion trend. Modern life is significantly impacted by restaurants. As there are several eateries available, it is difficult for consumers to choose one. According to a 2013 poll by Dimensional Research, ninety percent of buyers said that internet reviews impact their purchasing choices.

A branch of AI called "machine learning" explores the possibility that computers might learn like humans. In sentiment analysis, machine learning is used to figure out the overall tone of a piece of content.

Sentiment analysis employs NLP, text analysis, and computational linguistics to identify, extract, and evaluate user-specific information hidden within textual data. Reviews, surveys, social media, healthcare publications, and other platforms all play a role in conveying customers' opinions. Reviews being posted on social media at an

ever-increasing rate are akin to dealing with massive volumes of data. Customers and businesses alike benefit from reviews that thoughtfully consider diners’ subjective experiences by separating research findings into positive and negative categories depending on how people really feel.

## 1.1 Positive Comment

A favorable evaluation indicates consent, approval, or motivation. A positive review is a comment given by a satisfied customer or user of a service, product, or company. To better serve customers or maintain a high-quality offering, businesses can benefit from analyzing feedback from both current and former customers. Consumers and users are positively impacted by positive reviews since they encourage more assured decision-making and reduce anxiety, ultimately leading to a higher rate of purchase and adoption. The credibility and respect consumers have for the brand may also rise as a result. The expressions ”thank you,” ”pleased,” and ”happy” are all indicators of a favorable evaluation.



Figure 1.1: Positive Comment Case Ratio [23]

There are several case studies of positive comments on various online sites and on various platforms. According to a survey by the TrustPulse Marketing Blog, 94% of users or buyers buy their products after reading trusted comments, 73% of users believe a local business if it has positive comments; and 49% of users require the highest rating of at least 4 stars in order to trust the company or use the products.

It takes a lot of time and effort to manually locate positive comments. So now machine learning (ML) and the algorithm are used to solve the problem. As a result of the size and number of concurrent users on online platforms, there is widespread worry about the prevalence of positive reviews or comments that are automatically detected. ML is crucial in predicting positive comments.

Example of positive comments	
Type of language	Positive comment
English	That looks delicious.
Transliterate Bangla	etto yummy dekhte

## 1.2 Negative Comment

Negative reviews reflect a customer's firsthand poor experience with any sort of product or service. Furthermore, negative reviews can either be "good negative reviews" or "bad negative reviews". A customer's bad experience with the goods as told through reviews. As a result, potential customers will know that you don't hide feedback and appreciate openness. The reviews section is honest and unbiased, and the presence of negative reviews on product pages demonstrates this to prospective buyers. Reviews, even ones that aren't entirely positive, serve as information for other possible buyers when it comes to evaluating whether or not to buy the goods.

It is nothing new that some communities are filled with the negative vibes that lead to crime. Additionally, social media sites are a major factor in these kinds of crimes. Online customer reviews have been the primary determinant of a company's reputation. Consumers check products online 97% of the time before purchasing them. They read the evaluations. They can learn from them not only how the product functions but also exactly what features it has, how well the customer service is provided, and whether or not the seller is reliable.

Negative comments can seriously harm a company's reputation. Additionally, they have a psychological impact on the team. The marketing team is concerned about how this would impact lead generation; the sales team is naturally concerned about sales; and the executive team is concerned about the review itself and wonders "what can we possibly do to take it out?" Since there are also some anonymous sites, people feel free to start writing whatever they want on them.

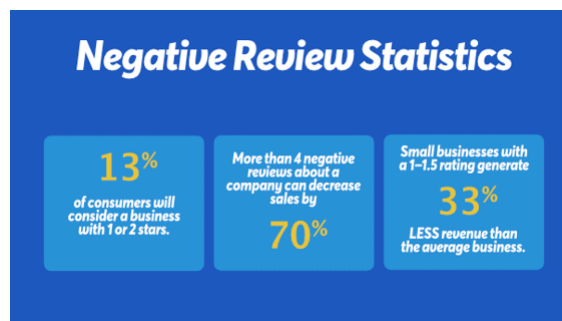


Figure 1.2: Negative Comment Case Ratio [25]

Numerous case studies of negative comments on different online platforms and websites exist. In a poll conducted by the TrustPulse Marketing Blog, there were a lot of negative comments or reviews online. The report also shows how bad reviews or comments hurt the company or business.

Negative comments must be manually found and removed, which is labor-intensive and time-consuming. So now the ML and the algorithm are used to fix the issue. Online platforms have a large user base and many concurrent users, so there is concern about the widespread use of bad evaluations or comments. This automatic identification of negative comments has a strong incentive. Over time, there have been more and more negative comments online, and machine learning has been a

key part of both predicting them and getting rid of them.

Example of negative comments	
Type of language	Negative comment
English	what kind of food is that? Fuck it looks like shit
Transliterate Bangla	oktopus tor goyar upor dimu

### 1.3 Sarcastic Comment

Sarcasm is the intentional misinterpretation of another person’s statements for the purpose of humorous effect or to cause offense. Basically, it’s a technique for being rude or insulting or mocking or making fun of someone by using words that don’t convey what is actually meant. Sarcasm is more common than usual on days when services are disrupted or reviews are manipulated, and negative sarcastic reviews generate significantly more social media feedback than actual bad remarks. For instance, it is sarcastic to refer to a group of people as ”very on top of things” when they are actually quite disorganized.

Some communities are filled with sardonic undertones that lead to atrocities. Furthermore, social media sites play a role in these kinds of crimes quite often. Since there are also some anonymous sites, people feel free to start writing whatever they want on them.

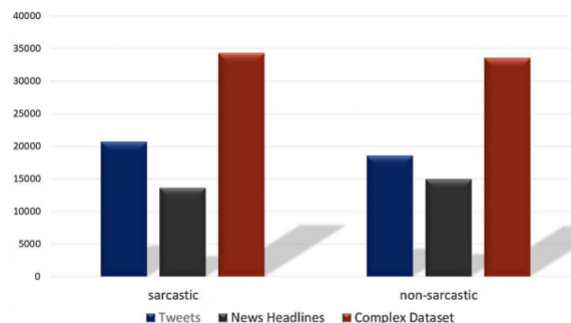


Figure 1.3: Sarcastic Comment Case Ratio 1 [21]

In the journal *Detecting Sarcasm in Multi-Domain Datasets Using Convolutional Neural Networks and a Long Short-Term Memory Network Model*, the sarcastic and non-sarcastic comments on tweets, news headlines, and complex datasets are looked at. This is shown in the graphic above. In the complex dataset and tweets, the sarcastic comments are greater than the non-sarcastic comments. The situation for news headlines is the exact reverse of that for complex datasets and tweets. For news headlines, sarcastic comments are less than non-sarcastic comments.

As we can see in the figure above, the journal *Developing Appreciation for Sarcasm and Sarcastic Gossip: It Depends on Perspective* looks at how sarcastic comments



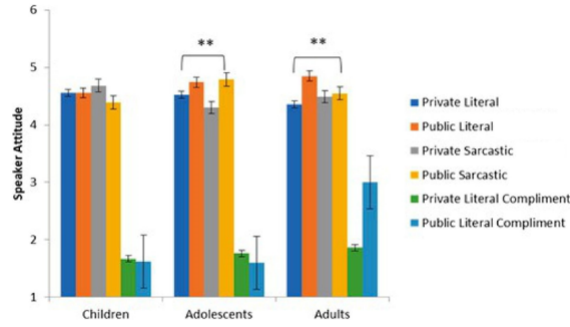


Figure 1.4: Sarcastic Comment Case Ratio 2 [22]

affect children, teens, and adults. The attitude of the speaker here ranges from 1 to 6. 1 represents the least sarcastic comments, and 6 represents the most extreme sarcastic comments. For children, literal sarcasm is equal in both private and public settings, although the most sarcastic comments are made in private. The lowest sarcastic comments are above public literal compliments, but private literal compliments are above public literal compliments and below public sarcastic comments. For adolescents, the public makes the most sarcastic comments, while the public’s literal sarcasm comes in second. The public literal compliment sarcastic comments are higher than the public literal compliment and lower than the private literal compliment sarcastic comments, but the private literal compliment sarcastic comments are lower than the private literal sarcastic comments. Adults tend to use public literal more frequently than public sarcastic comments, which are the second most common. The public literal compliment sarcastic comments are higher than the private literal compliment, lower than the private literal sarcastic comments, and higher than the private literal sarcastic comments. However, the public literal compliment sarcastic comments are higher than the private literal compliment and lower than the private literal sarcastic comments.

So, to solve the problem, we have ML (Machine Learning) and an algorithm. There is concern about the widespread use of sarcastic comments due to the size and number of concurrent users on online platforms, which is why software that automatically detects sarcastic comments or reviews has a compelling incentive. The frequency of online sarcastic comments has increased over recent years. ML has been essential in both predicting and eliminating them.

Example of sarcastic comments	
type of language	sarcastic comment
English	I think I had enough face-book for the day after seeing this review
Transliterate Bangla	ami toh bhabsi octopus er earring er advertisement eta

## 1.4 Problem Statement

It may be challenging to locate information that is relevant to food bloggers or vloggers since the definition differs from source to source. Some internet users experience cyberbullying and cyberhate because of their race or ethnicity.

There are still biases or inadequacies despite the fact that many algorithms have been employed to detect such unfriendly statements or items so far. Sometimes the AI or algorithm doesn't know the difference between enthusiastic, negative, or caustic material, or between that and certain generic remarks that are also considered. That's why it's so easy to have social media banned for no good reason. The feedback given to food bloggers and vloggers will be sorted into several emotional expressions, such as supportive, critical, and snarky. We propose a model that, when applied to certain data, may determine the nature of the remark being made. The way we suggest doing things will also put comments into different groups based on whether they are positive, negative, or sarcastic.

## 1.5 Research Motivation

The fast-moving consumer goods industry in Bangladesh has been expanding rapidly in recent years. There are essentially four distinct segments within the FMGC market.

1. Food and beverage
2. Personal and household care
3. Pharmaceuticals
4. Tobacco industries.

As a result, more eateries and catering businesses have sprung up to meet the rising demand. Food carts in Dhaka's residential neighborhoods marked the initial explosion of the city's fast food scene. The meal from these carts was fast and affordable compared to the less upscale places around. Meanwhile, because of our city's population and infrastructure, eating out has become an important way for us to socialize with our loved ones. As a result, people's opinions about those eateries and mobile kitchens have begun to circulate widely via social media. The restaurants and carts in question have inspired the creation of thousands of groups and pages where people can compare them to one another, share their own experiences with them, and give them overall ratings. These previously written food blogs or articles have recently transitioned to video form. Witness diners' reactions to a popular eatery and its cart in this video testimonial. This movie covers the entire dining experience, from entering and touring the restaurants to providing real-time feedback on the fare served. People who haven't been to the restaurant but are considering going there in the future can benefit from reading reviews that detail the cleanliness of the establishment, the helpfulness of the wait staff, the taste and presentation of the food, the value of the meal in relation to the cost, and so on. In addition, these characteristics have become cultural norms among foodies, and the restaurant industry has reached its pinnacle. However, despite its many admirable traits, this society has also developed some undesirable ones. Recently, in

the year 2020, Bangladesh was experiencing an economic catastrophe because of the COVID-19 epidemic. Few restaurant proprietors have followed his example since then. a few proprietors' covert participation in shady political or culinary practices Despite not maintaining the quality of their food and restaurant environment, some restaurant owners have resorted to questionable marketing strategies, such as posting fake reviews in social media groups devoted to food and hiring celebrity food bloggers, vloggers, and owners of social media groups and pages to promote their establishments. Some of the people who praised the food bloggers and vloggers on social media for their undying enthusiasm for food and their honest reviews have since turned on them. These days, it's not uncommon for social media groups and sites to have a false promotions, fraudulent reviews, and unrelated methods of rating the restaurant's cuisine and service. Some people will be drawn to you using this tactic; they'll be the ones who are initially astonished and hurt by your betrayal later on. Without proper oversight, the frequency of such events is rising. As a result, we intend to use statistics to prove these terrible crimes, and to publicize these findings so that more people are aware of what is going on. Further, health concerns are intertwined with this subject. So, we hope to apply techniques like sentiment analysis and ML to this problem in an effort to finally resolve it.

## 1.6 Research Objectives

The goal of this study is to use supervised machine learning algorithms to examine user sentiments in an effort to improve the quality of material on food blogs. Review content from platforms like Facebook and YouTube is mined to produce a dataset. A machine learning classification model may be used to achieve the desired outcomes by converting the preprocessed information into feature vectors.

The objectives of this research are:

1. To deeply understand sentiment analysis.
2. Create a machine learning-based approach to identify positive, negative, and sarcastic comments on social media platforms.
3. Work with the multiclass datasets.
4. Label the comments as positive, negative, or sarcastic in the appropriate order (1, 0, 2) and then classify them according to the food review groups.
5. Differentiating between positive, negative, and sarcastic evaluations can increase the precision or accuracy of detection.
6. Increase time efficiency by concentrating on faster prediction.
7. To find out the impact of online food reviews or blogging on customers.
8. To evaluate the current conditions of online food review quality.
9. To implement the improvement of online food content quality and restaurant customer service.

# Chapter 2

## Literature Review

Scholars from all around the world have conducted a huge and exhaustive study on the sentiment analysis of various forms of comment detection. Scholars from all over the world are trying to develop a system that can more accurately tell if something is positive, negative, or neutral.

Ayushi Mitra, of Bhubaneswar, Odisha, India, submitted a paper in 2020 titled Sentiment Analysis Using Machine Learning Approaches (Lexicon based on movie review dataset). She intended to employ rule-based techniques, which describe a set of standards and inputs such as classic NLP methods, stemming, tokenization, region of speech tagging, and machine learning parsing for sentiment analysis, implemented in Python. She has shown the test set, training set, and prediction set and attempted to give a general overview of the entire architecture and how the machine learning model functions by training the model first to gather experience (typically 80 percent of training data as input) and then making predictions (typically 20 percent of training data as input). Training and predictions are displayed. Using test cases, a model is trained to match input (text) to output (tag). Feature extractors convert text into feature vectors. Machine learning approaches use feature vectors and tags (such as positive, negative, or neutral) to create models. The feature extractor turns unread text into feature vectors. These feature vectors enter the model and offer expected tags (positive, negative, or neutral). She tested Naive Bayes, SKlearnBernoulliNB, Sklearn SVC, Decision Tree, Random Forest, and KNN on a dataset. She said as the lexicon increases, this technique becomes increasingly incorrect and time-consuming [1].

"Amazon Product Sentiment Analysis using Machine Learning Techniques" by S. Wassan, X. Chen, T. Shen, M. Waqar, and NZ. Jhanjhi (2021) proposed employing reflective aspects based on the item's qualities. They got the dataset from the data global center, where first opinion rate detection occurs. They utilized 28,000 customer ratings for more than 60 product categories. They measured polarity for Amazon's semantic analysis. When they study an Amazon customer review, a bar height of nearly-1 indicates bad reviews. The system preprocesses datasets by stone-coating, tokenizing, boxing, and deleting stop words to extract relevant information such as positivity or negativity. Through graph visualization, they introduce social network analysis. First, research-focused social networks like Amazon By testing Amazon's social network, they may describe its common features. Visualization

approaches help detect social network members' connections and features. Open-source software helps visualize and analyze network graphs. Then they read the CSV file to fulfill their tasks. They remove stop words before utilizing the NLTK package for NLP. Before text and feature extraction, they cleaned their dataset. Basic preprocessing on the training dataset provided these characteristics. They utilize the "textblob" library to fix spelling since it helps reduce word copies and understand nonsensical words while reading. Using tokenization, they categorized user reviews. Using a "textblob" library, they turned user evaluations into a collection of sentences. Stemming is the process of eliminating suffixes like "ing," "ly," "s," etc. using a portmanteau stemmer. Aspect-level data analysis will assist marketers in understanding customer preferences and altering their behavior. Last, they discuss text categorization [2].

Rajkumar S. Jagdale, Vishal S. Shirsat, and Sachin N. Deshmukh's Sentiment Analysis on Product Reviews Using Machine Learning Techniques (2019) research focused on product reviews. Using machine learning methods, a dataset consisting of six product reviews for cameras, laptops, mobile phones, tablets, TVs, and video surveillance was obtained from the Amazon website.com. They used machine learning techniques with the dictionary-based approach that they suggested, which is part of the lexicon-based approach. Every product evaluation has undergone sentiment analysis, which has been followed by classification utilizing NB and SVM machine learning algorithms. The dataset was gathered from Amazon in JSON format. Multiple reviews are included in each json file. In preprocessing, tokenization, stop word removal, stemming, and punctuation mark removal are used. The bag of words has evolved from it. The sentiment score was then derived for each sentence based on comparisons between the dataset and opinion lexicons, which included 4783 negative terms and 6,600 positive words. The accuracy of the Naive Bayes classifier for camera reviews was 98.17%, and the accuracy of the Support Vector Machine was 93.54% [3].

Another in-depth study, "Sentiment Analysis with NLP on Twitter Data," was produced by M. Arifuzzaman and Md. Rakibul Hasan of Bangladesh University of Engineering and Technology, Dhaka (2019). Twitter data is used to gauge public opinion of a product. "Samsung" and "iPhone" are popular gadgets. Compared to that, their iPhone had 100 tweets. To filter tweets, they constructed an NLP-based data structure (NLP). The proposed approach is built on three components: data extraction from a given project or product; preprocessing of retrieved tweets using NLTK; and a classifier model that assesses each tweet's sentiment. Twitter data is processed utilizing data extraction, tokenization, stemming, lemmatization, stopword removal, parts of speech tagging, named entity recognition, constructing a data frame, text modeling, and sentiment analysis. Each process has its own algorithm and packages. "vectorizer" and "classifier" are "clf" objects produced from "tfidfmodel.pickle" and "classifier.pickle." Then, iPhones reveal each tweet's predicted mood. Bag of Words (BoW) and TF-IDF models were employed to evaluate sentiment. They used the BoW model to examine Twitter. This study used BoW and TF-IDF to classify positive and negative tweets. Simulations showed that the TF-IDF vectorizer improved sentiment analysis accuracy. Using NLP with maximum features, minimum document frequency (min df), and maximum docu-

ment frequency (max df), they were able to accurately measure how people felt. They compared their technique to SVM, maximum entropy, naive Bayes, and the k-nearest neighbor classifier. They utilized the accuracy of 2000 characteristics for each approach to measure sentiment analysis techniques. They suggested that their method be used to figure out how people feel about any gadget, celebrity, or sports team because it is accurate [5].

Aliza Sarlan, Chayanit Nadam, and Shuib Basri's Twitter Sentiment Analysis (2014) study concentrated on social media. Social media disseminate public and private opinion on a range of themes. Twitter has gained popularity. Twitter provides enterprises with a rapid and efficient way to study clients' market views. Developing sentiment analysis software is one way to assess consumers' views. This study presented a sentiment analysis of many tweets. Development used prototypes. The pie chart and html page showed favorable and negative tweets from consumers. The program aimed to construct a web application system, but Django could only run on a Linux server or LAMP, thus this method had to be used. The project comprises two stages. First, literature investigation; then system development. Current literature research examines current sentiment analysis methodologies and methods. In phase 2, app requirements and features were defined. Also included are the program's architecture, interface, and interaction. The Twitter Sentiment Analysis software uses Python Shell 2.7.2 and Notepad. To use the Twitter API, developers must agree to Twitter's terms and conditions to access data. This method saved a JSON file. JSON (JavaScript Object Notation) is a lightweight, human-friendly data format. JSON is easy for machines to create and parse. JSON is a free text format. Twitter sentiment research analyzed client views on market success. The application uses machine-based learning and natural language processing to analyze sentiment. The program would classify positive and negative emotions in a pie chart and HTML page. Due to Django's constraints, the software was meant to be a web app. It could not be done. Future studies should improve this aspect [6].

In this study, sentiment analysis of scientific papers was performed using citation words on a previously created annotated corpus with 8736 citation sentences. DT generated the best micro F-score. LR also outperformed all other algorithms in the micro average without any additional characteristics. LR, DT, and NB, KNN, and RF work better when combined with other qualities. Only when using unigrams did LR performance become important. The consensus was that uni-grammars, bi-grammars, and tri-grammars without any additional features perform best when ranked first. SVM, LR, and RF performed extremely well overall and received the highest accuracy scores. N-grams influenced NB performance significantly; SVM was most accurate when only one n-gram was used; bigrams and tri-grams influenced LR performance significantly; and KNN performed best when only n-grams were used without any other words.

Research into categorizing text according to feelings or genre was initiated [47][52]. Current techniques recognize and categorize as positive, negative, or neutral the sentiment of individual phrases or whole publications. In certain instances, various sorts of sentiment classifications (such as "happy, sad, angry, fear, disgust, and surprise") were used to expose additional features of the text [65]. For text sentiment anal-

ysis, they might differentiate between symbolic and machine learning approaches. The symbolic method used rules and lexicons that were made by humans, while the machine learning method used supervised or weakly supervised learning to build a model from a large training corpus, with supervised models being the most common in this context [16].

In a symbolic context, a text was often treated as a collection of words (sometimes multi-word phrases) without regard to the relationships between the words (bag-of-words representation). A human expert defines the sentiment of a word, and the sentiment of a text is calculated as the sum of the sentiments of its constituent words (such as average or sum). Adjectives were strong emotion indicators [46]; nevertheless, their feelings might depend on the context, such as "a predictable storyline" vs. "predictable steering" [17]. examined the texts on the World Wide Web using the "near to" operator to detect the semantic orientation of adjectives and nouns by comparing the number of hits of the word as a close neighbor of "great" or "poor" [80]. used WordNet to identify a word's orientation [51]. Words in WordNet might be seen as nodes linked by synonymy connections, allowing a route to be constructed between them. The semantic orientation of a word might be determined by comparing the semantic similarity [33], i.e., the length of the route from a word to the terms "good" and "bad." Those who describe the relationship between emotion and object using particular mechanics that vary the intensity (e.g., intensification and quantification) or direction (e.g., negation and certain verbs) are examples of techniques other than the bag-of-words method.

Machine learning techniques for sentiment classification are gaining popularity due to their ability to model many features and, in doing so, capture context (refer to Polanyi and Zaenen, 2006), their more straightforward adaptability to changing input, and their capacity to quantify the degree of uncertainty in a classification. The most prevalent training techniques included examples manually categorized by humans. When defining training and test samples, the most prevalent ways of using single lowercase words (unigrams) as characteristics were absent. Certain attitudes are represented in two or more words in opinion mining, and the proper detection of negation is crucial since it inverts the polarity. [80] showed that word n-grams were good for word sense disambiguation, while [39] demonstrated that they could capture negation. In an alternate method to negation [68], the learning algorithm tags each word after a negation until the first punctuation with a negation tag. Other techniques choose just a subset of words [35], often by examining only adjectives recognized by a part-of-speech (POS) recognizer[20]. [83] built a vocabulary of opinion words, whereas identified and classified terms describing the attributes of a product (e.g., "The camera takes fantastic photographs"). The majority of supervised approaches are used to identify the emotions of whole texts [9]. [44] provides an example of the categorization of sentences (2003). [78] employs a step-by-step technique to categorize texts by first deleting objective sentences (using a minimum cut algorithm supported by machine learning) and then categorizing the remaining phrases.

They discovered only a small number of poorly supervised learning algorithms in the literature. In such contexts, manual labeling is restricted. In the work of [45],

for example, a hypothesis is formulated indicating that two adjectives united by "and" have the same orientation, but two adjectives joined by "but" have the opposite orientation. Using clustering, the semantic orientation of several adjectives in a corpus may be determined; the resulting clusters are then manually labeled [14]. [15] detects product qualities (PPs) and their linked opinion words (OWs) in texts by using an iterative cross-training strategy that begins with a tiny labeled portion of the corpus[18]. Two Naive Bayes classifiers are trained utilizing contextual characteristics (e.g., "presence of an OW/PP to the left/right" and "presence of an adjective to the left/right") to identify PPs and OWs, respectively. In each round, the PPs and OWs with the most accurate classifications are found and used as training examples in the following rounds.

With the strategies outlined above, it is currently possible to get quite decent outcomes, but there are still significant obstacles to overcome.

Opinion extraction from turbulent Web texts (such as blogs) still provides study challenges. The blog texts have several inconsistencies. New terms, abbreviations of existing words, and community jargon are no exception to this rule. On a syntactic level, it is often impossible to talk about true sentences. The TREC 2006 blog track included an opinion retrieval assignment to investigate the information-seeking behavior in the blogosphere, which was replicated by combining blogs on diverse themes with presumed spam blogs. Opinion mining on legal blogs has been conducted the processing of real malformed language has been identified in blog posts [55][56], and six basic emotion categories have been identified in blog sentences [7], but these studies rarely involved the processing of real malformed language [7]. In their tests for predicting political affiliation [59], they utilize an automated spell checker to replace misspelled words with their most probable repair. We don't know of any studies that try to figure out what people are feeling from (noisy) Dutch or French texts.

Researchers have only recently expressed an interest in gathering attitudes toward a specific item of interest (in our case, goods) or their properties (for example, the size of the Nokia mobile phone) [58][60]. Separate views might be expressed towards different entities or the same entity, but with variable intensity, in the same statement [15]. Both the TREC (2006) and (2007) Blog tracks featured a task involving the identification of blog postings that reflect an opinion towards a specific objective. The Blog track also had a subtask called "polarity," which asked participants to decide if the thoughts they found were positive or negative.

There was the issue of annotating adequate and representative training instances manually. In an active learning technique, all instances were labeled by a human, but the computer carefully selects the subset of examples to be labeled. We began with a seed set of labeled instances, but the presence of a seed set was not strictly required for the algorithm's proper application. At each iteration, a single example or a set of examples [38] [55] [50]. SaThe choice of instances was not arbitrary [10]. Typically, the picked instances were those that the current classifier deemed to be the most uncertain and hence the most revealing. Or, instances that were typical or varied from the pool of unlabeled examples were picked. Several methodologies



pertinent to our investigation were described below. Uses a committee of classifiers to determine a measure of classification (un)certainty for each case, finding uncertain instances to be the most illuminating [84]. Achieve the same objective using a probabilistic classifier. As a variant of support vector machines, [90] aimed to minimize the version space (the space of all viable hypotheses) when adding an example. For example, [50] [84] [86] have proposed a number of strategies for reducing future categorization errors (2004). People had come up with combinations of active learning techniques that could both explore the space of features and fine-tune the classification boundary [10] [77].

There was the challenge of categorization across domains and languages. Due to the absence of annotated training samples, it would be very beneficial to be able to transfer a learnt classification model to a different domain or language. Consequently, it was intriguing to see the impact of utilizing annotated examples from one topic domain (e.g., vehicle reviews) to train a classifier that was applied to another domain (e.g. movies)[8]. This issue had already been addressed by [4] and [44]. Footnote3 Overall, they demonstrated that sentiment analysis was a challenge that was very domain-specific and that it was challenging to develop a domain-independent classifier. An alternative to training the classifier on data from a single domain was to train it on data from many domains [41]. When we analyzed texts in English, Dutch, and French, we also evaluated how effectively the opinion extraction method might be adapted to multiple languages. This multilingual component of opinion mining was absolutely fresh.

# Chapter 3

## Dataset

### 3.1 Dataset Type

The Multiclass Dataset will be the focus of our research. The dataset includes English, Bangla, Banglish, and Emoji. The dataset will be divided into four datasets: a Bangla Dataset, an English Dataset, a Banglish Dataset, and an Emoji Dataset. But for now, we will concentrate on the English, Bangla and Banglish datasets.

### 3.2 Language of Dataset

The languages of the dataset that we have collected from "Facebook", "Instagram" and "YouTube" are divided into three types. They are Bangla, English and Benglish. For our English dataset, we used only characters from the English alphabet. As in, "I really like watching your videos." We used a Bangla dataset, therefore the language and alphabets used were both native to the region. Then we added a Banglish dataset. Banglish is not a specific language but as we are doing our thesis in Bangladesh this region's people write the Bangla language using English alphabets on social media or youtube etc. The specialty of this dataset is it is written in the Bangla language but the alphabets are English. Basically, it's the macaronic hybrid use of English and the language of Bangladesh(Bangla). For example, "Moja pelham na."

### 3.3 Dataset Collection

We have collected Facebook data through data scraping. Python and Python libraries were the tools we used. We used urllib, pandas, and BeautifulSoup . Urllib is used to fetch urls, Pandas is used for data analysis, and BeautifulSoup is used for scrapping. First we loaded a post and all its comments. Then saved it locally on the PC. We reached the file by declaring its destination using urllib.urlopen(). We used the classes of comments and profile names for the comments. To scrap comments and names, we applied soup.find().text.strip() and declared the classes of names and comments. used pandas.DataFrame() to contain comment and name lists. Then apply an array to read all the comments. and printed all the names and comments on a google sheet . We scrapped the date, restaurant name and location,

food item name and price manually.

However, there were difficulties because of the restrictions of Facebook's graph API requirements. No page or post allowed us to directly retrieve comments. However, we did manually gather the comments and labeled them accordingly. One of our primary objectives was to find various comments from the food review groups. Since the meaning of comment categories differed from person to person, we had to identify the comments as positive, negative, or sarcastic after we collected the comments. We named our dataset "Filtered Dataset."

We have done manual data collection from Instagram. For collecting data, we have selected several food review groups. The collected data had to be labeled as positive, negative, or sarcastic because the meaning of comment categories varied from person to person.

Web scraping is the process of taking information from a website and putting it on a user's own computer. Basically, web scraping is the action of transferring data from a website to your own computer's local storage. Afterward, we may analyze the data. Discover here how to extract data from YouTube comments and save it as an.csv file. Obtaining relevant data is the most challenging task for any data scientist working in a machine learning setting. Because good data produces a fantastic model and poor data produces a terrible one. Data may be gathered in a variety of ways. In-person visits to the site, reports, in-person observations, web scraping, and other methods may all be used to get this information. Data from a website may be "scraped" and stored locally. Scraping is known by many names on different websites. We may get data such as comments, likes, time, users, and user links using this method. The Google API Key is required. In fact, a Python module exists specifically for this purpose.

We may find the scraping at `youtube-comment-scraper-python 1.0.0`. DataKund published this library on April 12, 2021, and it may be used to automatically get YouTube comments from any web browser. It's used only for Windows app. Simply running this command will install the module. As the script above installs the module, it will also install the required dependencies, which are requested and bot studio. For browser automation, users turn to Bot Studio. By doing so, browsers will be prompted to begin the process of scraping the video's comments. The modules are well described, and they explain how to gather the data. Nevertheless, the information may be exported as a comma-separated value (CSV) and used in our machine learning engineering studies. As an example, if we want to scrape the URL of food blogging video. So we have to past the URLs into the inquiry box and then give the result. A new browser window will appear once some time has elapsed, and it will begin scraping the target website. It will take temporary control of our screen and do the necessary scraping. Since it just scrolls once by default, we should double-check whether or not our scrape was successful. If we get results similar to the one seen above, our scrape was successful.

### 3.4 Data Analysis

We have collected a total of over 18110 comments. Then we manually removed slang, tags, punctuation, and irrelevant comments from the dataset. However, we have created a separate filtered cleaning dataset. The filtered dataset contains 8330 data points, of which about 2856 were identified as positive comments, 4159 were identified as negative comments, and 1315 were sarcastic comments. We categorize the data by their origin, using terms like "Positive", "Negative" and "Sarcastic". As we categorized these divisive remarks, we found that some were really rather hostile. We have labeled the filtered data "0" as "Negative", "1" as "Positive", and "2" as "Sarcastic".

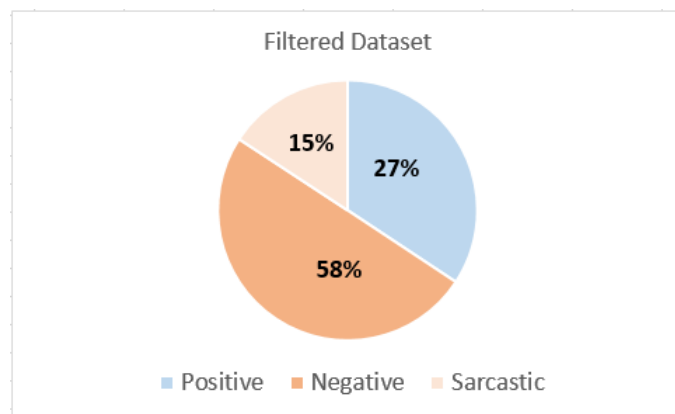


Figure 3.1: Filtered Dataset of Original Dataset

According to the analysis of the pie chart above, out of a total of 8330 comments, 27% are positive comments, 58% are negative comments, and 15% are sarcastic comments. We can therefore conclude that, when it comes to food vloggers, more people have had negative comments than positive ones.

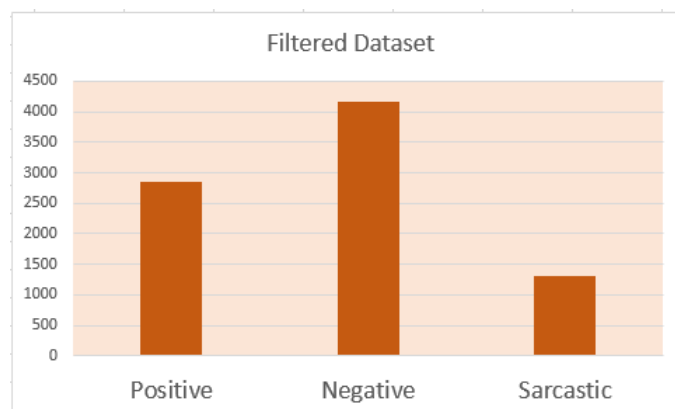


Figure 3.2: Filtered Dataset of Original Dataset

If we analyze the above bar chart, then we can see that in the filtered dataset, the negative comments are greater than the positive and sarcastic comments.

We have also separated Bangla and English data from the filtered dataset and created two different datasets, one for only the Bangla dataset and another for only the English dataset.

The Bangla dataset contained 2437 comments in total, of which about 1407 were identified as negative, 656 were found as positive, and 374 were sarcastic.

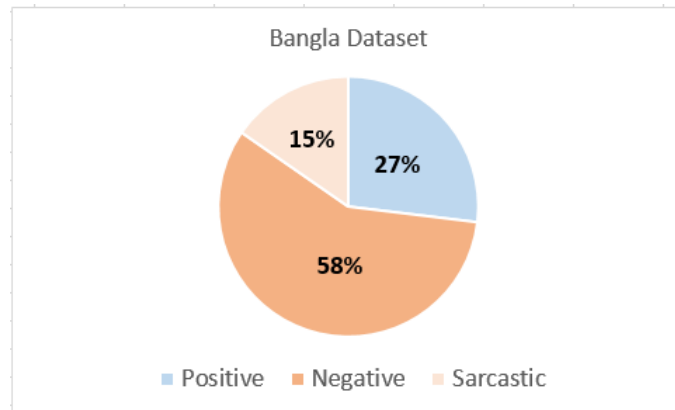


Figure 3.3: Bangla Dataset of Filtered Dataset

Based on the analysis of the pie chart above, the Bangla dataset contains 2437 comments, of which 27% are positive, 58% are negative, and 15% are sarcastic. Therefore, we can infer that more people have received negative comments on food bloggers than positive or sarcastic comments.

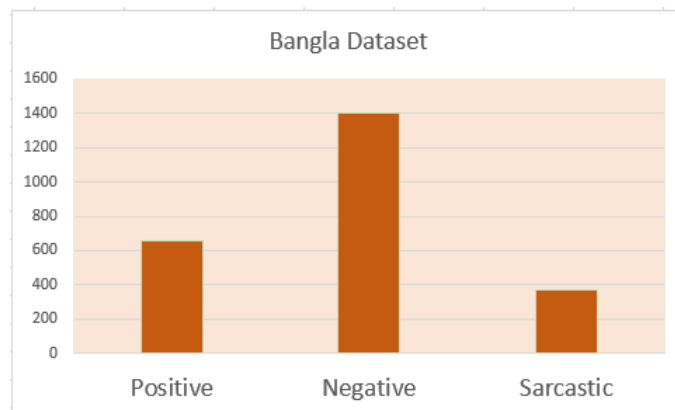


Figure 3.4: Bangla Dataset of Filtered Dataset

If we examine the aforementioned bar chart, we can observe that the negative comments outnumber the sarcastic and positive ones in the Bangla dataset.

The English dataset contained 3096 comments in total, of which about 1415 were identified as negative, 1204 were found as positive, and 477 were categorized as sarcastic.

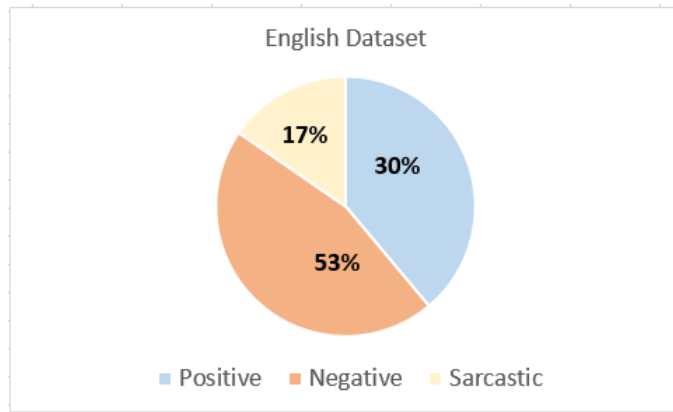


Figure 3.5: English Dataset of Filtered Dataset

After the study of the aforementioned pie chart, there are 3096 comments in the English dataset, of which 30% are positive, 53% are negative, and 17% are sarcastic. So, it stands to reason that more people have been subjected to negative comments about food bloggers than sarcastic or positive comments, which are at extreme levels.

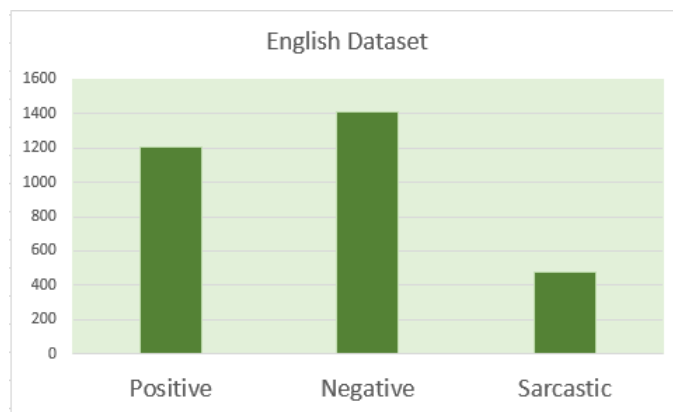


Figure 3.6: English Dataset of Filtered Dataset

Looking at the above bar chart, we can see that the English dataset has an extreme level of negative comments than sarcastic and positive ones.

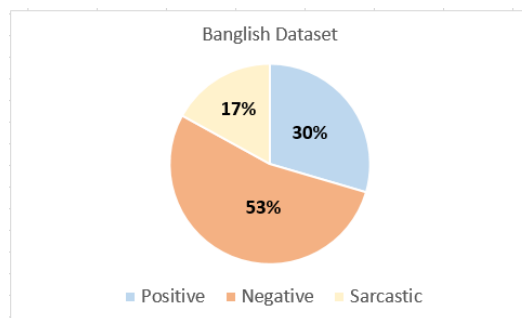


Figure 3.7: Banglish Dataset of Filtered Dataset

The total number of comments in the Banglish dataset was 1797 where about 961

were labeled as negative, 531 as positive, and 305 as sarcastic.

Based on the pie chart mentioned earlier, the Banglish dataset contains 1797 comments, of which 30% are positive, 53% are negative and 17% are sarcastic. Consequently, given the tremendous amounts of both sarcastic and complimentary remarks, it comes as the reason that more people have been exposed to negative comments regarding food bloggers.

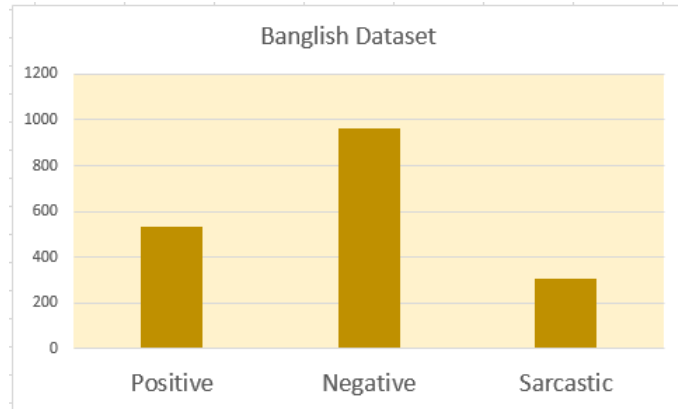


Figure 3.8: Banglish Dataset of Filtered Dataset

Taking a look at the aforementioned bar chart, we can see that the negative comments in the Banglish dataset far exceed the satirical and positive ones.

From Facebook, YouTube, and Instagram, we were able to compile a total of 18110 original data. In order to reduce the number of originals in our dataset from 12,000 to 7,000. We performed some manual filtering. We were given access to original data that was rife with details that were irrelevant to our study. For example, forty percent of the comments were people tagging their friends, while some remarks were completely unrelated, such as ones discussing the person’s fashion or style, how they apply their makeup, how they behave, and so on. Some of the comments were on point, but most of them were irrelevant to the topic manually. We are currently studying three distinct feelings: positive, negative, and sarcasm. All irrelevant feedback was eliminated from the filtered dataset. After filtering, the dataset is reduced to only the comments that mention one of these three emotions. To ensure that our data is relevant to the topic at hand, we removed all off-topic comments before compiling them.

### 3.5 Data Annotation

For further work, we classified the data into various,

- **Positive:** This category comprises comments that are related to praise, motivation, increasing self-esteem, and demonstrating people’s value and doesn’t contain any hate speech or unpleasant remarks.

- **Negative:** This course contains a negative comment on degrading and demeaning people based on their appearance, race, sex, religion, or slang.
- **Sarcastic:** This class contains a sarcastic remark that is intended to make fun of anything negatively.

## 3.6 Biasness

A biased stance is taken in favor of, or against, an individual or an issue due to the impact of one's own biases and preconceived notions. The concept of bias refers to how the model prioritizes certain features over others in order to better generalize to a larger dataset with a wider variety of qualities. In ML, bias does aid generalization and makes the model less sensitive to any one piece of input. It could be due to the internal biases of the human trainer. Or, it could be the result of an incomplete or inaccurate portrayal of any demographic in the data collection. It is not fair to draw conclusions from a dataset that has been skewed. It inaccurately portrays the machine learning model's intended purpose. Results are unreliable, judgments are biased, and the model's level of precision might shift depending on the conditions. A bias can taint information at numerous nodes. It can enter a dataset at any point, from data collection to data aggregation to model selection to final user interpretation. For example, When two people label the same set of photographs, they may arrive at different conclusions. Despite instructions to categorize images as boats or not as boats, a trainer may name one boat as a yacht and another as a ship, causing inconsistency in the final dataset. For all practical purposes, it is safe to presume that no data is available that is free from some form of bias.

We have completed our manual bias labeling in an effort to remove as much bias as possible from our dataset. As a means of eliminating potential bias from our data set. First, we select a large number of individuals without identities being revealed to label biases in our final dataset. Next, we classify people's comments as either positive, negative, or sarcastic. We make an actual sentiment segment of comments in our data set according to those chosen anonymous people's majority opinions. Those chosen people rapidly read all the comments in our dataset & give their stance as negative, positive, or sarcastic for one particular comment. We have also made another column to give numbers for negative positive sarcastic comments. For negative we give 0, for positive 1 and for sarcastic 2.

## 3.7 Data Pre-Processing

Raw data must be cleaned and prepared before it can be analyzed or used to train classifiers. due to the fact that the data is presented in English, Bangla and Banglish datasets.

For the English language, as it is possible to determine the particular language in which the data was originally collected, we calculate for the English language. This means there are also some stop words. However, we also begin by cleaning up the



text by omitting any punctuation or other symbols. Next, we convert all the sentences to lowercase, and then we use the split function to convert them all into tokens. Once that's done, we'll have usable data to go through some processing before moving on by removing common morphological and inflexional endings from words using the PorterStemmer function. A port stemmer from the NLTK library was utilized. Then we remove the unnecessary words by using stop words. The quantity of stop words can also be used to quantify lost information. Stop words have been imported using the Natural Language Toolkit (NLTK) package. This is the preprocessing for the English language.

For the Bangla language, it is also possible to determine the Bangla language in which the data was originally collected. However, we also begin by cleaning up the text by omitting unnecessary punctuation. Next, we remove all low-length sentences belonging to the Bangla dataset.

Since the data is presented in Banglish or transliterated Bengali, the data does not belong to any particular language. So, we don't have any stop words' either. We begin by cleaning up the text by omitting extra spaces and punctuation. After that, we convert every sentence to lowercase and use the split function to turn it into a series of tokens. After that, we may proceed with further processing by utilizing the PorterStemmer function to remove frequent morphological and inflexional endings from words. Then, we get rid of the fluff that doesn't contribute anything to the data's actual meaning.

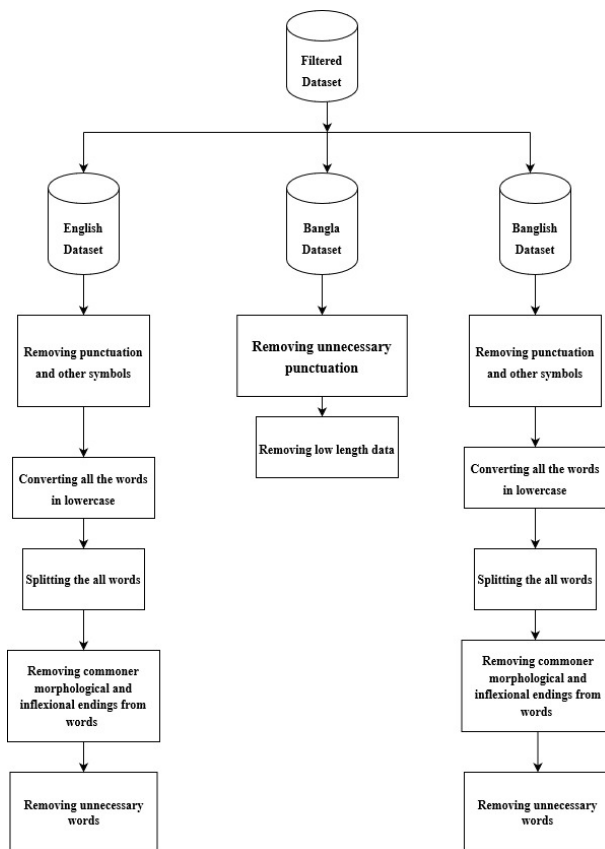


Figure 3.9: Data Pre-Processing

## 3.8 Pre-processing Technique

### 3.8.1 Bag of Words (BOW)

Word counting is a technique for generating fixed-length vectors from text using natural language processing by determining how often specific words occur in the text. Vectorization is a common term for this procedure. In the context of image processing, it is necessary to give each pixel its own number in order to extract information from the image. Unfortunately, a photograph is basically useless as an input for the algorithm. After that, the information is converted into a numerical format for better readability. Similarly, natural language requires quantification prior to comprehension. Our language has a word for this process: vectorization. The "bag of words" vectorization strategy is explained. Due to the absence of contextual information on the pattern or words that have been eliminated, the resulting text is referred to as a "bag" of words. The recognition of the words is more important to the model than the context in which they appear. It will return a total of the number of times a certain term occurs in a given collection of sentences.

In the first stage, the model dissects each word of the phrase individually. The punctuation marks will be erased. In addition, all the words will be spelled with lowercase letters. When it's done, it'll tally up how many times a certain term was used in that paragraph. There are three sentences in this example; the first has three words, the second has five, and the third has eight. There are a total of 16 words available. The algorithm then calculates a score for each phrase based on the frequency with which individual words occur. Each sentence is converted into a vector. That's really the point of the whole lexical grab bag, too. It employs the n-gram technique, which essentially utilizes a correspondingly large number of words to dispel any confusion about terms with identical meanings. This happens when words in two separate phrases have the same meaning. You could hear, "This is a terrific job. The words "This is not good at all" and "I will not miss it for anything." are shared by the two phrases, which might be taken to suggest that both of them have a bad connotation. This model may provide a different result if it is fed a string of two or more words in rapid succession. But there's a catch. The matrix and vector length will grow exponentially if the word diversity is high. The processing time will be significantly lengthened. Except that, it's a straightforward method of vectorization and tokenization.

### 3.8.2 Term Frequency Inverse Document Frequency (TF-IDF)

TF-IDF is an analytical measure used to determine a word's importance in a text. To abbreviate "Term Frequency" and "Inverse Document Frequency," we use "TF-IDF." There is also a heavy reliance on this technology in the field of Natural Language Processing, which employs several machine learning techniques to assign numerical values to individual words in a document (NLP). The original intent of TF-primary IDF was to help in the discovery of information inside documents through an indexed search. In practice, it works by becoming bigger according to the frequency with which a word occurs in a document and contracting directly pro-

portional to the number of papers that carry the same phrase. Even though they are used a lot, phrases like "this," "what," and "if" get a low significance score because they don't usually have anything to do with the context of the text in question.

The interrelatedness of words is included in the TF-IDF calculation. With the use of TF, the frequency with which a word occurs in a text may be determined. You can get the total number of words by dividing the frequency by that number. The information distribution function (IDF) is a second metric that takes into account how often certain search phrases occur in different texts.

The "term frequency" of a word measures how often that phrase occurs in a given text. A phrase is likely to appear much more often in longer texts than in shorter ones due to the different durations of each document. To do this, we divide the document-level frequency count by the total number of words in a text. In doing so, the outcomes are made to seem more typical.

$$TF = \frac{\text{Number of words used in document}}{\text{Total Number of words of the document}} \quad (3.1)$$

One method for evaluating a sentence's importance is the Inverse Document Frequency (IDF) method. While computing the TF, each sentence is given the same weight. It is generally agreed, however, that certain words have no real meaning despite their frequency of use, such as "is," "of," and "that." Because of these numbers, we can see where we can make cuts: in the usage of popular phrases and the frequency with which less common terms are employed.

$$IDF(t) = \frac{\text{Total number of documents}}{\text{Number of documents with term } t} \quad (3.2)$$

"Boy", "good", "girl", "good", and "good job" There are many more permutations than just these three. In order to determine the Term Frequency (TF) of Good, Boy, and Girl across three different sets of phrases, we divide the number of occurrences of each term by the total number of words in each set of phrases. IDF is computed by dividing the log of the total number of sentences (here 3,) by the percentage of sentences containing the target word.  $\log(3/3)$  is also the value of the Good because there are only three sentences and the word "good" occurs in all of them. It is now time to increase the TF by the IDF.

### 3.8.3 Comparison between the Bag of Words and TF-IDF Tokenizer

Bag of Words (BoW) tokenizer and TF-IDF tokenizer are two different methods for representing text data as numerical features that can be used as input to a machine learning classifier. They give different results in terms of accuracy, precision, recall, and F1 score because they represent the text data in different ways.

BoW tokenizer represents a text as a bag of its words, where each word is a feature and its frequency in the text is the value. This representation loses the order of words and the context in which they appear. BoW tokenizer gives a high weightage to the frequently appearing words in the dataset, which may or may not be informative to the problem you are trying to solve. This can lead to a high accuracy, but lower precision and recall.

On the other hand, the TF-IDF tokenizer represents a text as a weighted sum of its words, where each word is a feature and its importance is the value. This representation takes into account both the frequency of a word in the text (TF) and its rarity in the dataset (IDF). The idea behind using TF-IDF is that words that are rare across the entire dataset are likely to be more informative than words that are common. This helps in giving more weightage to the less frequently appearing words. This can lead to a lower accuracy but higher precision and recall.

It is important to test multiple tokenizers and choose the one that performs best for a specific task. It is also important to consider the trade-off between accuracy, precision, recall, and F1 score when choosing a tokenizer. You should use the one that prioritizes the metric that is more important for your problem.

# Chapter 4

## Methodology

Many studies have gone into dissecting online social networks. Classifying them using geometric, statistical, and topological categories is possible. Most analytic systems used detection, extraction, selection, and classification. They successfully identify OSN (Online Social Network) network analysis or graph visualization by combining many methods and techniques. Graph-based social network analysis is briefly explained in this study as well. Users of social networking sites may benefit from using social network visualization approaches by learning more about the relationships and traits between different entities. All sorts of network diagrams are produced and analyzed with the help of open-source software. To conduct our study, we collected feedback from various online forums devoted to rating and discussing cuisines.

The first step is to choose several SN sites, such as Facebook, Instagram and YouTube, for closer inspection. Then, we chose a number of food review groups on Facebook, Instagram and YouTube and collected comments for the study using sentiment analysis.

Second, we use Facebook and YouTube scrapers to collect data from Facebook and YouTube, respectively, but we also gather the comments by manually. The Instagram comments are also gathered manually. There is a lot of jargon, tags, and off-topic discussions in the bulk of the comments.

As a result, in the third stage, we manually removed the extraneous feedback from our data set and assigned ratings of 1 for positive, 2 for sarcastic, and 0 for negative to each piece of information. In the last step, we divide the cleaned data set into three languages: English, Bangla and Banglish. For the English dataset, we remove the stop words at the outset of solving natural language processing (NLP) issues. By counting the stop words, we may approximate the missing data. When importing, we used the NLTK package to import stop words. We have achieved this thus far by learning how to extract basic features from user feedback. We cleaned it up to ensure we got the most helpful information from our dataset. These capabilities were accomplished by elemental pre-processing activities carried out on the dataset. We used the bag of words and TF-IDF tokenization for the Bangla, English and Banglish datasets to divide the user feedback into separate words and phrases. Specifically, a porter stemmer from the NLTK library was used. The dataset is

divided into training and testing subsets using the training test split function. Then we apply several classifiers to each dataset. For the Bangla dataset, we use Linear and RBF Support Vector Machine, Logistic Regression, Multinomial Naive Bayes Algorithm, K Nearest Neighbors Algorithm and AdaBoost Classifier. We employ Logistic Regression, Multinomial Naive Bayes Algorithm, Gaussian Naive Bayes Algorithm, Decision Tree Classifier, Random Forest Classifier, and other classifiers for the Banglish and English datasets.

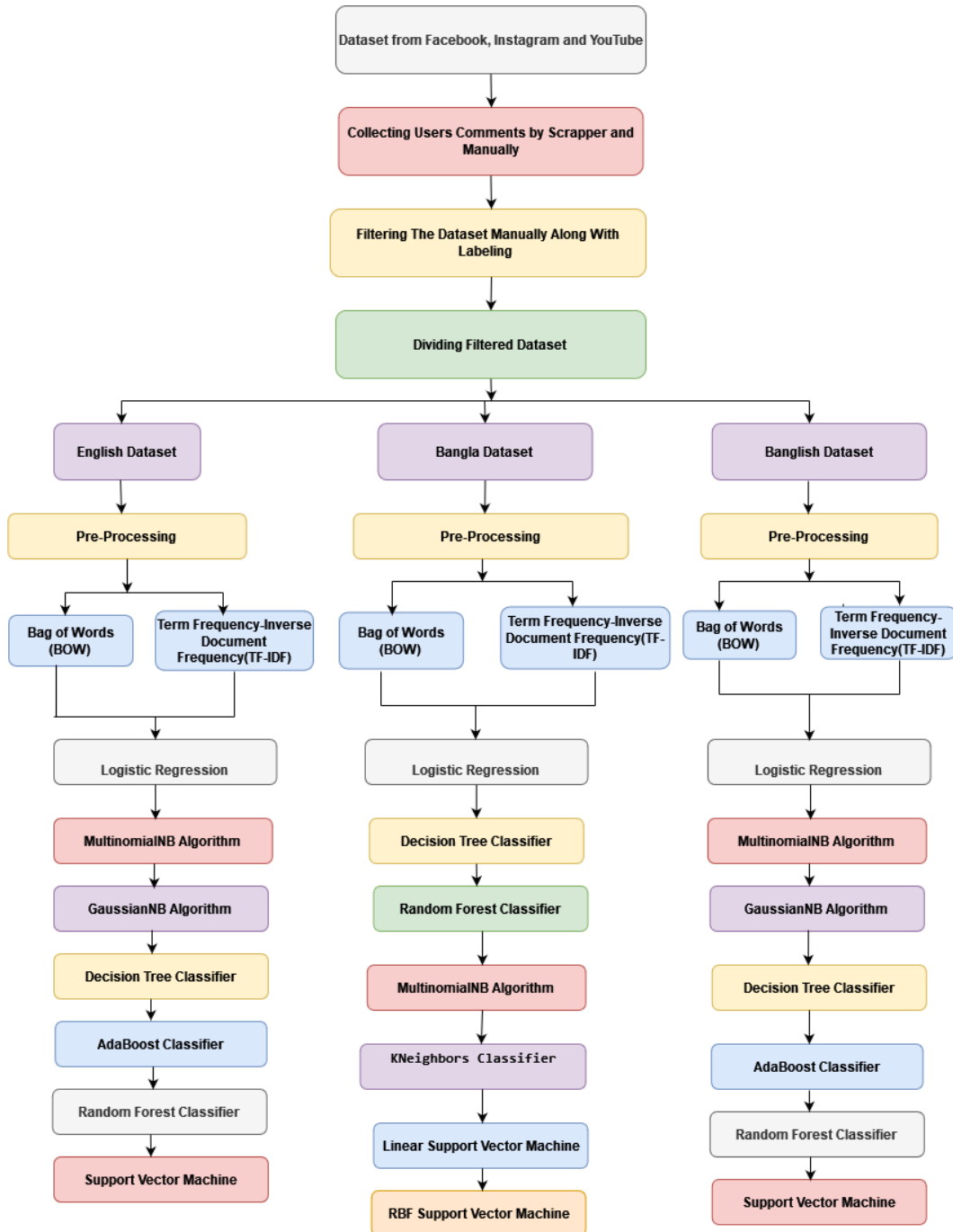


Figure 4.1: Methodology

## 4.1 Classifiers

### 4.1.1 GaussianNB

GaussianNB is an abbreviation for "Gaussian Naive Bayes," which is the full version of the acronym. One variant of the Naive Bayes algorithm is a set of supervised machine learning classification algorithms based on Bayes' theorem. This technique's usefulness lies even though it is a very straightforward classification method. Furthermore, conditional probability is calculated using the Bayes Theorem. This theorem is a valuable tool in the study of probability. It is also used in machine learning. Multinomial Naive Bayes, Gaussian Naive Bayes, and Bernoulli Naive Bayes are the three types of Naive Bayes algorithms.

One variant of the Naive Bayes algorithm is called Gaussian Naive Bayes. Specifically, GaussianNB will be utilized on features that take in continuous values or inputs. Furthermore, all of the characteristics adhere to a standard or Gaussian distribution. To simplify the modeling process, let us assume the data follows a Gaussian distribution with no co-variance (independent dimensions) between them. This model may be fitted using the mean and standard deviation of the points associated with each label. To define the Gaussian distribution, that is all that is required. Features and models with continuous values and a Gaussian (or standard) distribution operate best with this method.

For simplicity, while dealing with continuous data, we often assume that the corresponding continuous values for each class follow a standard (or Gaussian) distribution. It can be shown that the probability density function for a Gaussian distribution is-

$$f(x) = \frac{1}{\sigma(\sqrt{2\pi})} e^{-\frac{1}{2} \left(\frac{x - \mu}{\sigma}\right)^2} \quad (4.1)$$

The parameter  $\mu$  here stands for the distribution's mean or expectation (as well as its median and mode), while the parameters  $\sigma$  and  $\sigma^2$  represent the distribution's standard deviation and variance, respectively. The graph of this Gaussian function is often bell-shaped.

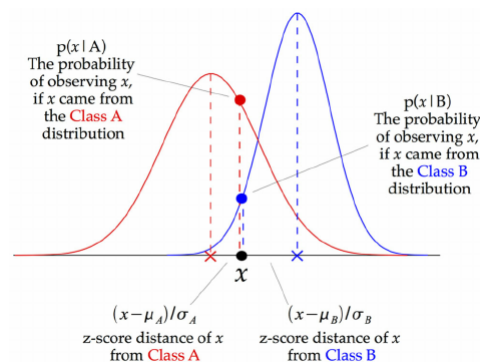


Figure 4.2: Illustration of how a Gaussian Naive Bayes (GNB) classifier works[24]

As seen in the above graph, a Gaussian Naive Bayes (GaussianNB) classifier operates as described. To do this, we divide the distance between each data point and the mean of each class by its standard deviation, or  $z$ -score. However, when dealing with numerous input dimensions, a significant difference between GaussianNB and other classifiers becomes apparent. Various classifiers take inter-dimensional covariance into consideration, but the GaussianNB does not. Now we can see that the Gaussian Naive Bayes has a somewhat different way of arriving at a result but can still be used successfully.

Due to the nature of the data we are dealing with, the sentences themselves cannot be fed into our classifier. Consequently, we use the terms frequency-TF and the total frequency-TF of a word in all text datasets (inverse document frequency-IDF). Then, we may potentially construct feature vectors that stand in for texts and include the probabilities of phrases inside those texts that belong to a specific category. The computer can determine the probability that the text fits into each of the specified buckets. The most plausible category is then selected to be used for content classification.

### 4.1.2 Decision Tree Classifier (DT)

A decision tree classifier is used for statistical, data mining, and machine learning predictive modeling. This is, in fact, a regulated machine learning algorithm. Like humans, it follows guidelines when deciding what to do. In this procedure, a decision tree is built using a classification model. As a bonus, it may be used to solve classification and regression issues. In this classifier organized as a tree, the inner nodes stand in for properties of the dataset, the branches for the rules used to make classifications, and the leaf nodes for the results.

There are two distinct kinds of nodes in a decision tree: the leaf node and the decision node. Right here at this decision node, there is still one more test that must be run before a conclusion can be reached. The characteristics of the data set are used to make the determination or test. Any given decision node can lead to more than one branch, but leaf nodes are what happens when a decision node has no children. The CART Algorithm (Classification and Regression Tree Algorithm) generates a decision tree from a dataset. Given a dataset at the root node of the decision tree, it is extended on extra branches from the root and seems to be a tree structure. Using information from the input dataset, a decision node asks a question and splits the tree into subtrees and leaf nodes based on the answer (Yes or No).

A decision tree is a recursive algorithm that has the advantage of sharing decision-making logic with other recursive algorithms, such as neural networks. The temporal complexity of decision trees depends on the number of records and at-attributes in a dataset. The decision tree makes no assumptions about probability. It gets a high level of accuracy when working with data that has a lot of dimensions.

As zero is the most useful characteristic, decision trees employ attribute selection measures (ASM). Subsets of the dataset are then created, with each property serving as a decision node. Next, it uses a portion of the dataset to generate new decision



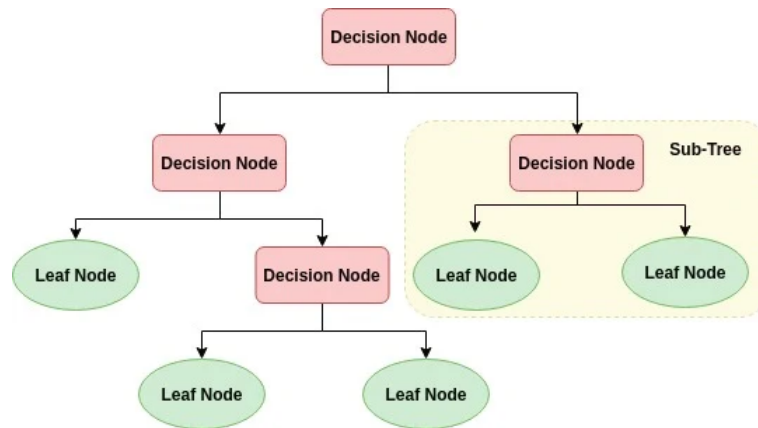


Figure 4.3: Decision Tree Classifier [26]

nodes and so on until it reaches a point where it can no longer categorize nodes, at which point it stops, and the remaining nodes are called leaf nodes. Since we are dealing with textual data, we can simply insert a decision tree classifier into our dataset to categorize the information based on a selected feature’s importance. However, the decision tree’s complexity may increase if it has many branches. Still, it has the potential to be very helpful in cases of decision-making difficulty, and it is easy to understand since it employs the same decision-making process that people use.

### 4.1.3 AdaBoost

AdaBoost (Adaptive Boosting) is a well-known boosting method that combines many classification techniques in such a way that each classification technique works to enhance the categorization of data that was improperly classified by the weak classifier that came before it. It’s a collaborative effort to develop and implement a tree hierarchy in a systematic manner. A fragile classifier is constructed out of the training set and the weighted samples. Its effectiveness stems from a number of factors, including its ease of use, real-time detection performance, feature selection, and short training time. When dealing with binary classification problems, AdaBoost is often employed to boost the efficiency of decision trees. By boosting, we might be able to make a weaker classifier stronger by linking together several weak ones.

Objects assigned by a poor classifier perform better than those assigned by random chance, but the classifier still has problems. ”Stump” refers to these feeble discriminators. A little tree with just two branches is called a stump. Boosting methods utilize ”stump” decision trees since they tend to be shallow models that do not overfit but may be biased. Each tree is instructed to look for problems only in the trees that came before it. Some stumps are also given a greater role in the classification scheme. If a sample was incorrectly labeled by the previous tree, its importance would be raised so that the following tree may concentrate on correctly labeling that sample. To summarize, AdaBoost excels on imbalanced data. Yet, it struggles to function well in noisy environments. Unlike other neural network architectures, AdaBoost is more challenging to train.

It is clear that AdaBoost creates numerous stumps from a single decision tree, giving more weight to each choice while giving a single stump less of a role in the final decision. At first, we had the sample weighted evenly.

All of the mistakes made on a prior stump will carry over to the one after it. We also evaluate the effectiveness of the stumps as classifiers.

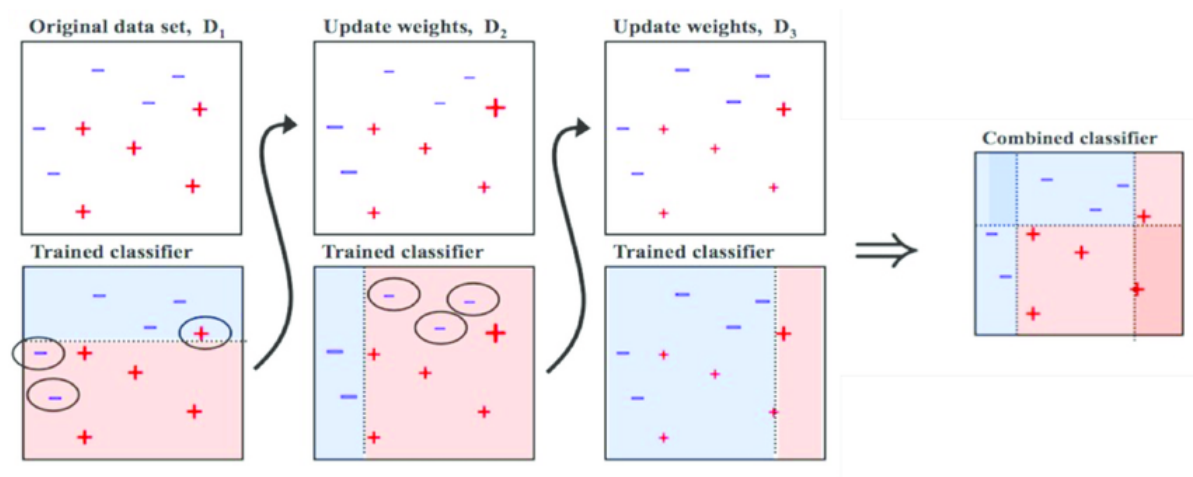


Figure 4.4: AdaBoost Working Procedure [27]

In the first step, we'll use  $N$  to represent the number of rows in our collection.

$$x_i \in \mathbb{R}^n, y_i \in \{-1, 1\} \quad (4.2)$$

Our data has a property called  $n$ . The group of numbers is denoted by  $X$ . The value of  $Y$  will tell us whether or not the grouping is suitable. As we all know, when AdaBoost was first implemented, it assigned the same value to each training sample in the dataset. The training set is more likely to be influenced by that collection of data points when the assigned weights are large. Similarly, the training dataset is not much altered when the weights are set quite low.

$$w = \frac{1}{N} \in [0, 1] \quad (4.3)$$

The value of  $w$  represents the importance of each piece of information. If there are 5,000 records, then each record counts for one-fifth of a percent. So, the range of weight will be 0–1 now.

$$\alpha_t = \frac{1}{2} \ln \frac{1 - TotalError}{TotalError} \quad (4.4)$$

Here, alpha is the contribution of a single stump to the final classification, and total error is the number of wrong predictions divided by the number of training sets.

#### 4.1.4 Logistic Regression (LR)

Logistic regression is only one of several techniques for solving classification issues that can be found in the field of machine learning. We have here a model for supervised classification. As its name suggests, this method classifies information in a dataset. Binary logistic regression and multinomial logistic regression are the two primary categories of this kind of statistical analysis. When working with binary data, logistic regression may be used. Multinomial logistic regression is employed in this situation since there are more than two classes in the underlying data. Since there are more than two categories in our data, we'll discuss multinomial logistic regression. Additionally known as multiclass logistic regression, softmax regression, polytomous logistic regression, and multinomial logit, multinomial logistic regression goes by a few other names as well. Multinomial logistic regression has additional categories similar to the logistic regression model. However, this allows us to make predictions about data classes when there are more than two possible categories.

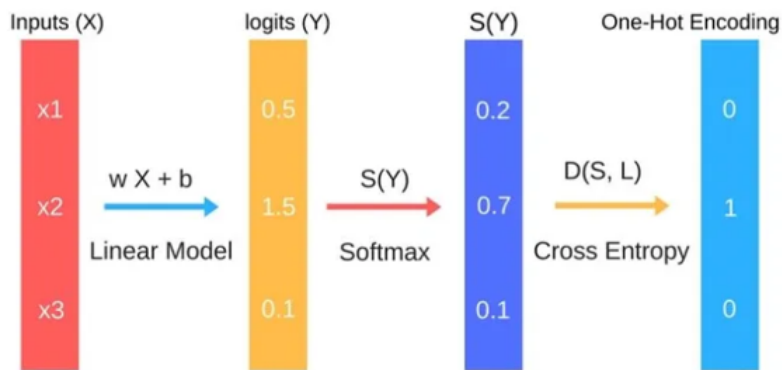


Figure 4.5: Multinomial Logistic Regression Classifier [28]

The model allows for the categorical prediction of a response from a set of dependent factors using a set of independent variables. Make sure your dependent variables are nominal or ordinal before using this model. Our research relies on a dataset with nominal values for its dependent variables, which implies that the categories within the dataset are referred to by their actual names. Multinomial logistic regression offers two potential solutions to this problem. The first is the concept of "k-class models." As the image below demonstrates, we divide our students into three distinct groups. As a result, we built three distinct models for use in three distinct categories. Any given model may be broken down into two sections: itself and all other classes. After that, we'll be able to make educated guesses about the probable result by calculating the probability involved.



Figure 4.6: Example of "k" models for k class

"Simultaneous Models" is the second. By doing so, we may create k1 models from the k classes. Let's make three groups, for which we'll create two models. Using the following formula, we can create a probability equation for the A-class.

$$\frac{p(A)}{p(C)} = a1 + b1x1 + \dots + bnxn \quad (4.5)$$

$$\frac{p(A)}{p(C)} = \exp(a1 + b1x1 + \dots + bnxn) \quad (4.6)$$

$$p(A) = p(C) * \exp(a1 + b1x1 + \dots + bnxn) \quad (4.7)$$

As with class A, we also apply to class B as shown below.

$$\frac{p(B)}{p(C)} = a2 + b1x1 + \dots + bnxn \quad (4.8)$$

$$\frac{p(B)}{p(C)} = \exp(a2 + b1x1 + \dots + bnxn) \quad (4.9)$$

$$p(B) = p(C) * \exp(a2 + b1x1 + \dots + bnxn) \quad (4.10)$$

Since

$$p(A) + p(B) + p(C) = 1 \quad (4.11)$$

then

$$p(C) * \exp.\exp(a1 + b1x1 + \dots + bnxn) + p(C) * \exp.\exp(a2 + b1x1 + \dots + bnxn) + p(C) = 1 \quad (4.12)$$

$$p(C) = \frac{1}{1 + \exp(a1 + b1x1 + \dots + bnxn) + \exp.\exp(a2 + b1x1 + \dots + bnxn)} \quad (4.13)$$

Finally, we obtain the expected value for the data.

## 4.1.5 Random Forest Classifier (RF)

For supervised machine learning, random forest utilizes several decision trees. During the training phase of this method, the algorithm generates a dense collection of decision trees. The most popular label chosen by the tree's branches is the result. In order to make an accurate forecast, a random forest builds a forest of decision trees and averages their results, with the majority tree being the one chosen. Random forest classifiers have dual use, applying to both classification and regression.

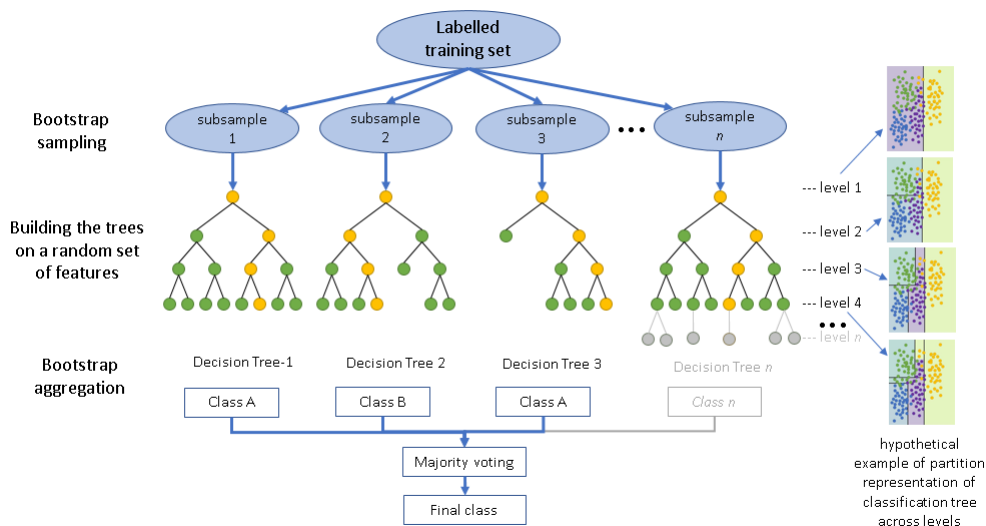


Figure 4.7: Choosing Decisions Using Random Forest

The random forest classifier is similar to a decision tree. However, the approach here introduces a small quantity of unpredictability as it traverses the decision tree. In a dataset, features represent a subset of the whole data. A dataset's final classification is determined by the characteristics it contains. The best characteristics are chosen randomly, and the algorithm begins to span the tree. Instead, the decision tree is utilized to zero in on the most critical factors that go into the final classification of a given data set. This yields a more varied collection of findings, which improves the final product. Growing the forest or tree canopy is possible even larger. In the end, the most accurate categorizations are picked as the outcome. For lack of data, random forests are also a great option.

For instance, the algorithm may choose to span a tree depending on random properties when training randomly. A separate category may be gleaned from each fork in the tree. The algorithm will then choose the most widely accepted classification from among those generated by the trees. Finally, the majority vote will decide how to classify the data using those trees. Since a random forest classifier integrates the output of several random trees, it is naturally resistant to overfitting. Overfitting is the practice of considering irrelevant information from a dataset. Extensive data sets can provide reliable outcomes. The main advantage of random forest is that it does not need scaling and can maintain adequate accuracy even if such data is absent. Several problems might arise when using a random forest classifier. The random forest method might shift if the input is altered even a little. One major

issue is the complexity of this algorithm. Compared with other algorithms, Random Forest is more involved. Also, training the model takes longer because of the vast number of trees that are involved.

#### 4.1.6 Multinomial Naive Bayes (MNB)

Multinomial Naive Bayes is a popular probabilistic learning approach in Natural Language Processing (NLP) for understanding the structure of human languages (NLP). Using Bayes' theorem, a text-tagging system may be able to predict the tag for a document, such as a message, email, or article. If a sample contains many tags, the algorithm determines which tag has the best chance of being chosen and returns that information. Multinomial NB classifiers are widely employed in data analysis and are a fan-favorite classifier.

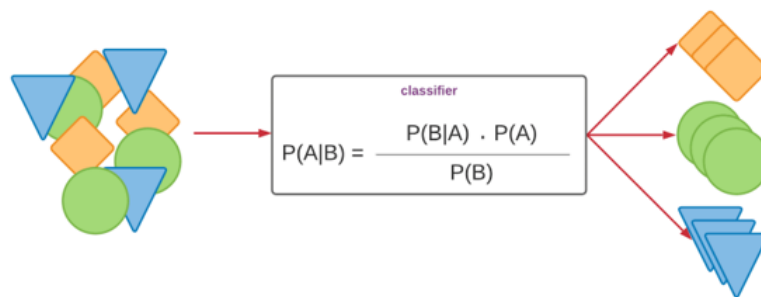


Figure 4.8: Multinomial Naive Bayes Classifier[29]

Naive Bayes is an effective approach for analyzing text input and solving problems with many different classes. The Naive Bayes theorem relies on the Bayes theorem concept, so it's important to understand it first. Thomas Bayes is credited with developing this theorem.

An applicable mathematical formula is:

$$P(A \setminus B) = \frac{P(A) * P(B \setminus A)}{P(B)} \quad (4.14)$$

$P(A)$  refers to the probability that occurs in  $A$  in the document.

$P(B)$  refers to the probability that occurs in  $B$  in the document.

The probability of events occurring on a document is denoted by the symbol  $P(A \setminus B)$ . (Depending on the circumstance connected to the incident).

$P(B \setminus A)$  is the probability that the prediction of  $B$  will occur in the class of  $A$ . This formula is applied to determine the probability that a tag will exist in a document.

#### 4.1.7 Support Vector Machine (SVM)

SVM's full name, Support Vector Machine, is more complicated. One of the most popular ML algorithms. Classification and regression issues are typical applications.

This method may be put to use in various contexts, including but not limited to face recognition, image labeling, text classification, and many more. We have settled on this approach because of the nature of the data we are working with: text. As our data consists mainly of texts, we have decided to use this technique.

To make it simpler to add new data points in the desired format in the long run, the SVM approach seeks to uncover the optimal route or choices limit for categorizing the supplied collection of data into classifications. For instance, suppose we use a categorized data set to train a support vector machine.

The classification of data will become possible after training. A hyperplane represents the ideal choice boundary.

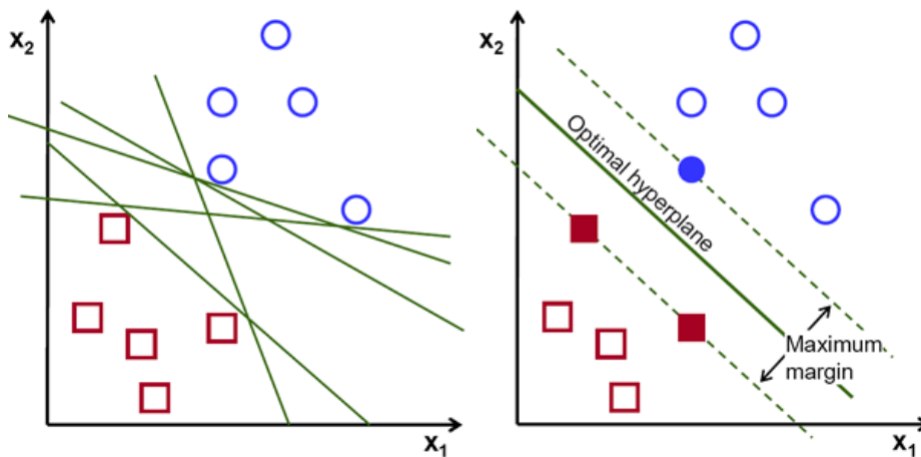


Figure 4.9: Graph of SVM [30]

Selecting extrema, either points or vectors, produces a hyperplane. When referring to such outlying points or vectors, the phrase "support vectors" is often used. The classification tool is referred to as a Support Vector Machine. The ideal hyperplane is being used to divide the world into two distinct categories in the image above.

As far as we know, there are two primary varieties of SVM. Linear SVM and Non-Linear SVM are the two types. The first is often used when there are just two classes that a straight line may arbitrarily split. The second is used if a linear model fails to provide an appropriate classification for the data. As we go further, we will examine how to choose the best hyperplane for a given data collection.

Here, we will utilize Linear SVM to determine which of the two hyperplanes above best categorizes the data. Now, in the diagram below, we will choose the hyperplane that dramatically separates the nearest data points to each group.

A star belonging to the opposite group is also visible in the previous image. To improve profitability, SVM will often disregard such an outlier in this situation.

Non-Linear SVM is required here. In this case, the circle equation will be used by SVM to classify the data. When searching for a hyperplane, SVM employs a novel strategy. Kernel(finds a pattern) describes this process. Before deciding how to divide the information based on the labels or outputs you supply, it performs a series of complex transformations on the data.

## Linear Support Vector Machine (SVM)

The Support Vector Machine, or SVM for short, is a kind of linear model that is used to address problems involving classification and regression. It is effective for a wide variety of problems that arise in the real world and has the capability of solving linear as well as non-linear challenges. The SVM is a basic idea. The method will output either a line or a hyperplane that splits the data into classes.

## RBF Support Vector Machine (SVM)

The RBF kernel (radial basis function kernel) is a popular kernel function in the area of machine learning. It may be implemented in several kernelized learning methods. Support vector machine classification, when it is used often, benefits greatly from its employment. The RBF Kernel has been widely adopted because of its resemblance to the K-Nearest Neighbors algorithm. Since RBF Kernel Support Vector Machines only need to store the support vectors instead of the whole dataset during training, they solve the issue of space complexity while still providing the benefits of K-NN.

### 4.1.8 k-Nearest Neighbors (KNN)

K-Nearest Neighbor is one of the most approachable Machine Learning algorithms since it is based on the Supervised Learning technique. K-Nearest Neighbors (K-NN) is a technique for data classification that uses the idea of similarities between newly encountered cases and previously labeled ones. When deciding how to classify a new data point, the K-Nearest Neighbors (K-NN) method considers all of the previously collected data. This suggests that the K- NN technique may be used to classify newly supplied data into an appropriate category efficiently. The K-NN method may be used for Regression in addition to Classification. Since K-NN is non-parametric, it makes no assumptions about the data it analyzes. This technique is commonly referred to as a "lazy learner algorithm" since it delays applying what it has learned from the training set. KNN stores the dataset during its training phase, and then, when presented with new data, it places it in the category to which it most closely belongs based on the primary data.

If we have two groups, say A and B, and we have a new data point  $x_1$ , we may ask which of the two groups  $x_1$  belongs to. A K-NN method is required for this kind of issue. K-NN is helpful for quickly and accurately determining a dataset's class or category. Take a look at the diagram below:

It is possible to illustrate how K-NN works by using the following algorithm:

To begin, choose K as the number of neighbors and figure out their average Euclidean distance. Next, choose K neighbors from the set whose Euclidean distances are the shortest. Next, determine how many data points there are in each group among these k neighbors. Next, put the new information into the cluster where there are the most neighbors. Now, at long last, our model is complete.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



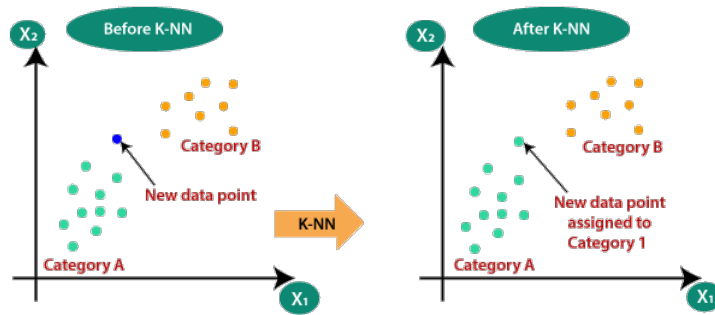


Figure 4.10: Graph of KNN [31]

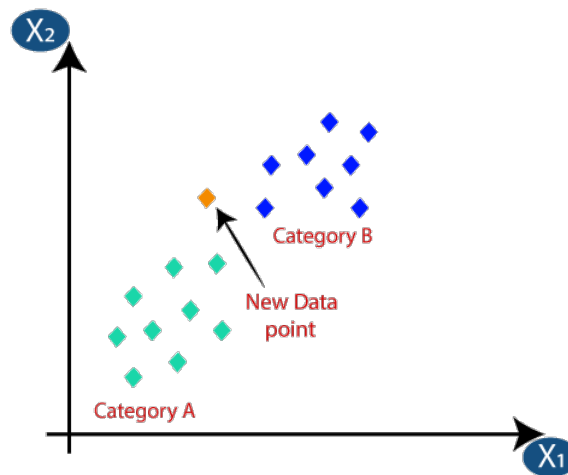


Figure 4.11: KNN Scenario-1 [31]

To begin, let's decide how many neighbors to have, so let's set  $k=5$ . The Euclidean distance will then be determined between each pair of points in the data. We have previously covered the Euclidean distance between two locations in our geometry classes. Specifically, it may be determined by using the formula:

We found the closest neighbors using the Euclidean distance formula, using three neighbors in group A and two neighbors in group B. Here's an example: Since the three closest neighbors all come from the same category (A), we may deduce that this new data point also belongs to the same category.

Consider the following before settling on a value for  $K$  to implement in the K-NN algorithm. No one has yet discovered a surefire way to determine the ideal value for " $K$ ," so we'll have to try out many possibilities to see which ones perform best. Let's assume for simplicity's sake that 5 is the most well-liked option for  $K$ . Small values of  $K$ , such as  $K = 1$  or  $K = 2$ , may increase the likelihood of noise and the significance of outliers. However, it may run into issues, even if  $K$  is large.

In a few situations, the KNN Algorithm may be useful. It's simple to implement in practice. It is resilient enough to survive the stormy waters of the training data. More data for training might improve performance. The KNN algorithm is not without its flaws. In all situations, the value of  $K$  must be established, which

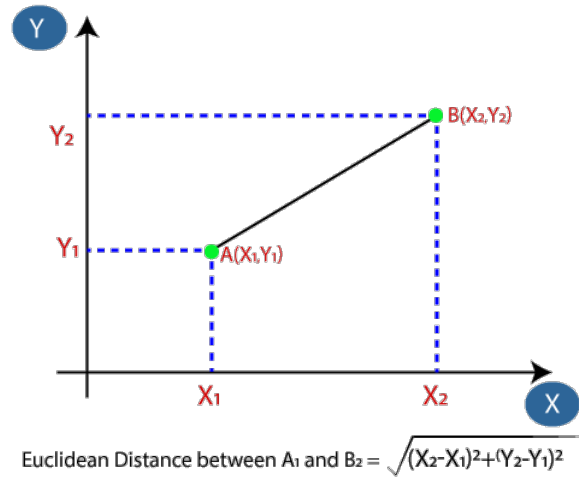


Figure 4.12: KNN Scenario-2 [31]

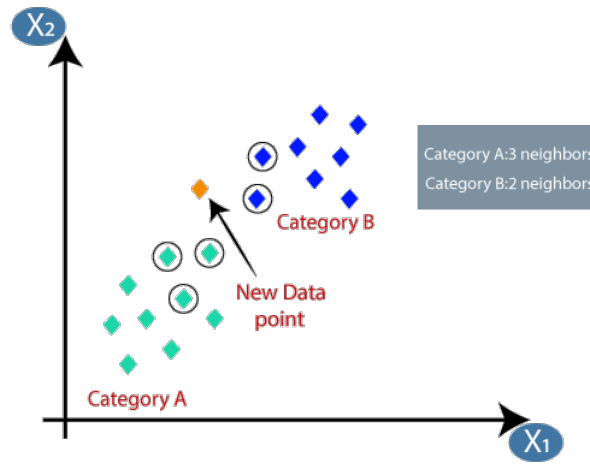


Figure 4.13: KNN Scenario-3 [31]

might be challenging. The computational cost is high because we must calculate the distance between every data point in all the training samples.

## 4.2 Features

### 4.2.1 N-Gram

An n-gram is a string of words that occurs consecutively in a text, where n is the number of words in the string. An n-gram is a string of n characters (letters, digits, symbols, and/or punctuation) found in a text document.

N-gram models are useful for sentiment analysis, text classification, and text generation, just to name a few text analytic applications where word sequences play a significant role. If N is 1, then the specified text has a single word; if N is 2, then the specified text contains a pair of words.

We used three distinct varieties of N-grams in our categorization, and found that they yielded notably diverse results. These N-grams with varying values of N are shown with respect to the sentence "I like to undertake research" [4].

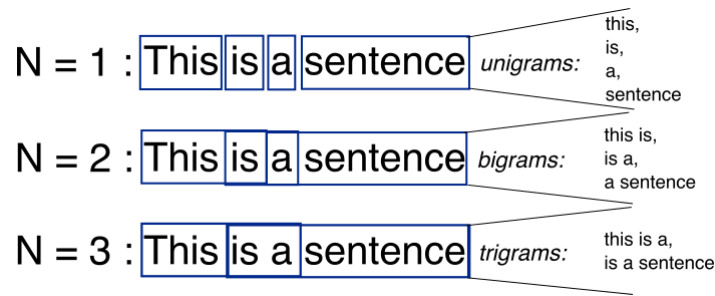


Figure 4.14: N-Gram [32]

## 4.2.2 Unigram Feature

Simply setting  $n = 1$  to the NLTK ngrams function will produce unigrams. The sentence is tokenized before being given to the ngrams function. Each tuple in the Unigram model consists of a single word. Every word in a Unigram appears on its own, regardless of the word that came before it. As a result, each word functions as a grammatical component in the sentence. For example, I, I ate, and banana is the three features we will receive for the unigram, and they are all distinct from one another. Real languages, however, do not operate this way.

## 4.2.3 Bigram Feature

We used NLTK's ngrams function to generate bigrams by setting  $n$  to 2. To begin, we used the ngrams function on the tokens we had extracted from the phrase. Each tuple in the Bigrams model consists of two neighboring words.

## 4.2.4 Trigram Feature

Trigrams are handled by telling NLTK's ngrams function to use length 3. The sentence is tokenized before being given to the ngrams function. Each tuple in the Trigram model consists of three words.

## 4.2.5 Comparison between the N-gram Features

Unigram, bigram, and trigram are different types of features that can be used to represent text data. The difference in the output results is due to the way they represent the text data.

Unigrams are single words, and it only captures the individual words in the text. This is the simplest way to represent the text data and it does not take into account any relationship between the words. Unigrams can provide a good understanding of the individual words in the text and their relative frequencies.

Bigrams are pairs of words. It captures the relationship between two consecutive words in the text. Bigrams can provide a better understanding of the context in which words appear. They can also provide useful information about collocations

(frequent word pairs) in the text.

Trigrams are a group of three consecutive words. It captures the relationship between three consecutive words in the text. Trigrams are more complex than unigrams and bigrams and can provide a more detailed understanding of the context in which words appear.

In general, unigrams provide a good understanding of the individual words and their relative frequencies in the text, bigrams provide a better understanding of the context in which words appear, and trigrams provide a more detailed understanding of the context in which words appear. The choice of features depends on the problem you are trying to solve and the type of information you want to extract from the text.

# Chapter 5

## Result Analysis

### 5.1 Precision, Recall and F1 Score

There are a number of ways to characterize the outcome of models, including precision, recall, F1 score, and confusion matrix. Overlapping but distinct methods are used to compare the anticipated value to the actual labeled values. Before trying to analyze the results of performance measurement methods, it is important to know what they are and how they work.

**True Positives:** If the labeled value and the anticipated value match up, we have a true positive. To put it another way, this may be thought of as the accurate prediction of positive values. If both the predicted and labeled values are "Negative," for instance, we say that the prediction is accurate (TP).

**True Negative:** Values that were correctly predicted to be negative are called "true negatives." If both the predicted value and the labeled value are negative, we may call it a true negative (TN). If both the anticipated and labeled values are "Positive," for instance, we classify this as a true negative.

**False Positives:** In a similar vein, a false positive result would be obtained if the expected value were yes, but the class was in fact not positive (FP). It is a false positive if the real class is "Positive" but the projected value is "Negative," for instance.

**False Negatives:** When the class label is "yes," but the projected value is "negative," something is wrong with the labeling. As an example, a false negative would occur if the label was "Negative" but the prediction was "Positive" (FN).

**Accuracy:** This is the most straight-forward method of evaluating efficiency. This is determined by contrasting the forecast with the actual results. It reveals the extent to which the model accurately anticipates the types of data.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (5.1)$$

**Recall:** When all outcomes are favorable, recall reveals what percentage of predictions were accurate. Here is the formula:

$$Recall = \frac{TP}{TP + FN} \quad (5.2)$$

**Precision:** Accuracy reveals the true extent to which all forecasts are correct. Here is the formula:

$$Precision = \frac{TP}{TP + FP} \quad (5.3)$$

**F1 Score:** As a measure of a classifier’s efficacy, F1 Score takes into account both precision and recall. The outcome is an accumulation of false positives and false negatives. As opposed to accuracy, F1 Score is preferable in situations when classes are not distributed uniformly. When the cost of a false positive and a false negative are equal, accuracy works very well.

$$F1Score = \frac{2(Recall * Precision)}{Recall + Precision} \quad (5.4)$$

## 5.2 Result of English Dataset

Our English comment data is divided into three distinct categories. First, we categorize them as "Positive," "Negative," or "Sarcastic." Each comment undergoes manual classification. To tokenize or vectorize the comments, for example, we employ two distinct vectorization algorithms (data). For tokenization, we use the TF-IDF and Bag of Words (BOW) techniques. The Logistic Algorithm, Multinomial Naive Bayes, GaussianNB, DT, RF, AdaBoost Classifier, and SVM are among the algorithms that have been used. We use BOW and TF-IDF vectorization to determine the accuracy of multiclass classifications. The dataset is essentially separated into two sections. In the table below, the classification results are organized by tokenization method and the result of accuracy.

### 5.2.1 Bag of Word tokenized English Dataset

Performance Table for Bag Of Words (BOW)				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	0.763	0.76	0.71	0.73
Multinomial Naive Bayes (MNB)	0.74	0.73	0.69	0.70
GaussianNB Algorithm	0.61	0.62	0.61	0.58
Decision Tree Classifier	0.54	0.78	0.47	0.45
AdaBoost Classifier	0.679	0.71	0.63	0.64
Random Forest Classifier	0.732	0.74	0.69	0.70
Support Vector Machine (SVM)	0.734	0.72	0.69	0.70

To tokenize the data in a multiclass dataset, the Bag of Words technique is used. The classification algorithm for Logistic Regression obtains the best accuracy score, 0.763. This is an outstanding outcome for our dataset. Multinomial Naive Bayes Algorithm takes second place with a score of 0.74. The accuracy of SVM and Random Forest, on the other hand, was 0.734 and 0.732, respectively. With an accuracy of 0.54, Decision Tree Classifier is the least accurate classifier. The accuracy of both the Adaboost Classifier and the Gaussian Naive Bayes is 0.679 and 0.61 respectively. Therefore, logistic regression is the optimal classifier. As compared to other classifiers, Decision Tree Classifier (DT) achieves the highest precision. The precision value of the Decision Tree Classifier (DT) is 0.78. The precision value for this classifier on this dataset is quite good. This means that the false-positive rate of this model is quite small. With a precision of 0.76, Logistic Regression (LR) ranks second highest. The precision value of 0.74 is good for third place, and it belongs to Random Forest Classifier (RF). While the Support Vector Machine (SVM) has a precision of 0.72, the Multinomial Naive Bayes (MNB) is slightly higher at 0.73. Adaboost Classifier has a precision value of 0.71. On the other hand, the 0.62 precision scores achieved by the Gaussian Naive Bayes Algorithm (GaussianNB) are quite disappointing. This dismal result shows that many classes were misidentified, as the score is so low. The 0.71 recall score achieved by Logistic Regression (LR) is outstanding. But all three of the most popular classifiers—MNB, SVM and RF—share an excellent recall value of 0.69. MultinomialNB, LR, RF and SVM appear to have generated the most pertinent results. Adaboost Classifier and GaussianNB have the recall value of 0.63 and 0.61. The anticipated results from using these models have materialized. However, Decision Tree Classifier (DT) has lower recall values, at 0.47. When compared to other classification methods, Decision Tree Classifier (DT) has a low recall. The resulting F1 score is 0.45, which is below 50%. Therefore, it does not qualify for this dataset. The F1 score for Gaussian Naive Bayes is also lower, at 0.58. In contrast, the Adaboost Classifier has F1 scores of 0.64. Similarly, RF,

MNB and SVM have a value of 0.70. The LR has the highest F1 Score at 0.73. In contrast to the other classifiers, Decision Tree has the lowest predicted F1 score of 0.45. Gaussian Naive Bayes (GaussianNB) has the same value of recall and accuracy values. The F1 score is 0.45, which is lower than anticipated. It is therefore ineligible for this dataset.

From the data, we can deduce that LR has the best accuracy (0.763), while DT has the lowest (0.54). The accuracy of the logistic regression classifier and the decision tree classifier vary for many reasons, such as the type of data being utilized, the algorithms' individual implementations, and the model's complexity.

Since it is a linear model, logistic regression uses a linear combination of the input features to create a prediction. Due to its ease of use and interpretability, this technique trains quickly and works well with huge datasets. When an issue can be broken down into two categories, logistic regression excels.

Whereas, decision tree classifiers construct a model of decisions using input features. As the input features are evaluated, the data is repeatedly partitioned into smaller groups. The decision tree classifier works well for both binary and multi-class classification problems and can process both numerical and categorical input. Overfitting is a problem for decision trees, because they are quite sensitive to variations in the training set.

Linearly separable issues are ideal for logistic regression, while complicated, non-linear problems are better tackled using decision trees. As a result of its more natural fit to the problem and the data, the logistic regression classifier is likely to have outperformed the decision tree classifier in this scenario.

## 5.2.2 TF-IDF tokenized English Dataset

Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	0.75	0.78	0.67	0.69
Multinomial Naive Bayes (MNB)	0.73	0.79	0.65	0.67
Gaussian Naive Bayes Algorithm	0.59	0.60	0.60	0.56
Decision Tree Classifier (DT)	0.54	0.78	0.47	0.44
AdaBoost Classifier	0.681	0.72	0.63	0.64
Random Forest Classifier (RF)	0.732	0.76	0.68	0.70
Support Vector Machine (SVM)	0.76	0.78	0.69	0.71

Using the TF-IDF algorithm, the data from a multiclass dataset is tokenized. The SVM classifier has a reasonable accuracy score of 0.76, as seen. That is an outstanding result for a multiclass dataset. The Logistic Regression comes in second with a



score of 0.75. In contrast, the accuracy of Random Forest Classifier and Multinomial Naive Bayes Algorithm is 0.732 and 0.73, respectively. With an accuracy of 0.54, the Decision Tree Classifier is the least dependable. The Gaussian Naive Bayes has an accuracy of 0.59, whereas the Adaboost Classifier has an accuracy of 0.681. The Support Vector Machine is thus the best classifier, if the English comments dataset is tokenized using the Term Frequency Inverse Document Frequency technique. The maximum value of precision, 0.79, is obtained by using the Multinomial Naive Bayes (MNB) algorithm for multiclass. This indicates we get 79% genuine positive values among the predictable positive values here. With a recall value of 0.69, the Support Vector Machine (SVM) has the highest true positive rate of any other classifier. Last but not least, SVM provides the greatest F1 score (0.71), telling us the true positive values that contain negative comments. That's why it's a good model. Decision Tree Classifier (DT), Logistic Regression (LR) and Support Vector Machine (SVM) classifier has the same precision value which is 0.78. This gives the second highest value of precision. Both the Random Forest Classifier (RF) and the AdaBoost Classifier have precision values of 0.76 and 0.72, respectively, which are above 50%. This indicates that by using these classifiers, we are able to distinguish better actual positive values from expected positive ones. Additionally, the Gaussian Naive Bayes algorithm yields the lowest actual positive values among the predictably positive values with a precision score of 0.60. Random Forest Classifier produces the same recall value of 0.68. So, the second highest recall values are provided by the Random Forest Classifier. We obtain recall values of 0.67, 0.65, 0.63 and 0.60 from Logistic Regression (LR), Multinomial Naive Bayes (MNB), Adaboost Classifier and Gaussian Naive Bayes Algorithm respectively. These classifiers achieve a rate greater than fifty percent, which is the proportion of real positive values relative to all other values. And the lowest recall value that Decision Tree Classifier (DT) provides is 0.47. This indicates that the Decision Tree Classifier (DT) has the lowest actual positive value rate compared to all other values in the dataset. The F1 Score for the Random Forest Classifier (RF) is 0.70. So, the Random Forest Classifier (RF) has the second-highest F1 Scores. The F1 Score for the Logistic Regression (LR), Multinomial Naive Bayes (MNB) and the Adaboost Classifier (RF) is 0.69, 0.67 and 0.64 respectively. The f1 score is the average of the values for precision and recall. Because of this, we can say that the Support Vector Machine (SVM), Logistic Regression (LR), Multinomial Naive Bayes (MNB), the Adaboost Classifier and the Random Forest Classifier (RF) are all good models. Gaussian Naive Bayes gives an F1 Score of 0.56, which is more than 50%. After that, we use the Decision Tree Classifier (DT) gives us the lowest F1 Score, which is 0.44, which is more than 50%.

From the data, we can determine that SVM has the best accuracy (0.76) while DT has the lowest (0.54). Two supervised machine learning methods that can be utilized for classification problems are the Support Vector Machine (SVM) and the decision tree classifier.

SVMs are a sort of linear classifier that seeks to determine the best boundary (or hyperplane) to separate the different classes in the data. They are particularly successful in high-dimensional areas and can also be utilized for non-linear classification by using kernels. When the data is clean and the classes are clearly defined, SVM classifiers typically perform well.

To classify data, decision tree classifiers iteratively divide the input features into

subsets based on their values. They produce a tree-like structure where each internal node represents a feature and each leaf node represents a class label. You can use a decision tree classifier for a variety of classification issues, from simple binary classification to complex multi-class classification. Overfitting is a problem for decision trees because they are quite sensitive to variations in the training set. Here, it is likely that the SVM classifier outperformed the decision tree classifier since it was better suited to the task and the data. However, this may not be the case for every datasets, thus it is vital to examine different models and choose the one that performs best for a certain task.

### 5.2.3 Score Generation

Each comment has been examined and its sentiment score has been computed during this step. To produce the sentiment score, we manually categorized the dataset as positive, negative, or sarcastic. Then, using 1204 positive, 1415 negative, and 477 sarcastic polarity scores, we matched the dataset to the machine's polarity scores by using Sentiment Intensity Analyzer and produced sentiment scores for each line. The score is 0.45704134366925064.

### 5.2.4 Prediction

Here, we perform the computation for the predicted comments. This algorithm aims to generate a single forecast. For this comment, the Counter Vectorizer transform method is utilized. Using this, we invoke the predict method, which provides a value based on the numerical labels used to label the data in our dataset. The prediction result for the remark "nice video" is `array([1])`, suggesting that it is a positive comment, but the forecast result for the comment "you are disgusting" is `array([0])`, indicating that it is a negative comment. In addition, the likelihood (score) of success in forecasting is computed. For example, the comment "nice video" returns `array([[0.09337054, 0.89518656, 0.0114429]])`, where the highest score, 0.89518656, corresponds to `array([1])` because our class is specified as `array([0, 1, 2])`, where 0 represents negativity, 1 represents positivity, and 2 denotes sarcasm.

## 5.3 Result of Bangla Dataset

Using three categories, we separated our Bangla comment data. In multiclass, they are initially classified as "Positive," "Negative," and "Sarcastic." Every comment is classified manually. Using two separate vectorization algorithms, we tokenize or vectorize the comments (data). We use the TF-IDF and Bag of Words (BOW) techniques for tokenization. Several classifiers have been used, including the Logistic Algorithm, Multinomial Naive Bayes, Gaussian Naive Bayes, Decision Tree Classifier, Random Forest Tree, AdaBoost Classifier, and SVM. We use BOW and TF-IDF vectorization to figure out the accuracy, precision, recall, and F1 score for multiclass classifications. Essentially, the dataset is divided into two categories: In the table below, the results of the classifiers are organized by tokenization approach and the result of accuracy, precision, recall, and F1 score.

### 5.3.1 Dataset Distribution

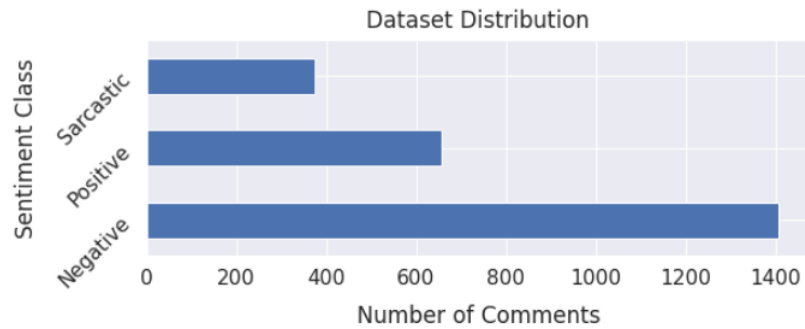


Figure 5.1: Dataset Distribution

As stated before, the Bangla comment data is divided into three categories: "Positive," "Negative," and "Sarcastic." We may conclude from the data distribution that negative comments outnumber both positive and sarcastic comments, which total 1407. In general, the majority of social media users are unwilling to believe food vloggers' evaluations and instead attack them with insults, shame, and even slang.

### 5.3.2 Data Statistics

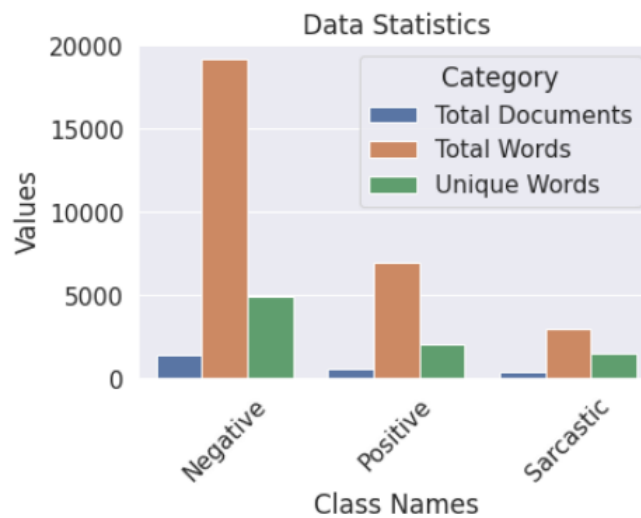


Figure 5.2: Data Statistics

Particularly in data statistics, We create a group for each positive negative and sarcastic comment. Total documents, total words, and unique terms are included for each group. So, we learn the total number of words in a document as well as the proportion of unique keywords. Examining the graph reveals that sarcastic and positive comments include less unique words and documents than negative ones. Even the overall number of negative words is far more than the number of positive and sarcastic remarks, with positive comments ranking second.

### 5.3.3 Length Frequency Distribution

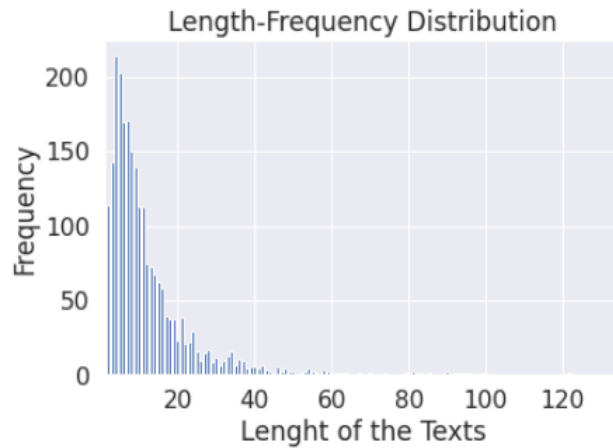


Figure 5.3: Length Frequency Distribution

From this diagram, we can determine the length of a comment or review within our datasets. The maximum length of a comment or review is 345, according to the length frequency distribution, while the minimum length is 2. A reviewer's length is 12.0 on average.

### 5.3.4 Bag of Word tokenizer

The LR and RF are more accurate than the Linear SVM, RBF SVM, DT, K-Nearest Neighbors (KNN) when the uni-gram feature of the Bangla dataset is looked at. The MNB has the lowest accuracy. Decision Tree (DT) produces the greatest F1-Score values, whereas KNN produces the lowest. Random Forest yields the second-lowest scores, whereas Multinomial Naive Bayes yields the second-highest. Comparing the accuracy to the F1-Score reveals that the accuracy for each classifier is much greater than the F1-Score.

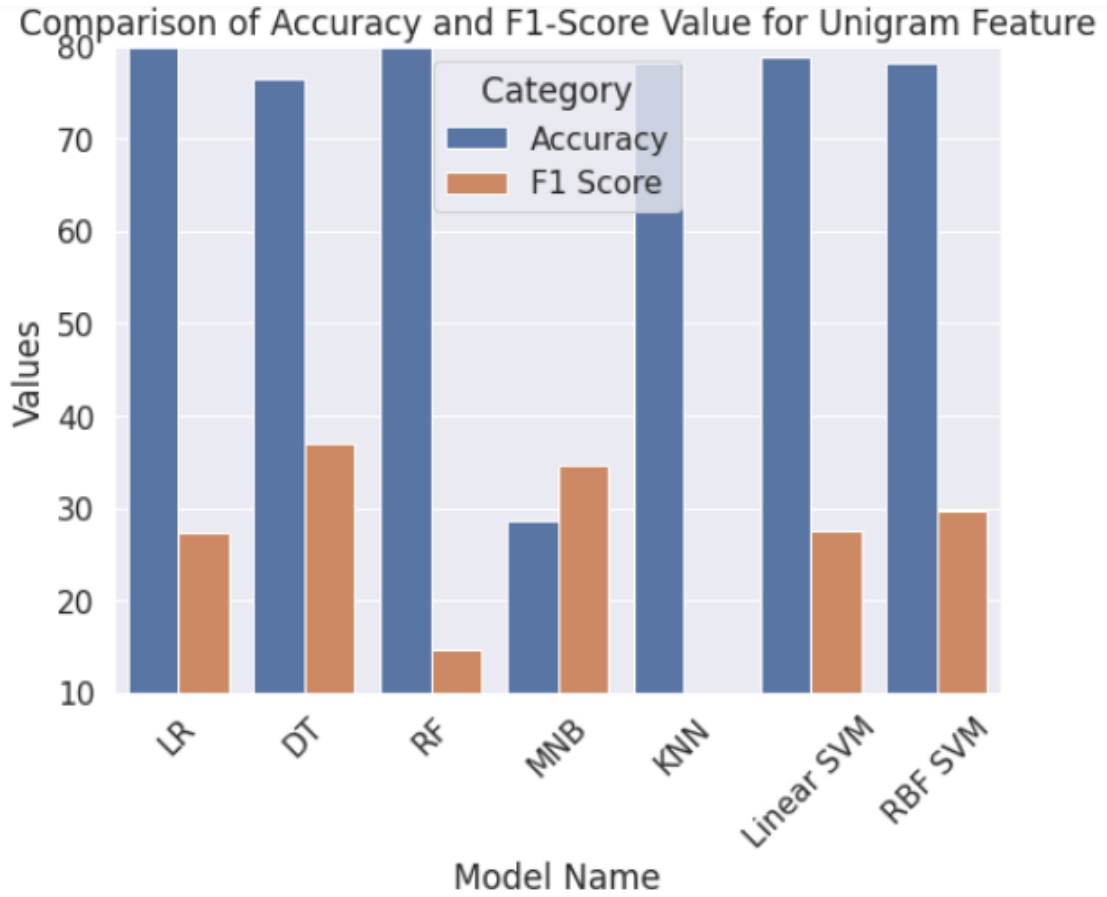


Figure 5.4: Unigram Feature

Performance Table for Unigram Feature				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	81.71	100.00	15.79	27.27
Decision Tree (DT)	76.57	44.44	31.58	36.92
Random Forest (RF)	80.00	100.00	7.89	14.63
Multinomial Naive Bayes (MNB)	28.57	21.57	86.84	34.55
k-Nearest Neighbors (KNN)	78.29	50.00	5.26	9.52
Linear Support Vector Machine (SVM)	78.86	53.85	18.42	27.45
RBF Support Vector Machine (SVM)	78.29	50.00	21.05	29.63

The data in a multiclass Bangla dataset is tokenized with the Bag of Words tech-

nique. The n-gram range used for tokenization was 1. We can observe that the LR classifier has the greatest accuracy score, with a score of 81.71. That is an outstanding result for our dataset. RF tied for second place with a total score of 80.00. In contrast, both the KNN and the RBF SVM have an accuracy of 78.29. The accuracy of the Linear SVM and Decision Tree is 78.86 and 76.57, respectively. With an accuracy of 28.57, the MNB is the least accurate classifier. The KNN algorithm's precision is 50.00. Using the Bag of Words technique to tokenize the Bangla comments dataset tells us that LR is the best classifier for our dataset and MNB is the worst.

Logistic Regression (LR) and Random Forest (RF) are better than the other classifiers when it comes to precision. The precision of both classifiers is 100. The Linear Support Vector Machine is somewhat more accurate than the RBF SVM. This dataset offers decent precision for classifiers that use LR and RF. As a consequence, these two classifiers have a low false positive rate. In contrast, the precision scores for the Linear SVM, KNN, RBF SVM, Decision Tree Classifier and MNB Classifiers are 53.85, 50, 50, 44.44 and 21.57, respectively. So we can state that both the KNN and the RBF SVM have a precision score of 50. The poor scores suggest that a substantial proportion of courses were incorrectly recognized. Here, MNB gives the lowest precision score. The recall value with the highest value for MNB is 86.84. DT, Linear SVM, RBF SVM, LR and RF get the recall value score of 31.58, 21.05, 18.42, 15.79 and 7.89 respectively. This demonstrates that MNB, RBF SVM and DT generated the most relevant results. In contrast, the KNN classifier has a lower recall value of 5.26. These models produced the anticipated outcomes. Both the accuracy and precision values of the MNB are low, but the recall value is the highest. The result is a 34.55 on the F1 scale. The KNN classifier's F1 score is lower, at 9.52. At 36.92, DT has the highest F1 score. The F1 scores of RBF SVM, Linear SVM, LR and RF on the other hand, are 29.63, 27.45, 27.27 and 14.63, respectively. KNN's estimated F1 score is the lowest of all classifiers, at 9.52. However, the second-highest F1 score for the MNB is 34.55. The KNN Classification F1 and recall scores are low. The best classifiers for our multi-class Bag of Word Tokenized Bangla Dataset are LR and both linear and RBF SVM. MNB is the poorest classifier.

Analysis of the Bi-gram feature of the Bangla dataset reveals that the LR and RF are more accurate than the Linear SVM, RBF SVM, DT, and K-Nearest Neighbors (KNN). MNB scores the worst in accuracy. In terms of F1-Score, the Decision Tree (DT) yields the highest results and KNN the lowest. The scores generated by Random Forest are the second-lowest, while those generated by Multinomial Naive Bayes are the second highest. When compared to the F1-Score, the accuracy of each classifier is seen to be significantly higher.

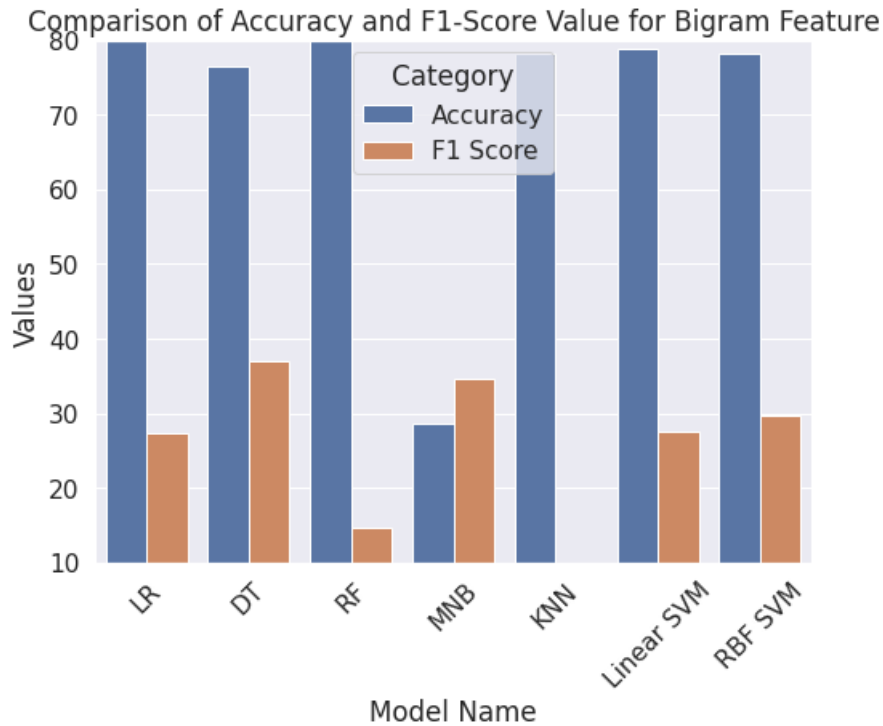


Figure 5.5: Bigram Feature

Performance Table for Bigram Feature				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	81.71	100.00	15.79	27.27
Decision Tree (DT)	76.57	44.44	31.58	36.92
Random Forest (RF)	80.00	100.00	7.89	14.63
Multinomial Naive Bayes (MNB)	28.57	21.57	86.84	34.55
k-Nearest Neighbors (KNN)	78.29	50.00	5.26	9.52
Linear Support Vector Machine (SVM)	78.86	53.85	18.42	27.45
RBF Support Vector Machine (SVM)	78.29	50.00	21.05	29.63

Tokens are created using the Bag of Words technique for a multiclass Bangla dataset. Tokenization was performed using an n-gram range of 2. By far, the LR classifier has the highest accuracy, coming in at 81.71 percent. In terms of our dataset, that is a fantastic finding. With an 80 total, RF was tied for second place. Comparatively, the KNN and the RBF SVM both perform at a 78.29 percent accuracy. Both the Linear SVM and the Decision Tress are quite precise, with accuracies of 78.86 and

76.57 respectively. The MNB is the least reliable classifier with an accuracy of 28.57. The accuracy of the KNN algorithm is 50. Bag of Words tokenization of the Bangla comments dataset informs us that LR is the best classifier and MNB is the worst.

When compared to other classifiers, Logistic Regression (LR) and Random Forest (RF) perform the best in terms of precision. Both classifiers have a 100% precision rate. When compared to the RBF SVM, the Linear Support Vector Machine offers slightly better precision. Classifiers based on LR and RF can achieve reasonable precision on this dataset. This results in a very small percentage of false positives for both classifiers. On the other hand, the Linear SVM, KNN, RBF SVM, Decision Tree Classifier, and MNB Classifier all have precision scores of 53.85, 50, 50, 44.44, and 21.57, respectively. In this case, we can say that the KNN and the RBF SVM both have a precision of 50. Results suggest a large number of misidentified courses contributed to the low scores. The MNB precision rating for this case is very low. MNB has a maximum recall of 86.84, making it the best. The recall values for DT, Linear SVM, RBF SVM, LR, and RF are 31.58, 21.05, 18.42, 15.79, and 7.89. This demonstrates that the best outcomes were achieved by MNB, RBF SVM, and DT. The KNN classifier, on the other hand, the KNN classifier has a lower recall value of 5.26. The predicted results were obtained using these models. Even though the MNB has poor accuracy and precision, it has excellent recall. On the F1 Score, this works out to a score of 34.55. The F1 Score is 9.52, which is lower than that of the KNN classifier. At 36.92, DT has the highest F1 Score. The F1 Scores of RBF SVM, Linear SVM, LR, and RF on the other hand, are 29.63, 27.45, 27.27, and 14.63, respectively. KNN's estimated F1 Score is the lowest of all classifiers, at 9.52. However, the second-highest F1 Score for the MNB is 34.55. The KNN Classification F1 and recall scores are low. The best classifiers for our multi-class Bag of Word Tokenized Bangla Dataset are LR and both linear and RBF SVM. MNB is the poorest classifier.

When evaluating the trigram features of the Bangla dataset, we see that the LR and RF are more accurate than the Linear SVM, RBF SVM, DT, and K-Nearest Neighbors (KNN). MNB scores the worst in accuracy. In terms of F1-Score, the Decision Tree (DT) yields the highest results and KNN the lowest. The scores generated by Random Forest are the second-lowest, while those generated by Multinomial Naive Bayes are the second highest. When the accuracy of each classifier is compared to the F1-Score, it is clear that the accuracy is higher than the F1-Score.



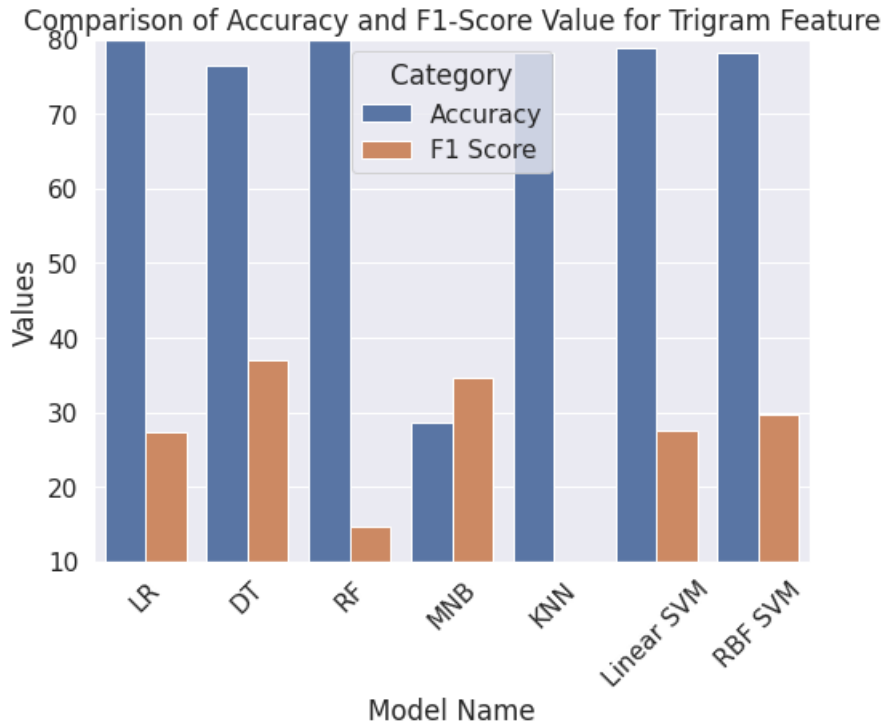


Figure 5.6: Trigram Feature

Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	81.71	100.00	15.79	27.27
Decision Tree (DT)	76.57	44.44	31.58	36.92
Random Forest (RF)	80.00	100.00	7.89	14.63
Multinomial Naive Bayes (MNB)	28.57	21.57	86.84	34.55
k-Nearest Neighbors (KNN)	78.29	50.00	5.26	9.52
Linear Support Vector Machine (SVM)	78.86	53.85	18.42	27.45
RBF Support Vector Machine (SVM)	78.29	50.00	21.05	29.63

Bag of Words is used for tokenization in a multiclass Bangla dataset. Tokenization occurred within a 3-gram n-gram range. As can be seen, the LR classifier achieves the highest accuracy, a score of 81.71. When applied to our dataset, that is a fantastic finding. With an 80 total, RF was tied for second. Conversely, the KNN and the RBF SVM both achieve an accuracy of 78.29. Both the Linear SVM and the

Decision Trees are relatively accurate, with respective accuracy rates of 78.86% and 76.57%. The MNB has the lowest accuracy of any classifier, at just 28.57 percent. KNN has a 50.00 accuracy rate. We find that LR is the best classifier for our dataset and MNB is the poorest by using the Bag of Words method to tokenize the Bangla comments dataset.

Among the available classifiers, the precision of Logistic Regression (LR) and Random Forest (RF) is the highest.

Both classifiers have perfect precision (100). In comparison to the RBF SVM, the Linear Support Vector Machine provides somewhat better precision.

For LR and RF classifiers, this dataset has acceptable precision. Therefore, both of these classifiers are highly accurate and produce few false positives. The precision scores for the Linear SVM, KNN, RBF SVM, Decision Tree Classifier, and MNB Classifiers are 53.85, 50, 50, 44.44, and 21.57, respectively. It may therefore be stated that the KNN and the RBF SVM both have a precision score of 50. The low score indicates that a sizable number of classes were misidentified. MNB assigns the lowest possible precision here. The greatest recall value for MNB is 86.84. Recall values of 31.58, 21.05, 18.42, 15.79, and 7.89 are earned by DT, Linear SVM, RBF SVM, LR, and RF, respectively.

These results show that MNB, RBF SVM, and DT produced the most pertinent findings. KNN classifier, on the other hand, has a recall value of only 5.26. The predicted results were obtained by using these models. The MNB has the lowest accuracy and precision, but excellent recall. The final score on the F1 Score is 34.55. The F1 Score of the KNN classifier is 9.52 which is the lowest F1-Score. DT has the greatest F1 Score by a wide margin, 36.92 to 0. Comparatively, the F1 Scores for RBF SVM, Linear SVM, LR, and RF are 29.63, 27.45, 27.27, and 14.63. To compare, the estimated F1 Score for KNN is 9.52, which is the lowest of any classifier. A score of 34.55 on the F1 Score is, however, the MNB's second-best. KNN Classification has poor F1 Score and recall ratings. Our multi-class Bag of Word Tokenized Bangla Dataset finds the most success with LR and both linear and RBF SVM classifiers and MNB is the worst classifier.

From the performance table of unigram, bigram and trigram features we can determine that LR achieves 81.71 percent accuracy, while MNB classifier achieves just 28.57 percent. There are a number of causes for the discrepancy in accuracy between logistic regression and multinomial naive bayes classifiers, including data type, algorithm implementation, and model complexity.

Since it is a linear model, logistic regression uses a linear combination of the input features to create a prediction. Due to its ease of use and interpretability, this technique trains quickly and works well with huge datasets. Both one-vs-all and softmax regression, two applications of logistic regression, are effective at handling multi-class classification problems.

In contrast, multinomial naive bayes is a probabilistic model founded on the Bayes theorem under the assumption of feature independence. Common applications include text categorization problems where words' presence or absence serve as characteristics. Though it's a quick and easy algorithm, it makes a significant assumption of independence that may not hold true for more complex real-world problems.

Multinomial Naive Bayes is more suited for text classification issues when the inde-

pendence condition holds true, while Logistic Regression is better suited for linearly separable situations. It is likely that the logistic regression classifier outperformed the multinomial naive bayes classifier on the dataset because it was better suited to the task and the data.

### 5.3.5 TFIDF tokenizer

When analyzing the unigram features of the Bangla dataset, the KNN, Linear SVM, RBF Support Vector Machine, DT, RF, and LR classifiers exhibit more accuracy than MNB. Linear SVM and RBF SVM are the approach with the lowest precision. The KNN method yields the best F1-Score values, whereas the Linear SVM and RBF SVM algorithm yields the lowest. LR has the second-lowest scores, while MNB has the second-highest values. When the accuracy of each classifier is compared to the F1-Score, it is clear that the accuracy is higher than the F1-Score.

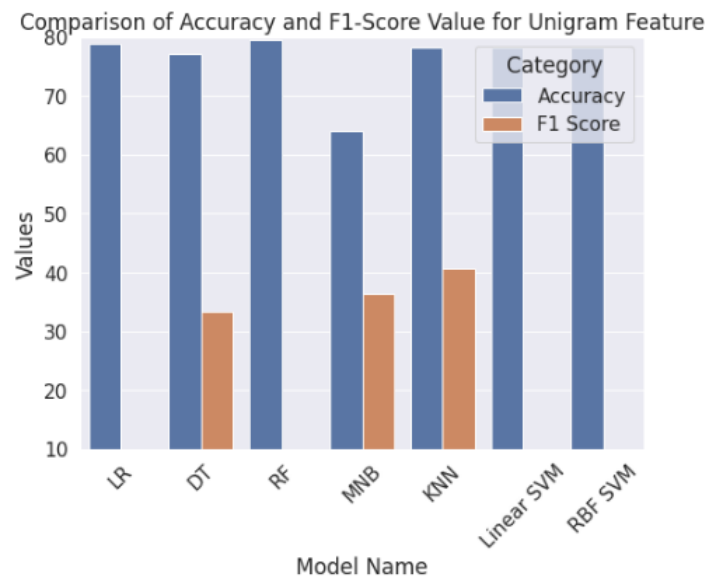


Figure 5.7: Unigram Feature

Performance Table for Unigram Feature				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	78.86	100.00	2.63	5.13
Decision Tree (DT)	77.14	45.45	26.32	33.33
Random Forest (RF)	79.43	100.00	5.26	10.00
Multinomial Naive Bayes (MNB)	64.00	29.51	47.37	36.36
k-Nearest Neighbors (KNN)	78.29	50.00	34.21	40.63
Linear Support Vector Machine (SVM)	78.29	0.00	0.00	0.00
RBF Support Vector Machine (SVM)	78.29	0.00	0.00	0.00

In the Bangla multiclass dataset, the data is tokenized using the TF-IDF approach, and the n-gram range of 1 is used for tokenization. In the TF-IDF technique, the RF Classifier has the greatest accuracy with a score of 79.43, followed by the LR with a score of 78.86. RBF Support Vector Machine, Linear SVM, and KNN, with an accuracy of 78.29. Moreover, DT accuracy is 77.14. The MNB yields the least precise outcomes. The algorithm’s accuracy is 64.00. In conclusion, we can say that RF, which gives the highest level of accuracy, is the ideal classifier for the TF-IDF approach if it is used to tokenize the comments from the multiclass dataset. Using Logistic Regression with RF and LR, the highest precision value of 100.00 is reached for multiclass analysis. This indicates that among the predicted positive numbers, 100% of the actual positive values are positive. MNB has the greatest recall value of all the values, which is 47.37, giving us a true positive value rate of 47.37%. KNN offers the highest F1 score by providing the true positive values that include comments. As a consequence, it is an excellent example. With KNN and DT precision scores of 50.00 which is over 50%, we are better able to discern between real positive values and predicted positive values and 45.45, respectively. Lastly, the MNB method yields precision values of 29.51, which is less than 50 percent. Also, the Linear SVM and RBF Support Vector Machine classifiers have the lowest precision score of 0. This means that none of the predicted positive values come true when using these classifiers.

Using MNB, we achieved a recall value of 47.37. The classifier produces a positive value less than half of the time, or the rate of positive values among all other values. KNN, DT and RF had respectable scores of 34.21, 26.32 and 5.26. These classifiers obtain an recall rate of less than 50%. Once again, LR classifiers attain a recall value of 2.63. In addition, both the Linear SVM and the RBF Support Vector Machine provide a recall value of zero, which is the lowest possible result. This means that, among all other values in the dataset, neither the Linear Support Vector Machine nor the RBF Support Vector Machine produces a real positive value

rate. The F1 score generated by the KNN algorithm is 40.63, which is less than 50%. The f1 score shows the average accuracy and recall value. The F1 scores obtained from LR, DT, RF, and MNB are 5.13, 33.33, 10 and 36.36, respectively. Also, the RBF Support Vector Machine and Linear SVM classifiers give 0 for the lowest f1 scores, which suggests that they are not good models. Finally, we can say that KNN and DT, are the best classifiers for the TF-IDF technique in the multi-class dataset, whereas Linear SVM and RBF Support Vector Machine are the worst.

When looking at the bigram characteristics of the Bangla dataset, KNN, Linear SVM, RBF Support Vector Machine, DT classifier, RF classifier, and LR are more accurate than MNB. Linear SVM and RBF SVM are the approach with the lowest precision. The KNN method yields the best F1-Score values, whereas the Linear SVM and RBF SVM algorithm yields the lowest. LR has the second-lowest scores, while MNB has the second-highest values. When comparing the F1-Score to the accuracy, it is obvious that the accuracy is greater than the F1-Score for each classifier.

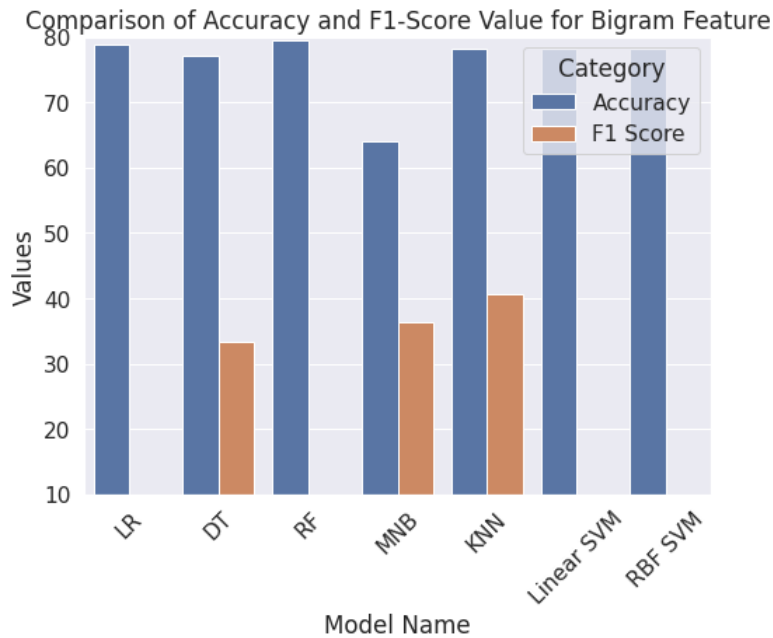


Figure 5.8: Bigram Feature

Performance Table for Bigram Feature				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	78.86	100.00	2.63	5.13
Decision Tree (DT)	77.14	45.45	26.32	33.33
Random Forest (RF)	79.43	100.00	5.26	10.00
Multinomial Naive Bayes (MNB)	64.00	29.51	47.37	36.36
k-Nearest Neighbors (KNN)	78.29	50.00	34.21	40.63
Linear Support Vector Machine (SVM)	78.29	0.00	0.00	0.00
RBF Support Vector Machine (SVM)	78.29	0.00	0.00	0.00

We employ the n-gram range of 2 for tokenization in the multiclass Bangla dataset, which is tokenized using the TF-IDF method. In the TF-IDF method, the RF classifier has the highest accuracy of 79.43, followed by the LR, which has an accuracy of 78.86. Using a combination of the RBF SVM, the linear SVM, and the KNN, we obtain an accuracy of 78.29 percent. In addition, DT has a 77.14 percent accuracy rate. The results from the MNB are the least precise. This algorithm has a 64.00 percent accuracy rate. To sum up, RF, which provides the highest accuracy, is the best classifier for the TF-IDF method if it is used to tokenize the comments from the multiclass dataset. It is possible to get a precision of 100.00 in multiclass analysis using Logistic Regression with both RF and LR. This means that all anticipated positive numbers have corresponding real positive values (i.e., 100%). When compared to all other values, MNB has the highest recall value, at 47.37, giving us a real positive value rate of 47.37 percent. The true positive values with comments are provided by KNN, which results in the greatest F1 Score. Consequently, it serves as a great illustration. KNN and DT both have over 50% precision, therefore we can tell the difference between actual positive values and predicted positive values of 45.45 and 50.00, respectively. Finally, the MNB approach provides values for precision that are less than 50%, at 29.51. In addition, the Linear SVM and RBF Support Vector Machine classifiers also have a precision score of 0. No correct positive predictions can be made using these classifiers. We were able to increase the recall to 47.37 by using MNB.

It is fewer than half the time that the classifier generates a positive value, as measured by the proportion of positive values to all values.

The results for KNN, DT, and RF were all above average: 34.21, 26.32, and 5.26. The recall rate these classifiers achieve is under 50%. For the second time, LR classifiers achieve a recall of 2.63. Further, the Linear SVM and the RBF Support Vector Machine both produce the worst possible result (recall = 0). So, neither the Linear Support Vector Machine nor the RBF Support Vector Machine generates an

actual positive value rate among the other values in the dataset. KNN's calculated F1 score is 40.63, which is below 50%. The f1 score represents an average of the precision and recall values.

The LR, DT, RF, and MNB F1 scores are 5.13%, 33.33%, 10, and 36.36%, respectively. Furthermore, the RBF Support Vector Machine and Linear SVM classifiers also return 0 for the worst possible f1 scores, indicating that they are not reliable models. When it comes to the multiclass dataset used by the TF-IDF method, we can conclude that the best classifiers are KNN and DT, while the poorest are Linear SVM and RBF Support Vector Machine.

When looking at the trigram characteristics of the Bangla dataset, KNN, Linear SVM, RBF Support Vector Machine, DT classifier, RF classifier, and LR are more accurate than MNB. Linear SVM and RBF SVM are the approach with the lowest precision. The KNN method yields the best F1-Score values, whereas the Linear SVM and RBF SVM algorithm yields the lowest. LR has the second-lowest scores, while MNB has the second-highest values. When comparing the F1-Score to the accuracy, it is obvious that the accuracy is greater than the F1-Score for each classifier.

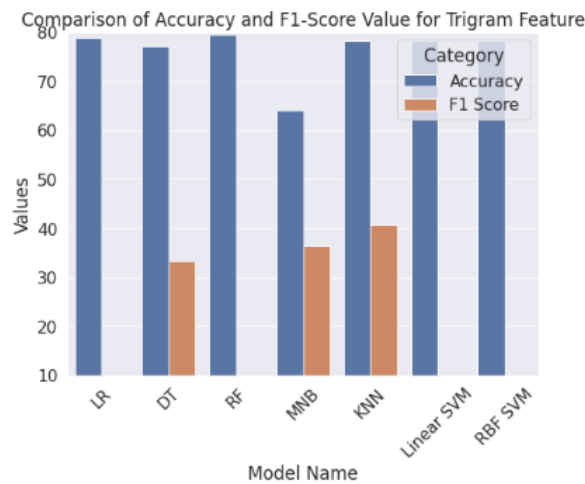


Figure 5.9: Trigram Feature

Performace Table for Trigram Feature				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	78.86	100.00	2.63	5.13
Decision Tree (DT)	77.14	45.45	26.32	33.33
Random Forest (RF)	79.43	100.00	5.26	10.00
Multinomial Naive Bayes (MNB)	64.00	29.51	47.37	36.36
k-Nearest Neighbors (KNN)	78.29	50.00	34.21	40.63
Linear Support Vector Machine (SVM)	78.29	0.00	0.00	0.00
RBF Support Vector Machine (SVM)	78.29	0.00	0.00	0.00

The Bangla multiclass dataset is tokenized using the TF-IDF method, and the n-gram range used in this instance is 3. The RF classifier in the TF-IDF method has the greatest accuracy, 79.43, while the LR has the second-highest accuracy, 78.86. Using a combination of the RBF SVM, the linear SVM, and the KNN, we obtain an accuracy of 78.29 percent. In addition, DT has a 77.14 percent accuracy rate. The results from the MNB are the least precise. This algorithm has a 64.00 percent accuracy rate. To sum up, RF, which provides the highest accuracy, is the best classifier for the TF-IDF method if it is used to tokenize the comments from the multiclass dataset. It is possible to get a precision of 100.00 in multiclass analysis using Logistic Regression with both RF and LR. This means that all anticipated positive numbers have corresponding real positive values (i.e., 100%).

When compared to all other values, MNB has the highest recall value, at 47.37, giving us a real positive value rate of 47.37 percent.

The true positive values with comments are provided by KNN, which results in the greatest F1 Score. Consequently, it serves as a great illustration. KNN and DT both have over 50% precision, therefore we can tell the difference between actual positive values and predicted positive values of 45.45 and 50.00, respectively. Finally, the MNB approach provides values for precision that are less than 50%, at 29.51.

In addition, the Linear SVM and RBF Support Vector Machine classifiers also have a precision score of 0. No correct positive predictions can be made using these classifiers. When we used MNB, we were able to increase the recall to 47.37.

The classifier outputs a negative result more often than a positive one, or the rate of positive values relative to all other values is less than 50%.

The values of 34.21 for KNN, 26.32 for DT, and 5.26 for RF are all rather good. The recall rate achieved by these classifiers is under 50%. Recall values of 2.63 are again achieved by LR classifiers. Recall values of 0 are also provided by the Linear SVM and the RBF Support Vector Machine, which is the worst conceivable result. Therefore, neither the Linear Support Vector Machine nor the RBF Support



Vector Machine generates a real positive value rate among the other values in the dataset. Using the KNN technique, we find an F1 score of 40.63, which is below 50%. Measured by the f1 score, accuracy and recall are shown on average.

There is a significant difference between the F1 scores of LR (5.13), DT (33.33), RF (10) and MNB (36.36). As a further piece of evidence that they are not reliable models, the RBF Support Vector Machine and Linear SVM classifiers both return 0 for the worst possible f1 scores. We conclude that KNN and DT, are the best classifiers for the TF-IDF method in the multiclass dataset, whereas Linear SVM and RBF Support Vector Machine are the worst.

From the performance table of unigram, bigram and trigram feature We can observe from the table that RF has the highest accuracy of 89.43 percent and on the other hand MNB classifier has the lowest accuracy of 64.00 percent. The difference in accuracy between the Random Forest classifier and the multinomial naive bayes classifier can be due to several factors, including the nature of the data being used, the specific implementation of the algorithms, and the complexity of the model.

Random Forest is an ensemble learning method that creates multiple decision trees and combines them to make predictions. Each tree is created using a random subset of the data, and a random subset of the features. This helps to reduce overfitting and increase the generalization performance of the model. Random Forest can handle both categorical and numerical data and can be used for both binary and multi-class classification problems. They are robust to noise, outliers and handle high dimensionality.

On the other hand, multinomial naive bayes is a probabilistic model which is based on the Bayes theorem with the assumption of independence between every pair of features. It is commonly used for text classification problems, where the features are the presence or absence of words in a text. It is a fast and simple algorithm but it has a strong independence assumption which may not always be true for complex real-world problems.

In general, Random Forest is more suited for problems that have complex relationships between features and Multinomial Naive Bayes is more suited for text classification problems where the independence assumption holds true.

In this case, it is likely that the Random Forest classifier performed better on the given dataset because it was better suited to the task and the data than the multinomial naive bayes classifier.

### **5.3.6 Prediction**

Here, the computation for the predicted comments is performed. This algorithm is designed to produce a single forecast. The Counter Vectorizer transform method is used for this comment. Using this, the predict method is invoked, which returns a result based on the numeric labels used to label the data in our dataset. In addition, the probability (score) of forecast success is computed.

## 5.4 Result of Banglish Dataset

Our Banglish comment data is divided into three distinct categories. First, we categorize them as "Positive," "Negative," or "Sarcastic." Each comment undergoes manual classification. To tokenize or vectorize the comments, for example, we employ two distinct vectorization algorithms (data). For tokenization, we use the TF-IDF and Bag of Words (BOW) techniques. The LR, MNB, GaussianNB, DT, RF, AdaBoost Classifier, and SVM are among the algorithms that have been used. We use BOW and TF-IDF vectorization to determine the accuracy of multiclass classifications. The dataset is essentially separated into two sections. In the table below, the classification results are organized by tokenization method and the result of accuracy.

### 5.4.1 Bag Of Word tokenized Banglish Dataset

Performace Table for Bag Of Word (BOW)				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	0.74	0.69	0.63	0.65
Multinomial Naive Bayes (MNB)	0.76	0.69	0.67	0.67
Gaussian Naive Bayes Algorithm	0.64	0.58	0.58	0.58
AdaBoost Classifier	0.683	0.68	0.55	0.56
Decision Tree Classifier (DT)	0.58	0.40	0.41	0.37
Random Forest (RF)	0.69	0.62	0.60	0.61
Support Vector Machine (SVM)	0.700	0.64	0.62	0.63

Tokenization using the Bag of Words technique is applied to the data in a multiclass dataset. At 0.76, the accuracy of the Multinomial Naive Bayes (MNB) classifier is quite good. That's excellent performance on a multiclass dataset. At 0.74, Logistic Regression (LR) is in second place. With a score of 0.700, Support Vector Machine (SVM) is in third place. However, the accuracy for Random Forest (RF), Adaboost Classifier and Gaussian Naive Bayes Algorithm (GaussianNB) are 0.69, 0.683 and 0.64. However, the 0.58 accuracy of the Decision Tree Classifier (DT) is the lowest of all the classifiers. If we use Bag of Words to tokenize the Banglish comments, we may then infer that Multinomial Naive Bayes (MNB) is the best classifier while Decision Tree Classifier (DT) is the worst classifier for a multiclass dataset. The classifiers with the highest precision are Logistic Regression (LR) and Multinomial Naive Bayes (MNB). Both Logistic Regression (LR) and Multinomial Naive Bayes (MNB) have a precision of 0.69. Both Logistic Regression and Multinomial Naive Bayes (MNB) can achieve respectable levels of precision with this dataset. This means that the false-positive rate of these two models is quite small.

The second-highest precision value of 0.68 is achieved by the AdaBoost Classifier. When compared to the Random Forest Classifier's precision value of 0.62, the Support Vector Machine's (SVM) value of 0.64 is somewhat higher (RF). The precision value of the Gaussian Naive Bayes (GaussianNB) algorithm is 0.58. Contrarily, the 0.40 precision value below 50% is achieved by the Decision Tree Classifier (DT). A large number of misidentified courses likely contributed to this poor score. The recall score of 0.67 for Multinomial Naive Bayes (MNB) is outstanding. We also get the recall scores of 0.63 for Logistic Regression (LR), 0.62 for Support Vector Machine (SVM), and 0.60 for Random Forest Classifier (RF). From this, we may infer that Multinomial Naive Bayes (MNB), Logistic Regression (LR), Support Vector Machine (SVM), and Random Forest Classifier (RF) yielded the most pertinent findings. When compared to Multinomial Naive Bayes (MNB), Logistic Regression (LR), Support Vector Machine (SVM), and Random Forest Classifier (RF), Gaussian Naive Bayes (GaussianNB) and Adaboost Classifier have lower recall values (0.58 and 0.55, respectively). In contrast, the 0.41 recall value achieved using Decision Tree Classifiers (DT) is significantly lower. The anticipated results from applying these models have materialized. As can be seen, the Decision Tree Classifier's (DT) recall and accuracy values both are the lowest. Consequently, an F1 Score of 0.37 is achieved. Therefore, it cannot be included in this dataset. We also get the F1 Scores are likewise lower (0.56 and 0.58) for the AdaBoost Classifier and the Gaussian Naive Bayes Algorithm (GaussianNB). However, the F1 Scores for Logistic Regression (LR), Support Vector Machine (SVM), and Random Forest Classifier (RF) are 0.65, 0.63 and 0.61. A maximum of 0.67 is achieved with Multinomial Naive Bayes (MNB). Decision Tree's predicted F1 score of 0.37 is the lowest of all of the classifiers. Random Forest Classifier's (RF) recall and accuracy numbers are quite close.

We can see from the table that MNB has the highest accuracy of 0.76 and on the other hand DT classifier has the lowest accuracy of 0.58. The difference in accuracy between the multinomial naive bayes classifier and the decision tree classifier can be due to several factors, including the nature of the data being used, the specific implementation of the algorithms, and the complexity of the model.

Multinomial Naive Bayes is a probabilistic model which is based on the Bayes theorem with the assumption of independence between every pair of features. It is commonly used for text classification problems, where the features are the presence or absence of words in a text. It is a fast and simple algorithm but it has a strong independence assumption which may not always be true for complex real-world problems.

On the other hand, Decision tree classifiers work by recursively splitting the data based on the values of the input features. They create a tree-like structure where each internal node represents a feature and each leaf node represents a class label. Decision tree classifiers can handle both categorical and numerical data and can be used for both binary and multi-class classification problems. However, decision tree classifiers are prone to overfitting and are sensitive to small changes in the training data.

In this case, it is likely that the multinomial naive bayes classifier performed better on the given dataset because it was better suited to the task and the data than the

decision tree classifier.

### 5.4.2 TFIDF tokenized Banglish Dataset

Performace Table for Term Frequency-Inverse Document Frequency (TF-IDF)				
Model Name	Accuracy	Precision	Recall	F1 Score
Logistic Regression (LR)	0.736	0.75	0.58	0.57
Multinomial Naive Bayes (MNB)	0.74	0.86	0.57	0.57
Gaussian Naive Bayes Algorithm	0.65	0.58	0.58	0.58
AdaBoost Classifier	0.694	0.66	0.55	0.54
Decision Tree Classifier (DT)	0.58	0.41	0.41	0.37
Random Forest Classifier (RF)	0.708	0.68	0.58	0.59
Support Vector Machine (SVM)	0.75	0.80	0.61	0.62

In the TF-IDF technique, the Support Vector Machine (SVM) Algorithm gives the most accuracy, which is 0.75. Multinomial Naive Bayes, with an accuracy of 0.74, is the second best classifier. After that, the accuracy of Logistic Regression (LR) and the Random Forest Algorithm (RF) is 0.736 and 0.708, respectively. Again, the accuracy of the AdaBoost Classifier and Gaussian Naive Bayes Algorithm (GaussianNB) is 0.694 and 0.65, respectively. The Decision Tree Classifier (DT) has an accuracy of 0.58 at its worst. So, in short, we can say that if we use the TF-IDF method to tokenize Banglish comments from the multiclass dataset, Support Vector Machine (SVM) is the best algorithm because it gives the most accuracy, while Decision Tree Classifier (DT) gives the least accuracy. The greatest value for precision is 0.86 when using the Multinomial Naive Bayes (MNB) classifier for multiclass data. This suggests that 86% of the predicted positive values actually occur. That's mean 86% of actual positive values among the predictable positive values. The precision of the second best classifier, the Support Vector Machine (SVM), is 0.80. Each of the four algorithms—Logistic regression (LR), Random Forest Classifier (RF), Adaboost Classifier (Adaboost) and Gaussian Naive Bayes (GaussianNB)—gives a precision value above 50%, with values of 0.75, 0.68, 0.66, and 0.58, respectively. The Decision Tree Classifier (DT) has the lowest precision score, at 0.41%, which is below 50% and indicates that it predicts fewer true positives than it actually produces. When compared to the other values, Support Vector Machine (SVM) has the highest recall value of 0.61, meaning that 61% of the predicted values are correct. That means a 61% actual positive value rate among all the other values. Recalls of 0.58 are also achieved by the classifiers that follow: Logistic Regression (LR), the Gaussian Naive Bayes Algorithm (GaussianNB), and the Random Forest Classifier (FR). Recall values of 0.57 and 0.55 are obtained using Multinomial Naive Bayes (MNB) and Adaboost Classifier, respectively. The true positive rate for these

classifiers is more than 50%. The smallest recall we get from Decision Tree Classifier (DT) is 0.41. As a result, when compared to all other values in the dataset, the true positive rate is lowest with Decision Tree Classifier (DT). The Support Vector Machine (SVM), which provides the actual positive values with Banglish comments has the greatest F1 Score of 0.62 and is thus an excellent model. The second highest F1 Score is then obtained from the Random Forest Classifier (RF), which has an F1 Score of 0.59. The F1 Scores for Multinomial Naive Bayes (MNB) and Logistic Regression (LR) are both 0.57. The Gaussian Naive Bayes Algorithm (GaussianNB) and Adaboost classifier give us an F1 Score of 0.58 and 0.54, respectively, which is greater than 50%. The average precision and recall value is represented by the F1 Score. In light of this, we can claim that the Logistic Regression (LR), Multinomial Naive Bayes (MNB), Gaussian Naive Bayes Algorithm (GaussianNB), Adaboost classifier and Random Forest Classifier (RF) models are effective ones. The Decision Tree Classifier (DT), which produces the lowest F1 Score of 0.37 is the worst model.

We can see from the table that SVM has the highest accuracy of 0.75 and on the other hand DT classifier has the lowest accuracy of 0.58. Support Vector Machine (SVM) and Decision Tree classifiers are both supervised machine learning algorithms that can be used for classification problems, but they work in different ways.

SVMs are a type of linear classifier that attempts to find the best boundary (or hyperplane) to separate the different classes in the data. They are particularly effective in high-dimensional spaces and can also be used for non-linear classification by using kernels. SVM classifiers tend to perform well when the data is clean and the classes are well separated.

Decision Tree classifiers, on the other hand, work by recursively splitting the data based on the values of the input features. They create a tree-like structure where each internal node represents a feature and each leaf node represents a class label. Decision Tree classifiers can handle both categorical and numerical data and can be used for both binary and multi-class classification problems. However, Decision Tree classifiers are prone to overfitting and are sensitive to small changes in the training data.

In this case, it is likely that the SVM classifier performed better on the given dataset because it was better suited to the task and the data than the Decision Tree classifier. The SVM classifier may have been able to find a better boundary to separate the classes, while the Decision Tree classifier may have been overfitting the data, resulting in poor performance.

### 5.4.3 Score Generation

Each comment has been examined and its sentiment score has been computed during this step. To produce the sentiment score, we manually categorized the dataset as positive, negative, or sarcastic. Then, using 531 positive, 961 negative, and 305 sarcastic polarity scores, we matched the dataset to the machine's polarity scores by using Sentiment Intensity Analyzer and produced sentiment scores for each line. The score is 0.5347801892042293 which denotes that the score is more than 50%.

## 5.4.4 Prediction

We do the computation for the predicted comments here. The goal of this computation is to produce a single forecast. For this comment, we use the Counter Vectorizer transform method. Using that, we call the predict method, which returns a result based on the numerical labels that we used in our dataset for labeling the data. In this case, the prediction result for the comment "valo laglo video ta" is `array([1])`, indicating that it is a positive comment, while the result for the comment "baje video" is `array([0])`, indicating that it is a negative comment. The probability (score) of success in forecasting is also calculated. For instance, the comment "valo laglo video ta" returns like `array([[0.11936296, 0.851468, 0.02916903]])`, where the highest score, 0.851463, represents `array([1])` because our class is defined as `array([0, 1, 2])`, where 0 represents negativity, 1 represents positivity, and 2 represents sarcasm.

## 5.5 Confusion Matrix

The Confusion matrix demonstrates how perfectly indicated expected values are in a classification task. It specifically demonstrates the effectiveness of the model that we are using. This is applicable to both binary and multiclass problems. Incorporating an  $n \times n$  matrix, it demonstrates to us the type and quantity of errors we have. Here,  $n$  is based on the number of classes present in our problem. Recall, Precision, and Accuracy can all be calculated extremely effectively using this method.

		Actual Value (as confirmed by experiment)	
		positives	negatives
Predicted Value (predicted by the test)	positives	<b>TP</b> True Positive	<b>FP</b> False Positive
	negatives	<b>FN</b> False Negative	<b>TN</b> True Negative

Figure 5.10: Confusion Matrix

The confusion matrix of the Multiclass Dataset using TF-IDF and Bag of Words conducted on various classifiers, including MultinomialNB, GaussianNB, Decision Tree, AdaBoost, Random Forest, Logistic Regression, and SVM, is displayed on the following page.

## 5.5.1 Confusion Matrix of English Dataset for Bag of Words (BOW)

The following shows how the various classifiers performed on the confusion matrix while using the Bag of Words (BOW) tokenizer:

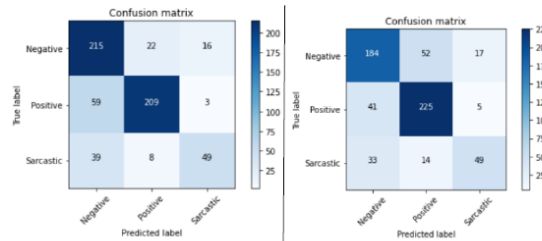


Figure 5.13: Logistic Regression

Figure 5.14: MultinomialNB Algorithm

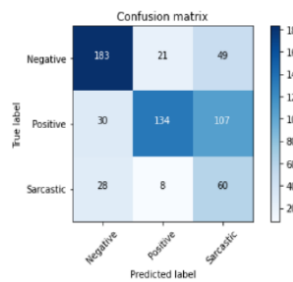


Figure 5.15: GaussianNB Algorithm

Figure 5.11: Confusion Matrices of Multi-Class Dataset of English

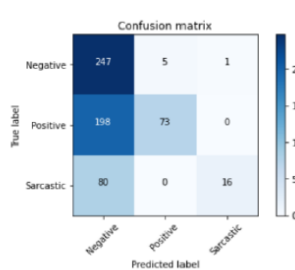


Figure 5.16: Decision Tree Classifier

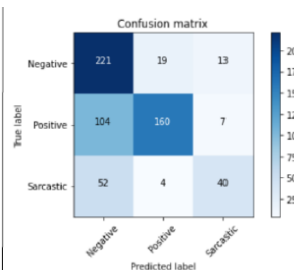


Figure 5.17: AdaBoost Classifier

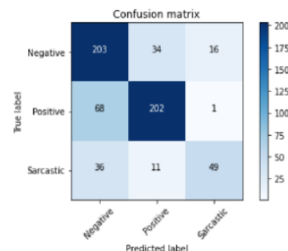


Figure 5.18: Random Forest Classifier

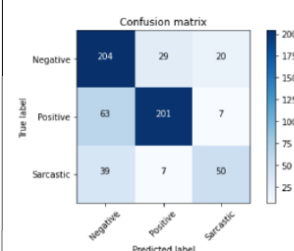


Figure 5.19: Support Vector Machine

Figure 5.12: Confusion Matrices of Multi-Class Dataset of English

## 5.5.2 Confusion Matrix of English Dataset for Term Frequency Inverse Document Frequency (TF-IDF)

The following shows how well each classifier performed on the confusion matrix created by the Term Frequency Inverse Document Frequency (TF-IDF) tokenizer:

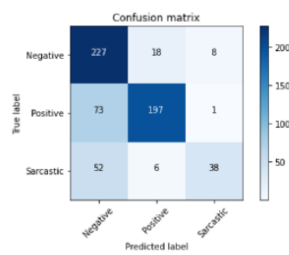


Figure 5.13: Logistic Regression

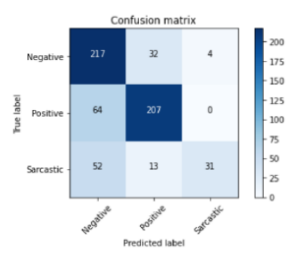


Figure 5.14: MultinomialNB Algorithm

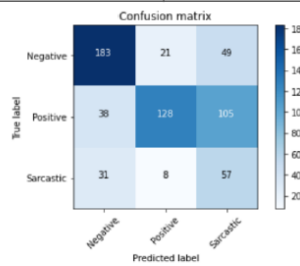


Figure 5.15: GaussianNB Algorithm

Figure 5.13: Confusion Matrices of Multi-Class Dataset of English

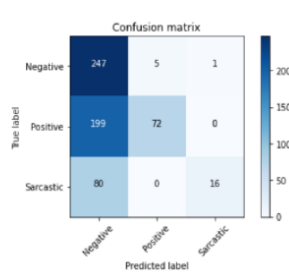


Figure 5.16: Decision Tree Classifier

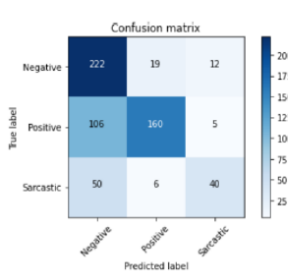


Figure 5.17: AdaBoost Classifier

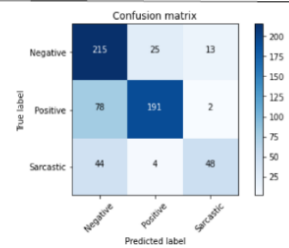


Figure 5.18: Random Forest Classifier

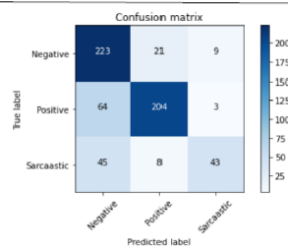


Figure 5.19: Support Vector Machine

Figure 5.14: Confusion Matrices of Multi-Class Dataset of English



### 5.5.3 Confusion Matrix of Banglish Dataset for Bag of Words (BOW)

Using the Bag of Words (BOW) tokenizer, the following demonstrates the results of various classifiers on the confusion matrix:

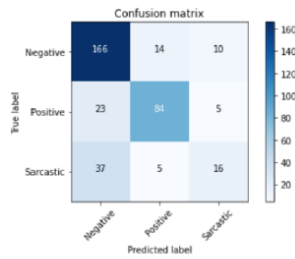


Figure 5.15: Logistic Regression

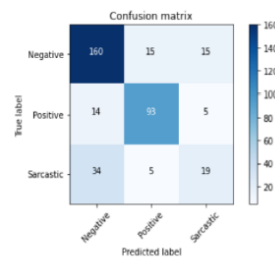


Figure 5.16: MultinomialNB Algorithm

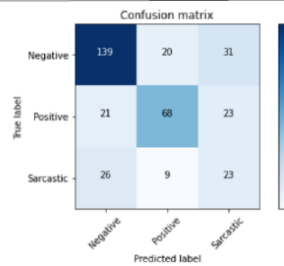


Figure 5.17: GaussianNB Algorithm

Figure 5.15: Confusion Matrices of Multi-Class Dataset of Banglish

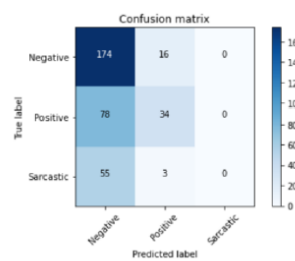


Figure 5.18: Decision Tree Classifier

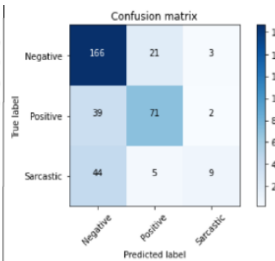


Figure 5.19: AdaBoost Classifier

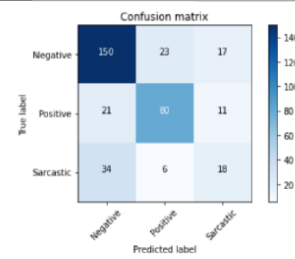


Figure 5.20: Random Forest Classifier

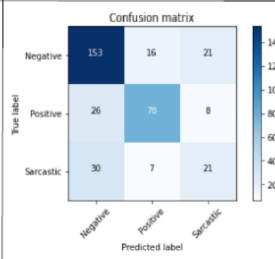


Figure 5.21: Support Vector Machine

Figure 5.16: Confusion Matrices of Multi-Class Dataset of Banglish

## 5.5.4 Confusion Matrix of Banglish Dataset for Term Frequency Inverse Document Frequency (TF-IDF)

For the confusion matrix generated by the TF-IDF tokenizer, the following demonstrates how well each classifier performed:



Figure 5.19: Logistic Regression

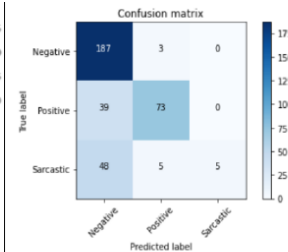


Figure 5.20: MultinomialNB Algorithm

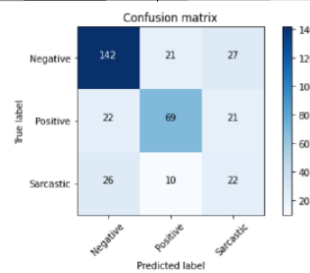


Figure 5.21: GaussianNB Algorithm

Figure 5.17: Confusion Matrices of Multi-Class Dataset of Banglish

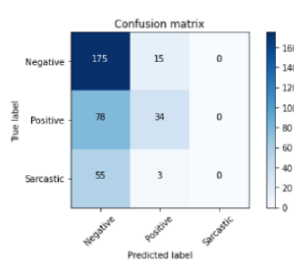


Figure 5.22: Decision Tree Classifier

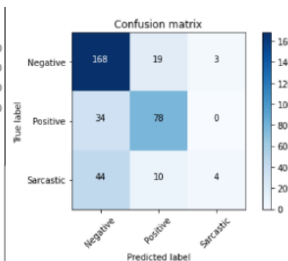


Figure 5.23: AdaBoost Classifier

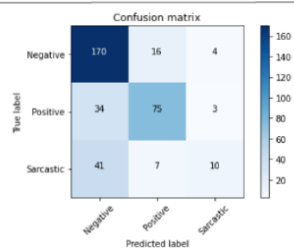


Figure 5.24: Random Forest Classifier

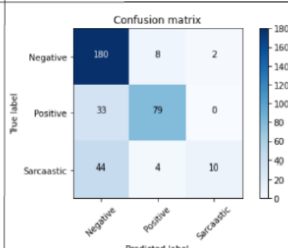


Figure 5.25: Support Vector Machine

Figure 5.18: Confusion Matrices of Multi-Class Dataset of Banglish

## 5.6 Result Comparison

For the bag of words and TF-IDF, we used Logistic Regression, Multinomial NB, Gaussian NB, DecisionTree, AdaBoost, Random Forest, and SVM classifiers. These

are the resulting scores.

For English dataset, the LR model demonstrated a moderate level of accuracy, with a score of 0.75 for the TF-IDF representation and a slightly higher score of 0.763 for the Bag of Words representation. The MNB algorithm yielded a higher level of accuracy, with a score of 0.79 for the Bag of Words representation and a slightly lower score of 0.73 for the TF-IDF representation. The SVM model exhibited a moderate level of accuracy, with scores of 0.734 and 0.76 for the Bag of Words and TF-IDF representations, respectively. The GaussianNB algorithm demonstrated relatively low levels of accuracy, with scores of 0.61 and 0.59 for the Bag of Words and TF-IDF representations, respectively. The AdaBoost Classifier achieved moderate levels of accuracy, with scores of 0.679 and 0.681 for the Bag of Words and TF-IDF representations, respectively. The DT Classifier exhibited a relatively low level of accuracy, with a score of 0.54 for both the Bag of Words and TF-IDF representations. Lastly, the Random Forest Classifier had moderate level of accuracy, with scores of 0.732 for both the Bag of Words and TF-IDF representations.

The SVM, LR, RF, DT, KNN, Linear, and RBF Support Vector Machines are among the classifiers used for the Bangla dataset. These are the resulting scores. SVM in linear form yielded an accuracy of 78.86, a recall score of 18.42, a precision score of 53.85, and an f1 value of 27.45 for the bag containing the remaining words. The TF-IDF score for recall, accuracy, and f1 is 0. For LR, the score for the bag of words was recall 15.79, accuracy of 81.71, precision 100.00, and f1 score 27.27. For the TF-IDF, we obtained a recall of 2.63, an accuracy of 78.86, and an f1 score of 5.13.

The Banglish dataset was analyzed using various machine learning algorithms, including LR, MNB, Gaussian NB, AdaBoost, Decision Tree Classifier, and Random Forest. The results of this analysis indicate that the accuracy of LR for Bag of Words is 0.74, and for TF-IDF is 0.736. The MNB algorithm demonstrates higher accuracy when using Bag of Words, with an accuracy of 0.76, compared to 0.74 for TF-IDF. The GNB algorithm shows a difference in accuracy of 0.1, with Bag of Words having an accuracy of 0.64, and TF-IDF having an accuracy of 0.65. AdaBoost has an accuracy of 0.694 for TF-IDF and 0.683 for Bag of Words. Decision Tree Classifier has an accuracy of 0.58 for both Bag of Words and TF-IDF. Random Forest has an accuracy of 0.69 for Bag of Words and 0.708 for TF-IDF. Finally, SVM has an accuracy of 0.7 for Bag of Words and 0.75 for TF-IDF. Overall, it seems that the use of TF-IDF generally results in higher accuracy across the different algorithms.

## 5.7 Further Discussion

We are working with the keywords of our dataset. We take the keywords from our Bangla, English, and Banglish databases and use them to make comparisons between the food vloggers and the relevant social media.

Keyword extraction is a quick and easy way to locate the most relevant terms in our data. The meanings of these expressions are not to be underestimated. We utilize the collections python library to extract keywords. Counter is imported from the collection. We then construct a method and supply it with our data in the form of a

parameter. Here, we employ the split function to partition our data and the tolist() function to retrieve a list of the partitioned pieces.

Comparison Among the food reviewers within different social media and youtube			
Elements of Comparison	Facebook	Instagram	YouTube
Total Comments	2483	97	5750
Total Positive Comments	841	68	1962
Total Negative Comments	1292	23	2854
Total Sarcastic Comments	355	11	949

Facebook, Instagram, and YouTube food bloggers each have their own distinct following and methods of interaction with their viewers.

With 2483 comments and most of them being negative (1292), Facebook has the most negative impression. It has a large number of sarcastic comments (355), but also a large number of positive comments (841). The results reveal that when it comes to rating and reviewing restaurants, Facebook users are the most outspoken and involved.

Contrarily, Instagram has the fewest comments at just 97. Still, with 70% positive feedback, it dominates the competition. This shows that Instagram users are more inclined to interact positively with food bloggers, maybe through likes and shares, than they are to leave critical comments.

Over 5750 comments have been made on YouTube, with 2854 of them being negative. However, there are also many enthusiastic endorsements (1962) and critical remarks (949). This points to an active and outspoken YouTube audience that is also more likely to be abusive and outraged by the videos they watch.

Food bloggers should be aware of the differences in audiences and engagement between the various platforms before deciding where to devote their time and energy. Additionally, they need to be cognizant of the fact that interaction and criticism levels differ between platforms, and tailor their material accordingly.

Comparison between the food reviewers and food review groups												
Elements of Comparison	Food Bank	Bangladesh Eats	Zoltan BD	Rafsan The Choto Bhai	Metro Man	Petuk Couple	Khudalagse	Foodiesm	KHAI DAI. COM	Bangladesh Food Reviewer	Food BD	Food Appi
Total Comments	124	137	359	1749	1237	366	1395	294	269	994	1161	196
Total Positive Comments	18	80	170	534	484	192	662	118	123	362	9	99
Total Negative Comments	73	50	160	910	543	146	595	144	105	450	866	87
Total Sarcastic Comments	33	7	29	305	210	28	138	32	41	182	286	10

RafsanTheChotobhai, the Petuk couple, MetroMan, Khudalagse, ZoltanBD, Foodiesm, Food Appi, and Bangladeshi Food Reviewer are eight important names in the realm of food reviewing. Each one of them has a sizeable audience, as well as a big number of people who have commented on their reviews. On the other hand, when it comes to the content of such remarks, there are a number of noticeable distinctions between each of them.

To begin, some reviewers have a greater total number of comments than others have when it comes to the total number of remarks. The user with the most comments, RafsanTheChotobhai, has 1749, and the user with the fewest comments, Petuk couple, has 366. This leads one to believe that RafsanTheChotobha has the greatest number of followers and the highest level of engagement with their audience.

When it comes to the tone of the feedback, each of the reviewers has contributed a unique combination of positive, negative, and sarcastic remarks. However, the percentage of each category of remark differs significantly amongst these sites. For instance, RafsanTheChotobhai has the largest percentage of negative comments, with 910 out of 1749 total, which shows that their audience is more critical of the viewpoints that they express. On the other hand, ZoltanBD has the biggest proportion of positive comments, with 170 out of 359 total, which shows that their audience is more likely to agree with their thoughts and find them to be favorable. In terms of a comparison of the reviews, there are some reviewers, like MetroMan and Khudalagse, who have an equal number of positive, negative, and sarcastic comments. While other platforms, such as Food Appi, have received a greater number of good comments and a lesser number of negative comments overall. This would imply that the readers of Food Appi are more inclined to agree with their thoughts

and look favorably upon them.

Last but not least, it is important to point out that the quality of a reviewer's reviews can be determined by the number and proportion of comments they receive. This is because the number of comments can provide an indication of how relevant and engaged a reviewer is.

Although all eight of the food reviewers have a significant following and have a large number of comments on their reviews, the levels of engagement that each reviewer has with their audience and the ratios of positive, negative, and sarcastic comments that each reviewer receives are significantly different. While other reviewers, like ZoltanBD, have a higher proportion of positive remarks, some reviewers, like RafsanTheChotobhai, have a bigger proportion of negative comments.

Additionally, Food Bank, Bangladesh Eats, KHAIDAI.COM and Food BD are well known Facebook groups related to food. Food Bank has got total of 124 comments. Among them total 73 comments are negative, 18 total positive comments and 33 sarcastic comments. On the other hand, Bangladesh Eats has got total 137 comments. Among them 80 total positive comments, 50 total negative comments and 7 sarcastic comments. Again, KHAIDAI.COM has got total of 269 comments. Among them total 105 comments are negative, 123 total positive comments and 41 sarcastic comments. Moreover, Food BD has got total of 1161 comments. Among them total 866 comments are negative, 9 total positive comments and 266 sarcastic comments. Now write a compare and contrast essay about them.

Food Bank, Bangladesh Eats, KHAIDAI.COM and Food BD are four well-known Facebook groups related to food. Each of them has a significant following and has a large number of comments on their posts. However, when it comes to the nature of those comments, there are some notable differences among them.

Firstly, when it comes to the number of comments, some groups have more comments than others. Food BD has the most comments with 1161, while Food Bank has the least with 124. This suggests that Food BD has the largest following and the most engagement with their audience.

In terms of the sentiment of the comments, all of the groups have a mix of positive, negative, and sarcastic comments. However, the proportion of each type of comment varies among them. For example, Food Bank has the highest proportion of negative comments with 73 out of 124, which suggests that their audience is more critical of their opinions. On the other hand, Bangladesh Eats has the highest proportion of positive comments with 80 out of 137, which suggests that their audience is more likely to agree with their opinions and find them favorable.

When it comes to the comparison of the groups, some groups such as KHAIDAI.COM have a balanced proportion of positive, negative, and sarcastic comments. While others such as Food BD have a higher proportion of negative comments and lower proportion of positive comments. This suggests that the audience of Food BD is more critical of their opinions.

Lastly, it's worth noting that while the number and proportion of comments can give an idea of the popularity and engagement of a group, it indicates the quality of their posts or the relevance of their content

### 5.7.1 Suggested Improvements

Improving the quality of the content is one approach to the issue of negative and sarcastic comments on food bloggers' and reviewers' posts on social media platforms. This can be accomplished by giving more in-depth details regarding the dish under review, including its ingredients, preparation techniques, and nutritional data. In addition, bloggers and reviewers can concentrate on offering a more thorough analysis of the cuisine, including its flavor, texture, and presentation. Engaging the audience by answering questions, addressing issues, and responding to comments is one way to do this. This can foster beneficial connections between the bloggers and reviewers and their audience and help create a sense of community. Bloggers and reviewers can use social media channels to interact with their audience and market their material as well. This can involve the use of hashtags, coming up with catchy captions, and employing social media analytics tools to monitor reach and engagement. Another alternative is to delete offensive information using social media moderation tools and reporting systems, and to teach bloggers and reviewers how to utilize these tools efficiently. The issue of unfavorable and troll remarks on food bloggers' and reviewers' content on social media platforms can be solved in a number of ways. Bloggers and reviewers can foster a more happy and productive online environment for themselves and their audience by enhancing the quality of the content, interacting with the audience, and employing social media moderation tools.

We have also found that in social media platforms like Facebook and Instagram people tend to abuse more by using negative comments against food reviewers than Youtube platform which is a matter of sorrow. There are many different ways to address the difficult issue of abusive content on social media networks. Using social media moderation tools and reporting processes to delete offensive content is one option. This can involve employing software filters to automatically find and delete objectionable language as well as giving users the option to report offensive material. Implementing community rules and norms that specify what constitutes appropriate conduct on the platform is another option. This can involve establishing rules about harassment, bullying, and hate speech as well as supplying users with information and tools on how to behave politely when interacting online. Utilizing machine learning and tools for natural language processing is another strategy for identifying offensive information. This can involve employing sentiment analysis to find derogatory or abusive words as well as image and video analysis to find potentially offensive visual items. In order to combat the issue of offensive content, social media companies can also collaborate with law enforcement officials and other groups. This can involve disclosing details on obnoxious users and helping with investigations into illegal practices like cyberbullying and harassment. Last but not least, social media platforms can contribute to the development of a more welcoming and pleasant online community by fostering the transmission of inspiring and upbeat messages and by enticing users to have courteous and constructive online conversations. In summary, the issue of offensive information on social media sites is complicated and necessitates a diversified approach. Social media platforms can foster a more positive and welcoming online environment for all users by utilizing social media moderation tools and reporting mechanisms, putting community guidelines and policies into place, utilizing machine learning and natural language process-

ing techniques, collaborating with law enforcement agencies and organizations, and promoting uplifting and positive messages.



# Chapter 6

## Conclusion

Web 2.0 allows Internet users to publish, debate, and exchange self-generated ideas across websites. Reviews, posts, blogs, vlogs and online social media platforms supply a wealth of client preference data. Most of them enable users to digitally buy delivery or takeout items (Food Panda, Uber Eats, Pathao Food etc.) and digitally rate the products or services they have eaten. Rapid data creation (from assessments) may impact consumer behavior. Sentiment analysis is the suggested solution. While much has been written and researched on sentiment analysis in several domains and languages, this study approached the problem at numerous levels. An analysis of social media reviews reveals many tendencies (quality of food, customer service, image of a restaurant, pricing, and quantity of food). Based on the dataset's confusion matrix and using accuracy, precision, and recall for each assessed function, we found a high classification performance. Due to customer sentiments on the Bangladeshi food business, Facebook and YouTube may adopt the recommended approach and execute it in English and Bengali. Future study will focus on a larger sample of online reviews so we can predict all the words and phrases customers use to describe food and restaurants on opinion and review websites. The proposed technique will be compared to machine learning. Several restrictions were identified that reduced the reliability of the results in our thesis on sentimental analysis. One major restriction was the very small size of the dataset that was employed. The accuracy of the sentiment analysis may have been increased with a more comprehensive dataset. Additionally, emoji classification was not available. Emojis are frequently utilized to express crucial emotional information in online communications. There was a severe drop in precision because emoji classification wasn't implemented. In addition, the sentiment analysis was double-checked by an entire team of people before being released. Sometimes it was hard to agree on how to feel about a piece of data, and thus reduced the reliability of the study as a whole. These restrictions collectively indicate the necessity for more extensive datasets, as well as sophisticated emoji categorization methods, to enhance the precision of the sentimental analysis. We concluded our thesis on sentiment analysis by identifying several areas for future research and development. Increasing the size and variety of the dataset employed is a primary objective in order to boost the analysis's accuracy. By doing so, we can get more reliable data that accurately reflects the whole. Additionally, incorporating emoji classification into the analysis will be crucial in order to accurately capture the sentiment conveyed by these commonly used symbols. Another important area for future research is the detection of fake reviews by

analyzing the sentiment of comments. This will help to identify and remove biased or manipulated reviews, which can skew the results of the analysis. Overall, these future plans aim to improve the accuracy and robustness of sentimental analysis by addressing limitations and incorporating new techniques.

# Bibliography

- [1] A. Mitra. (2020). Sentiment Analysis Using Machine Learning Approaches (Lexicon based on movie review dataset). *Journal of Ubiquitous Computing and Communication Technologies (UCCT)* (2020), Vol.02/ No.03, Pages: 145-152, <https://www.irojournals.com/jucct/>, DOI: <https://doi.org/10.36548/jucct.2020.3.004>.
- [2] S. Wassan, X. Chen, T. Shen, M. Waqar & NZ. Jhanjhi. (2021). Amazon Product Sentiment Analysis using Machine Learning Techniques. *Revista Argentina de Clínica Psicológica* 2021, Vol. XXX, N°1, 695-703, DOI: [10.24205/03276716.2020.2065](https://doi.org/10.24205/03276716.2020.2065).
- [3] R. S. Jagdale, V. S. Shirsat & S. N. Deshmukh. (2019). Sentiment Analysis on Product Reviews Using Machine Learning Techniques. Springer Nature Singapore Pte Ltd. 2019 P. K. Mallick et al. (eds.), *Cognitive Informatics and Soft Computing, Advances in Intelligent Systems and Computing* 768, [https://doi.org/10.1007/978-981-13-0617-4\\_61](https://doi.org/10.1007/978-981-13-0617-4_61).
- [4] H. Raza, M. Faizan, A. Hamza, A. Mushtaq & N. Akhtar. (2019). Scientific Text Sentiment Analysis using Machine Learning Techniques. (IJACSA) *International Journal of Advanced Computer Science and Applications*, Vol. 10, No. 12, 2019.
- [5] M. R. Hasan, M. Maliha, & M. Arifuzzaman. (2019). Sentiment Analysis with NLP on Twitter Data. *International Conference on Computer, Communication, Chemical, Materials, and Electronic Engineering (IC4ME2)*, 11-12 July 2019, 978-1-7281-3060-6/19/\$31.00 c©2019 IEEE.
- [6] A. Sarlan, C. Nadam, & S. Basri. (2014). Twitter Sentiment Analysis. 2014 *International Conference on Information Technology and Multimedia (ICIMU)*, November 18 – 20, 2014, Putrajaya, Malaysia, 978-1-4799-5423-0/14/\$31.00 ©2014 IEEE.
- [7] Aman, S., & Szpakowicz, S. (2008). Using Roget’s thesaurus for fine-grained emotion recognition. In *Proceedings of the International Joint Conference on NLP (IJCNLP)* (pp. 296–302).
- [8] Aue, A., & Gamon, M. (2005). Customizing sentiment classifiers to new domains: A case study. Technical report, Microsoft Research.
- [9] Bai, X., Padman, R., & Airoidi, E. (2005). On learning parsimonious models for extracting consumer opinions. In *Proceedings of HICSS-05, 38th Annual Hawaii International Conference on System Sciences* (pp. 75–82). Washington, DC: IEEE Computer Society.

- [10] Baram, Y., El-Yaniv, R., & Luz, K. (2004). Online choice of active learning algorithms. *Journal of Machine Learning Research*, 5, 255–291.
- [11] Berger, A. L., Pietra, S. D., & Pietra, V. J. D. (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, 22, 39–71.
- [12] Bondu, A., Lemaire, V., & Poulain, B. (2007). Active learning strategies: A case study for detection of emotions in speech. In *Industrial Conference on Data Mining*, volume 4597 of *Lecture Notes in Computer Science* (pp. 228–241). Springer.
- [13] Brinker, K. (2003). Incorporating diversity in active learning with support vector machines. In *Proceedings of ICML-03, 20th International Conference on Machine Learning* (pp. 59–66). Washington, DC: AAAI Press.
- [14] Zhu, J., Wang, H., & Hovy, E. H. (2008). Learning a stopping criterion for active learning for word sense disambiguation and text classification. In *Proceedings of the International Joint Conference on NLP (IJCNLP)*.
- [15] Wang, B., & Wang, H. (2007). Bootstrapping both product properties and opinion words from Chinese reviews with cross-training. In *Proceedings of the IEEE/WICACM International Conference on Web Intelligence* (pp. 259–262). Washington, DC, USA: IEEE Computer Society.
- [16] R Jamil, I Ashraf, F Rustam, E Saad, A Mehmood & G S Choi. (2021). Detecting sarcasm in multi-domain datasets using convolutional neural networks and long short term memory network model. <https://doi.org/10.7717/peerj-cs.645>.
- [17] M Glenwright, B Tapley, J K. S. Rano & P M. Pexman. (2017). Developing Appreciation for Sarcasm and Sarcastic Gossip: It Depends on Perspective. [https://doi.org/10.1044/2017\\_JSLHR-L-17-0058](https://doi.org/10.1044/2017_JSLHR-L-17-0058).
- [18] TrustPulse Marketing Blog. (2020). Available at: <https://trustpulse.com/online-review-statistics/>.
- [19] Jon Clark. (2020). 15 Online Review Stats Every Marketer Should Know. Available at: <https://www.searchenginejournal.com/online-review-statistics329701#close>.
- [20] A Idris. (2018). Confusion Matrix. Available at: <https://medium.com/awabmohammedomer/confusion-matrix-b504b8f8e1d1>. <https://www.timecamp.com/blog/2016/11/case-study-reviews/>
- [21] A. Navlani. (2018). Decision Tree Classification in Python Tutorial. Available at: <https://www.datacamp.com/tutorial/decision-tree-classification-python>.
- [22] N. Sadki. (2022). Understand AdaBoost and Implement it Effectively. Available at: <https://iq.opengenus.org/adaboost/>.
- [23] S. Polamuri. (2017). How Multinomial Logistic Regression Model Works In Machine Learning. Available at: <https://dataaspirant.com/multinomial-logistic-regression-model-works-machine-learning/>.

- [24] B. Alam. (2022). Implementing Naive Bayes Classification using Python. Available at: <https://hands-on.cloud/implementing-naive-bayes-classification-using-python/>.
- [25] N. Islam. (2021). Support Vector Machines (SVMs). Available at: <https://islam-navaid.medium.com/support-vector-machines-svms-47d53e60a75f>.
- [26] K-Nearest Neighbor(KNN) Algorithm for Machine Learning. (2021). Available at: <https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning>.
- [27] N-Grams. (2022). Available at: <https://deepai.org/machine-learning-glossary-and-terms/n-gram>.
- [28] Raizada, R. (2013, July 12). Illustration of how a Gaussian Naive Bayes (GNB) classifier work[Illustration].[https://www.researchgate.net/figure/Illustration-of-how-a-Gaussian-Naive-Bayes-GNB-classifier-works-For-each-data-point\\_fig1\\_255695722](https://www.researchgate.net/figure/Illustration-of-how-a-Gaussian-Naive-Bayes-GNB-classifier-works-For-each-data-point_fig1_255695722)
- [29] Multinomial Logistic Regression Example. (n.d.). [Graph]. <https://www.statstest.com/multinomial-logistic-regression/>
- [30] Generating Unigram, Bigram, Trigram and Ngrams in NLTK - MLK - Machine Learning Knowledge. (2021, May 17). MLK - Machine Learning Knowledge. <https://machinelearningknowledge.ai/generating-unigram-bigram-trigram-and-ngrams-in-nltk/>
- [31] Wankhade, M., Sekhara Rao, A. C., & Kulkarni, C. (2022, February 7). A survey on sentiment analysis methods, applications, and challenges - Artificial Intelligence Review. SpringerLink. <https://link.springer.com/article/10.1007/s10462-022-10144-1>
- [32] Liapakis, A. (2020). A Sentiment Lexicon-Based Analysis for Food and Beverage Industry Reviews. The Greek Language Paradigm. SSRN Electronic Journal. Published. <https://doi.org/10.2139/ssrn.3606071>
- [33] Budanitsky, A., ]& Hirst, G. (2004). Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 1, 1–49.
- [34] Chambers, N., Tetreault, J., & Allen, J. (2006). Certainty identification in texts: Categorization model and manual tagging results. In J. Shanahan, Y. Qu, & J. Wiebe (Eds.), *Computing attitude and affect in text: Theory and applications* (pp. 143–158). Springer.
- [35] Chesley, P., Vincent, B., Xu, L., & Srihari, R. (2006). Using verbs and adjectives to automatically classify blog sentiment. In *Proceedings of AAAI-CAAW-06, the Spring Symposia on Computational Approaches to Analyzing Weblogs*. Stanford, CA.
- [36] Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge, UK: Cambridge University Press.

- [37] Croft, W., & Lafferty, B. (2003). *Language modeling for information retrieval*. Boston, MA: Kluwer Academic Publishers.
- [38] Dagan, I., & Engelson, S. P. (1995). Committee-based sampling for training probabilistic classifiers. In *Proceedings of ICML-95, 12th International Conference on Machine Learning* (pp. 150–157).
- [39] Dave, K., Lawrence, S., & Pennock, D. M. (2003). Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of WWW-03, 12th International Conference on the World Wide Web* (pp. 519–528). New York: ACM Press.
- [40] De Smet, W., & Moens, M. F. (2007). Generating a topic hierarchy from dialect texts. In *DEXA Workshops* (pp. 249–253). IEEE Computer Society.
- [41] Finn, A., & Kushmerick, N. (2003). Learning to classify documents according to genre. *Journal of the American Society for Information Science*, 57, 1506–1518. Special issue on Computational Analysis of Style.
- [42] Freund, Y., Seung, H. S., Shamir, E., & Tishby, N. (1997). Selective sampling using the query by committee algorithm. *Machine Learning*, 28, 133–168.
- [43] Galley, M., & McKeown, K. (2007). Lexicalized Markov grammars for sentence compression. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference* (pp. 180–187). Rochester, New York: Association for Computational Linguistics.
- [44] Gamon, M. (2004). Sentiment classification on customer feedback data: Noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of COLING-04, the 20th International Conference on Computational Linguistics* (pp. 841–847). Geneva, CH.
- [45] Hatzivassiloglou, V., & McKeown, K. R. (1997). Predicting the semantic orientation of adjectives. In *Proceedings of ACL-97, 35th Annual Meeting of the Association for Computational Linguistics* (pp. 174–181). Madrid, Spain: Association for Computational Linguistics.
- [46] Hatzivassiloglou, V., & Wiebe, J. M. (2000). Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of COLING-00, 18th International Conference on Computational Linguistics* (pp. 299–305). San Francisco, CA: Morgan Kaufmann.
- [47] Hearst, M. A. (1992). Direction-based text interpretation as an information access refinement. In P. Jacobs (Ed.), *Text-based intelligent systems: Current research and practice in information extraction and retrieval* (pp. 257–274). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.
- [48] Huang, T., Dagli, C., Rajaram, S., Chang, E., Mandel, M., Poliner, G., & Ellis, D. (2008). Active learning for interactive multimedia retrieval. *Proceedings of the IEEE*, 96, 648–667.

- [49] Huber, R., Batliner, A., Buckow, J., Nöth, E., Warnke, V., & Niemann, H. (2000). Recognition of emotion in a realistic dialogue scenario. In Proceedings of the International Conference on Spoken Language Processing (Vol. 1, pp. 665–668). Beijing, China.
- [50] Iyengar, V. S., Apte, C., & Zhang, T. (2000). Active learning using adaptive resampling. In Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 92–98). New York: ACM.
- [51] Kamps, J., & Marx, M. (2002). Words with attitude. In Proceedings of the 1st International Conference on Global WordNet (pp. 332–341). Mysore, India.
- [52] Kessler, B., Nunberg, G., & Schütze, H. (1997). Automatic detection of text genre. In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics (pp. 32–38). Somerset, NJ: Association for Computational Linguistics.
- [53] Knight, K., & Marcu, D. (2000). Statistics-based summarization - step one: Sentence compression. In Proceedings of AAAI/IAAI-00, 12th Conference on Innovative Applications of AI (pp. 703–710). San Francisco, CA: AAAI Press.
- [54] Kobayashi, N., Inui, K., & Matsumoto, Y. (2007). Extracting aspect-evaluation and aspect-of relations in opinion mining. In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL) (pp. 1065–1074).
- [55] McCallum, A. K., & Nigam, K. (1998). Employing EM in pool-based active learning for text classification. In Proceedings of ICML-98, 15th International Conference on Machine Learning (pp. 350–358). San Francisco, CA: Morgan Kaufmann Publishers.
- [56] Mishne, G. (2005). Experiments with mood classification in blog posts. In Style2005, 1st Workshop on Stylistic Analysis of Text for Information Access at SIGIR 2005.
- [57] Mishne, G., & de Rijke, M. (2006). A study of blog search. In European Conference on Information Retrieval (pp. 289–301). Berlin, Germany: Springer.
- [58] Mulder, M., Nijholt, A., den Uyl, M., & Terpstra, P. (2004). A lexical grammatical implementation of affect. In Proceedings of TSD-04, 7th International Conference on Text, Speech and Dialogue, volume 3206 of Lecture Notes in Computer Science (pp. 171–178). Berlin, Germany: Springer.
- [59] Mullen, T., & Collier, N. (2004). Sentiment analysis using support vector machines with diverse information sources. In Proceedings of EMNLP-04, 9th Conference on Empirical Methods in Natural Language Processing (pp. 412–418). Barcelona, Spain.
- [60] Xu, Z., Yu, K., Tresp, V., Xu, X., & Wang, J. (2003). Representative sampling for text classification using support vector machines. In European Conference on Information Retrieval (pp. 393–407). Berlin, Germany: Springer.

- [61] Zagibalov, T., & Carroll, J. (2008). Unsupervised classification of sentiment and objectivity in Chinese text. In Proceedings of the International Joint Conference on NLP (IJCNLP).
- [62] Mullen, T., & Malouf, R. (2006). A preliminary investigation into sentiment analysis of inform
- [63] political discourse. In AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW 2006) (pp. 125–126)
- [64] Nguyen, H. T., & Smeulders, A. (2004). Active learning using pre-clustering. In Proceedings of ICML-04, 21st International Conference on Machine Learning (p.79). New York: ACM Press.
- [65] Nijholt, A. (2003). Humor and embodied conversational agents. CTIT Technical Report series No. 03-03, University of Twente.
- [66] Osugi, T. (2005). Exploration-Based Active Machine Learning. Master’s thesis, University of Nebraska.
- [67] Pang, B., & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of ACL-04, 42nd Meeting of the Association for Computational Linguistics (pp. 271–278). East Stroudsburg, PA: Association for Computational Linguistics.
- [68] Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. In Proceedings of EMNLP-02, the Conference on Empirical Methods in Natural Language Processing (pp. 79–86). Philadelphia, PA: Association for Computational Linguistics.
- [69] Pedersen, T. (2001). A decision tree of bigrams is an accurate predictor of word sense. In Proceedings of the Second Annual Meeting of the North American Chapter of the Association for Computational Linguistics (pp. 79–86).
- [70] Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14, 130–137.
- [71] Raina, R., Battle, A., Lee, H., Packer, B., & Ng, A. Y. (2007). Self-taught learning: transfer learning from unlabeled data. In Proceedings of ICML-07, 24th International Conference on Machine Learning (pp. 759–766). New York: ACM.
- [72] Riloff, E., Wiebe, J., & Wilson, T. (2003). Learning subjective nouns using extraction pattern bootstrapping. In Proceedings of CoNLL-03, 7th Conference on Natural Language Learning (pp. 25–32). Edmonton, CA.
- [73] Roy, N., & McCallum, A. (2001). Toward optimal active learning through sampling estimation of error reduction. In Proceedings of ICML-01, 18th International Conference on Machine Learning (pp. 441–448). San Francisco, CA: Morgan Kaufmann.
- [74] Rubin, V. L., Liddy, E. D., & Kando, N. (2006). Certainty identification in texts: Categorization model and manual tagging results. In J. Shanahan, Y. Qu, & J. Wiebe (Eds.), *Computing attitude and affect in text: Theory and applications* (pp. 61–76). Berlin, Germany: Springer.



- [75] Saar-Tsechansky, M., & Provost, F. (2004). Active sampling for class probability estimation and ranking. *Machine Learning*, 54, 153–178.
- [76] Salvetti, F., Lewis, S., & Reichenbach, C. (2004). Impact of lexical filtering on overall opinion polarity identification. In *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*. Stanford, CA.
- [77] Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In *Computational learning theory* (pp. 287–294).
- [78] Tong, R., & Yager, R. (2006). Characterizing buzz and sentiment in internet sources: Linguistic summaries and predictive behaviors. In J. Shanahan, Y. Qu, & J. Wiebe (Eds.), *Computing attitude and affect in text: Theory and applications* (pp. 281–296). Springer.
- [79] Tong, S., & Koller, D. (2002). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2, 45–66.
- [80] Turney, P. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL-02, 40th Annual Meeting of the Association for Computational Linguistics* (pp. 417–424). Philadelphia, PA: Association for Computational Linguistics.
- [81] Viola, P., & Jones, M. (2001). Robust real-time object detection. Technical report, Cambridge Research Lab, Compaq.
- [82] Wang, B., & Wang, H. (2007). Bootstrapping both product properties and opinion words from Chinese reviews with cross-training. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence* (pp. 259–262). Washington, DC, USA: IEEE Computer Society.
- [83] Whitelaw, C., Garg, N., & Argamon, S. (2005). Using appraisal taxonomies for sentiment analysis. In *Proceedings of MCLC-05, 2nd Midwest Computational Linguistic Colloquium*. Columbus, OH.
- [84] Wiebe, J. (2000). Learning subjective adjectives from corpora. In *Proceedings of AAAI-00, 17th Conference of the American Association for Artificial Intelligence* (pp. 735–740). Austin, TX: AAAI Press/The MIT Press.