Forgery Detection Using Machine Learning (Fake Review Detection)

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

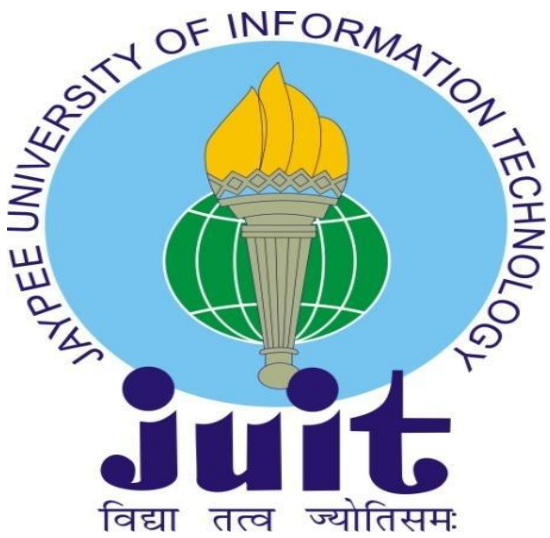
By

Kanishak Vyas (191552)
Devesh Kumar Singh (191556)

Under the supervision of

Dr. Abhilasha Sharma

to



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

We hereby declare that the work presented in this report entitled Forgery Detection Using Machine Learning (Credit Card Fraud) in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2015 to December 2015 under the supervision of **Dr. Abhilasha Sharma** Assistant Professor (SG) Dept. of CSE & IT.

We also authenticate that we have carried out the above-mentioned project work under the proficiency stream **Information Security**

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Kanishak Vyas, 191552

Devesh Kumar Singh, 191556

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr Abhilasha Sharma
Assistant Professor (SG)
CSE & IT
Dated:

Plagiarism Certificate

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ……

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: _____ **Department:** _____ **Enrolment No** _____

Contact No. _____ **E-mail.** _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

_____

_____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ……..(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	**Excluded**	**Similarity Index (%)**	**Generated Plagiarism Report Details** (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature **Librarian**

……

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

Acknowledgement

Firstly, we express my heartiest thanks and gratefulness to almighty God for His divine blessing makes us possible to complete the project work successfully.

We are really grateful and wish our profound my indebtedness to Supervisor Dr. Abhilasha Sharma, Assistant Professor (SG), Department of CSE/IT Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of our supervisor in the field of "Machine Learning" to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stage have made it possible to complete this project.

We would like to express my heartiest gratitude to Dr. Abhilasha Sharma, Department of CSE & IT, for her kind help to finish my project.

We would also generously welcome each one of those individuals who have helped us straight forwardly or in a roundabout way in making this project a win. In this unique situation, we might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated our undertaking.

Finally, we must acknowledge with due respect the constant support and patients of our parents.

Kanishak Vyas
(191552)

Devesh Kumar Singh
(191556)

Table of Content

List of Abbreviations

Serial No.	Abbreviations	Meaning
1	E-Commerce	Electronic Commerce
2	TF-IDF	Term Frequency – Inverse Document Frequency
3	SMS	Short Message Service
4	AMT	Amazon Mechanical Turk
5	SLM	Semantic Language Model
6	CSV	Comma Separated Values
7	ML	Machine Learning
8	Algo’s	Algorithms
9	i.e.	That is
10	KNN	K-Nearest Neighbor
11	RF	Random Forest
12	DT	Decision Tree
13	XGB	eXtreme Gradient Boosting
14	SVM	Support Vector Machines
15	NB	Naïve Bayes
16	w.r.t.	With respect to

List of Figures

List of Graphs

List of Tables

Abstract

In e-commerce, user reviews can play a significant role in determining the revenue of an organization. Online users rely on reviews before making decisions about any product and service. As such, the credibility of online reviews is crucial for businesses and can directly affect companies' reputation and profitability. That is why some businesses are paying spammers to post fake reviews. These fake reviews exploit consumer purchasing decisions. Consequently, the techniques for detecting fake reviews have extensively been explored in the past twelve years. However, there still lacks a survey that can analyze and summarise the existing approaches. To bridge up the issue, this survey paper details the task of fake review detection, summing up the existing datasets and their collection methods. It analyses the existing feature extraction techniques. It also summarises and analyses the existing techniques critically to identify gaps based on two groups: traditional statistical machine learning and deep learning methods. Further, we conduct a benchmark study to investigate the performance of different neural network models and transformers that have not been used for fake review detection yet. The experimental results on two benchmark datasets show that RoBERTa performs about 7% better than the state-of-the-art methods in a mixed domain for the deception dataset with the highest accuracy of 91.2%, which can be used as a baseline for future studies. Finally, we highlight the current gaps in this research area and the possible future directions.

CHAPTER 1
INTRODUCTION

1.1 Introduction

Fake reviews have become a serious issue in the digital world. Fake reviews can mislead consumers, damage the reputation of businesses, and cause financial harm. Machine learning techniques can be used to identify fake reviews and prevent their spread.

Fake review detection is the process of identifying whether a review is genuine or not. Machine learning algorithms can be trained on large datasets of reviews and their corresponding labels (genuine or fake) to learn patterns and features that distinguish between genuine and fake reviews.

Reviews play a very vital role for a customer before buying any goods from any E-Commerce site as it helps a customer in making a correct decision to whether to buy a item or not and fake reviews misguide a customer by reviewing it as a good product considering the fact that it's useless for the customer. It is important to detect and remove the fake reviews from various E-commerce, Hotel Booking sites which can help in making a customer's decision fruitful and correct.

1.1.1 Fake Review

A fake review may be defined as a fraudulent or deceptive evaluation of a product, service, or business, typically posted online. These reviews may be created by individuals or companies with the intention of manipulating the perception of the product or service by misleading consumers. Fake reviews can be positive, negative, or neutral and are often used to artificially boost or damage the reputation of a business or product. They can also be used to create an illusion of support or opposition for a particular idea or cause.

1.1.2 Type of Fake Review Frauds

Fake reviews are a type of online fraud that involve the creation of false or misleading reviews in order to influence consumer behaviour. There are several types of fake review frauds, including:

1. Paid Reviews: These are reviews that are written by individuals who have been paid to write positive reviews about a product or service. The reviews are often vague and may not include any specific details about the product or service.

2. Review Swapping: This is when businesses agree to exchange positive reviews with each other in order to boost their own ratings. This is usually done between businesses that offer similar products or services.

3. Incentivized Reviews: This is when individuals are offered incentives, such as discounts or free products, in exchange for writing positive reviews. These reviews are often biased and may not be a true reflection of the product or service.

4. Negative Reviews: Sometimes businesses or individuals may write fake negative reviews about their competitors in order to make their own products or services look better by comparison.

5. Employee Reviews: This is when employees of a business write fake reviews about their employer's products or services. These reviews are often overly positive and may not accurately reflect the true quality of the product or service.

1.1.3 Fake Review Fraud in India

In India many online sites like Amazon, Zomato and various other sites are full of Fake reviews and Indian Government is taking action towards them,

recently a guideline was issued which came into effect from November 25, 2022 which was issued to curb the fake reviews from various E-Commerce sites.

Under this the Government has instituted a new standard named “IS 19000:2022” under BIS , which requires the platforms to have review administrators , mandating them to moderate reviews using some tools of by manually altering the reviews.

And also issued a guideline that if any site is found with any Fake review then it will be highly penalized for practising unfair practices and a penal action from the consumer court can be taken.

1.2 Problem Statement

Nowadays people write unworthy positive reviews about products to promote them and in some cases write malicious negative reviews just to degrade the reputation of a product or service. And some of these have ads and promotions which do not have any link with the product.

So the first challenge is that a word can be positive for a situation and at the same time can be negative for any other situation for eg. While buying a laptop the word long used for battery can be a positive but the same word used for boot time will be a negative review.

The second challenge is that people do not always express their views the same way and majority of conventional text processing methods make the assumption that minor textual variations do not significantly alter meaning. However, in opinion mining, whether the service was excellent or not makes a significant impact.

1.3 Objectives

The main objective of using machine learning for fake review detection is to identify and flag fake or fraudulent reviews on online platforms. These fake reviews can be misleading and harmful to consumers who rely on reviews to make informed decisions about products or services. By detecting and removing fake reviews, the authenticity and credibility of the review platform can be maintained, and consumers can have more confidence in the reviews they read.

Specifically, the objectives of fake review detection using machine learning include:

1. Identifying fake reviews: Machine learning models can be trained to identify patterns and characteristics of fake reviews, such as the use of certain words, the frequency of reviews from certain users, or unusual review timings.

2. Improving review quality: By removing fake reviews, the overall quality of reviews on the platform can be improved, providing more accurate and useful information to consumers.

3. Maintaining platform credibility: A review platform that is known for containing fake reviews can lose credibility and trust among consumers. By using machine learning to detect and remove fake reviews, the platform can maintain its reputation for providing authentic and trustworthy reviews.

4. Saving time and resources: Manually reviewing each and every review for authenticity is a time-consuming and resource-intensive task. By using machine learning to automatically flag fake reviews, platform owners can save time and resources that can be used for other purposes.

Overall, the main objective of fake review detection using machine learning is to ensure that consumers are able to make informed decisions based on authentic and reliable reviews.

1.4 Methodology

1. Data collection: The first step is to collect data from different sources, including online marketplaces, social media platforms, and review sites. These data sources can provide labelled data that includes both fake and genuine reviews. It is important to ensure that the data is diverse and represents various product categories, industries, and languages.

2. Data pre-processing: Once the data is collected, it needs to be pre-processed to remove irrelevant information and perform text cleaning operations such as tokenization, stemming, and stop-word removal. The pre-processed data is then converted into a numerical representation, such as a bag of words or TF-IDF matrix.

3. Feature selection: Feature selection is important to reduce the dimensionality of the data and improve the model's performance. This step involves selecting the most relevant features that contribute to the classification of reviews as fake or genuine. Feature selection techniques include mutual information, chi-square test, and correlation-based feature selection.

4. Model selection: Machine learning models such as logistic regression, support vector machines, decision trees, and random forests can be used to classify reviews. The model selection depends on the characteristics of the data and the performance metrics required. The models can be trained using a supervised learning approach, where labelled data is used to train the model, or unsupervised learning approach, where the model learns patterns in the data without any labels.

5. Model evaluation: Once the model is trained, it is important to evaluate its performance using metrics such as accuracy, precision, recall, and F1 score. Cross-validation techniques such as k- fold cross-validation can be used to ensure the model's generalizability.

6. Model Compression: Best model in all applied models is selected as the best one

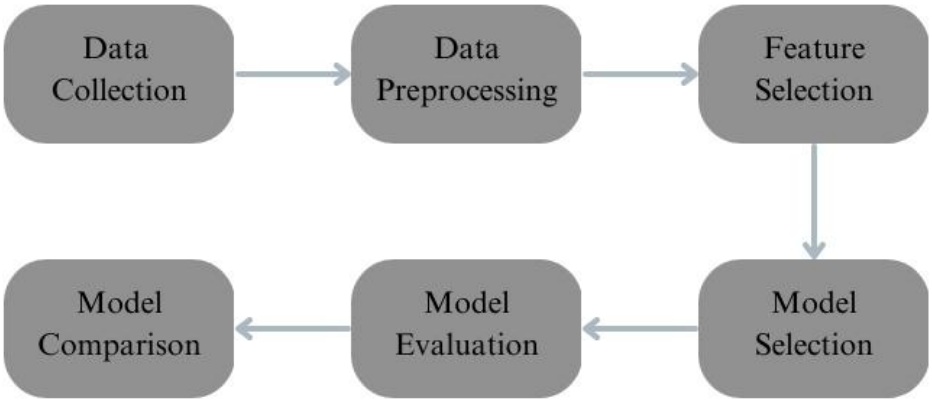


Fig 1. System Methodology

1.5 Organization

In Chapter 2 we talked about the Literature we gone through for the study of the various algorithms we applied on the dataset and also to get the knowledge of how to make the model more accurate and precise. We read the articles mentioned in the Literature Survey to enhance our knowledge on Machine Learning and its algorithms.

In Chapter 3 we elaborated system design of our model, we explained each and every step we took to build our model in a proper manner. First of all we talked about the data set we used and explained it. Next we explained the various steps we performed like data visualization, correlating the data, classifying, next we did the cleaning of unnecessary columns and at last did vectorization of data and then applied all the algorithms..

Chapter 4 tells you about the result we got and the steps we performed to get the best algorithm for our dataset.

Chapter 5 talks about the conclusions we found by building our model.

CHAPTER 2
LITERATURE SURVEY

Machine learning techniques have been used to a variety of issues during the past few decades, including face recognition, speech recognition, font recognition, fraud detection, and disease diagnosis. The fight against spam, a problem that is spreading across numerous web applications like SMS, email, and blogs, has also been studied in recent years. Below, we provide the methods for spotting bogus reviews along with their benefits and drawbacks.

Supervised Learning in Detecting Fake Reviews:-

To determine whether reviews are false, supervised learning techniques are employed. The supervised learning methods that are currently used in the literature will be compiled in this section. created a supervised learning system to look at duplicate reviews to find false reviews. There were two phases to the suggested model. Unigram and bigram were the features employed in the first phase, and Naive Bayes, Random forest, and support vector machine were used as the classification algorithms. The performance of the classification methods was improved in the second phase using two ensemble methods (stacking and voting). The AMT dataset results demonstrated that the ensemble approaches outperformed the Naive Bayes random forest and SVM classification algorithms in terms of performance. The effectiveness of identifying bogus reviews can be improved by using simple feature and ensemble algorithms.

Unsupervised Learning in Detecting Fake Reviews:-

Depending on how challenging it is to get adequately labelled datasets, supervised learning may not always be the best option. The unsupervised learning methods that are now in use are compiled in this subsection from the literature . This problem can be solved via unsupervised learning because it does not require tagged data. established a Semantic Language Model (SLM)

and an unsupervised model to identify bogus reviews. The hypothesis put out by Jindal and Liu [4] that two duplicate reviews were classified as false reviews was followed by the suggested model. The detection of bogus reviews and manual confirmation of them were done using the cosine similarity algorithm. In contrast, the reviews that did not have a cosine resemblance with any other reviews over a certain level were retained as accurate reviews and were not personally examined. The dataset from Amazon.com contains 54,618 reviews, of which 6% were labelled as fake. SLM method was used to give each review a spamming score.

Semi-Supervised Learning in Detecting Fake Reviews:-

Semi-supervised learning is method in machine learning which is used to identify fake reviews using unspecified data, since tagged reviews are hard to find. This subsection summarizes the semi-supervised learning methods available in the literature presented in Table 1.

Sr. No	Ref	Summary				Data Set Information		Performance Metric	
		Problem	Objective	Methodology/ Algorithm	Features/Parameters	Name	Comments	Accuracy	F1-Score
1	[10]	Finding Duplicate Reviews	To find duplicate reviews	Unsupervised Model and developed SLM	Considering Duplicate reviews Fake	Amazon Reviews	Outperformed SVM	87%	
2	[15]	Selecting Appropriate Features in Reviews	To find fake reviews from yelp	Decision Tree Method	Feature selection method	1.Reviews from Yelp.com 2.Gold standard dataset consists of Three Domains	The performance could be improved by considering the data correlation in selecting the appropriate features.	Yelp: 76.91% Hotel Domain: 78.3% Restaurant Domain: 81.8% Doctor Domain: 75.0%	
3	[19]	Searching Fake Reviews	To find fake reviews in reviews about a hotel and restaurant	Unsupervised topic sentiment joint probabilistic method	Latent Dirichlet Allocation (LDA)	Yelp Chi Dataset	1. Incorporating behavioral features with LDA can improve the performance. 2. Integrating the proposed model with work could improve the performance		Restaurant: 83.92% Hotel: 85.03%
4	[17]	To do the sentimental analysis on Amazon reviews	To perform sentimental Analysis on AMT Dataset	Unsupervised Multi-iterative graph-based method	Content Based Features behavior based features Relation	1. AMT Dataset 2. Collected reviews from Amazon.com	1. Combined features improved the performance compared to a single model 2. The performance could be enhanced by integrating network structure with iterative algorithm	AMT : 95.3% Crown sourced : 93%	

					based features				
5	[13]	Searching and removing duplicate fake reviews	To find and remove fake duplicate reviews from JD	Unsupervised Learning	LDA	Dataset collected from JD.com	Considering duplicate content as fake is unreliable	96.42%	

Table 1. Literature Survey

3.1 Dataset Used

The dataset contains reviews which were made on products by various people and are categorized as cool, useful and funny. There are a total of 10000 reviews.

The Dataset was taken from Kaggle and was highly imbalanced; it was in CSV format as in these kinds of documents lines include an example corresponding to a single voice recording.

The dataset is shown below:

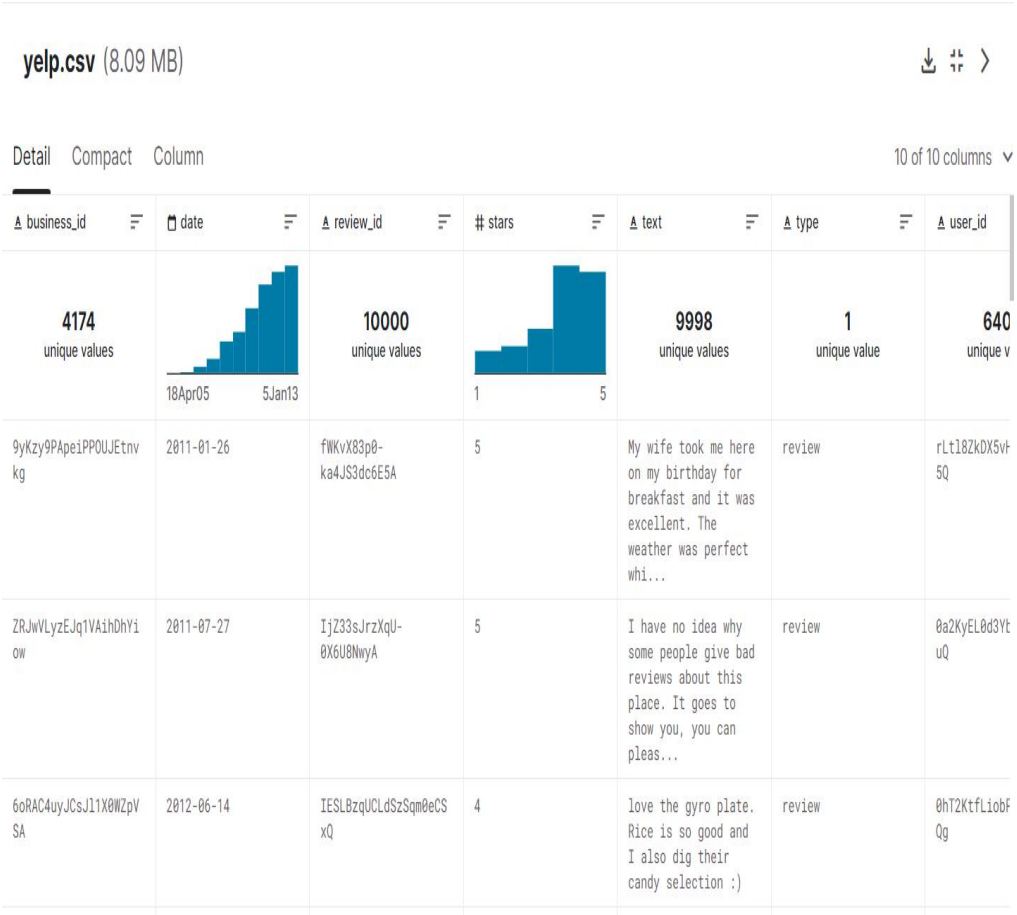


Fig 2. Dataset Description

3.2 Dataset Features

The Dataset has the following features as shown in the table below:

Column number	Column Name	Description
1	business_id	Unique Business ID
2	date	Date of Review
3	review_id	Review ID
4	stars	Stars given by the user
5	text	Review given by the user
6	type	Type of text entered
7	user_id	Unique Used ID
8	cool	The no of cool votes received
9	useful	The no of useful votes received
10	funny	The no of funny votes received

Table 2 – Dataset Features

3.3 Importing the necessary libraries and packages in python

In this step we imported various libraries and packages which we were going to use in our system, we imported the following packages/libraries:

1. Pandas: It is a python library used for working with datasets.

2. Numpy : It is a python library used while working with arrays and it stands for Numerical Python

3. Matplotlib.pyplot: It is a collection of functions which make matplotlib work like MATLAB.

4. Seaborn : It is a library used for making statistical graphs in python

5. Ntlk: It stands for Natural Language Toolkit and is used to pre-process text data for further analysis like with ML models.

6. String: It's a built in module and we have to import it before using any of it constraints and classes.

7. Math: It's a built in module that can be used for mathematical tasks.

8. Sklearn: It stands for Scikit-learn and is the most useful and robust library for using ML in python.

The code for the same is shown below:

```
#importing Necessary libraries and packages
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.corpus import stopwords
import string
import math
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix, accuracy_score, roc_auc_score, roc_curve
from sklearn.model_selection import GridSearchCV
%matplotlib inline
```

Fig 3. Importing Libraries and Packages

3.4 Loading and analyzing the dataset

In this step we loaded the dataset using the read_csv function of Pandas module and also analyzed it by checking the shape of dataset, name of columns, datatype of each column and at last checked a few entries in the dataset.

The code of the following is shown below:

```
#Loading the dataset and analyzing it
```

```
data = pd.read_csv('yelp.csv')
print("Shape of the dataset:")
print(data.shape)
print("Column names:")
print(data.columns)
print("Datatype of each column:")
print(data.dtypes)
print("Few dataset entries:")
print(data.head())
data.describe(include='all')
```

```
Shape of the dataset:
(10000, 10)
Column names:
Index(['business_id', 'date', 'review_id', 'stars', 'text', 'type', 'user_id',
       'cool', 'useful', 'funny'],
      dtype='object')
Datatype of each column:
business_id    object
date           object
review_id      object
stars           int64
text           object
type           object
user_id        object
cool            int64
useful          int64
funny           int64
dtype: object
Few dataset entries:
            business id       date             review id  stars  \
```

```
0  9yKzy9PApeiPPOUJEtnvkg  2011-01-26  fWKvX83p0-ka4JS3dc6E5A     5
1  ZRJwVLyzEJq1VAihDhYiow  2011-07-27  IjZ33sJrzXqU-0X6U8NwyA     5
2  6oRAC4uyJCsJl1X0WZpVSA  2012-06-14  IESLBzqUCLdSzSqm0eCSxQ     4
3  _1QQZuf4zZOyFCvXc0o6Vg  2010-05-27  G-WvGaISbqqaMHlNnByodA     5
4  6ozycU1RpktNG2-1BroVtw  2012-01-05  1uJFq2r5QfJG_6ExMRCaGw     5

                                                text    type  \
0  My wife took me here on my birthday for breakf...  review
1  I have no idea why some people give bad review...  review
2  love the gyro plate. Rice is so good and I als...  review
3  Rosie, Dakota, and I LOVE Chaparral Dog Park!!...  review
4  General Manager Scott Petello is a good egg!!!...  review

                user_id  cool  useful  funny
0  rLtl8ZkDX5vH5nAx9C3q5Q     2       5      0
1  0a2KyEL0d3Yb1V6aivbIuQ     0       0      0
2  0hT2KtfLiobPvh6cDC8JQg     0       1      0
3  uZetl9T0NcROGOyFfughhg     1       2      0
4  vYmM4KTsC8ZfQBg-j5MWkw     0       0      0
```

	business_id	date	review_id	stars	text	type	user_id	cool	useful	funny
count	10000	10000	10000	10000.000000	10000	10000	10000	10000.000000	10000.000000	10000.000000
unique	4174	1995	10000	NaN	9998	1	6403	NaN	NaN	NaN
top	JokKtdXU7zXHcr20Lrk29A	2011-03-28	fWKvX83p0-ka4JS3dc6E5A	NaN	Great service	review	fczQCSmaWF78toLEmb0Zsw	NaN	NaN	NaN
freq	37	21	1	NaN	2	10000	38	NaN	NaN	NaN
mean	NaN	NaN	NaN	3.777500	NaN	NaN	NaN	0.876800	1.409300	0.701300
std	NaN	NaN	NaN	1.214636	NaN	NaN	NaN	2.067861	2.336647	1.907942
min	NaN	NaN	NaN	1.000000	NaN	NaN	NaN	0.000000	0.000000	0.000000
25%	NaN	NaN	NaN	3.000000	NaN	NaN	NaN	0.000000	0.000000	0.000000
50%	NaN	NaN	NaN	4.000000	NaN	NaN	NaN	0.000000	1.000000	0.000000
75%	NaN	NaN	NaN	5.000000	NaN	NaN	NaN	1.000000	2.000000	1.000000
max	NaN	NaN	NaN	5.000000	NaN	NaN	NaN	77.000000	76.000000	57.000000

Fig 4. Loading and Analyzing the Dataset

3.5 Creating a new column for no of words in review

In this step we have created a new column to enter the length or find the no of words that are there in a review using the apply(len) function the code of the following is shown below :

```
#creating a new coloumn for the number of words in the review
```

```
data['length'] = data['text'].apply(len)
data.head()
```

	business_id	date	review_id	stars	text	type	user_id	cool	useful	funny	length
0	9yKzy9PApeiPPOUJEtnvkg	2011-01-26	fWKvX83p0-ka4JS3dc6E5A	5	My wife took me here on my birthday for breakf...	review	rLtl8ZkDX5vH5nAx9C3q5Q	2	5	0	889
1	ZRJwVLyzEJq1VAihDhYiow	2011-07-27	ljZ33sJrzXqU-0X6U8NwyA	5	I have no idea why some people give bad review...	review	0a2KyEL0d3Yb1V6aivbluQ	0	0	0	1345
2	6oRAC4uyJCsJl1X0WZpVSA	2012-06-14	IESLBzqUCLdSzSqm0eCSxQ	4	love the gyro plate. Rice is so good and I als...	review	0hT2KtfLiobPvh6cDC8JQg	0	1	0	76
3	_1QQZuf4zZOyFCvXc0o6Vg	2010-05-27	G-WvGaISbqqaMHlNnByodA	5	Rosie, Dakota, and I LOVE Chaparral Dog Park!!...	review	uZetl9T0NcROGOyFfughhg	1	2	0	419
4	6ozycU1RpktNG2-1BroVtw	2012-01-05	1uJFq2r5QfJG_6ExMRCaGw	5	General Manager Scott Petello is a good egg!!!...	review	vYmM4KTsC8ZfQBg-j5MWkw	0	0	0	469

Fig 5. Creation of new column length

3.6 Visualization

In this step we perform the step of visualizing the dataset to find the correlation between stars and the length of review , we are using seaborn to plot the graph.

Visualization is a very important step while applying the algorithms as it helps in a better understanding of the dataset and its features.

We will visualize the dataset by making a Histogram of the stars and the length.

The code of the following is shown below :

```
#performing the visulaization
```

```
graph = sns.FacetGrid(data=data,col='stars')
graph.map(plt.hist,'length',bins=50,color='blue')
```

```
<seaborn.axisgrid.FacetGrid at 0x18387a13070>
```

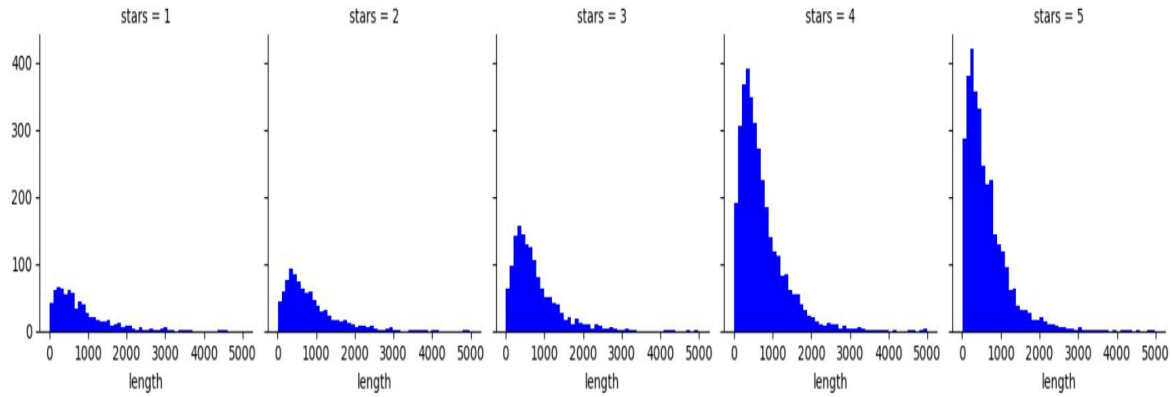


Fig 6. Visualization

3.7 Finding mean values of the vote columns

There are three vote columns – Cool, Useful and Funny , so in this step we will find the mean values of the votes with respect to the starts given to the reviews.

Here we will be using the mean function

The code of the following is :

```
#finding the mean value of vote columns
```

```
stval = data.groupby('stars').mean()
stval
```

```
C:\Users\kanis\AppData\Local\Temp\ipykern
GroupBy.mean is deprecated. In a future v
columns which should be valid for the fun
  stval = data.groupby('stars').mean()
```

stars	cool	useful	funny	length
1	0.576769	1.604806	1.056075	826.515354
2	0.719525	1.563107	0.875944	842.256742
3	0.788501	1.306639	0.694730	758.498289
4	0.954623	1.395916	0.670448	712.923142
5	0.944261	1.381780	0.608631	624.999101

Fig 7. Mean Values of Votes

3.8 Correlation between the vote columns

In this step we find the correlation between the three vote columns and found that there was a negative correlation between Cool and Useful, Cool and Funny & Cool and Length columns and a positive correlation between the Columns Funny and Useful, Funny and Length, Useful and Length. By this we concluded that longer reviews tend to be funny and useful.

The code for the following is shown below :

```
#finding correlation between the vote columns
```

```
stval.corr()
```

	cool	useful	funny	length
cool	1.000000	-0.743329	-0.944939	-0.857664
useful	-0.743329	1.000000	0.894506	0.699881
funny	-0.944939	0.894506	1.000000	0.843461
length	-0.857664	0.699881	0.843461	1.000000

Fig 8. Correlation between vote columns

3.9 Classifying the dataset and splitting it in reviews and stars

In this step we are going to classify the dataset and will split it in reviews and stars.

It is being done to help us in running the algorithms more efficiently and get the best result possible from the dataset.

The code is shown below :

```
#classifiying the dataset and splitting it in reviews and stars
```

```
data_classes = data[(data['stars']==1) | (data['stars']==3) | (data['stars']==5)]
data_classes.head()
print(data_classes.shape)

x = data_classes['text']
y = data_classes['stars']
print(x.head())
print(y.head())
```

```
(5547, 11)
0    My wife took me here on my birthday for breakf...
1    I have no idea why some people give bad review...
3    Rosie, Dakota, and I LOVE Chaparral Dog Park!!...
4    General Manager Scott Petello is a good egg!!!...
6    Drop what you're doing and drive here. After I...
Name: text, dtype: object
0    5
1    5
3    5
4    5
6    5
Name: stars, dtype: int64
```

Fig 9. Classifying and splitting the Dataset

3.10 Data Cleaning

It is the most important step as the data which is to be used while performing the ML algo's must be cleaned and must not contains the stopwords like the, a , an, in etc as these words make the processing time longer, therefore it is important to remove these words.

The code of the same is as follows :

```
#data cleaning
```

```
def text_process(text):
    nopunc = [char for char in text if char not in string.punctuation]
    nopunc = ''.join(nopunc)
    return [word for word in nopunc.split() if word.lower() not in stopwords.words('english')]
```

Fig 10. Data Cleaning

3.11 Vectorization

Vectorization is a major step done to speed up the python code without using loops. Using these can help in minimizing the running time of code efficiently. Various operations are being performed over vector such as dot product of vectors.

We also used the vectorization in our system the code is shown below :

```
#vectorization
```

```
vocab = CountVectorizer(analyzer=text_process).fit(x)
print(len(vocab.vocabulary_))
r0 = x[0]
print(r0)
vocab0 = vocab.transform([r0])
print(vocab0)
print("Getting the words back:")
print(vocab.get_feature_names_out()[19648])
print(vocab.get_feature_names_out()[10643])
```

```
31336
My wife took me here on my birthday for breakfast and it was excellent.  The weather was perfect which made sitting outside ove
rlooking their grounds an absolute pleasure.  Our waitress was excellent and our food arrived quickly on the semi-busy Saturday
morning.  It looked like the place fills up pretty quickly so the earlier you get here the better.

Do yourself a favor and get their Bloody Mary.  It was phenomenal and simply the best I've ever had.  I'm pretty sure they only
use ingredients from their garden and blend them fresh when you order it.  It was amazing.

While EVERYTHING on the menu looks excellent, I had the white truffle scrambled eggs vegetable skillet and it was tasty and del
icious.  It came with 2 pieces of their griddled bread with was amazing and it absolutely made the meal complete.  It was the b
est "toast" I've ever had.

Anyway, I can't wait to go back!
  (0, 292)      1
  (0, 1213)     1
  (0, 1811)     1
  (0, 3537)     1
  (0, 5139)     1
```

```
(0, 5256)     2
(0, 6275)     1
(0, 8521)     1
(0, 10646)    1
(0, 10647)    1
(0, 11128)    2
(0, 11479)    1
(0, 11779)    1
(0, 12206)    2
(0, 12221)    1
(0, 12297)    1
(0, 12386)    1
(0, 12675)    1
(0, 12689)    1
(0, 13135)    1
(0, 13186)    1
(0, 14247)    1
(0, 15385)    1
(0, 16292)    1
(0, 16412)    1
:     :
(0, 23318)    1
(0, 23801)    1
(0, 23902)    1
(0, 23976)    1
(0, 24080)    1
(0, 24177)    1
(0, 24544)    2
(0, 24972)    2
(0, 26383)    1
(0, 26543)    1
```

```
  (0, 26978)    1
  (0, 27029)    1
  (0, 27068)    1
  (0, 28403)    1
  (0, 28735)    1
  (0, 29230)    1
  (0, 29313)    1
  (0, 29620)    1
  (0, 30135)    1
  (0, 30240)    1
  (0, 30471)    1
  (0, 30488)    1
  (0, 30672)    1
  (0, 30854)    1
  (0, 30900)    1
Getting the words back:
hygienist
absence
```

Fig 11. Vectorization

3.12 Vectorization of whole review set and checking of sparse matrix

In this step we performed vectorization of the whole review set and also checked the sparse matrix which is termed as matrix in which most of the elements are 0 and it contains data in few positions, also the memory consumed by this is made up of zeroes.

The code of the following is as shown below:

```
#vecotorization of whole review set and checking the sparse matrix
```

```
x = vocab.transform(x)
print("Shape of the sparse matrix: ", x.shape)
print("Non-Zero occurences: ",x.nnz)

density = (x.nnz/(x.shape[0]*x.shape[1]))*100
print("Density of the matrix = ",density)
```

```
Shape of the sparse matrix:  (5547, 31336)
Non-Zero occurences:  312457
Density of the matrix =  0.17975812697942373
```

Fig 12. Vectorization and Checking of Sparse Matrix

3.13 Splitting the dataset into testing and training set

It is very important to train and develop an algorithm to help predict the data if machine learning is to ensure efficient model computation. The information collected by specialists is usually divided into datasets, which are then used for training and testing. These sets typically include a training, validation, and test set. The method can be used to make educated guesses on de-identified data to evaluate the overall performance of ML algorithms. This is a quick and easy way to do it, and the results allow testing the effectiveness of ML algorithms against the complexity of predictive modelling. Although the process is easy to use and compile, there are cases where the process is not worth implementing immediately, such as when the database is inconsistent and the database is small, or when more setup is needed. After developing the

model using a supervised learning technique, mtxlel was initially added to the training data. The current model or the model we built uses the training data to get the result, and based on the result, we can determine whether the model has succeeded in predicting the prices or not.

In our model we use a train test distribution with a sample size of 0.2, the method results in two X train, X test, Y train and Y test, these values are then stored in an array, now the stored. Value would be used in his analysis for fun. For this we use the sklearn library in python.

The code for the following is shown below:

```
#splitting the dataset into testing and training set

x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=101)
```

Fig 13. Splitting the Dataset in Testing and Training set

3.14 Models Used

We used six machine learning algorithms in our project i.e. as follows:

1. Multinomial Naive Bayes
2. Random Forest Classifier
3. Decision Tree Classifier
4. Support Vector Machines
5. Gradient Boosting Classifier
6. K-Nearest Neighbor Classifier

3.14.1 Multinomial Naive Bayes

The Multinomial Naive Bayes algorithm is based on a Bayesian learning approach popular in Natural Language Processing(NLP). This program guesses the tag of a text, such as an email or a newspaper story, using the

Bayes theorem. It calculates each tag's likelihood for each and every sample and outputs the tag with highest chance.

It is a strong tool used for analyzing text input and solving problems with numerous classes, as the Naive Bayes Algorithm is based on Bayes Theorem it is necessary to know about what Bayes Theorem is. The Bayes Theorem was developed by Thomas Bayes which estimates the likelihood of occurrence based on prior knowledge of the events conditions.

When the predictor B is itself available then we calculate the likelihood of class A, we use the formula:

$P(A|B) = P(A)*P(B|A)/P(B)$

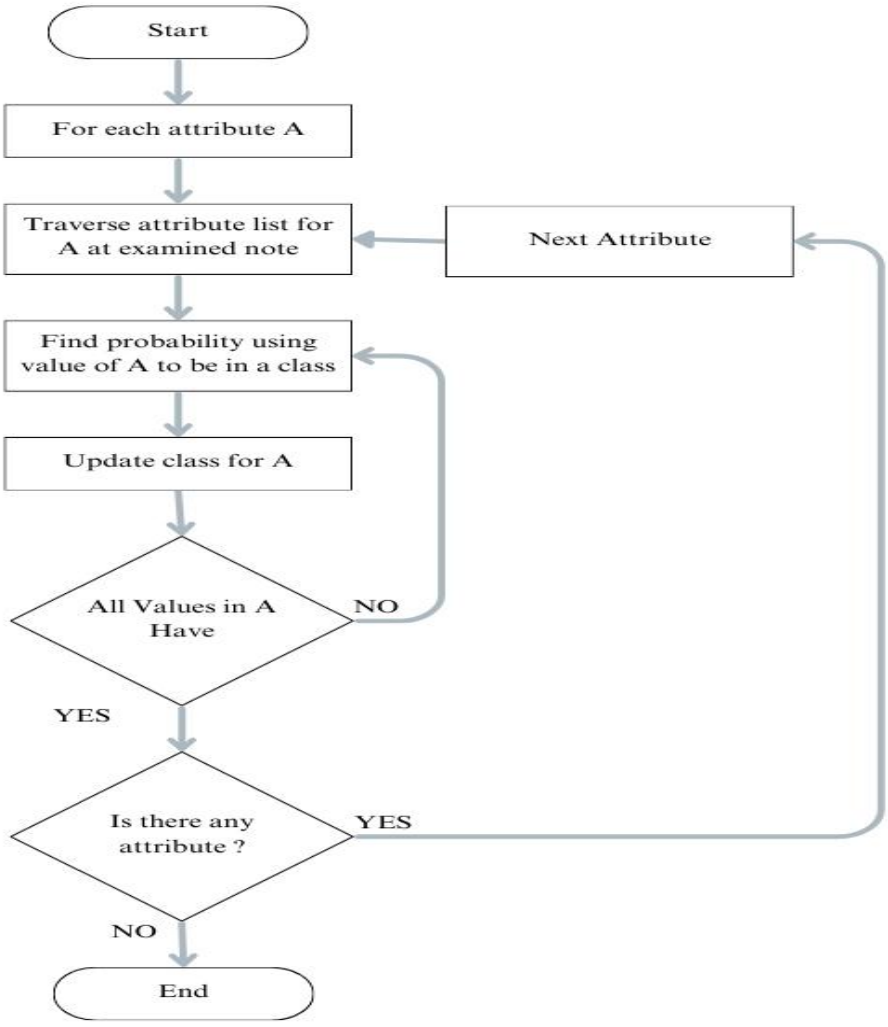


Fig 14. Flowchart of Multinomial Naive Bayes

Advantages:

- ✓ It is a simple algorithm to use because all we have to do is to calculate probability.
- ✓ This approach works for both continuous and discrete data.
- ✓ It's a straightforward algo. And can be used to forecast real-time applications

Disadvantages:

- ✧ The prediction accuracy is lower than other
- ✧ Non-appropriate for regression
- ✧ It can only be used for classifying textual input but no numeric values.

We are using Multinomial Naive Bayes over Gaussian because assumptions of Gaussian Naive Bayes aren't met with sparse data and a simple Gaussian fit over the data will not result in a good fit or prediction.

The code we used is as follows:

```
#Multinomial Naive Bayes
```

```
from sklearn.naive_bayes import MultinomialNB
model1 = MultinomialNB()
model1.fit(x_train,y_train)
pred = model1.predict(x_test)
print("Confusion Matrix for Multinomial Naive Bayes:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
```

Fig 15. Code for Multinomial Naive Bayes

3.14.2 Random Forest Classifier

Both classification and regression can be done using a supervised learning method called a random forest. On the other hand, its main use is in classification problems. Since trees are the main component of a forest, a forest with more trees is naturally stronger.

Similarly, the random forest method builds a decision tree from data samples, receives predictions from each side, and then votes to select the best answer. Averaging the results reduces over fitting, making this ensemble technique better than a single decision tree.

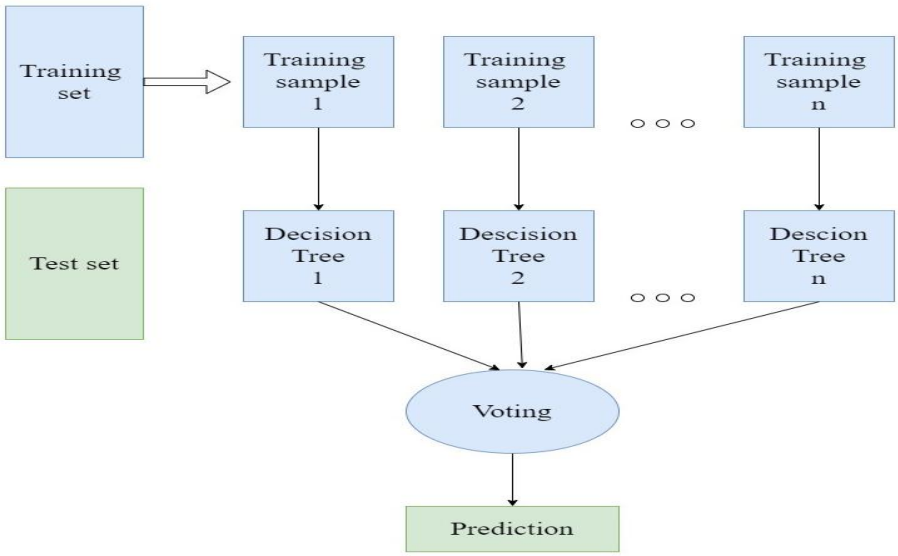


Fig 16. Flowchart of Random Forest Classifier

Advantages:

- ✓ Random Forest is capable of performing both Classification and Regression tasks.
- ✓ It is capable of handling large datasets with high dimensionality.
- ✓ It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages:

- ✧ It can be used for both classification and regression tasks, but are not suitable for regression tasks

We used the algo. and the code for the following is as shown below:

```
#Random Forest Classifier
```

```
from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier()
model2.fit(x_train,y_train)
pred = model2.predict(x_test)
print("Confusion Matrix for Random Forest Classifier:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
```

Fig 17. Code for Random Forest Classifier

3.14.3 Decision Tree Classifier

A decision tree is a supervised learning method that can be used to solve classification and regression problems, but is mostly used to solve

classification problems. It is a tree-structured classifier, where internal nodes represent the features of the dataset, branches represent the classification rules, and individual leaf nodes represent the result. Two nodes – a decision node and a leaf node – form a decision tree. A decision is made by a decision node that has multiple branches, while a leaf node represents the result of a decision and has no other branches.

The functions of the transferred data set are used to perform tests or make decisions. It is a graphical representation that shows all the possibilities of solving a problem or making a decision depending on certain parameters.

Because it starts from the root node and develops into a tree-like structure through other branches, it is called a decision tree.

To build the tree, we use the CART algorithm, which stands for Classification and Regression Tree Algorithm.
A decision tree simply asks a question and based on the answer (yes/no) further divides the tree into sub trees.

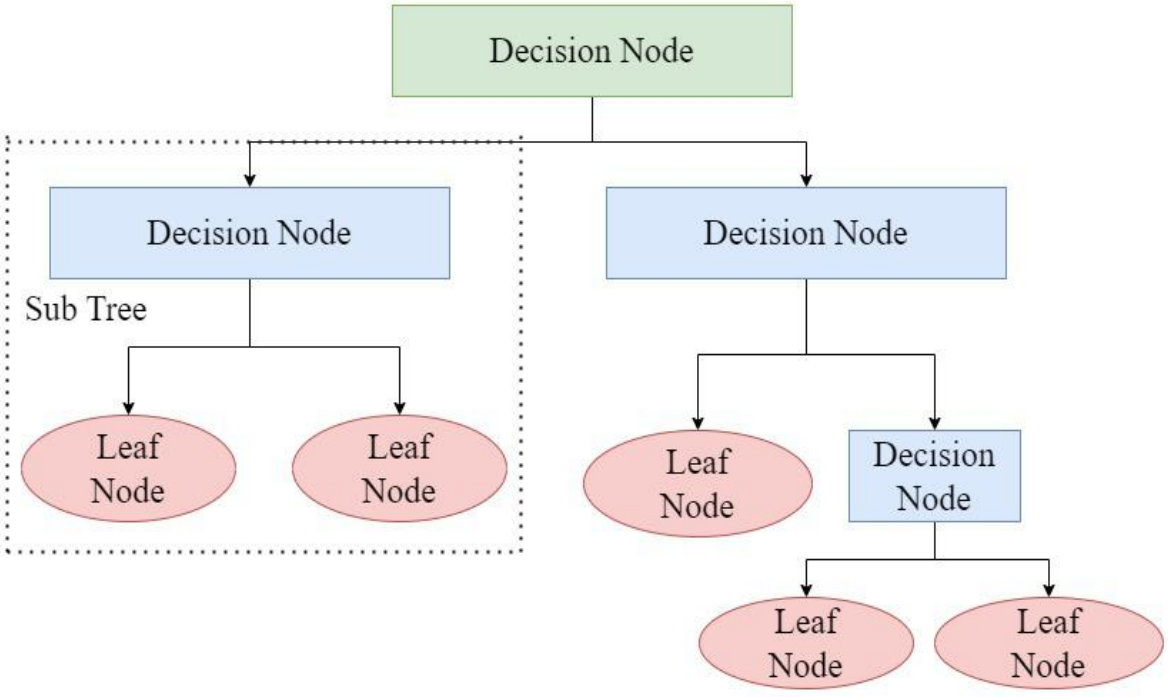


Fig 18. Flowchart of Decision Tree Classifier

Advantages:

- ✓ It is simple to understand as it is based on the process on how human makes day to day decisions.

- ✓ It is very useful for solving decision related problems.

- ✓ It helps us to think about all the possible outcomes of a problem.

- ✓ It requires less data cleaning than others algorithms.

Disadvantages:

- ✧ It is complex as it contains a lot of layers.

- ✧ An Overfitting issue can arise at any point, which can be resolved by using Random Forest Algorithm.

- ✧ The computational complexity of the algo may increase with an increase in labels.

The Code we used in our system is shown below:

```
#Decision Tree
```

```
from sklearn.tree import DecisionTreeClassifier
model3= DecisionTreeClassifier()
model3.fit(x_train,y_train)
pred = model3.predict(x_test)
print("Confusion Matrix for Decision Tree:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
```

Fig 19. Code for Decision Tree Classifier

3.14.4 Support Vector Machines

One of the most commonly used supervised learning methods, Support Vector Machine (SVM), is used to solve classification and regression problems. In machine learning, it is often used to solve classification problems.

The purpose of the SVM method is to create an optimal decision boundary or line that can divide the n-dimensional space into classes so that we can quickly classify new data points in the future. This optimal decision boundary is also called the hyperplane.

The hyperplane is created with SVM by selecting extreme points and vectors. The support vectors associated with these extreme cases are the name of an algorithm known as SVM.

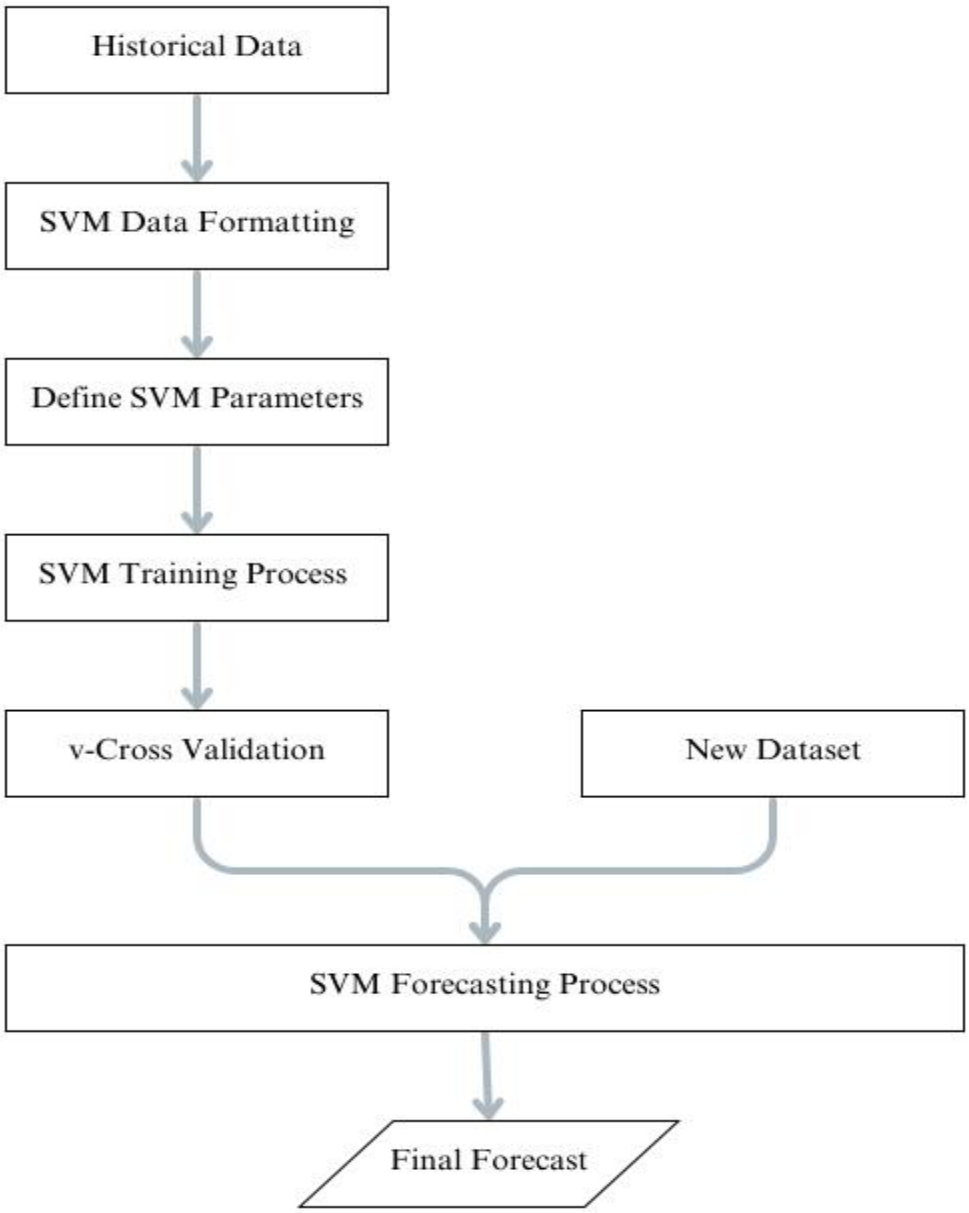


Fig 20. Flowchart of Support Vector Machine

Advantages:

- ✓ It is very effective in high-dimensional cases.
- ✓ It is memory efficient as it uses a subset of training points in the decision defined as support vectors.

- ✓ Different kernel functions can be used for decision functions and it is possible to specify custom kernel also.

The code we implemented is as follows:

```
#Support Vector Machines
```

```
from sklearn.svm import SVC
model4 = SVC(random_state=101)
model4.fit(x_train,y_train)
pred = model4.predict(x_test)
print("Confusion Matrix for Support Vector Machines:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
```

Fig 21. Code for Support Vector Machines

3.14.5 Gradient Boosting Classifier

Gradient Boosting Classifier is a popular machine learning algorithm that is used for classification tasks. It belongs to the family of boosting algorithms and is based on the principle of ensemble learning, where multiple weak learners are combined to form a strong learner.

In gradient boosting, decision trees are used as the weak learners. The algorithm works by sequentially adding decision trees to the ensemble, with each new tree attempting to correct the errors made by the previous ones. The process continues until the desired level of accuracy is achieved or a specified number of trees have been added.

The “gradient” in the name refers to the fact that the algorithm uses gradient descent optimization to minimize the loss function while training the model. The loss function measures the difference between the predicted and actual

values of the target variable, and the gradient descent algorithm updates the model parameters in the direction of steepest descent of the loss function to minimize it.

Overall, Gradient Boosting Classifier is a powerful and widely-used algorithm for classification tasks, known for its ability to handle complex datasets with high-dimensional feature spaces and achieve high accuracy.

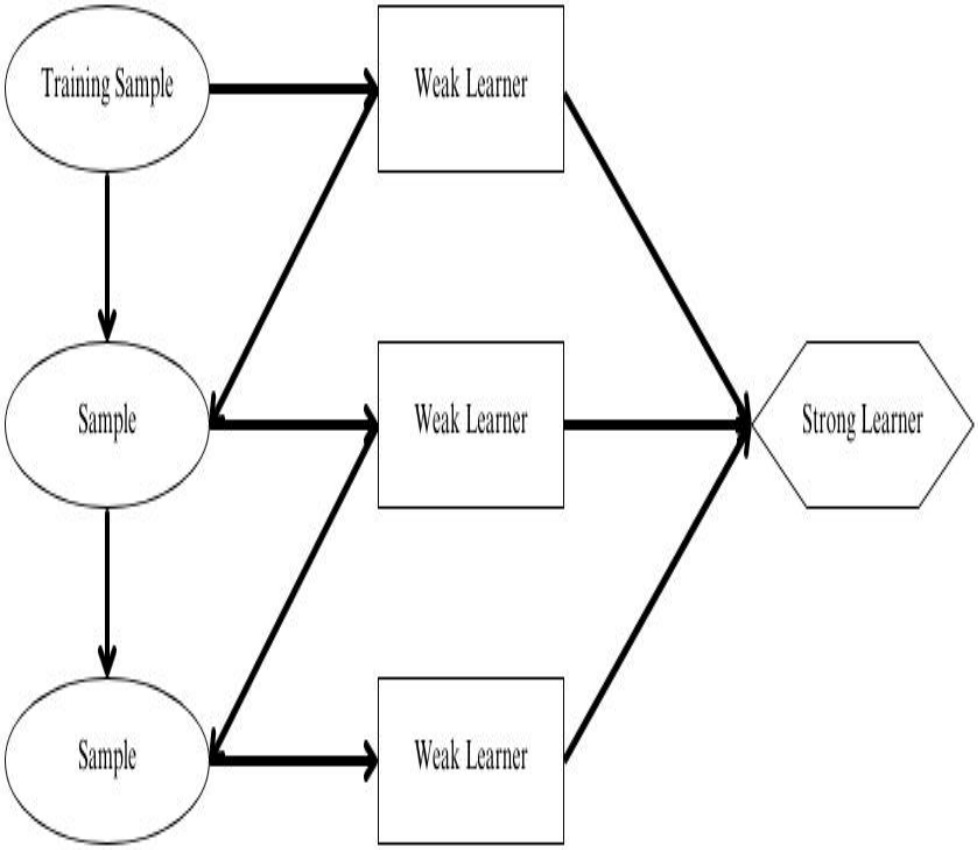


Fig 22. Flowchart of Gradient Boosting Classifier

Advantages:

- ✓ It is more accurate than other models.
- ✓ Trains faster on larger datasets.

✓ They can provide support handling categorical features

✓ Some of them can handle missing values natively.

Disadvantages

✧ These are prone to overfiting.

✧ In this the models can be computationally expensive i.e. can take long tim e to train.

✧ It is hard to interpret the final models.

The code we used is as follows

```
#Gradient Boosting Classifier
```

```
from sklearn.ensemble import GradientBoostingClassifier
model5 = GradientBoostingClassifier(learning_rate=0.1,max_depth=5,max_features=0.5,random_state=999999)
model5.fit(x_train,y_train)
pred = model5.predict(x_test)
print("Confusion Matrix for Gradient Boosting Classifier:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
```

Fig 23. Code for Gradient Boosting Classifier

3.14.6 K-Nearest Neighbor

KNN is an easy-to-use, supervised machine learning (ML) technique that is fre quently used in missing value imputation. It can be utilized for classification o r regression problems. The user can choose K to specify how many Neighbour ing observations will be used in the algorithm.

While comparing the large values of K with very small values of k large ones are more robust to the outliers and can provide us more solid boundaries of de cision. For example, K=4 is preferable than K=2, which may result in utcom e that are unfavourable in nature.

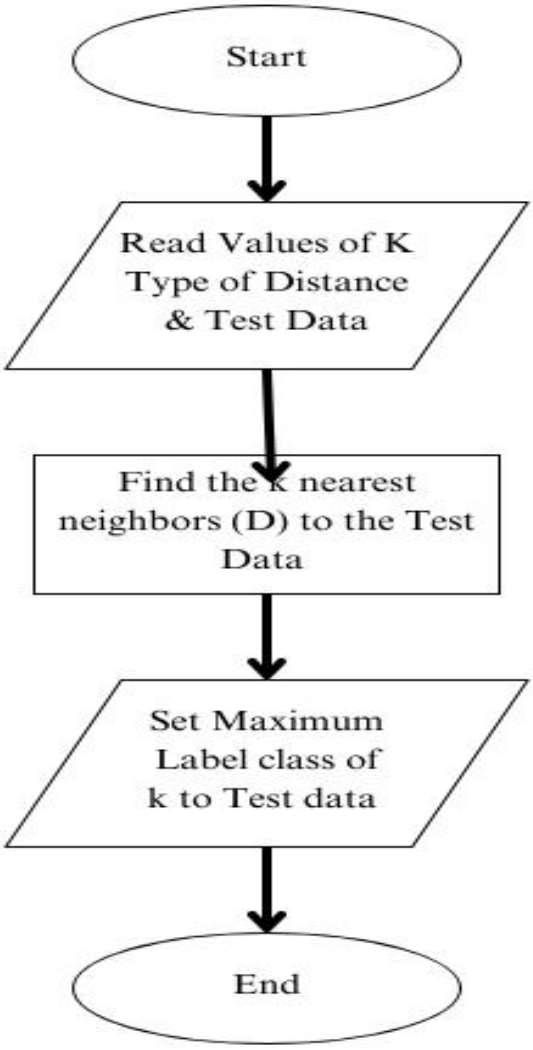


Fig 24. Flowchart of KNN Classifier

Advantages

- ✓ It does not require any training period as it stores the training dataset and l earns from it while making the predictions and also termed as Lazy Learn er.

- ✓ In this new data can be added seamlessly as it does not require training period.

- ✓ Very easy to implement.

Disadvantages

- ✧ Does not work well with large dataset.

- ✧ Unable to work with high dimensions.

- ✧ Needs feature scaling.

- ✧ Very sensitive to noisy/unbalanced data, missing values and outliers.

The code we used is shown below

```
#K - Nearest Neighbor Classifier
```

```
from sklearn.neighbors import KNeighborsClassifier
model6 = KNeighborsClassifier(n_neighbors=10)
model6.fit(x_train,y_train)
pred = model6.predict(x_test)
print("Confusion Matrix for K Neighbors Classifier:")
print(confusion_matrix(y_test,pred))
print("Score: ",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:")
print(classification_report(y_test,pred))
```

Fig 25. Code for KNN Classifier

3.15 Score Comparison Visualization

This was the step in which we compared all the models we used in our system by making a line graph using the pyplot function of matplotlib in python

In this we also printed the Scores of all the algorithms for which we used the following code with output:

```
#Score Comparision Visualization
```

```
scorenb=round(model1.score(x_test,y_test)*100,2)
scorerf=round(model2.score(x_test,y_test)*100,2)
scoredt=round(model3.score(x_test,y_test)*100,2)
scoresvm=round(model4.score(x_test,y_test)*100,2)
scorexgb=round(model5.score(x_test,y_test)*100,2)
scoreknn=round(model6.score(x_test,y_test)*100,2)
print("Scores are as follows:")
print("Multinomial Naive Bayes : "+str(scorenb))
print("Random Forest Classifier : "+str(scorerf))
print("Decision Tree : "+str(scoredt))
print("Support Vector Machines : "+str(scoresvm))
print("Gradient Boosting Classifier : "+str(scorexgb))
print("K Neighbors Classifier : "+str(scoreknn))
```

```
Scores are as follows:
Multinomial Naive Bayes : 76.94
Random Forest Classifier : 68.92
Decision Tree : 64.59
Support Vector Machines : 71.08
Gradient Boosting Classifier : 73.78
K Neighbors Classifier : 61.35
```

Fig 26. Comparing score of all algorithms

The code we used for plotting the line graph was as shown below

```
#creating graph to visualize better

import matplotlib.pyplot as plt

models=['NB','RF','DT','SVM','XGB','KNN']
accuracy=[scorenb,scorerf,scoredt,scoresvm,scorexgb,scoreknn]

plt.plot(models,accuracy,'b-o',label='Accuracy comparision plot');
plt.xlabel('Models')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

Fig 27. Code for Line Graph

3.16 Selecting the best algorithm and predicting Positive, Negative and Average review

In this step we will select the algorithm having the best accuracy and use it for predicting the Positive, Average and Negative review

For Positive review the code is

```
# POSITIVE REVIEW
pr = data['text'][0]
print(pr)
print("Actual Rating: ",data['stars'][0])
pr_t = vocab.transform([pr])
print("Predicted Rating:")
model1.predict(pr_t)[0]
```

Fig 28. Code for Positive Review

For Average review the code is

```
#Average Review
ar = data['text'][16]
print(ar)
print("Actual Rating: ",data['stars'][16])
ar_t = vocab.transform([ar])
print("Predicted Rating:")
model1.predict(ar_t)[0]
```

Fig 29. Code for Average Review

For Negative review the code is

```
#Negative Review
nr = data['text'][16]
print(nr)
print("Actual Rating: ",data['stars'][23])
nr_t = vocab.transform([nr])
print("Predicted Rating:")
model1.predict(nr_t)[0]
```

Fig 30. Code for Negative review

3.17 Classifying the review by no of stars

In this step we will be finding the no of reviews under each no of stars i.e. from 1 to 5

The code with output for the following is

```
count = data['stars'].value_counts()
print(count)
```

```
4    3526
5    3337
3    1461
2     927
1     749
Name: stars, dtype: int64
```

Fig 31. Classification w.r.t no of stars

CHAPTER 4
EXPERIMENTS & RESULT ANALYSIS

4.1 Discussion on the results we got

We applied all the algorithms on our system and checked the accur acies, then from them selected the best algo to predict the positive, average, and negative review

The models we used in our project were as follows

1. Multinomial Naive Bayes
2. Random Forest Classifier
3. Decision Tree Classifier
4. Support Vector Machines
5. Gradient Boosting Classifier
6. K-Nearest Neighbor Classifier

We calculated the following parameters of each algorithm

1. Accuracy/Score
2. Confusion Matrix

The Accuracy results were as follows

1. Multinomial Naïve Bayes

```
Confusion Matrix for Multinomial Naive Bayes:
[[ 75  49  38]
 [  7 180 105]
 [ 12  45 599]]
Score: 76.94
Classification Report:              precision    recall  f1-score   support

           1       0.80      0.46      0.59       162
           3       0.66      0.62      0.64       292
           5       0.81      0.91      0.86       656

    accuracy                           0.77      1110
   macro avg       0.75      0.66      0.69      1110
weighted avg       0.77      0.77      0.76      1110
```

Fig 32. Result of Multinomial Naïve Bayes

Here the Score is 76.94.

2. Random Forest Classifier

```
Confusion Matrix for Random Forest Classifier:
[[ 27  25 110]
 [  1  98 193]
 [  0  16 640]]
Score: 68.92
Classification Report:              precision    recall  f1-score   support

           1       0.96      0.17      0.28       162
           3       0.71      0.34      0.45       292
           5       0.68      0.98      0.80       656

    accuracy                           0.69      1110
   macro avg       0.78      0.49      0.51      1110
weighted avg       0.73      0.69      0.63      1110
```

Fig 33. Result of Random Forest Classifier

Here the Score is 68.92.

3. Decision Tree Classifier

```
Confusion Matrix for Decision Tree:
[[ 58  50  54]
 [ 32 138 122]
 [ 43  92 521]]
Score: 64.59
Classification Report:              precision    recall  f1-score   support

           1       0.44      0.36      0.39       162
           3       0.49      0.47      0.48       292
           5       0.75      0.79      0.77       656

    accuracy                           0.65      1110
   macro avg       0.56      0.54      0.55      1110
weighted avg       0.64      0.65      0.64      1110
```

Fig 34. Result of Decision Tree Classifier

Here the Score is 64.59.

4. Support Vector Machine

```
Confusion Matrix for Support Vector Machines:
[[ 31  23 108]
 [  5 122 165]
 [  1  19 636]]
Score: 71.08
Classification Report:              precision    recall  f1-score   support

           1       0.84      0.19      0.31       162
           3       0.74      0.42      0.54       292
           5       0.70      0.97      0.81       656

    accuracy                           0.71      1110
   macro avg       0.76      0.53      0.55      1110
weighted avg       0.73      0.71      0.67      1110
```

Fig 35. Result of Support Vector Machine

Here the Score is 71.08

5. Gradient Boosting Classifier

```
Confusion Matrix for Gradient Boosting Classifier:
[[ 61  33  68]
 [  9 139 144]
 [  4  33 619]]
Score: 73.78
Classification Report:              precision    recall  f1-score   support

           1       0.82      0.38      0.52       162
           3       0.68      0.48      0.56       292
           5       0.74      0.94      0.83       656

    accuracy                           0.74      1110
   macro avg       0.75      0.60      0.64      1110
weighted avg       0.74      0.74      0.71      1110
```

Fig 36. Result of Gradient Boosting Classifier

Here the Score is 73.78

6. K-Nearest Neighbor Classifier

```
Confusion Matrix for K Neighbors Classifier:
[[ 12  10 140]
 [  3  33 256]
 [  8  12 636]]
Score:  61.35
Classification Report:
              precision    recall  f1-score   support

           1       0.52      0.07      0.13       162
           3       0.60      0.11      0.19       292
           5       0.62      0.97      0.75       656

    accuracy                           0.61      1110
   macro avg       0.58      0.39      0.36      1110
weighted avg       0.60      0.61      0.51      1110
```

Fig 37. Result of K-Nearest Neighbor Classifier

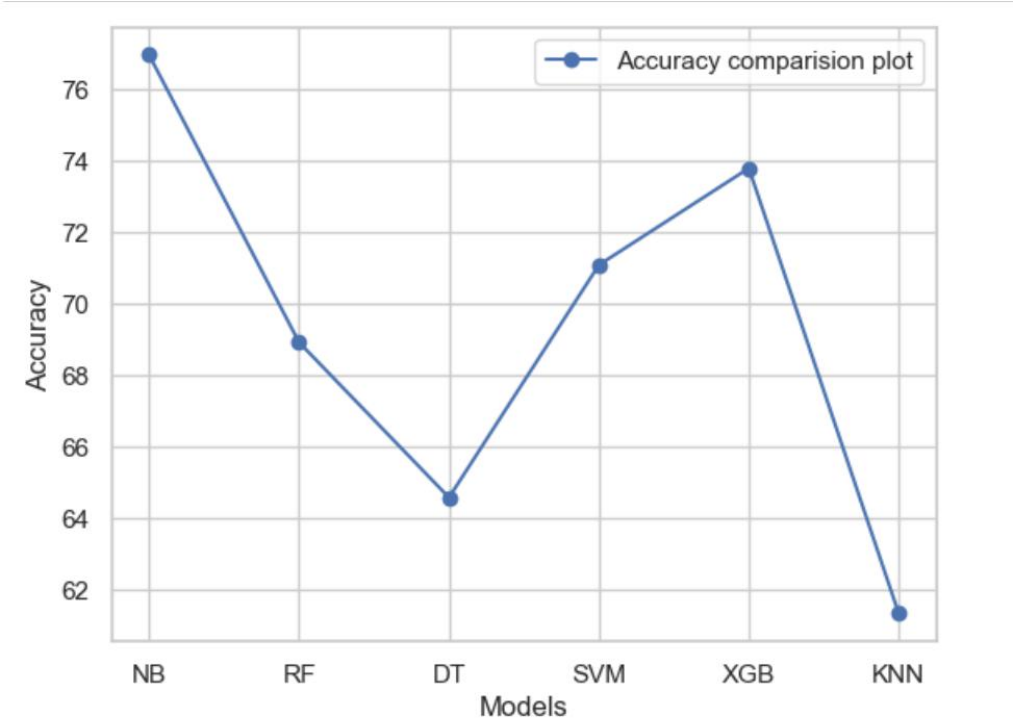
Here the Score is 61.35

So the results we got are shown in the table below

Sr. No.	Name of Algorithm	Accuracy/Score	Precision
1	Multinomial Naïve Bayes	76.94	0.75
2	Random Forest	68.92	0.78
3	Decision Tree	64.59	0.56
4	Support Vector Machine	71.08	0.76
5	Gradient Boosting	73.78	0.75
6	K-Nearest Neighbor	61.35	0.58

Table 3 Algorithms Accuracy and Precision

We also made a graph for visualizing our comparison of Algorithms



Graph 1 Accuracy Comparison Graph

4.2 Project Outcome

Best score: 76.94

Best Model: Multinomial Naive Bayes

Worst Model: K-Nearest Neighbor Classifier

As we concluded that Multinomial Naïve Bayes has the best Accuracy/Score so we will use it for predicting all the positive, average and negative reviews

The codes with output are as follows

For Positive

```
# POSITIVE REVIEW
pr = data['text'][0]
print(pr)
print("Actual Rating: ",data['stars'][0])
pr_t = vocab.transform([pr])
print("Predicted Rating:")
model1.predict(pr_t)[0]
```

```
My wife took me here on my birthday for breakfast and it was excellent.  The weather was perfect whic
h made sitting outside overlooking their grounds an absolute pleasure.  Our waitress was excellent an
d our food arrived quickly on the semi-busy Saturday morning.  It looked like the place fills up pret
ty quickly so the earlier you get here the better.

Do yourself a favor and get their Bloody Mary.  It was phenomenal and simply the best I've ever had.
I'm pretty sure they only use ingredients from their garden and blend them fresh when you order it.
It was amazing.

While EVERYTHING on the menu looks excellent, I had the white truffle scrambled eggs vegetable skille
t and it was tasty and delicious.  It came with 2 pieces of their griddled bread with was amazing and
it absolutely made the meal complete.  It was the best "toast" I've ever had.

Anyway, I can't wait to go back!
Actual Rating:  5
Predicted Rating:

5
```

Fig 38. Positive Review Prediction

For Average

```
#Average Review
ar = data['text'][16]
print(ar)
print("Actual Rating: ",data['stars'][16])
ar_t = vocab.transform([ar])
print("Predicted Rating:")
model1.predict(ar_t)[0]
```

```
We went here on a Saturday afternoon and this place was incredibly empty.  They had brunch specials going on, including $2 bloo
dy mary's and mimosas, but we were more in the mood for lunch.  Except for the bloody mary, I had to try one.  It came out in a
high-ball-sized glass.  Boo!  But it was really tasty. Yay!  The hubby remembered a sign outside the restaurant a few weeks bac
k that said they had Arrogant Bastard, and he got a 22 oz bottle for $4.75.  Hey, that's not fair!!

Next up: the wings.  We were a bit hesitant to order them when the waitress informed us that they are "seasoned" but not sauce
d, so they can't be ordered hot.  We did ask for them crispy though, and the waitress even asked the cooks to throw them back i
n for a few minutes when they came out not visibly crispy.  These non-traditional wings were actually pretty damn good.  The se
asoning was a little spicy and salty with just a hint of sweet.  If I were in the mood for the tang and kick of Frank's Hot Sau
ce, these wouldn't cut it, but otherwise they were good enough to go back again for.

My entree was the Tilapia salad, and I was a bit disappointed.  The fish was a bit dry and uninspired. And the greens underneat
h were overdressed and wilted.  I ate the greens around the fish and picked out the almonds and Mandarin oranges, but I had to
leave the mush hiding underneath the fish.

It wasn't bad enough to say I wouldn't go back, but I won't be anxiously awaiting my next trip.
Actual Rating:  3
Predicted Rating:

3
```

Fig 39. Average Reviews Prediction

For Negative

```
#Negative Review
nr = data['text'][16]
print(nr)
print("Actual Rating: ",data['stars'][23])
nr_t = vocab.transform([nr])
print("Predicted Rating:")
model1.predict(nr_t)[0]
```

We went here on a Saturday afternoon and this place was incredibly empty. They had brunch specials going on, including $2 bloo dy mary's and mimosas, but we were more in the mood for lunch. Except for the bloody mary, I had to try one. It came out in a high-ball-sized glass. Boo! But it was really tasty. Yay! The hubby remembered a sign outside the restaurant a few weeks bac k that said they had Arrogant Bastard, and he got a 22 oz bottle for $4.75. Hey, that's not fair!!

Next up: the wings. We were a bit hesitant to order them when the waitress informed us that they are "seasoned" but not sauce d, so they can't be ordered hot. We did ask for them crispy though, and the waitress even asked the cooks to throw them back i n for a few minutes when they came out not visibly crispy. These non-traditional wings were actually pretty damn good. The se asoning was a little spicy and salty with just a hint of sweet. If I were in the mood for the tang and kick of Frank's Hot Sau ce, these wouldn't cut it, but otherwise they were good enough to go back again for.

My entree was the Tilapia salad, and I was a bit disappointed. The fish was a bit dry and uninspired. And the greens underneat h were overdressed and wilted. I ate the greens around the fish and picked out the almonds and Mandarin oranges, but I had to leave the mush hiding underneath the fish.

It wasn't bad enough to say I wouldn't go back, but I won't be anxiously awaiting my next trip.
Actual Rating: 1
Predicted Rating:

3

Fig 40. Negative Reviews Prediction

CHAPTER 5
CONCLUSION

5.1 Conclusions

Firstly we took a dataset from kaggle named yelp.csv and it contained 10000 r eviews made by various people and it also contained the votes which were giv en by people like funny, useful and cool.

First of all we implemented all the required packages and libraries in python, t hen we loaded and analyzed the dataset in order to check the level of unstabilit y of the dataset

Then we visualized it and did the cleaning of dataset by removing the stop wor ds which is an important step as it results in slow computation time.

Then we performed the vectorization of the whole dataset.

Then we trained our dataset and performed all the algorithms and got the follo wing results:

Sr. No.	Name of Algorithm	Accuracy/Score	Precision
1	Multinomial Naïve Bayes	76.94	0.75
2	Random Forest	68.92	0.78
3	Decision Tree	64.59	0.56
4	Support Vector Machine	71.08	0.76
5	Gradient Boosting	73.78	0.75
6	K-Nearest Neighbor	61.35	0.58

Table 4 Final Results

At last we concluded that Multinomial Naïve Bayes was the best algorithm a mong all the algorithms we used and further we used to predict the reviews as Positive, Average and Negative.

We found the following values after prediction

Review Type	Actual Rating	Predicted Rating
Positive	5	5

Average	3	3
Negative	3	3

Table 5 Prediction Table

We found that our prediction was 100 % Accurate.

5.2 Applications of the project

The Fake Review Detection Project can have several applications in various in dustries. Here are a some examples:

1. In E-commerce: Online marketplaces like Amazon, eBay, and others c an use fake review detection tools to identify and remove fraudulent re views.

2. In Hospitality and Travel: Online travel agencies like TripAdvisor and Booking.com can use fake review detection tools to identify fake revie ws and ensure that the feedback provided by customers is genuine.

3. In Restaurants: Restaurants can use fake review detection tools to ident ify fake reviews on platforms like Yelp, Google, and Facebook.

4. In Healthcare: Online healthcare service providers can use fake review detection tools to identify fake reviews and ensure that patients receive accurate information about their services.

5. In Finance: Financial service providers can use fake review detection t ools to identify fake reviews and ensure that customers receive genuine feedback about their services.

Overall, fake review detection can help businesses build trust with customers, maintain their online reputation, and improve the overall quality of their servic es.

5.3 Limitations of the Project

1. We found an accuracy of 76.94 i.e. for Multinomial Naïve Bayes and our still searching an algo. with 100 % accuracy

5.4 Contributions

1. Devesh Kumar Singh

Found the Dataset, Classified the data, cleaned the unnecessary columns, train ed the dataset, implemented NB, RF, DT Algo's.

2. Kanishak Vyas

Implemented the SVM, XGB, KNN Algo's, Analysed the accuracy and precision, Discovered the final result.

5.5Future Scope

This project can help us in reducing the trend of fake reviews which are contri buting highly in manipulating the customers to make wrong decision while pur chasing anything online. So we have made this project and got a highest accur acy of 76.94 which is very low and we need to find different algo. which can g ive us a accuracy of absolute 100 %.

REFERENCES

[1] E. F. Cardoso, R. M. Silva and T. A. Almeida, "Towards automatic filtering of fake reviews", Neurocomputing, vol. 309, pp. 106-116, Oct. 2018.

[2] L. Da Xu, W. He and S. Li, "Internet of Things in industries: A survey", IEEE Trans. Ind. Informat., vol. 10, no. 4, pp. 2233-2243, Nov. 2014.

[3] Y. Ren and Y. Zhang, "Deceptive opinion spam detection using neural network", Proc. 26th Int. Conf. Comput. Linguistics: Tech. Papers (COLING), pp. 140-150, 2016.

[4] N. Jindal and B. Liu, "Opinion spam and analysis", Proc. Int. Conf. Web Search Web Data Mining (WSDM), pp. 219-230, 2008.

[5] A. Heydari, M. Tavakoli and N. Salim, "Detection of fake opinions using time series", Expert Syst. Appl., vol. 58, pp. 83-92, Oct. 2016.

[6] L. Li, W. Ren, B. Qin and T. Liu, "Learning document representation for deceptive opinion spam detection" in Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, Nanjing, China:Springer, pp. 393-404, 2015.

[7] H. Aghakhani, A. Machiry, S. Nilizadeh, C. Kruegel and G. Vigna, "Detecting deceptive reviews using generative adversarial networks", Proc. IEEE Secur. Privacy Workshops (SPW), pp. 89-95, May 2018.

[8] A. Mukherjee, V. Venkataraman, B. Liu and N. Glance, "Fake review detection: Classification and analysis of real and pseudo reviews", 2013.

[9] R. Yafeng, J. Donghong, Z. Hongbin and Y. Lan, "Deceptive reviews detection based on positive and unlabeled learning", J. Comput. Res. Develop., vol. 52, no. 3, pp. 639, 2015.

[10] R. Y. K. Lau, S. Y. Liao, R. C.-W. Kwok, K. Xu, Y. Xia and Y. Li, "Text mining and probabilistic language modeling for online review spam detection", ACM Trans. Manage. Inf. Syst., vol. 2, no. 4, pp. 1-30, Dec. 2011.

[11] A. Heydari, M. A. Tavakoli, N. Salim and Z. Heydari, "Detection of review spam: A survey", Expert Syst. Appl., vol. 42, no. 7, pp. 3634-3642, May 2015.

[12] Y. Lin, T. Zhu, X. Wang, J. Zhang and A. Zhou, "Towards online review spam detection", Proc. 23rd Int. Conf. World Wide Web (WWW Companion), pp. 341-342, 2014.

[13] J. Li, P. Lv, W. Xiao, L. Yang and P. Zhang, "Exploring groups of opinion spam using sentiment analysis guided by nominated topics", Expert Syst. Appl., vol. 171, Jun. 2021.

[14] Z. You, T. Qian and B. Liu, "An attribute enhanced domain adaptive model for cold-start spam review detection", Proc. 27th Int. Conf. Comput. Linguistics, pp. 1884-1895, 2018.

[15] Z. Sedighi, H. Ebrahimpour-Komleh and A. Bagheri, "RLOSD: Representation learning based opinion spam detection", Proc. 3rd Iranian Conf. Intell. Syst. Signal Process. (ICSPIS), pp. 74-80, Dec. 2017.

[16] S. Zhao, Z. Xu, L. Liu and M. Guo, "Towards accurate deceptive opinion spam detection based on word order-preserving CNN", arXiv:1711.09181, 2017, [online] Available: http://arxiv.org/abs/1711.09181.

[17] S. Noekhah, N. B. Salim and N. H. Zakaria, "Opinion spam detection: Using multi-iterative graph-based model", Inf. Process. Manage., vol. 57, no. 1, Jan. 2020.

[18] F. Khurshid, Y. Zhu, Z. Xu, M. Ahmad and M. Ahmad, "Enactment of ensemble learning for review spam detection on selected features", Int. J. Comput. Intell. Syst., vol. 12, no. 1, pp. 387-394, 2018.

[19] L.-Y. Dong, S.-J. Ji, C.-J. Zhang, Q. Zhang, D. W. Chiu, L.-Q. Qiu, et al., "An unsupervised topic-sentiment joint probabilistic model for detecting deceptive reviews", Expert Syst. Appl., vol. 114, pp. 210-223, Dec. 2018.

[20] E. Fitzpatrick, J. Bachenko and T. Fornaciari, "Automatic detection of verbal deception", Synth. Lectures Hum. Lang. Technol., vol. 8, no. 3, pp. 1-119, Sep. 2015.

(1) Major Codes of the file:

First we imported the Important liabraries

The code is:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.corpus import stopwords
import string
import math
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix, accuracy_score, roc_auc_score, roc_curve
from sklearn.model_selection import GridSearchCV
%matplotlib inline
```

Then Loaded the dataset and analyzed it the code for that is:

```
data = pd.read_csv('yelp.csv')
print("Shape of the dataset:")
print(data.shape)
print("Column names:")
print(data.columns)
print("Datatype of each column:")
print(data.dtypes)
print("Few dataset entries:")
```

```
print(data.head())
data.describe(include='all')
```

Then we created a new column for length to make applying of algorithms a bit easier the code is:

```
data['length'] = data['text'].apply(len)
data.head()
```

Then we did the visualization by making a graph and the code is:

```
graph = sns.FacetGrid(data=data,col='stars')
graph.map(plt.hist,'length',bins=50,color='blue')
```

Then found the mean value of the columns by using the code:

```
stval = data.groupby('stars').mean()
stval
```

Then found the correlation between the vote columns i.e. Cool,Funny,Useful

The code is:

```
stval.corr()
```

Then we performed the classification and split it in reviews and stars, the code for the same is:

```
data_classes = data[(data['stars']==1) | (data['stars']==3) | (data['stars']==5)]
data_classes.head()
print(data_classes.shape)

x = data_classes['text']
y = data_classes['stars']
print(x.head())
print(y.head())
```

Then we performed the data cleaning which is a necessary step before performing the algorithms the code is:

```
def text_process(text):
    nopunc = [char for char in text if char not in string.punctuation]
    nopunc = ''.join(nopunc)
    return [word for word in nopunc.split() if word.lower() not in stopwords.words('english')]
```

Then we performed the vectorization on some reviews the code is:

```
vocab = CountVectorizer(analyzer=text_process).fit(x)
print(len(vocab.vocabulary_))
r0 = x[0]
print(r0)
vocab0 = vocab.transform([r0])
print(vocab0)
print("Getting the words back:")
print(vocab.get_feature_names_out()[19648])
print(vocab.get_feature_names_out()[10643])
```

Then we did the vectorization of whole dataset and also checked the sparse matrix the code is:

```
x = vocab.transform(x)
print("Shape of the sparse matrix: ", x.shape)
print("Non-Zero occurences: ",x.nnz)

density = (x.nnz/(x.shape[0]*x.shape[1]))*100
print("Density of the matrix = ",density)
```

Then we did the splitting and training of the dataset and the code is:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=101)
```

Then did the modeling and performed the algorithms the code is:

```
#Multinomial Naïve Bayes
from sklearn.naive_bayes import MultinomialNB
model1 = MultinomialNB()
model1.fit(x_train,y_train)
pred = model1.predict(x_test)
print("Confusion Matrix for Multinomial Naive Bayes:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
#Random Forest
from sklearn.ensemble import RandomForestClassifier
model2 = RandomForestClassifier()
model2.fit(x_train,y_train)
pred = model2.predict(x_test)
print("Confusion Matrix for Random Forest Classifier:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
#Decision Tree
from sklearn.tree import DecisionTreeClassifier
model3= DecisionTreeClassifier()
model3.fit(x_train,y_train)
pred = model3.predict(x_test)
print("Confusion Matrix for Decision Tree:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
#Support Vector Machines
from sklearn.svm import SVC
```

```
model4 = SVC(random_state=101)
model4.fit(x_train,y_train)
pred = model4.predict(x_test)
print("Confusion Matrix for Support Vector Machines:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
#Gradient Boosting
from sklearn.ensemble import GradientBoostingClassifier
model5=GradientBoostingClassifier(learning_rate=0.1,max_depth=5,max_features=0.5,random_state=999999)
model5.fit(x_train,y_train)
pred = model5.predict(x_test)
print("Confusion Matrix for Gradient Boosting Classifier:")
print(confusion_matrix(y_test,pred))
print("Score:",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:",classification_report(y_test,pred))
#K-Nearest Neighbor
from sklearn.neighbors import KNeighborsClassifier
model6 = KNeighborsClassifier(n_neighbors=10)
model6.fit(x_train,y_train)
pred = model6.predict(x_test)
print("Confusion Matrix for K Neighbors Classifier:")
print(confusion_matrix(y_test,pred))
print("Score: ",round(accuracy_score(y_test,pred)*100,2))
print("Classification Report:")
print(classification_report(y_test,pred))
```

Then we did the score comparison visualization to check the best algorithms among the algorithms we used.

The code is:

```
scorenb=round(model1.score(x_test,y_test)*100,2)
scorerf=round(model2.score(x_test,y_test)*100,2)
scoredt=round(model3.score(x_test,y_test)*100,2)
scoresvm=round(model4.score(x_test,y_test)*100,2)
scorexgb=round(model5.score(x_test,y_test)*100,2)
scoreknn=round(model6.score(x_test,y_test)*100,2)
print("Scores are as follows:")
print("Multinomial Naive Bayes : "+str(scorenb))
print("Random Forest Classifier : "+str(scorerf))
print("Decision Tree : "+str(scoredt))
print("Support Vector Machines : "+str(scoresvm))
print("Gradient Boosting Classifier : "+str(scorexgb))
print("K Neighbors Classifier : "+str(scoreknn))
```

Then we created a graph for a better visualization of our results the code of the same is:

```
import matplotlib.pyplot as plt
models=['NB','RF','DT','SVM','XGB','KNN']
accuracy=[scorenb,scorerf,scoredt,scoresvm,scorexgb,scoreknn]
plt.plot(models,accuracy,'b-o',label='Accuracy comparision plot');
plt.xlabel('Models')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```

Then we used the best algorithms to predict our reviews as Positive, Average and Negative the code for all three is shown below:

For Positive

```
# POSITIVE REVIEW
```

```
pr = data['text'][0]
print(pr)
print("Actual Rating: ",data['stars'][0])
pr_t = vocab.transform([pr])
print("Predicted Rating:")
model1.predict(pr_t)[0]
```

For Average:

```
#Average Review
ar = data['text'][16]
print(ar)
print("Actual Rating: ",data['stars'][16])
ar_t = vocab.transform([ar])
print("Predicted Rating:")
model1.predict(ar_t)[0]
```

For Negative:

```
#Negative Review
nr = data['text'][16]
print(nr)
print("Actual Rating: ",data['stars'][23])
nr_t = vocab.transform([nr])
print("Predicted Rating:")
model1.predict(nr_t)[0]
```

Then at last we used a code to show the distribution of reviews on the stars they got the code is:

```
count = data['stars'].value_counts()
print(count)
```