-

MINI PROJECT

(2020-21)

“Consumer Sentimental Analysis ”

Project Report

Institute of Engineering & Technology

Submitted By -

Atul kumar Sharma (191500178)

Ashutosh Pachouri (191500170)

Bishesh Babu (191500222)

Jatin Singh (191500365)

Under the Supervision Of

Md. Farmanul Haque

Technical Trainer

Department of Computer Engineering & Applications

-

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road, Chaumuha, Mathura – 281406 U.P (India)

Declaration

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project **"Consumer Sentimental Analysis"**, in partial fulfillment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Md. Farmanul Haque, Technical Trainer, Dept. of CEA,GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign: *Atul Kumar Sharma*

Name of Candidate: Atul kumar sharma

University Roll No.:191500178

Sign: *Ashutosh Pachouri*

Name of Candidate: Ashutosh Pachouri

University Roll No.:191500170

Sign: *Bishesh Babu*

Name of Candidate: *Bishesh Babu*

University Roll No.:191500222

Sign: *Jatin Singh*

Name of Candidate: Jatin Singh

University Roll No.:191500365

-

-

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Certificate

This is to certify that the project entitled “Consumer Sentimental Analysis”, carried out in Mini Project – I Lab, is a bonafide work by Atul kumar Sharma, Ashutosh pachouri, Bishesh babu, Jatin singh and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

Signature of Supervisor:

Name of Supervisor: Md. Farmanul Haque

Date:

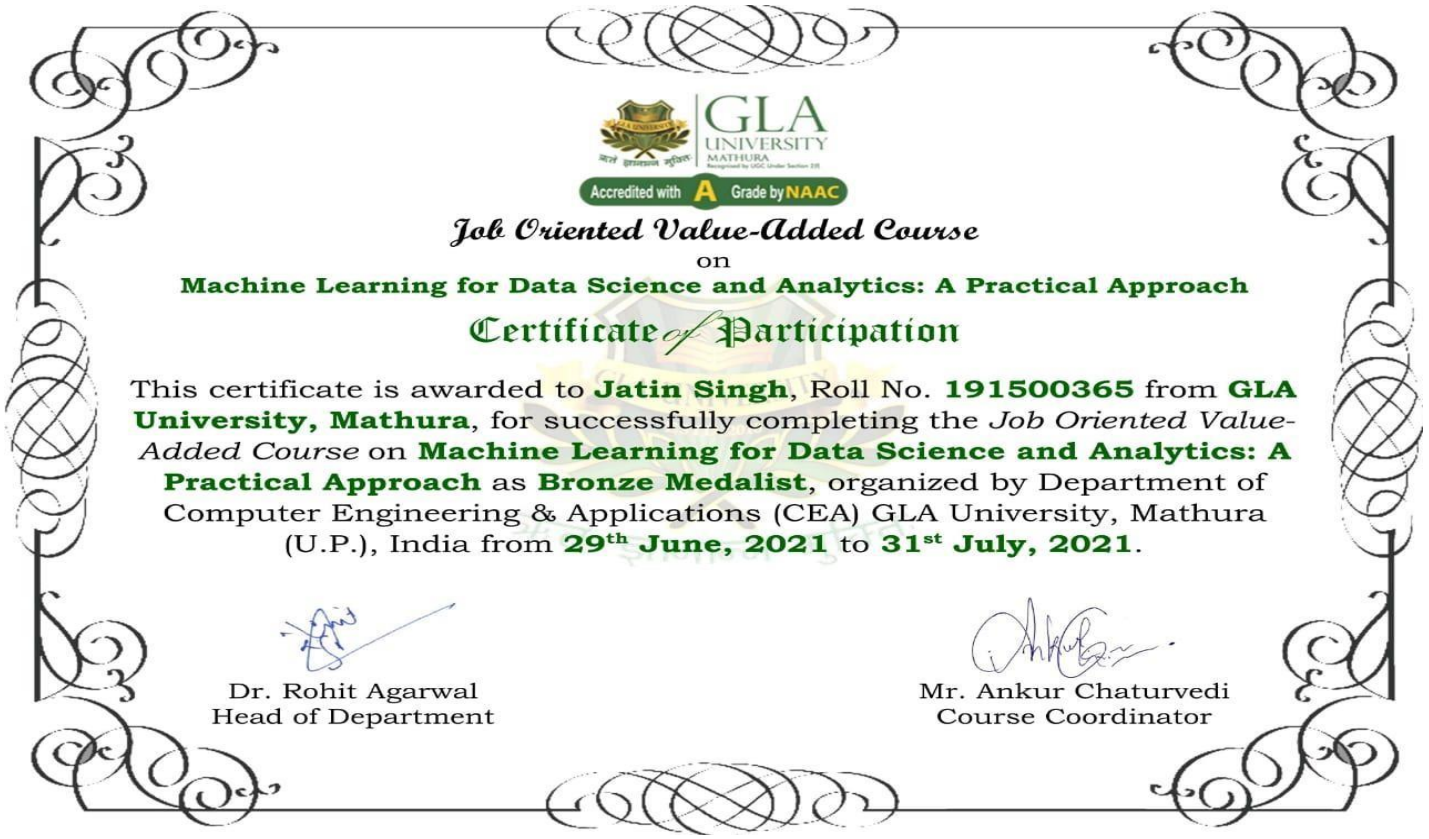
-

Training Certificates

- **Ashutosh Pachori**

-

- **Jatin Singh**



GLA UNIVERSITY MATHURA

Accredited with A Grade by NAAC

Job Oriented Value-Added Course

on

Machine Learning for Data Science and Analytics: A Practical Approach

Certificate of Participation

This certificate is awarded to **Jatin Singh**, Roll No. **191500365** from **GLA University, Mathura**, for successfully completing the *Job Oriented Value-Added Course* on **Machine Learning for Data Science and Analytics: A Practical Approach** as **Bronze Medalist**, organized by Department of Computer Engineering & Applications (CEA) GLA University, Mathura (U.P.), India from **29th June, 2021** to **31st July, 2021**.

Dr. Rohit Agarwal
Head of Department

Mr. Ankur Chaturvedi
Course Coordinator

-

- **Atul Kumar Sharma**

-

- **Bihesh Babu**

GLA UNIVERSITY MATHURA

Accredited with A Grade by NAAC

Job Oriented Value-Added Course

on

Machine Learning for Data Science and Analytics: A Practical Approach

Certificate of Participation

This certificate is awarded to **Bishesh Babu**, Roll No. **191500222** from **GLA University, Mathura**, for successfully completing the *Job Oriented Value-Added Course* on **Machine Learning for Data Science and Analytics: A Practical Approach** as **Gold Medalist**, organized by Department of Computer Engineering & Applications (CEA) GLA University, Mathura (U.P.), India from **29th June, 2021** to **31st July, 2021**.

Dr. Rohit Agarwal
Head of Department

Mr. Ankur Chaturvedi
Course Coordinator

-

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor MD Farmanul Haque, our technical trainer and supervisor.

He has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped me find resources during the training.

Thanking You

Sign: *Atul kumar sharma*

Name of Candidate: Atul kumar sharma

University Roll No.:191500178

Sign: *Ashutosh Pachouri*

Name of Candidate: Ashutosh pachouri

University Roll No.:191500170

Sign: *Bishesh Babu*

Name of Candidate: Bishesh Babu

University Roll No.:191500222

Sign: *Jatin Singh*

Name of Candidate: Jatin Singh

University Roll No.:191500365

-

ABSTRACT

Traditional approaches for studying consumer behavior, such as marketing survey and focus group, require a large amount of time and resources. Moreover, some products, such as smartphones, have a short product life cycle. As an alternative solution, we propose a system, the Micro-blog Sentiment Analysis System (MSAS), based on sentiment analysis to automatically analyze customer opinions from the Twitter micro-blog service. The MSAS consists of five main functions to collect Twitter posts, filter for opinionated posts, detect polarity in each post, categorize product features and summarize and visualize the overall results. We used the product domain of smartphone as our case study. The experiments on 100,000 collected posts related to smartphones showed that the system could help indicating the customers' sentiments towards the product features, such as Application, Screen, and Camera. Further evaluation by experts in smartphone industry confirmed that the system yielded some valid results.

-

CONTENTS

-

-

LIST OF FIGURES

-

-

LIST OF TABLES

-

CHAPTER-1

INTRODUCTION

1.1 CONTEXT

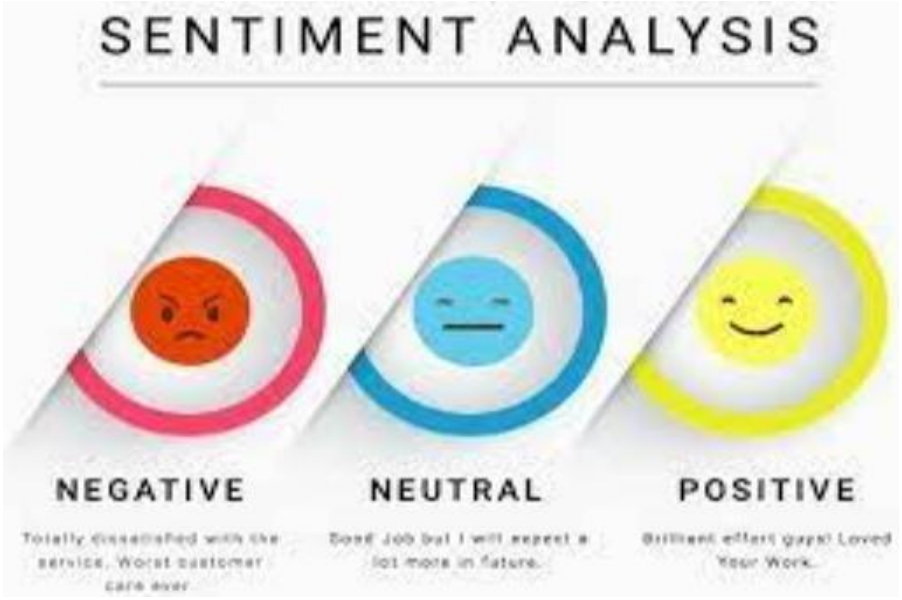
This Machine learning Application "Consumer Sentimental Analysis" has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at GLA University, Mathura supervised by Md. Farmanul Haque. This project has been completed approximately three months and has been executed in modules, meetings have been organised to check the progress of the work and for instructions and guidelines.

Sentiment analysis refers to identifying as well as classifying the sentiments that are expressed in the text source. Tweets are often useful in generating a vast amount of sentiment data upon analysis. These data are useful in understanding the opinion of the people about a variety of topics.

Therefore we need to develop an **Automated Machine Learning Sentiment Analysis Model** in order to compute the customer perception. Due to the presence of non-useful characters (collectively termed as the noise) along with useful data, it becomes difficult to implement models on them.

In this article, we aim to analyze the sentiment of the tweets provided from the **Sentiment140 dataset** by developing a machine learning pipeline involving the use of three classifiers (**Logistic Regression, Bernoulli Naive Bayes, and SVM**)along with using **Term Frequency- Inverse Document Frequency** (**TF-IDF)**. The performance of these classifiers is then evaluated using **accuracy** and **F1 Scores**.

-


SENTIMENT ANALYSIS
NEGATIVE
Totally dissatisfied with the service. Worst customer care ever
NEUTRAL
Good Job but I will expect a lot more in future.
POSITIVE
Brilliant effort guys! Loved Your Work.

.

-

CHAPTER -2

SOFTWARE REQUIREMENT ANALYSIS

Requirement Analysis System requirement is the ability of the system to meet the condition desired by the users. System requirement analysis is performed by grouping the needs into functional requirements and nonfunctional requirements. Functional Requirements The functional requirements of the Sentiment Analysis System for Moview Review are as follows:

1. The system can manage (add, read, update, and delete) movie review datasets,

2. The system can manage (add, read, update, and delete) the dictionary of root words in Bahasa Indonesia,

3. The system can manage (add, read, update, and delete) the dictionary of stop words,

4. The system can perform the classifier training process and display the model in the form of feature sets of the term data from the training data,

5. The system can display the test data result and display confusion matrix generated from the classifier testing,

6. The system can display a set of movie review dataset terms derived from tokenizing, filtering, and stemming processes, and

7. The system can display sentiment analysis result derived from reviews submitted by users.

Non-Functional Requirements

Meanwhile, the non-functional requirements of the Sentiment Analysis System for Movie Review are as follows:

1. The system uses an authentication through login process in order to differentiate user level.

2. The system can run in various web browsers which support the system environment,

3. The system gives a fast response, and

-

4. The system has a user-friendly interface design.

System Design The design of the Sentiment Analysis System for Movie Review uses Unified-Modeling Language (UML) as the modeling language and object-oriented programming concept. The documentation of the created system includes:

Business Process Business process diagram describes a series of inputs, needs, and outputs processed by the system in order to produce the desired destination by the user. Description of business process system can be seen in

Figure :- 3

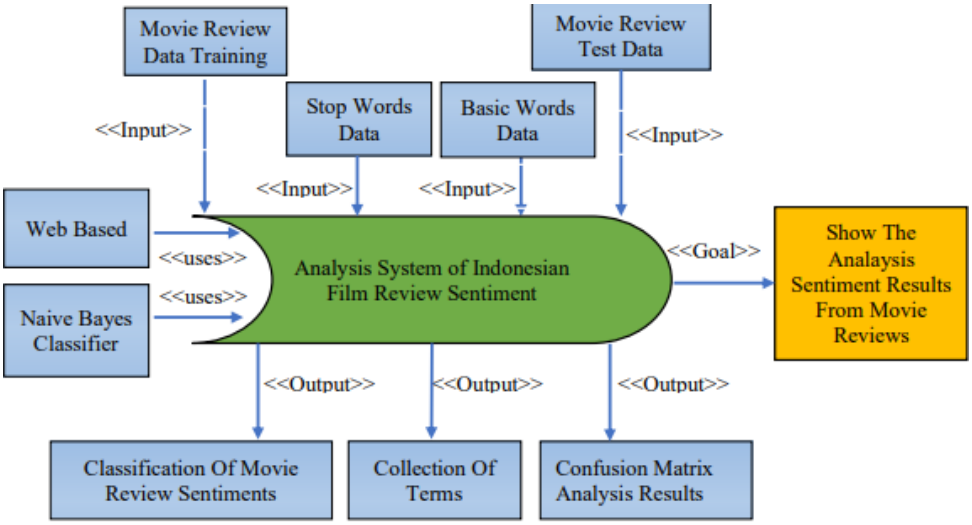


Figure 3. Bussiness Process Movie Sentiment Analysis System Review

-

2.3 HARDWARE AND SOFTWARE REQUIREMENTS

Software Specification:

- Technology Implemented :Machine learning , IOT
- Language Used :Python
- Database :Mongo DB
- User Interface Design : Graphical user interface
- Web Browser : Chrome , Firefox, Microsoft Edge

Hardware Requirements:

- Processor :Core i3/ i5/i7 processor
- Operating System :UNIX/LINUX/WINDOWS Operating System
- RAM :At least 4 GB RAM
- Hardware Devices :Mac OS, window , Linux
- Hard disk :At least 60 GB of Usable Hard Disk Space

-

CHAPTER- 3

SOFTWARE DESIGN

3.1 USE-CASE DIAGRAM:

Use case diagram describes the system's functional needs in the form of diagram. Use case is needed to describe typical interactions between users and system. Use case diagram of the Sentiment Analysis System for Movie Review is illustrated in

Figure 4.

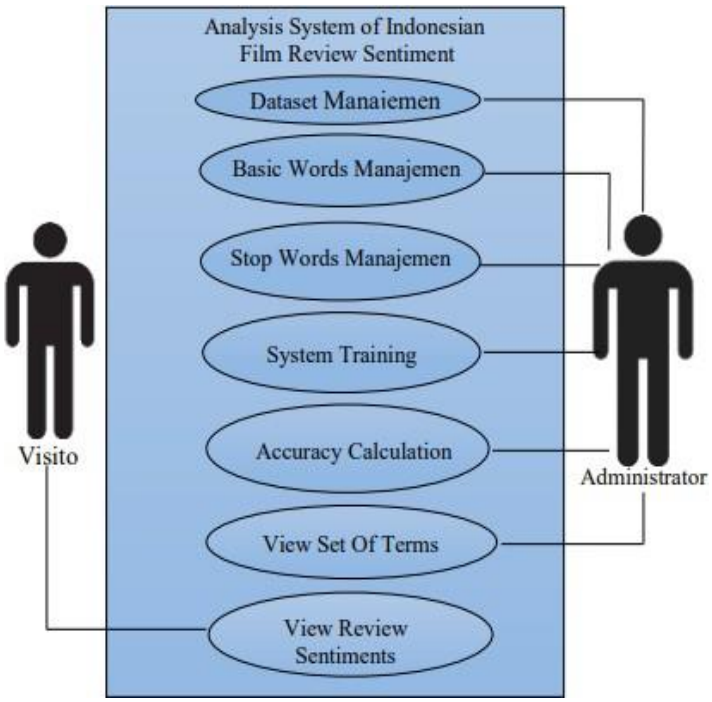


Figure 4. Use Case Diagram Sentiment Analysis Movie Review System

-

Deployment Diagram :-

Deployment diagram is a diagram illustrating the physical layout of the system. This diagram is included in order to describe the hardware and software required for the system to run according to the system needs. Deployment diagram of the Sentiment Analysis System for Movie Review can be seen in
Figure 5.

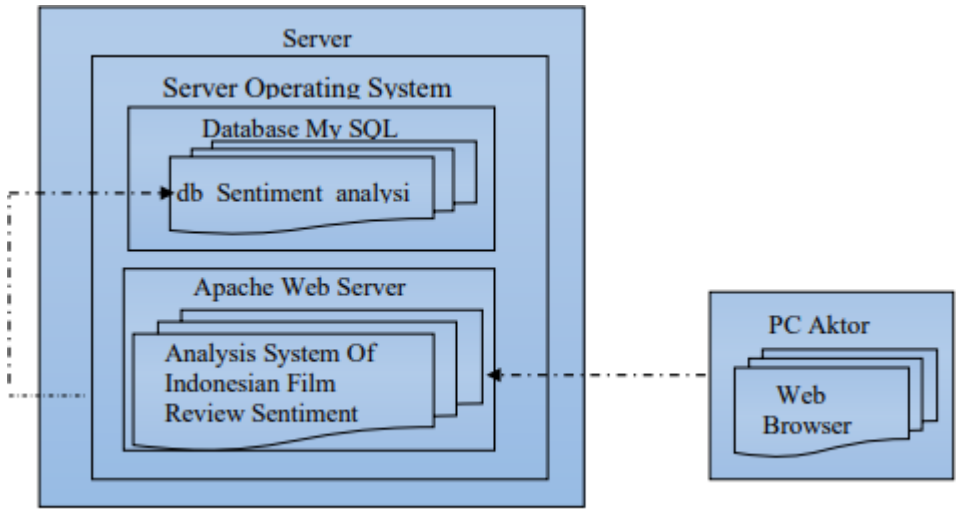


Figure 5. Deployment diagram of the Sentiment Analysis System for Movie

Sentiment Analysis System for Movie Review is developed as a web-based application. The webpage of Sentiment Analysis System for Movie Review is divided into two levels based on the user level: private page for the administrators and public page for the visitors. The private page contains the management dataset menu of the movie review, the stop words management menu, the training system menu, the accuracy calculation menu, and menu containing the set of terms from each movie review on the dataset. The public page contains an interface for visitors wanting to obtain sentiment analysis result from the movie reviews within the system. The main features displayed in the discussion are the dashboard page, system training page, system accuracy calculation page for the visitors.

System Interface Implementation` The dashboard page is the first accessed page once the administrator has successfull gged in to the system. The dashboard contains brief information on datasets, such as the number of review for each category, the ratio of training and test data as well as the ratio of positive and negative training data. The information is displayed in the form of numbers and graphs of donut charts. System training page contains a table on the occurrences and likelihood values of each term in all train data. The process of system training in order to form feature sets is performed by the administrators through the interface page by selecting

-

the button of “Latih”. The accuracy calculation page contains a list of reviews which become test data within the dataset, probability of the positive sentiment and negative sentiment for each test data, the result of the class analysis with the highest probability, and the accuracy value of the analysis result towards the original sentiment of the reviews. In addition to the review table of the test data, there are accuracy values of the system’s last accuracy test result which have been converted into percentage format and confusion matrix table of the last accuracy test result. Accuracy calculation process is also initiated by the administrators through the interface page by selecting the “Hitung (calculate)” button. The visitor page is the page which is for the visitors. The visitors can access this page in order to enter the review data they want to know the sentiment of. After selecting the “Lihat Sentiment (view sentiment)” button, the system classifies using the Naive Bayes method in order to obtain positive and negative sentiment probabilities as well as the sentiment result obtained based on the probability ratio of the two sentiment classes.

Accuracy Calculation Result The testing process is conducted five times. The first test was conducted by selecting 600 reviews as the training data and 200 reviews as the random test data. Further tests were performed by adding 100 movie reviews as the test data and test data of 20 movie reviews into the dataset utilized for the previous test. Details of the composition of each training and test data for each test process can be seen .

-

CHAPTER-4
TECHNOLOGY USED

4.1 Machine Learning

Machine learning is important because it gives enterprises a view of trends in customer behavior and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies.

- Supervised Learning :- in this type of machine learning, Data scientists supply algorithm with labeled training data.
- Unsupervised Learning :- In this type of machine learning, Data scientists supply algorithm
- Semi- Supervised Learning :- In this type of machine Learning. Data scientist supply algorithm
- Reinforcement Learning :- In this type of Machine Learning . Data scientist supply algorithm

-

4.1 VERSION OF PYTHON

Here is the table of list of versions of python .

Figure-5: Python

Release version	Release date
Python 2.7.17	Oct. 19, 2019
Python 3.7.5	Oct. 15, 2019
Python 3.8.0	Oct. 14, 2019
Python 3.7.4	July 8, 2019
Python 3.6.9	July 2, 2019
Python 3.7.3	March 25, 2019
Python 3.4.10	March 18, 2019

Table -1: Versions of Python

-

-

Chapter -5

IMPEMENTATION AND TESTING

Sentiment analysis refers to identifying as well as classifying the sentiments that are expressed in the text source. Tweets are often useful in generating a vast amount of sentiment data upon analysis. These data are useful in understanding the opinion of the people about a variety of topics.

Therefore we need to develop an **Automated Machine Learning Sentiment Analysis Model** in order to compute the customer perception. Due to the presence of non-useful characters (collectively termed as the noise) along with useful data, it becomes difficult to implement models on them.

In this article, we aim to analyze the sentiment of the tweets provided from the **Sentiment140 dataset** by developing a machine learning pipeline involving the use of three classifiers (**Logistic Regression, Bernoulli Naive Bayes, and SVM**)along with using **Term Frequency- Inverse Document Frequency** (**TF-IDF)**. The performance of these classifiers is then evaluated using **accuracy** and **F1 Scores**.

-

Image Source: Google Images

Problem Statement

In this project, we try to implement a **Twitter sentiment analysis model** that helps to overcome the challenges of identifying the sentiments of the tweets. The necessary details regarding the dataset are:

The dataset provided is the **Sentiment140 Dataset** which consists of **1,600,000 tweets** that have been extracted using the Twitter API. The various columns present in the dataset are:

- **target:** the polarity of the tweet (positive or negative)
- **ids:** Unique id of the tweet
- **date:** the date of the tweet
- **flag:** It refers to the query. If no such query exists then it is NO QUERY.
- **user:** It refers to the name of the user that tweeted
- **text:** It refers to the text of the tweet

Project Pipeline

The various steps involved in the **Machine Learning Pipeline** are :

- Import Necessary Dependencies
- Read and Load the Dataset
- Exploratory Data Analysis
- Data Visualization of Target Variables
- Data Preprocessing
- Splitting our data into Train and Test Subset
- Transforming Dataset using TF-IDF Vectorizer
- Function for Model Evaluation
- Model Building

-

Let’s get started.

Step-1: Import Necessary Dependencies

```
# utilities
import  re
import numpy as np
import pandas as pd
# plotting
import seaborn as sns
from wordcloud import WordCloud
import  matplotlib.pyplot   as   plt
#  nltk
from nltk.stem import WordNetLemmatizer
#  sklearn
from sklearn.svm import LinearSVC
from sklearn.naive_bayes import BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix, classification_report
```

Step-2: Read and Load the Dataset

```
# Importing the dataset
DATASET_COLUMNS=['target','ids','date','flag','user','text']
DATASET_ENCODING = "ISO-8859-1"
df = pd.read_csv('Project_Data.csv', encoding=DATASET_ENCODING,
names=DATASET_COLUMNS)
df.sample(5)
```

Output:

-

	target	ids	date	flag	user	text
305165	0	1999924339	Mon Jun 01 21:04:24 PDT 2009	NO_QUERY	tweetyred25	man my b-day is coming up but i dont know what...
673186	0	2247413241	Fri Jun 19 19:03:37 PDT 2009	NO_QUERY	belenneleb	@officialuti_ i need they to come back here. ...
573387	0	2209824277	Wed Jun 17 10:50:29 PDT 2009	NO_QUERY	TeenieWahine	@krystyn13 Sorry to hear that
246882	0	1982346860	Sun May 31 11:01:36 PDT 2009	NO_QUERY	BrianWCollins	@TheJoeLynch I've only seen 3 (Leon, 5th Eleme...
669112	0	2246153162	Fri Jun 19 17:10:15 PDT 2009	NO_QUERY	heidioftheopera	kinda feels bad for missing out on the Solstic...

Step-3: Exploratory Data Analysis

Five top records of data

```
df.head()
```

Output:

	target	ids	date	flag	user	text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

Columns/features in data

```
df.columns
```

Output:

```
Index(['target', 'ids', 'date', 'flag', 'user', 'text'], dtype='object')
```

Length of the dataset

```
print('length of data is',  len(df))
```

Output:

```
length of data is 1048576
```

Shape of data

-

```
df. shape
```

Output:

```
(1048576, 6)
```

Data information

```
df.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048576 entries, 0 to 1048575
Data columns (total 6 columns):
 #   Column  Non-Null Count    Dtype
---  ------  --------------    -----
 0   target  1048576 non-null  int64
 1   ids     1048576 non-null  int64
 2   date    1048576 non-null  object
 3   flag    1048576 non-null  object
 4   user    1048576 non-null  object
 5   text    1048576 non-null  object
dtypes: int64(2), object(4)
memory usage: 48.0+ MB
```

Datatypes of all columns

```
df.dtypes
```

-

Output:

```
target      int64
ids         int64
date       object
flag       object
user       object
text       object
dtype: object
```

Checking for Null values

```
np.sum(df.isnull().any(axis=1))
```

Output:

```
0
```

Rows and columns in the dataset

```
print('Count of columns in the data is:  ', len(df.columns))
print('Count of rows in the data is:  ', len(df))
```

Output:

```
Count of columns in the data is:6
Count of rows in the data is:  1048576
```

Check unique Target Values

```
df['target'].unique()
```

Output:

-

```
array([0, 4], dtype=int64)
```

Check the number of target values

```
df['target'].nunique()
```

Output:

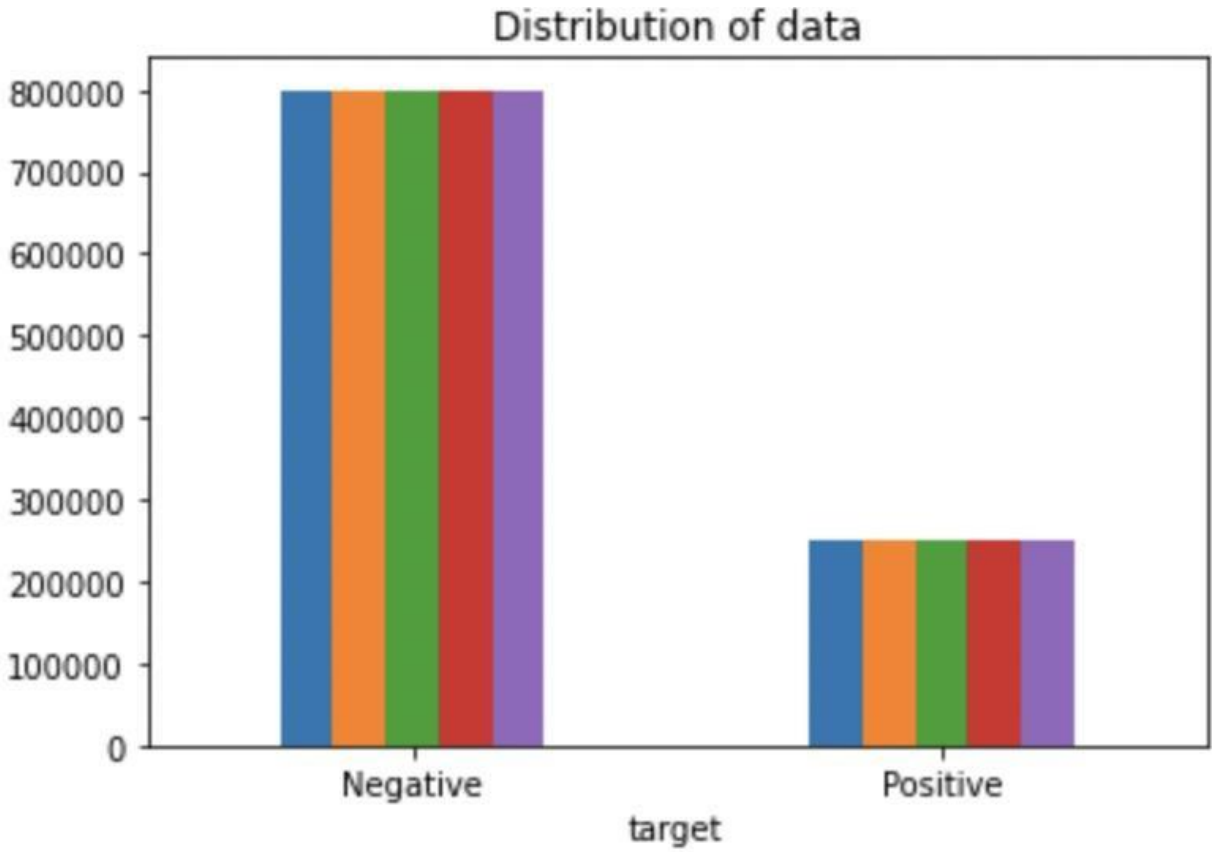
```
2
```

Step-4: Data Visualization of Target Variables

```
# Plotting the distribution for dataset.
ax = df.groupby('target').count().plot(kind='bar', title='Distribution of data',legend=False)
ax.set_xticklabels(['Negative','Positive'],    rotation=0)
# Storing data in lists.
text, sentiment = list(df['text']), list(df['target'])
```

Output:

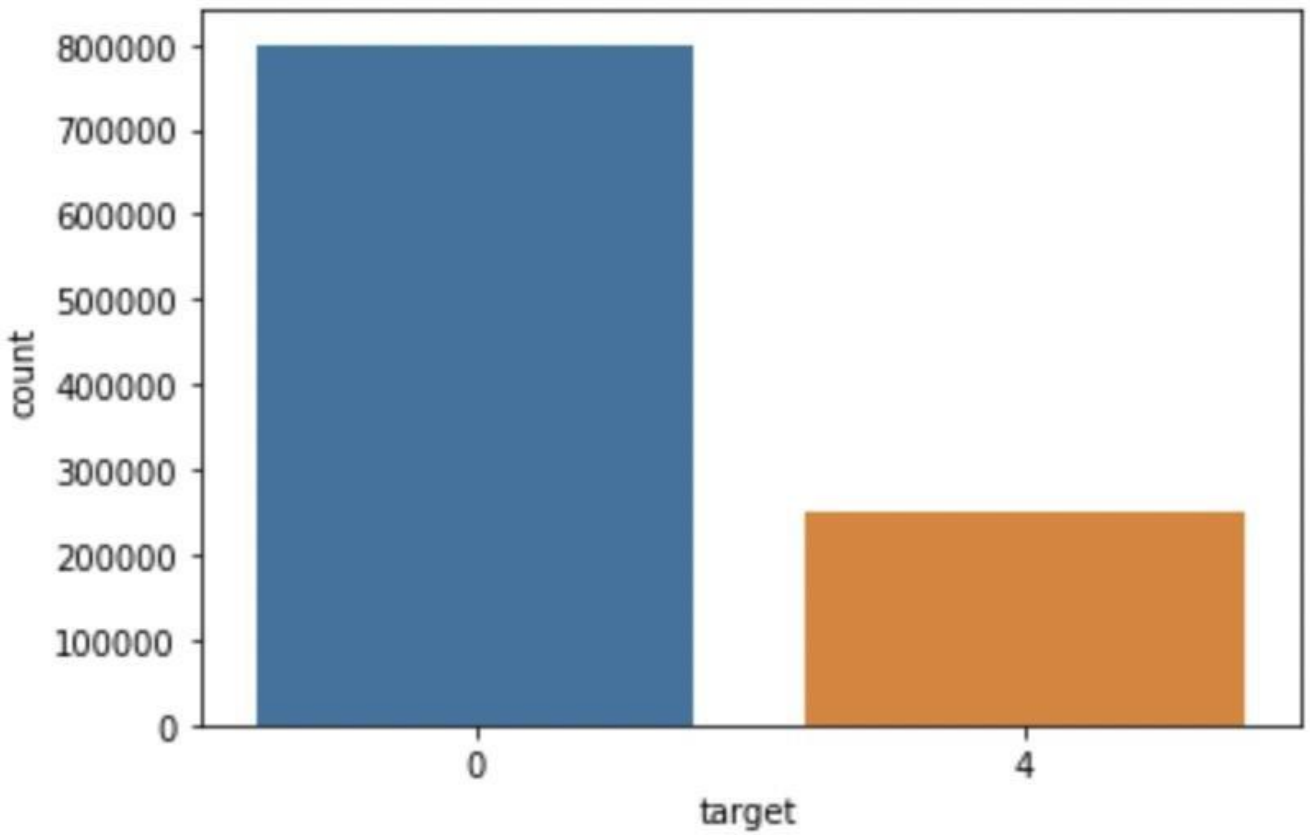
-




```
import seaborn as sns
sns.countplot(x='target', data=df)
```

Output:

-



Step-5: Data Preprocessing

In the above-given problem statement before training the model, we have performed various pre-processing steps on the dataset that mainly dealt with removing stopwords, removing emojis. The text document is then converted into the lowercase for better generalization.

Subsequently, the punctuations were cleaned and removed thereby reducing the unnecessary noise from the dataset. After that, we have also removed the repeating characters from the words along with removing the URLs as they do not have any significant importance.

At last, we then performed **Stemming(reducing the words to their derived stems)** and **Lemmatization(reducing the derived words to their root form known as lemma)** for better results.

5.1 : Selecting the text and Target column for our further analysis

-

```
data=df[['text','target']]
```

Replacing the values to ease understanding. (Assigning 1 to Positive sentiment 4)

```
data['target'] = data['target'].replace(4,1)
```

Print unique values of target variables

```
data['target'].unique()
```

Output:

```
array([0, 1], dtype=int64)
```

Separating positive and negative tweets

```
data_pos = data[data['target'] == 1]
data_neg = data[data['target'] == 0]
```

taking one fourth data so we can run on our machine easily

```
data_pos = data_pos.iloc[:int(20000)]
data_neg = data_neg.iloc[:int(20000)]
```

Combining positive and negative tweets

```
dataset = pd.concat([data_pos, data_neg])
```

Making statement text in lower case

```
dataset['text']=dataset['text'].str.lower()
dataset['text'].tail()
```

Output:

-

```
19995    not much time off this weekend, work trip to m...
19996                              one more day of holidays
19997    feeling so down right now .. i hate you damn h...
19998    geez,i hv to read the whole book of personalit...
19999    i threw my sign at donnie and he bent over to ...
Name: text, dtype: object
```

Defining set containing all stopwords in English.

```
stopwordlist = ['a', 'about', 'above', 'after', 'again', 'ain', 'all', 'am', 'an',
             'and','any','are', 'as', 'at', 'be', 'because', 'been', 'before',
             'being', 'below', 'between','both', 'by', 'can', 'd', 'did', 'do',
             'does', 'doing', 'down', 'during', 'each','few', 'for', 'from',
             'further', 'had', 'has', 'have', 'having', 'he', 'her', 'here',
             'hers', 'herself', 'him', 'himself', 'his', 'how', 'i', 'if', 'in',
             'into','is', 'it', 'its', 'itself', 'just', 'll', 'm', 'ma',
             'me', 'more', 'most','my', 'myself', 'now', 'o', 'of', 'on', 'once',
             'only', 'or', 'other', 'our', 'ours','ourselves', 'out', 'own',
're','s', 'same', 'she', "shes", 'should', "shouldve",'so', 'some', 'such',
             't', 'than', 'that', "thatll", 'the', 'their', 'theirs', 'them',
             'themselves', 'then', 'there', 'these', 'they', 'this', 'those',
             'through', 'to', 'too','under', 'until', 'up', 've', 'very', 'was',
             'we', 'were', 'what', 'when', 'where','which','while', 'who', 'whom',
             'why', 'will', 'with', 'won', 'y', 'you', "youd","youll", "youre",
             "youve", 'your', 'yours', 'yourself', 'yourselves']
```

Cleaning and removing the above stop words list from the tweet text

```
STOPWORDS = set(stopwordlist)
def cleaning_stopwords(text):
    return " ".join([word for word in str(text).split() if word not in STOPWORDS])
dataset['text'] = dataset['text'].apply(lambda text: cleaning_stopwords(text))
```

-

```
dataset['text'].head()
```

Output:

```
800000                love @health4uandpets u guys r best!!
800001    im meeting one besties tonight! cant wait!! - ...
800002    @darealsunisakim thanks twitter add, sunisa! g...
800003    sick really cheap hurts much eat real food plu...
800004                         @lovesbrooklyn2 effect everyone
Name: text, dtype: object
```

Cleaning and removing punctuations

```
import string
english_punctuations = string.punctuation
punctuations_list = english_punctuations
def cleaning_punctuations(text):
    translator = str.maketrans('', '', punctuations_list)
    return text.translate(translator)
dataset['text']= dataset['text'].apply(lambda x: cleaning_punctuations(x))
dataset['text'].tail()
```

Output:

```
19995    not much time off weekend work trip malmï¿½ fr...
19996                                      one day holidays
19997                        feeling right  hate damn humprey
19998    geezi hv read whole book personality types emb...
19999    threw sign donnie bent over get but thingee ma...
Name: text, dtype: object
```

Cleaning and removing repeating characters

```
def cleaning_repeating_char(text):
    return re.sub(r'(.)1+', r'1', text)
dataset['text'] = dataset['text'].apply(lambda x: cleaning_repeating_char(x))
```

-

```
dataset['text'].tail()
```

Output:

```
19995    not much time of wekend work trip malmï¿½ fris...
19996                                      one day holidays
19997                          feling right hate damn humprey
19998    gezi hv read whole bok personality types embar...
19999    threw sign donie bent over get but thinge made...
Name: text, dtype: object
```

Cleaning and removing URL's

```
def cleaning_URLs(data):
    return re.sub('((www.[^s]+)|(https?://[^s]+))',' ',data)
dataset['text'] = dataset['text'].apply(lambda x: cleaning_URLs(x))
dataset['text'].tail()
```

Output:

```
19995    not much time of wekend work trip malmï¿½ fris...
19996                                      one day holidays
19997                          feling right hate damn humprey
19998    gezi hv read whole bok personality types embar...
19999    threw sign donie bent over get but thinge made...
Name: text, dtype: object
```

Cleaning and removing Numeric numbers

```
def cleaning_numbers(data):
    return re.sub('[0-9]+', '', data)
dataset['text'] = dataset['text'].apply(lambda x: cleaning_numbers(x))
dataset['text'].tail()
```

Output:

-

```
19995    not much time of wekend work trip malmï¿½ fris...
19996                                        one day holidays
19997                          feling right hate damn humprey
19998    gezi hv read whole bok personality types embar...
19999    threw sign donie bent over get but thinge made...
Name: text, dtype: object
```

Getting tokenization of tweet text

```
from nltk.tokenize import RegexpTokenizer
tokenizer = RegexpTokenizer(r'w+')
dataset['text'] = dataset['text'].apply(tokenizer.tokenize)
dataset['text'].head()
```

Output:

```
800000              [love, healthuandpets, u, guys, r, best]
800001    [im, meting, one, besties, tonight, cant, wait...
800002    [darealsunisakim, thanks, twiter, ad, sunisa, ...
800003    [sick, realy, cheap, hurts, much, eat, real, f...
800004                          [lovesbroklyn, efect, everyone]
Name: text, dtype: object
```

Applying Stemming

```
import nltk
st =  nltk.PorterStemmer()
def stemming_on_text(data):
    text = [st.stem(word) for word in data]
    return  data
dataset['text']= dataset['text'].apply(lambda x: stemming_on_text(x))
dataset['text'].head()
```

Output:

-

```
800000              [love, healthuandpets, u, guys, r, best]
800001    [im, meting, one, besties, tonight, cant, wait...
800002    [darealsunisakim, thanks, twiter, ad, sunisa, ...
800003    [sick, realy, cheap, hurts, much, eat, real, f...
800004                        [lovesbroklyn, efect, everyone]
Name: text, dtype: object
```

Applying Lemmatizer

```
lm = nltk.WordNetLemmatizer()
def lemmatizer_on_text(data):
    text = [lm.lemmatize(word) for word in data]
    return  data
dataset['text'] = dataset['text'].apply(lambda x: lemmatizer_on_text(x))
dataset['text'].head()
```

Output:

```
800000              [love, healthuandpets, u, guys, r, best]
800001    [im, meting, one, besties, tonight, cant, wait...
800002    [darealsunisakim, thanks, twiter, ad, sunisa, ...
800003    [sick, realy, cheap, hurts, much, eat, real, f...
800004                        [lovesbroklyn, efect, everyone]
Name: text, dtype: object
```

Separating input feature and label

```
X=data.text
y=data.target
```

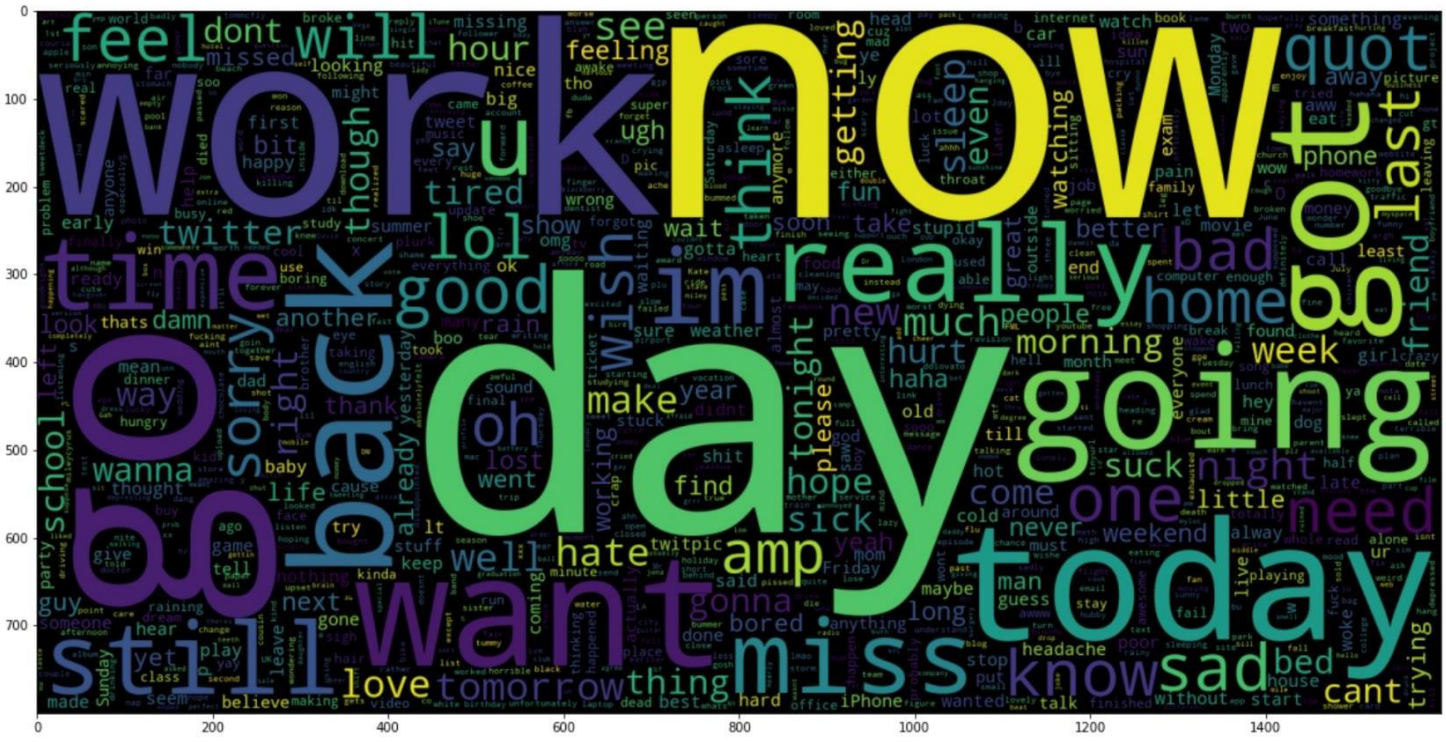
Plot a cloud of words for negative tweets

```
data_neg = data['text'][:800000]
plt.figure(figsize = (20,20))
```

-

```
wc = WordCloud(max_words = 1000 , width = 1600 , height = 800,
                    collocations=False).generate(" ".join(data_neg))
plt.imshow(wc)
```

Output:



Plot a cloud of words for positive tweets

```
data_pos = data['text'][800000:]
wc = WordCloud(max_words = 1000 , width = 1600 , height = 800,
                    collocations=False).generate(" ".join(data_pos))
plt.figure(figsize = (20,20))
plt.imshow(wc)
```

Output:

-

Step-6: Splitting our data into Train and Test Subset

```
# Separating the 95% data for training data and 5% for testing data
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.05,
random_state =26105111)
```

Step-7: Transforming Dataset using TF-IDF Vectorizer

Fit the TF-IDF Vectorizer

```
vectoriser = TfidfVectorizer(ngram_range=(1,2), max_features=500000)
vectoriser.fit(X_train)
print('No. of feature_words: ', len(vectoriser.get_feature_names()))
```

Output:

```
No.  of  feature_words:   500000
```

Transform the data using TF-IDF Vectorizer

```
X_train = vectoriser.transform(X_train)
```

-

```
X_test  =  vectoriser.transform(X_test)
```

Step-8: Function For Model Evaluation

After training the model we then apply the evaluation measures to check how the model is performing. Accordingly, we use the following evaluation parameters to check the performance of the models respectively :

- Accuracy Score
- Confusion Matrix with Plot
- ROC-AUC Curve

```
def  model_Evaluate(model):
# Predict values for Test dataset
y_pred =  model.predict(X_test)
# Print the evaluation metrics for the dataset.
print(classification_report(y_test, y_pred))
# Compute and plot the Confusion matrix
cf_matrix = confusion_matrix(y_test, y_pred)
categories = ['Negative','Positive']
group_names  =  ['True  Neg','False  Pos',  'False  Neg','True  Pos']
group_percentages = ['{0:.2%}'.format(value) for value in cf_matrix.flatten() /
np.sum(cf_matrix)]
labels = [f'{v1}n{v2}' for v1, v2 in zip(group_names,group_percentages)]
labels = np.asarray(labels).reshape(2,2)
sns.heatmap(cf_matrix, annot = labels, cmap = 'Blues',fmt = '',
xticklabels  =  categories,  yticklabels  =  categories)
plt.xlabel("Predicted values", fontdict = {'size':14}, labelpad = 10)
plt.ylabel("Actual values" ,  fontdict = {'size':14}, labelpad = 10)
plt.title ("Confusion Matrix", fontdict = {'size':18}, pad =  20)
```

Step-9: Model Building

In the problem statement we have used three different models respectively :

-

- Bernoulli Naive Bayes
- SVM (Support Vector Machine)
- Logistic Regression

The idea behind choosing these models is that we want to try all the classifiers on the dataset ranging from simple ones to complex models and then try to find out the one which gives the best performance among them.

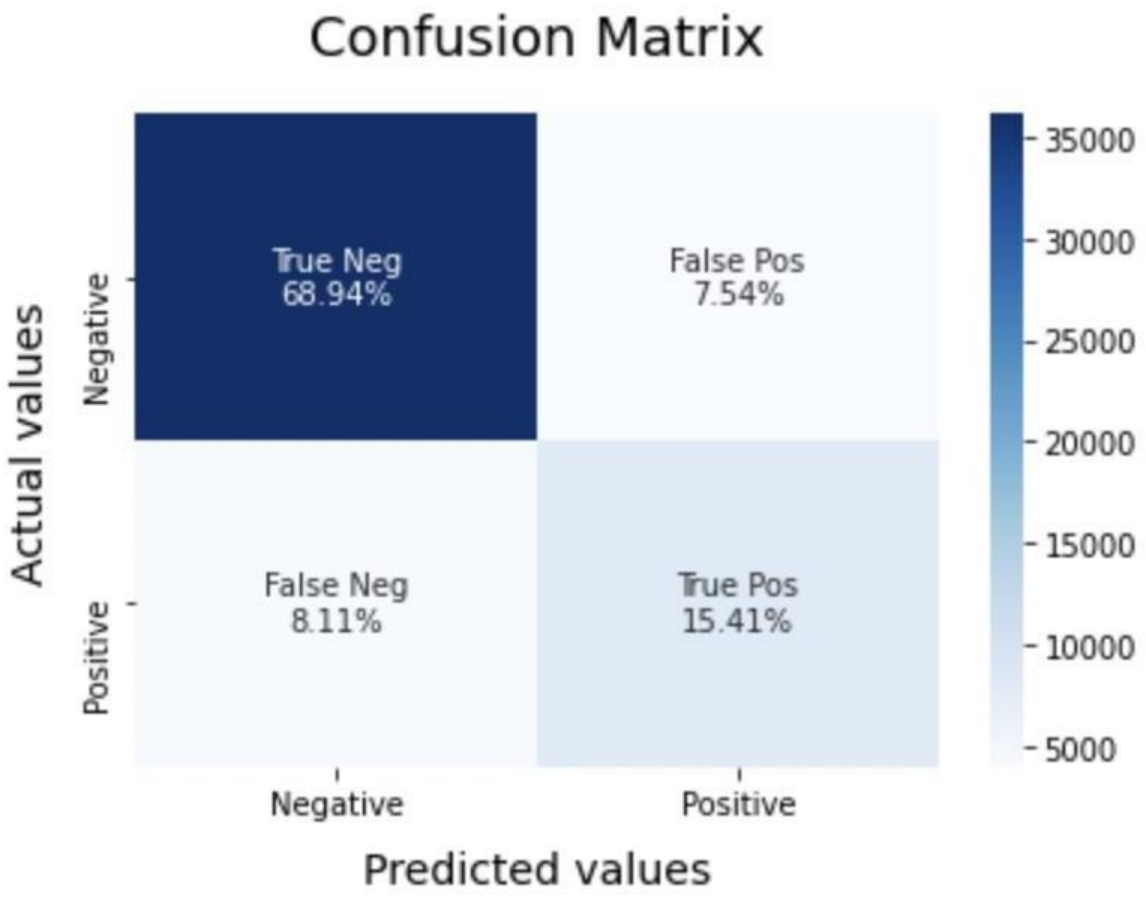
Model-1

```
BNBmodel = BernoulliNB()
BNBmodel.fit(X_train, y_train)
model_Evaluate(BNBmodel)
y_pred1 = BNBmodel.predict(X_test)
```

Output:

-

	precision	recall	f1-score	support
0	0.89	0.90	0.90	40097
1	0.67	0.66	0.66	12332
accuracy			0.84	52429
macro avg	0.78	0.78	0.78	52429
weighted avg	0.84	0.84	0.84	52429



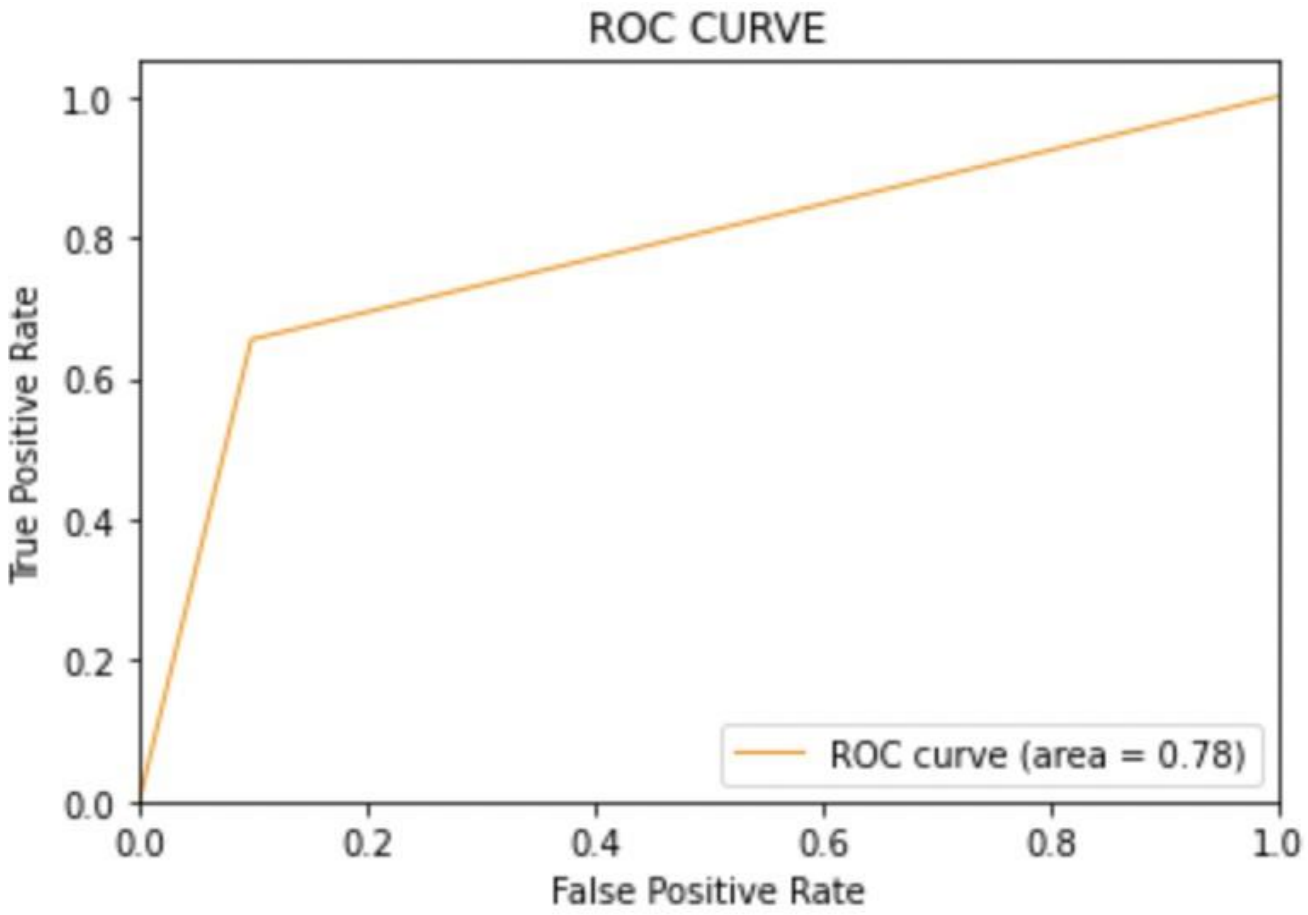
Plot the ROC-AUC Curve for model-1

```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_pred1)
roc_auc = auc(fpr, tpr)
plt.figure()
```

-

```
plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (area = %0.2f)' %
roc_auc)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC CURVE')
plt.legend(loc="lower right")
plt.show()
```

Output:



Model-2:

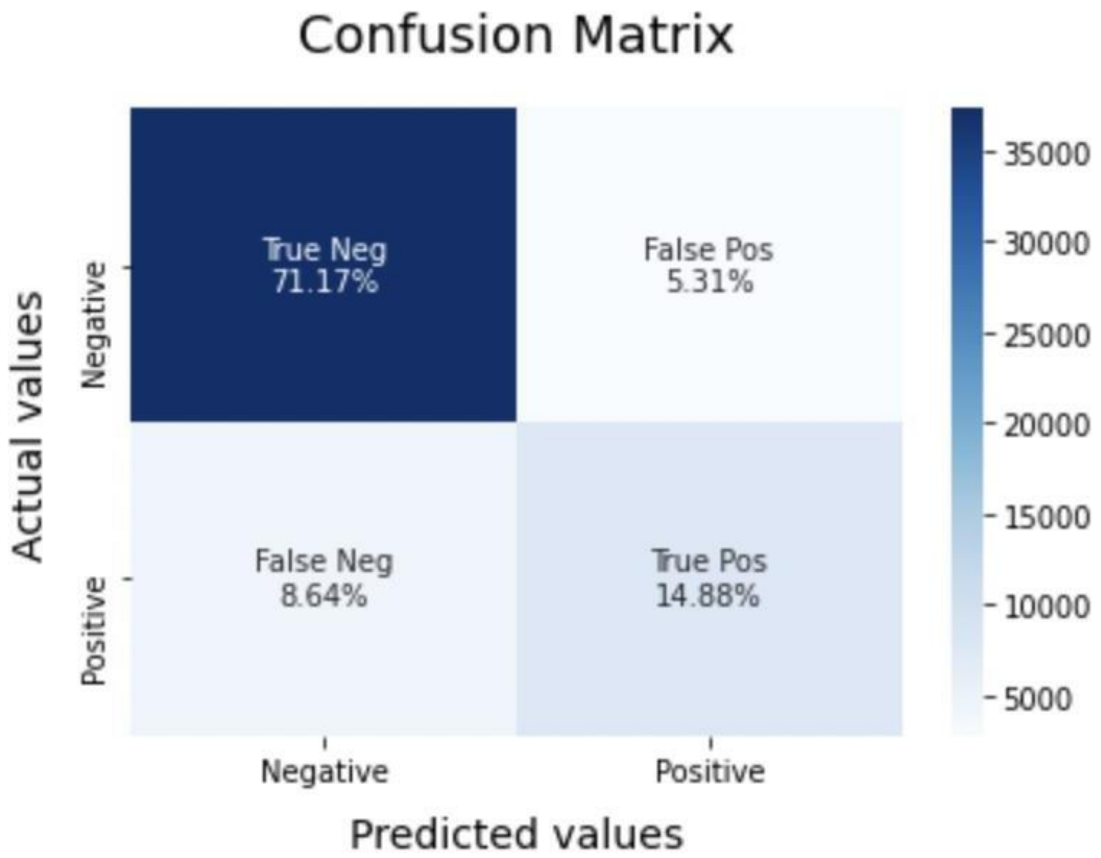
```
SVCmodel = LinearSVC()
SVCmodel.fit(X_train, y_train)
```

-

```
model_Evaluate(SVCmodel)
y_pred2 = SVCmodel.predict(X_test)
```

Output:

	precision	recall	f1-score	support
0	0.89	0.93	0.91	40097
1	0.74	0.63	0.68	12332
accuracy			0.86	52429
macro avg	0.81	0.78	0.80	52429
weighted avg	0.86	0.86	0.86	52429



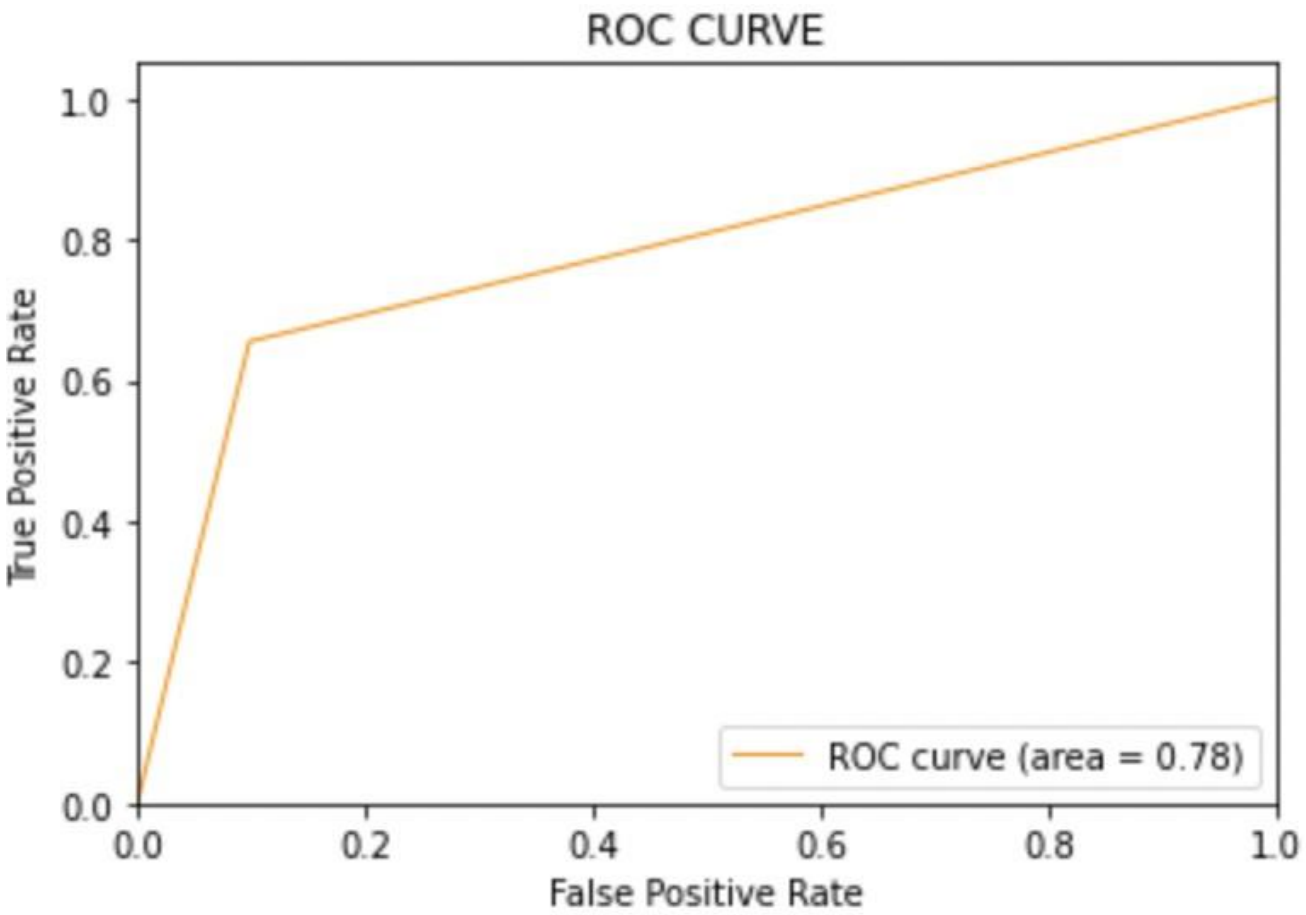
Plot the ROC-AUC Curve for model-2

```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_pred2)
```

-

```
roc_auc = auc(fpr, tpr)
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (area = %0.2f)' %
roc_auc)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC CURVE')
plt.legend(loc="lower right")
plt.show()
```

Output:



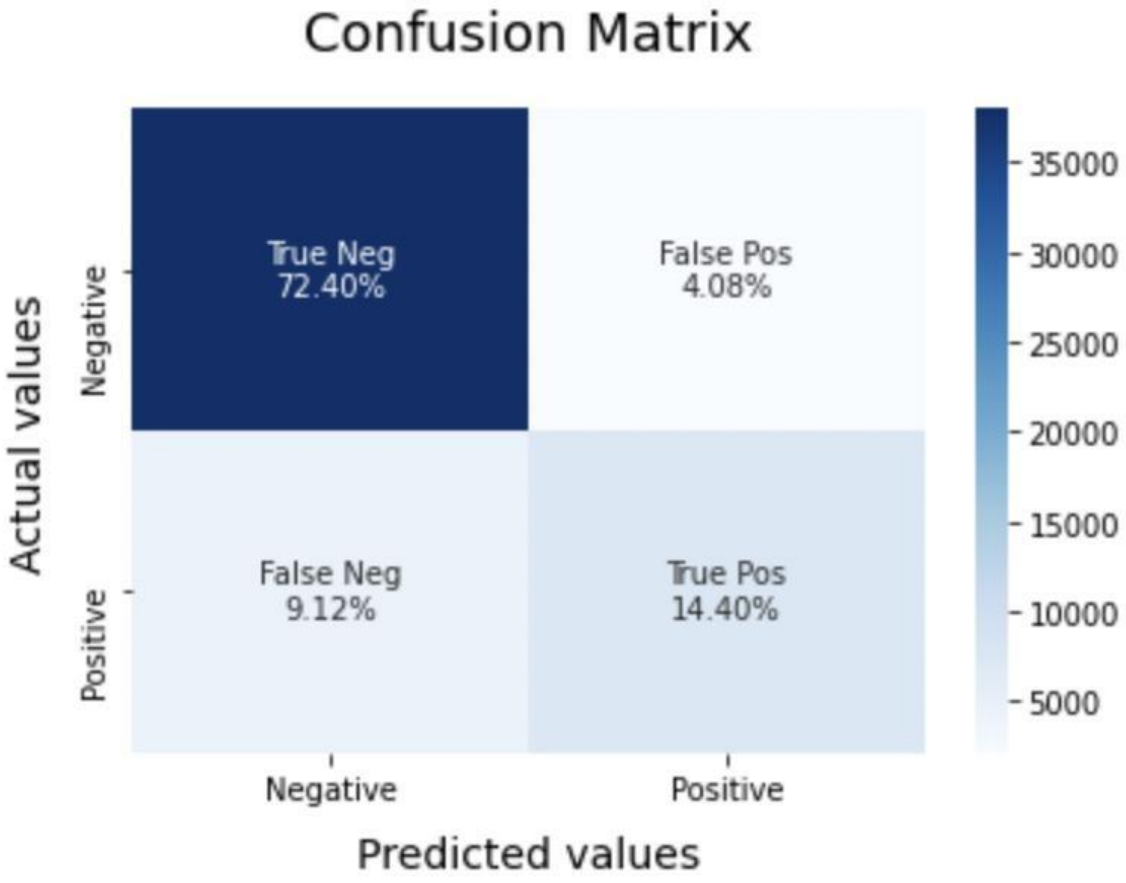
Model-3

-

```
LRmodel = LogisticRegression(C = 2, max_iter = 1000, n_jobs=-1)
LRmodel.fit(X_train,  y_train)
model_Evaluate(LRmodel)
y_pred3 = LRmodel.predict(X_test)
```

Output:

	precision	recall	f1-score	support
0	0.89	0.95	0.92	40097
1	0.78	0.61	0.69	12332
accuracy			0.87	52429
macro avg	0.83	0.78	0.80	52429
weighted avg	0.86	0.87	0.86	52429

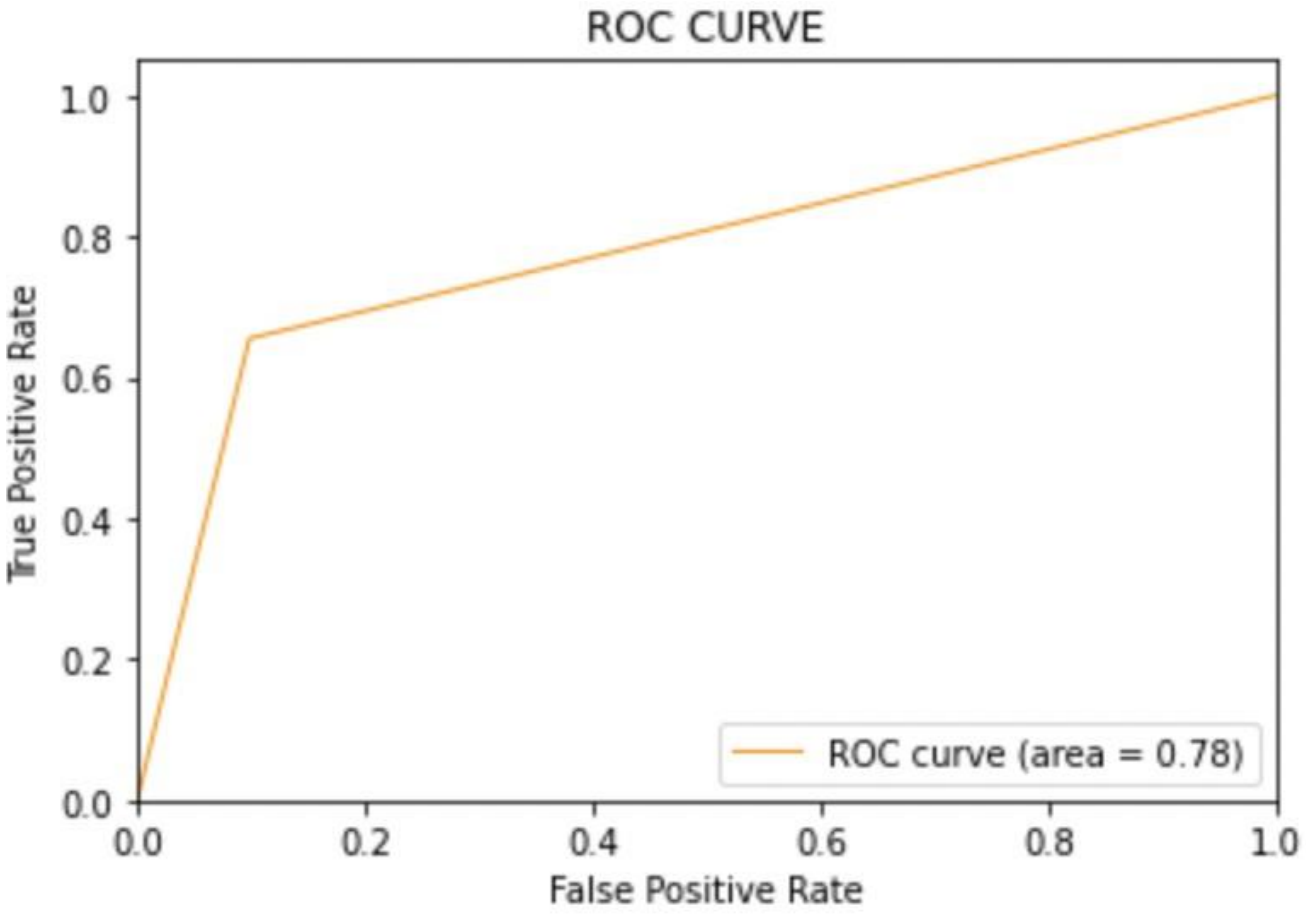


Plot the ROC-AUC Curve for model-3

-

```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(y_test, y_pred3)
roc_auc = auc(fpr, tpr)
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (area = %0.2f)' %
roc_auc)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC CURVE')
plt.legend(loc="lower right")
plt.show()
```

Output:



-

Step-10: Conclusion

Upon evaluating all the models we can conclude the following details i.e.

Accuracy: As far as the accuracy of the model is concerned Logistic Regression performs better than SVM which in turn performs better than Bernoulli Naive Bayes.

F1-score: The F1 Scores for class 0 and class 1 are :
(a) For class 0: Bernoulli Naive Bayes(accuracy = 0.90) < SVM (accuracy =0.91) < Logistic Regression (accuracy = 0.92)
(b) For class 1: Bernoulli Naive Bayes (accuracy = 0.66) < SVM (accuracy = 0.68) < Logistic Regression (accuracy = 0.69)

AUC Score: All three models have the same ROC-AUC score.

We, therefore, conclude that the Logistic Regression is the best model for the above-given dataset.

In our problem statement, **Logistic Regression** is following the principle of **Occam's Razor** which defines that for a particular problem statement if the data has no assumption, then the simplest model works the best. Since our dataset does not have any assumptions and Logistic Regression is a simple model, therefore the concept holds true for the above-mentioned dataset.

-