

PROJECT REPORT

ON

.....

Submitted in partial fulfilment of the Requirement for the award of the degree Of
Bachelor of Technology In
COMPUTER SCIENCE & ENGINEERING



Submitted By

Vidushi Gandhi

20015004058

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING ECHELON
INSTITUTE OF TECHNOLOGY, FARIDABAD

JANUARY 2023 – JUNE 2023



CERTIFICATE

I hereby certify that the work which is being presented in the B.Tech. Project Report entitled, **Depression Detection** in partial fulfilment of the requirements for the award of the Bachelor of Technology in Computer Science & Engineering and submitted to the Department of Computer Science & Engineering of Echelon Institute of Technology, Faridabad is an authentic record of my own work carried out during a period from January 2023 to June 2023.

I have not submitted the matter presented in this thesis for the award of any other degree elsewhere.

Signature of Candidate

Vidushi Gandhi

20-CSE-058

20015004058



TO WHOM IT MAY CONCERN

This is to certify that the Project entitled “**Depression Detection**” submitted by “**Vidushi Gandhi**” (20- CSE-058) Department of Computer Science and Engineering, Echelon Institute of Technology Under YMCA University, Faridabad, for partial fulfilment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering; is a bonafide record of the work and investigations carried out by him under my supervision and guidance

Signature of HOD

Prof. (Dr.) Jugnesh Kumar

Head of Department

Signature of the Supervisor

Ms. Stuti Saxena

Assistant Professor

ACKNOWLEDGEMENTS

Thank all those who have helped me in completing the project successfully.

I would like to express my gratitude to **Ms. Stuti Saxena**, who as my guide/mentor provided me with every possible support and guidance throughout the development of the project. This project would never have been completed without her encouragement and support.

I would also like to show my gratitude to **Dr Jugneesh Sir**, Head of Department for providing us with well-trained Team members and giving us all the required resources and a healthy environment for carrying out our project work.

Vidushi Gandhi

20015004058

ABSTRACT

Depression or Mood Disorder has been one of the most prevalent mental disorders. According to WHO in 2020, more than 280 million people were diagnosed with Depression and more than 850 thousand deaths have been reported globally. that can cause a loss of interest in general action that can direct to suicidal thoughts. This project aims to use the machine-learning technique Naive Bayes and Natural Language Processing for depression detection. Early detection and accurate prediction are fundamental to identifying patients who could benefit from the treatment and reduce the mortality rate due to depression. With the help of various ML techniques, we can detect depression more efficiently and effectively. This research's primary contribution is the exploration part of the features and their impact on detecting depression levels.

TABLE OF CONTENTS

S.NO	NAME OF CONTENT
1	Certificate
2	Candidate Declaration
3	Acknowledgements
4	Abstract
5	Chapter 1 Introduction
6	Chapter 2 Project Flow
7	2.1 Dataset
8	2.2 Data Pre-Processing
9	2.3 Data Partitioning
10	2.4 ML Model

11	2.5 Result Analysis
12	2.6 Predictions
13	Chapter 3 Requirement Analysis
14	3.1 Python Model Used
15	Chapter 4 ML Model Used
16	Chapter 5 Code
17	Chapter 6 Output Screen
18	Chapter 7 Future Scope

Chapter 1

Introduction to the Project

What is Depression?

Depression (major depressive disorder) is a common and serious medical illness that negatively affects how you feel, the way you think and how you act. Fortunately, it is also treatable. Depression causes feelings of sadness and/or a loss of interest in activities you once enjoyed. It can lead to a variety of emotional and physical problems and can decrease your ability to function at work and at home.

Depression symptoms can vary from mild to severe and can include:

- Feeling sad or having a depressed mood
- Loss of interest or pleasure in activities once enjoyed
- Changes in appetite — weight loss or gain unrelated to dieting
- Trouble sleeping or sleeping too much
- Loss of energy or increased fatigue
- Increase in purposeless physical activity (e.g., inability to sit still, pacing, handwringing) or slowed movements or speech (these actions must be severe enough to be observable by others)
- Feeling worthless or guilty
- Difficulty thinking, concentrating or making decisions
- Thoughts of death or suicide

What is Depression Detection?

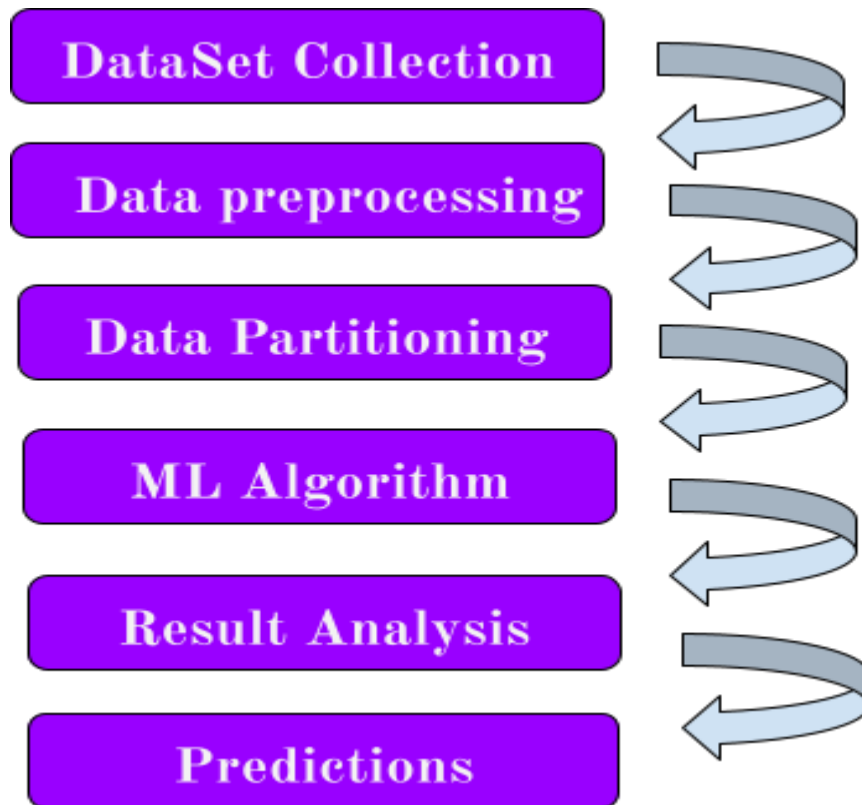
Depression Detection is the problem of identifying signs of depression in individuals. These signs might be identified in peoples' speech, facial expressions and in the use of language.

Why Depression Detection System?

Early recognition and treatment of depression can improve the negative impacts of the disorder. Therefore, it is vital to provide early identification of subjects suffering from depression to intervene as soon as possible and minimize the impact on public health by potentially reducing the escalation of the disease. However, provisions and services for the early detection and treatment of depression and other mental health disorders remain limited. Although there are also some validated laboratory tests to diagnose depression, such as Beck Depression Inventory-II, Center for Epidemiologic Studies Depression Scale (CES-D), Geriatric Depression Scale, Hospital Anxiety and Depression Scale, Patient Health Questionnaire-9 and Hamilton Rating Scale for Depression, most diagnoses are formed on the basis of self- or family reports.

Chapter 2

Project Flow



1. DATASET
2. DATA PREPROCESSING
3. DATA PARTITIONING
4. ML MODEL
5. RESULT ANALYSIS
6. PREDICTION

2.1 DATASET

This is the sentiment140 dataset. It contains 1,600,000 tweets extracted using the Twitter API. The tweets have been annotated (0 = negative, 4 = positive) and they can be used to detect sentiment . In this data set, there were six attributes out of these six attributes in this we have used three fields that are target, text, and user.

Depression Detection Dataset

The data gathered has the following details:

It contains the following 6 fields:

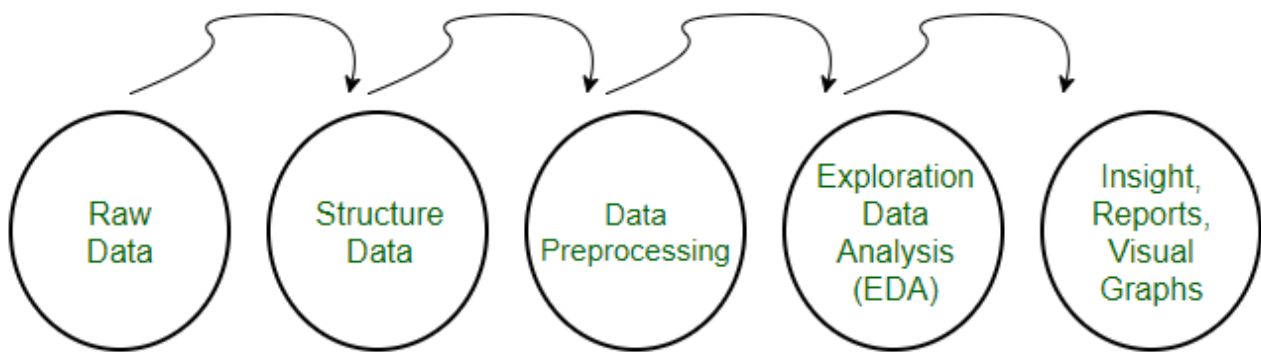
1. **target:** the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)
2. **ids:** The id of the tweet (2087)
3. **date:** the date of the tweet (*Sat May 16 23:58:44 UTC 2009*)
4. **flag:** The query (*lyx*). If there is no query, then this value is NO_QUERY.
5. **user:** the user that tweeted (*robotickilldozr*)
6. **text:** the text of the tweet (*Lyx is cool*)

```
df.head()
```

	target	ids	date	flag	user	text
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	http://twitpic.com/2y1zl - Awww, t... @switchfoot
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....

2.2 DATA PRE-PROCESSING

Pre-processing refers to the transformations applied to our data before feeding it to the algorithm. Data Pre-processing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.



Need of Data Pre-processing

- For achieving better results from the applied model in Machine Learning projects the format of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, Naive Bayes algorithm does not support null values, therefore to execute naive bayes algorithm null values have to be managed from the original raw data set.
- Another aspect is that the data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithm are executed in one data set, and best out of them is chosen.

2.3 DATA PARTITIONING

Data partitioning in data mining is the division of the whole data available into two or three non-overlapping sets: the training set , the validation set , and the test set . If the data set is very large, often only a portion of it is selected for the partitions. Partitioning is normally used when the model for the data at hand is being chosen from a broad set of models. The basic idea of data partitioning is to keep a subset of available data out of analysis, and to use it later for verification of the model.

For example, a researcher developed a method for prediction of time series of stock prices data. The parameters of the model have been fitted to the available data, and the model demonstrates high prediction accuracy on these data. But this does not necessarily mean that the model will predict new data that well -- the model has been especially tuned to the characteristics (including random chance aspects) of the data used to fit it. Data partitioning is used to avoid such overly optimistic estimates of the model precision.

Data partitioning is normally used in supervised learning techniques in data mining where a predictive model is chosen from a set of models, using their performance on the training set as the validation of choice. Some examples of such techniques are classification trees , regression trees , neural networks , nonlinear variants of the discriminant analysis .

In this Machine Learning model we have split the dataset according to 80-20 percentage in test and train dataset .

```
from sklearn.model_selection import train_test_split
X = new_df['text']
y = new_df['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.05, random
print(X_train.shape,X_test.shape,y_train.shape,y_test.shape)
```

```
(1520000,) (80000,) (1520000,) (80000,)
```

```
y_train.value_counts()
```

```
0    760001
1    759999
Name: label, dtype: int64
```

2.4 ML MODEL

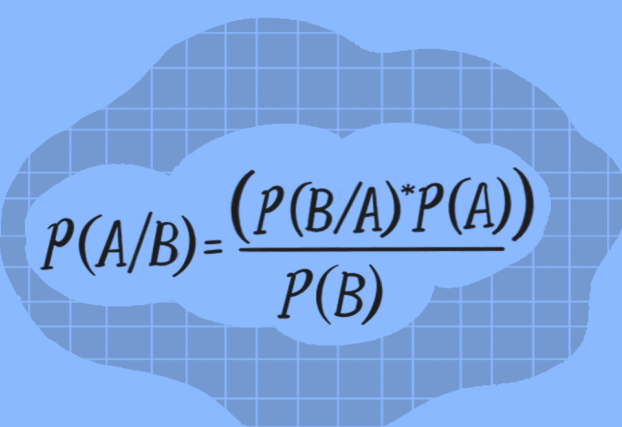
Machine learning is an artificial intelligence technology that becomes proficient at autonomously performing a task, when given data and examples of desired behavior. It can identify meaningful patterns that humans may not have been able to detect as quickly without the machine's help.

A machine learning model is a program that can find patterns or make decisions from a previously unseen dataset. For example, in natural language processing, machine learning models can parse and correctly recognize the intent behind previously unheard sentences or combinations of words. In image recognition, a machine learning model can be taught to recognize objects - such as cars or dogs. A machine learning model can perform such tasks by having it 'trained' with a large dataset. During training, the machine learning algorithm is optimized to find certain patterns or outputs from the dataset, depending on the task. The output of this process - often a computer program with specific rules and data structures - is called a machine learning model.

In supervised machine learning, the algorithm is provided an input dataset, and is rewarded or optimized to meet a set of specific outputs. For example, supervised machine learning is widely deployed in image recognition, utilizing a technique called classification. Supervised machine learning is also used in predicting demographics such as population growth or health metrics, utilizing a technique called regression.

The Machine Learning Algorithm we have apply in this model is **Naive Bayes Machine Learning Algorithm**

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. Bayes' theorem states the following relationship, given class variable


$$P(A/B) = \frac{(P(B/A) * P(A))}{P(B)}$$

Bayes' Theorem

[ˈbāz ˈthē-ə-rəm]

A mathematical formula for determining conditional probability.

The Multinomial Naive Bayes algorithm is a Bayesian learning approach popular in Natural Language Processing (NLP). The 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 a given sample and outputs the tag with the greatest chance.

Text data classification is gaining popularity because there is an enormous amount of information available in email, documents, websites, etc. that needs to be analyzed. Knowing the context around a certain type of text helps in finding the perception of a software or product to users who are going to use it.

target		text	clean_text
0	0	@switchfoot http://twitpic.com/2y1zI - Awww, t...	switchfoot awww thats a bumner you shoulda ...
1	0	is upset that he can't update his Facebook by ...	is upset that he cant update his facebook by t...
2	0	@Kenichan I dived many times for the ball. Man...	kenichan i dived many times for the ball manag...
3	0	my whole body feels itchy and like its on fire	my whole body feels itchy and like its on fire
4	0	@nationwideclass no, it's not behaving at all....	nationwideclass no its not behaving at all im ...

2.5 RESULT ANALYSIS

By using this ML Model we receive an accuracy of 84% on the train dataset and 77.34 % on the test dataset

```
validation = model.predict(X_test)
```

```
validation1 = model.predict(X_train)
```

```
from sklearn.metrics import accuracy_score  
accuracy_score(y_train, validation1)
```

```
0.843316447368421
```

```
from sklearn.metrics import accuracy_score  
accuracy_score(y_test, validation)
```

```
0.7736
```

2.6 PREDICTIONS

```
train = pd.DataFrame()  
train['label'] = y_train  
train['text'] = X_train  
  
def predict_category(s, train=X_train, model=model):  
    pred = model.predict([s])  
    return pred[0]
```

```
predict_category("i wanna shot myself")
```

```
0
```

```
predict_category("i love you")
```

```
1
```

Activate V

CHAPTER 3

REQUIREMENT ANALYSIS

- **PROJECT AIM -**

- **Depression Detection**
- **Work on Twitter API**
- **Detect whether you are suffering from depression or not**

- **HARDWARE REQUIREMENTS**

- **4 GB RAM**
- **64 Bit OS**

- **SOFTWARE REQUIREMENTS**

- **VS CODE**
- **JUPYTER Notebook**
- **Chrome**
- **Dataset**

- **PYTHON MODULES -**

- **Pandas**
- **NLP**
- **NLTK**

- **Multinomial Naive Bayes**

- **Gaussian Naive Bayes**

- **Numpy**

CHAPTER 4

ML MODEL

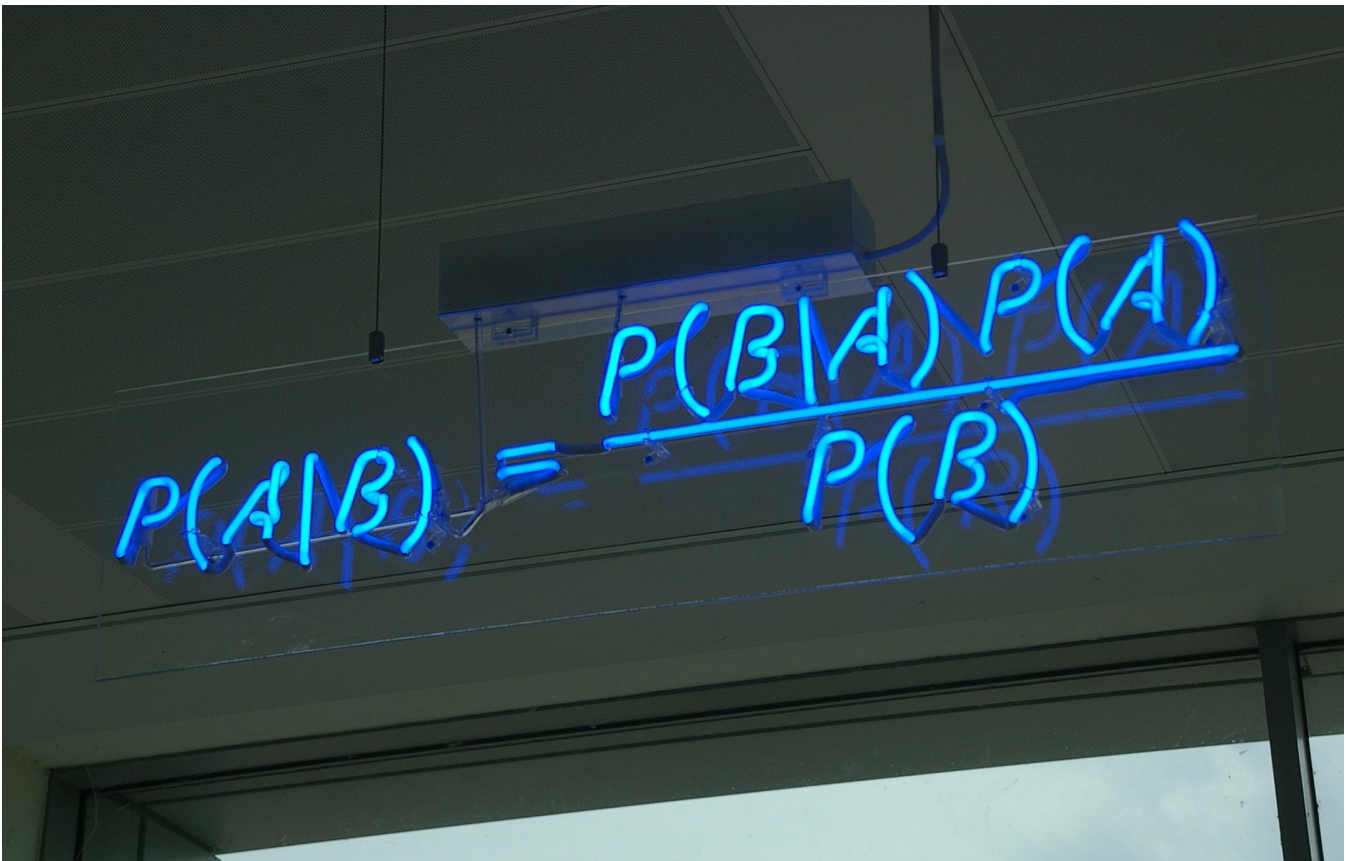
Naive Bayes

Naive Bayes is a machine learning model that is used for large volumes of data, even if you are working with data that has millions of data records the recommended approach is Naive Bayes. It gives very good results when it comes to NLP tasks such as sentimental analysis. It is a fast and uncomplicated classification algorithm.

Bayes Theorem

It is a theorem that works on conditional probability. Conditional probability is the probability that something will happen, given that something else has already occurred. The conditional probability can give us the probability of an event using its prior knowledge.

Conditional Probability:


$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Where,

P(A): The probability of hypothesis H being true. This is known as the prior probability.

P(B): The probability of the evidence.

$P(A|B)$: The probability of the evidence given that hypothesis is true.

$P(B|A)$: The probability of the hypothesis given that the evidence is true.

Naive Bayes Classifier

- It is a kind of classifier that works on Bayes theorem.
- Prediction of membership probabilities is made for every class such as the probability of data points associated to a particular class.
- The class having maximum probability is appraised as the most suitable class.
- This is also referred as Maximum A Posteriori (MAP).
- The MAP for a hypothesis is:
 - $MAP(H) = \max P((H|E))$
 - $MAP(H) = \max P((H|E) * (P(H)) / P(E))$
 - $MAP(H) = \max(P(E|H) * P(H))$
 - $P(E)$ is evidence probability, and it is used to normalize the result. The result will not be affected by removing $P(E)$.
- NB classifiers conclude that all the variables or features are not related to each other.
- The Existence or absence of a variable does not impact the existence or absence of any other variable.
- Example:
 - Fruit may be observed to be an apple if it is red, round, and about 4" in diameter.
 - In this case also even if all the features are interrelated to each other, and NB classifier will observe all of these independently contributing to the probability that the fruit is an apple.
- We experiment with the hypothesis in real datasets, given multiple features.
- So, computation becomes complex.

Types Of Naive Bayes Algorithms

1. Gaussian Naïve Bayes: When characteristic values are continuous in nature then an assumption is made that the values linked with each class are dispersed according to Gaussian that is Normal Distribution.
2. Multinomial Naïve Bayes: Multinomial Naive Bayes is favored to use on data that is multinomial distributed. It is widely used in text classification in NLP. Each event in text classification constitutes the presence of a word in a document.
3. Bernoulli Naïve Bayes: When data is dispensed according to the multivariate Bernoulli distributions then Bernoulli Naive Bayes is used. That means there exist multiple features but each one is assumed to contain a binary value. So, it requires features to be binary-valued.

CHAPTER 5

CODE

JUPYTER CODING

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as py
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import classification_report
import string
import nltk
import re

dataset_columns = ["target", "ids", "date", "flag", "user", "text"]
dataset_encode = "ISO-8859-1"
df=pd.read_csv("training.1600000.processed.noemoticon.csv", encoding =
dataset_encode,names=dataset_columns)

df.head()

df.drop(['ids','date','flag','user'],axis = 1,inplace = True)

df['target'].value_counts()

#remove punctuation
def remove_punctuation(text):
    no_punct=[words for words in text if words not in string.punctuation]
    words_wo_punct="".join(no_punct)
    return words_wo_punct
df['clean_text']=df['text'].apply(lambda x: remove_punctuation(x))
df.head()

#remove hyperlink
df['clean_text'] = df['clean_text'].str.replace(r"http\S+", "")
#remove emoji
```

```

df['clean_text'] = df['clean_text'].str.replace('[^\w\s#@/:%.,_-]', '', flags=re.UNICODE)
#convert all words to lowercase
df['clean_text'] = df['clean_text'].str.lower()
df.head()
#tokenization
nltk.download('punkt')
def tokenize(text):
    split=re.split("\W+",text)
    return split
df['clean_text_tokenize']=df['clean_text'].apply(lambda x: tokenize(x.lower()))

import nltk

nltk.download('stopwords')

from nltk.corpus import stopwords

stopwords.words('english')

# #stopwords
# nltk.download('stopwords')
stopword = nltk.corpus.stopwords.words('english')
def remove_stopwords(text):
    text=[word for word in text if word not in stopword]
    return text
df['clean_text_tokenize_stopwords'] = df['clean_text_tokenize'].apply(lambda x:
remove_stopwords(x))
df.head(10)

new_df = pd.DataFrame()
new_df['text'] = df['clean_text']
new_df['label'] = df['target']
new_df['label'] = new_df['label'].replace(4,1)

print(new_df.head())
print('Label: \n', new_df['label'].value_counts())

from sklearn.model_selection import train_test_split
X = new_df['text']
y = new_df['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.05, random_state=42)

print(X_train.shape,X_test.shape,y_train.shape,y_test.shape)

y_train.value_counts()

```

```
model = make_pipeline(TfidfVectorizer(), MultinomialNB())
```

```
model.fit(X_train,y_train)
```

```
validation = model.predict(X_test)
```

```
validation1 = model.predict(X_train)
```

```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(y_train, validation1)
```

```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(y_test, validation)
```

```
cf_matrix = confusion_matrix(y_test, validation)
```

```
sns.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%', cmap='Greens')
```

```
print(classification_report(y_test, validation))
```

```
train = pd.DataFrame()
```

```
train['label'] = y_train
```

```
train['text'] = X_train
```

```
def predict_category(s, train=X_train, model=model):
```

```
    pred = model.predict([s])
```

```
    return pred[0]
```

```
predict_category("i wanna shot myself")
```

```
predict_category("i love you")
```

CHAPTER 6

OUTPUT SCREEN

1) Import Dependency and Dataset

```
In [ ]: import numpy as np
import pandas as pd
from matplotlib import pyplot as py
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import classification_report
import string
import nltk
import re
```

```
In [2]: dataset_columns = ["target", "ids", "date", "flag", "user", "text"]
dataset_encode = "ISO-8859-1"
df=pd.read_csv("training.1600000.processed.noemoticon.csv", encoding = dataset_e
```

2) Data Preprocessing

```
#remove punctuation
def remove_punctuation(text):
    no_punct=[words for words in text if words not in string.punctuation]
    words_wo_punct=''.join(no_punct)
    return words_wo_punct
df['clean_text']=df['text'].apply(lambda x: remove_punctuation(x))
df.head()
```

	target	text	clean_text
0	0	@switchfoot http://twitpic.com/2y1zl - Awww, t...	switchfoot httptwitpiccom2y1zl Awww thats a b...
1	0	is upset that he can't update his Facebook by ...	is upset that he cant update his Facebook by t...
2	0	@Kenichan I dived many times for the ball. Man...	Kenichan I dived many times for the ball Manag...
3	0	my whole body feels itchy and like its on fire	my whole body feels itchy and like its on fire
4	0	@nationwideclass no, it's not behaving at all....	nationwideclass no its not behaving at all im ...

```
new_df = pd.DataFrame()
new_df['text'] = df['clean_text']
new_df['label'] = df['target']
new_df['label'] = new_df['label'].replace(4,1)
```

```
print(new_df.head())
print('Label: \n', new_df['label'].value_counts())
```

```

              text  label
0  switchfoot    awww thats a bummer  you shoulda ...      0
1  is upset that he cant update his facebook by t...      0
2  kenichan i dived many times for the ball manag...      0
3    my whole body feels itchy and like its on fire      0
4  nationwideclass no its not behaving at all im ...      0
Label:
0      800000
1      800000
Name: label, dtype: int64
```

3) Dataset Partitioning

```

: from sklearn.model_selection import train_test_split
X = new_df['text']
y = new_df['label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.05, random
print(X_train.shape,X_test.shape,y_train.shape,y_test.shape)

(1520000,) (80000,) (1520000,) (80000,)
```

```

: y_train.value_counts()

: 0      760001
  1      759999
  Name: label, dtype: int64
```

4) ML Model

```
In [16]: model = make_pipeline(TfidfVectorizer(), MultinomialNB())
```

```
In [17]: model.fit(X_train,y_train)
```

```
Out[17]: Pipeline(steps=[('tfidfvectorizer', TfidfVectorizer()),
                          ('multinomialnb', MultinomialNB())])
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.

On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

5) Result Analysis

```
validation = model.predict(X_test)
```

```
validation1 = model.predict(X_train)
```

```
from sklearn.metrics import accuracy_score  
accuracy_score(y_train, validation1)
```

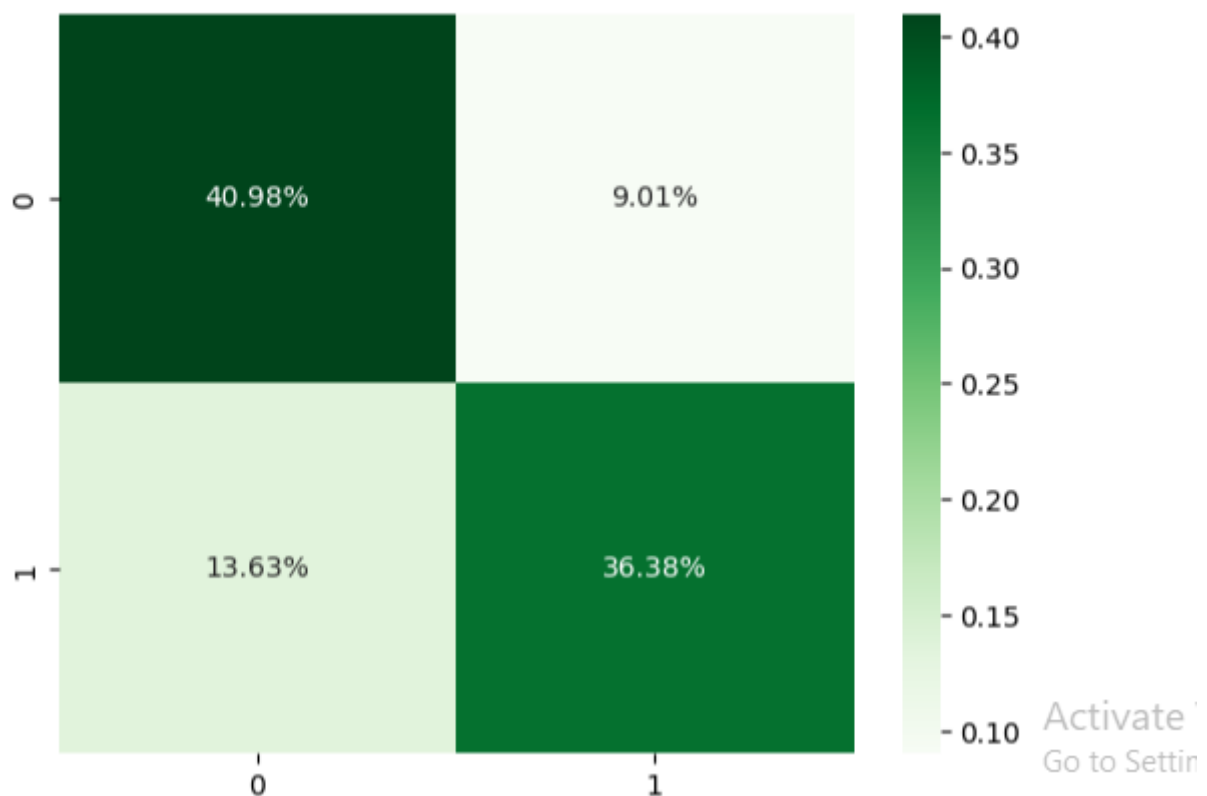
```
0.843316447368421
```

```
from sklearn.metrics import accuracy_score  
accuracy_score(y_test, validation)
```

```
0.7736
```

```
cf_matrix = confusion_matrix(y_test, validation)  
sns.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%', cmap='Greens')
```

<AxesSubplot: >



```
print(classification_report(y_test, validation))
```

	precision	recall	f1-score	support
0	0.75	0.82	0.78	39999
1	0.80	0.73	0.76	40001
accuracy			0.77	80000
macro avg	0.78	0.77	0.77	80000
weighted avg	0.78	0.77	0.77	80000

6) Predictions

```
train = pd.DataFrame()
train['label'] = y_train
train['text'] = X_train

def predict_category(s, train=X_train, model=model):
    pred = model.predict([s])
    return pred[0]
```

```
predict_category("i wanna shot myself")
```

0

```
predict_category("i love you")
```

1

Activate V

CHAPTER 7

FUTURE SCOPE

The objective of this project was to detect Depression using sentiment Analysis. For this purpose, a dataset was collected from the Kaggle . It was pre-processed and clean to prepare it for further use. Then the dataset was split into 80%-20% partitions for Training and Testing respectively. Further, popular ML algorithms, Naive Bayes . The results obtained after experiments show that the Naive Bayes technique resulted in a classification accuracy of more than 84%.

As a future work in this research, other Machine Learning techniques and Deep Learning techniques can be applied for better prediction accuracy. Moreover, a dataset consisting of many instances can be collected and used for performing experiments.