**ABSTRACT**

To develop a machine learning model that can accurately classify sentiments expressed in social media posts as positive, negative, or neutral, and provide insights into public opinions on various topics.

In the age of digital communication, social media platforms serve as critical channels for public expression, making sentiment analysis a vital tool for understanding user opinions and behaviours. This project aims to implement a machine learning-based sentiment analysis Model that can effectively classify sentiments from social media data into positive, negative, or neutral categories. The methodology involves several key stages: data collection from platforms like Twitter and Facebook, preprocessing to clean and prepare the text data, and feature extraction using techniques such as TfidfVectorizer. The project will employ machine learning algorithm like LogisticRegression Performance evaluation will be conducted using metrics such as accuracy, precision, recall, and F1-score to ensure the effectiveness of the model. By leveraging a large dataset of social media posts, the system aims to provide real-time sentiment analysis capabilities that can be applied in multiple domains, including brand monitoring and public opinion tracking. The anticipated outcomes include not only high classification accuracy but also insights into user behaviour trends over time. This project highlights the potential of machine learning to enhance sentiment analysis processes, addressing challenges such as informal language and contextual variability inherent in social media communications. Future work may explore further optimizations and the integration of multimodal data sources to improve sentiment classification accuracy.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER** | **TITLE** | **PAGE NO** |
| **I** | **INTRODUCTION**  1.1About the project | **01-10** |
| **II** | **SYSTEM ANALYSIS**      2.1 Existing System  2.2 Proposed System  2.3 Hardware and Software  Requirements |  |
| **III** | **SYSTEM DESIGN**  3.1 Module Description |  |
| **IV** | **SYSTEM IMPLEMENTATION**      4.1 Language Selection  4.2 Sample code  4.3 Screenshots |  |
| **V** | **CONCLUSION AND FUTURE**  **ENCHANCEMENTS**    5.1 Conclusion  5.2 Future Enhancements  5.3 References |  |

**CHAPTER - I**

**INTRODUCTION**

Sentiment analysis, also known as opinion mining, is a process in natural language processing (NLP) used to identify and quantify emotions, opinions, and attitudes within text data. It has become an essential tool for understanding the vast amount of unstructured data generated daily, especially on social media, product reviews, forums, and other online platforms. Sentiment analysis aims to determine whether a piece of text expresses a positive, negative, or neutral sentiment.

**What is Sentiment Analysis?**

Sentiment analysis works by leveraging machine learning techniques and language processing models to detect and classify sentiment within text automatically. By training models on annotated datasets, these algorithms can learn to recognize and predict the sentiment of unseen data based on word patterns, tone, and context. This analysis can be applied on various levels:

**Types of Sentiment Analysis:**

1. Binary Sentiment Analysis: Classifying text as positive or negative.

2. Multi-Class Sentiment Analysis: Classifying text into multiple sentiment categories (e.g., happy, sad, angry).

3. Aspect-Based Sentiment Analysis: Identifying sentiments towards specific aspects or features.

4. Emotion Detection: Identifying emotions such as happiness, sadness, fear, or anger.

**1.1 About the project:**

The primary objective of this project is to create a model capable of accurately categorizing sentiments as positive, negative, or neutral based on social media interactions. By employing machine learning algorithm, the system will analyse vast amounts of data from platforms like Twitter, Facebook, and Instagram, allowing organizations to gauge audience reactions to their products, services, or campaigns.

**Project Deliverables:**

1. A trained machine learning model for sentiment analysis

2. A report on model performance and evaluation metrics

3. A presentation summarizing the project findings

4. Code repository with implementation details

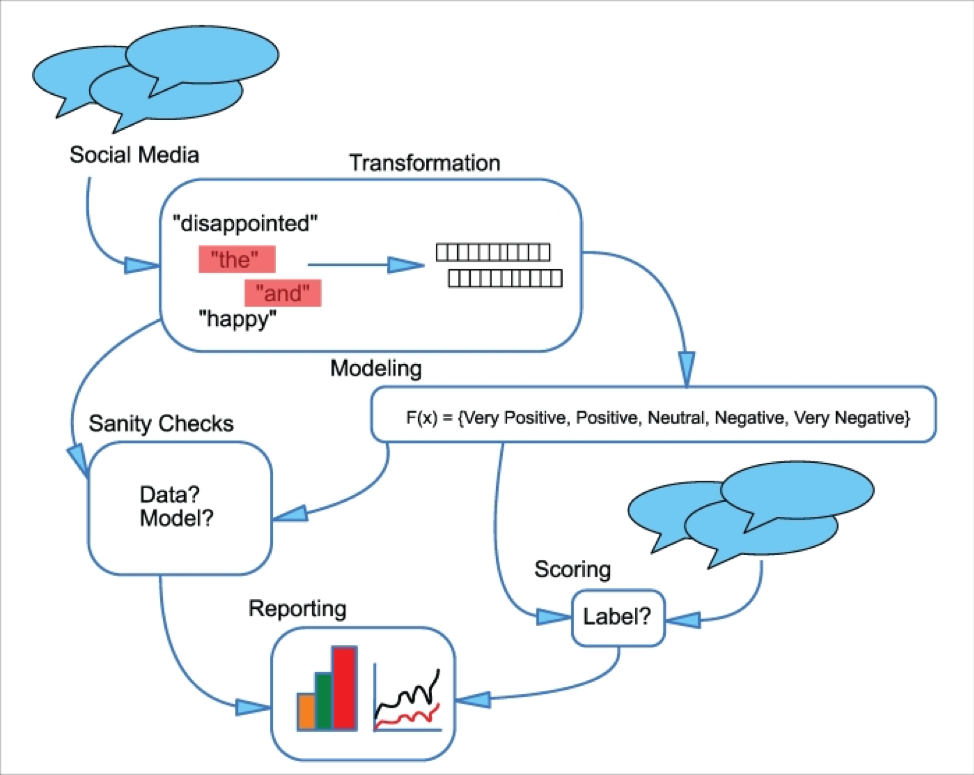


Figure:1.1Social media sentiment analysis model

**Methodology**

**Data Collection**: The project will begin with gathering a diverse dataset from social media platforms. This dataset will include posts, comments, and user interactions relevant to specific brands or topics.

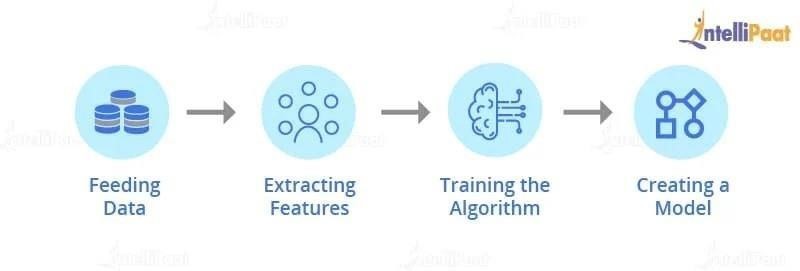
**Data Preprocessing**: Collected data will undergo preprocessing to remove noise and irrelevant information. This includes tokenization, removing stop words, and normalizing text for better analysis.

**Feature Extraction**: Techniques such as Term Frequency-Inverse Document Frequency (TF-ID) (e.g., TfidfVectorizer) will be utilized to convert text data into numerical representations suitable for machine learning models.

**Model Development**: A machine learning algorithm will be tested, Like LogisticRegression model will be trained on labeled datasets to learn how to classify sentiments accurately.

**Evaluation**: The performance of the models will be assessed using metrics such as accuracy, precision, recall, and F1-score. This evaluation will help determine the effectiveness of the sentiment analysis system.

**1.2 How does a Machine Learning system work for Sentiment analysis in Social media?**

The below picture shows the basic structure of the working of fraud detection algorithms using Machine Learning:

**1.2.1 Feeding Data:**

The data is fed into First the model. The accuracy of the model depends on the amount of data on which it is trained, the more data the better the model performs. For Predicting the sentiment, we are giving in your social media platforms

**1.2.2 Extracting Features:**

Feature extraction basically works on extracting the information of each and every tweet, post and comment.

**Content:** This parameter is used to check the content user posted

**1.2.3 Training the Algorithm:**

Once you have created a model, you need to train it by providing User Data with data so that the sentimental analysis model learns how to distinguish between ‘Positive’ and ‘negative’ statement.

**1.2.4 Creating a Model:**

Once you have trained your sentimental analysis model on a specific dataset, you are ready with a model that works for detecting ‘sentiment’ in user’s posts.

**CHAPTER-II**

**SYSTEM ANALYSIS**

**2.1 Existing System**

Before machine learning models came into existence for the social media sentiment analysis social media developers and researchers used various rule-based and manual approaches to categorize sentiment in text data. Here are some methods used in the past:

* **Manual Annotation**: Human annotators labeled text data as positive, negative, or neutral based on predefined guidelines.
* **Rule-Based Systems**: Developers created hand-coded rules using linguistic patterns, keywords, and phrases to identify sentiment. For example:
  + Keyword matching: "good," "bad," "happy," etc.
  + Phrase patterns: "I love," "I hate," etc.

**Disadvantages:**

* It consumes lot of time to check the sentiment manually.
* The great majority of Posts are legitimate which makes the sentiment detection difficult.
* Lack of Accuracy

**2.2 Proposed System**

The aim of the proposed system is to develop the capable model to detect the sentiment before a user post a content in social media platform using machine learning algorithms such as Logistic regression. which detects the sentiment in an efficient manner The proposed system tries to detect sentiment before the content posted in social media platforms . This project has been developed using Python language using jupyter notebook.

**Advantages:**

* This proposed system overcomes all the disadvantages of existing system
* It learns from the patterns and thus distinguishes between ‘Positive’ and ‘Negative’ sentiment .
* The main advantage is easy and quickly detecting sentiment by analysing the already trained model.

**2.3 Hardware and software Requirements:**

**Software Requirements**

* Operating System: Windows 10 & above.
* IDE: Jupyter Notebook
* Libraries Used: NumPy, Pandas, scikit learn
* Technology: Python

**Hardware Requirements**

* Processor: Core I3
* Hard Disk: 160GB
* RAM: 4Gb & Above
* Input Device: Standard Keyboard, Laptop/pc

**CHAPTER-III**

**SYSTEM DESIGN**

**3.1 Module Description**

This project has the following Modules.

* + 1. Data Collection and importing packages and Libraries
    2. Data Exploration
    3. Applying the Algorithms on data set
    4. Manual analysis on algorithms

**3.1. 1. Data Collection and importing Packages and Libraries:**

1. Data Collection:

Initially, Dataset named **sentimental\_analysis.csv** file is downloaded from the Kaggle Website and upload that data set into Jupyter Notebook tool by importing the panda’s library. Import all the required libraries and packages into Jupyter Notebook IDE to run the project.

Dataset Link: https://www.kaggle.com/datasets/mlg[-ulb/creditcardfraud](http://www.kaggle.com/datasets/mlg-ulb/creditcardfraud)

1. **Data Exploration:**

* Data exploration is the first step in data analysis, during which data analysts utilize data visualization and statistical tools to characterize data set characterizations like size, amount, and correctness to gain a better understanding of the data.
* Data is represented in various ways like Correlation Matrices, graphs, etc.
* Representing the data set and total number of Positive, Negative and Neutral Contents posts.

**3. Applying the Algorithm on Sentimental\_analysis Dataset:**

**1.Logistic Regression:**

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic Regression is a supervised learning technique that is used when the decision is categorical. It means that the result will be either ‘Positive or ‘Negative’ or ‘Neutral ’ in the given Dataset.

**Data Exploration:**

The dataset was retrieved from an open-source website, Kaggle.com. it contains a set of Data of

Social media platforms posting contents from various countries and various days in different times with the content posted and its sentiment. The dataset consists of 7 attributes, 499 rows. Among 7 attributes 3 are integer(numeric) type, and remaining four attributes are object type, In the Dataset “Sentiment” which contains binary variables where “1” is a case of Positive text content, and “0” is for negative text content, and “4” for neutral text content . And in this Dataset there is a platform which contains from which Platform the post is posted

**Split Data set:**

After uploading the data set into Jupyter Notebook IDE, In machine learning, Train Test Split activity is done to measure the performance of the machine learning algorithm when they are used to predict the new data which is not used to train the model. You can use the train\_ test split () method available in the SK\_learn library to split the data into train test sets.

Usually we split Dataset and dedicate 20% of Dataset for testing model and 80% of Dataset is dedicated for training the model

**1. Train Data:**

A subset of data set to train in the machine learning model, and we already know the output.

**2. Test Data:**

A subset of data set to test the machine learning model, and by using the test set, model predicts.

**CHAPTER-IV**

**SYSTEM IMPLEMENTATION**

**4.1 Language Description:**

**4.1.1 The Python Programming Language:**

Python is a set of instructions that we give in the form of a Program to our computer to perform any specific task. It is a Programming language having properties like it is interpreted, object-oriented and it is high-level too. Due to its beginner-friendly syntax, it became a clear choice for beginners to start their programming journey. The major focus behind creating it is making it easier for developers to read and understand, also reducing the lines of code

**Python Features**

* Interpreted Language
* Object-Oriented Language
* Open-Source Language
* Dynamic Memory Allocation

**4.1.2 Python Packages**

Python has numerous packages, libraries, and frameworks that make development easier. Here's a comprehensive list of popular Python packages:

1. NumPy (numerical computing)

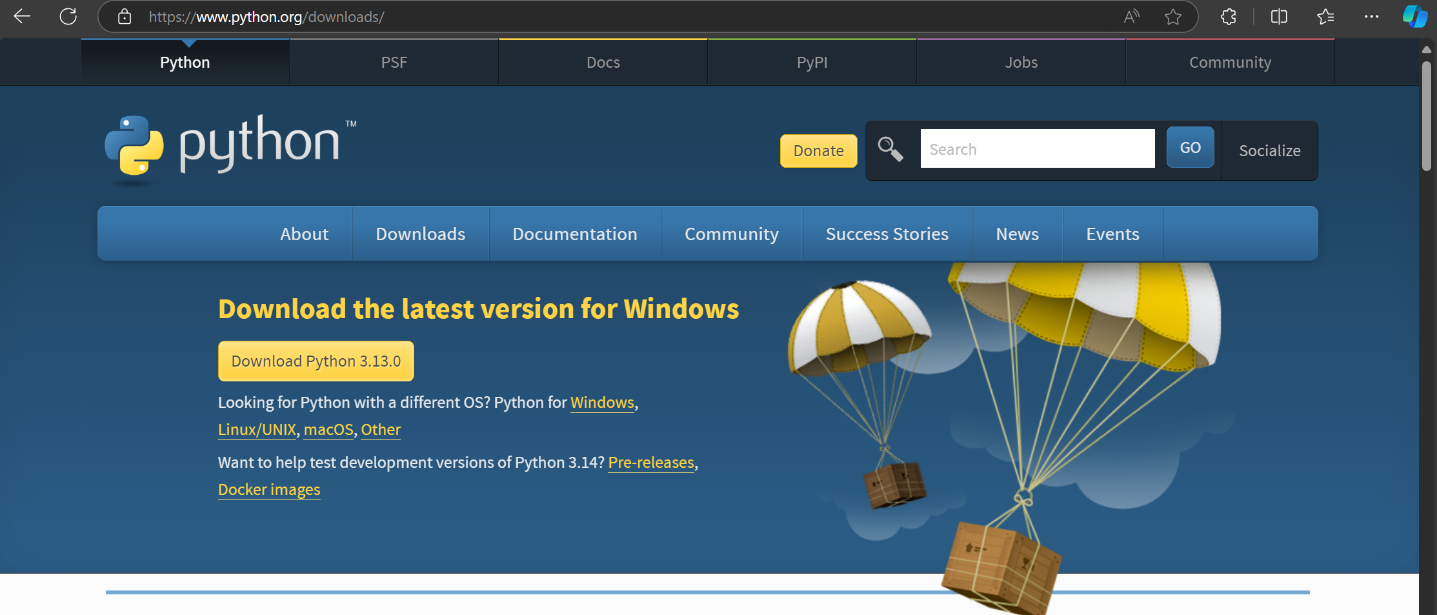
2. Pandas (data manipulation and analysis)

3. Matplotlib (data visualization)

4. Scikit-learn (machine learning).

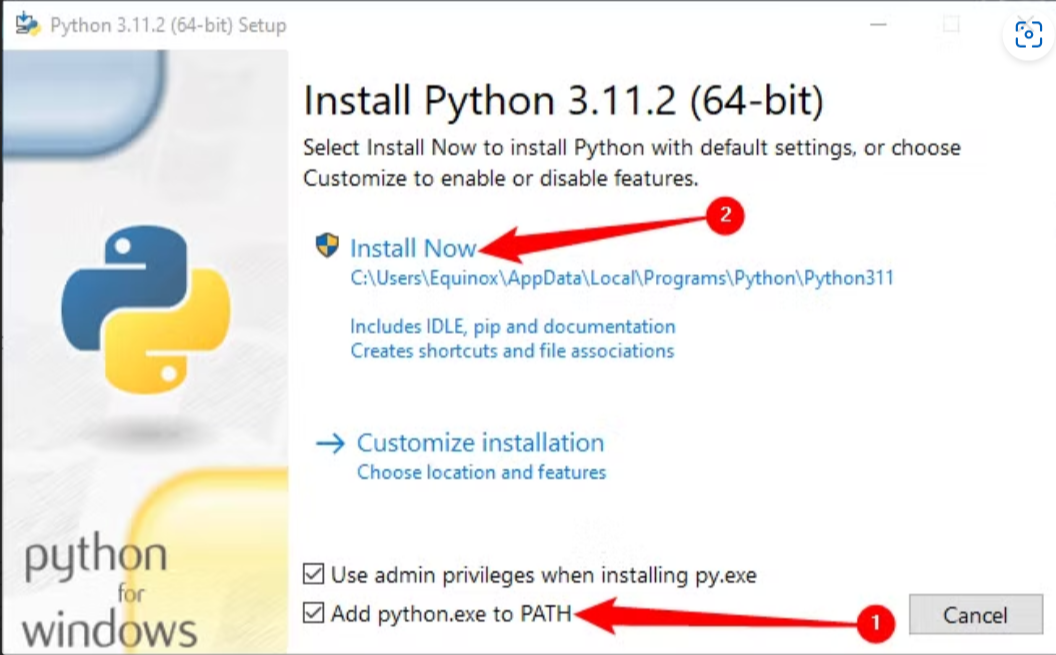
**4.1.2 Steps to implement the Project**

**Installation of python**

****In general, you should just download and install the latest version of Python. You'll see a big banner at the top indicating the latest version of Python on the main download page. Click "Download Python 3.x.x."

First, download the latest version of Python 3 if you haven't already, then run the executable.

On the first screen, enable the "Add Python.exe PATH" option and then click "Install Now."



Now check whether the python install correctly by navigate into command prompt and type

* + - **>>python –version**

If the downloaded version is displayed in the command prompt then we can understand that the python file installed successfully

**Installing jupyter notebook without using anaconda**

In this article, We will cover how to install Jupyter Notebook without Anaconda on Windows

Jupyter Notebook is a free, open-source, and interactive web application that allows us to create and share documents containing live code, equations, visualizations, and narrative text. Some of the most important uses of Jupyter Notebook are data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more interesting.

* To Install the Jupyter, the command is as given below:
  + - **>> python -m pip install jupyter notebook**

During the installation of the jupyter notebook it automatically downloads and install all the required modules and packages for Data science

* After the installation begins you will see this:

**Launching jupyter notebook after installing:**

To launch the Jupyter you need to write the following command**>> jupyter notebook**

after writing the above command in your command prompt, you will see the server is initialised in the particular folder in which you initiate the serve and opens workspace in browser.

* Now To create a Notebook where you will store and run your code you just need to make a click on "**New**" on the top on the right side; there you have four options: Python 3(notebook), Text File, Folder, and Terminal. Just click on **Python 3** and start your coding.

The notebook is always saved with (**.ipynb**) extension.

**4.1.3Required Libraries for implementing our model**

**NumPy**

NumPy (Numerical Python) is a library for working with arrays and mathematical operations in Python. It provides support for large, multi-dimensional arrays and matrices, and is the foundation of most scientific computing in Python. There are some key features of this library are like Multi-dimensional arrays, Vectorized operations, Matrix operations, Random number generation, Linear algebra functions, Statistical function,. Integration with other libraries (e.g., Pandas, Matplotlib)

Mostly used in scientific computing, Data analysis, Machin learning, Signal processing, image processing etc.. Basic Data types of NumPy are ndarray(N\_Dimensional array).

**Sklearn**

Scikit-learn (sklearn) is an open-source machine learning library for Python, providing simple and efficient tools for data analysis, classification, regression, clustering, and more.

Sklearn is widely used for classification, regression, clustering, dimensionality reduction and preprocessing.

**Pandas**

Pandas is a powerful open-source library in Python for data manipulation and analysis. It provides data structures and functions to efficiently handle structured data, including tabular data such as spreadsheets and SQL tables and useful for handling Datasets.

**NLTk**

NLTK (Natural Language Toolkit) is a popular open-source library used in machine learning and natural language processing (NLP) tasks. It provides tools and resources for tokenization, stemming, lemmatization, parsing, and semantic reasoning. Applications are Sentiment Analysis,

Text Classification, Information Retrieval, Question Answering, Machine Translation, Text Summarization

**4.2 Source Code**

# Importing Dependencies

import math

Import pandas as pd

Import numpy as np

Import re

From nltk.corpus import stopwords

From nltk.stem.porter import PortStemmer

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report

import seaborn as sns

# Loading the dataset to a Pandas\_DataFrame

Sanalysis=pd.read\_csv(r"sentiment\_analysis.csv")

# printing shape of DataFrame

Sanalysis.shape()

# First 5 rows of the dataset

Sanalysis.head())

# Last 5 rows of the dataset

Sanalysis.tail()

# Dataset information

Sanalysis.info()

# Description of dataset

Sanalysis.describe()

# Checking the number of missing values in each column

Sanalysis.isnull().sum()

#ploting a heat map to check if there are any null values in dataset

sns.heatmap(Sanalysis.isnull(),yticklabels=False,cbar=False,cmap='viridis')

# Checking for records in dataset

Sanalysis.duplicate()

# Checking distribution of positive and negative and neutral values.

Sanalysis[‘sentiment’].value\_counts()

# Replacing the Neutral with Negative

sentiment.replace({'sentiment':{'neutral':'negative'}},inplace=True)

# Checking distribution of positive and negative

Sanalysis.['sentiment'].value\_counts()

# Replacing negative=0 and positive=1

Sanalysis.replace({'sentiment':{'negative':0}},inplace=True)

Sanalysis.replace({'sentiment':{'positive':1}},inplace=True)

#checking the info of dataset after replacement for datatype

Sanalysis.info()

#Ploting a graph for better visual of sentiment column

sns.countplot(x=Sanalysis ["sentiment"])

# Ploting a graph for better visual of sentiment column and years

sns.countplot(x='sentiment',hue='Year',data=Sanalysis)

# Ploting a graph for better visual of sentiment column and Platform

sns.countplot(x='sentiment',hue=’Platform’,data=Sanalysis)

# Droping unnecessary column for better results

Sanalysis=sentiment.drop('Time of Tweet',axis=1)

# Print first 5 line of dataset after drop a column

Sanalysis.head()

#performing stemming operation for splitting each word and remove garbage characters from the Dataset

port\_stem=PorterStemmer()

def stemming(content):

stemmed\_content=re.sub('[^a-zA-Z]',' ',content)

stemmed\_content=stemmed\_content.lower()

stemmed\_content=stemmed\_content.split()

stemmed\_content=[port\_stem.stem(word) for word stemmed\_content if not

wordstopwords.words('english')]

stemmed\_content=' '.join(stemmed\_content)

return stemmed\_content

Sanalysis['stemmed\_content']=Sanalysis['text'].apply(stemming)

#Printing First 5 Lines on data set after Performing Portstemmer

Portstemmer

Sanalysis.head()

#Printing only stemmed content

print(Sanalysis ['stemmed\_content'])

#Printing only the target of the content

print(Sanalysis ['sentiment'])

#Splitting the features and target

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,stratify=y,random\_state=2)

print(x.shape,y\_train.shape,x\_test.shape)

# Printing training dataset

print(x\_train)

# Printing training testing

print(x\_test)

# Encoding the text data into float type using vectorizer and printing x\_train data

vectorizer=TfidfVectorizer()

x\_train=vectorizer.fit\_transform(x\_train)

x\_test=vectorizer.transform(x\_test)

print(x\_train)

# Printing x\_test data after encoding

Print(x\_test)

# Training the Logistic Regression Model with Training

model=LogisticRegression(max\_iter=10000)

model.fit(x\_train,y\_train)

# Printing accuracy score on training data

x\_train\_prediction=model.predict(x\_train)

training\_data\_accuracy=accuracy\_score(y\_train,x\_train\_prediction)

print('Accuracy score on the training data:',training\_data\_accuracy)

training\_data\_accuracy=(math.ceil(training\_data\_accuracy\*100))

print(f'Accuracy score on training data is: {training\_data\_accuracy}%')

# Printing accuracy score on testing data

x\_test\_prediction=model.predict(x\_test)

test\_data\_accuracy=accuracy\_score(y\_test,x\_test\_prediction)

print(f'Accuracy score on the testing data: {math.ceil(test\_data\_accuracy\*100)}%')

# Printing confusion metrics

print(classification\_report(y\_test,x\_test\_prediction))

# To Save the trained model as csv type and import pickle library and saving model

import pickle

filename='trained\_model.csv'

pickle.dump(model,open(filename,'wb'))

# Evaluating the saved model by programmer

loaded\_model=pickle.load(open('trained\_model.csv','rb'))

x\_new=x\_test[1]

print(y\_test[1])

prediction = loaded\_model.predict(x\_new)

print(prediction)

if (prediction[0]==0):

print('The tweet is negative')

elif (prediction[0]==1):

print('The tweet is positive')

else:

print('the tweet is neutral')

# second time evaluating

loaded\_model=pickle.load(open('trained\_model.csv','rb'))

x\_new=x\_test[3]

print(y\_test[3])

prediction = loaded\_model.predict(x\_new)

print(prediction)

if (prediction[0]==0):

print('The tweet is negative')

elif (prediction[0]==1):

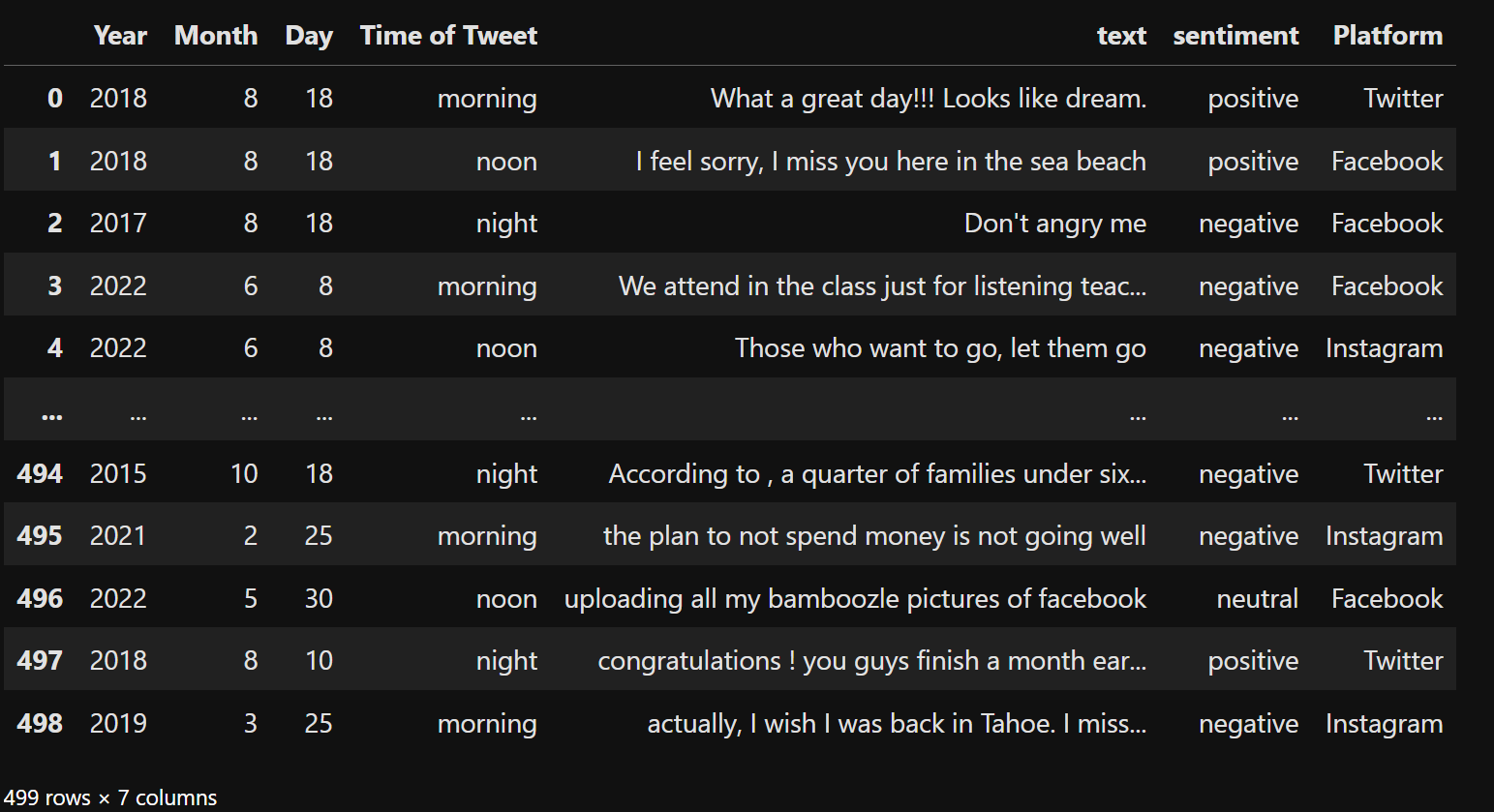
print('The tweet is positive')

else:

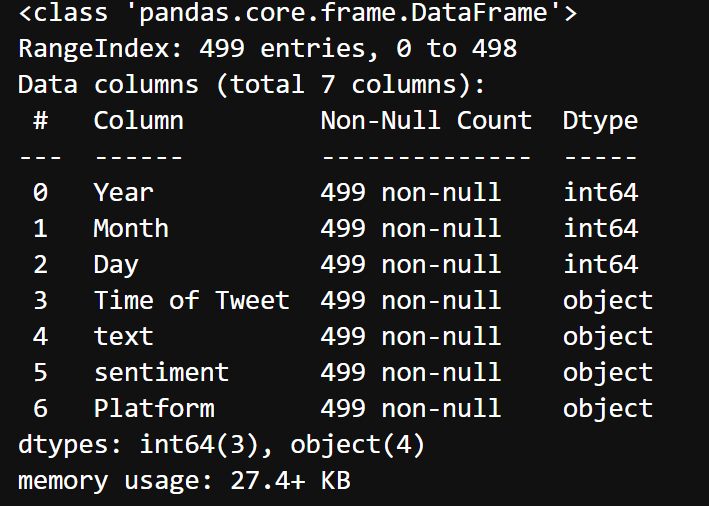
print('the tweet is neutral')

**4.3 SCREENSHOTS**

**Dataset figure:**

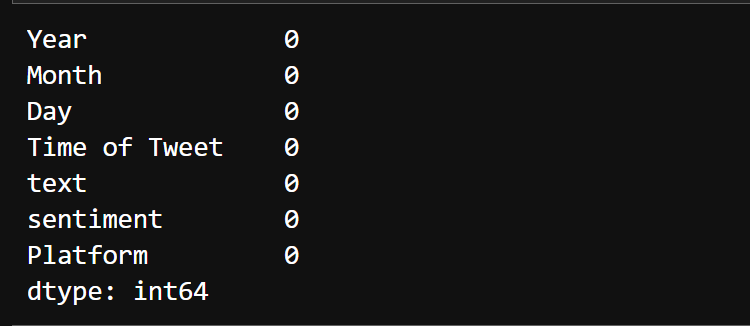


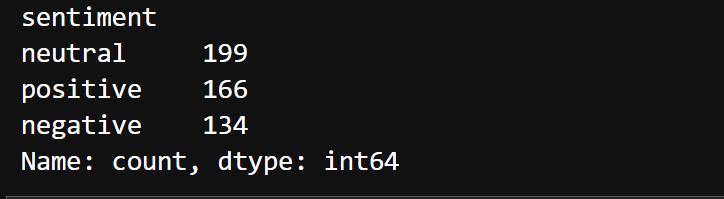
**Information of Dataset**

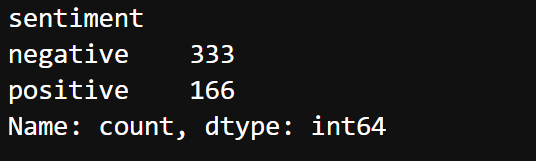
****

 **Description of the dataset:**

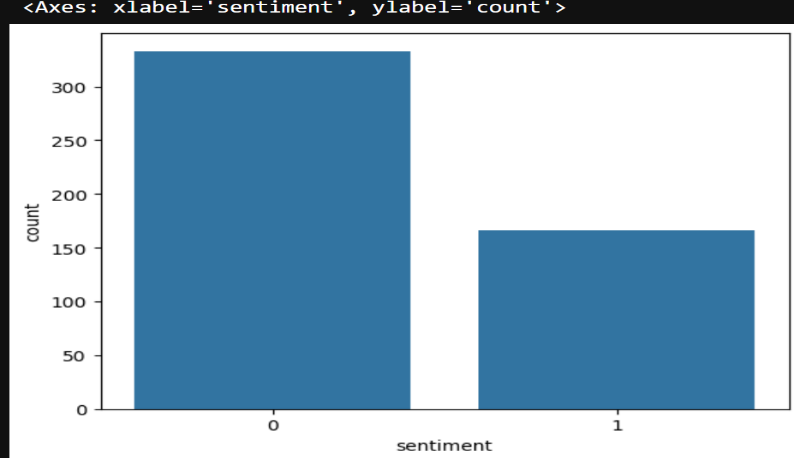
**Missing values of the dataset:**

**Categorisation of Positive and Negative :**

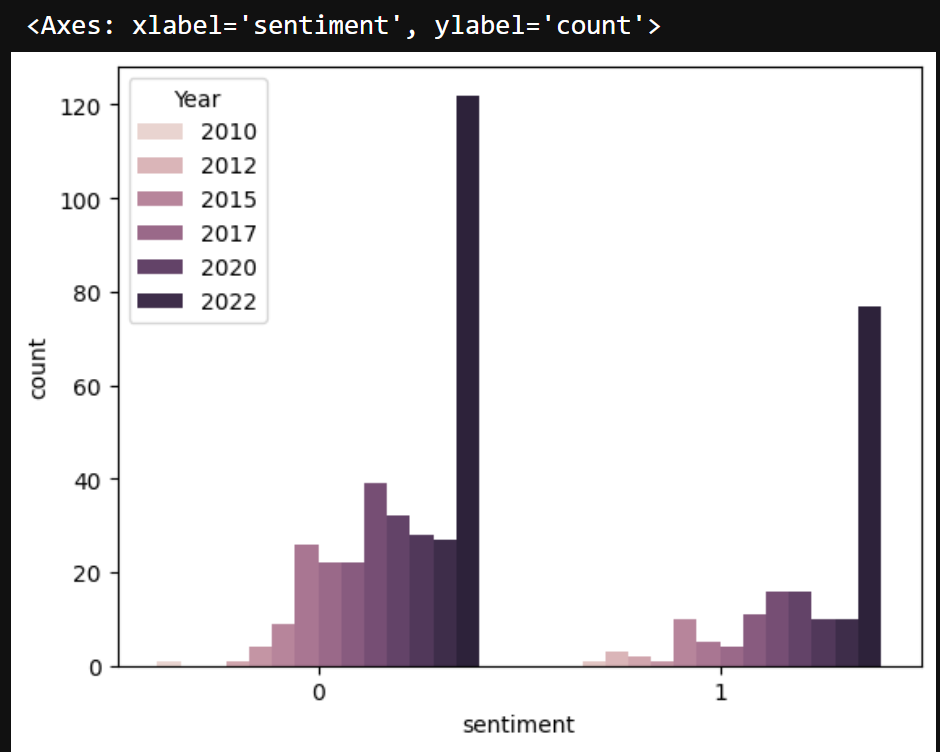




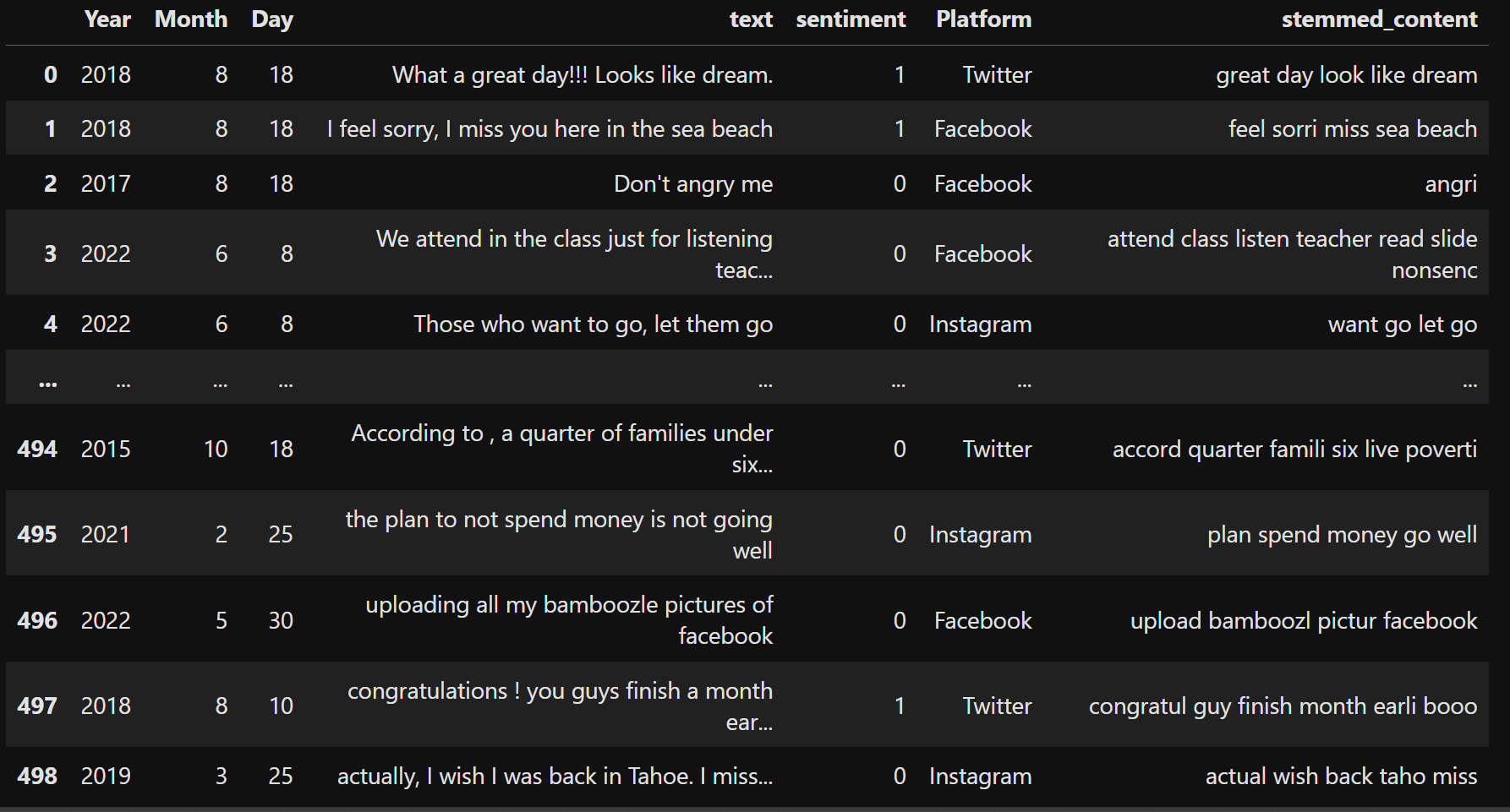
**Plotting tweets Data :**

****

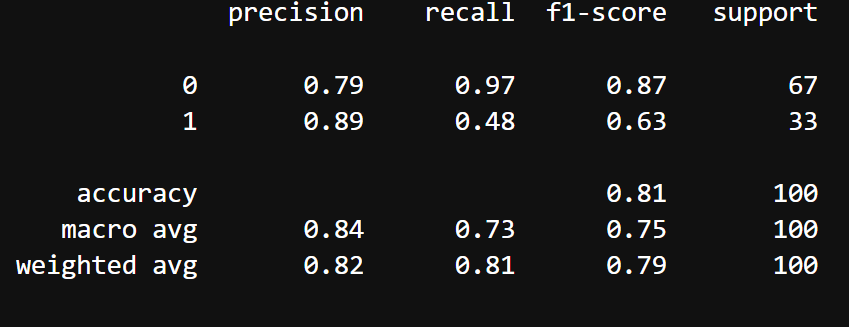
**Plotting sentiment with year column:**

****

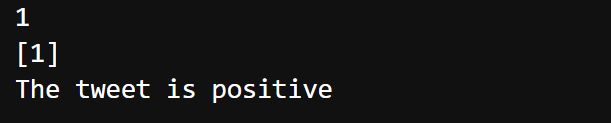
**After stemming:**

****

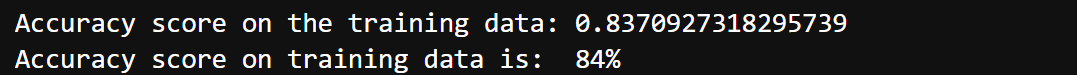
**Confusion Matrix:**



**Output after training the model**

****

**Accuracy on Training and Testing date:**

****

**CHAPTER-V**

**CONCLUSION & FUTURE ENCHANCEMENTS**

**5.1 CONCLUSION:**

Sentiment analysis on social media has emerged as a vital tool for understanding public opinion and user behaviour. Recent advancements in machine learning (ML) and deep learning techniques have significantly enhanced the accuracy and efficiency of sentiment classification, making it applicable across various sectors, including marketing, policymaking, and social research.

Sentiment analysis in social media using machine learning has proven to be an effective tool for understanding public opinion, sentiment, and emotions, with applications in brand monitoring, customer service, market research, and political campaign analysis. To further enhance this system, technical advancements such as deep learning architectures, transfer learning, multimodal analysis, and real-time processing can be implemented. Additionally, linguistic enhancements including sarcasm and irony detection, emotion detection, multilingual support, and domain adaptation can improve accuracy. Data enhancements such as data augmentation, quality improvement, social media platform expansion, and historical data analysis can also contribute to better outcomes. Application enhancements like visualization tools, alert systems, integration with CRM systems, and predictive analytics can make the system more practical. Future research directions include exploring explainable AI, addressing adversarial attacks, incorporating human-in-the-loop approaches, and addressing ethical concerns. By implementing these features, sentiment analysis can become more accurate, efficient, and effective, providing valuable insights for businesses, organizations, and researchers. Furthermore, integrating natural language processing (NLP) techniques, named entity recognition (NER), and part-of-speech (POS) tagging can refine sentiment detection. Utilizing cloud-based infrastructure and edge computing can also optimize performance and scalability.

**5.2 FUTURE ENHANCEMENT:**

To improve sentiment analysis systems further, several feature enhancements can be considered:

1. **Advanced Data Preprocessing:** Implementing sophisticated preprocessing techniques to clean and structure data effectively can significantly enhance model performance. This includes noise reduction methods to filter out irrelevant data.
2. **Innovative Feature Extraction:** Utilizing modern feature extraction methods such as TF-IDF (Term Frequency-Inverse Document Frequency), Word2Vec, and GloVe can help capture the semantic meaning of words more effectively. These techniques allow models to understand context better, which is critical for accurate sentiment classification.
3. **Real-Time Analysis Capabilities**: Developing systems capable of real-time sentiment analysis will enable businesses and researchers to respond dynamically to public sentiment changes. This requires efficient algorithms that can process large volumes of data quickly.
4. **Handling Multilingual Data**: As social media is a global platform, enhancing models to handle multiple languages and dialects will broaden their applicability. This includes training models on diverse datasets that reflect various linguistic nuances.
5. **Incorporating Emotion Recognition**: Beyond basic sentiment classification, integrating emotion detection capabilities can provide deeper insights into user feelings. This involves recognizing emotional tones conveyed through emojis, GIFs, or specific word choices.
6. **Model Optimization**: Continuous tuning of hyperparameters and exploring advanced configurations of deep learning frameworks can lead to improved model performance. Utilizing tools like Scikit-learn for hyperparameter optimization is recommended.
7. **Predictive Analytics**: Developing predictive models that forecast trends based on historical sentiment data can provide valuable insights for strategic decision-making in various fields.

By addressing these areas for enhancement, future sentiment analysis frameworks can become more robust and effective in capturing the complexities of human emotions expressed on social media platforms.

**5.3 REFERENCES**

* "Sentiment Analysis of Social Media Text" by A. Pak and P. Paroubek (2010)
* "Machine Learning for Sentiment Analysis of Twitter Data" by K. Zhang et al. (2011)
* "Social Media Analytics: Analysis and Visualization of Social Media Data" by D. Hutchison (2015)
* "Social Media Mining: An Introduction" by R. Zafarani et al. (2014)
* Kaggle Sentiment Analysis model
* From Geeks for Geeks web site