NATURAL LANGUAGE PROCESSING

(Twitter Sentiment Analysis)

Summer Internship Report Submitted in partial fulfilment
of the requirement for an undergraduate degree of

Bachelor of Technology

In

COMPUTER SCIENCE AND ENGINEERING

By

AKS VIBHAKAR
HU21CSEN0101156

Under the Guidance of

Dr. G. Rathnamma
Assistant Professor

GITAM
(DEEMED TO BE UNIVERSITY)
VISAKHAPATNAM ✻ HYDERABAD ✻ BENGALURU

Department Of Computer Science and Engineering
GITAM School of Technology
GITAM (Deemed to be University)
Hyderabad-502329
December 2023

DECLARATION

I submit this industrial training work entitled **“Twitter Sentiment Analysis**” to GITAM (Deemed To Be University), Hyderabad in partial fulfilment of the requirements for the award of the degree of “**Bachelor of Technology**” in “**Computer Science and Engineering**”. I declare that it was carried out independently by me under the guidance of **Dr. G. Rathnamma,** Asst. Professor, GITAM (Deemed To Be University), Hyderabad, India.

The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma.

Place: HYDERABAD

Name: AKS VIBHAKAR

Date: 20-12-2023

Student Roll No:
HU21CSEN0101156

GITAM (DEEMED TO BE UNIVERSITY)

Hyderabad-502329, India

Dated: 20 -12-2023

CERTIFICATE

This is to certify that the Industrial Training Report entitled **“Twitter Sentiment Analysis”** is being submitted by AKS VIBHAKAR (HU21CSEN0101156) in partial fulfilment of the requirement for the award of **Bachelor of Technology in Computer Science And Engineering** at GITAM (Deemed to Be University), Hyderabad during the academic year 2023-2024.

It is faithful record work carried out by him at the **Computer Science And Engineering Department**, GITAM University Hyderabad Campus under my guidance and supervision.

Dr. G. Rathnamma

Assistant Professor

Department of CSE

Dr. Mahaboob Basha Shaik

Professor and HOD

Department of CSE

ACKNOWLEDGEMENT

Apart from my effort, the success of this internship largely depends on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have helped me in the successful competition of this internship.

I would like to thank respected **Dr.D.Sambasiva Rao**, Pro Vice Chancellor, GITAM Hyderabad and **Dr. N. Seetharamaiah**, Principal, GITAM Hyderabad.

I would like to thank respected **Dr. Mahaboob Basha Shaik,** Head of the Computer Science and Engineering for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present an internship report. It helped me a lot to realize of what we study for.

I would like to thank the respected faculty **Dr.G.Rathnamma** who helped me to make this internship a successful accomplishment.

I would also like to thank my friends who helped me to make my work more organized and well-stacked till the end.

AKS VIBHAKAR

HU21CSEN0101156

ABSTRACT

Natural Language Processing (NLP) is a branch of artificial intelligence that deals with the interaction between humans and machines using natural language. NLP enables machines to understand, analyze, generate and manipulate natural language in various applications such as machine translation, sentiment analysis, text summarization, question answering and more. In this internship, I will explore the current state-of-the-art methods in NLP, and apply them to Twitter . I will also learn how to use different tools and frameworks for NLP, such as NLTK and more. My goal is to gain practical experience and skills in NLP, and contribute to the advancement of this field.

Twitter Sentiment Analysis is a challenging task that aims to extract the emotions of users from their tweets. In this internship, I propose to use Natural Language Processing (NLP) and Machine Learning (ML) techniques to build a robust and accurate system for this task. I will first collect and preprocess a large dataset of tweets labeled with positive and negative sentiment. Then, I will apply various NLP methods, such as Stemming, lemmatization, Stopword removal, Vectorization to transform the tweets into numerical features. Next, I will train and evaluate a logistic regression ML Model, to classify the tweets based on their sentiment. Finally, I will compare the performance of the models and analyze the results using metrics, such as accuracy score. The main goal of this internship is to develop a state-of-the-art system for Twitter Sentiment Analysis that can be used for various applications, such as marketing, customer service, social media monitoring, and public opinion mining..

Table of Contents

CHAPTER 1: Natural Language Processing

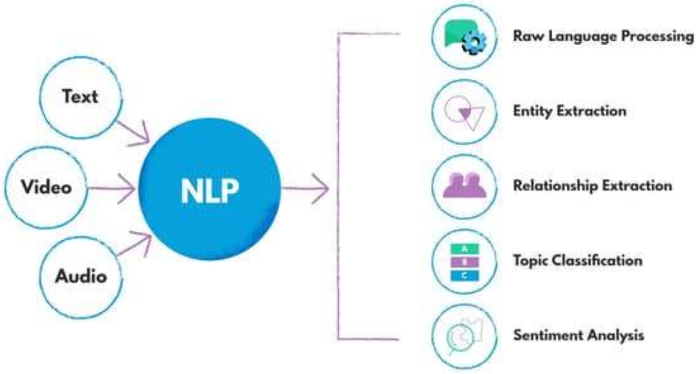
CHAPTER 2: Sentiment Analysis

CHAPTER 3: Twitter Sentiment Analysis

CHAPTER 1:
NATURAL LANGUAGE PROCESSING

1.1 INTRODUCTION:

Natural language processing (NLP) is a branch of artificial intelligence that deals with the interaction between computers and human languages. NLP aims to enable computers to understand, analyse, generate, and manipulate natural language texts or speech. Some of the applications of NLP include machine translation, speech recognition, sentiment analysis, text summarization, question answering, and chatbots.



1.2 IMPORTANCE OF NATURAL LANGUAGE PROCESSING :

In the past, computers were constrained to working exclusively with structured languages that demanded precision and lacked ambiguity. Programming a computer to execute tasks required explicit and clear instructions, limited to a specific set of commands that the computer comprehended. The syntax had to be flawless, and even end-users had to provide computers with precise commands – a memory many will recall when navigating a PC meant mastering common MS-DOS commands.

The advent of graphical user interfaces, such as Windows, partially overcame this barrier by allowing users to interact with computers through intuitive visual elements. Users could now point to files using a mouse, eliminating the need to memorize file names.

Natural Language Processing (NLP) emerges as a transformative technology poised to further alleviate the necessity for precision. Instead of humans adapting to the computer's language, NLP promises a paradigm shift where computers learn to comprehend and interpret human language, liberating users from the need to master specific commands.

1.3 USES OF NATURAL LANGUAGE PROCESSING :

Natural Language Processing (NLP) has become an invaluable tool across various domains, revolutionizing the way we interact with and extract insights from human language. several prominent uses of NLP are :

- **Information Retrieval and Search Engines**:
  - NLP enhances search engine capabilities by understanding user queries more effectively and delivering more accurate and relevant results.
- **Sentiment Analysis:**
  - Businesses use NLP to analyze social media and customer reviews to gauge public sentiment, helping them make informed decisions and improve products or services.
- **Chatbots and Virtual Assistants:**
  - NLP powers chatbots and virtual assistants, enabling more natural and context-aware interactions between users and machines. This is extensively used in customer support, enhancing user experience.
- **Language Translation:**
  - NLP is crucial for language translation services, making it possible for applications like Google Translate to provide real-time translations between multiple languages.
- **Text Summarization:**
  - NLP algorithms can automatically generate concise summaries of large volumes of text, aiding in information extraction and comprehension.
- **Speech Recognition:**
  - NLP is employed in speech recognition systems, allowing devices like smartphones and smart speakers to understand and respond to spoken commands.
- **Named Entity Recognition (NER):**

  - NLP is used to identify and classify entities such as names, locations, and organizations within a given text, facilitating information extraction and knowledge management.

- **Medical and Healthcare Applications:**
  - NLP is applied in healthcare for tasks like extracting information from medical records, assisting in diagnosis, and processing vast amounts of medical literature for research purposes.
- **Financial Analysis:**
  - NLP aids in processing financial news, reports, and market data, helping analysts make informed investment decisions by extracting relevant information and sentiments.
- **Legal Document Analysis:**
  - Legal professionals use NLP to sift through and analyze large volumes of legal documents, facilitating research, due diligence, and contract review processes.
- **Education and E-Learning:**
  - NLP is employed in educational technology to develop intelligent tutoring systems, assess student performance, and provide personalized learning experiences.
- **Content Generation:**
  - NLP can be used to automatically generate human-like text, assisting in content creation for various purposes, such as writing articles, generating product descriptions, and creating chatbot responses.

NLP Use Cases

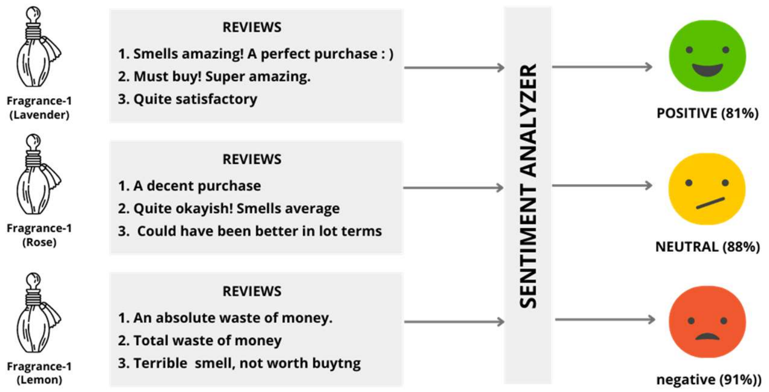
Now, Let us discuss about Sentiment Analysis in detail.

CHAPTER 2 :
SENTIMENT ANALYSIS

2.1 INTRODUCTION

With the explosion of user generated opinions there is the need by companies, politicians, service providers, social psychologists, researchers and other actors to analyse them in order to implement better decision choices.

Sentiment analysis, also known as opinion mining, is a subfield of Natural Language Processing (NLP) that focuses on determining the sentiment or emotional tone expressed in a piece of text. The goal of sentiment analysis is to understand and interpret the subjective information present in the text, whether it is positive, negative, or neutral.



.

2.2 IMPORTANCE OF SENTIMENT ANALYSIS

One of the main challenges of NLP is to deal with the ambiguity and variability of natural language. Different words and phrases can have different meanings and connotations depending on the context and the speaker's intention. For example, the word "great" can be used to express satisfaction, sarcasm, or irony. Similarly, the phrase "I'm fine" can mean different things depending on the tone of voice and the facial expression of the speaker. Therefore, it is not enough to analyze the literal meaning of the words, but also to capture the underlying sentiment and emotion behind them.

Sentiment analysis can help to overcome this challenge by providing a more comprehensive and nuanced representation of natural language. By detecting the sentiment and emotion of a text, we can better understand the purpose, intention, and attitude of the speaker or writer. We can also use this information to generate more appropriate and natural responses or feedback. For example, if we know that a customer is unhappy with a product, we can offer a sincere apology and a solution, rather than a generic thank you message. Conversely, if we know that a customer is satisfied with a product, we can express our gratitude and appreciation, rather than a bland confirmation.

Sentiment analysis can also help to improve the quality and reliability of natural language generation (NLG). NLG is the process of creating natural language texts from non-linguistic data, such as numbers, facts, or images. It is often used for tasks such as summarization, translation, captioning, and storytelling. However, generating natural language texts is not only about conveying information, but also about engaging and persuading the audience. Therefore, it is important to consider the sentiment and emotion of the target audience and tailor the tone and style of the generated texts accordingly.

For example, if we want to generate a summary of a news article, we should not only include the main facts and events, but also reflect the sentiment and emotion of the original article. If the article is positive and optimistic, we should use words and phrases that convey enthusiasm and hope. If the article is negative and pessimistic, we should use words and phrases that convey concern and caution. By doing so, we can create more faithful and coherent summaries that capture the essence and tone of the original article.

2.3 USES OF SENTIMENT ANALYSIS

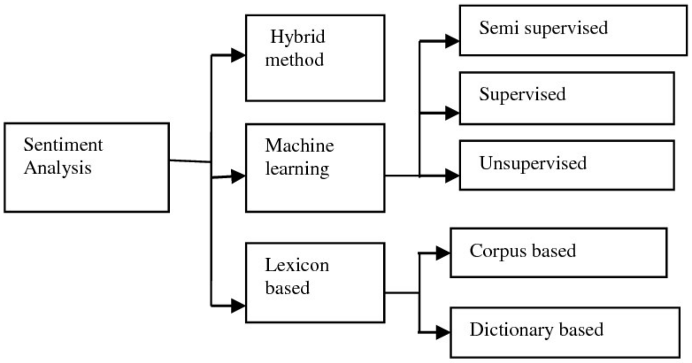
- **Understanding Customer Feedback**:
  - By analysing the sentiment of customer feedback, companies can identify areas where they need to improve their products or services.
- **Reputation Management**:
  - Sentiment analysis can help companies monitor their brand reputation online and quickly respond to negative comments or reviews.
- **Political Analysis**:
  - Sentiment analysis can help political campaigns understand public opinion and tailor their messaging accordingly.
- **Crisis Management:**
  - In the event of a crisis, sentiment analysis can help organizations monitor social media and news outlets for negative sentiment and respond appropriately.
- **Marketing Research:**
  - Sentiment analysis can help marketers understand consumer behavior and preferences, and develop targeted advertising campaigns.
- **Public Actions:**
  - Sentiment analysis also is used to monitor and analyse social phenomena, for the spotting of potentially dangerous situations and determining the general mood of the blogosphere

2.4 APPROACHES IN SENTIMENT ANALYSIS

- **Lexicon-Based Approach:**
    - Dictionary-Based Methods: Lexical approaches rely on sentiment lexicons or dictionaries containing words annotated with their associated sentiment polarity (positive, negative, or neutral). The sentiment of a text is determined by aggregating the sentiment scores of its constituent words.

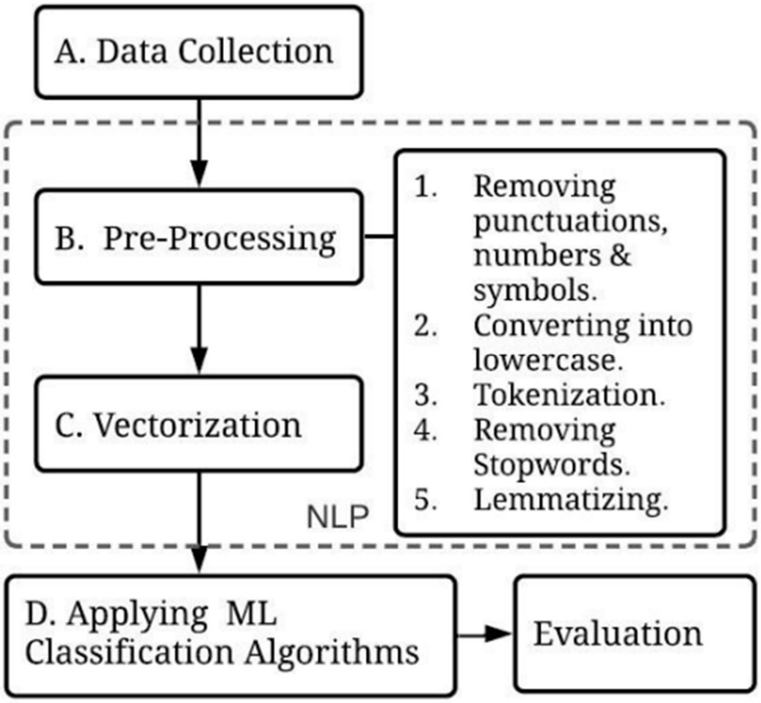
- **Machine Learning-Based Approach**:
    - This approach involves training a machine learning model on a labelled dataset where each text is associated with a sentiment label. Common algorithms include Support Vector Machines (SVM), Naive Bayes, and Logistic Regression.

- **Hybrid Approach :**
    - A hybrid approach in sentiment analysis seeks to amalgamate the advantages of both lexical analysis and machine learning, presenting a comprehensive solution that enhances accuracy and adaptability.
        - Combination of Lexical and Machine Learning:
            - This hybrid strategy involves the fusion of sentiment lexicons or dictionaries with machine learning models. Lexical methods, which assign sentiment scores based on pre-defined word sentiment polarities, work in tandem with machine learning algorithms that provide context-aware sentiment analysis. By leveraging the simplicity of lexical analysis and the contextual awareness of machine learning, this approach addresses the limitations of each method.



2.5 METHODOLOGY OF SENTIMENT ANALYSIS

`



2.5.1 Data Collection :

Data collection in sentiment analysis refers to the process of gathering relevant textual data from various sources to analyse and determine the sentiment expressed within the text.
In sentiment analysis, data collection is a pivotal step that involves gathering text data for training, validating, or testing sentiment analysis models. The choice of data sources, sampling methods, and annotation techniques significantly influences the performance of sentiment analysis algorithms. Sources such as social media platforms, review websites, and news articles provide diverse expressions of sentiments.

Data sampling techniques, including random and stratified sampling, ensure a representative dataset. Annotation involves manually or automatically labeling the data with sentiment classes. Multimodal data collection may include additional metadata for context. Ensuring a sufficient volume of diverse data from various sources enhances the model's adaptability.

Temporal considerations involve collecting time-stamped data to analyze temporal trends in sentiment. Ethical considerations, including privacy and bias, are crucial, and preprocessing steps such as text cleaning and normalization prepare the data for analysis. Effective data collection forms the cornerstone for developing sentiment analysis models capable of understanding and interpreting the nuances of sentiments in diverse textual data.

2.5.2 DATA PRE-PROCESSING :

Data preprocessing in sentiment analysis is a crucial step aimed at refining raw text data for effective analysis by machine learning models. It involves tasks such as text cleaning, which removes noise and irrelevant information, tokenization to break text into units, lowercasing for standardization, and stopword removal to eliminate common, less informative words. These actions collectively enhance the quality and relevance of the text data, setting the foundation for accurate sentiment analysis.

- **Text Cleaning:**
    - Clean the text data by removing irrelevant information, such as URLs, special characters, and emojis. This step helps in preparing the text for analysis.
- **Tokenization:**
    - Break down the text into individual words or tokens. This step is crucial for the subsequent analysis, as it helps in understanding the structure of the text.
- **Stopword Removal:**
    - Eliminate common words (stopwords) that do not carry much meaning. This helps reduce noise in the data and focuses on the more meaningful terms.
- **Stemming or Lemmatization:**
    - Reduce words to their root form to standardize the text. For example, "running" and "ran" might be reduced to the common root "run.

2.5.3 TRAIN-TEST SPLIT

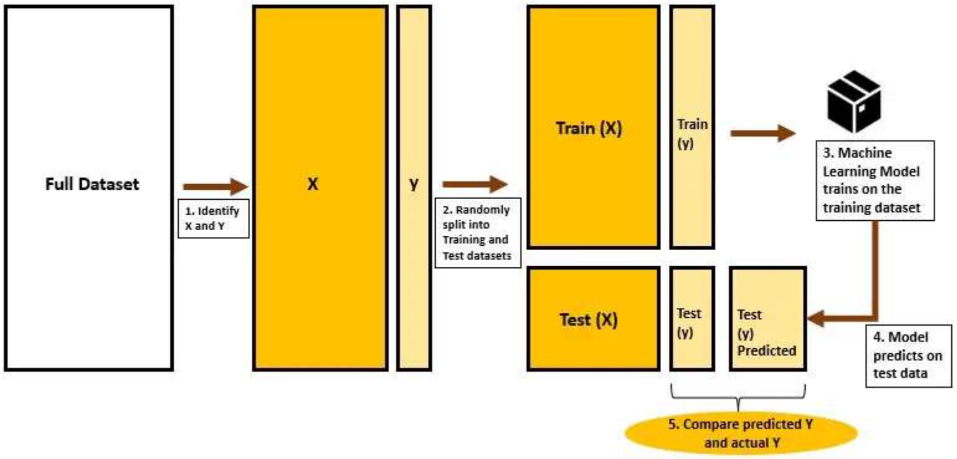
In sentiment analysis, the "train-test split" is a fundamental step in model development. This process involves dividing the labeled dataset into two subsets: one for training the model and the other for testing its performance. The training set is used to teach the model to recognize patterns and relationships between features and sentiments, while the test set assesses how well the model generalizes to new, unseen data. This practice helps ensure that the sentiment analysis model performs effectively on real-world data and provides reliable insights.

Training Set:

The training set is a portion of the dataset used to train or teach the sentiment analysis model. During this phase, the model learns patterns and relationships within the data to make predictions about sentiment.

Test Set:

The test set is a separate portion of the dataset that the model has not seen during training. It is used to evaluate the model's performance and assess how well it can generalize to new, unseen data.


Full Dataset
1. Identify X and Y
X
y
2. Randomly split into Training and Test datasets
Train (X)
Train (y)
Test (X)
Test (y)
Test (y) Predicted
3. Machine Learning Model trains on the training dataset
4. Model predicts on test data
5. Compare predicted Y and actual Y

2.5.4 TRAINING THE MACHINE LEARNING MODEL

Model training in sentiment analysis involves using a machine learning or deep learning algorithm to teach the model how to recognize patterns and relationships in the data that correspond to different sentiments (positive, negative, or neutral).
The steps involved are :

- **Text Vectorization:**

Convert the textual data into a numerical format that the model can understand.

- **Model Selection:**

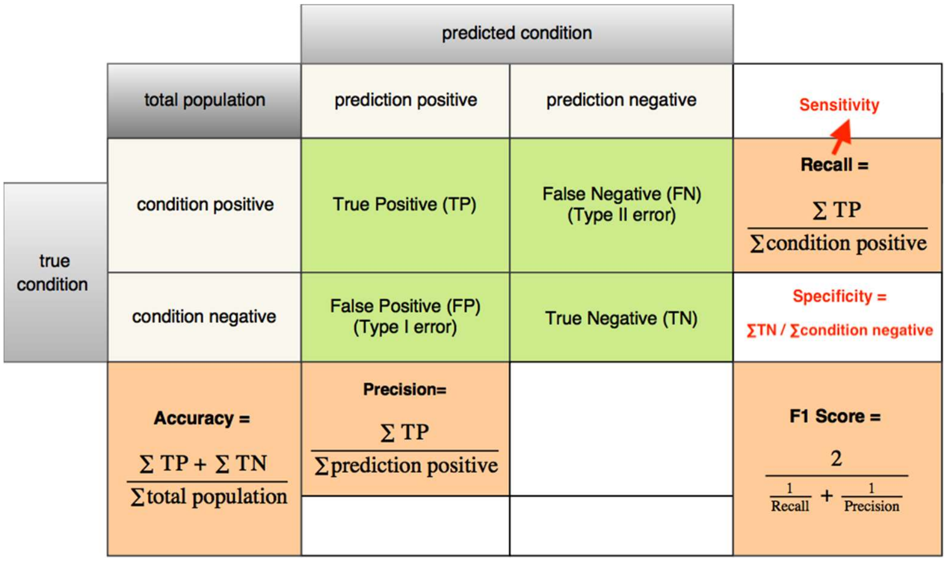
  Choose a suitable sentiment analysis model. Depending on the complexity of the task and the size of the dataset, E.g. : Logistic Regression

- **Training The Model :**

  Feeding the vectorized data to the Machine Learning Model.

2.5.5 EVALUATING THE MACHINE LEARNING MODEL

Evaluating a machine learning model is a critical phase in assessing its performance and generalization capabilities. In the context of sentiment analysis, this process involves measuring how accurately the model predicts sentiments on unseen data. Common evaluation metrics include accuracy, precision, recall, F1 score, and area under the Receiver Operating Characteristic (ROC) curve. A robust evaluation ensures the model's reliability and informs potential refinements for better sentiment analysis outcomes.



		predicted condition		
	total population	prediction positive	prediction negative	Sensitivity
true condition	condition positive	True Positive (TP)	False Negative (FN) (Type II error)	Recall = $\frac{\sum \text{TP}}{\sum \text{condition positive}}$
	condition negative	False Positive (FP) (Type I error)	True Negative (TN)	Specificity = $\sum$TN / $\sum$condition negative
	Accuracy = $\frac{\sum \text{TP} + \sum \text{TN}}{\sum \text{total population}}$	Precision= $\frac{\sum \text{TP}}{\sum \text{prediction positive}}$		F1 Score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$

CHAPTER 3: TWITTER SENTIMENT ANALYSIS

3.1 INTRODUCTION

Twitter sentiment analysis involves applying machine learning and natural language processing techniques to analyze and interpret the sentiments expressed in tweets on the Twitter platform. This area of study is particularly valuable for understanding public opinion, tracking trends, and gauging the emotional tone surrounding various topics. Researchers and businesses use algorithms to classify tweets as positive, negative, or neutral, providing insights into the sentiment landscape on Twitter. Techniques range from traditional machine learning approaches to more advanced methods, including deep learning models, to effectively capture the nuances of language in short-form tweets. Twitter sentiment analysis finds applications in social listening, brand monitoring, and public opinion analysis.

3.2 CREATING A TWEET CLASSIFYING MACHINE

2.3.1 DATA COLLECTION

In Twitter sentiment analysis, data collection is a crucial step, and one common dataset used for this purpose is the Sentiment140 dataset. This dataset comprises tweets labeled with sentiments: positive, negative. The data collection process involves aggregating a diverse range of tweets from the Twitter platform, covering various topics and sentiments expressed by users.

Sentiment140 is a valuable resource as it provides a labeled dataset, simplifying the task of training and evaluating sentiment analysis models. The dataset is often used for research and development purposes in the field of natural language processing and sentiment analysis. Researchers and practitioners can leverage this dataset to train models to recognize patterns in Twitter text and classify sentiments accurately.

The Sentiment140 dataset simplifies the data collection aspect for sentiment analysis, allowing practitioners to focus on refining and implementing machine learning or natural language processing algorithms to analyze sentiments expressed on Twitter.

Description :

It contains 1,600,000 tweets extracted using the twitter API. The tweets have been annotated (0 = negative, 1 = positive) and they can be used to detect sentiment.

It contains the following 6 fields:

target: the polarity of the tweet (0 = negative, 2 = neutral, 4 = positive)

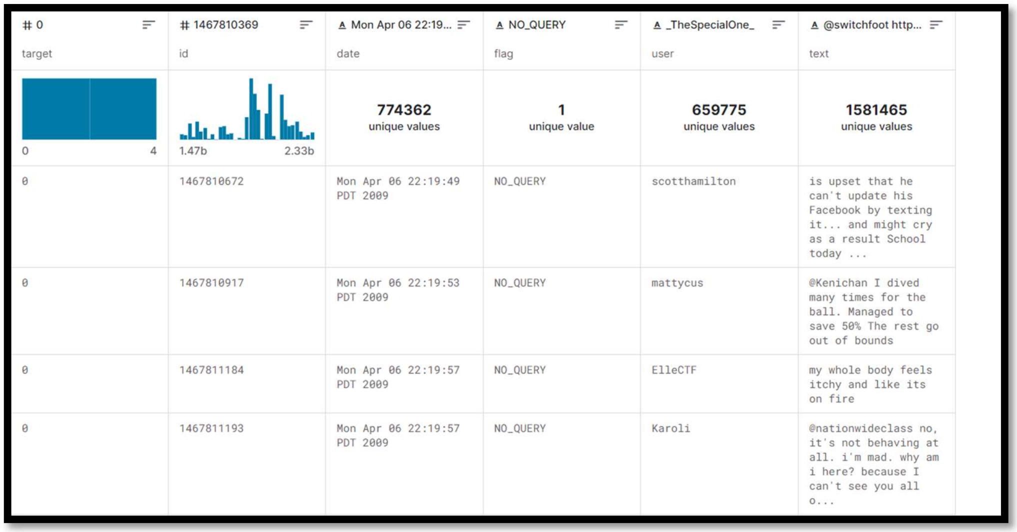
id: The ID of the tweet (2087)

date: the date of the tweet (Sat May 16 23:58:44 UTC 2009)

flag: The query (lyx). If there is no query, then this value is NO_QUERY.

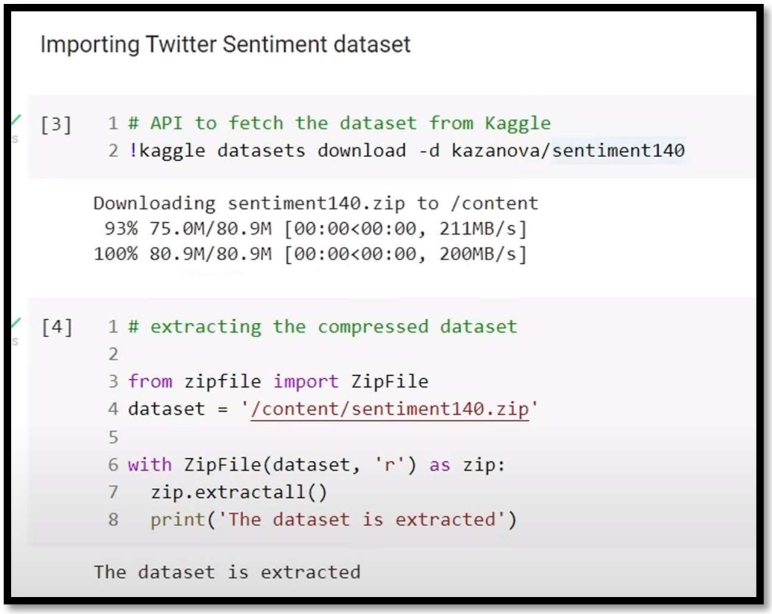
user: the user that tweeted (robotickilldozr)

Text: the text of the tweet (Lyx is cool)



# 0	# 1467810369	Mon Apr 06 22:19...	NO_QUERY	_TheSpecialOne_	@switchfoot http...
target	id	date	flag	user	text
0 4	1.47b 2.33b	774362 unique values	1 unique value	659775 unique values	1581465 unique values
0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by texting it... and might cry as a result School today ...
0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Managed to save 50% The rest go out of bounds
0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all. i'm mad. why am i here? because I can't see you all o...

Importing The Dataset Into Colab Notebook :



Importing Twitter Sentiment dataset

[3]

```
# API to fetch the dataset from Kaggle
!kaggle datasets download -d kazanova/sentiment140
```

```
Downloading sentiment140.zip to /content
 93% 75.0M/80.9M [00:00<00:00, 211MB/s]
100% 80.9M/80.9M [00:00<00:00, 200MB/s]
```

[4]

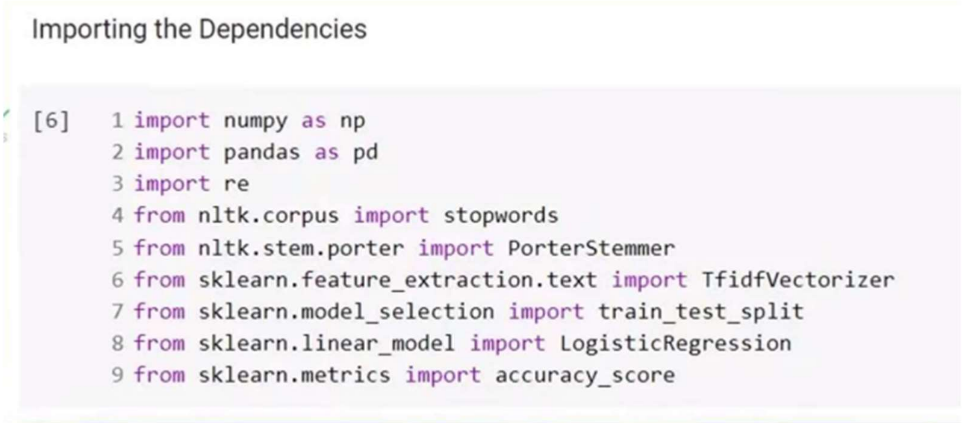
```
# extracting the compressed dataset

from zipfile import ZipFile
dataset = '/content/sentiment140.zip'

with ZipFile(dataset, 'r') as zip:
  zip.extractall()
  print('The dataset is extracted')
```

```
The dataset is extracted
```

Importing Necessary Dependencies



Importing the Dependencies

[6]

```
import numpy as np
import pandas as pd
import re
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
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
```

NumPy:

NumPy, short for Numerical Python, is a fundamental package for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently. NumPy is a cornerstone for numerical and data manipulation tasks in Python.

Pandas:

Pandas is a powerful data manipulation and analysis library. It provides data structures like DataFrames and Series that are designed to handle and analyze structured data seamlessly. Pandas simplifies tasks such as data cleaning, exploration, and transformation, making it a go-to tool for data scientists and analysts.

re (Regular Expressions):

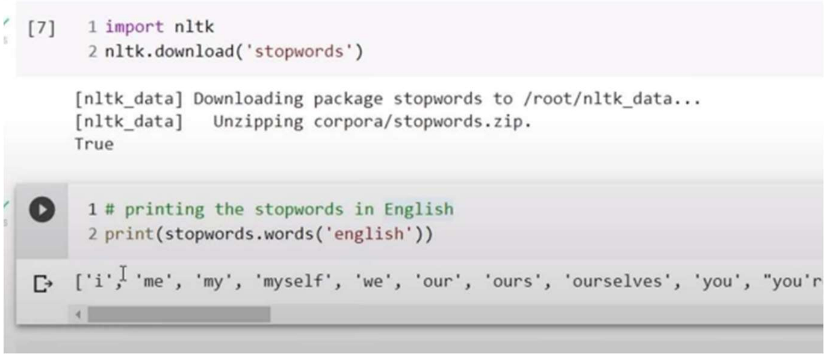
The re module in Python is used for regular expression operations. Regular expressions (regex) are powerful tools for string manipulation, search, and pattern matching. The re module enables developers to define complex search patterns and perform various text-processing tasks with ease.

NLTK (Natural Language Toolkit):

NLTK is a comprehensive library for natural language processing (NLP) in Python. It provides tools for working with human language data, including functionalities for tokenization, stemming, tagging, parsing, and more. NLTK is widely used for building applications that involve text analysis and language understanding.

scikit-learn (sklearn):

Scikit-learn is a versatile machine learning library that simplifies the process of building and implementing machine learning models in Python. It includes a wide array of tools for classification, regression, clustering, dimensionality reduction, and model selection. Scikit-learn is renowned for its user-friendly API and extensive documentation, making it accessible for both beginners and experienced machine learning practitioners.



```
[7]  import nltk
     nltk.download('stopwords')

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
# printing the stopwords in English
print(stopwords.words('english'))

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you'r
```

We are Going to use Pandas, as the dataset is a CSV file and it is easier to handle them as data frames.

Regular Expressions and Stopwords and PorterStemmer will be useful for us in the Data PreProcessing Stage and are taken from the Natural Language Processing Tool Kit (NLTK).

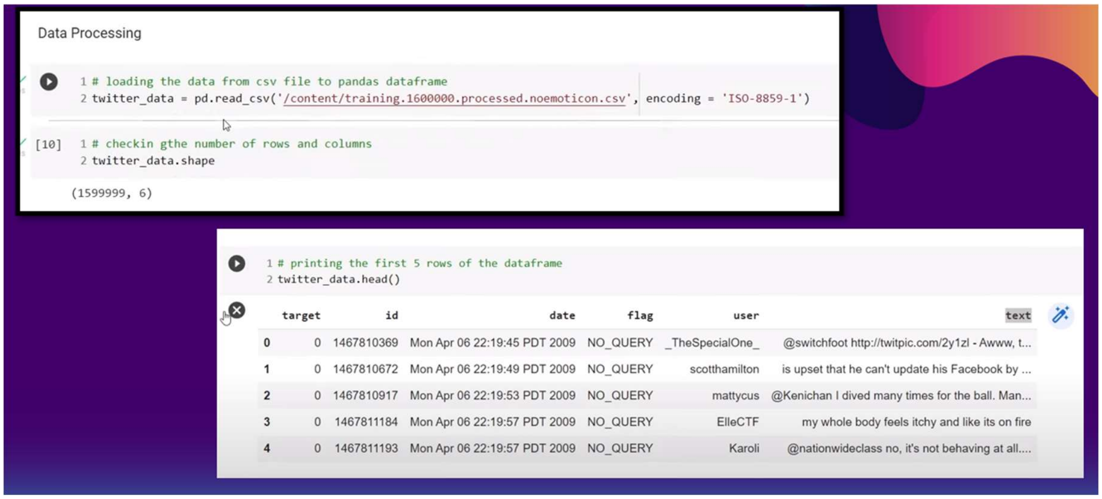
TfidfVectorizer is used to Vectorize the data. We need the train_test_split to separate the training set and the test set from the original dataset.

The Machine Learning Model we are going to use is LogisticRegression as this is a classification problem.

We evaluate the performance of our Model using the accuracy_score.

These libraries collectively form a robust ecosystem for data manipulation, analysis, regular expression operations, natural language processing, and machine learning in the Python programming language.

2.3.2 DATA PRE-PROCESSING



Data Processing

```
# loading the data from csv file to pandas dataframe
twitter_data = pd.read_csv('/content/training.1600000.processed.noemoticon.csv', encoding = 'ISO-8859-1')
```

```
[10]
# checkin gthe number of rows and columns
twitter_data.shape
```

```
(1599999, 6)
```

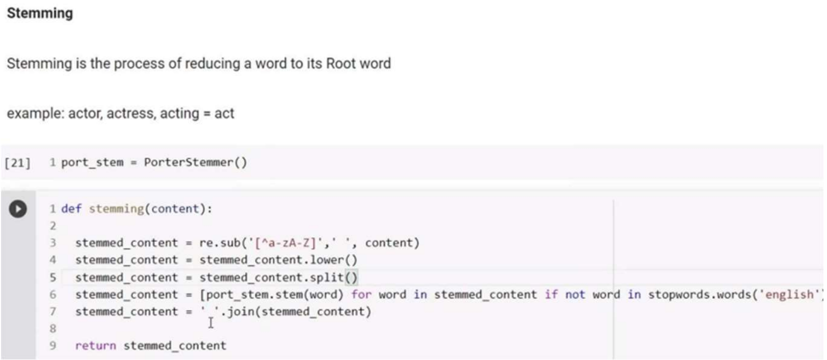
```
# printing the first 5 rows of the dataframe
twitter_data.head()
```

	target	id	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....

As the dataset we have chosen is in .csv we load it into pandas dataframe   for easy and efficient manipulation

We can see the Dataframe and the first 5 rows in the above image.

STEMMING



Stemming

Stemming is the process of reducing a word to its Root word

example: actor, actress, acting = act

```
[21] port_stem = PorterStemmer()
```

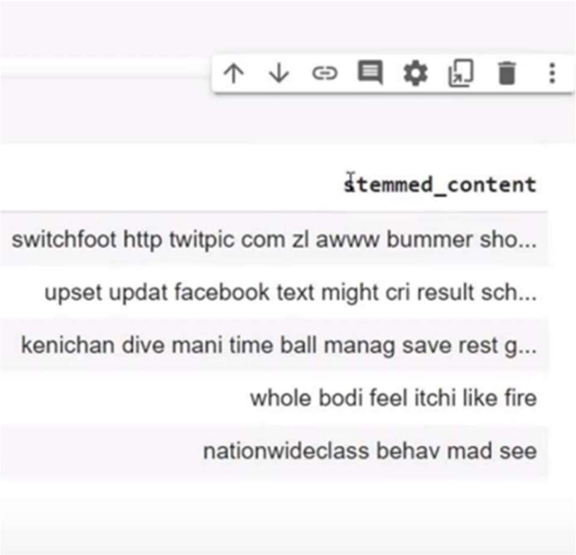
```python
def stemming(content):

  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 in stemmed_content if not word in stopwords.words('english'
  stemmed_content = ' '.join(stemmed_content)

  return stemmed_content
```

Here,
We clean the text data using regex, we remove everything from the text data that isn't an alphabet [^a-zA-Z]
Then, all the alphabets are converted to lower case and are split individually and put in a list.

Stem every word of the data and filter out the stopwords
.
We create a new column 'stemmed_content' which we are going to use in further processing.



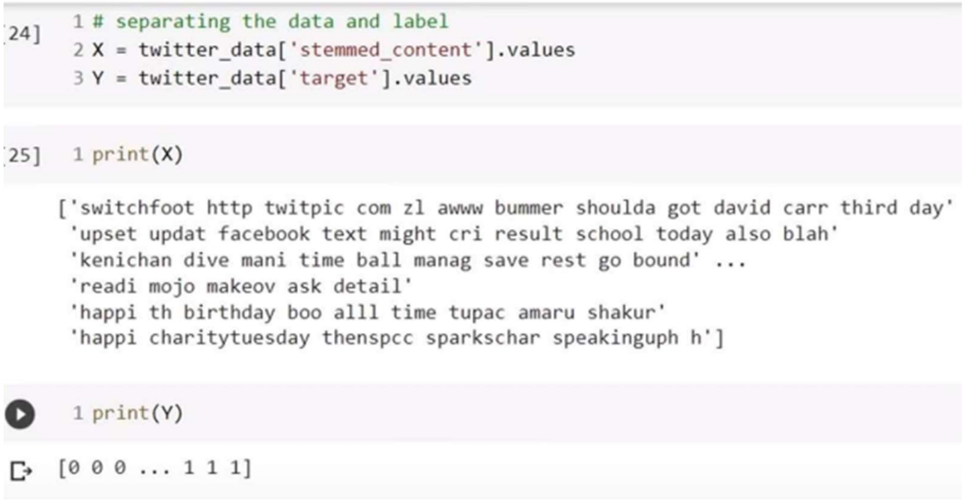
stemmed_content
switchfoot http twitpic com zl awww bummer sho...
upset updat facebook text might cri result sch...
kenichan dive mani time ball manag save rest g...
whole bodi feel itchi like fire
nationwideclass behav mad see

2.3.3 TRAIN-TEST SPLIT

SPLITTING TARGET AND STEMMED CONTENT :

Assign X the data present in ‘stemmed_content’, similarly we assign Y the respective ‘target’ values.

This Makes test-train split easier.



```
[24]  # separating the data and label
      X = twitter_data['stemmed_content'].values
      Y = twitter_data['target'].values
```

```
[25]  print(X)
```

```
['switchfoot http twitpic com zl awww bummer shoulda got david carr third day'
 'upset updat facebook text might cri result school today also blah'
 'kenichan dive mani time ball manag save rest go bound' ...
 'readi mojo makeov ask detail'
 'happi th birthday boo alll time tupac amaru shakur'
 'happi charitytuesday thenspcc sparkschar speakinguph h']
```

```
print(Y)
```

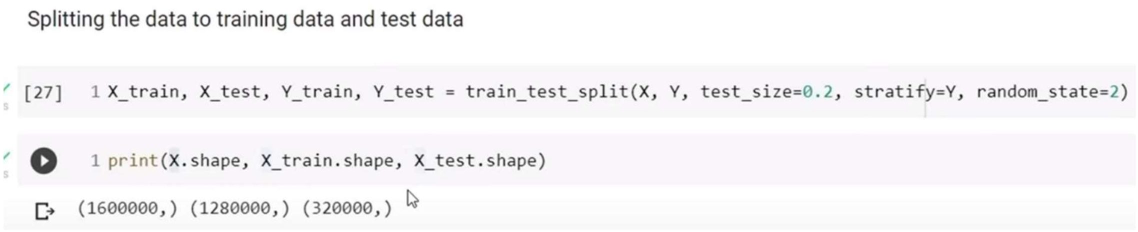
```
[0 0 0 ... 1 1 1]
```

APPLYING TRAIN-TEST SPLIT:

We split the dataset into X_train, X_test and Y_train, Y_test.

‘ train_test_split() ’ performs the splitting

‘test_size = 0.2’ defines that 20% of the data set will be used for the Test Set. Remaining 80% will be used for Train set i.e. training the model.



Splitting the data to training data and test data

```
[27]  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, X_train.shape, X_test.shape)
```

```
(1600000,) (1280000,) (320000,)
```

VECTORIZATION :

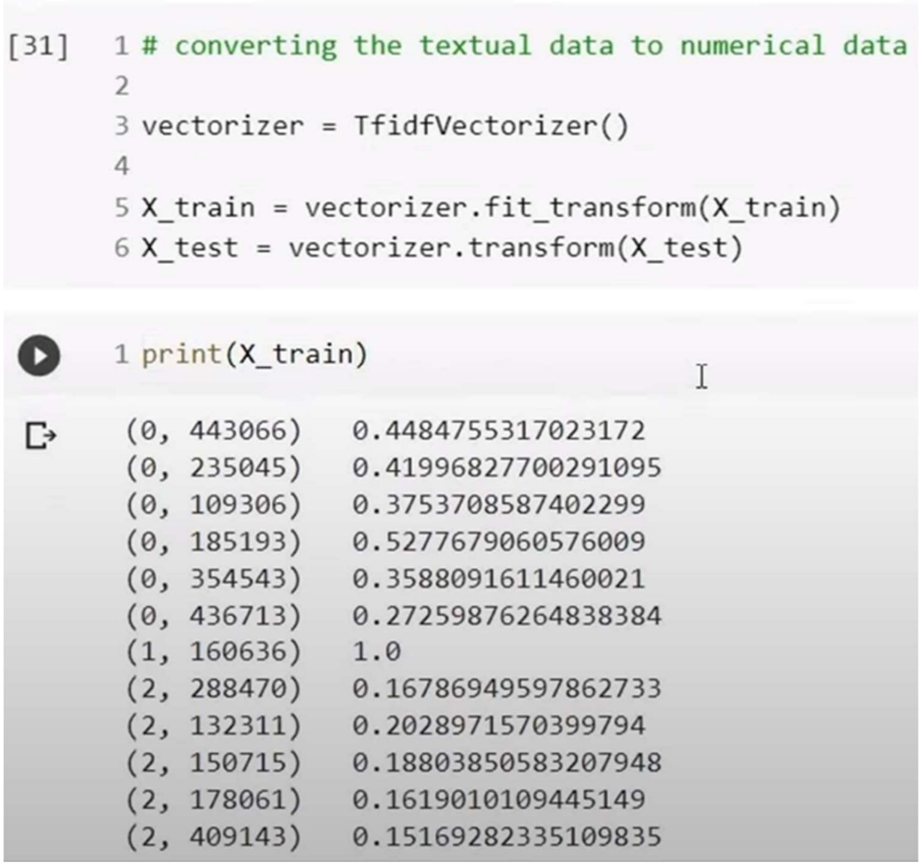
TF-IDF Vectorizer is a valuable tool for converting raw text data into a format suitable for machine learning models.

TF-IDF stands for Term Frequency-Inverse Document Frequency.

It is a numerical statistic that reflects the importance of a word in data

‘vectoriser.transform(X_train)’ transforms the training text data.

‘vectoriser.transform(X_test) ‘ transforms the testing text data.



```
# converting the textual data to numerical data

vectorizer = TfidfVectorizer()

X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

```
print(X_train)
```

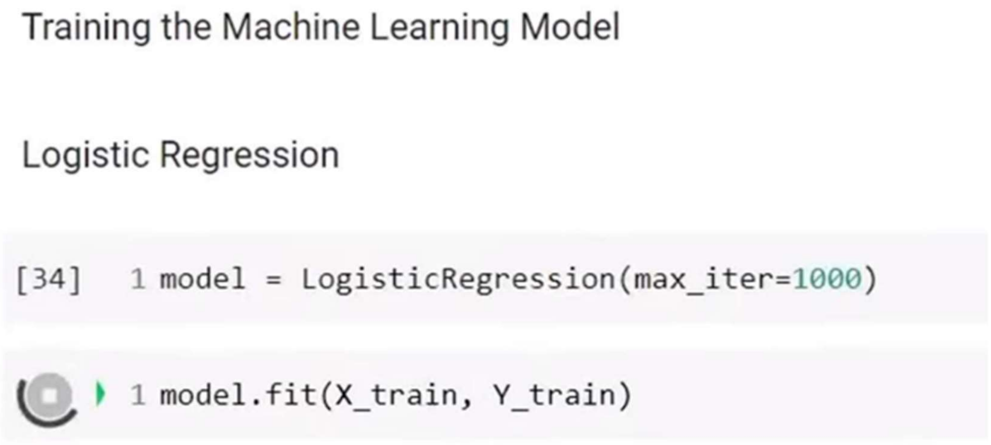
```
  (0, 443066)	0.4484755317023172
  (0, 235045)	0.41996827700291095
  (0, 109306)	0.3753708587402299
  (0, 185193)	0.5277679060576009
  (0, 354543)	0.3588091611460021
  (0, 436713)	0.27259876264838384
  (1, 160636)	1.0
  (2, 288470)	0.16786949597862733
  (2, 132311)	0.2028971570399794
  (2, 150715)	0.18803850583207948
  (2, 178061)	0.1619010109445149
  (2, 409143)	0.15169282335109835
```

2.3.4 TRAINING THE MACHINE LEARNING MODEL

We Feed the Data present in the Train Set for Training our Machine Learning Model. We are Using the Logistic Regression Model.

max_iter=1000: Setting the maximum number of iterations.

model.fit(X_train, Y_train): Training the model with the training data.



Training the Machine Learning Model

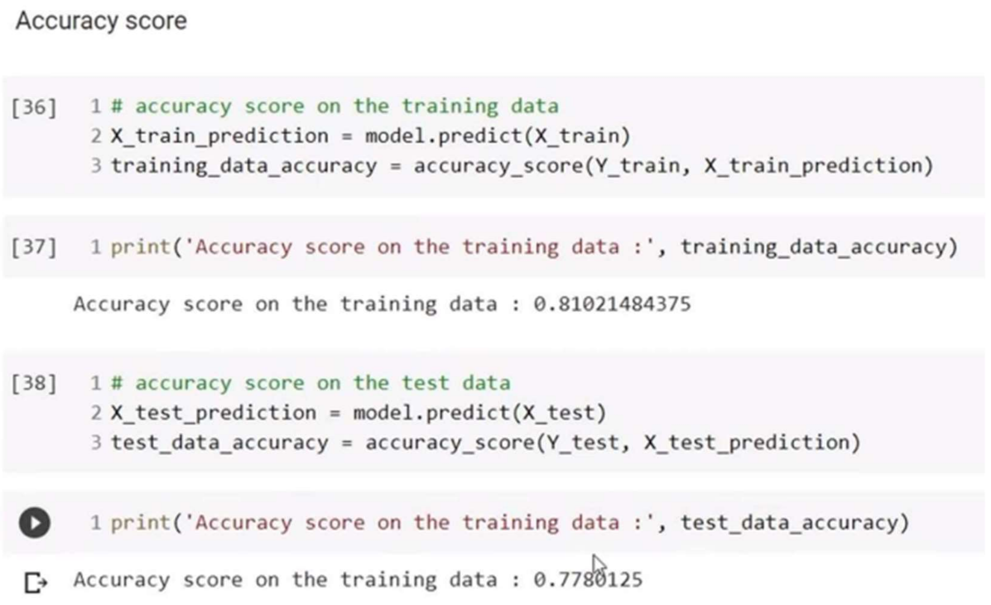
Logistic Regression

```
[34]  model = LogisticRegression(max_iter=1000)
```

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

2.3.5 EVALUATING THE MACHINE LEARNING MODEL

After completion of training we evaluate the performance of our trained model. To know whether the model has been properly trained or not.



Accuracy score

```
[36]  # accuracy score on the training data
      X_train_prediction = model.predict(X_train)
      training_data_accuracy = accuracy_score(Y_train, X_train_prediction)
```

```
[37]  print('Accuracy score on the training data :', training_data_accuracy)
```

```
Accuracy score on the training data : 0.81021484375
```

```
[38]  # accuracy score on the test data
      X_test_prediction = model.predict(X_test)
      test_data_accuracy = accuracy_score(Y_test, X_test_prediction)
```

```
print('Accuracy score on the training data :', test_data_accuracy)
```

```
Accuracy score on the training data : 0.7780125
```

‘

‘X_train_prediction = model.predict(X_train) ‘ will use the trained model to predict the labels for the training data.

‘ accuracy_score(Y_train, X_train_prediction) ‘ will Calculate the accuracy of the model’s predictions.

As we can see in the above image the Accuracy Score on Training Data is 81%
Accuracy Score on Test Data is 77%

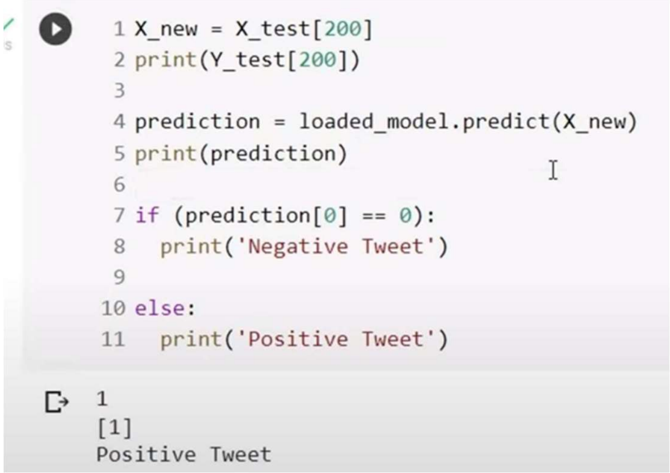
This Implies that the model has been trained successfully and is ready to be deployed.

2.3.6 DEPLOYING THE TRAINED MACHINE LEARNING MODEL :

The Trained Model that we have just created is put to use.

‘ Y_test[200] ’ gives the respective true Target value on the 200th index of the Dataset

‘ X_new ’ gives Predicted Target value on the 200th index of the dataset.



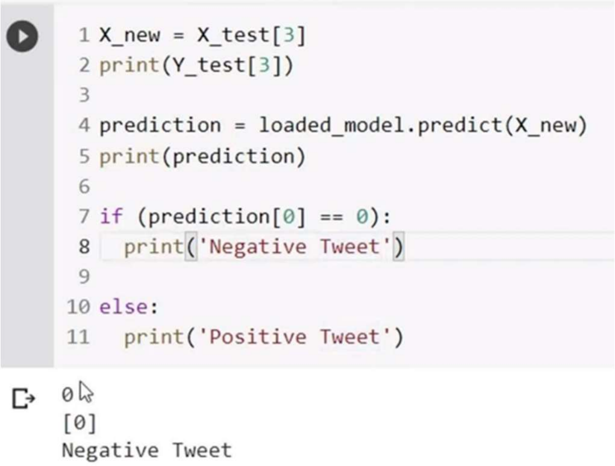
```
X_new = X_test[200]
print(Y_test[200])

prediction = loaded_model.predict(X_new)
print(prediction)

if (prediction[0] == 0):
  print('Negative Tweet')

else:
  print('Positive Tweet')
```

```
1
[1]
Positive Tweet
```



```
X_new = X_test[3]
print(Y_test[3])

prediction = loaded_model.predict(X_new)
print(prediction)

if (prediction[0] == 0):
  print('Negative Tweet')

else:
  print('Positive Tweet')
```

```
0
[0]
Negative Tweet
```

We can see that the machine has succesully classified the respective tweets as positive and negative.

CONCLUSION

In conclusion, the implementation of a tweet-classifying machine using logistic regression and sentiment analysis has yielded a test accuracy score of 77%. This indicates a reasonable level of effectiveness in predicting sentiment classes for tweets. While the model demonstrates competency, there is room for improvement, and further refinements could potentially enhance its performance. The utilization of logistic regression, coupled with sentiment analysis, offers a pragmatic approach for classifying tweets and understanding the sentiments expressed within them. Overall, this system serves as a foundation for sentiment analysis on Twitter, with future opportunities for optimization and expansion.

REFRENCES

- D’Andrea, A. (2020). Approaches, Tools and Applications for Sentiment Analysis Implementation [Research]. https://www.ijcaonline.org/research/volume125/number3/dandrea-2015-ijca-905866.pdf

- Qaid Aqlan, A. A. (2019). A Study of Sentiment Analysis: Concepts, Techniques, and Challenges [Research, Department of Computer Science, Kakatiya University]. https://www.researchgate.net/publication/332451019_A_Study_of_Sentiment_Analysis_Concepts_Techniques_and_Challenges

- Goyal, G. (2023, December 3). Twitter Sentiment Analysis Using Python : Introduction & Techniques. Analytics Vidya. Retrieved December 17, 2023, from https://www.analyticsvidhya.com/blog/2021/06/twitter-sentiment-analysis-a-nlp-use-case-for-beginners/

- [GeeksForGeeks]. (2023, November 13). TWITTER SENTIMENT ANALYSIS (NLP) | Machine Learning Projects | GeeksforGeeks [Video]. Youtube. https://youtu.be/4YGkfAd2iXM?feature=shared