DATA SCIENCE

Classify tweets from the dataset as either as positive or negative

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

Bachelor of Technology

In

Computer Science Engineering

By

Harshitha Palligunthala

221810306041

Under the Guidance of

Assistant Professor

Department of Computer Science Engineering

GITAM School of Technology

GITAM (Deemed to be University)

Hyderabad-502329

month year

DECLARATION

I submit this industrial training work entitled “DATA SCIENCE” 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 Engineering**”. I declare that it was carried out independently by me under the guidance of _____, 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

Date:

Student Name: P.HARSHITHA

Student Roll No: 221810306041

GITAM (DEEMED TO BE UNIVERSITY)

Hyderabad-502329, India

Dated:

CERTIFICATE

This is to certify that the Industrial Training Report entitled “DATA SCIENCE” is being submitted by P. HARSHITHA (221810306041) in partial fulfilment of the requirement for the award of **Bachelor of Technology in Computer Science Engineering** at GITAM (Deemed to Be University), Hyderabad during the academic year 2021-2022

ACKNOWLEDGMENT

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 completion of this internship.

I would like to thank respected **Dr. N. Siva Prasad**, Pro Vice Chancellor, GITAM Hyderabad and **Dr. N Seetha Ramaiah**, Principal, GITAM Hyderabad.

I would like to thank respected **Dr. S. Phani Kumar**, Head of the Department of Computer Science 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 what we study for.

I would like to thank the respected faculties who helped me 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.

Student name: P. HARSHITHA

Student pin number: 221810306041

INDEX

LIST OF FIGUURES

CHAPTER 1
MACHINE LEARNING

1.1 INTRODUCTION

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics.

1.2 IMPORTANCE OF MACHINE LEARNING

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

The practical applications of machine learning drive business results which can dramatically affect a company’s bottom line. New techniques in the field are evolving rapidly and expanded the application of machine learning to nearly limitless possibilities. Industries that depend on vast quantities of data—and need a system

to analyse it efficiently and accurately, have embraced machine learning as the best way to build models, strategize, and plan.

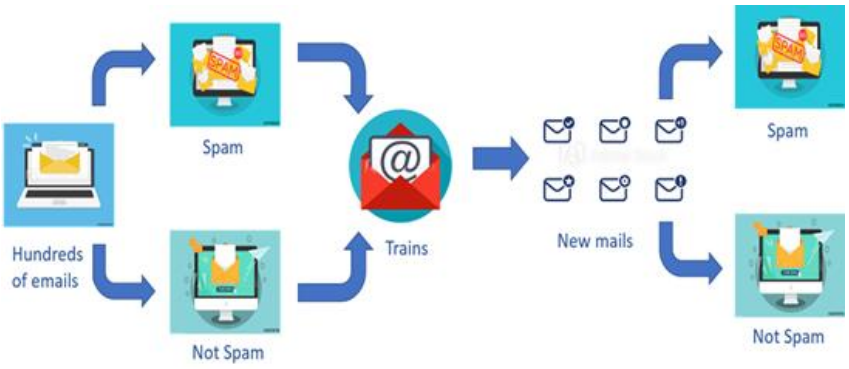

Figure 1.2.1 Email spam filter

1.3 MACHINE LEARNING MODELS

Machine Learning is broadly divided into three main areas, supervised learning, unsupervised and reinforcement learning. Each one of these has a specific action and purpose, yielding particular results by using various types of data.

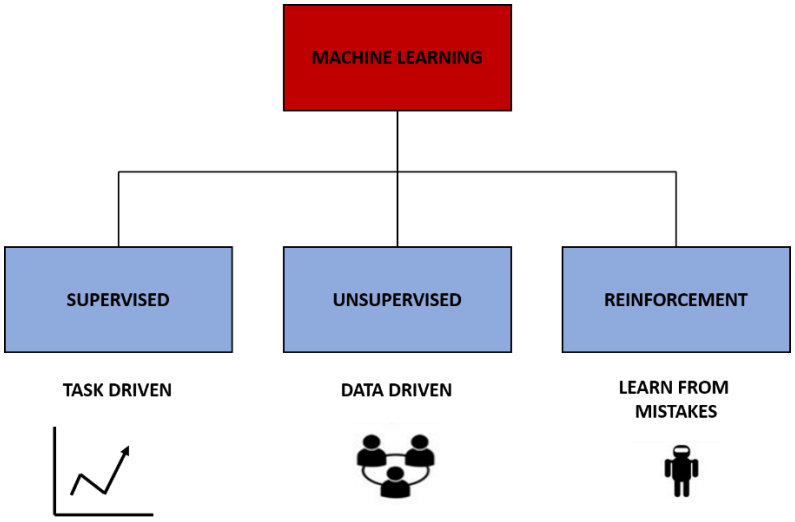

Figure 1.3.1 Types of Machine learning models

1.3.1 Supervised Machine Learning

Supervised learning in simple language means training the machine learning model just like a coach trains a batsman. In Supervised Learning, the machine learns under the guidance of labelled data i.e. known data. This known data is fed to the machine learning model and is used to train it. Once the model is trained with a known set of data, you can go ahead and feed unknown data to the model to get a new response.

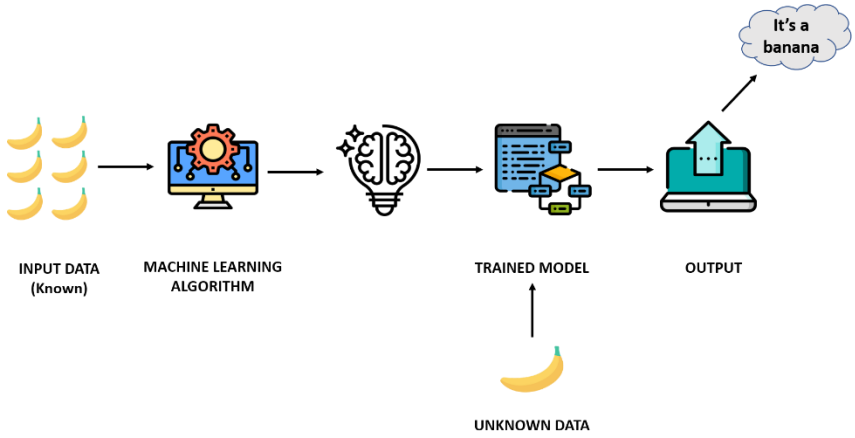

Figure 1.3.1.1 Supervised learning

In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the supervision of the teacher. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to map the input variable(x) with the output variable(y).

In the real-world, supervised learning can be used for Risk Assessment, Image classification, Fraud Detection, spam filtering, etc.

1.3.1.1 Classification

A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes.

1.3.1.2 Regression

A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points.

1.3.2 Unsupervised Machine Learning

Unsupervised machine learning in simple language means the ml model is self-sufficient in learning on its own.

In unsupervised machine learning, there is no such provision of labeled data. The training data is unknown or unlabelled. This unknown data is fed to the machine learning model and is used to train the model. The model tries to find patterns and relationships in the dataset by creating clusters in it. The thing to be noted here is that unsupervised learning is not able to add labels to the clusters. For example, it cannot say this is a group of oranges or mangoes, but it will separate all the oranges from mangoes.

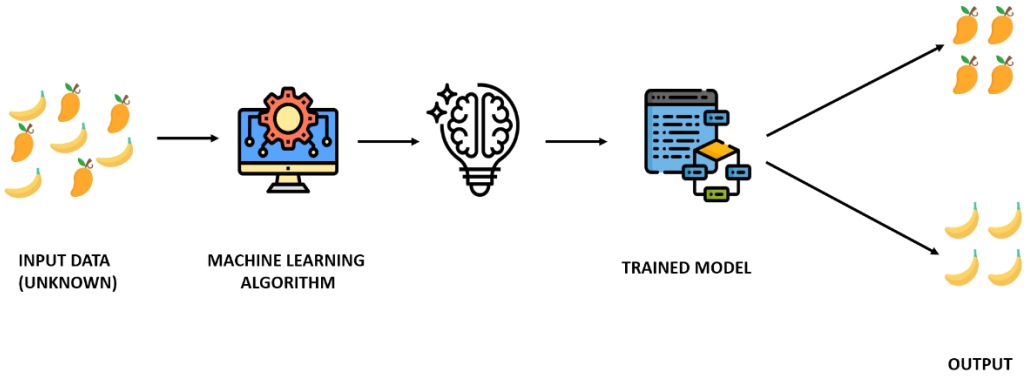

Figure 1.3.2.1 Unsupervised learning

It uses machine learning algorithms to analyse and cluster un labelled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, and image recognition.

1.3.3 Reinforcement Machine Learning

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

CHAPTER 2

PYTHON

2.1 INTRODUCTION TO PYTHON

Python is a high-level object-oriented programming language that was created by Guido van Rossum. It is also called general-purpose programming language as it is used in almost every domain we can think of as mentioned below:

- Web Development
- Software Development
- Game Development
- AI & ML
- Data Analytics

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

2.2 HISTORY OF PYTHON

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989.

2.3 FEATURES OF PYTHON

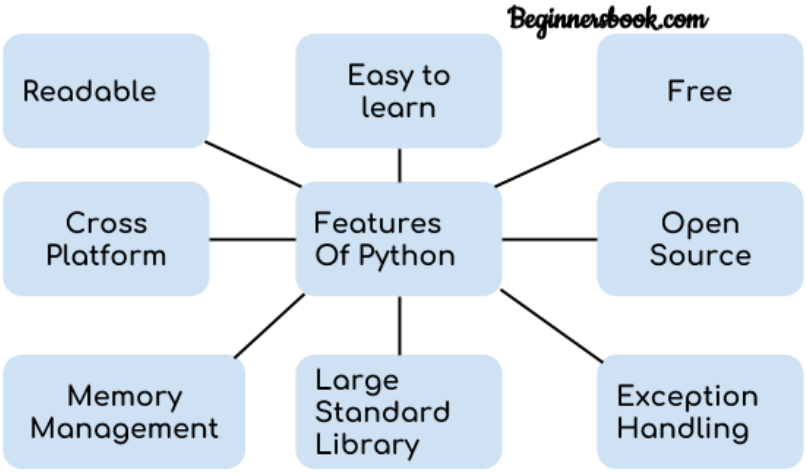

Figure 2.3.1 Python Features

1. **Readable:** Python is a very readable language.

2. **Easy to Learn:** Learning python is easy as this is expressive and high level programming language, which means it is easy to understand the language and thus easy to learn.

3. **Cross platform:** Python is available and can run on various operating systems such as Mac, Windows, Linux, Unix etc. This makes it a cross platform and portable language.

4. **Open Source:** Python is a open source programming language.

5. **Large standard library:** Python comes with a large standard library that has some handy codes and functions which we can use while writing code in Python.

6. **Free:** Python is free to download and use. This means you can download it for free and use it in your application.

7. **Supports exception handling:** Python supports exception handling which means we can write less error prone code and can test various scenarios that can cause an exception later on.

8. **Advanced features:** Supports generators and list comprehensions. We will cover these features later.

9. **Automatic memory management:** Python supports automatic memory management which means the memory is cleared and freed automatically. You do not have to bother clearing the memory.

2.4 INSTALLATION OF PYTHON

Python installation using anaconda (In Windows):

Step 1: Open Anaconda.com/downloads in web browser.

Step 2: Download python 3.4 version for 32-bit graphic installer / 64-bit graphic installer

Step 3: Select installation type

Step 4: Select path (i.e add anaconda to path and register anaconda as default python 3.4) next click install and next click finish

Step 5: Open jupyter notebook (it open sin default browser)

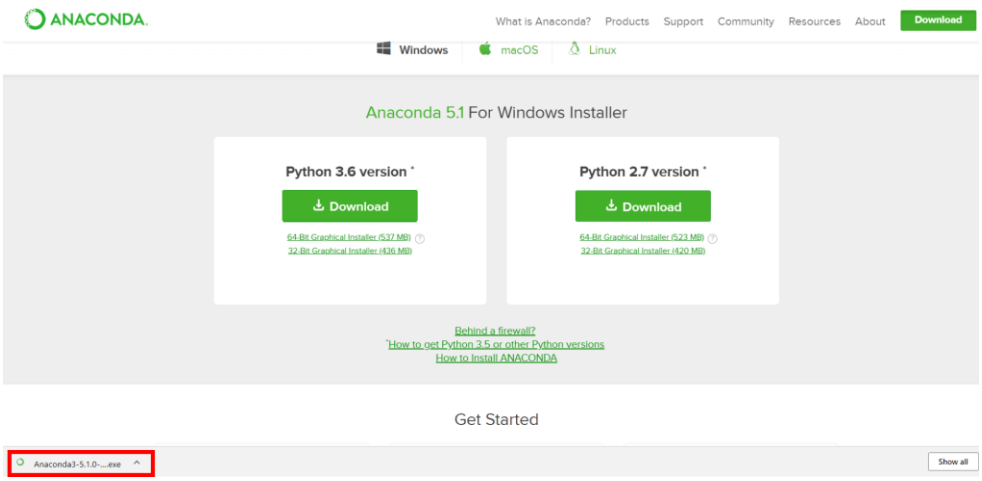

Figure 2.4.1 Anaconda download

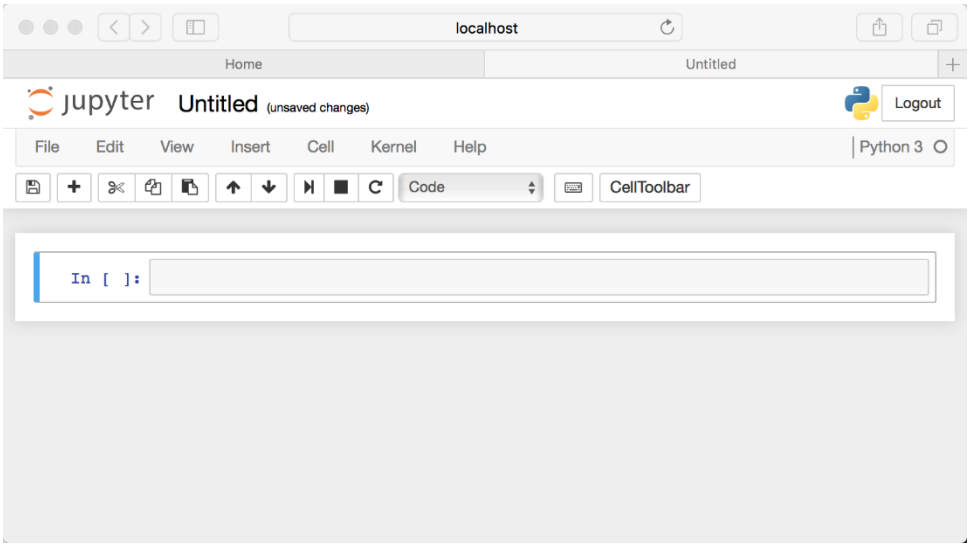

Figure 2.4.2 Jupyter notebook

2.5 PYTHON VARIABLES

In Python, Variable types in the program are entirely dependent on the type of data that are to be used for declaring, defining and performing mathematical functions on the input provided by the user.

2.5.1 Integers and floats

- Integers are numbers, and floats are decimal numbers.

- Integers are a number that can be positive or negative or 0, but they cannot have a decimal point. They have unlimited precision and support all kinds of mathematical and arithmetical operations.
- The integer type is int, and the floating number type is afloat. These types of names to convert or cast a variable to an integer or to afloat.

2.5.2 Strings

- We make use of strings to symbolize text. Automatically, it is Unicode text in Python 3 yet ASCII text through Python 2.
- Strings can be defined using single quotes, double quotes, or three times the quotes, either single or double.
- Python support many methods, including many useful utility methods.
- Some of them capitalize, which will make the first character a capital letter. replace() method takes two arguments, the first one being the character to be replaced, and the second one is the character to replace it with. Then we have alpha() or isdigit(), which will return true if all characters are letters or digits respectively.

2.5.3 Lists

- Lists are used to store multiple items in a single variable.
- Lists are one of 4 built-in data types in Python used to store collections of data. Lists are created using square brackets.
- List items are ordered, changeable, and allow duplicate values.
- List items are indexed, the first item has index [0], the second item has index [1] etc.
- They are used to store an ordered collection of items, which might be of different types but usually they aren't

2.5.4 Tuples

- A Tuple is a collection of Python objects separated by commas.
- Tuple is similar to a list in terms of indexing, nested objects and repetition but a tuple is immutable unlike lists which are mutable.
- Tuples are used to store multiple items in a single variable. Tuples are written with round brackets.
- Tuples are fixed size in nature whereas lists are dynamic. In other words, a tuple is immutable whereas a list is mutable. You can't add elements to a tuple.
- Tuples have no append or extend method. You can't remove elements from a tuple. Tuples have no remove or pop method.

2.5.5 Dictionaries

- Dictionaries are used to store data values in key: value pairs, and can be referred to by using the key name.

- A dictionary is a collection which is ordered*, changeable and does not allow duplicates.

- Dictionaries are enclosed by curly braces ({}) and values can be assigned and accessed using square braces ([]).
- Dictionaries are very useful when it comes to storing some kind of structured data.

2.6 PYTHON FUNCTIONS

A function is a block of code which only runs when it is called. You can pass data, known as parameters, into a function. A function can return data as a result.

A Python function is a group of code. To run the code in a function, you must call the function. A function can be called from anywhere

after the function is defined. Functions can return a value using a return statement.

2.6.1 Function call & Definition

Defining a function refers to creating the function. This involves writing a block of code that we can call by referencing the name of our function. A function is denoted by the def keyword, followed by a function name, and a set of parenthesis.

A function can be called anywhere after the function has been declared. By itself, a function does nothing. But, when you need to use a function, you can call it, and the code within the function will be executed.

2.7 OOPS CONCEPT IN PYTHON

Object-Oriented Programming (OOP), is all about creating “objects”. An object is a group of interrelated variables and functions. These variables are often referred to as properties of the object and functions are referred to as the behaviour of the objects. These objects provide a better and clear structure for the program.

2.7.1 class

A class is a collection of objects. Unlike the primitive data structures, classes are data structures that the user defines. They make the code more manageable.

We define a class with a keyword “class” following the class_name and semicolon. And we consider everything you write under this after using indentation as its body.

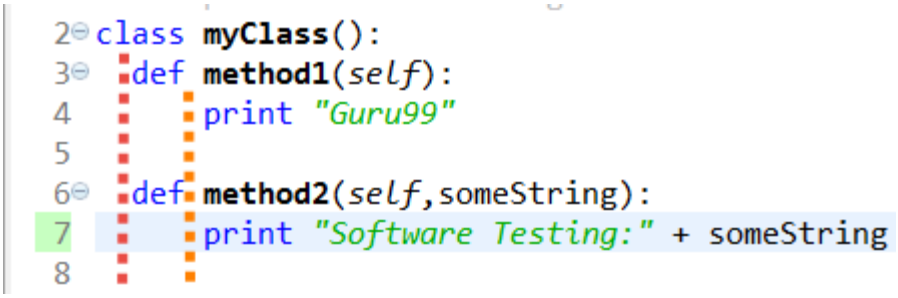

Figure 2.7.1.1 Defining a class

CHAPTER 3

DATA VISUALIZATION

3.1 INTRODUCTION

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends and correlations that might not otherwise be detected can be exposed.

Python offers multiple great graphing libraries that come packed with lots of different features. No matter if we want to create interactive, live or highly customized plots python has an excellent library.

Few popular plotting libraries:

- Matplotlib: low level, provides lots of freedom
- Pandas Visualization: easy to use interface, built on Matplotlib
- Seaborn: high-level interface, great default styles
- ggplot: based on R’s ggplot2, uses Grammar of Graphics
- Plotly: can create interactive plots

3.2 IMPORTANCE OF DATA VISUALIZATION

Data visualization helps to tell stories by curating data into a form easier to understand, highlighting the trends and outliers. A good visualization tells a story, removing the noise from data and highlighting the useful information.This makes the data more natural for the human mind to comprehend and therefore makes it easier to identify trends, patterns, and outliers within large data sets.

- Data visualization strengthens the impact of messaging for your audiences and presents the data analysis results in the most persuasive manner. It unifies the messaging systems across all the groups and fields within the organization.
- Visualization lets you comprehend vast amounts of data at a glance and in a better way. It helps to understand the data better to measure its impact on the business and communicates the insight visually to internal and external audiences.
- Decisions can’t be made in a vacuum. Available data and insights enable decision-makers to aid decision analysis. Unbiased data without inaccuracies allows access to the right kind of information and visualization to represent that information and keep it relevant.

CHAPTER 4

CASE STUDY

4.1 PROBLEM STATEMENT

Classify tweets from the datasets as either as positive or negative using two different Machine Learning Algorithms. They are:

1. Logistic Regression
2. Decision tree classifier

4.2 DATA SET

The given data set consists of the following parameters:

A. id: The id associated with the tweets in the given dataset.

B. labels: A tweet with label ‘0’ is of positive sentiment while a tweet with label ‘1’ is of negative sentiment.

C. tweets: The tweets collected from various sources and having either positive or negative sentiments associated with it.

4.3 OBJECTIVE OF THE CASE STUDY

Sentiment analysis, also refers as opinion mining, is a sub machine learning task where we want to determine which is the general sentiment of a given document. Using machine learning techniques and natural language processing we can extract the subjective information of a document and try to classify it according to its polarity such as positive, neutral or negative. It is a really useful analysis since we could possibly determine the overall opinion about a selling objects , or predict stock markets for a given company like, if most people think positive about it, possibly its stock markets will increase, and so on. In this project we classify tweets from Twitter into “positive” or “negative” sentiment by building a model based on probabilities

CHAPTER 5

MODEL BUILDING

5.1 PREPROCESSING OF THE DATA

Preprocessing of data involves following steps

5.1.1 Getting the data set

We can get the data set from the database or we can get the data from client.

5.1.2 Importing the Packages

We import all the packages that are necessary

```
# import necessary packages
import re
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import string
import nltk
import warnings
warnings.filterwarnings("ignore" , category=DeprecationWarning)

%matplotlib inline
```

Figure 5.1.2.1 Importing libraries

5.1.3 Importing the data set

Pandas in python provide an interesting method read_csv(). The read_csv function reads the entire dataset from a comma separated values file and we can assign it to a Data Frame to which all the operations can be performed. It helps us to access each and every row as well as columns and each and every value can be access using the data frame.

```
# loading training data set
train=pd.read_csv("train.csv")

```

Figure 5.1.3.1 Reading the data set

5.1.4 Data set

```
train
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation
...	...	...	...
31957	31958	0	ate @user isz that youuu?ð□□□ð□□□ð□□□ð□□□ð□□□ð...
31958	31959	0	to see nina turner on the airwaves trying to...
31959	31960	0	listening to sad songs on a monday morning otw...
31960	31961	1	@user #sikh #temple vandalised in in #calgary,...
31961	31962	0	thank you @user for you follow

31962 rows × 3 columns

Figure 5.1.4.1 Data set

5.1.5 Vectorizer

Here NumPy Vectorization np.vectorize() is used because it is much more faster than the conventional for loops when working on datasets of medium to large sizes. We will use this to remove the pattern '@user' from all the tweets in our data.

```
combine['new_tweets'] = np.vectorize(remove_pattern)(combine['tweet'], "@[\w]*")

combine.head()
```

	id	label	tweet	new_tweets
0	1	0.0	@user when a father is dysfunctional and is s...	when a father is dysfunctional and is so sel...
1	2	0.0	@user @user thanks for #lyft credit i can't us...	thanks for #lyft credit i can't use cause th...
2	3	0.0	bihday your majesty	bihday your majesty
3	4	0.0	#model i love u take with u all the time in ...	#model i love u take with u all the time in ...
4	5	0.0	factsguide: society now #motivation	factsguide: society now #motivation

Figure 5.1.5.1 Applying vectorizer

5.1.6 Tokenization

Tokens are individual terms or words, and tokenization is the process of splitting a string of text into tokens.

```
# splitting text into tokens (tokenization)
token_tweets=combine['new_tweets'].apply(lambda x:x.split())
token_tweets
```

```
0        [when, father, dysfunctional, selfish, drags, ...
1        [thanks, #lyft, credit, cause, they, offer, wh...
2                                  [bihday, your, majesty]
3                        [#model, love, take, with, time]
4                      [factsguide, society, #motivation]
                               ...
49154    [thought, factory, left, right, polarisation, ...
49155    [feeling, like, mermaid, #hairflip, #neverread...
49156    [#hillary, #campaigned, today, #ohio, used, wo...
49157    [happy, work, conference, right, mindset, lead...
49158    [song, glad, free, download, #shoegaze, #newmu...
Name: new_tweets, Length: 49159, dtype: object
```

Figure 5.1.6.1 Applying tokenization

5.1.7 Stemming

Stemming is a rule-based process of stripping the suffixes (“ing”, “ly”, “es”, “s” etc) from a word.

```
# removing suffixes like 'ly','ing' etc(stemming)
from nltk import PorterStemmer
ps=PorterStemmer()
token_tweets=token_tweets.apply(lambda x:[ps.stem(i) for i in x])
token_tweets
```

```
0        [when, father, dysfunct, selfish, drag, kid, i...
1        [thank, #lyft, credit, caus, they, offer, whee...
2                                  [bihday, your, majesti]
3                        [#model, love, take, with, time]
4                            [factsguid, societi, #motiv]
                               ...
49154    [thought, factori, left, right, polaris, #trum...
49155    [feel, like, mermaid, #hairflip, #neverreadi, ...
49156    [#hillari, #campaign, today, #ohio, use, word,...
49157    [happi, work, confer, right, mindset, lead, cu...
49158    [song, glad, free, download, #shoegaz, #newmus...
Name: new_tweets, Length: 49159, dtype: object
```

Figure 5.1.7.1 Stemming

5.2 FEATURE EXTRACTION

5.2.1 Bag of words

Bag of Words is a method to extract features from text documents. These features can be used for training machine learning algorithms. It

creates a vocabulary of all the unique words occurring in all the documents in the training set.

```
# applying bag of words feature extraction
from sklearn.feature_extraction.text import CountVectorizer

bag_of_words=CountVectorizer(max_df=0.90,min_df=2,max_features=1000,stop_words='english').fit_transform(combine['new_tweets'
df_bag_of_words=pd.DataFrame(bag_of_words.todense())
df_bag_of_words
```

	0	1	2	3	4	5	6	7	8	9	...	990	991	992	993	994	995	996	997	998	999
0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
49154	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
49155	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
49156	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
49157	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
49158	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

49159 rows × 1000 columns

Figure 5.2.1.1 Applying bag of words

5.2.2 TF-IDF

TF-IDF stands for Term frequency-Inverse document frequency and the TF-IDF weight is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus.

TF=(no.of times the word occurs in the text) / (Total no.of words in the text)

IDF= (Total no.of documents) / no.of documents with word t in it)

TF-IDF= TF*IDF

```
# applying TF-IDF feature extraction
from sklearn.feature_extraction.text import TfidfVectorizer

tfidf=TfidfVectorizer(max_df=0.90,min_df=2,max_features=1000,stop_words='english').fit_transform(combine['new_tweets'])
df_tfidf=pd.DataFrame(tfidf.todense())
df_tfidf
```

	0	1	2	3	4	5	6	7	8	9	...	990	991	992	993	994	995	996	997	998	999
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
49154	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
49155	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
49156	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
49157	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
49158	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

49159 rows × 1000 columns

Figure 5.2.2.1 Applying TF-IDF

5.3 ALGORITHMS USED

Here we use 2 different models and then we will compare their performance and choose the best possible model with the best possible feature extraction technique for predicting results on our test data.

5.3.1 Logistic Regression

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. it is a predictive analysis algorithm and based on the concept of probability.

The hypothesis of logistic regression tends it to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

Logistic regression is used to obtain odds ratio in the presence of more than one explanatory variable. The procedure is quite similar to multiple linear regression, with the exception that the response variable is binomial. The result is the impact of each variable on the odds ratio of the observed event of interest.

5.3.1.1 Applying algorithm and predict probability

Fitting the logistic regression model

Here we apply the algorithm and predict the probabilities for a tweet falling into either positive or negative class of both the features extracted using Bag of words technique and TF-IDF technique

```
from sklearn.linear_model import LogisticRegression
logis_reg=LogisticRegression(random_state=0,solver='lbfgs')
```

```
# fitting bag of words features
logis_reg.fit(x_train_bgwords,y_train_bgwords)
# predicting the probabilities for a tweet falling into either positive or negative class
predict_bgwords=logis_reg.predict_proba(x_valid_bgwords)
predict_bgwords
```

```
array([[9.86501156e-01, 1.34988440e-02],
       [9.99599096e-01, 4.00904144e-04],
       [9.13577383e-01, 8.64226167e-02],
       ...,
       [8.95457155e-01, 1.04542845e-01],
       [9.59736065e-01, 4.02639345e-02],
       [9.67541420e-01, 3.24585797e-02]])
```

```
# fitting tfidf features
logis_reg.fit(x_train_tfidf,y_train_tfidf)
# predicting the probabilities for a tweet falling into either negative or positive class
predict_tfidf=logis_reg.predict_proba(x_valid_tfidf)
predict_tfidf
```

```
array([[0.98487907, 0.01512093],
       [0.97949889, 0.02050111],
       [0.9419737 , 0.0580263 ],
       ...,
       [0.98630906, 0.01369094],
       [0.96746188, 0.03253812],
       [0.99055287, 0.00944713]])
```

Figure 5.3.1.1.1 Applying algorithm

5.3.1.2 F1 Score calculation

Here we calculate the F1 score of both the techniques

```
# calculating f1_score
predict1=predict_bgwords[:,1]>=0.3
predict1=predict1.astype(np.int)
predict1
logist_bgwords_score=f1_score(y_valid_bgwords,predict1)
print("F1 score of Logistic Regression with Bag of words features :",logist_bgwords_score)
```

```
F1 score of Logistic Regression with Bag of words features : 0.5721352019785655
```

```
# calculating F1 score
predict2=predict_tfidf[:,1]>=0.3
predict2=predict2.astype(np.int)
predict2
logis_tfidf_score=f1_score(y_valid_tfidf,predict2)
print("F1 score of Logistic Regression with TFIDF features is:",logis_tfidf_score)
```

```
F1 score of Logistic Regression with TFIDF features is: 0.5862068965517241
```

Figure 5.3.1.2.1 F1 Score

5.3.2 Decision Tree

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.
- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

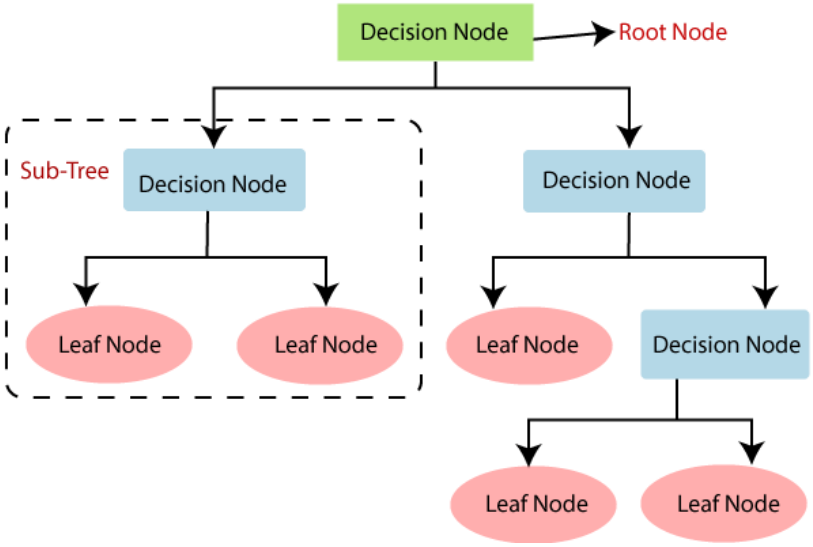

Figure 5.3.2.1 Decision tree

5.3.2.1 Applying algorithm and Predicting probability

Fitting the Decision tree model

Here we apply the algorithm and predict the probabilities for a tweet falling into either positive or negative class of both the features extracted using Bag of words technique and TF-IDF technique.

```
from sklearn.tree import DecisionTreeClassifier
dtc=DecisionTreeClassifier(criterion='entropy',random_state=1)
```

```
# fitiing decision tree model for bag of wwords feature
dtc.fit(x_train_bgwords,y_train_bgwords)
# predicting the probabilities for a tweet falling into either negative or positive class
predict_bgwords_dt=dtc.predict_proba(x_valid_bgwords)
predict_bgwords_dt
```

```
array([[1., 0.],
       [1., 0.],
       [1., 0.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]])
```

```
# fitting decision tree model for tfidf features
dtc.fit(x_train_tfidf,y_train_tfidf)
# predicting the probabilities for a tweet falling into either negative or positive class
predict_tfidf_dtc=dtc.predict_proba(x_valid_tfidf)
predict_tfidf_dtc
```

```
array([[1., 0.],
       [1., 0.],
       [1., 0.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]])
```

Figure 5.3.2.1.1 Applying algorithm

5.3.2.2 F1 Score Calculation

Here we calculate the F1 score of both the techniques

```
# calculating f1 score
predict3=predict_bgwords_dt[:,1]>=0.3
predict3=predict3.astype(np.int)
predict3
dtc_bgw_score=f1_score(y_valid_bgwords,predict3)
print("F1 score of decision tree with bag of words features is:",dtc_bgw_score)
```

```
F1 score of decision tree with bag of words features is: 0.5141776937618148
```

```
# calculating f1 score
predict4=predict_tfidf_dtc[:,1]>=0.3
predict4=predict4.astype(np.int)
predict4
dtc_tfidf_score=f1_score(y_valid_tfidf,predict4)
print("F1 score of Decision tree with TFIDF features is:",dtc_tfidf_score)
```

```
F1 score of Decision tree with TFIDF features is: 0.5498821681068342
```

Figure 5.3.2.2.1 F1 score

5.4 MODEL COMPARISION

Now, we compare the different models we have applied on our dataset with different word embedding techniques.

```
# comparing bag of words f1 score on different models used
models=['Logistic Regression','Decision Tree']
score1=[logist_bgwords_score,dtc_bgw_score]
compare1=pd.DataFrame({'Model':models,'F1Score for BagOfWords':score1},index=[i for i in range(1,3)])
compare1
```

	Model	F1Score for BagOfWords
1	Logistic Regression	0.572135
2	Decision Tree	0.514178

```
# comparing tfidf f1 score on different models used
models_=['Logistic Regression','Decision Tree']
score2=[logis_tfidf_score,dtc_tfidf_score]
compare2=pd.DataFrame({'Model':models_,'F1Score for TFIDF':score2},index=[i for i in range(1,3)])
compare2
```

	Model	F1Score for TFIDF
1	Logistic Regression	0.586207
2	Decision Tree	0.549882

Figure 5.4.1 Decision tree vs Logistic Regression

Here, we can see that best F1 score is obtained by Logistic regression. Therefore best possible model from both features is Logistic Regression

Now we compute the best feature extraction technique for Logistic Regression

```
tech=['Bag of Words','TF-IDF']
score3=[logist_bgwords_score,logis_tfidf_score]
compare3=pd.DataFrame({'Technique':tech,'F1_Score':score3},index=[i for i in range(1,3)])
compare3
```

	Technique	F1_Score
1	Bag of Words	0.572135
2	TF-IDF	0.586207

Figure 5.4.2 Bag of words vs TF-IDF

Hence it is clear that the best F1 score is obtained by the Logistic Regression model using TF-IDF features

5.5 PREDICTING RESULT

```
# Prediction the results
pred_res=logis_reg.predict_proba(tfidf[31962:])
pred_res=pred_res[:,1]>=0.3
pred_res=pred_res.astype(np.int)
test['label']=pred_res
test[['id','label']].to_csv('result.csv',index=False)
```

```
df=pd.read_csv('result.csv')
```

```
df
```

	id	label
0	31963	0
1	31964	0
2	31965	0
3	31966	0
4	31967	0
...	...	...
17192	49155	1
17193	49156	0
17194	49157	0
17195	49158	0
17196	49159	0

17197 rows × 2 columns

Figure 5.5.1 Result

From the above output we can see that our Logistic Regression model with TF-IDF features predicts whether a tweets falls into the category of positive(label:0) or negative(label:1)

CONCLUSION

In this project we tried to show the basic way of classifying tweets into positive or negative category by performing different feature extraction techniques and applying those features to different machine learning models i.e Logistic Regression and Decision Tree . Performance measure of each method is calculated using F1 score and compared. Hence after comparison, we used the model which is more efficient and gave more accurate results.

REFERENCES

[1] https://en.wikipedia.org/wiki/Python_(programming_language)

[2] https://www.kaggle.com/learn/intro-to-machine-learning

[3] https://sproutsocial.com/insights/twitter-data/