## YOUTUBE COMMENT ANALYSIS USING NATURAL LANGUAGE PROCESSING

### A PROJECT REPORT

***Submitted by***

## PRABHU M [REGISTER NO:211418104192] KUMARESH P [REGISTER NO:211418104134] SANJAY KUMAR R [REGISTER NO:211418104227]

***in partial fulfillment for the award of the degree of***

## BACHELOR OF ENGINEERING

**IN**

### COMPUTER SCIENCE AND ENGINEERING



**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

### MAY 2022

**PANIMALAR ENGINEERING COLLEGE**

**(An Autonomous Institution, Affiliated to Anna University, Chennai)**

## BONAFIDE CERTIFICATE

Certified that this project report **“YOUTUBE COMMENT ANALYSIS USING NATURAL LANGUAGE PROCESSING”** is the bonafide work of “**PRABHU M(211418104192), KUMARESH P(211418104134), SANJAY KUMAR**

**R(211418104227)”** who carried out the project work under my supervision.

### SIGNATURE SIGNATURE

**Dr.S.MURUGAVALLI,M.E.,Ph.D., Mr.M.KRISNAMOORTHY,M.E., HEAD OF THE DEPARTMENT SUPERVISOR**

### PROFESSOR PROFESSOR

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE, NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on...........................

**INTERNAL EXAMINER EXTERNAL EXAMINER**

## DECLARATION BY THE STUDENT

We, **PRABHU M (211418104192), KUMARESH P (211418104134),**

**SANJAY KUMAR R(211418104227)** hereby declare that this project

report titled , under the guidance of **Mr. M. KRISHNAMOORTHY** is the

orginial work done by us and we have not plagiarized or submitted to any other degree in any university by us.

**ACKNOWLEDGEMENT**

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr. P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt. C.VIJAYARAJESWARI**, **Dr. C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr. SARANYASREE SAKTHI KUMAR**

**B.E., M.B.A., Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr. K.MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,**

for the support extended throughout the project.

We would like to thank my **Mr. KRISHNAMOORTHY M , M.E., M.B.A.,** and all the faculty members of the Department of CSE for their advice and encouragement for the successful completion of the project.

**PRABHU M KUMARESH P SANJAY KUMAR R**

## ABSTRACT

With the rise of social media and publicly available data, opinion mining is more accessible than ever. It is valuable for content creators, companies and advertisers to gain insights into what users think and feel. Commentator (YouTube comment analysis using Natural Language Processing) examines comments on YouTube videos, and builds a Naive Bayes Classifier to automatically determine their sentiment. The Naive Bayes Classifier categorizes the sentiment into 6 categories of human emotions : Sad, Angry, Surprise, Fear, Happy and Love. The classification of comments into these categories expresses people’s opinions more accurately rather than classifying it as positive or negative comments. Based on analysis done by the Commentator for a video, viewers also can understand the nature of the video and also can prevent watching that particular video if they are not interested. Commentator also can spot promotion done by the content creators or the inaccurate facts mentioned by the content creators through the analysis of the YouTube Video Comments.

## TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | i |
|  | **LIST OF TABLES** | v |
|  | **LIST OF FIGURES** | vi |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview | 2 |
|  | 1.2 Problem Definition | 3 |
| **2.** | **SYSTEM ANALYSIS** | 4 |
|  | 2.1 Existing System | 5 |
|  | 2.2 Proposed system | 5 |
|  | 2.3 Requirement Analysis and Specification | 5 |
|  | 2.3.1 Input Requirements | 6 |
|  | 2.3.2 Output Requirements | 6 |
|  | 2.3.3 Functional Requirements | 6 |
|  | 2.3.4 Development Environment | 7 |
|  | 2.4 Software Description | 7 |
| **3.** | **SYSTEM DESIGN** | 16 |
|  | 3.1 UML Diagrams | 17 |
|  | 3.1.1 Use Case Diagram | 17 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | 3.1.2 Activity Diagram | 18 |
| **4.** | **SYSTEM ARCHITECTURE** | 19 |
|  | 4.1 System Architecture Overview | 20 |
|  | 4.2 Libraries/APIs Used | 20 |
|  | 4.2.1 Pandas | 20 |
|  | 4.2.2 Natural Language Toolkit (NLTK) | 26 |
|  | 4.2.3 Scikit-Learn | 28 |
|  | 4.2.4 YouTube API | 30 |
|  | 4.3 Module Design Specification | 35 |
|  | 4.3.1 Dataset Importing Module | 35 |
|  | 4.3.2 NLTK Libraries Importing Module | 35 |
|  | 4.3.3 Dataset Cleaning Module | 36 |
|  | 4.3.4 Dataset Transformation Module | 36 |
|  | 4.3.5 Model Building and Training Module | 36 |
|  | 4.3.6 YouTube Comment Extraction Module | 37 |
|  | 4.3.7 Sentiment Prediction Module | 37 |
|  | 4.3.8 Performance Analysis Module | 37 |
|  | 4.4 Algorithm Description | 38 |
|  | 4.4.1 Multinomial Naive Bayes Algorithm | 38 |
| **5.** | **SYSTEM IMPLEMENTATION** | 44 |
|  | 5.1 Module Coding | 45 |
| **6.** | **PERFORMANCE ANALYSIS** | 57 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | 6.1 Sample Training/Testing Dataset | 58 |
|  | 6.2 Performance Parameters | 59 |
| **7.** | **CONCLUSION** | 63 |
|  | 7.1 Conclusion | 64 |
|  | 7.2 Future Enhancements | 65 |
|  | **REFERENCES** | 66 |

### LIST OF TABLES

|  |  |  |
| --- | --- | --- |
| **Table No** | **Name** | **Page No** |
| 4.2.2.1 | NLTK Modules | 28 |
| 4.2.4.1 | YouTube API  CommentsThreads: list | 31 |
| 6.1.1 | Sample Training/Testing  Dataset | 58 |
| 6.2.1 | True/False Positives and  Negatives | 59 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Figure No** | **Name** | **Page No** |
| 2.4.1 | Sample Google Collab  Code-1 | 7 |
| 2.4.2 | Sample Google Collab  Code-2 | 8 |
| 2.4.3 | Data Visualization | 9 |
| 2.4.4 | Code Cell | 10 |
| 2.4.5 | System Aliases | 11 |
| 2.4.6 | String Interpolation | 12 |
| 2.4.7 | Code Magics-1 | 12 |
| 2.4.8 | Code Magics-2 | 13 |
| 2.4.9 | Numpy Module | 13 |
| 2.4.10 | Exception Formatting | 14 |
| 3.1.1.1 | Use Case Diagram | 17 |
| 3.1.2.1 | Activity Diagram | 18 |
| 4.1 | System Architecture | 20 |
| 4.2.1.1 | Pandas Series | 22 |
| 4.2.1.2 | Pandas DataFrame | 24 |
| 5.1.1 | Training Dataset | 45 |
| 5.1.2 | Training Labels | 46 |
| 5.1.3 | Cleaned Training Dataset | 47 |
| 5.1.4 | Vectorized Training  Dataset | 48 |

|  |  |  |
| --- | --- | --- |
| 5.1.5 | Vectorized and Transformed Training  Dataset | 49 |
| 5.1.6 | Extracted YouTube Video  Comments | 51 |
| 5.1.7 | Cleaned YouTube Video  Comments | 52 |
| 5.1.8 | Vectorized and  Transformed YouTube Video Comments | 52 |
| 5.1.9 | Predicted Sentiments of YouTube Video  Comments | 53 |
| 5.1.10 | Predicted Sentiments of  Testing Dataset | 54 |
| 5.1.11 | Categorization of  Predicted Sentiments | 55 |
| 5.1.12 | Accuracy of Predicted  Sentiments | 56 |

# CHAPTER 1 INTRODUCTION

## OVERVIEW

Emotions and opinions play an essential role in human behaviour as we often make decisions based on other people’s experience and thoughts. Since everyone’s opinions are subjective and affected by their personal experiences and beliefs, learning about them helps to get a broader perspective. Public opinion has been of interest to organisations, companies, and politicians for a long time. The collective opinion of people can help predict who will win elections, generate knowledge about future trends and topics. It also helps determine general opinion about products and services, which helps marketing teams decide on a marketing strategy, improving existing or production of a new product, and improving customer support. Therefore, recognition of sentiment in various fields is of the essence.

Initially, people’s opinions were collected through surveys or questionnaires and manually reviewed. However, as the usage of the internet increased, people started expressing their opinion online more frequently. The boom of social media platforms in recent years has allowed its users to share all kinds of information and introduce a wider variety and ways to express their opinions. Blogs, discussion forums, reviews, comments and micro-blogging services, such as Twitter or Facebook, serve as rich data sources, composed of audio files, videos, images and opinions.

One such platform, YouTube, is currently one of the most popular social media platforms in the world. It allows its users to share their videos or view videos uploaded by other users and provide feedback. A user can provide feedback by supporting or rejecting a video by rating it positively or negatively, respectively. Users may further also provide textual feedback in the form of comments posted to videos. Each video can hold thousands of comments making YouTube a platform.

## PROBLEM DEFINITION

Commentator focuses on building a YouTube comment dataset, developing a sentiment classifier and performing sentiment analysis on the dataset. Sentiment analysis - sometimes referred to as opinion mining - is a natural language processing (NLP) technique used to determine textual data’s sentiment polarity. It analyses people’s sentiment and attitude towards products, digital content, organisations, events, issues and others. While the large volumes of opinion data can provide an in-depth understanding of overall sentiment, they require a lot of time to process. Not only is it time-consuming and challenging to review large quantities of texts, but some texts might also be long and complex, expressing reasoning for different sentiments, making it challenging to understand overall sentiment quickly.

Currently, YouTube allows comments of 10,000 characters in length, which is a considerable amount and can result in long texts. Naturally, this amount of data calls for tools to ease and automate the process of classification and sentiment extraction from a text. Text classification is a process of categorising texts into groups. It is applied in various domains, such as organisation of news articles, opinion mining of product reviews, spam filtering, document organisation in digital libraries, such as literature, social feeds, and more. There can be different ways to classify text - by its topic, whether it meets specific criteria or by its sentiment. Text sentiment classification is a process of determining the overall sentiment of some given text. This problem can either result in binary classification or multi-label classification, where binary classification produces two outputs, positive and negative, and multi-label classification produces more than two classification outputs. Common techniques to solve text sentiment classification problems often use traditional machine learning algorithms or deep-learning based approaches but Commentator use Multinomial Naive Bayes Model Approach to solve the text classification problem.

# CHAPTER 2 SYSTEM ANALYSIS

## EXISTING SYSTEM

The major flaw in the existing system (YouTube) is that the nature of the comments of a particular YouTube video is not displayed and to add fire to flame the dislike count of a particular YouTube video is removed by Google to stop spreading hatred and it is visible only to the content creators. The removal of dislike button means that the viewers need to go through the top comments of the YouTube video to get the gist of the nature of the video.

## PROPOSED SYSTEM

The proposed system is a Multinomial Naive Bayes Model with a classifier that classifies the comments of a YouTube video into 6 categories: Sad, Angry, Surprise, Fear, Happy and Love. Commentator thus reduces the hassle of viewers to go through each and every comments of the comment section to determine the nature of the video. Commentator analyses the nature of the video by extracting the comments with the help of a YouTube API which returns the different parameters of YouTube Video as a response from which comments are extracted. The Multinomial Naive Bayes classifier is trained with the training dataset of 9,000 values for the prediction of the accurate sentiment expressed in the comments.

## REQUIREMENT ANALYSIS AND SPECIFICATION

Requirement specification and analysis identify, analyze, and model the functionality or “what's” of a prospective software system. The requirements specification and analysis phase of a software project is the most important phase of software development and should not be omitted under

any condition.

## INPUT REQUIREMENTS

Input requirements of the Commentator are the training dataset and testing dataset. The model is first trained with the training dataset and tested with testing dataset for calculation of the model’s various performance parameters. The user is then required to enter the Video ID of the YouTube video for which he/she wants to know the nature.

## OUTPUT REQUIREMENTS

Output requirements include a Laptop with a minimum of 4 GB ram and internet speed of minimum 1 Mbps. All the computations of Commentator are done by a cloud Google Colaboratory system and the results are displayed as per different categories of comments of YouTube Video ID entered by the user.

## FUNCTIONAL REQUIREMENTS

The main functional requirement of Commentator is to have a stable internet connection to upload the training dataset and testing dataset to the Google Colaboratory platform and for the results to be fetched after the training and evaluation of the trained Multinomial Naive Bayes model. Another major functional requirement is to assign the training and testing dataset as per the resources allocated to you by Google Colaboratory to prevent the Cloud system from crashing. The Video ID entered by the user should also be valid in-order to fetch the comments through YouTube API.

* + 1. **DEVELOPMENT ENVIRONMENT** Operating System : Windows 10 Languages used : Python 3.9

Tools : Google Colaboratory

Libraries : NLTK. Pandas, Scikit-Learn

API : YouTube API

## SOFTWARE DESCRIPTION

**GOOGLE COLABORATORY INTRODUCTION**

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

* + - Zero configuration required
    - Access to GPUs free of charge
    - Easy sharing

## GETTING STARTED

The document you are reading is not a static web page, but an interactive environment called a Colab notebook that lets you write and execute code.

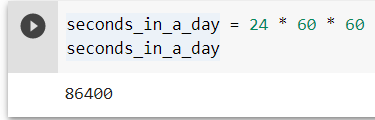
For example, here is a code cell with a short Python script that computes a value, stores it in a variable, and prints the result:

Figure 2.4.1 Sample Google Collab Code-1

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing. Variables that you define in one cell can later be used in other cells:

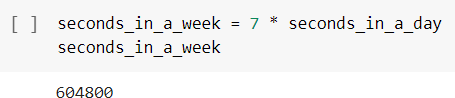


Figure 2.4.2 Sample Google Collab Code-2

Colab notebooks allow you to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more. When you create your own Colab notebooks, they are stored in your Google Drive account. User can easily share your Colab notebooks with co-workers or friends, allowing them to comment on your notebooks or even edit them. To learn more, see [Overview of](https://colab.research.google.com/notebooks/basic_features_overview.ipynb) [Colab](https://colab.research.google.com/notebooks/basic_features_overview.ipynb). To create a new Colab notebook user can use the File menu.

## DATA SCIENCE

With Colab user can harness the full power of popular Python libraries to analyze and visualize data. The code cell below uses numpy to generate some random data, and uses matplotlib to visualize it. To edit the code, just click the cell and start editing.

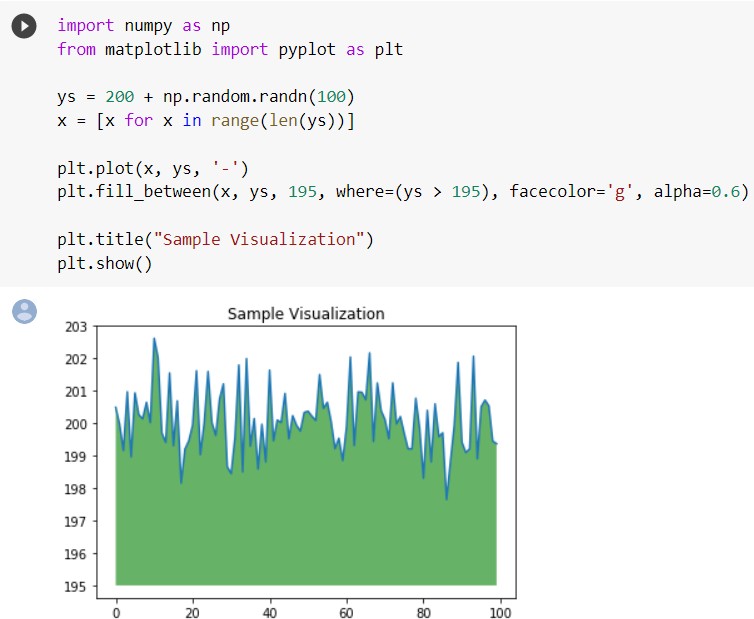


Figure 2.4.3 Data Visualization

User can import your own data into Colab notebooks from your Google Drive account, including from spreadsheets, as well as from Github and many other sources.

## MACHINE LEARNING

With Colab user can import an image dataset, train an image classifier on it, and evaluate the model, all in just [a few lines of code](https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/quickstart/beginner.ipynb). Colab notebooks execute code on Google's cloud servers, meaning user can leverage the power of Google hardware, including [GPUs and TPUs](https://colab.research.google.com/?utm_source=scs-index&using-accelerated-hardware), regardless of the power of your machine. All you need is a browser.

Colab is used extensively in the machine learning community with applications including:

* + - Getting started with TensorFlow
    - Developing and training neural networks
    - Experimenting with TPUs
    - Disseminating AI research
    - Creating tutorials

## CODE CELL

Below is a code cell. Once the toolbar button indicates CONNECTED, click in the cell to select it and execute the contents in the following ways:

* + - Click the Play icon in the left gutter of the cell;
    - Type Cmd/Ctrl+Enter to run the cell in place;
    - Type Shift+Enter to run the cell and move focus to the next cell (adding one if none exists); or
    - Type Alt+Enter to run the cell and insert a new code cell immediately below it.

There are additional options for running some or all cells in the Runtime menu.

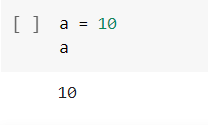


Figure 2.4.4 Code Cell

## ADDING AND MOVING CELLS

User can add new cells by using the + CODE and + TEXT buttons that show when you hover between cells. These buttons are also in the toolbar above the notebook where they can be used to add a cell below the currently selected cell. User can move a cell by selecting it and clicking Cell Up or Cell Down in the top toolbar.

Consecutive cells can be selected by "lasso selection" by dragging from outside one cell and through the group. Non-adjacent cells can be selected concurrently by clicking one and then holding down Ctrl while clicking another. Similarly, using Shift instead of Ctrl will select all intermediate cells.

## SYSTEM ALIASES

Google Collab includes shortcuts for common operations, such as ls:



Figure 2.4.5 System Aliases

That !ls probably generated a large output. User can select the cell and clear the output by either:

1. Clicking on the clear output button (x) in the toolbar above the cell; or
2. Right clicking the left gutter of the output area and selecting "Clear output" from the context menu.

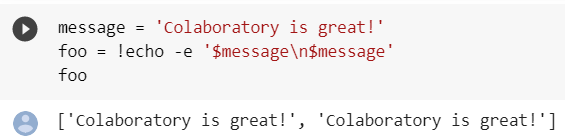
Execute any other process using ! with string interpolation from python variables, and note the result can be assigned to a variable:

Figure 2.4.6 String Interpolation

## MAGICS

Colaboratory shares the notion of magics from Jupyter.

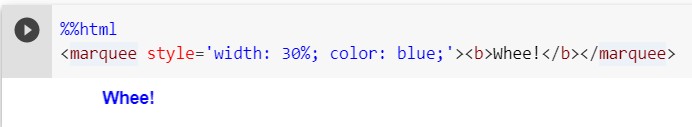


Figure 2.4.7 Code Magics-1



Figure 2.4.8 Code Magics-2

## AUTOMATIC COMPLETIONS AND EXPLORING CODE

Colab provides automatic completions to explore attributes of Python objects, as well as to quickly view documentation strings. As an example, first run the following cell to import the [numpy](http://www.numpy.org/) module.



Figure 2.4.9 Numpy Module

If you now insert your cursor after np and press Period(.), you will see the list of available completions within the np module. Completions can be opened again by using Ctrl+Space. The documentation can be opened again using Ctrl+Shift+Space or user can view the documentation for method by mouse hovering over the method name.

When hovering over the method name the Open in tab link will open the documentation in a persistent pane. The View source link will navigate to the source code for the method.

## EXCEPTION FORMATTING

Exceptions are formatted nicely in Colab outputs:

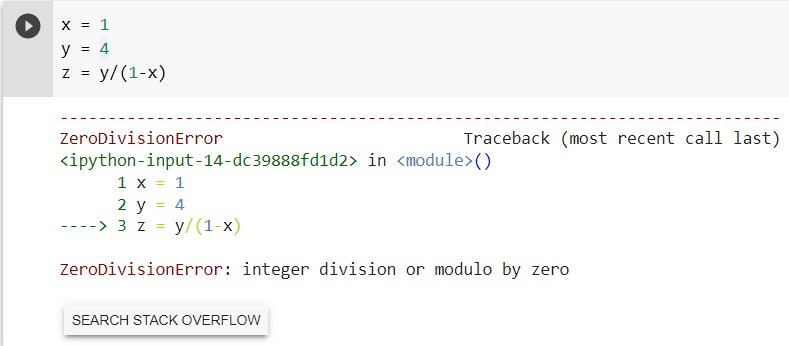


Figure 2.4.10 Exception Formatting

## INTEGRATION WITH DRIVE

Colaboratory is integrated with Google Drive. It allows you to share, comment, and collaborate on the same document with multiple people:

* + - The SHARE button (top-right of the toolbar) allows you to share the notebook and control permissions set on it.
    - File->Make a Copy creates a copy of the notebook in Drive.
    - File->Save saves the File to Drive. File->Save and checkpoint pins the version so it doesn't get deleted from the revision history.
    - File->Revision history shows the notebook's revision history.

## COMMENTING ON A CELL

User can comment on a Colaboratory notebook like you would on a Google Document. Comments are attached to cells, and are displayed next to the cell they

refer to. If you have comment-only permissions, you will see a comment button on the top right of the cell when you hover over it.

If you have edit or comment permissions user can comment on a cell in one of three ways:

* + - Select a cell and click the comment button in the toolbar above the top- right corner of the cell.
    - Right click a text cell and select Add a comment from the context menu.
    - Use the shortcut Ctrl+Shift+M to add a comment to the currently selected cell.

User can resolve and reply to comments, and user can target comments to specific collaborators by typing +[email address] (e.g., +user@domain.com). Addressed collaborators will be emailed.

The Comment button in the top-right corner of the page shows all comments attached to the notebook.

# CHAPTER 3 SYSTEM DESIGN

## UML DIAGRAMS

* + 1. **USE CASE DIAGRAM**

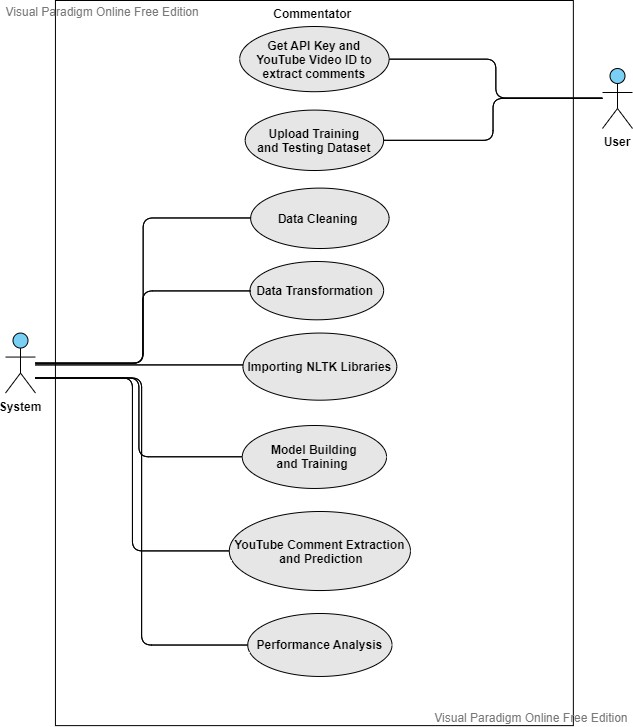
A use case illustrates a unit of functionality provided by the system. The main purpose of the use-case diagram is to help development teams visualize the functional requirements of a system, including the relationship of "actors" (human beings who will interact with the system) to essential processes, as well as the relationships among different use cases. The use case has two actors: user and server. User gives the image as input and server performs the operation.

Figure 3.1.1.1 Use Case Diagram

## ACTIVITY DIAGRAM

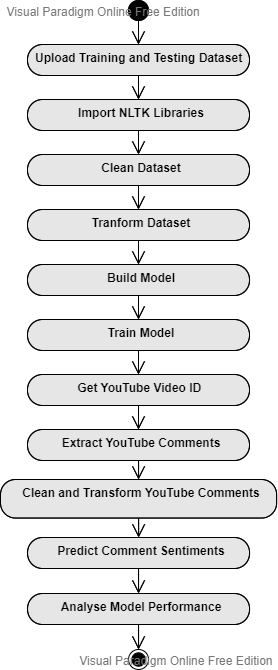
An activity diagram is a behavioral diagram i.e. it depicts the behavior of a system. An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.

Figure 3.1.2.1 Activity Diagram

# CHAPTER 4 SYSTEM ARCHITECTURE

## SYSTEM ARCHITECTURE OVERVIEW

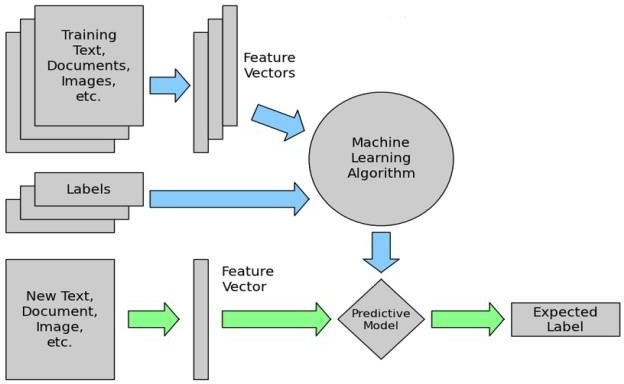


Figure 4.1 System Architecture

## LIBRARIES/APIS USED

Commentator works with the help of following libraries and API’s

* + - Pandas
    - Natural Language Toolkit (NLTK)
    - Scikit-Learn
    - YouTube API

## PANDAS

Pandas is an open-source library that is built on top of NumPy library. It is a Python package that offers various data structures and operations for manipulating numerical data and time series. It is mainly popular for importing and analyzing data much easier. Pandas is fast and it has high-performance & productivity for users.

## ADVANTAGE OF PANDAS

* + - * Fast and efficient for manipulating and analyzing data.
      * Data from different file objects can be loaded.
      * Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data.
      * Size mutability: columns can be inserted and deleted from Data Frame and higher dimensional objects.
      * Data set merging and joining.
      * Flexible reshaping and pivoting of data sets.
      * Provides time-series functionality.
      * Powerful group by functionality for performing split-apply-combine operations on data sets.

Pandas generally provide two data structure for manipulating data, They are:

* + - * Series
      * DataFrame

## PANDAS SERIES

Pandas series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called index. Pandas Series is nothing but a column in an excel sheet. Labels need not be unique but must be a hashable type. The object supports both integer and label-based indexing and provides a host of methods for performing operations involving the index.

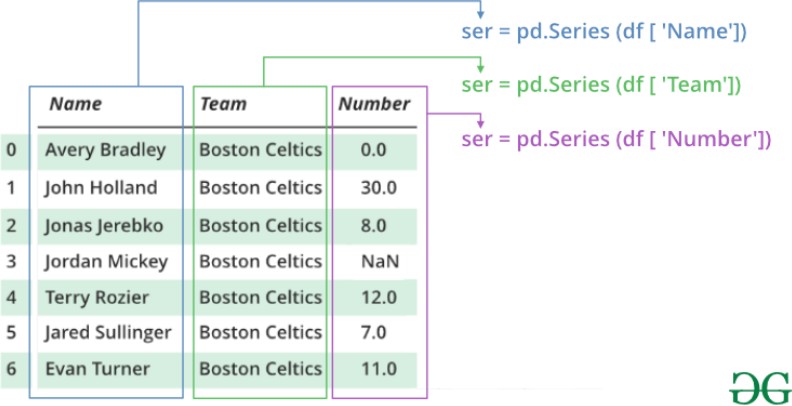


Figure 4.2.1.1 Pandas Series

## CREATING A PANDAS SERIES

In the real world, a Pandas Series will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file. Pandas Series can be created from the lists, dictionary, and from a scalar value etc.

Example:

import pandas as pd import numpy as np

# Creating empty series ser = pd.Series()

print(ser)

# simple array

data = np.array([‘p’, ‘a’, ’n’, ‘d’, ‘a’, ‘s’])

ser = pd.Series(data) print(ser)

Output:

Series([], dtype: float64)

1. p
2. a
3. n
4. d
5. a
6. sdtype: object

## PANDAS DATAFRAME

Pandas DataFrame is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the data, rows, and columns.

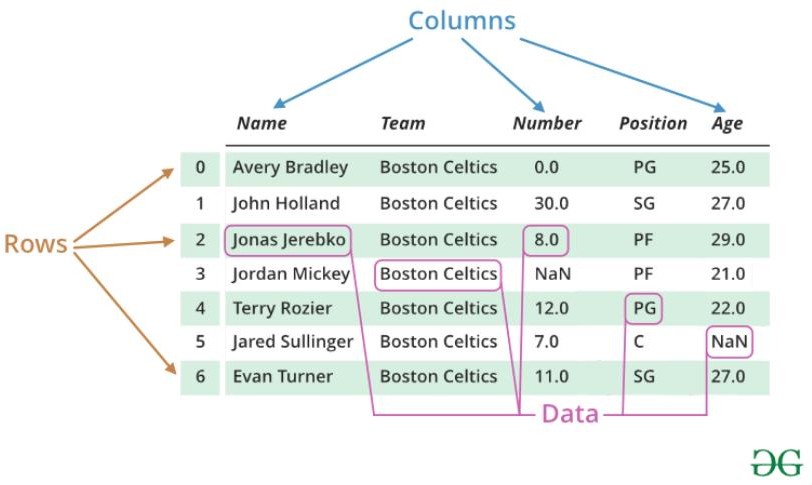


Figure 4.2.1.2 Pandas DataFrame

## CREATING A PANDAS DATAFRAME

In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file. Pandas DataFrame can be created from the lists, dictionary, and from a list of dictionary etc.

Example:

import pandas as pd

# Calling DataFrame constructor df = pd.DataFrame()

print(df)

# list of strings

lst = [‘Python’, ‘For’, ‘Data’, ‘Science’, ] # Calling DataFrame constructor on list

df = pd.DataFrame(lst) print(df)

Output:

Empty DataFrame Columns: [] Index: []

1. Python
2. For
3. Data
4. Science

## PANDAS FOR DATA SCIENCE

Pandas is generally used for data science but have you wondered why? This is because pandas is used in conjunction with other libraries that are used for data science. It is built on the top of the NumPy library which means that a lot of structures of NumPy are used or replicated in Pandas. The data produced by Pandas is often used as input for plotting functions of Matplotlib, statistical analysis in SciPy, machine learning algorithm in Scikit-learn.

Pandas program can be run from any text editor but it is recommended to use Jupyter Notebook for this as Jupyter given the ability to execute code in a particular cell rather than executing the entire file. Jupyter also provides an easy way to visualize pandas dataframe and plots.

## NATURAL LANGUAGE TOOLKIT (NLTK)

The NLTK module is a huge toolkit designed to help you with the entire Natural Language Processing (NLP) approach. NLTK will provide you with everything from splitting paragraphs to sentences, splitting words, identifying the part of speech, highlighting themes, and even helping your machine understand what the text is about.

* + - * Corpus — The body of the text, singular. Corpora is its plural. Example: A collection of medical journals .
      * Lexicon — vocabulary and its meaning. For example: English dictionary. However, considering that there are different thesaurus in each field. For example, for financial investors, the first meaning of the word Bull is the person who is confident in the market. Compared with the “common English vocabulary”, the first meaning of the word is animal. Therefore, financial investors, doctors, children, mechanics, etc. all have a special vocabulary.
      * Token — Each “entity” is part of a rule based split. For example, when a sentence is “split” into words, each word is a tag. If you split a paragraph into sentences, each sentence can also be a marker.

## DATA PREPROCESSING USING NLTK

The process of cleaning unstructured text data, so that it can be used to predict, analyze, and extract information. Real-world text data is unstructured, inconsistent. So, Data preprocessing becomes a necessary step.

The various Data Preprocessing methods are:

* + - * Tokenization
      * Frequency Distribution of Words
      * Filtering Stop Words
      * Stemming
      * Lemmatization
      * Parts of Speech(POS) Tagging
      * Name Entity Recognition
      * WordNet

## NLTK MODULES

Natural Language Toolkit (NLTK) is suite of libraries and programs in Python for Natural Language Processing Tasks. It is one of the most widely used NLP Python libraries.It can perform various NLP tasks like tokenization, stemming, POS tagging, lemmatization and classification to name a few.

A summary of popular modules of NLTK is given in the table below.

|  |  |  |
| --- | --- | --- |
| **S.No** | **Language Processing Task** | **NLTK module** |
| 1 | Accessing Corpora | nltk.corpus |
| 2 | String Processing | nltk.tokenize, nltk.stem |
| 3 | Collocation discovery | nltk.collocations |
| 4 | Part-of-speech tagging | nltk.tag |
| 5 | Classification | nltk.classify, nltk.cluster |
| 6 | Chunking | nltk.chunk |
| 7 | Parsing | nltk.parse |
| 8 | Semantic interpretation | nltk.sem, nltk.inference |
| 9 | Evaluation metrics | nltk.metrics |

|  |  |  |
| --- | --- | --- |
| **S.No** | **Language Processing Task** | **NLTK module** |
| 10 | Probability and Estimation | nltk.probability |
| 11 | Applications | nltk.app, nltk.chat |
| 12 | Linguistic fieldwork | nltk.toolbox |

Table 4.2.2.1 NLTK Modules

## SCIKIT-LEARN

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.

Sci-kit learn features various classification, regression, and clustering algorithms including support vector machines, random forests, gradient boosting, *k*-means, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before user can use scikit-learn. This stack includes:

* + - * NumPy: Base n-dimensional array package
      * SciPy: Fundamental library for scientific computing
      * Matplotlib: Comprehensive 2D/3D plotting
      * IPython: Enhanced interactive console
      * Sympy: Symbolic mathematics
      * Pandas: Data structures and analysis

## ADVANTAGES OF SCIKIT-LEARN

* + - * Clustering: for grouping unlabeled data such as KMeans.
      * Cross-Validation: for estimating the performance of supervised models on unseen data.
      * Datasets: for test datasets and for generating datasets with specific properties for investigating model behavior.
      * Dimensionality Reduction: for reducing the number of attributes in data for summarization, visualization, and feature selection such as Principal component analysis.
      * Ensemble methods: for combining the predictions of multiple supervised models.
      * Feature extraction: for defining attributes in image and text data.
      * Feature selection: for identifying meaningful attributes from which to create supervised models.
      * Parameter Tuning: for getting the most out of supervised models.
      * Manifold Learning: For summarizing and depicting complex multi- dimensional data.
      * Supervised Models: a vast array not limited to generalized linear models, discriminate analysis, naive Bayes, lazy methods, neural networks, support vector machines, and decision trees.

## YOUTUBE API

The YouTube Application Programming Interface (YouTube API) allows developers to access video statistics and [YouTube channels](https://en.wikipedia.org/wiki/Youtube_channel) data via two types of calls, [REST](https://en.wikipedia.org/wiki/Representational_State_Transfer) and [XML-RPC](https://en.wikipedia.org/wiki/XML-RPC). Google describe YouTube's API resources as "APIs and Tools that let you bring the YouTube experience to your webpage, application or device.

## YOUTUBE API REQUIREMENTS

* + - * You need a [Google Account](https://www.google.com/accounts/NewAccount) to access the Google API Console, request an API key, and register your application.
      * Create a project in the [Google Developers Console](https://console.developers.google.com/) and [obtain](https://developers.google.com/youtube/registering_an_application) [authorization credentials](https://developers.google.com/youtube/registering_an_application) so your application can submit API requests.
      * After creating your project, make sure the YouTube Data API is one of the services that your application is registered to use:
      * Go to the [API Console](https://console.developers.google.com/) and select the project that you just registered.
      * Visit the [Enabled APIs page](https://console.developers.google.com/apis/enabled). In the list of APIs, make sure the status is ON for the YouTube Data API v3.
      * If your application will use any API methods that require user authorization, read the [authentication](https://developers.google.com/youtube/v3/guides/authentication) guide to learn how to implement OAuth 2.0 authorization.
      * Select a [client library](https://developers.google.com/youtube/v3/libraries) to simplify your API implementation.
      * Familiarize yourself with the core concepts of the JSON (JavaScript Object Notation) data format. JSON is a common, language-independent data format that provides a simple text representation of arbitrary data structures.

## YOUTUBE API COMMENTTHREADS: LIST

CommentThreads: list is a part of YouTube API and returns a list of comment threads that match the API request parameters.The table below shows common use cases for this method. User can click on a use case name to load sample parameter values in the APIs Explorer.

|  |  |
| --- | --- |
|  | Use cases |
| [list (by](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [video ID)](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) | This example retrieves all comment threads associated with a particular video. The request's [videoId](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list#videoId) parameter identifies the video. |
| [list (by](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [channel](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [ID)](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) | This example retrieves all comment threads about the specified channel. The request's [channelId](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list#channelId) parameter identifies the channel. The response does not include comments left on videos that the channel uploaded. |
| [list (all](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [threads](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [related to](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [channel](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) [ID)](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list_9c4da36c2baa5e290899ede3a36931fe3ef44370028172884fc2c1fce5cde692.frame) | This example retrieves all comment threads associated with a particular channel. The response could include comments about the channel or about the channel's videos. The  request's [allThreadsRelatedToChannelId](https://developers-dot-devsite-v2-prod.appspot.com/youtube/v3/docs/commentThreads/list#allThreadsRelatedToChannelId) parameter identifies the channel. |

Table 4.2.4.1 YouTube API CommentsThreads: list

## REQUIRED PARAMETERS OF YOUTUBE API RESPONSE

part: string

The part parameter specifies a comma-separated list of one or more commentThread resource properties that the API response will include.

The following list contains the part names that user can include in the parameter value:

* + - * id
      * replies
      * snippet

## FILTERS OF YOUTUBE API RESPONSE

allThreadsRelatedToChannelId: string

The allThreadsRelatedToChannelId parameter instructs the API to return all comment threads associated with the specified channel. The response can include comments about the channel or about the channel's videos.

channelId: string

The channelId parameter instructs the API to return comment threads containing comments about the specified channel. (The response will not include comments left on videos that the channel uploaded.)

id: string

The id parameter specifies a comma-separated list of comment thread IDs for the resources that should be retrieved.

videoId: string

The videoId parameter instructs the API to return comment threads associated with the specified video ID.

maxResults: unsigned integer

The maxResults parameter specifies the maximum number of items that should be returned in the result set. This parameter is not supported for use in conjunction

with the [id](https://developers.google.com/youtube/v3/docs/commentThreads/list#id) parameter. Acceptable values are 1 to 100, inclusive. The default value is 20.

moderationStatus: string

This parameter can only be used in a properly [authorized request](https://developers.google.com/youtube/v3/guides/authentication). Set this parameter to limit the returned comment threads to a particular moderation state. This parameter is not supported for use in conjunction with the [id](https://developers.google.com/youtube/v3/docs/commentThreads/list#id) parameter. The default value is published.

Acceptable values are:

* + - * heldForReview – Retrieve comment threads that are awaiting review by a moderator. A comment thread can be included in the response if the top- level comment or at least one of the replies to that comment are awaiting review.
      * likelySpam – Retrieve comment threads classified as likely to be spam. A comment thread can be included in the response if the top-level comment or at least one of the replies to that comment is considered likely to be spam.
      * published – Retrieve threads of published comments. This is the default value. A comment thread can be included in the response if its top-level comment has been published.

order: string

The order parameter specifies the order in which the API response should list comment threads. This parameter is not supported for use in conjunction with the [id](https://developers.google.com/youtube/v3/docs/commentThreads/list#id) parameter. Valid values are:

time - Comment threads are ordered by time. This is the default behavior. relevance - Comment threads are ordered by relevance.

pageToken: string

The pageToken parameter identifies a specific page in the result set that should be returned. In an API response, the nextPageToken property identifies the next page of the result that can be retrieved. This parameter is not supported for use in conjunction with the [id](https://developers.google.com/youtube/v3/docs/commentThreads/list#id) parameter.

searchTerms: string

The searchTerms parameter instructs the API to limit the API response to only contain comments that contain the specified search terms. This parameter is not supported for use in conjunction with the [id](https://developers.google.com/youtube/v3/docs/commentThreads/list#id) parameter.

textFormat: string

Set this parameter's value to html or plainText to instruct the API to return the comments left by users in html formatted or in plain text. The default value is html.

Acceptable values are:

* + - * html – Returns the comments in HTML format. This is the default value.
      * plainText – Returns the comments in plain text format.

## YOUTUBE API RESPONSE STRUCTURE

If successful, method returns a response body with the following structure:

{

"kind": "youtube#commentThreadListResponse", "etag": *etag*,

"nextPageToken": *string*, "pageInfo": { "totalResults": *integer*, "resultsPerPage": *integer*

},

"items": [

[commentThread Resource](https://developers.google.com/youtube/v3/docs/commentThreads#resource)

]

}

## MODULE DESIGN SPECIFICATION

* + - Dataset Importing Module
    - NLTK Libraries Importing Module
    - Dataset Cleaning Module
    - Dataset Transformation Module
    - Model Building and Training Module
    - YouTube Comment Extraction Module
    - Sentiment Prediction Module
    - Performance Analysis Module

## 4.3.1 DATASET IMPORTING MODULE

The dataset importing module is the first step where the training dataset and testing dataset for the Multinomial Naive Bayes Model are uploaded in the Google Colaboratory and these files need to uploaded everytime the runtime is restarted or in other words the particular window is closed. The imported data is classified as Comment and Emotion with the help of Pandas DataFrame.

## NLTK LIBRARIES IMPORTING MODULE

The NLTK Libraries Importing Module is where all the NLTK libraries required for data preprocessing of both the training and testing dataset are imported. The Libraries imported are NLTK RegexpTokenizer to tokenize the words in sentences of the training and testing datasets, NLTK PorterStemmer to stem the tokens obtained from the sentences of the training and testing datasets and NLTK Stopwords Remover to remove the common stopwords of the English Language from the tokenized stemmed words of the training and testing datasets.

## DATASET CLEANING MODULE

The Dataset Cleaning Module is where the imported NLTK Libraries such as NLTK RegexpTokenizer, NLTK PorterStemmer and NLTK Stopwords Remover are used to clean both the training and testing dataset. The datasets are first converted to lower case and converted to tokens by the use of lower() and tokenize() functions. The stopwords are removed followed by the stemming of tokens in training and testing dataset. Finally, the clean training and testing datasets are sent to the next module for Transformation of Data.

## DATASET TRANSFORMATION MODULE

The Dataset Transformation Module is where the training and testing datasets are transformed into vectors to be trained and tested by the Multinomial Naive Bayes Model. The CountVectorizer and TfidfTransformer are imported from the scikit- learn(sklearn) library for which the objects are first created. The clean training and testing datasets are converted into vectors by the means of CountVectorizer and the terms are given importance or ignored by the means of TfidfTransformer. The fit\_transform() method of both CountVectorizer and TfidfTransformer is used for the transformation of training data and transform() method of both CountVectorizer and TfidfTransformer are used for transformation of testing data.

## MODEL BUILDING AND TRAINING MODULE

The Model Building Module is where the Multinomial Naive Bayes Model is built by importing MultinomialNB from the scikit-learn(sklearn) library. The object is created for MultinomialNB that is the model is successfully built. The fit() method is used to train the model with the cleaned and vectorized training dataset and labelled dataset and system resources are utilized by the model to be

trained such as RAM and Disk Space and after the model is trained it will able to classify the nature of the YouTube comments into one of these categories: Sad, Happy, Angry, Fear, Love and Surprise.

## YOUTUBE COMMENT EXTRACTION MODULE

The YouTube Comment Extraction Module is where the comments of a particular YouTube video entered by the users are extracted for prediction of sentiment by the trained Multinomial Naive Bayes Model. The request to the YouTube API is created with the means of API Key and Video ID and Video ID is accepted as input from the user. From the YouTube API response, comments are extracted and stored in a list. The list is then cleaned and transformed into vectors for the trained Multinomial Naive Bayes Model to predict the sentiment.

## SENTIMENT PREDICTION MODULE

The Sentiment Prediction Module is where the sentiments of comments extracted by the YouTube API and classified and the sentiments of the testing dataset are predicted by the trained Multinomial Naive Bayes Model. The cleaned and transformed testing dataset and extracted YouTube video comments are predicted by the model by the means of predict() method to which the datasets are passed as arguments.

## PERFORMANCE ANALYSIS MODULE

The Performance Analysis Module is where the model is assessed based on the various performance parameters. The performance parameters include Accuracy Score, Classification Report and Confusion Matrix. The various libraries required for accessing the performance of the trained Naive Bayes Model are imported from the scikit-learn(sklearn) library. The objects are created for the performance

paramteters and finally the various performance parameters of the trained Multinomial Naive Bayes Model are displayed.

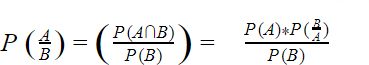
## ALGORITHM

* + 1. **MULTINOMIAL NAIVE BAYES ALGORITHM BAYES’ THEOREM**

Naive Bayes is based on Bayes’ theorem, where the adjective Naïve says that features in the dataset are mutually independent. Occurrence of one feature does not affect the probability of occurrence of the other feature. For small sample sizes, Naïve Bayes can outperform the most powerful alternatives. Being relatively robust, easy to implement, fast, and accurate, it is used in many different fields. For Example, Spam filtering in email, Diagnosis of diseases, making decisions about treatment, Classification of RNA sequences in taxonomic studies, to name a few. However, we have to keep in mind about the type of data and the type of problem to be solved that dictates which classification model we want to choose. Strong violations of the independence assumptions and non- classification problems can lead to poor performance. In practice, it is recommended to use different classification models on the same dataset and then consider the performance, as well as computational efficiency.

To understand how Naïve Bayes works, first, we have to understand the concept of Bayes’ rule. This probability model was formulated by Thomas Bayes (1701- 1761) and can be written as:





where,

PA= the prior probability of occurring A

PBA= the condition probability of B given that A occurs PAB= the condition probability of A given that B occurs PB= the probability of occuring B

The posterior probability, can be interpreted as: What is the revised probability of an event occurring after taking new information into consideration? It is a better reflection of the underlying truth of a data generating process because it includes more information.

## CONDITIONAL PROBABILIITY

The probability of one event A occurring when another event B with some relationship to A has already occurred is called conditional probability.



This expression is valid only when P(A) is greater than zero. Multinomial Naïve Bayes uses term frequency i.e. the number of times a given term appears in a document. Term frequency is often normalized by dividing the raw term frequency by the document length. After normalization, term frequency can be used to compute maximum likelihood estimates based on the training data to estimate the conditional probability.

## PRIOR PROBABILIITY

This [probability](https://www.mygreatlearning.com/blog/understanding-probability-distribution/) can be defined as the prior knowledge or belief i.e. the probability of an event computed before the collection of new data. This probability is revised as new information becomes available to produce more accurate results. If the prior observations are used to calculate the probability, we call it prior probability.

## BAG OF WORDS MODEL

Feature extraction and Selection are the most important sub-tasks in pattern classification. The three main criteria of good features are:

* Salient: The features should be meaningful and important to the problem
* Invariant: The features are resistant to scaling, distortion and orientation etc.
* Discriminatory: For training of classifiers, the features should have enough information to distinguish between patterns.

Bag of words is a commonly used model in [Natural Language Processing](https://www.mygreatlearning.com/blog/natural-language-processing-tutorial/). The idea behind this model is the creation of vocabulary that contains the collection of different words, and each word is associated with a count of how it occurs. Later, the vocabulary is used to create d-dimensional feature vectors.

### For Example:

**D1:** Each country has its own constitution

**D2:** Every country has its own uniqueness Vocabulary could be written as:

V= {each : 1, state : 1, has : 2, its : 2, own : 2, constitution : 1, every : 1, country

: 2,}

## SYSTEM ALGORITHM

* + - * The Pandas Library is imported as pd and read\_csv() of Pandas Library is used to read the training and testing datasets and classify it as Comment and Emotion accordingly based on the delimiter present in the training and testing datasets, the delimiter is also given as input to the read\_csv()
      * The training and testing datasets are now converted to list with the help of to\_list() method
      * From nltk.tokenize library RegexpTokenizer is imported, From
      * nltk.stem.porter library PorterStemmer is imported, From
      * nltk.corpus library stopwords is imported and the stopwords are downloaded by the means download() where stopwords is passed as parameter
      * The Tokenizer, Stemmer and Stopwords are intialized as tokenizer=RegexpTokenizer(r'\w+'), en\_stopwords=set(stopwords.words('english')) and

ps=PorterStemmer()

* + - * A clean\_text function is initialized which takes uncleaned data as input and cleaned data as output. It consists of the following steps

1. First the data is converted into a lowercase by the means of a lower() which takes uncleaned data as input
2. The lowercased data is converted into tokens by the means of tokenize() which takes lowercased data as input
3. The stopwords are then removed from the tokenized data by the means of a for loop and not in keyword which takes lowercased tokenized data as input
4. Now, the data is stemmed by the means of stem() which takes lowercased tokenized stopwords-removed data as input
5. Finally the obtained clean data is joined with space by the means of “ “.join()
6. which takes the lowercased tokenized stopwords-removed stemmed data as input and the clean data obtained is returned as output by the clean\_text function
   * + - The training dataset is now cleaned by passing the sentences one by one as input to the clean\_text() by the means of a for loop
       - The CountVectorizer is imported from from sklearn.feature\_extraction.text and object for CountVectorizer cv is created by means of CountVectorizer() in which ngram range is initialized as 1 to 2
       - The training dataset is transformed into vectors by the means of fit\_transform() to which the cleaned training dataset is passed as input and after the vector transformation the training dataset is converted to an array by the means of a toarray()
       - The training dataset is then weighted by the means of a fit\_transform() to which the vector array of training dataset is passed as input and the vector array is weighted by the means of term frequencies and inverse document frequencies which alters the weight by the increasing or decreasing the weight of the terms.
       - The Multinomial Naive Bayes model is imported with from sklearn.naive\_bayes for which the object is also created and fit() is used to train the model with the vectorized transformed training dataset and the labels
       - Now the model is trained and the Video ID of the YouTube Video for which the comments are to be extracted for prediction is entered by the user.
       - The resource to give request to the YouTube API is built with the help of API key from the Google Developer Console.
       - The request to the YouTube API is created with the help of commentThreads() to which the part, videoID, maxResults, pageToken, order are given as input
       - The request is executed by the means of execute() and from the response obtained by the YouTube API, the comments are extracted and at a time only 100 comments can be given as response by the API and hence 100 comments are extracted 5 times by the means of for loop accounting for 500 comments extracted from the YouTube video and are appended in a list by the means of append()
       - The testing data and extracted YouTube Comments data are now cleaned by the means of clean\_text() which was used to clean the training dataset and transform() and toarray() methods are used to transform the data into vectors for the trained Multinomial Naive Bayes Model to predict the sentiments.
       - The model predicts the sentiment of the testing dataset and YouTube Comments extracted by the means of predict() to which the repective data are passed as inputs. From the predicted sentiments of 500 YouTube comments, count of different categories of sentiments are calculated by the means of a for loop
       - Finally accuracy\_score, classification\_report and confusion\_matrix are imported from sklearn.metrics library and the respective values are printed by comparing the predicted sentiments of testing data and correct sentiments of testing data.

# CHAPTER 5

**SYSTEM IMPLEMENTATION**

## 5.1. MODULE CODING

**IMPORTING PANDAS LIBRARY AND TRAINING DATASET**

import pandas as pd

df=pd.read\_csv("/content/mod\_train.txt", header=None, sep=" ", names=["Com ment","Emotion"], encoding="utf-8",nrows=10000) tdf=pd.read\_csv("/content/test.txt", header=None, sep=";", names=["Comment", "Emotion"], encoding="utf-8",nrows=1000)

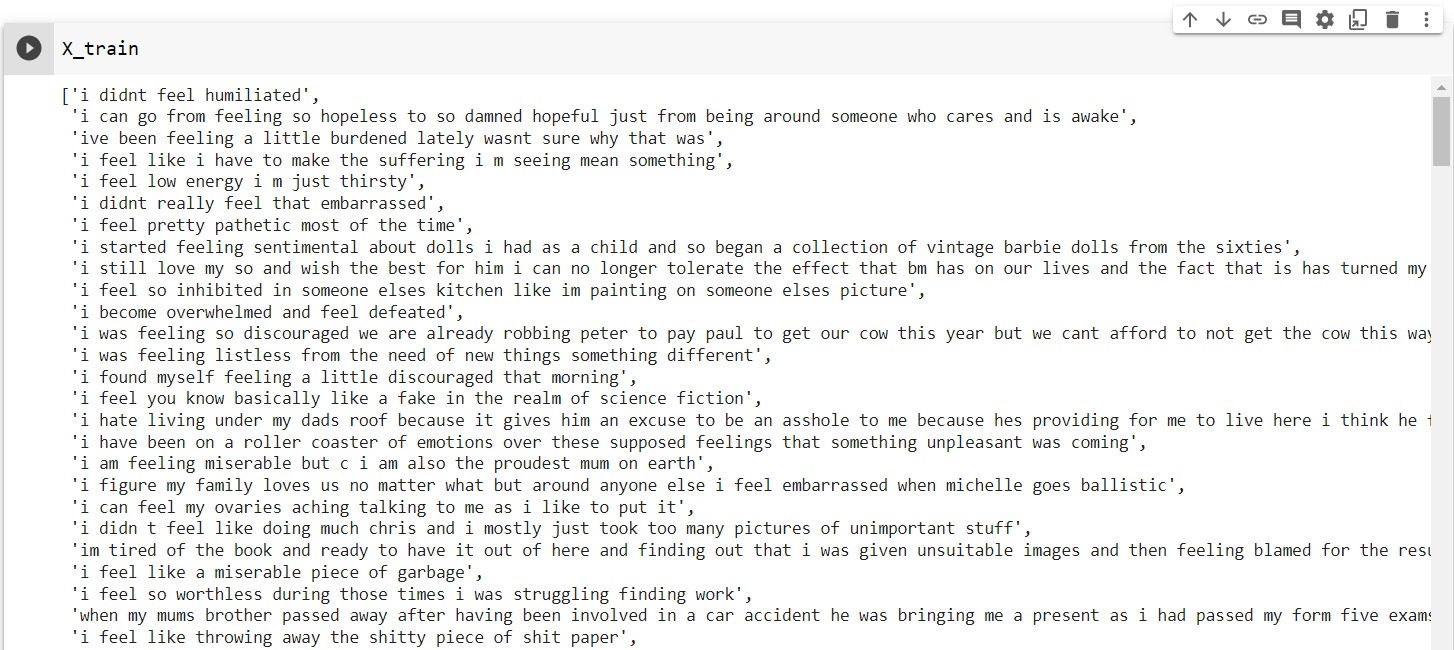
X\_train=df['Comment'].tolist() print(len(X\_train)) X\_accu=tdf['Comment'].tolist() y\_train=df['Emotion'].tolist() y\_result=tdf['Emotion'].tolist()

Figure 5.1.1. Training Dataset

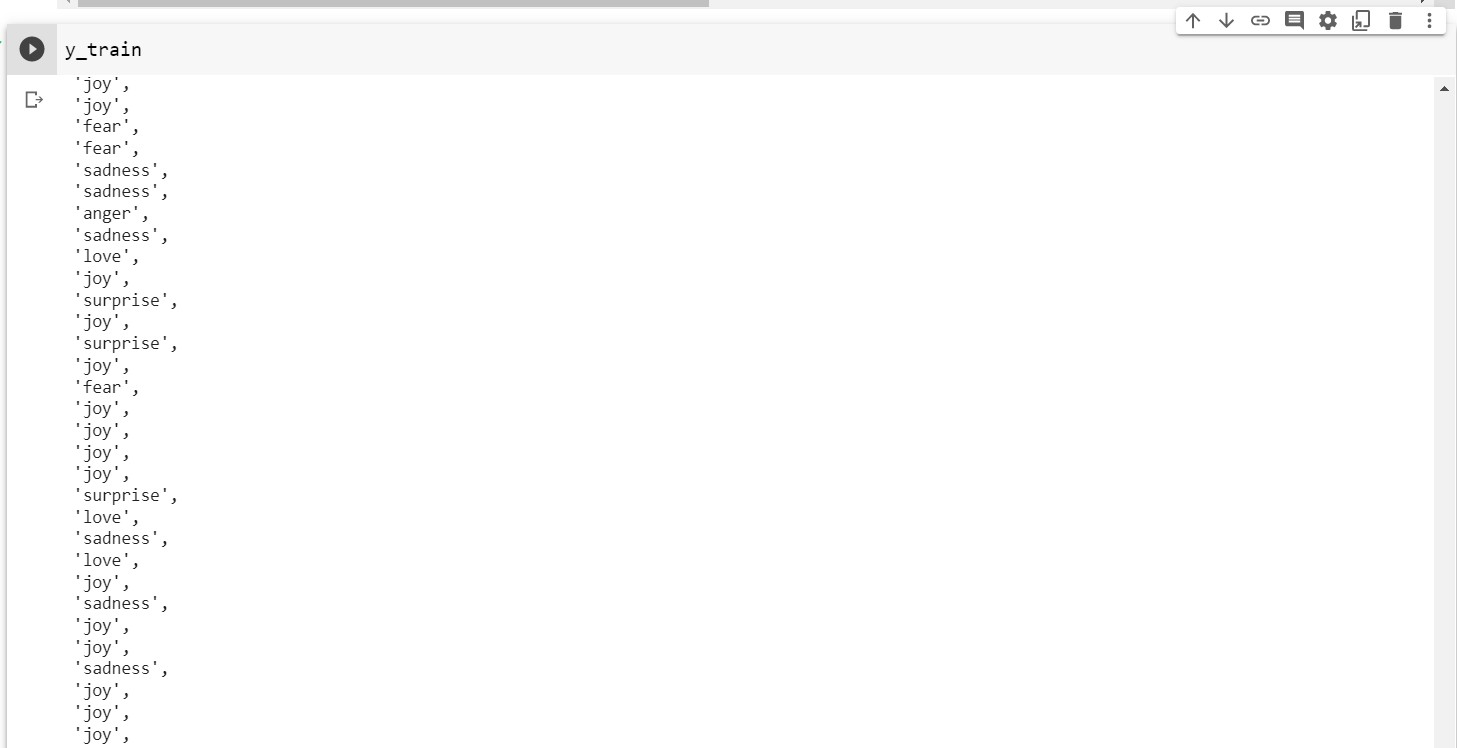


Figure 5.1.2. Training Labels

## IMPORTING NLTK TOKENIZER, STEMMER AND STOPWORDS REMOVER

from nltk.tokenize import RegexpTokenizer from nltk.stem.porter import PorterStemmer from nltk.corpus import stopwords

import nltk nltk.download('stopwords')

## CREATING OBJECTS FOR NLTK TOKENIZER, PORTERSTEMMER AND STOPWORDS REMOVER

tokenizer=RegexpTokenizer(r'\w+') en\_stopwords=set(stopwords.words('english')) ps=PorterStemmer()

print(en\_stopwords)

## FUNCTION FOR DATA CLEANING (TOKENIZING, STOPWORDS REMOVAL, STEMMING)

def clean\_text(text):

text=text.lower() tokens=tokenizer.tokenize(text) #Tokenizing

new\_token=[token for token in tokens if token not in en\_stopwords] #Stopwor ds Removal

stemmed\_tokens=[ps.stem(tokens) for tokens in new\_token] #Stemming clean\_texts=" ".join(stemmed\_tokens)

return clean\_texts

## DATA CLEANING (TOKENIZING, STOPWORDS REMOVAL, STEMMING) FOR TRAINING DATASET

X\_clean=[clean\_text(ct) for ct in X\_train]

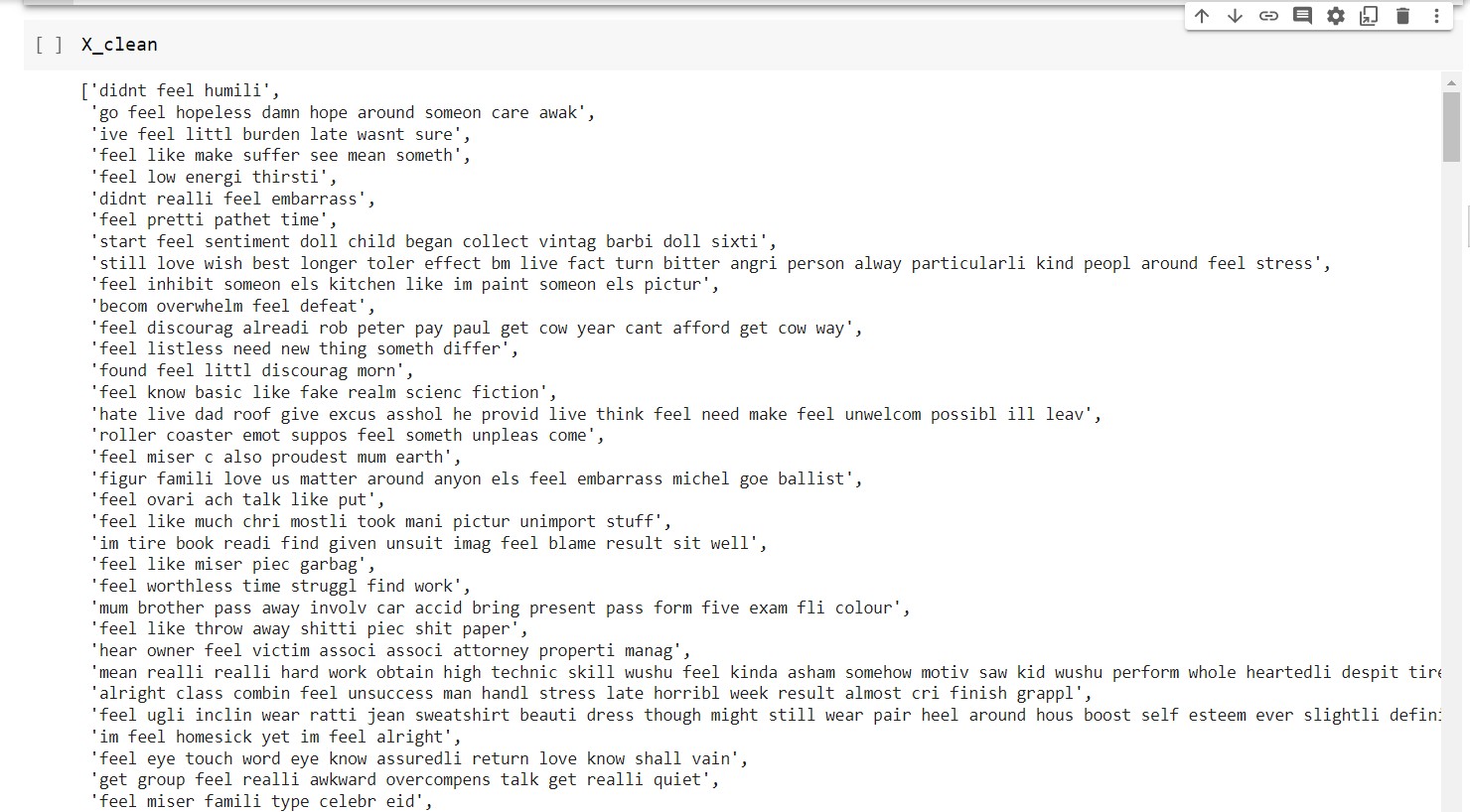


Figure 5.1.3. Cleaned Training Dataset

## IMPORTING COUNTVECTORIZER AND CREATING OBJECT

from sklearn.feature\_extraction.text import CountVectorizer cv=CountVectorizer(ngram\_range=(1,2))

## USING COUNTVECTORIZER TO TRANSFORM TRAINING DATA TO VECTORS

X\_vec = cv.fit\_transform(X\_clean).toarray()

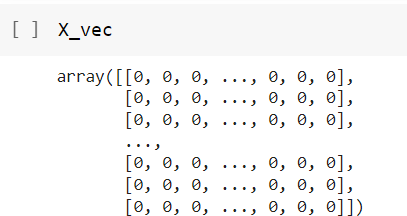


Figure 5.1.4. Vectorized Training Dataset

## USING TFIDFTRANSFORMER TO TRANSFORM TRAINING DATA BASED ON TERM FREQUENCY AND INVERSE DOCUMENT FREQUENCY

from sklearn.feature\_extraction.text import TfidfTransformer tfidf=TfidfTransformer() X\_vec\_tfidf=tfidf.fit\_transform(X\_vec).toarray()

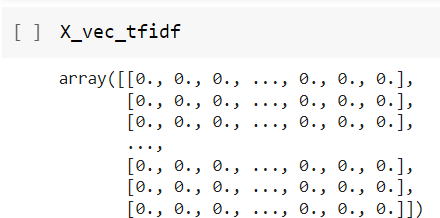


Figure 5.1.5. Vectorized and Transformed Training Dataset

## BUILDING MULTINOMIAL NAIVE BAYES MODEL

from sklearn.naive\_bayes import MultinomialNB mn=MultinomialNB()

## TRAINING MULTINOMIAL NAIVE BAYES MODEL USING TRAINING DATASET

mn.fit(X\_vec\_tfidf,y\_train)

## YOUTUBE COMMENT EXTRACTION FOR TESTING MULTINOMIAL NAIVE BAYES MODEL

def youtube\_request(video\_id,pgtoken):

#credentials

api\_key ="AIzaSyByzbpRtN9DSkL8RvcJBL1KtOwCdlzItMk"

#build a resource for youtube

resource = build('youtube', 'v3', developerKey=api\_key)

if pgtoken == 0:

#create a request to get 100 comments on the video request = resource.commentThreads().list(

part="snippet", videoId=video\_id,

maxResults= 100,#get 100 comments order="relevance")

else:

request = resource.commentThreads().list(

part="snippet", videoId=video\_id,

maxResults= 100,#get 100 comments pageToken=pgtoken, order="relevance",)

#execute the request response =request.execute() return response

from googleapiclient.discovery import build video\_id= str(input("Enter Video ID:")) X\_test=[]

for counter in range(5):

if counter == 0: res=youtube\_request(video\_id,0) nxtpgtoken=res["nextPageToken"] else:

res=youtube\_request(video\_id,nxtpgtoken) nxtpgtoken=res["nextPageToken"]

#get first 100 items to extract 100 comments items = res["items"][:100]

for item in items:

item\_info = item["snippet"]

#the top level comment can have sub reply comments topLevelComment = item\_info["topLevelComment"] comment\_info = topLevelComment["snippet"] X\_test.append(comment\_info["textDisplay"])

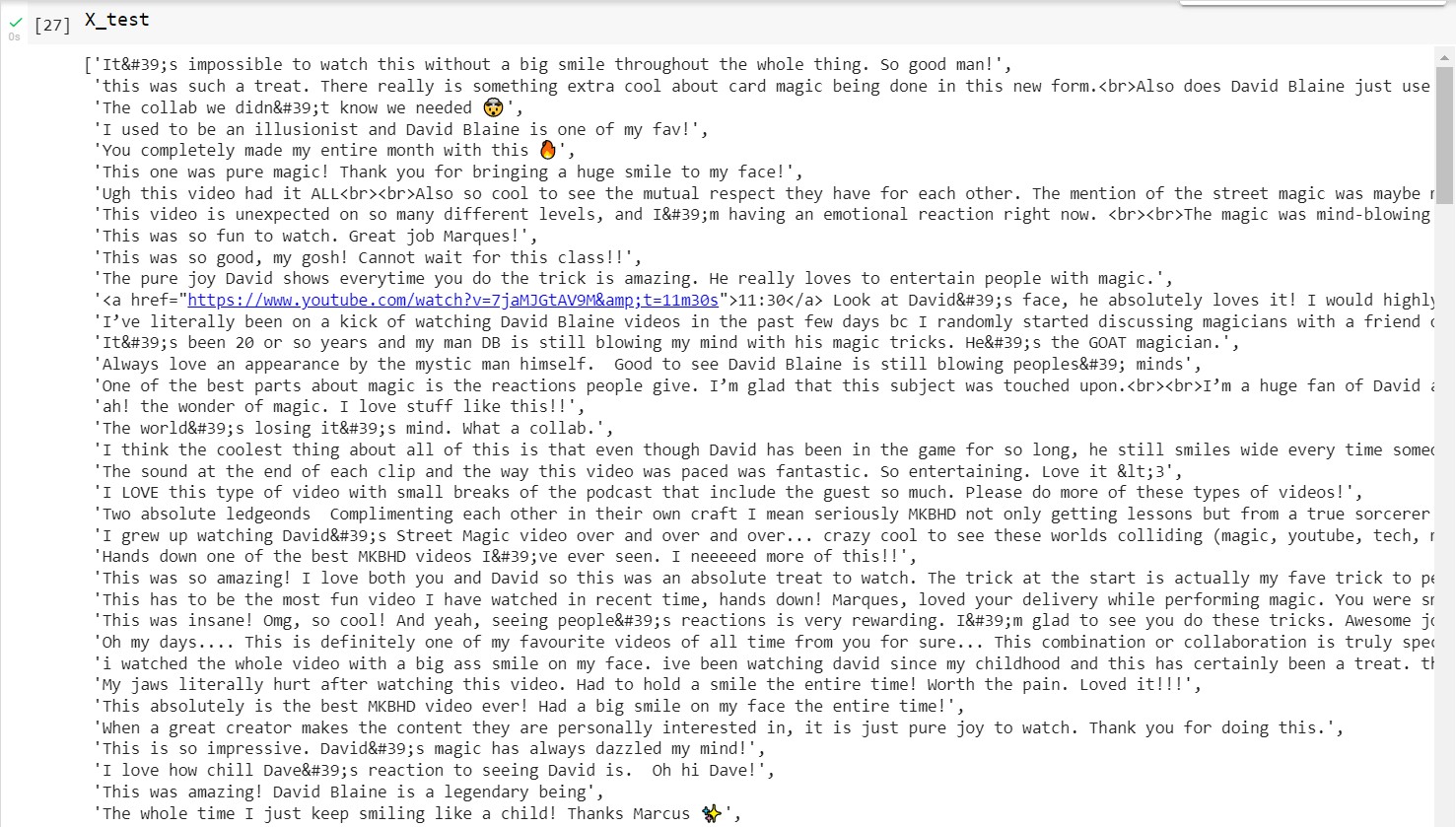


Figure 5.1.6. Extracted YouTube Video Comments

## DATA CLEANING (TOKENIZING, STOPWORDS REMOVAL, STEMMING) AND VECTOR TRANSFORMATION OF YOUTUBE COMMENTS

Xt\_clean=[clean\_text(ct) for ct in X\_test] Xa\_clean=[clean\_text(ct) for ct in X\_accu]

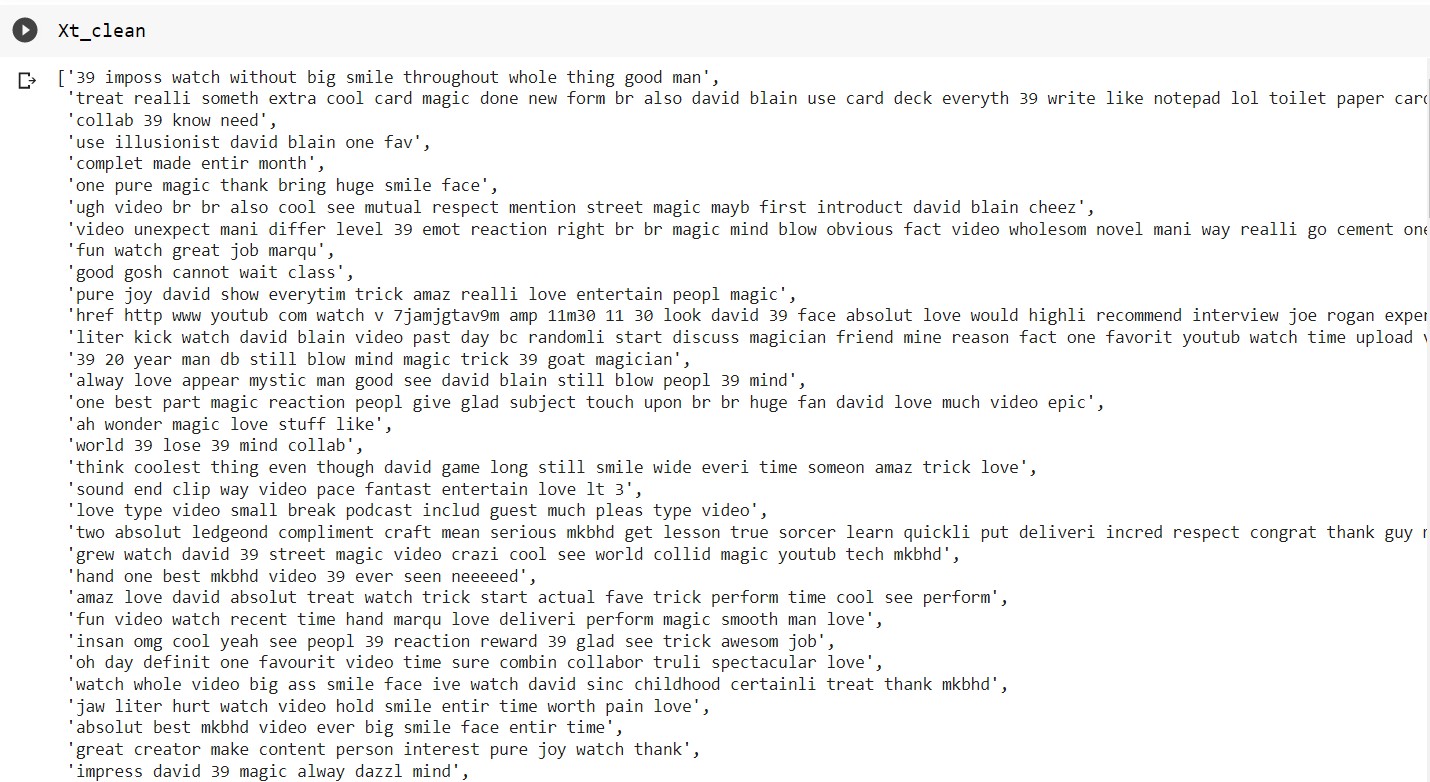


Figure 5.1.7. Cleaned YouTube Video Comments

Xt\_vec = cv.transform(Xt\_clean).toarray() Xa\_vec = cv.transform(Xa\_clean).toarray() Xt\_vec\_tfidf=tfidf.transform(Xt\_vec).toarray() Xa\_vec\_tfidf=tfidf.transform(Xa\_vec).toarray()

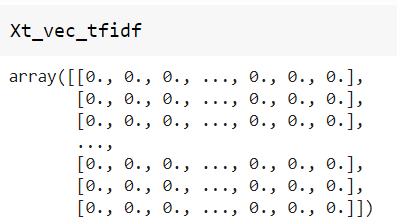


Figure 5.1.8. Vectorized and Transformed YouTube Video Comments

## PREDICTION OF SENTIMENT FROM YOUTUBE COMMENTS BY MULTINOMIAL NAIVE BAYES MODEL

ct\_prediction=mn.predict(Xt\_vec\_tfidf)

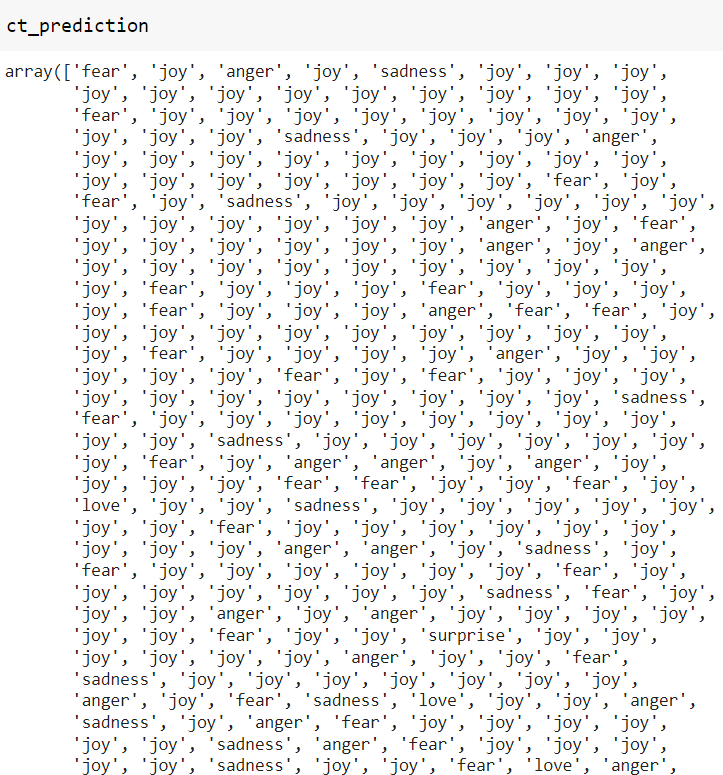


Figure 5.1.9. Predicted Sentiments of YouTube Video Comments

## PREDICTION OF SENTIMENT OF TESTING DATASET BY MULTINOMIAL NAIVE BAYES MODEL

test\_prediction=mn.predict(Xa\_vec\_tfidf)

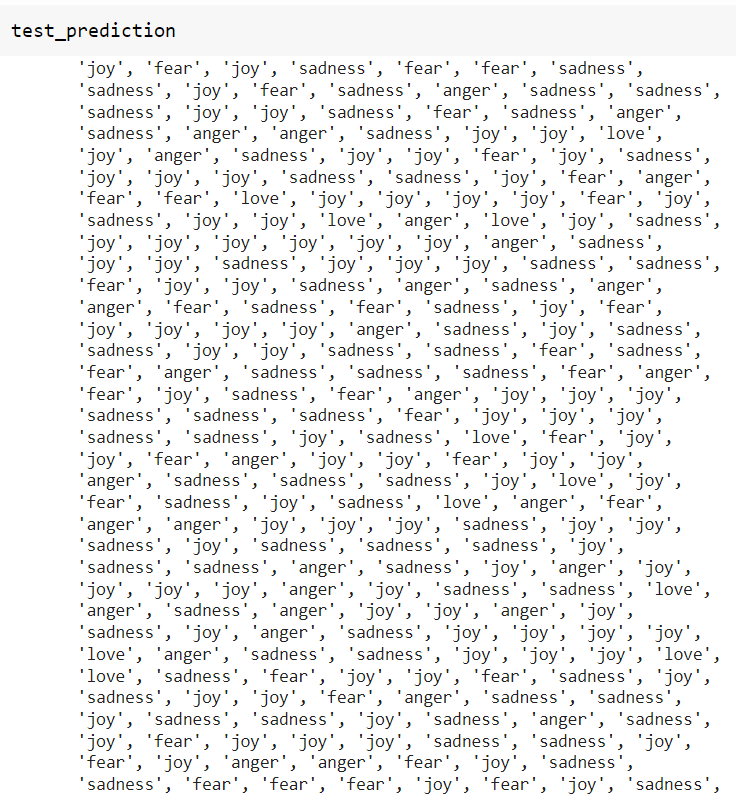


Figure 5.1.10. Predicted Sentiments of Testing Dataset

## EXTRACTION OF RESULTS PREDICTED BY MULTINOMIAL NAIVE BAYES MODEL

sad,angr,lv,sur,fr,joy=0,0,0,0,0,0 for result in ct\_prediction:

if result == "sadness":

sad+=1

elif result == "joy":

joy+=1

elif result == "anger":

angr+=1

elif result == "fear":

fr+=1

elif result == "surprise":

sur+=1 else:

lv+=1

print("Sad Comments:",sad) print("Happy Comments:",joy) print("Angry Comments:",angr) print("Fearful Comments:",fr) print("Love Comments:",lv) print("Surprise Comments:",sur)

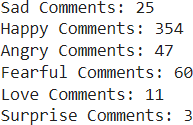


Figure 5.1.11. Categorization of Predicted Sentiments

## ACCURACY OF RESULTS PREDICTED BY MULTINOMIAL NAIVE BAYES MODEL

from sklearn.metrics import accuracy\_score from sklearn.metrics import classification\_report from sklearn.metrics import confusion\_matrix

print("Accuracy:",accuracy\_score(test\_prediction,y\_result),end="\n\n\n") possible\_values=["sadness","joy","anger","fear","surprise","love"] print("Classification Report:",end="\n\n") print(classification\_report(test\_prediction,y\_result,target\_names=possible\_value s),end="\n\n\n")

print("Confusion Matrix:",end="\n\n") print(confusion\_matrix(test\_prediction,y\_result))

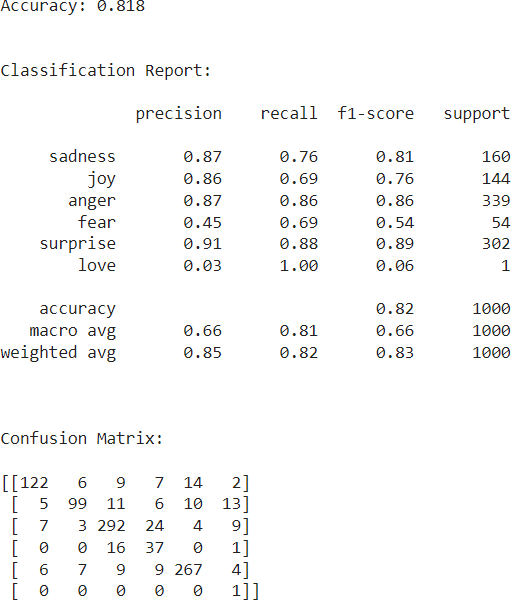


Figure 5.1.12. Accuracy of Predicted Sentiments

# CHAPTER 6 PERFORMANCE ANALYSIS

## SAMPLE TRAINING/TESTING DATASET

|  |  |
| --- | --- |
| **Comment** | **Emotion** |
| i didnt feel humiliated | sadness |
| i can go from feeling so hopeless to so damned hopeful just from  being around someone who cares and is awake | sadness |
| im grabbing a minute to post i feel greedy wrong | anger |
| i think it s the easiest time of year to feel dissatisfied | anger |
| i do not feel reassured anxiety is on each side | joy |
| i do feel that running is a divine experience and that i can expect to  have some type of spiritual encounter | joy |
| i now feel compromised and skeptical of the value of every unit of  work i put in | fear |
| i remember feeling acutely distressed for a few days | fear |
| i am now nearly finished the week detox and i feel amazing | surprise |
| i too feel as if i am a stranger in a strange land and i am raising my  son in a place that is not his father s ancestral home | surprise |
| i called it god because i d seen god in a book and figured god was  the right name for feeling so utterly affirmed and accepted without question | love |
| i seriously feel so blessed for the support that i have at home it s  amazing | love |

Table 6.1.1. Sample Training/Testing Dataset

## PERFORMANCE PARAMETERS TRUE/FALSE POSITIVES AND NEGATIVES

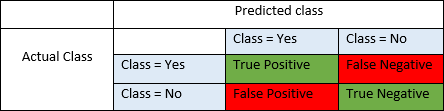


Table 6.2.1. True/False Positives and Negatives

True positive and True Negatives are the observations that are correctly predicted and therefore shown in green. We want to minimize false positives and false negatives so they are shown in red color. These terms are a bit confusing. So let’s take each term one by one and understand it fully.

True Positves (TP) are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN) are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g. if actual class says this passenger did not survive and predicted class tells you the same thing.

False Positives and False negatives, these values occur when your actual class contradicts with the predicted class.

False Positives (FP) are when actual class is no and predicted class is yes. E.g. if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

False Negatives (FN) are when actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die. Once you understand these four parameters then we can calculate Accuracy, Precision, Recall and F1 score.

## ACCURACY

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same. Therefore, you have to look at other parameters to evaluate the performance of your model.

Accuracy = TP+TN/TP+FP+FN+TN

## PRECISION

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate.

Precision = TP/TP+FP

## RECALL(SENSITIVITY)

Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the passengers that truly survived, how many did we label?

Recall = TP/TP+FN

## F1 SCORE

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it’s better to look at both Precision and Recall.

F1 Score = 2\*(Recall \* Precision) / (Recall + Precision)

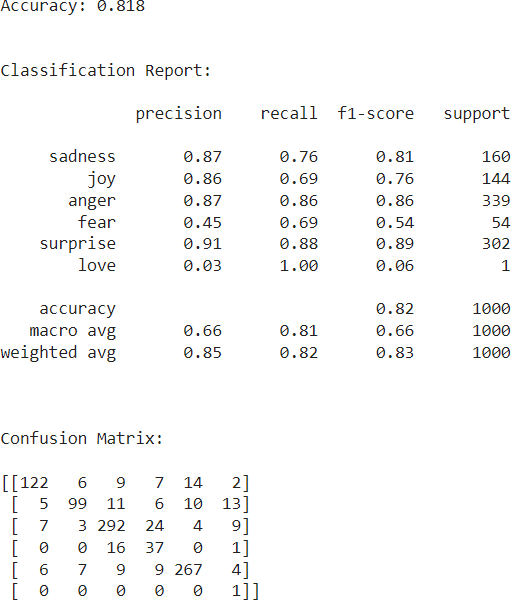


Figure 6.2.1. Performance Parameters

# CHAPTER 7 CONCLUSION

## CONCLUSION

In this project, the aim was to construct a YouTube comment dataset and build a Multinomial Naive Bayes Model to classify YouTube comments as one of the following categories: Sad, Happy, Angry, Fear, Love, Surprise. A total of 11,000 comments were manually labelled based on their sentiment which are equally divided based on one of the above categories. Following that, the performance of Multinomial Naive Bayes Model on the YouTube comment dataset was evaluated and the accuracy was found to be 81.8%. It was also observed that the sentiments of comments expressed by the people in the YouTube Comments section is more expressive when categorized into more categories rather than positive and negative. With the help of the analysis of Comments of YouTube video from Commentator, viewers can clearly understand the true nature of the opinions expressed by the people who viewed the video.

YouTube content creators and other media agencies can also use this system to analyse people’s opinions expressed for a video. This system also serves as an alternative solution to the removal of dislike button as of 2022. Thus, different categories of videos are tested and the sentiments of the videos and result was that positive videos are found to be dominant in emotions Happy, Surprise, Love and negative videos are found to be dominant in emotions Anger, Fear and Sad.

## FUTURE ENHANCEMENTS

The future enhancements that can be done to the Multinomial Naive Bayes Model to furthermore analyse and improve the prediction of sentiments expressed by people in Comment Section are:

* + - Increasing the number of training and testing datasets and implementing more classification labels in training and testing datasets to increase the overall accuracy of the sentiment predicted by the model and to to give a much more brief view of the emotions expressed by the viewers respectively.
    - Creating a more complex dataset rich in vocabulary and long sentences to further increase the accuracy of the sentiment predicted by the model
    - Implementing the model in a user friendly GUI interface though an Android App or Website for better user experience and for non-power users to be able to use the model effectively.

# REFERENCES

## REFERENCES

1. C. Day, “The importance of sentiment analysis in social media analysis.” [Online]. Available: https://[www.linkedin.com/pulse/](http://www.linkedin.com/pulse/) importance-sentiment- analysis-social-media-christine-day
2. B. Liu, Sentiment Analysis and Opinion Mining. Morgan & Claypool Publishers, 2012.
3. C. C. Aggarwal and C. Zhai, A Survey of Text Classification Algorithms. Boston, MA: Springer US, 2012, pp. 163–222.
4. B. Liu, Sentiment Analysis: Mining Opinions, Sentiments, and Emotions, 2nd ed. Cambridge University Press, 2020.
5. K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, “Text classification algorithms: A survey,” Information, vol. 10, no. 4, 2019.
6. “Youtube partner program overview & eligibility,” https://support.google. com/youtube/answer/72851?hl=en, accessed: 2021-02-09.
7. “How to earn money on youtube,” https://support.google.com/adsense/ answer/72857?hl=en, accessed: 2021-01-05.
8. A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. [Online]. Available: <http://www.aclweb.org/anthology/P11-1015>
9. B. Pang, L. Lee, and S. Vaithyanathan, “Thumbs up? sentiment classification using machine learning techniques,” in Proceedings of the 2002

Conference on Empirical Methods in Natural Language Processing (EMNLP 2002). Association for Computational Linguistics, Jul. 2002, pp. 79–86.

1. A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau, “Sentiment analysis of twitter data,” Proceedings of the Workshop on Languages in Social Media, 01 2011.
2. M. S. Neethu and R. Rajasree, “Sentiment analysis in twitter using machine learning techniques,” in 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), 2013, pp. 1–5.
3. R. Benkhelifa and F. Z. Laallam, “Opinion extraction and classification of realtime youtube cooking recipes comments,” in The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018). Springer International Publishing, 2018, pp. 395–404. [13] A.-K. Al-Tamimi, A. Shatnawi, and E. Bani-Issa, “Arabic sentiment analysis of youtube comments,” in 2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT), 2017, pp. 1– 6. 48
4. S. Sun, C. Luo, and J. Chen, “A review of natural language processing techniques for opinion mining systems,” Information Fusion, vol. 36, pp. 10– 25, 2017.
5. Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, pp. 436–44, 05 2015.
6. L. Deng and D. Yu, “Deep learning: Methods and applications,” Microsoft, Tech. Rep., May 2014.
7. Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 157–166, 1994.
8. S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural computation, vol. 9, pp. 1735–80, 12 1997.
9. “Understanding lstm networks,” https://colah.github.io/posts/ 2015-08- Understanding-LSTMs/, accessed: 2021-03-19.
10. M. Schuster and K. Paliwal, “Bidirectional recurrent neural networks,” IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681, 1997.
11. L. Zhang, S. Wang, and B. Liu, “Deep learning for sentiment analysis : A survey,” 2018.
12. “Convolutional neural networks,” https://[www.ibm.com/cloud/learn/](http://www.ibm.com/cloud/learn/) convolutional-neural-networks, accessed: 2021-03-23.
13. J. Brownlee, “How do convolutional layers work in deep learning neural networks?” 2021. [Online]. Available: https://machinelearningmastery.com/ convolutional-layers-for-deep-learning-neural-networks/
14. D. Goularas and S. Kamis, “Evaluation of deep learning techniques in sentiment analysis from twitter data,” in 2019 International Conference on Deep Learning and Machine Learning in Emerging Applications (Deep-ML), 2019, pp. 12–17.
15. N. Kalchbrenner, E. Grefenstette, and P. Blunsom, “A convolutional neural network for modelling sentences,” 2014.