APPENDIX 1

*SENTIMENT ANALYSIS*

Project Based Learning (PBL) project report

by

**Mr. Dharm Raj**

NAME OF THE CANDIDATES

**Sheikh Parvez Ahmed**

System ID- 2018003801

Roll no- 180101294

**Siddharth Gautam**

System ID- 2018011877

Roll no- 180101319

**Muheed Muzamil**

System ID- 2018000355

Roll no- 180101196

Department of Computer Science – AI&ML

SHARDA UNIVERSITY

Greater Noida

November 2019

APPENDIX 2

BONAFIDE CERTIFICATE

This is to certify that this project report entitled “**SENTIMENT ANALYSIS**” submitted to Sharda University, Greater Noida, is a bonafide record of work done by “**Sheikh Parvez Ahmed, Siddharth Gautam, Muheed Muzamil**” under my supervision from “**23rd July 2019**” to “**15th November 2019**”

(Signature)

Mr. Dharm Raj

Place-

Date-

APPENDIX 3

Declaration by Authors

This is to declare that this report has been written by us. No part of the report is plagiarized from other sources. All information included from other sources have been duly acknowledged. We aver that if any part of the report is found to be plagiarized, we are shall take full responsibility for it.

(Signature)

**Sheikh Parvez Ahmed**

System ID- 2018003801

Roll no- 180101294

(Signature)

**Siddharth Gautam**

System ID- 2018011877

Roll no- 180101319

(Signature)

**Muheed Muzamil**

System ID- 2018000355

Roll no- 180101196

Place-

Date-

APPENDIX 4

# *Table of Contents*

# *Chapter No Title Page No*

# *Abstract I*

# *Abbreviations and Nomenclature I*

# *Contents*

# Introduction --------------------------------------------------------------- 1

# What is Sentiment Analysis -------------------------------------------- 2

# System Requirements -------------------------------------------------- 3

# Some Python IDE --------------------------------------------------------- 4

# Modules and Sources ---------------------------------------------------- 5

# Block Diagrams and Algorithms --------------------------------------- 6

# Code For Sentiment Analysis ------------------------------------------ 7-11

# Test Cases and their Outputs ----------------------------------------- 11-14

# Conclusion ----------------------------------------------------------------- 15

*Abstract*

Understanding the sentiment and emotions of other people is very important in today’s world. Humans beings are emotionally driven social animals, but due to the eruption of internet and social media, we need similarly technologically advanced methods to analyse the sentiments and emotions of a person. Using Natural Language Processing, we have created an interface where a user can analyse the emotions of a person using his twitter or Reddit username, or an audio clip of the person talking. User can also analyse the reaction of audience to a certain Reddit post.

*Abbreviations and Nomenclature*

IDE- Integrated Development Environment

RAM- Random Access Memory

GB- Giga Bytes

PIP- Preffered Installer Program

PyDAAL- Python Data Analytics Acceleration Library

API- Application Program Interface

PRAW- Python Reddit API Wrapper

GUI- Graphical User Interface

NLP- Natural Language Processing

I

*Introduction*

The best businesses understand the sentiment of their customers—what people are saying, how they’re saying it, and what they mean. Customer sentiment can be found in tweets, comments, reviews, or other places where people mention your brand. Sentiment Analysis is the domain of understanding these emotions with software, and it’s a must-understand for developers and business leaders in a modern workplace.

In the last decade, sentiment analysis (SA), also known as opinion mining, has attracted an increasing interest. It is a hard challenge for language technologies, and achieving good results is much more difficult than some people think. The task of automatically classifying a text written in a natural language into a positive or negative feeling, opinion or subjectivity (Pang and Lee, 2008), is sometimes so complicated that even different human annotators disagree on the classification to be assigned to a given text.

As with many other fields, advances in deep learning have brought sentiment analysis into the foreground of cutting-edge algorithms. Today we use natural language processing, statistics, and text analysis to extract, and identify the sentiment of text into positive, negative, or neutral categories; or emotions like happy, angry, sad, excited, fear, bored.

*What is Sentiment Analysis :*

* Sentiment Analysis is text classification tool for determining whether a piece of writing is positive, negative or neutral.
* A sentiment analysis system for text analysis combines natural language processing (NLP) and Machine Learning techniques to assign weighted sentiment scores to the entities, topics, themes and categories within a sentence or phrase.
* Sentiment analysis is extremely useful in social media monitoring as it allows us to gain an overview of the wider public opinion behind certain topics.
* The ability to extract insights from social data is a practice that is being widely adopted by organizations across the world.
* Most of the sentiment analysis is done by using the infamous K-Nearest-Neighbor algorithm of Machine Learning.
* Various branches of Sentiment Analysis are:
  + Emotional Analysis
  + Contextual Semantic Search
  + Named Entity Extractor

*System Requirements*

* **Recommended System Requirements**
* **Processors:**
  + Intel® Core™ i5 processor 4300M at 2.60 GHz or 2.59 GHz (1 socket, 2 cores, 2 threads per core), 8 GB of DRAM
  + Intel® Xeon® processor E5-2698 v3 at 2.30 GHz (2 sockets, 16 cores each, 1 thread per core), 64 GB of DRAM
  + Intel® Xeon Phi™ processor 7210 at 1.30 GHz (1 socket, 64 cores, 4 threads per core), 32 GB of DRAM, 16 GB of MCDRAM (flat mode enabled)
* **Disk space:** 2 to 3 GB
* **Operating systems:** Windows® 10, macOS®, and Linux®

**Minimum System Requirements**

* **Processors:** Intel Atom® processor or Intel® Core™ i3 processor
* **Disk space:** 1.5 GB
* **Operating systems:** Windows® 7 or later, macOS®, and Linux®
* **Python versions:** 3.X
* **Included development tools:** Conda, Conda-Env, Jupyter Notebook (IPython)
* **Compatible tools:** Microsoft Visual Studio®, PyCharm
* **Included Python packages:** NumPy, SciPy, scikit-learn, pandas, Matplotlib, Numba, Intel® Threading Building Blocks, pyDAAL, Jupyter, mpi4py, PIP, and others.

*Some Python IDE :*

**PyCharm:** PyCharm is one of the widely used Python IDE which was created by Jet Brains. It is one of the best IDE for Python. PyCharm is all a developer’s need for productive Python development.

With PyCharm, the developers can write a neat and maintainable code. It helps to be more productive and gives smart assistance to the developers. It takes care of the routine tasks by saving time and thereby increasing profit accordingly.

**Platform Support:** WINDOWS, LINUX, MAC etc.

-----------------------------------------------------------------------------------------------------------------

**SPYDER:** SPYDER is another big name in the IDE market. It is a good python compiler.

It is famous for python development. It was mainly developed for scientists and engineers to provide a powerful scientific environment for Python. It offers an advanced level of edit, debug, and data exploration feature. It is very extensible and has a good plugin system and API. As SPYDER uses PYQT, a developer can also use it as an extension. It is a powerful IDE.

**Platform Support:** QT, WINDOWS, LINUX, MAC OS etc.

-----------------------------------------------------------------------------------------------------------------

**IDLE:** IDLE is a popular Integrated Development Environment written in Python and it has been integrated with the default language. It is one of the best IDE for python.

IDLE is a very simple and basic IDE which is mainly used by the beginner level developers who want to practice on python development. It is also a cross-platform thus helping the trainee developers a lot but it also called as a disposable IDE as a developer moves to more advance IDE after learning the basics.

**Platform Support:** WINDOWS, LINUX, MAC OS etc.

*Modules*

***Tweepy***- Tweepy supports accessing Twitter via Basic Authentication and the newer method, OAuth. Twitter has stopped accepting Basic Authentication so OAuth is now the only way to use the Twitter API.

Tweepy provides access to the well documented Twitter API. With tweepy, it's possible to get any object and use any method that the official Twitter API offers.

Main Model classes in the Twitter API are Tweets,Users, Entities and Places. Access to each returns a JSON-formatted response and traversing through information is very easy in Python.

Source:- <https://tweepy.readthedocs.io/en/latest/>

***PRAW***- PRAW, an acronym for “Python Reddit API Wrapper”, is a python package that allows for simple access to Reddit's API. PRAW aims to be easy to use and internally follows all of Reddit's API rules. With PRAW there's no need to introduce sleep calls in your code. Give your client an appropriate user agent and you're set.

Source:- <https://praw.readthedocs.io/en/latest/>

***SpeechToText***- Speech to text conversion means that to recognize words spoken by a person, and convert the voice to written text.

To convert text to speech, you need to install the win32com library and import it.

Source:- <https://www.ibm.com/cloud/watson-speech-to-text>

***Paralleldots API***- ParallelDots API, is a webservice that can comprehend a huge amount of unstructured textual content to enhance your textual cognition. We have state-of-the-art qualitative content analysis tools vis-à-vis Named Entity Recognition, Text Classification, Sentiment Analysis, Semantic Analysis & Keyword extraction that will help you parse the electronic texts(new articles, blogs, tweets, Facebook comments etc). For more information on ParallelDots API and it’s capabilities, check our interactive Demo. We are coming up with Vision APIs soon.

Source:- <https://www.paralleldots.com/sentiment-analysis>

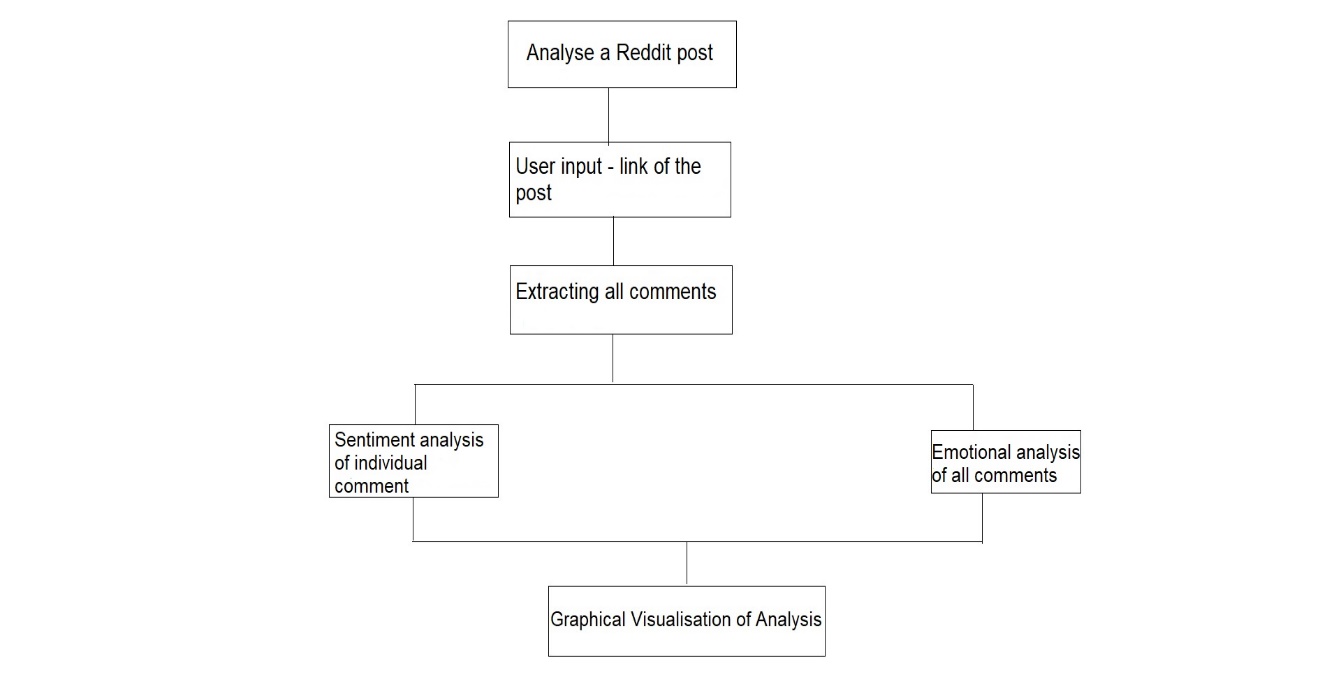
***Tkinter***- Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. The name Tkinter comes from Tk interface. Tkinter was written by Fredrik Lundh. Tkinter is free software released under a Python license.

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets. The Button widget is used to

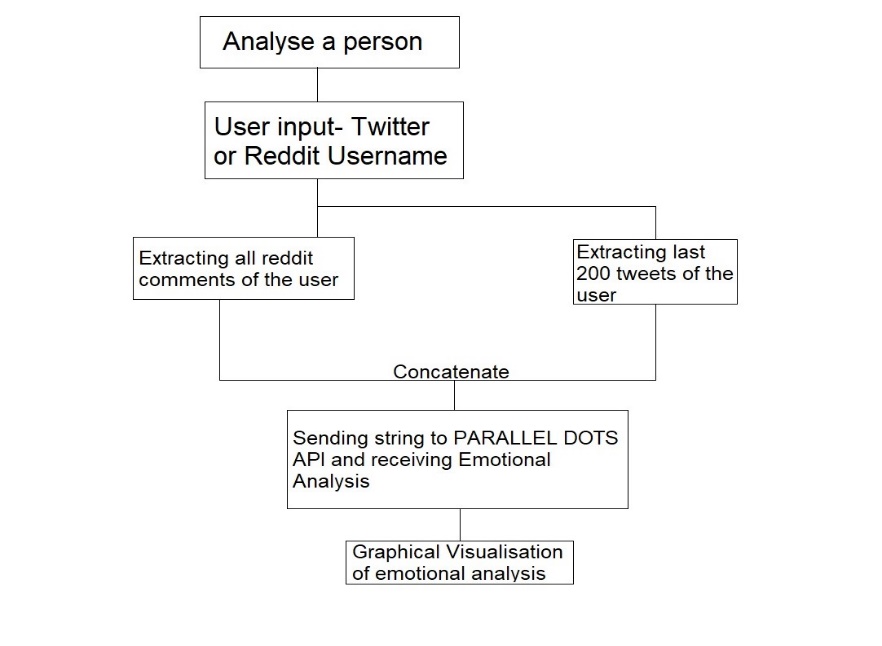
display buttons in your application. The Canvas widget is used to draw shapes, such as lines, ovals, polygons and rectangles, in your application.

Source:- <https://wiki.python.org/moin/TkInter>

*Block Diagrams*

**Block Diagram:** For Analysis of Reddit Data

**Block Diagram:** For Analysis of a particular user



**Block Diagram:** For Analysis of an Audio file (mp3)

Graphical representation of Emotional Analysis

Emotional analysis on the text of speech

Transcribing speech to text

User Input-Path of Audio File(mp3)

Analyse an audio file

*Code for Sentiment Analysis*

***#Importing all libraries***

from tkinter import \*

import praw

import tweepy

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sns

import json

from os.path import join, dirname

import paralleldots

from ibm\_watson import SpeechToTextV1

from praw.models import MoreComments

from ibm\_watson import SpeechToTextV1

from ibm\_watson.websocket import RecognizeCallback, AudioSource

from ibm\_cloud\_sdk\_core.authenticators import IAMAuthenticator

authenticator = IAMAuthenticator('2GLz9FSNQKpCutmiRtpVMG3AJQPh\_DItUe89rTwy4QJC')

speech\_to\_text = SpeechToTextV1(

authenticator=authenticator

)

speech\_to\_text.set\_service\_url('https://gateway-lon.watsonplatform.net/speech-to-text/api')

***#Client details for Twitter API***

consumer\_key = "pZb2PizxEsB1eyMcMtPKt0"

consumer\_secret = "MLAIqKyYVkdykZ6eibeoWcPOX56gOgfV0vkWIVzmKEVxGQ"

access\_key = "3046753945-qtQ26I5KlW9Ng1LmX44OxdbJTQ8vpcDBPBE"

access\_secret = "yxNsmJMzxB7WumR5u2HB9NYM5Cdftj3pRSwwaxQCTw"

auth = tweepy.OAuthHandler(consumer\_key, consumer\_secret)

auth.set\_access\_token(access\_key, access\_secret)

api = tweepy.API(auth)

***#Client details Sentiment Analysis API service***

paralleldots.set\_api\_key("SLlVzk4on9gsErs1SUKzIAgotNmB33j7HcZT77Lo")

***#Client details for Reddit API***

reddit = praw.Reddit(client\_id = "bqa8w7IjEhQw",

client\_secret = "e-nSLf7A6gP1U4U4K15BoArM8",

password= "thisisword",

username="dotthrowaway123",

user\_agent="taining",)

***#Code for the main Window***

root=Tk()

frame=Frame(root)

#Code for functioning of "Checking Personality"

def window1():

first\_window=Toplevel(frame, width=400, height=300)

def UserSentiment():

#taking input from text field

twitteruser = e1.get()

reddituser = e2.get()

***#Code for extracting User's Input Twitter data***

tmp=[]

b = ""

thetweet = []

***#in case there's no input***

if len(twitteruser)==0 and len(reddituser)==0:

c=Label(first\_window).config(text="")

c=Label(first\_window, text = "Enter at least one ID")

c.grid(row=2, columnspan=3)

else:

if len(twitteruser)>0:

number\_of\_tweets=200

try:

***#calling API***

tweets = api.user\_timeline(screen\_name=twitteruser, count=number\_of\_tweets, tweet\_mode="extended")

tweets\_for\_csv = [tweet.full\_text for tweet in tweets]

#making the tweets into coherent sentences

for j in tweets\_for\_csv:

thetweet = j.split()

thetweet.pop(-1)

for word in thetweet:

b = b + " " + str(word)

tmp.append(b)

twerror=False

except Exception:

c=Label(first\_window).config(text="")

c=Label(first\_window, text = "Make sure that the ID you entered is correct")

c.grid(row=2, columnspan=3)

twerror=True

else:

twerror=False

***#Code for extracting User's Input data***

commentvar = ""

if len(reddituser)>0:

try:

#calling Reddit's API

for comment in reddit.redditor(reddituser).comments.new(limit=None):

commentvar = commentvar + " " + str(comment.body)

rederror=False

except Exception:

c=Label(first\_window).config(text="")

c=Label(first\_window, text = "Make sure that the ID you entered is correct")

c.grid(row=2, columnspan=3)

rederror=True

else:

rederror=False

#join words from both reddit and twitter

text = str(tmp) + " " + commentvar

***#Code for the functioning of Personality Insight using Extarcted Data for Checking Personality***

if twerror==False and rederror==False and len(text.split())>25:

try:

UserDict = paralleldots.emotion(text)['emotion']

df = pd.DataFrame.from\_dict(UserDict, orient = 'index')

df.reset\_index(inplace=True)

df.columns=['Sentiment', 'Percentile']

plt.figure(figsize=(15,5))

sns.barplot(x="Sentiment", y="Percentile", data = df)

plt.show()

except Exception:

c=Label(first\_window).config(text="")

c=Label(first\_window, text = "Not enough words to scan, please enter ID of a different

person")

c.grid(row=2, columnspan=3)

else:

c=Label(first\_window).config(text="")

c=Label(first\_window, text = "Not enough words to scan, please enter ID of a different person")

c.grid(row=2, columnspan=3)

***#Code for "Checking Personality" GUI Window***

a = Label(first\_window,text="Twitter Username")

a.grid(row=0, column=0)

b = Label(first\_window,text="Reddit Username")

b.grid(row=1, column=0)

c = Label(first\_window, text="")

c.grid(row=2, columnspan=3)

e1 = Entry(first\_window)

e2 = Entry(first\_window)

e1.grid(row=0, column=2, pady=10, padx=15)

e2.grid(row=1, column=2)

Button(first\_window,text='Begin analysis', command=UserSentiment).grid(row=3,sticky=W,pady=4,column=2,padx=20)

***#Code for "Checking your Interest" using Stack Overflow data***

def window2():

third\_window=Toplevel(frame)

***#Code for "Check your Interest" GUI Window excluding Button***

stackidlabel=Label(third\_window, text="Enter Reddit post url")

stackidlabel.grid(row=0, column=0, padx=10, pady=10)

stackidentry=Entry(third\_window)

stackidentry.grid(row=0, column=2, padx=10, pady=10)

exceptionlabel=Label(third\_window)

exceptionlabel.grid(row=2, column=1)

def getuserdatared():

***#Code for getting User Input's data and extracting data from Stack Overflow with Try Exception Handling***

try:

redurl=stackidentry.get()

submission = reddit.submission(url=redurl)

SenDict = {"negative":0, "neutral":0, "positive":0}

for top\_level\_comment in submission.comments:

if isinstance(top\_level\_comment, MoreComments):

continue

text = top\_level\_comment.body

#print(text)

RedditPostSentiment = paralleldots.sentiment(text)['sentiment']

LoopSen=max(RedditPostSentiment, key = lambda x: RedditPostSentiment.get(x))

if LoopSen == "negative":

SenDict["negative"]+=1

elif LoopSen == "positive":

SenDict["positive"]+=1

elif LoopSen == "neutral":

SenDict["neutral"]+=1

else:

pass

df = pd.DataFrame.from\_dict(SenDict, orient = 'index')

df.reset\_index(inplace=True)

df.columns=['Sentiment', 'Frequency']

plt.figure(figsize=(15,5))

sns.barplot(x="Sentiment", y="Frequency", data = df)

plt.show()

except Exception:

errortext=Label(fourth\_window).config(text="")

errortext=Label(fourth\_window, text="Please enter a correct url")

errortext.grid(row=2, column=1)

***#Code for Submit Button and it's command***

enterstackdata=Button(third\_window, text="Submit", command=getuserdatared)

enterstackdata.grid(row=1, column=1, padx=10, pady=10)

***#Code for the functioning of "Analyze via Audio"***

def window3():

fourth\_window=Toplevel(frame)

***#Code for "Analyze via Audio" GUI Window excluding Button***

audiolabel=Label(fourth\_window, text="Please enter the file path for your audio file")

audiolabel.grid(row=0, column=0, padx=10, pady=10)

errortext=Label(fourth\_window, text="")

errortext.grid(row=1)

audfilepath=Entry(fourth\_window)

audfilepath.grid(row=0, column=2, padx=10, pady=10)

wait=Label(fourth\_window, text="Please wait patiently after clicking Submit, transcribing audio will take as long as the audio is")

wait.grid(row=2, columnspan=4)

def transcribe():

***#Code for getting the Audio file and coverting the speech into text and then analyzing it using Personality Insight with Try Exception Handling***

script=audfilepath.get()

try:

with open(script,

'rb') as audio\_file:

prof = speech\_to\_text.recognize(audio\_file, content\_type="audio/mp3").result

a = json.dumps(prof, indent = 4)

z=0

y=0

for i in prof['results']:

z=z+1

trlist=[]

for i in range(z):

transcripts = prof['results'][i]['alternatives'][0]['transcript']

trlist.append(transcripts)

trstr=""

for j in trlist:

trstr = trstr + " " + j

UserDict = paralleldots.emotion(trstr)['emotion']

print(trstr)

df = pd.DataFrame.from\_dict(UserDict, orient = 'index')

df.reset\_index(inplace=True)

df.columns=['Sentiment', 'Percentile']

plt.figure(figsize=(15,5))

sns.barplot(x="Sentiment", y="Percentile", data = df)

plt.show()

except Exception:

errortext=Label(fourth\_window).config(text="")

errortext=Label(fourth\_window, text="Error in finding file, please make sure the path is correct and the format is mp3")

errortext.grid(row=2, column=1)

***#Code for Submit button and it's command***

speechtotext=Button(fourth\_window, text="Submit", command=transcribe)

speechtotext.grid(row=1, column=1, padx=10, pady=10)

***#Code for the main GUI Window***

checkpersonality=Button(frame, text="Check sentiments of a person", command=window1)

checkpersonality.grid(row=0, column=0, padx=10, pady=10)

comparetwopeople=Button(frame, text="Analyse a Reddit post", command=window2)

comparetwopeople.grid(row=0, column=1, padx=10, pady=10)

analyzeaudio=Button(frame, text="Analyze via audio", command=window3)

analyzeaudio.grid(row=0, column=3, padx=10, pady=10)

cleargraph=Label(frame, text="If there are any graph open, please close them before preeceding")

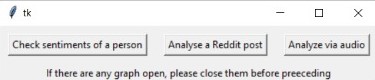
cleargraph.grid(row=1,columnspan=5)

frame.pack()

root.mainloop()

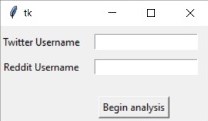
*Test Cases*

This is the main window of the Sentiment Analysis program.



1. Analysing a Twitter/Reddit user

This the window for Sentiment Analysis of a particular Twitter or Reddit user.

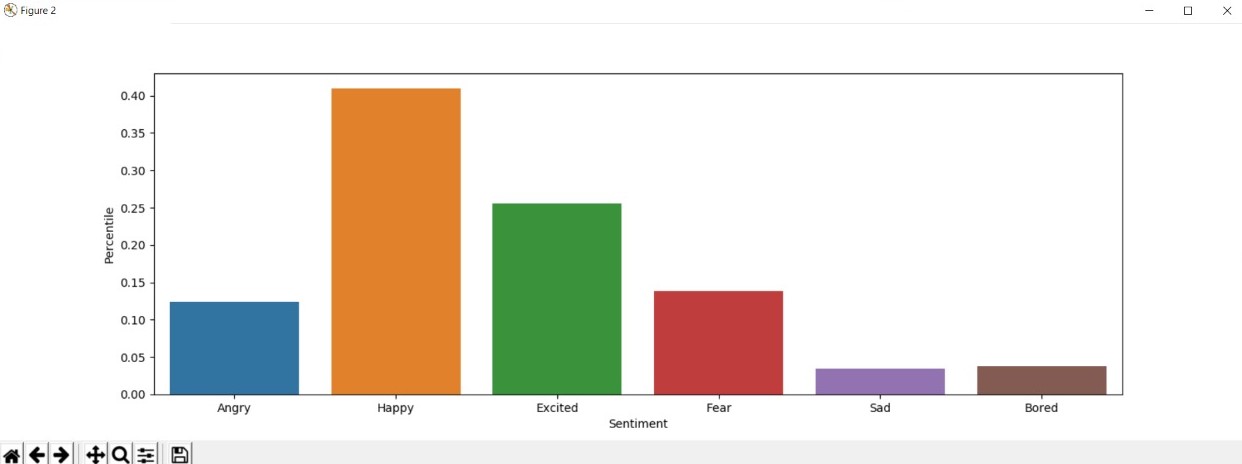


The output generated is based on Twitter data of Shahrukh Khan “iamsrk”.

It extracts only the last 200 tweets of “iamsrk”.

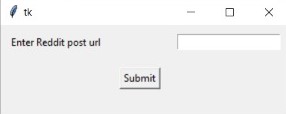
The output shows the sentiment of “iamsrk” as-

* Angry
* Happy
* Excited
* Fear
* Sad
* Bored



1. Analysing a Reddit post

This the window for Sentiment Analysis of a post of Reddit user.

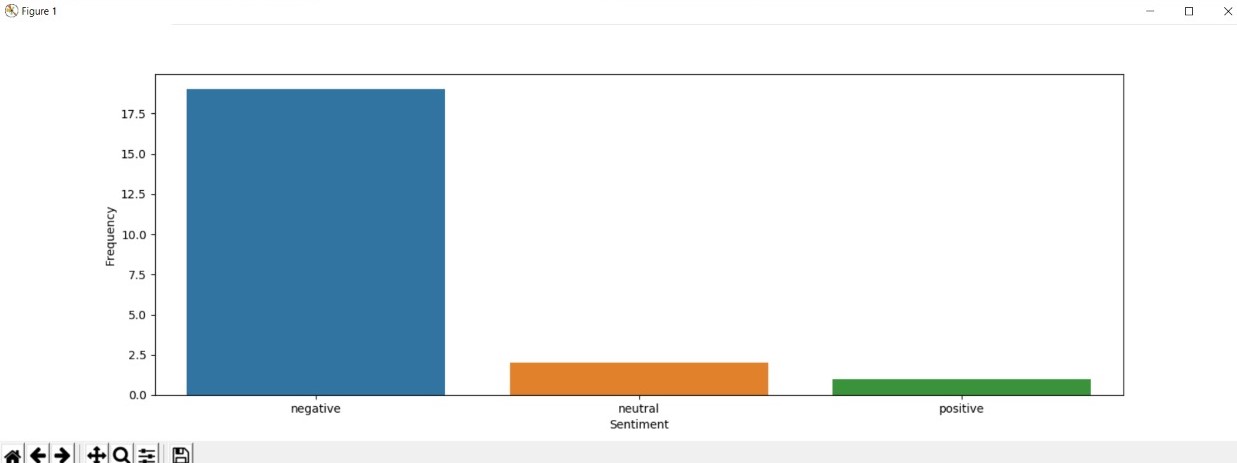


The output generated is based on Reddit post data of “<https://www.reddit.com/r/programming/comments/dve2xp/intels_cascade_lake_cpus_impacted_by_new/>”

It extracts all the comments of the post and does sentiment analysis upon them.

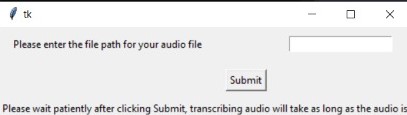
The output shows the sentiment of “<https://www.reddit.com/r/programming/comments/dve2xp/intels_cascade_lake_cpus_impacted_by_new/> “ as-

* Negative
* Neutral Sentiment
* Positive



1. Analysing an Audio file

This the window for Sentiment Analysis of an audio file.

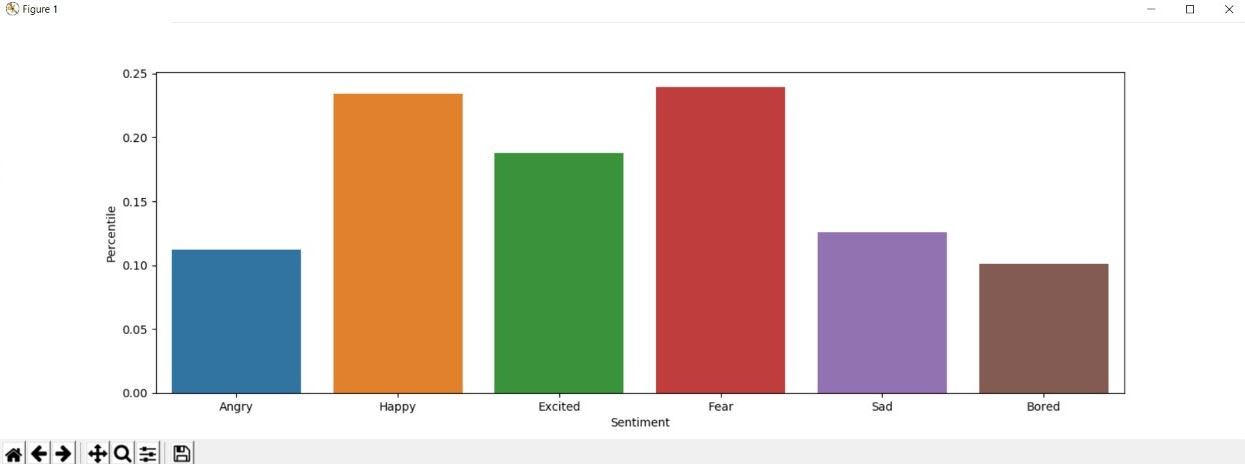


The output generated is based on the audio of “obama2011ladenSpeech.mp3 “

It transcribes the audio speech to text and does emotional analysis on the text of speech.

The output shows the sentiment of “obama2011ladenSpeech.mp3 “ as--

* Angry
* Happy
* Excited
* Fear
* Sad
* Bored



*Conclusion*

NLP, as a field in machine learning, has been unsurprisingly rising in both usability and popularity. The exponentially increasing amount of data available on the internet has a huge potential for insight to be derived from. To have a better understanding of the emotions and sentiments of an entity, ranging from a human to an entire community, we use machine learning with basis in psychology to analyse it. This, practically, is a step in the direction of large-scale mirror neuron system of our brains.

The immediate uses of this system is in recognising the reaction of a certain audience or a community to any given topic of discussion. This can help up automate the consensus if the masses and react accordingly.