****

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**A Skill Development Project Report**

on

**Natural Language Processing**

Submitted in fulfillment of the requirements for the award of the Degree of

Bachelor of Technology

Submitted by

Lohith K: R20EF239

Manoj R: R20EF241

Pranav R: R20EF248

S B Keerthan: R20EF254

2022-2023

Rukmini Knowledge Park, Kattigenahalli, Yelahanka, Bengaluru-560064

www.reva.edu.in

**DECLARATION**

We, Mr. S B Keerthan, Mr. Pranav R, Mr. Manoj R and Mr. Lohith K the students of Bachelor of Technology, belong in to School of Computer Science And Engineering, REVA University, declare that this Skill development Project Report / Dissertation entitled “Natural Language Processing” is the result the of Skill development program done at School of Computer Science And Engineering, REVA University.

We are submitting this Skill development Project Report / Dissertation in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering by the REVA University, Bangalore during the academic year 2022-2023.

*Signature of the candidates with dates*

*Certified that this project work submitted by S B Keerthan, Pranav R, Manoj R and Lohith K has been carried out and the declaration made by the candidate is true to the best of my knowledge.*

|  |  |  |
| --- | --- | --- |
|  |  | *Signature of Director of School* |
|  |  | *Date:…………….* |
|  |  | *Official Seal of the School* |

****

**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING.**

**CERTIFICATE**

Certified that the Skill Development project work entitled **Natural Language Processing** carried out under our guidance by<\_\_> are bonafide students of REVA University during the academic year 2022-2023, are submitting the Skill development project report in partial fulfillment for the award of **Bachelor of Technology** in Computer Science And Engineering during the academic year **2022-2023.**

|  |  |  |
| --- | --- | --- |
|  |  | **Signature with date** |
|  |  |  |
|  |  |  |
|  |  | **Dr. Ashwin Kumar U M** |
|  |  | **Director** |

**Index**

|  |  |  |
| --- | --- | --- |
| **SL.No** | **Contents** | **Page No.** |
| 1 | Abstract | 05 |
| 2 | Introduction (to NLP) | 06 |
| 3 | Problem Statement, Objectives and Goals | 07 |
| 4 | Project Scope | 08 |
| 5 | Methodology | 09 |
| 6 | Modules identified and code | 10-14 |
| 7 | Project Implementation | 15 |
| 8 | Findings / Results of Analysis | 16 |
| 9 | Conclusions | 17 |
| 10 | References | 18 |

**ABSTRACT**

*Natural language processing, or NLP, is a type of artificial intelligence that deals with analyzing, understanding, and generating natural human languages so that computers can process written and spoken human language without using computer-driven language. Natural language processing, sometimes also called “computational linguistics,” uses both semantics and syntax to help computers understand how humans talk or write and how to derive meaning from what they say. This field combines the power of artificial intelligence and computer programming into an understanding so powerful that programs can even translate one language into another reasonably accurately. This field also includes voice recognition, the ability of a computer to understand what you say well enough to respond appropriately.*

*Hate speech detection is the task of detecting if communication such as text, audio, and so on contains hatred and or encourages violence towards a person or a group of people. This is usually based on prejudice against 'protected characteristics' such as their ethnicity, gender, sexual orientation, religion, age etc.*

*Hate speech is currently of broad and current interest in the domain of social media. The anonymity and flexibility afforded by the Internet has made it easy for users to communicate in an aggressive manner. And as the amount of online hate speech is increasing, methods that automatically detect hate speech is very much required. Moreover, these problems have also been attracting the Natural Language Processing and Machine Learning communities a lot. Therefore, the goal of this paper is to look at how Natural Language Processing applies in detecting hate speech.*

*Using various machine learning algorithms like Naive Bayes, Max Entropy, and Support Vector Machine, we provide research on twitter data streams. We have also discussed general challenges and applications of Sentiment Analysis on Twitter****.***

**INTRODUCTION**

Nowadays, the age of Internet has changed the way people express their views, opinions. It is now mainly done through blog posts, online forums, product review websites, social media, etc. Nowadays, millions of people are using social network sites like Facebook, Twitter, Google Plus, etc. to express their emotions, opinion and share views about their daily lives. Through the online communities, we get an interactive media where consumers inform and influence others through forums. Social media is generating a large volume of sentiment rich data in the form of tweets, status updates, blog posts, comments, reviews, etc. Moreover, social media provides an opportunity for businesses by giving a platform to connect with their customers for advertising. People mostly depend upon user generated content over online to a great extent for decision making. For e.g. if someone wants to buy a product or wants to use any service, then they firstly look up its reviews online, discuss about it on social media before taking a decision. The amount of content generated by users is too vast for a normal user to analyze. So there is a need to automate this, various sentiment analysis techniques are widely used.

Sentiment analysis (SA) tells user whether the information about the product is satisfactory or not before they buy it. Marketers and firms use this analysis data to understand about their products or services in such a way that it can be offered as per the user‟s requirements.

Textual Information retrieval techniques mainly focus on processing, searching or analyzing the factual data present. Facts have an objective component but,there are some other textual contents which express subjective characteristics. These contents are mainly opinions, sentiments, appraisals, attitudes, and emotions, which form the core of Sentiment Analysis (SA).

It offers many challenging opportunities to develop new applications, mainly due to the huge growth of available information on online sources like blogs and social networks. For example, recommendations of items proposed by a recommendation system can be predicted by taking into account considerations such as positive or negative opinions about those items by making use of SA.

**Problem Statement:**

In this project, we try to implement a  NLP **Twitter sentiment analysis model** that helps to overcome the challenges of identifying the sentiments of the tweets.

**Objectives:**

The objective of a Twitter sentimental analysis project is to identify and understand the sentiment of a given tweet or set of tweets. This is typically accomplished by using natural language processing and machine learning techniques to analyze the text of the tweets and determine their sentiment. The sentiment of a tweet can be positive, negative, or neutral, and the goal of the project is to accurately classify the sentiment of each tweet.

Once the sentiment of the tweets has been determined, the project can then be used to gain insights into how people are feeling about a particular topic or issue. This information can be valuable for businesses and organizations that want to understand public opinion on a given subject, as well as for researchers and analysts who are studying sentiment on a specific topic.

Overall, the objective of a Twitter sentimental analysis project is to provide a better understanding of the sentiment of a given population on a specific topic, which can be useful for a variety of applications.

**Goals:**

The goal of a Twitter sentimental analysis project is to accurately classify the sentiment of a given tweet or set of tweets. This involves using natural language processing and machine learning techniques to analyze the text of the tweets and determine their sentiment. The goal is to achieve a high level of accuracy in classifying the sentiment of each tweet, so that the results of the analysis can be used to gain valuable insights into how people are feeling about a particular topic or issue.

Once the sentiment of the tweets has been accurately classified, the project can be used to gain a better understanding of public opinion on a particular subject. This information can be valuable for businesses and organizations that want to monitor and understand public sentiment on a given topic, as well as for researchers and analysts who are studying sentiment on a specific topic.

Overall, the goal of a Twitter sentimental analysis project is to provide a better understanding of the sentiment of a given population on a specific topic, which can be useful for a variety of applications.

**Project Scope:**

Natural language processing (NLP) Twitter sentimental analysis project aims to analyze the sentiments expressed in tweets on the social media platform Twitter. The project scope may include tasks such as collecting and preprocessing Twitter data, applying NLP techniques to extract insights from the data, and visualizing the results to understand the overall sentiments expressed in the tweets.

The specific objectives and deliverables of the project will depend on the specific needs and goals of the project stakeholders. For example, the project may aim to identify trends in sentiments about a particular topic or brand, compare sentiments across different groups or demographics, or provide real-time updates on the sentiments expressed in tweets.

In terms of technical requirements, the project may require a combination of NLP algorithms and techniques, such as sentiment analysis and topic modeling, as well as tools for collecting, storing, and analyzing large amounts of Twitter data. The project may also require a user-friendly interface for visualizing and interpreting the results.

Overall, the scope of a NLP Twitter sentimental analysis project will vary depending on the specific goals and requirements of the project, but it typically involves applying NLP techniques to Twitter data to extract insights on sentiments expressed in tweets.

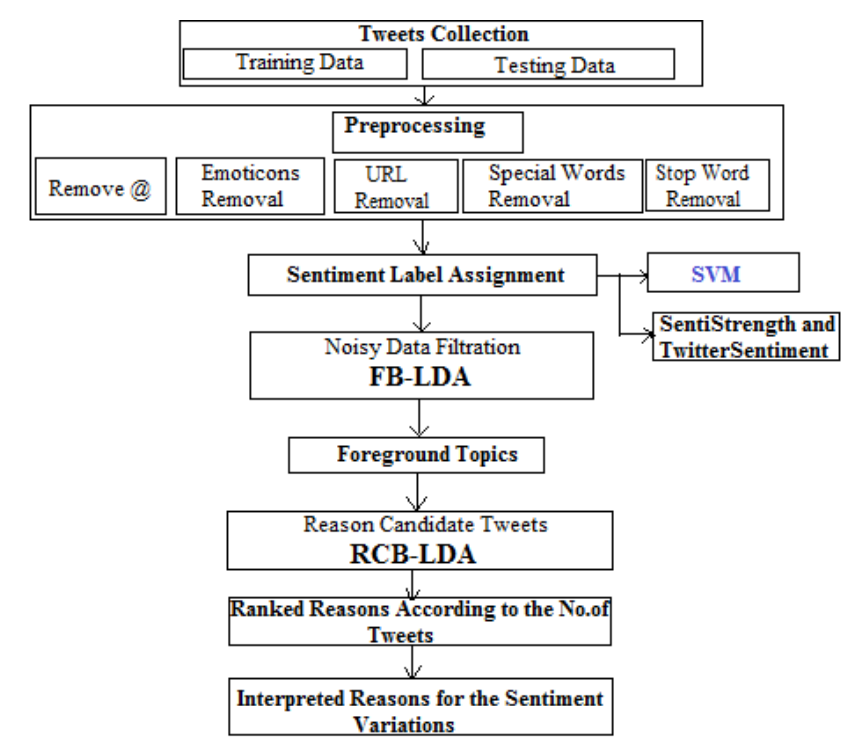
**Methodology**

In this paper we classify sentiments with the help of machine learning and natural language processing (NLP) algorithms, we use the datasets from Kaggle which was crawled from the internet and labeled positive/negative. The data provided comes with emoticons (emoji), usernames and hashtags which are required to be processed (so as to be readable) and converted into a standard form.

We also need to extract useful features from the text such unigrams and bigrams which is a form of representation of the “tweet”.

We use various machine learning algorithms based on NLP (Natural Language Processing) to conduct sentiment analysis using the extracted features. Finally, we report our experimental results and findings at the end.

**ACHITECTURAL DIAGRAM**



**Code:**

#Import Basic Libraries

import sklearn

import numpy as np

import pandas as pd

# training data

train = pd.read\_csv("train.csv")

# test data

test = pd.read\_csv("test.csv")

Data Exploration (Exploratory Data Analysis)

train.head()

test.tail()

# non-racist/sexist related tweets

sum(train["label"] == 0)

# racist/sexist related tweets

sum(train["label"] == 1)

# check if there are any missing values

train.isnull().sum()

#train.isnull().values.any()

Data cleaning

#install tweet-preprocessor to clean tweets

!pip install tweet-preprocessor

# remove special characters using the regular expression library

import re

#set up punctuations we want to be replaced

REPLACE\_NO\_SPACE = re.compile("(\.)|(\;)|(\:)|(\!)|(\')|(\?)|(\,)|(\")|(\|)|(\()|(\))|(\[)|(\])|(\%)|(\$)|(\>)|(\<)|(\{)|(\})")

REPLACE\_WITH\_SPACE = re.compile("(<br\s/><br\s/?)|(-)|(/)|(:).")

import preprocessor as p

# custum function to clean the dataset (combining tweet\_preprocessor and reguar expression)

def clean\_tweets(df):

tempArr = []

for line in df:

# send to tweet\_processor

tmpL = p.clean(line)

# remove puctuation

tmpL = REPLACE\_NO\_SPACE.sub("", tmpL.lower()) # convert all tweets to lower cases

tmpL = REPLACE\_WITH\_SPACE.sub(" ", tmpL)

tempArr.append(tmpL)

return tempArr

# clean training data

train\_tweet = clean\_tweets(train["tweet"])

train\_tweet = pd.DataFrame(train\_tweet)

# append cleaned tweets to the training data

train["clean\_tweet"] = train\_tweet

# compare the cleaned and uncleaned tweets

train.head(10)

# clean the test data and append the cleaned tweets to the test data

test\_tweet = clean\_tweets(test["tweet"])

test\_tweet = pd.DataFrame(test\_tweet)

# append cleaned tweets to the training data

test["clean\_tweet"] = test\_tweet

# compare the cleaned and uncleaned tweets

test.tail()

Test and Train split

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

# extract the labels from the train data

y = train.label.values

# use 70% for the training and 30% for the test

x\_train, x\_test, y\_train, y\_test = train\_test\_split(train.clean\_tweet.values, y,

stratify=y,

random\_state=1,

test\_size=0.3, shuffle=True)

Vectorize tweets using CountVectorizer

from sklearn.feature\_extraction.text import CountVectorizer

documents = ["Hi this is pranav",

"Data science is my passion and it is fun!",

"everyone please support me by funding"]

# initializing the countvectorizer

vectorizer = CountVectorizer()

# tokenize and make the document into a matrix

document\_term\_matrix = vectorizer.fit\_transform(documents)

# check the result

pd.DataFrame(document\_term\_matrix.toarray(), columns = vectorizer.get\_feature\_names())

from sklearn.feature\_extraction.text import CountVectorizer

# vectorize tweets for model building

vectorizer = CountVectorizer(binary=True, stop\_words='english')

# learn a vocabulary dictionary of all tokens in the raw documents

vectorizer.fit(list(x\_train) + list(x\_test))

# transform documents to document-term matrix

x\_train\_vec = vectorizer.transform(x\_train)

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

Model building

Apply Support Vetor Classifier (SVC)

from sklearn import svm

# classify using support vector classifier

svm = svm.SVC(kernel = 'linear', probability=True)

# fit the SVC model based on the given training data

prob = svm.fit(x\_train\_vec, y\_train).predict\_proba(x\_test\_vec)

# perform classification and prediction on samples in x\_test

y\_pred\_svm = svm.predict(x\_test\_vec)

Accuracy score for SVC

from sklearn.metrics import accuracy\_score

print("Accuracy score for SVC is: ", accuracy\_score(y\_test, y\_pred\_svm) \* 100, '%**')**

**Project Implementation:**

The steps in implementation are :

a) Load Twitter API

b) Load word dictionaries

c) Search twitter feeds

d) Getting text from feeds

e) Defining text cleaning functions

f) cleaning and splitting twitter feeds

g) Analyzing twitter feeds

h) Plotting high frequency negative and positive

words

**a)Load Twitter API :**

The first step is to register in the twitter application developer’s portal and get the authorization.

You need :

consumer\_key<- "Your Twitter Consumer key"

consumer\_secret<- "Your Twitter Consumer Secret key"

access\_token<- "Your Twitter Access Token key"

access\_secret<- "Your Twitter Access Secret key"

**b) Load word dictionaries :**

Next stride is to stack the arrangement of positive and negative assumptions words into your working catalog. The words are then gotten to and relegated to factors, positive and negative as demonstrated as follows.

**c) Search twitter feeds:**

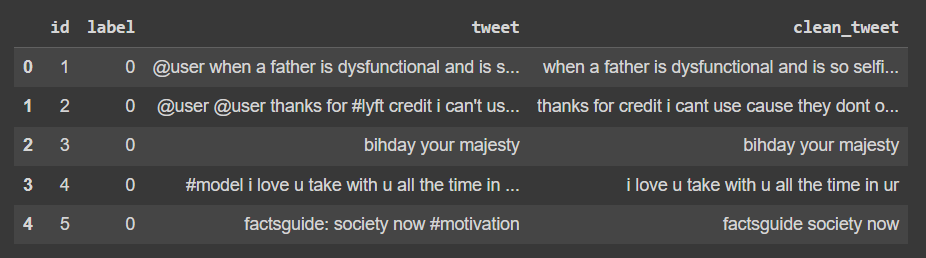
The following stride is characterizing a twitter seek string and relegating to a variable, Number of tweets to be removed is alloted to another variable, number. An ideal opportunity to play out the twitter hunt and extraction is influenced by this number. A moderate web association as well as unpredictable inquiry fields may bring about extra postponements.

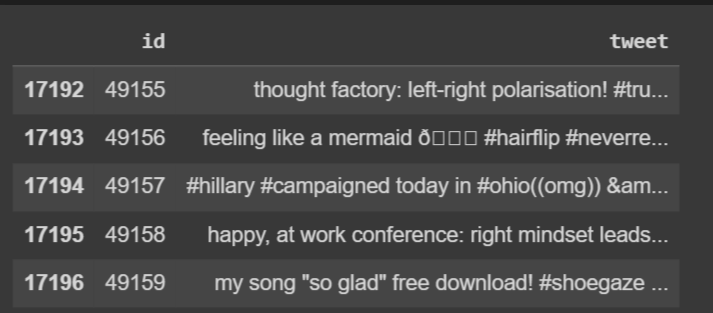
**d) Getting text from feeds :**

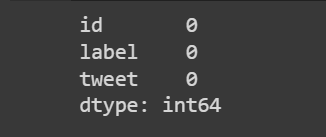
Twitter sustains have huge amounts of extra fields and implanted pointless data. We utilize the gettext() capacity to remove the content fields andappoint the rundown to a variable tweetT. The capacity is connected to every one of the 5000 tweets. The code beneath likewise indicates consequences of extraction for the initial 5

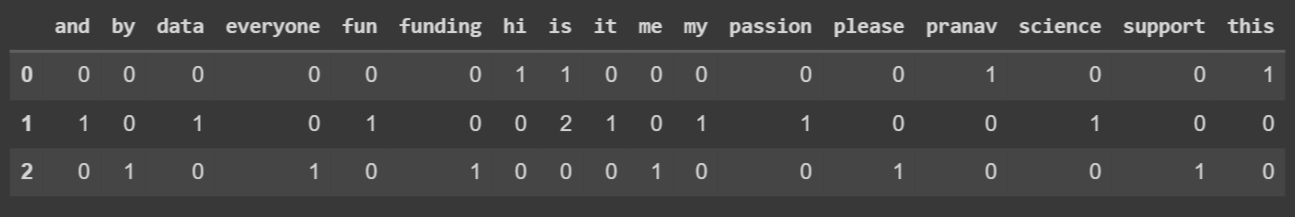
sustains.tweetT=lapply(tweet,function(t)t$getText()) head(tweetT,5)

**Results of Analysis:**









Accuracy score for SVC is: 94.86912086766085 %

**Conclusion:**

The task of sentiment analysis, especially in the domain of micro-bloging, is still in the developing stage and far from complete. So we propose a couple of ideas which we feel are worth exploring in the future and may result in further improved performance. Right now we have worked with only the very simplest svm model and count vectorizer we can improve those models by adding extra information like closeness of the word with a negation word.

The closer the negation word is to the unigram word whose prior polarity is to be calculated, the more it should affect the polarity. For example if the negation is right next to the word, it may simply reverse the polarity of that word and farther the negation is from the word the more minimized ifs effect should be. Apart from this, we are currently only focusing on unigrams and the effect of bigrams and trigrams may be explored. As reported in the literature review section when bigrams are used along with unigrams this usually enhances performance.

Right now we are exploring Parts of Speech separate from the svm model, we can try to incorporate POS information within our svm models in future. So say instead of calculating a single probability for each word like P(word | obj) we could instead have multiple probabilities for each according to the Part of Speech the word belongs to. For example we may have P(word | obj, verb), P(word | obj, noun) and P(word | obj, adjective). Used a somewhat similar approach and claims that appending POS information for every unigram results in no significant change in performance, while there is a significant decrease in accuracy if only adjective unigrams are used as features.

However these results are for classification of reviews and may be verified for sentiment analysis on micro blogging websites like Twitter. One more feature we that is worth exploring is whether the information about relative position of word in a tweet has any effect on the performance of the classifier. Although Pang et al. explored a similar feature and reported negative results, their results were based on reviews which are very different from tweets and they worked on an extremely simple model.

**References:**

[1] Albert Biffet and Eibe Frank. Sentiment Knowledge Discovery in Twitter Streaming Data. Discovery Science, Lecture Notes in Computer Science, 2010, Volume 6332/2010, 1-15, DOI: 10.1007/978-3-642-16184-1\_1

[2] Alec Go, Richa Bhayani and Lei Huang. Twitter Sentiment Classification using Distant Supervision. Project Technical Report, Stanford University, 2009.

[3] Alexander Pak and Patrick Paroubek. Twitter as a Corpus for Sentiment Analysis and Opinion Mining. In Proceedings of international conference on Language Resources and Evaluation (LREC), 2010.

[4] Andranik Tumasjan, Timm O. Sprenger, Philipp G. Sandner and Isabell M. Welpe. Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment. In Proceedings of AAAI Conference on Weblogs and Social Media (ICWSM), 2010. Project Thesis Report 51

[5] Bo Pang, Lillian Lee and Shivakumar Vaithyanathan. Thumbs up? Sentiment Classification using Machine Learning Techniques. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2002.

[6] Chenhao Tan, Lilian Lee, Jie Tang, Long Jiang, Ming Zhou and Ping Li. User Level Sentiment Analysis Incorporating Social Networks. In Proceedings of ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD), 2011.

[7] Efthymios Kouloumpis, Theresa Wilson and Johanna Moore. Twitter Sentiment Analysis: The Good the Bad and the OMG! In Proceedings of AAAI Conference on Weblogs and Social Media (ICWSM), 2011.

[8] Hatzivassiloglou, V., & McKeown, K.R.. Predicting the semantic orientation of adjectives. In Proceedings of the 35th Annual Meeting of the ACL and the 8th Conference of the European Chapter of the ACL, 2009.

[9] Johann Bollen, Alberto Pepe and Huina Mao. Modelling Public Mood and Emotion: Twitter Sentiment and socio-economic phenomena. In Proceedings of AAAI Conference on Weblogs and Social Media (ICWSM), 2011.

[10] Luciano Barbosa and Junlan Feng. Robust Sentiment Detection on Twitter from Biased and Noisy Data. In Proceedings of the international conference on Computational Linguistics (COLING), 2010.

[11] Peter D. Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL), 2002.

[12] Rudy Prabowo and Mike Thelwall. Sentiment Analysis: A Combined Approach. Journal of Infometrics, Volume 3, Issue 2, April 2009, Pages 143-157, 2009.

[13] Samuel Brody and Nicholas Diakopoulus. Cooooooooooooooollllllllllllll!!!!!!!!!!!!!! Using Word Lengthening to Detect Project Thesis Report 52 Sentiment in Microblogs. In Proceedings of Empirical Methods on Natural Language Processing (EMNLP), 2011.

[14] Soo-Min Kim and Eduard Hovy. Determining the Sentiment of Opinions. In Proceedings of International Conference on Computational Linguistics (ICCL), 2004.

[15] Stefano Baccianella, Andrea Esuli, Fabrizio Sebastiani. SENTIWORDNET 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In Proceedings of international conference on Language Resources and Evaluation (LREC), 2010.

[16] Theresa Wilson, Janyce Wiebe and Paul Hoffmann. Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis. In the Annual Meeting of Association of Computational Linguistics: Human Language Technologies (ACL-HLT), 2005.

[17] Ian H. Witten, Eibe Frank & Mark A. Hall. Data Mining – Practical Machine Learning Tools and Techniques.

[18] Ricgard O. Duda, Peter E. Hart & David G. Stork: Pattern Classification.

[19] Steven Bird, Even Klein & Edward Loper. Natural Language Processing with Python.