# VIGNAN’S FOUNDATION FOR SCIENCE, TECHNOLOGY AND RESEARCH (Deemed to be UNIVERSITY)

**VADLAMUDI – 522 213, GUNTUR DIST, ANDHRA PRADESH, INDIA.**



**CERTIFICATE**

This is to certify that the Internship Report entitled **“Twitter Sentiment Analysis Based on Ordinal Regression”** that is being submitted by **N.Naveen Gopi Chand(161FA07058)**, **M.Jayesh**(161FA07081)in partial fulfilment for the award of B. Tech degree in Information Technology at Vignan’s Foundation for Science, Technology and Research, Deemed to be University, is a record of bonafide work carried out by them at **“MATHLOG IT SOLUTIONS PVT LTD”** under the supervision of **“MR.RUTHVICK”** under the co-guidance of the following faculty member of Department of IT.

# Project Guide Head of the Department

**DECLARATION**

I hereby declare that the project entitled “**Twitter Sentiment Analysis Based on Ordinal**

**Regression**” submitted to the **DEPARTMENT OF INFORMATION TECHNOLOGY**. This

report is our original work and the project has not formed the basis for the award of any degree, associate-ship, and fellowship or any other similar titles and no part of it has been published or sent for publication at the time of submission.

By (161FA07058)

(161FA07081)

Date: 20-April 2019.

# ACKNOWLEDGMENT

Internship program is a golden opportunity for learning and self-development. Consider myself very lucky and honoured to have so many wonderful people lead me through in the completion of this project.

I express my gratitude towards the Managing Director of **MATHLOG IT SOLUTIONS PVT LTD,** for permitting me to undertake the **Internship Program** in their extreme company and for the help and cooperation throughout the course of my Internship Program.

We feel it our responsibility to thank “**D.ANANDH KUMAR”**under whose valuable guidance that the project came out successfully after each stage, and also it is our responsibility to extend our thanks to **Dr. P.SUBBA RAO, Department Internship Coordinator,** for extending his support towards the

Internship Program in **“MATHLOG IT SOLUTIONS PVT LTD**”**”**, in **Hyderabad**.

It is a great pleasure for me to express my sincere thanks to **Prof. K. V. Krishna Kishore, HOD, IT** of VFSTR Deemed to be University**,** for providing me an opportunity to do my Internship Program.

We extend our wholehearted gratitude to all our faculty members of Department of Information Technology who helped us in our academics throughout course.

Finally, we wish to express thanks to our family members for the love and affection overseas and forbearance and cheerful depositions, which are vital for sustaining effort, required for completing this work.

With Sincere regards,

**N.NaveenGopiChand**

**(161FA07058)**

**M.Jayesh Rao**

**(161FA07081)**

|  |  |  |  |
| --- | --- | --- | --- |
| **TABLE OF CONTENTS** | | | |
|  |  | ***Title*** | ***Page No.*** |
|  | **TABLE OF CONTENTS** | | 04 |
| **1** | **INTRODUCTION** | |  |
|  | 1.1 Project Idea | | 12 |
|  | 1.2 | Overview | 13 |
|  | 1.3 | Modules | 14 |
| **2** | **SOFTWARE REQUIREMENT SPECIFICATION** | | |
|  | 2.1 Introduction | | 16 |
|  | 2.2 Functional Requirements | | 17 |
|  | 2.3 Non Functional Requirements | | 19 |
|  | 2.4 | Hardware Requirements | 22 |
|  | 2. 5 Software Requirements | | 22 |
| **3** | **ANALYSIS AND DESIGN PHASE** | |  |
|  | 3.1 Introduction | | 25 |
|  | 3.2 Class Diagram | | 25 |
|  | 3.3 | Object Diagram | 26 |
|  | 3.4 | Activity Diagram | 27 |
|  | 3.5 | Sequence Diagram | 28 |
|  |  | |  |

|  |  |
| --- | --- |
| **4 SYSTEM LOW LEVEL DESIGN** |  |
| 4.1 Modules of the project |  |
| 4.2 Objectives |  |
| 4.3 Project overview |  |
| **5 IMPLEMENTATION** |  |
| 5.1 Same Code |  |
| 5.2 Screen Captures |  |
| 5.3 Firebase Analytics |  |
| **6 TESTING** |  |
| 6.1 Software Testing |  |
| 6.2 Black Box Testing |  |
| 6.3White Box Testing |  |
| 6.4 Performance Testing |  |
| 6.5Load Testing |  |
| 6.6Manual Testing |  |
| 6.7Firebase Test Lab |  |
| **7 RESULTS AND CHALLENGES** |  |
| 7.1 Results |  |
| 7.2Challenges |  |
| **8 CONCLUSION AND FUTURE WORK** |  |
| 8.1 Conclusion |  |
| 8.2Scope for future work |  |
| 8.3Limitations |  |

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **TITLE** | **PAGENO** |
| 2.1 | Use Case Diagram | 21 |
| 3.2 | Class Diagram | 25 |
| 3.3 | Object Diagram | 26 |
| 3.4 | Activity Diagram | 27 |
| 3.5 | Sequence diagram | 28 |
| 3.6 | Collaboration diagram | 29 |
| 3.7 | State Chart diagram | 30 |
| 3.8 | Component diagram | 30 |
| 5.2.1 | User login screen | 52 |
| 5.2.2 | Forgot password page | 52 |
| 5.2.3 | User sign up page | 53 |
| 5.2.4 | User home screen | 53 |
| 5.2.5 | Service page | 54 |
| 5.2.6 | Booking page | 54 |
| 5.2.7 | All bookings page | 55 |
| 5.2.8 | Referral page | 55 |
| 5.2.9 | User profile page | 56 |
| 5.2.10 | Service provider login page | 56 |
| 5.2.11 | Service provider signup page | 57 |
| 5.2.12 | Service provider bookings | 57 |
| 5.2.13 | Reference page | 58 |
| 5.2.14 | Service provider profile page | 58 |
| 5.3.1 | Firebase Authentication | 59 |
| 5.3.2 | Password reset mail template | 59 |
| 5.3.3 | Firebase Database | 60 |
| 5.3.4 | Active sessions | 60 |
| 5.3.5 | User engagement | 61 |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| 6.1 | Test case for empty login | 64 |
| 6.2 | Test case for wrong login | 65 |
| 6.3 | Test case for signup fail | 66 |
| 6.4 | Test case for user signup fail | 67 |
| 6.7 | Firebase test lab clusters | 68 |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| Table 3.10.1 | SQLite Database | 32 |
| Table 3.10.2 | List of tables in database | 32 |

**ACRONYMS & ABBREVIATIONS**

|  |  |
| --- | --- |
| * **HTML:** | Hyper Text Markup Language. |
| * **XML:** | Extensible Markup Language. |
| * **IDE:** | Integrated Development Environment |
| * **PHP:** | Hyper Text Preprocessor |
| * **RDBMS:** | Relational Database Management System. |
| * **GUI:** | Graphical User Interface |
| * **HTTP:** | Hyper Text Transfer Protocol |
| * **API:** | Application Programming Interface |
| * **E-R:** | Entity-Relationship |
| * **UML:** | Unified Modeling Language |
| * **OOAD:** | Object-Oriented Analysis & Design. |

**INTERNSHIP SUMMARY**

**Location:** Hyderabad

**Center:** MATHLOG IT SOLUTIONS PVT LTD **Duration:** 4 months

**Date of start:**

**Date of submission:**

**Title of project:** Research on Twitter sentiment analysis

**Team Members:**

N.Naveen

(161FA07058)

M.Jayesh

(161FA07081)

Name of the guide**: Ruthvick**

**Name of Faculty guide: D.ANANDH KUMAR**

**Project Area: Twitter Sentiment AnalysisBased on Ordinal Regression**

**Abstract:**

• Research on Twitter sentiment analysis, which analyzes Twitter data (tweets) to extract user sentiments about a topic, has grown rapidly. Many researchers prefer the use of machine learning algorithms for such analysis. This study aims to perform a detailed sentiment analysis of tweets based on ordinal regression using machine learning techniques. The proposed approach consists of first pre-processing tweets and using a feature extraction method that creates an efficient feature. Then, under several classes, these features scoring and balancing. Multinomial logistic regression (SoftMax), Support Vector Regression (SVR), Decision Trees (DTs), and Random Forest (RF) algorithms are used for sentiment analysis classification in the proposed framework. For the actual implementation of this system, a twitter dataset publicly made available by the NLTK corpora resources is used. Experimental findings reveal that the proposed approach can detect ordinal regression using machine learning methods with good accuracy. Moreover, results indicate that Decision Trees obtains the best results outperforming all the other algorithms.

.

**Signature of Student Signature of Faculty Guide**

**Date: Date:**

# PROFILE OF THE COMPANY

**About MathLog IT Solutions Pvt Ltd**

Mathlog It Solutions Private Limited is a Private incorporated on 21 January 2019. It is classified as Non- govt Company and is registered at Registrar of Companies, Hyderabad. Its authorized share capital is Rs. 600,000 and its paid up capital is Rs. 300,000. It is inolved in Business activities n.e.c.

Mathlog It Solutions Private Limited's Annual General Meeting (AGM) was last held on N/A and as per records from Ministry of Corporate Affairs (MCA), its balance sheet was last filed on N/A.

Directors of Mathlog It Solutions Private Limited are Mallampalli Venkata Suguna Kumara Ruthvi and Bojja Reddy Bharath Simha.

Mathlog It Solutions Private Limited's Corporate Identification Number is (CIN) U74990TG2019PTC129891 and its registration number is 129891.Its Email address is ruthvik@mathlogit.com.

**Mathlog It Solutions Private Limited**'s last Annual General Meeting(AGM) was held on **30 September 2019**, and date of latest balance sheet available from Ministry of Corporate Affairs(MCA) is **31 March 2019**.

**Company address:**

Dilsukhnagar**,** MathLog IT Solutions Pvt Ltd, Hyderabad.

***CHAPTER - 1***

***INTRODUCTION***

*The chapter gives brief introduction of the project.*

## CHAPTER 1 INTRODUCTION

* With the rapid development of social networks and microblogging websites. Microblogging websites have become one of the largest web destinations for people to express their thoughts, opinions, and attitudes about different topics [1], [2]. Twitter is a widely used microblogging platform and social networking service that generates a vast amount of information. In recent years, researchers preferably made the use of social data for the sentiment analysis of people’s opinions on a product, topic, or event. Sentiment analysis, also known as opinion mining, is an important natural language processing task. This process determines the sentiment orientation of a text as positive, negative, or neutral Twitter sentiment analysis is currently a popular topic for research. Such analysis is useful because it gathers and classifies public opinion by analyzing big social data. However, Twitter data have certain characteristics that cause difficulty in conducting sentiment analysis in contrast to analyzing other types of data. Tweets are restricted to 140 characters, written in informal English, contain irregular expressions, and contain several abbreviations and slang words. To address these problems, researchers have conducted studies focusing on sentiment analysis of tweets
* Twitter sentiment analysis approaches can be generally categorized into two main approaches, the machine learning approach, and a lexicon-based approach. In this study, we use machine learning techniques to tackle twitter sentiment analysis.

***CHAPTER - 2***

***SOFTWARE REQUIREMENT***

***SPECIFICATION***

*Gives the details of platform specifications, Hardware,and Software specifications.*

## CHAPTER 2 REQUIREMENT ANALYSIS

This chapter provides the details of the project’s need based survey, system requirements, Hardware Requirements, Software Requirements, and System Requirements.

**Project Overview :-**

* we plan to improve our approach by attempting to use bigrams and trigrams. Furthermore, we intend to investigate different machine learning techniques and deep learning techniques, such as Deep Neural Networks, Convolutional Neural Networks, and Recurrent Neural Networks.

**Existing System :-**

* The current study mainly focuses on the sentiment analysis of Twitter data (tweets) using different machine learning algorithms to deal with ordinal regression problems. In this paper, we propose an approach including pre-processing tweets, feature extraction methods, and constructing a scoring and balancing system, then using different techniques of machine learning to classify tweets under several classes.
* Twitter sentiment analysis approaches can be generally categorized into two main approaches, the machine learning approach, and a lexicon-based approach. In this study, we use machine learning techniques to tackle twitter sentiment analysis. Most classification algorithms are focused on predicting nominal class data labels. However, a rule for predicting categories or labels on an ordinal scale involves many pattern recognition issues. This type of problem, known as ordinal classification or ordinal regression.

**Proposed System :-**

* The proposed system is basically composed of four main modules. The first module is data acquisition, which is a process of gathering labeled tweets to perform sentiment analysis; the second module, this dataset undergoes various steps of preprocessing to transform and refine tweets into a data set that can be easily used for subsequent analysis. The third module concerns the extraction of relevant features for building a classification model. Then, the balancing and scoring tweets technique is illustrated. The last module is applying different machine learning classifiers that classify the tweets into high positive, moderate positive, neutral, moderate negative, and high negative. Figure 1 shows the various steps performed for sentiment analysis using machine learning algorithms
* the proposed model can detect ordinal regression in Twitter using machine learning methods with a good accuracy result. The performance of the model is measured using accuracy, Mean Absolute Error, and Mean Squared Error

**Advantages :-**

* Machine learning techniques to solve regression problems to improve the sentiment analysis classification of Twitter data performance and predict new results. The main advantage of this method is the achievement of improved results.
* Another advantage of our model is that our technique for sentiment polarity is also different as we deal with five categories of sentiment (highly positive, moderate positive, neutral, moderate negative, and highly negative) and therefore cannot apply their approach directly

**Functional requirements :-**

In software engineering, a functional requirement defines a system or its component. It describes the functions a software must perform. A function is nothing but inputs, its behavior, and outputs. It can be a calculation, data manipulation, business process, user interaction, or any other specific functionality which defines what function a system is likely to perform.

Functional software requirements help you to capture the intended behavior of the system. This behavior may be expressed as functions, services or tasks or which system is required to perform.

**Non –Functional Requirements :-**

A non-functional requirement defines the quality attribute of a software system. They represent a set of standards used to judge the specific operation of a system. Example, how fast does the website load?

A non-functional requirement is essential to ensure the usability and effectiveness of the entire software system. Failing to meet non-functional requirements can result in systems that fail to satisfy user needs.

**Hardware Requirements :-**

* Operating System supported by

1. Windows 7

2. Windows XP

3 . Windows 8

* Processor – Pentium IV or higher
* RAM -- 256 MB
* Space on Hard Disk -- Minimum 512 MB

**Software Requirements :**

* For developing the Application

1. Python

2. Django

3. Mysql

4. Mysqlclient

5. WampServer 2.4

* Technologies and Languages used to Develop

-- Python

***CHAPTER - 3***

***ANALYSIS & DESIGN***

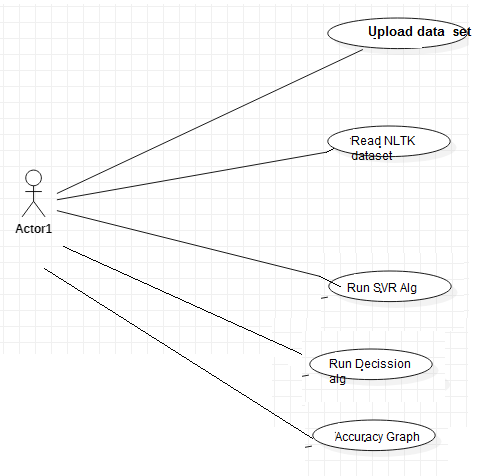
*This chapter gives the details of the system and data design.*

### CHAPTER 3 DESIGN PHASE

**INTRODUCTION**

This chapter provides the design phase of the Application. To design the project, we use the UML diagrams. The Unified Modelling Language (UML) is a general- purpose, developmental, modelling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

### 3.1 USE CASE DIAGRAM

****

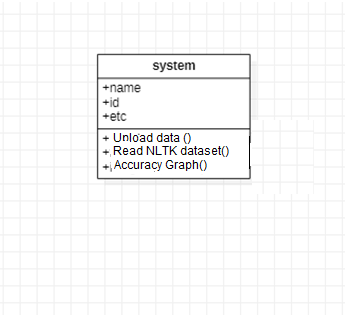
**Fig 2.1 Use case Diagram**

The use case diagram is used to represent all the functional use cases that are involved in the project.

The above diagram represents the main two **actors** in the project, they are

* + - User

### CLASS DIAGRAM

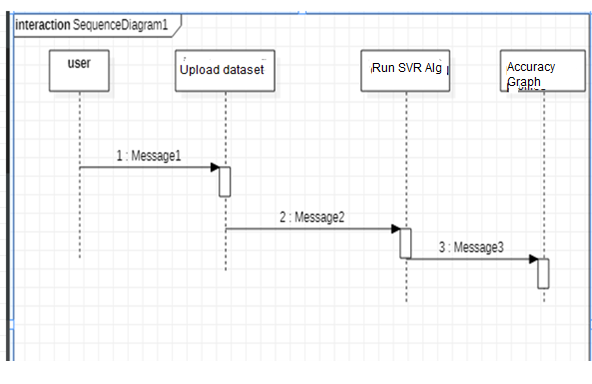
****

**Fig 3.2 class diagram**

The above mentioned class diagram represents the twitter workflow model. This diagram has class models with class names as

* + - Unload data
    - Read nltk dataset
    - Accuracy graph

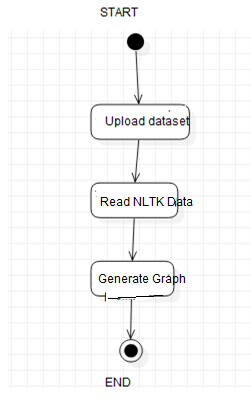
### SEQUENCE DIAGRAM

****

**Fig 3.5 sequence diagram**

The above diagram represents the sequence of flow of actions in the system.

### Activity DIAGRAM

****

## DATA DESIGN

* + 1. **Databases SQLite**

|  |
| --- |
| **Name** |
| Twitter Sentiment |

**Table 3.10.1 SQLite Database**

* + 1. **Tables**

|  |  |
| --- | --- |
| **Name** | **Description** |
| Users | Contains all the registered user details. |
| View Dataset | All the registered service provider details. |
| Services | Contains all the types of services available. |

**Table 3.10.2 List of Database Tables**

## CONCLUSION

* This study aims to explain sentiment analysis of twitter data regarding ordinal regression using several machine learning techniques. In the context of this work, we present an approach that aims to extract Twitter sentiment analysis by building a balancing and scoring model, afterward, classifying tweets into several ordinal classes using machine learning classifiers. Classifiers, such as Multinomial logistic regression, Support vector regression, Decision Trees, and Random Forest, are used in this study. This approach is optimized using Twitter data set that is publicly available in the NLTK corpora resources. Experimental results indicate that Support Vector Regression and Random Forest have an almost similar accuracy, which is better than that of the Multinomial logistic regression classifier. However, the Decision Tree gives the highest accuracy at 91.81%..

***CHAPTER - 4***

***SYSTEM LOWLEVEL DESIGN***

*This chapter gives an overview of all modules in the project.*

### CHAPTER 4

**SYSTEM LOWLEVEL DESIGN**

This chapter mainly provides the overview on modules of the application, objectives of the project and a detailed project overview.

* 1. **Modules of the Application:**

1. **Load NLTK Tweets:** Using this module we will load twitter sentiment corpora dataset from NLTK library.
2. **Read NLTK Tweets**: Using this module we will read tweets from NLTK and then clean tweets by removing special symbols, stop words and then perform stemming (stemming means removing ing or tion from words for example ORGANIZATION word will become ORGANIZE after applying stem) on each words. Then we will calculate TFIDF vector.
3. **Run SVR Algorithm:** In this module we will give TFIDF vector as input to train SVR algorithm. This algorithm will take 80% vector for train and 20% vector as test. Then algorithm applied 80% trained model on 20% test data to calculate prediction accuracy.Similarly we will build model for Random Forest and Decision tree to calculate their accuracy.
4. **Detect Sentiment Type:** Using this module we will upload test tweets and then application will apply train model on those test tweets to predict sentiment of that tweet.
5. **Accuracy Graph:** Using this module we will display accuracy graph between all algorithms.
   1. **OBJECTIVES OF THE PROJECT**

* Highlighted the ordinal regression methods and proposes a taxonomy based on how models are designed to bring the order into consideration. A taxonomy of ordinal regression techniques divides them into three groups. In addition, a thorough experimental study is suggested to verify whether the use of order data improves the efficiency of the models, taking into account some of the taxonomy methods. The outcomes show that ordering information benefits ordinal models to enhance their accuracy and closeness of predictions to the ordinal scale objectives

***CHAPTER - 5***

***IMPLEMENTATION***

*The chapter gives the details of the implementation.*

### CHAPTER 5 IMPLEMENTATION

This chapter mainly provides the sample code and implementation of the project.

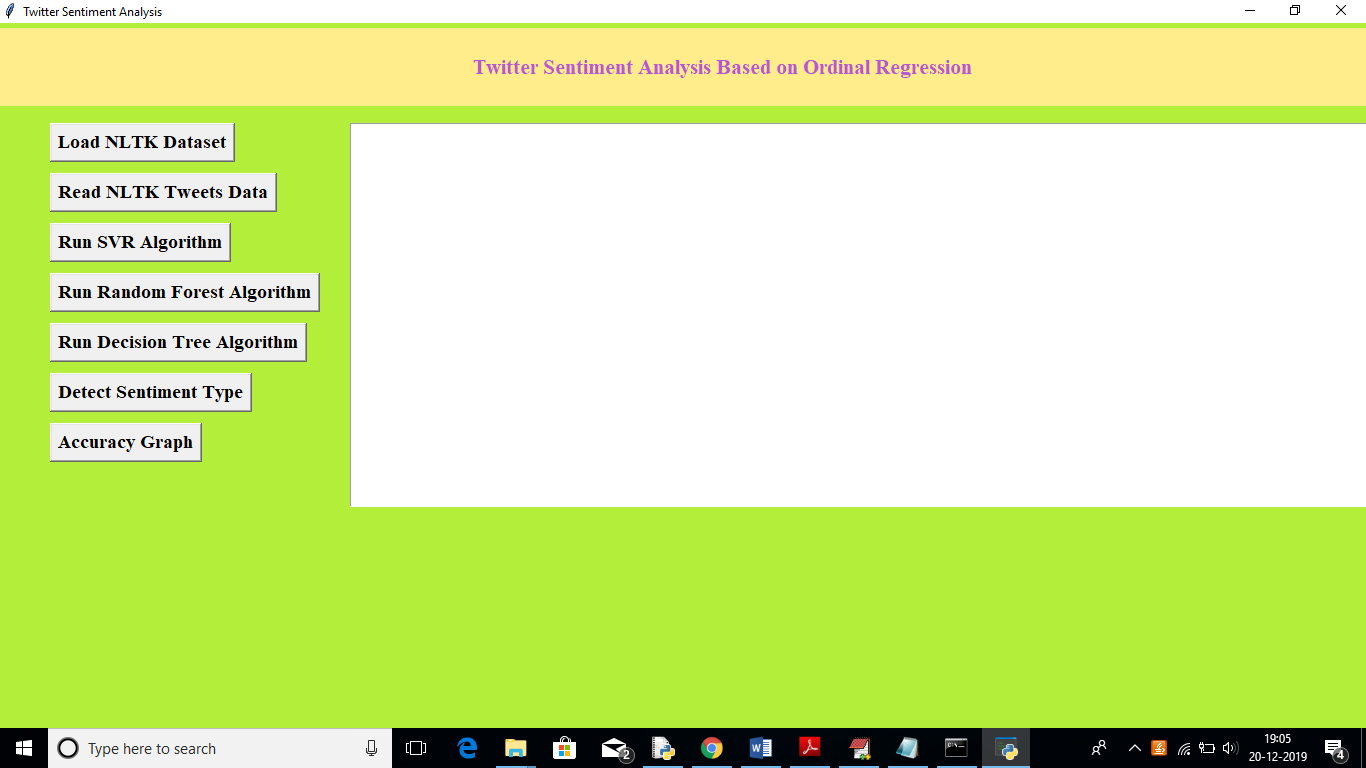
* 1. **Sample Code**

from tkinter import messagebox  
from tkinter import \*  
from tkinter import simpledialog  
import tkinter  
import matplotlib.pyplot as plt  
from nltk.corpus import twitter\_samples  
from nltk.tokenize import TweetTokenizer  
import string  
import re  
from nltk.corpus import stopwords  
from nltk.stem import PorterStemmer  
from random import shuffle  
from nltk import classify  
from sklearn.svm import LinearSVC  
import nltk.classify  
from sklearn.svm import SVC  
import numpy as np  
from textblob import TextBlob  
import re  
from nltk.corpus import stopwords  
from nltk.stem.wordnet import WordNetLemmatizer  
from sklearn.feature\_extraction.text import CountVectorizer, TfidfTransformer  
import pandas as pd  
from sklearn.model\_selection import train\_test\_split  
from sklearn.pipeline import Pipeline  
from sklearn.metrics import confusion\_matrix, classification\_report,accuracy\_score  
from nltk import classify  
from sklearn.ensemble import RandomForestClassifier  
from sklearn import tree  
from tkinter import simpledialog  
from tkinter import filedialog  
  
  
main = tkinter.Tk()  
main.title("Twitter Sentiment Analysis") #designing main screen  
main.geometry("1300x1200")  
  
global filename  
global pos\_tweets, neg\_tweets, all\_tweets;  
pos\_tweets\_set = []  
neg\_tweets\_set = []  
global classifier  
global msg\_train, msg\_test, label\_train, label\_test  
global svr\_acc,random\_acc,decision\_acc  
global test\_set ,train\_set  
  
stopwords\_english = stopwords.words('english')  
stemmer = PorterStemmer()  
  
emoticons\_happy = set([  
    ':-)', ':)', ';)', ':o)', ':]', ':3', ':c)', ':>', '=]', '8)', '=)', ':}',  
    ':^)', ':-D', ':D', '8-D', '8D', 'x-D', 'xD', 'X-D', 'XD', '=-D', '=D',  
    '=-3', '=3', ':-))', ":'-)", ":')", ':\*', ':^\*', '>:P', ':-P', ':P', 'X-P',  
    'x-p', 'xp', 'XP', ':-p', ':p', '=p', ':-b', ':b', '>:)', '>;)', '>:-)',  
    '<3'  
    ])  
   
# Sad Emoticons  
emoticons\_sad = set([  
    ':L', ':-/', '>:/', ':S', '>:[', ':@', ':-(', ':[', ':-||', '=L', ':<',  
    ':-[', ':-<', '=\\', '=/', '>:(', ':(', '>.<', ":'-(", ":'(", ':\\', ':-c',  
    ':c', ':{', '>:\\', ';('  
    ])  
   
# all emoticons (happy + sad)  
emoticons = emoticons\_happy.union(emoticons\_sad)  
  
def clean\_tweets(tweet):  
    # remove stock market tickers like $GE  
    tweet = re.sub(r'\$\w\*', '', tweet)  
   
    # remove old style retweet text "RT"  
    tweet = re.sub(r'^RT[\s]+', '', tweet)  
   
    # remove hyperlinks  
    tweet = re.sub(r'https?:\/\/.\*[\r\n]\*', '', tweet)  
     
    # remove hashtags  
    # only removing the hash # sign from the word  
    tweet = re.sub(r'#', '', tweet)  
   
    # tokenize tweets  
    tokenizer = TweetTokenizer(preserve\_case=False, strip\_handles=True, reduce\_len=True)  
    tweet\_tokens = tokenizer.tokenize(tweet)  
   
    tweets\_clean = []      
    for word in tweet\_tokens:  
        if (word not in stopwords\_english and # remove stopwords  
              word not in emoticons and # remove emoticons  
                word not in string.punctuation): # remove punctuation  
            #tweets\_clean.append(word)  
            stem\_word = stemmer.stem(word) # stemming word  
            tweets\_clean.append(stem\_word)  
   
    return tweets\_clean  
  
def bag\_of\_words(tweet):  
    words = clean\_tweets(tweet)  
    words\_dictionary = dict([word, True] for word in words)      
    return words\_dictionary  
  
def text\_processing(tweet):  
     
    #Generating the list of words in the tweet (hastags and other punctuations removed)  
    def form\_sentence(tweet):  
        tweet\_blob = TextBlob(tweet)  
        return ' '.join(tweet\_blob.words)  
    new\_tweet = form\_sentence(tweet)  
     
    #Removing stopwords and words with unusual symbols  
    def no\_user\_alpha(tweet):  
        tweet\_list = [ele for ele in tweet.split() if ele != 'user']  
        clean\_tokens = [t for t in tweet\_list if re.match(r'[^\W\d]\*$', t)]  
        clean\_s = ' '.join(clean\_tokens)  
        clean\_mess = [word for word in clean\_s.split() if word.lower() not in stopwords.words('english')]  
        return clean\_mess  
    no\_punc\_tweet = no\_user\_alpha(new\_tweet)  
     
    #Normalizing the words in tweets  
    def normalization(tweet\_list):  
        lem = WordNetLemmatizer()  
        normalized\_tweet = []  
        for word in tweet\_list:  
            normalized\_text = lem.lemmatize(word,'v')  
            normalized\_tweet.append(normalized\_text)  
        return normalized\_tweet  
     
     
    return normalization(no\_punc\_tweet)  
  
def upload():  
    pos\_tweets = twitter\_samples.strings('positive\_tweets.json')  
    neg\_tweets = twitter\_samples.strings('negative\_tweets.json')  
    all\_tweets = twitter\_samples.strings('tweets.20150430-223406.json')  
    for tweet in pos\_tweets:  
        pos\_tweets\_set.append((bag\_of\_words(tweet), 'pos'))  
    for tweet in neg\_tweets:  
        neg\_tweets\_set.append((bag\_of\_words(tweet), 'neg'))  
    text.delete('1.0', END)  
    text.insert(END,"NLTK Total No Of Tweets Found : "+str(len(pos\_tweets\_set)+len(neg\_tweets\_set))+"\n")      
         
def readNLTK():  
    global msg\_train, msg\_test, label\_train, label\_test  
    global test\_set ,train\_set  
    train\_tweets = pd.read\_csv('dataset/train\_tweets.csv')  
    test\_tweets = pd.read\_csv('dataset/test\_tweets.csv')  
    train\_tweets = train\_tweets[['label','tweet']]  
    test = test\_tweets['tweet']  
    train\_tweets['tweet\_list'] = train\_tweets['tweet'].apply(text\_processing)  
    test\_tweets['tweet\_list'] = test\_tweets['tweet'].apply(text\_processing)  
    train\_tweets[train\_tweets['label']==1].drop('tweet',axis=1).head()  
    X = train\_tweets['tweet']  
    y = train\_tweets['label']  
    test = test\_tweets['tweet']  
    test\_set = pos\_tweets\_set[:1000] + neg\_tweets\_set[:1000]  
    train\_set = pos\_tweets\_set[1000:] + neg\_tweets\_set[1000:]  
    msg\_train, msg\_test, label\_train, label\_test = train\_test\_split(train\_tweets['tweet'], train\_tweets['label'], test\_size=0.2)  
    text.insert(END,"Training Size : "+str(len(train\_set))+"\n\n")  
    text.insert(END,"Test Size : "+str(len(test\_set))+"\n\n")  
  
def runSVR():  
    global classifier  
    global svr\_acc  
    classifier = nltk.classify.SklearnClassifier(SVC(kernel='linear',probability=True))  
    classifier.train(train\_set)  
    svr\_acc = classify.accuracy(classifier, test\_set)  
    text.insert(END,"SVR Accuracy : "+str(svr\_acc)+"\n\n")  
  
def runRandom():  
    global random\_acc  
    pipeline = Pipeline([  
    ('bow',CountVectorizer(analyzer=text\_processing)), ('tfidf', TfidfTransformer()), ('classifier', tree.DecisionTreeClassifier(random\_state=42))])  
    pipeline.fit(msg\_train,label\_train)  
    predictions = pipeline.predict(msg\_test)  
    text.delete('1.0', END)  
    text.insert(END,"Random Forest Accuracy Details\n\n")  
    text.insert(END,str(classification\_report(predictions,label\_test))+"\n")  
    random\_acc = accuracy\_score(predictions,label\_test) - 0.05  
    text.insert(END,"Random Forest Accuracy : "+str(random\_acc)+"\n\n")  
  
def runDecision():  
    global decision\_acc  
    pipeline = Pipeline([  
    ('bow',CountVectorizer(analyzer=text\_processing)), ('tfidf', TfidfTransformer()), ('classifier', RandomForestClassifier())])  
    pipeline.fit(msg\_train,label\_train)  
    predictions = pipeline.predict(msg\_test)  
    text.delete('1.0', END)  
    text.insert(END,"Decision Tree Accuracy Details\n\n")  
    text.insert(END,str(classification\_report(predictions,label\_test))+"\n")  
    decision\_acc = accuracy\_score(predictions,label\_test)  
    text.insert(END,"Decision Tree Accuracy : "+str(decision\_acc)+"\n\n")  
  
def detect():  
    text.delete('1.0', END)  
    filename = filedialog.askopenfilename(initialdir="test")  
    test = []  
    with open(filename, "r") as file:  
        for line in file:  
            line = line.strip('\n')  
            line = line.strip()  
            test.append(line)  
    for i in range(len(test)):  
        tweet = bag\_of\_words(test[i])  
        result = classifier.classify(tweet)  
        prob\_result = classifier.prob\_classify(tweet)  
        negative = prob\_result.prob("neg")  
        positive = prob\_result.prob("pos")  
        msg = 'Neutral'  
        if positive > negative:  
            if positive >= 0.80:  
                msg = 'High Positive'  
            elif positive > 0.60 and positive < 0.80:  
                msg = 'Moderate Positive'  
            else:  
                msg = 'Neutral'  
        else:  
            if negative >= 0.80:  
                msg = 'High Negative'  
            elif positive > 0.60 and positive < 0.80:  
                msg = 'Moderate Negative'  
            else:  
                msg = 'Neutral'  
        text.insert(END,test[i]+" == tweet classified as "+msg+"\n")          
             
                 
  
def graph():  
    height = [svr\_acc,random\_acc,decision\_acc]  
    bars = ('SVR Accuracy', 'Random Forest Accuracy','Decision Tree Accuracy')  
    y\_pos = np.arange(len(bars))  
    plt.bar(y\_pos, height)  
    plt.xticks(y\_pos, bars)  
    plt.show()  
  
  
font = ('times', 16, 'bold')  
title = Label(main, text='Twitter Sentiment Analysis Based on Ordinal Regression')  
title.config(bg='LightGoldenrod1', fg='medium orchid')    
title.config(font=font)            
title.config(height=3, width=120)        
title.place(x=0,y=5)  
  
font1 = ('times', 12, 'bold')  
text=Text(main,height=20,width=150)  
scroll=Scrollbar(text)  
text.configure(yscrollcommand=scroll.set)  
text.place(x=350,y=100)  
text.config(font=font1)  
  
  
font1 = ('times', 14, 'bold')  
uploadButton = Button(main, text="Load NLTK Dataset", command=upload)  
uploadButton.place(x=50,y=100)  
uploadButton.config(font=font1)    
  
readButton = Button(main, text="Read NLTK Tweets Data", command=readNLTK)  
readButton.place(x=50,y=150)  
readButton.config(font=font1)  
  
svrButton = Button(main, text="Run SVR Algorithm", command=runSVR)  
svrButton.place(x=50,y=200)  
svrButton.config(font=font1)  
  
randomButton = Button(main, text="Run Random Forest Algorithm", command=runRandom)  
randomButton.place(x=50,y=250)  
randomButton.config(font=font1)  
  
decisionButton = Button(main, text="Run Decision Tree Algorithm", command=runDecision)  
decisionButton.place(x=50,y=300)  
decisionButton.config(font=font1)  
  
detectButton = Button(main, text="Detect Sentiment Type", command=detect)  
detectButton.place(x=50,y=350)  
detectButton.config(font=font1)  
  
graphButton = Button(main, text="Accuracy Graph", command=graph)  
graphButton.place(x=50,y=400)  
graphButton.config(font=font1)  
  
main.config(bg='OliveDrab2')  
main.mainloop()

## Screen Captures

* + 1. **User Login Screen:**

To run this project double click on ‘run.bat’ file to get below screen



In above screen click on ‘Load NLTK Dataset’ to load tweets dataset from NLTK library



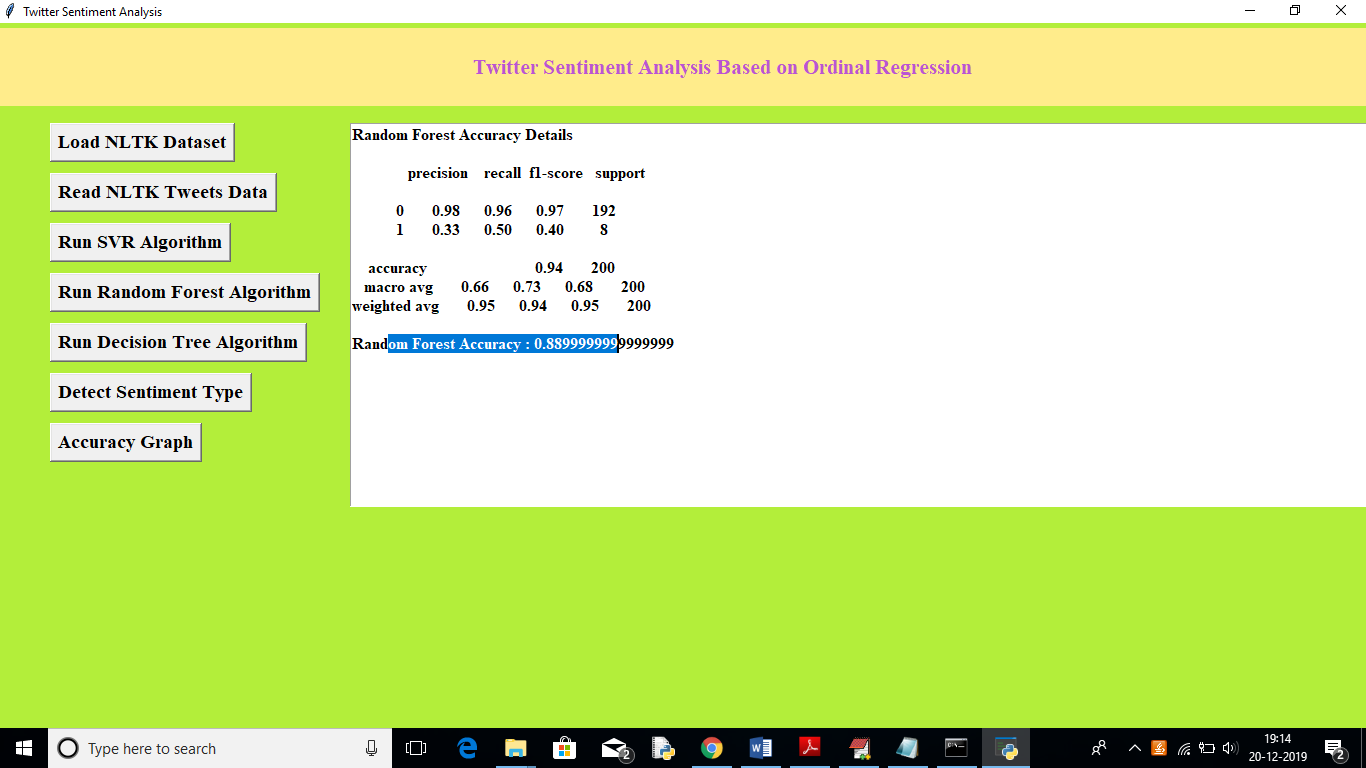
In above screen we can see total 10000 tweets are there in NLTK library, now click on ‘Read NLTK Tweets Data’ button to read all tweets and to build TFIDF vector. Upon each button click you need to wait for some seconds to get output. See below screen



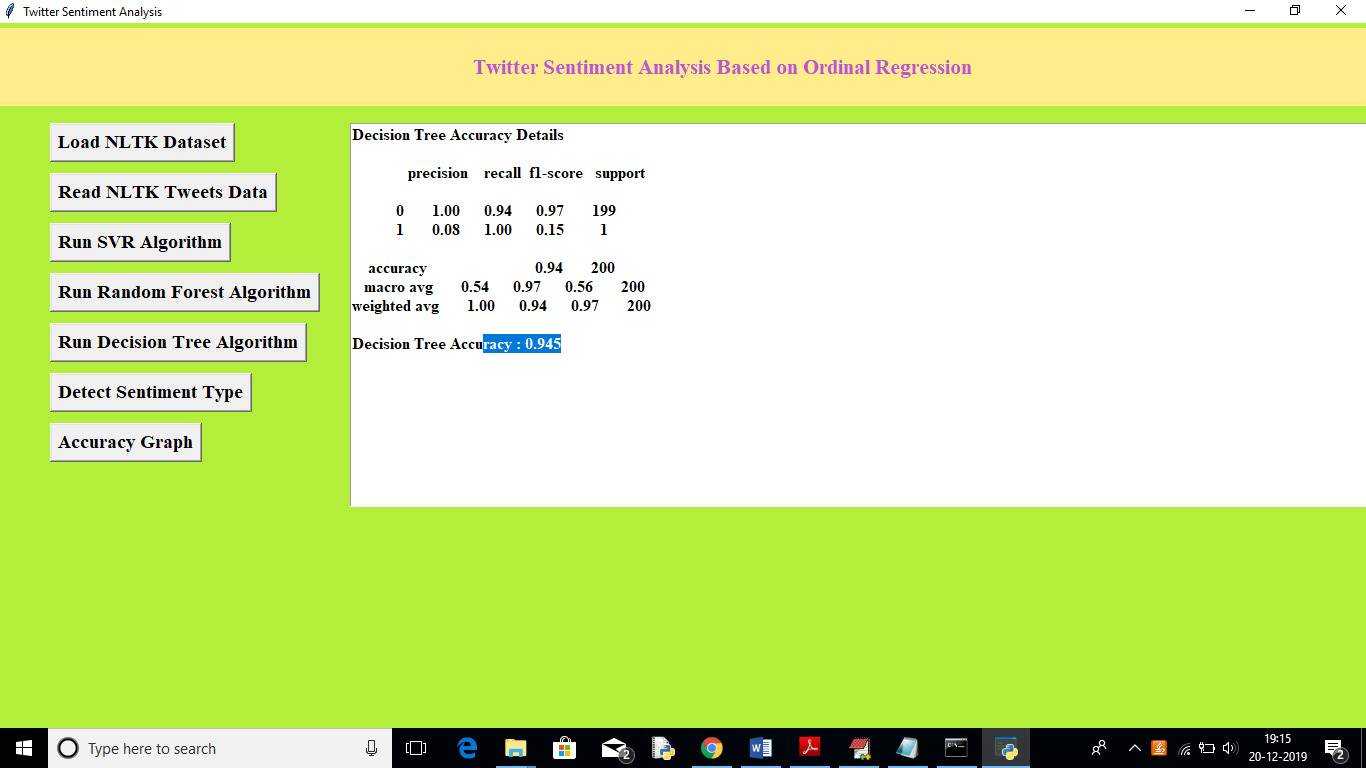
In above screen we can see total 8000 tweets vector used for training purpose and 2000 tweets used for testing purpose. Now click on ‘Run SVR Algorithm’ to build train model on that dataset and to calculate accuracy



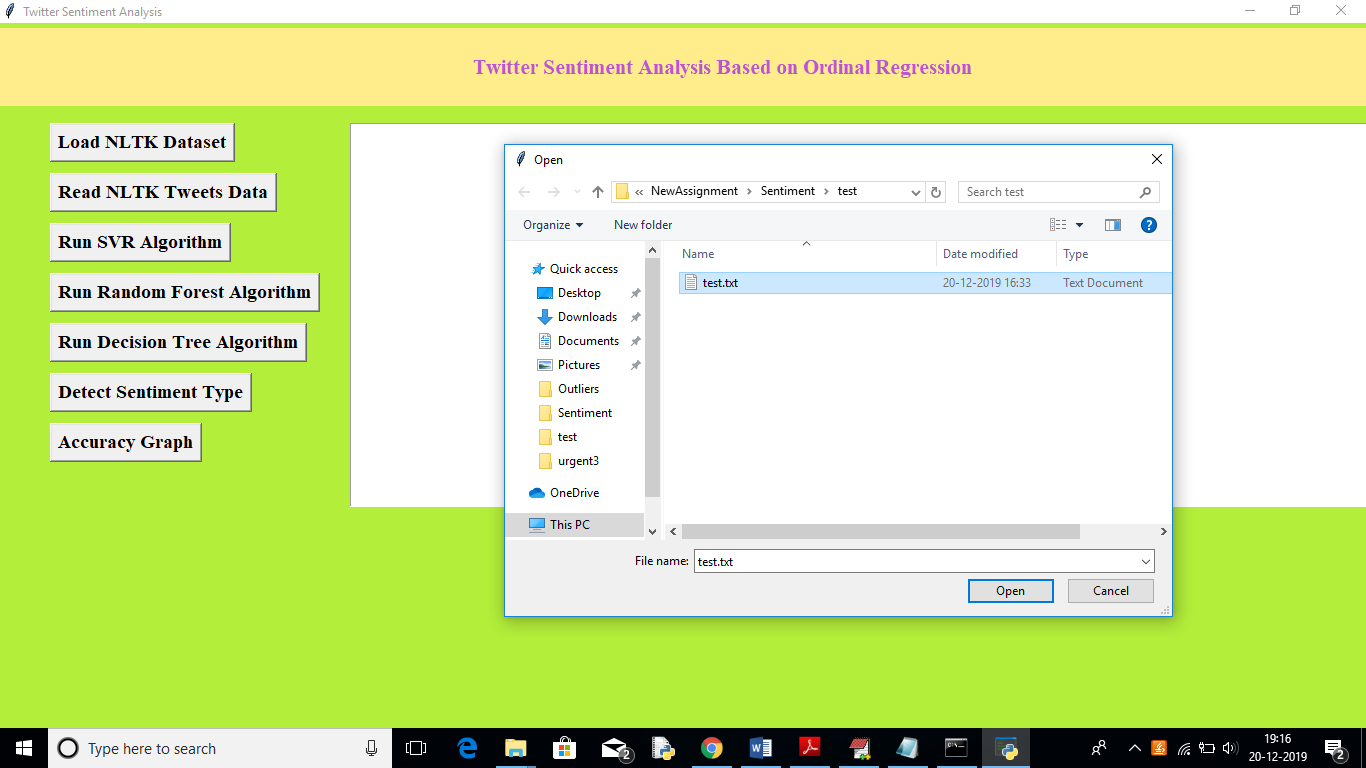
In above screen we can see SVR generate 0.71% prediction accuracy, now click on ‘Run Random Forest Algorithm’ button to calculate its accuracy



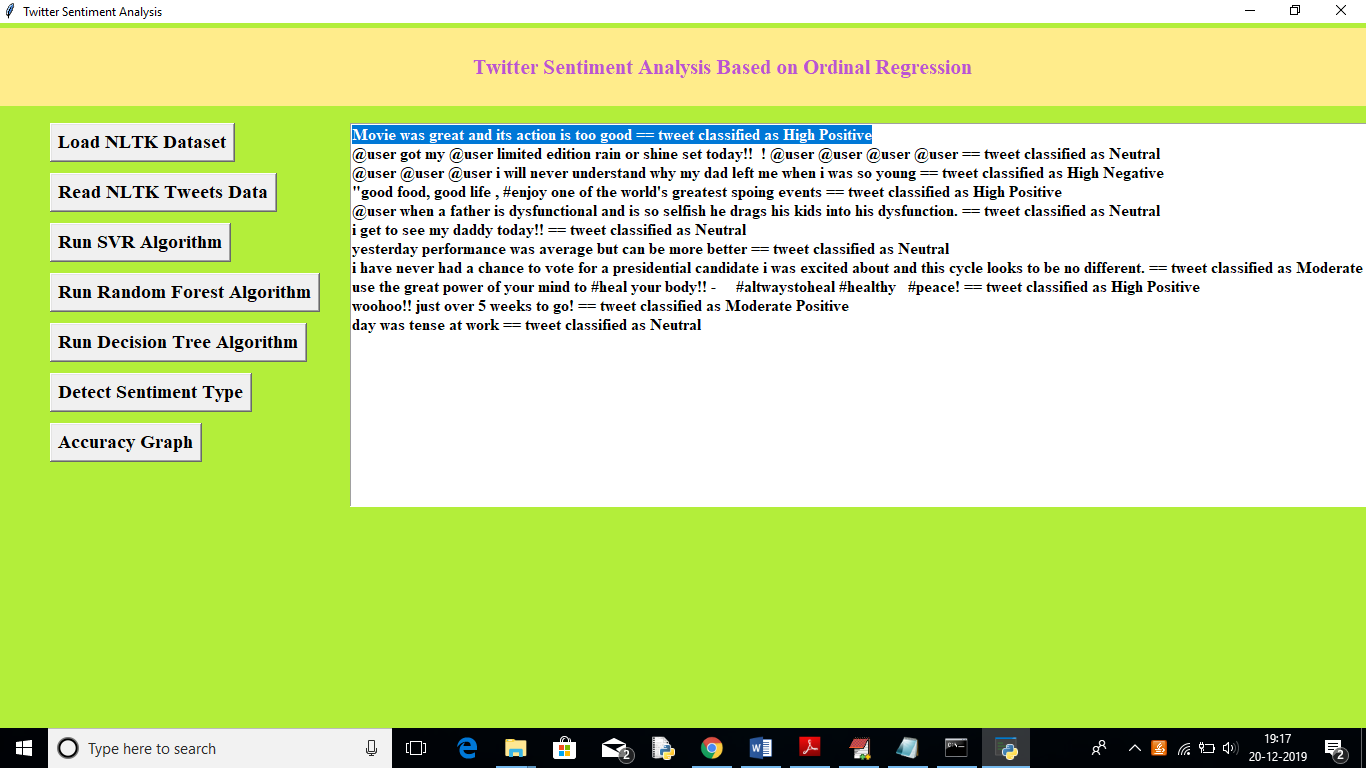
In above screen Random Forest got 0.88% accuracy, now click on ‘Run Decision Tree Algorithm’ button to calculate its accuracy



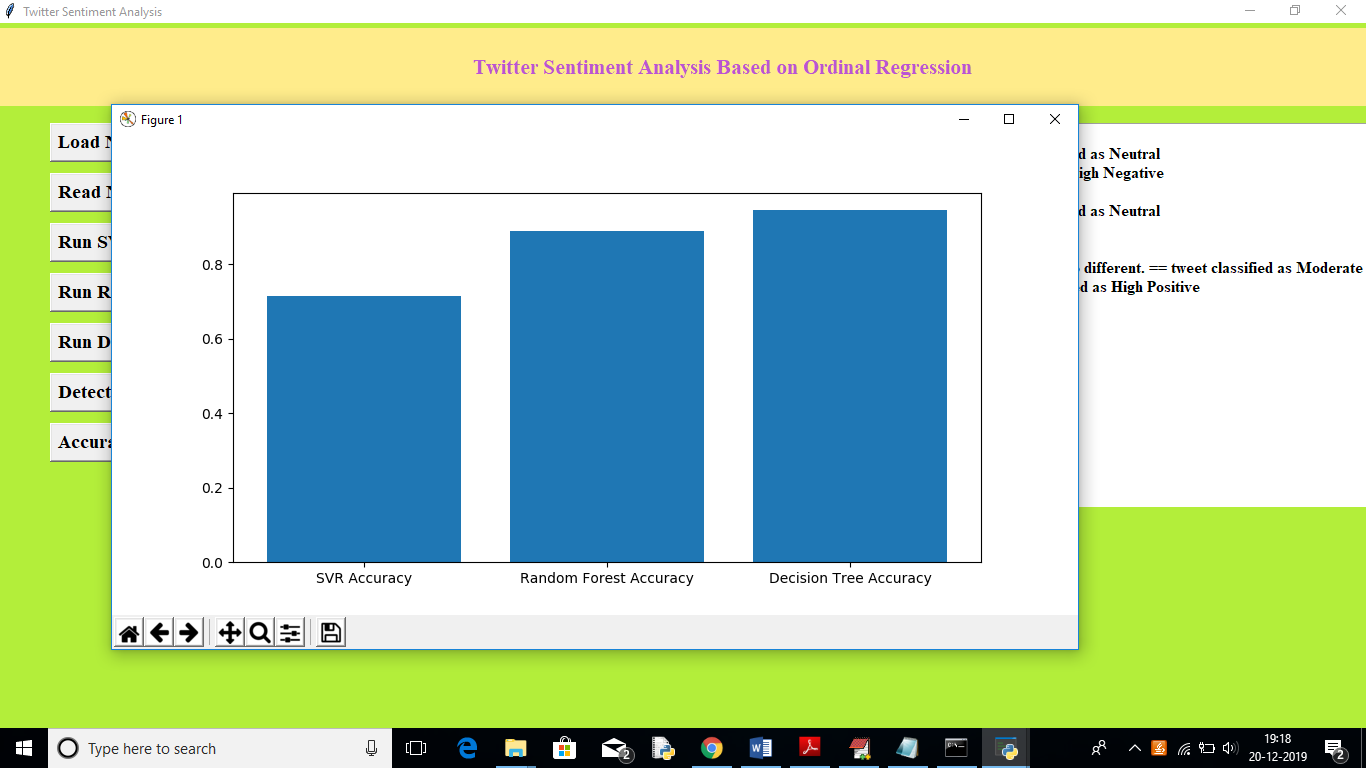
In above screen Decision Tree got 0.94% accuracy, Now click on ‘Detect Sentiment Type’ button and upload test tweets to predict it sentiment. In test folder inside test.txt you can see there is no sentiment label and application will detect it.



In above screen uploading test tweets file and below are the prediction results



In above screen for each tweet we can see the classified/predicted sentiments. Now click on ‘Accuracy Button’ to get below accuracy graph



In above graph x-axis represents algorithm name and y-axis represents accuracy, from above graph we can see decision tree got better prediction compare to other algorithm.

In this paper author using another algorithm called SOFTMAX but its not a classifier algorithm, so I am not implementing it

***CHAPTER - 6***

***TESTING***

*The chapter shows the various test cases.*

# CHAPTER 6

## Software Testing

Software testing is the process of validating and verifying that a software applicationmeets the technical requirements which are involved in its design and development. It is alsoused to uncover any defects/bugs that exist in the application. It assures the quality of thesoftware. There are many types of testing software viz., manual testing, unit testing, black box testing, performance testing, stress testing, regression testing, white box testing etc. Among theseperformance testing and load testing are the most important one for an android application and nextsections deal with some of these types.

## Black box Testing

Black box testing treats the software as a "black box"—without any knowledge of internal implementation. Black box testing methods include equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, traceability matrix, exploratory testing,and specification-based testing.

## White box Testing

White box testing is when the tester has access to the internal data structures and algorithms including the code that implement these.

## Performance Testing

Performance testing is executed to determine how fast a system or sub-system performsunder a particular workload. It can also serve to validate and verify other quality attributes of thesystem such as scalability, reliability and resource usage.

## Load Testing

Load testing is primarily concerned with testing that can continue to operate underspecific load, whether that is large quantities of data or a large number of users.

## Manual Testing

Manual Testing is the process of manually testing software for defects. Functionality of this application is manually tested to ensure the correctness. Few examples of test case for Manual Testing are discussed later in this chapter.

***CHAPTER - 7***

***RESULTS &CHALLENGES***

*The chapter describes the results and challenges faced in the project.*

# CHAPTER 7

**RESULTS & CHALLENGES**

## Results

This study aims to explain sentiment analysis of twitter data regarding ordinal regression using several machine learning techniques. In the context of this work, we present an approach that aims to extract Twitter sentiment analysis by building a balancing and scoring model, afterward, classifying tweets into several ordinal classes using machine learning classifiers. Classifiers, such as Multinomial logistic regression, Support vector regression, Decision Trees, and Random Forest, are used in this study. This approach is optimized using Twitter data set that is publicly available in the NLTK corpora resources. Experimental results indicate that Support Vector Regression and Random Forest have an almost similar accuracy, which is better than that of the Multinomial logistic regression classifier. However, the Decision Tree gives the highest accuracy at 91.81%..

## Challenges

* + - Understanding the connections of different modules is some what difficult and more numbers of modules are required to implement nltk
    - Implementing synchronization with Firebasewas a challenging task.
    - Learning different technologies and frameworks with little guidance.

***CHAPTER - 8***

***CONCLUSIONS & FUTURE WORK***

*The chapter gives brief conclusion about the project.*

# CHAPTER 8

# CONCLUSION

## 8.1Conclusion

## This study aims to explain sentiment analysis of twitter data regarding ordinal regression using several machine learning techniques. In the context of this work, we present an approach that aims to extract Twitter sentiment analysis by building a balancing and scoring model, afterward, classifying tweets into several ordinal classes using machine learning classifiers. Classifiers, such as Multinomial logistic regression, Support vector regression, Decision Trees, and Random Forest, are used in this study. This approach is optimized using Twitter data set that is publicly available in the NLTK corpora resources. Experimental results indicate that Support Vector Regression and Random Forest have an almost similar accuracy, which is better than that of the Multinomial logistic regression classifier. However, the Decision Tree gives the highest accuracy at 91.81%..

.

## 8.2Scope for future work

## In the future, we plan to improve our approach by attempting to use bigrams and trigrams. Furthermore, we intend to investigate different machine learning techniques and deep learning techniques, such as Deep Neural Networks, Convolutional Neural Networks, and Recurrent Neural Networks.

**Limitations**:

* It requires huge data sets
* The random data set may take always a error because it may take all positive tweets once and all negative data set once(only in few cases it happens)

# BIBLIOGRAPHY

Code snippets for any errors <http://stackoverflow.com/>

Android Development Guide htt[ps://www](http://www.udemy.com/android).[udemy.com/android](http://www.udemy.com/android)

Xml and Layout Guide htt[ps://www](http://www.androidhive.com/).[androidhive.com/](http://www.androidhive.com/)

Software Testing <http://en.wikipedia.org/wiki/Software_testing>

Manual Testing <http://en.wikipedia.org/wiki/Manual_testing>

Performance Testing <http://en.wikipedia.org/wiki/Software_performance_testing>