Personalized News Recommendation System Using Twitter Tweets

PROJECT REPORT

Submitted by

SARANSH DABAS	15BCE0834
G LAKSHAYE VAIKUNTH	15BCE0069
PRAGADEESH DARSHA	15BCE0707
SATYAK BABAR	15BCE0150
VINEET NALAWADE	15BCE0284

Prepared For

Web Mining (CSE3024) – J_COMPONENT

Submitted To

Prof. BALAKRISHNAN P

School of Computer Science and Engineering

Table of Contents:

1	Introduction	Pg 3
2	Problem Statement	Pg 3
3	Literature Survey	Pg 4
4	Methodology	Pg 4
5	Project resource requirements	Pg 5
6	Components	Pg 5
7	Code	Pg 8
8	Challenges	Pg 26
9	Result and Screenshots	Pg 26
10.	Future Enhancements	Pg 28
11.	References	Pg 28

1. INTRODUCTION

Owing largely to the ever-increasing volume and sophistication of information on the web, we are able to access an enormous amount of information from around the globe. The key challenge today is for the users is to find relevant information based from an almost infinite source. This problem has led to the evolution of the recommender systems that proactively present users with information related to their interests. Online news reading has become a popular way to read news articles from a huge collection of news sources around the globe. News recommender systems help users manage this flood by suggesting articles based on user interests rather than presenting articles in order of their occurrence. We present our research on developing personalized news recommendation system with the help of a popular micro-blogging service, "Twitter." News articles are ranked based on the popularity of the article identified with the help of the tweets from Twitter's public timeline.

2. PROBLEM STATEMENT

Build a system that recommends news based on user's twitter activity i.e., given a user id, analyse the tweets and recommend the news that are best suited to his interests User's interests analysed through his tweets

3. LITERATURE SURVEY

Some research initiatives show that the exploitation of tweets allows for valuable applications such as early warning system [Sakaki et al. , 2010] or discovery of fresh Web sites [Dong et al. , 2010] . These applications mainly utilize the wisdom of the crowds as source of information rather than relying on individual tweets.

There are many different ways in which news are recommended using twitter to an individual like Recommendation Based on the log analysis, we developed a Bayesian framework for predicting users' current news interests from the activities of that particular user and the news trends demonstrated in the activity of all users. Also News articles are ranked based on the popularity of the article identified with the help of the tweets from Twitter's public timeline.

4. METHODOLOGY

Process divided into two parts:

1.Indexing the news dataset

Named Entities are extracted from The Hindu news corpus and

are stored as inverted index

2.Querying the tweets

o User's recent 10 tweets fetched

o Ranking algorithms applied on the news results

o Top 10 news articles with the highest rank displayed

5. PROJECT RESOURCE REQUIREMENTS

- twython api
- Jsoup api
- Eclipse IDE
- Python
- Java

6. COMPONENTS

1. Stemming

Stemmer implements the Porter Stemming Algorithm. The Stemmer class transforms a word into its root form.

The input word can be provided a character at time (by calling add()), or at once by calling one of the various stem(something) methods.

It adds a character to the word being stemmed. When you are finished adding characters, you can call stem(void) to stem the word.

It also adds when characters to the word being stemmed contained in a portion of a char[] array. This is like repeated calls of add(char ch), but faster.

After a word has been stemmed, it can be retrieved by toString(),or a reference to the internal buffer can be retrieved by get ResultBuffer and getResultLength (which is generally more efficient.

Finally it stems the word placed into the Stemmer buffer through calls to add().Returns true if the stemming process resulted in a word different from the input. You can retrieve the result with getResultLength() /getResultBuffer() or toString().

2. <u>Stop words Removal</u>

It eliminates the stop words from the query tweets. The stop words array contains:

String[] stopList = {"a","able","about","across","after","all","almost","also",
"among","an","and","any","are","as","at","be","because","been",
"but","by","can","cannot","could","dear","did","do","does",
"either","else","ever","every","for","from","get","got","had",
"has","have","he","her","hers","him","his","how","however","i",
"if","in","into","is","it","its","just","least","let","like",
"likely","may","me","might","most","must","my","neither","no",
"nor","not","of","off","often","on","only","or","other","our",
"own","rather","said","say","says","she","should","since","so",
"some","than","that","the","their","them","then","there",
"these","they","this","tis","to","too","twas","us","wants",
"was","we","were","what","when","where","which","while","who",
"whom","why","will","with","would","yet","you","your","am","infobox","lt","gt","day","
monday" } ;

3. <u>UrlFile</u>

This file contains all the URL of the articles from which our search is conducted.

Eg:

- http://www.thehindu.com/news/national/congress-not-interested-in-bringing-back-black-money-modi/article5854340.ece
- http://www.thehindu.com/news/national/sonia-accuses-the-opposition-of-dividing-the-nation/article5851336.ece
- http://www.thehindu.com/news/national/bjps-divisive-politics-wont-bring-development-says-manmohan/article5848337.ece

- http://www.thehindu.com/news/national/congress-fighting-for-secularism-sonia/article5854591.ece

4. Parser

The parser is a method which will be passed the doc at the end of the parsing. Useful if TableParser is within an inner loop and you want to automatically process the document. If it is omitted then it will do nothing.

5. Indexing

Index creation takes into account the following categories

1. Named entities in the article
2. Unigrams of title and topic
3. Unigrams of article

6. Querying

Query part basically contains two broad subcategories:

Retrieval

o Retrieve the documents that contain the tweet keywords

Ranking

o Rank the retrieved documents so that the most relevants documents are displayed as the result

7. Ranking Formula

Weight[i] = N/ sqrt(td[i])

- where td[i] = # of documents that containing the tweet keywords and i can be 1,2,3
- doc[j] + = weight[i]

Each document weight is assigned by the above formula

7. CODE

Main.py:

```
#/usr/bin/python


#/usr/bin/python
from bs4 import BeautifulSoup
from table_parser import *
import urllib, urllib2, json
import string
import sys

global_min = 100000000000
global_list = []
global_edit_min = 100000000000
# initialize this for every new search

def substCost(x,y):
        if x == y:
                return 0
        else:
                return 2

def minEditDistR(word1, word2):
        len_1=len(word1)
        len_2=len(word2)
        x =[[0]*(len_2+1) for _ in range(len_1+1)]
        for i in range(0,len_1+1):
                x[i][0]=i
        for j in range(0,len_2+1):
                x[0][j]=j
        for i in range (1,len_1+1):
                for j in range(1,len_2+1):
                        if word1[i-1]==word2[j-1]:
                                x[i][j] = x[i-1][j-1]
```

```
                else :
                    x[i][j]= min(x[i][j-1],x[i-1][j],x[i-1][j-1])+1


        return x[i][j]


def extract_data(lst,query):
        global global_min
        global global_list
        global global_edit_min
        l = len(query)
        dic = {}
        for i in range(l):
                dic[query[i]]=0
        dis = 0


        newlst=[]
        for x in lst:
                if(len(re.compile(r'[\s\w-]+').findall(x))>0):
                        newlst.append(re.compile(r'[\s\w-]+').findall(x)[0])
        lst = [x for x in newlst if len(x)>0]
        lst = [x for x in lst if not str.isdigit(x)]
        for words in lst:
                if(words == ""):
                        continue
                flag = 0
                for needle in query:
                        for needles in needle.split(' '):
                                if(len(needles)<=1):
                                        continue;
                                if(needles[-1] == 's'):
                                        needles = needles[0:-1]
                                try:
                                        p = re.search(needles.lower(),words.lower())
                                except:
                                        p = False
```

```
                                if p:
                                        dis += minEditDistR(needle.lower(),words.lower())
                                        dic[needle]=1
                                        flag = 1
                                        break
                        if(flag == 1):
                                break
        for key in dic.keys():
                if dic[key] != 1:
                        return
        if dis < global_edit_min and len(lst)>3:
                global_edit_min = dis
                global_list = lst
        elif dis == global_edit_min:
                if len(lst) < global_min and len(lst)>3:
                        global_list = lst
                        global_min = len(lst)
        return


def search_google(query):
#       proxy = urllib2.ProxyHandler({'http': 'http://proxy.iiit.ac.in:8080/'})
#       opener = urllib2.build_opener(proxy)
#       urllib2.install_opener(opener)
        url="http://www.google.co.in/search?q="
        for token in query.split():
                url+="+";
                url+=token;
        user_agent = 'Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.7) Gecko/2009021910 Firefox/3.0.7'
        headers={'User-Agent':user_agent,}
        req = urllib2.Request(url,None,headers)
        response = urllib2.urlopen(req)
        html = response.read()
        parsed_html = BeautifulSoup(html,'html.parser')
        resultLinks=[]
        for link in parsed_html.body.findAll('h3'):
```

```
                resultLink = link.find('a')['href'].split("=")[1].split("&")[0]
                if "http://" in resultLink and "wikipedia" in resultLink:
                        resultLinks.append(resultLink)
        for link in parsed_html.body.findAll('h3'):
                resultLink = link.find('a')['href'].split("=")[1].split("&")[0]
                if "http://" in resultLink and "wikipedia" not in resultLink:
                        resultLinks.append(resultLink)
        return resultLinks #array containing resulting wikipeida links

def extract2(data,query):
        soup = BeautifulSoup(data,'html.parser')
        for table in soup.findAll('tr'):
                for each in table.find_all('td'):

                        p = each.get_text().encode('utf-8').strip('\n').split('\n')
                        extract_data(p,query)
                        for rows in each.find_all('ul'):
                                p =  rows.get_text().encode('utf-8').strip('\n').split('\n')
                                extract_data(p,query)

        for rows in soup.find_all('ul'):
                p =  rows.get_text().encode('utf-8').strip('\n').split('\n')
                extract_data(p,query)
        lis = []
        rows = soup.find_all('span',attrs={'class':"toctext"})
        for each in rows:
                lis.append(each.get_text().encode('utf-8'))
        extract_data(lis,query)

        return


def main():
#       print "HIII",
#       f = "".join(sys.argv[1:])
        # f1 = open("input.txt","r").read().split('\n')
```

```
        f1 = []
        f1.append(sys.argv[1]);
#       f2 = open("output.txt,"w")
#       f1 = []
#       f1.append(f)
#       sys.stdout.flush()
#       return
        for each in f1:
                if len(each)==0:
                        continue;

                query = [x.lower().strip(' ') for x in each.split(',')]



                quer = "list "
                for el in query:
                        quer +=el.lower()
                        quer+=" "
                quer += " wikipedia"
                urls = search_google(quer)
                i=0;
                while(True):
                        global global_min
                        global global_list
                        global global_edit_min
#print url
                        try:
                                req = urllib2.Request(urls[i])
                                response = urllib2.urlopen(req)
                                data = response.read()
                        except:
                                i+=1
                                if(len(global_list)>0 or len(urls)<=i):
                                        for each in global_list:
                                                f2.write(each)
                                        print global_list,
```

```
                                        break
                                else:
                                        continue


                        global_min =  100000000000
                        global_list = []
                        global_edit_min = 100000000000
                        extract2(data,query)
                        i+=1
                        if(len(global_list)>0 or len(urls)<=i):
#                               for each in global_list:
#                                       f2.write(each)
                                # print global_list,
                                for item in global_list:
                                        print item
                                break
if __name__=='__main__':
main()
```

GUI.java :

```
package SearchPart;

import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Cursor;
import java.awt.Desktop;
import java.awt.Dimension;
import java.awt.FlowLayout;
import java.awt.Graphics;
import java.awt.GridBagConstraints;
import java.awt.Image;
import java.awt.ScrollPane;
import java.awt.Toolkit;
import java.awt.event.ActionEvent;
```

```
import java.awt.event.ActionListener;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.io.IOException;
import java.net.URI;
import java.net.URISyntaxException;

import javax.swing.BorderFactory;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;
import javax.swing.border.Border;
import javax.swing.border.EtchedBorder;
import javax.swing.border.TitledBorder;


public class GUI extends JFrame{

	//Screen Dimensions
	private double width;
	private double height;

	private JFrame myFrame;
	private JPanel topPanel;
	private JPanel tweetPanel;
	private JPanel newsPanel;
	private ScrollPane upperCenterPanel;
	private ScrollPane lowerCenterPanel;
	//private JPanel bottompanel;

	private JLabel  bgTopPanel;
```

```
private JLabel label;
private JTextField textBox;
private JLabel []tweetLabel;
private JLabel []newsLabel;
//private JLabel msgLabel;
private JButton submitButton;
private MyHandler mHandler;




public void createFrame(){
        Border lowerEtched = BorderFactory.createEtchedBorder(EtchedBorder.LOWERED);
        TitledBorder title = BorderFactory.createTitledBorder(lowerEtched, "Recent Tweets");
        TitledBorder title1 = BorderFactory.createTitledBorder(lowerEtched, "Recommended News");
        //Border lineBorder = BorderFactory.createLineBorder(Color.black);
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        width = (int) screenSize.getWidth();
        height = (int) screenSize.getHeight();

        int calHeight,calWidth,temp;
        mHandler  = new MyHandler();

        myFrame = new JFrame("Personalized News Recommendation Based On Twitter Activity");
        myFrame.setLayout(null);
        myFrame.setSize((int)width,(int)height);
        myFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        myFrame.setVisible(true);

        //GridBagConstraints gbc = new GridBagConstraints();

        topPanel = new JPanel();
        tweetPanel = new JPanel();
        newsPanel = new JPanel();
        upperCenterPanel = new ScrollPane();
        lowerCenterPanel = new ScrollPane();
```

```
tweetPanel.setBorder(title);
tweetPanel.setAutoscrolls(true);
newsPanel.setBorder(title1);
tweetPanel.setLayout(null);
newsPanel.setLayout(null);
//size


label=new JLabel("Twitter User Screen Name");
textBox = new JTextField(50);
tweetLabel = new JLabel[10];
bgTopPanel = new JLabel();
newsLabel = new JLabel[10];
submitButton = new JButton("submit");

ImageIcon ic = new ImageIcon("/home/kapil/twitter.png");
bgTopPanel.setSize((int)width,(int)(0.06*height));
bgTopPanel.setLayout(new FlowLayout());
bgTopPanel.setIcon(ic);
bgTopPanel.add(label);
bgTopPanel.add(textBox);
bgTopPanel.add(submitButton);
topPanel.add(bgTopPanel);

for(int i=0;i<10;i++){
        tweetLabel[i] = new JLabel("");
        tweetLabel[i].setSize((int) width , 20);
        tweetLabel[i].setLocation(10,i*30+20);

        newsLabel[i] = new JLabel("");
        newsLabel[i].setSize((int) width , 20);
        newsLabel[i].setLocation(10,i*30+20);

        tweetPanel.add(tweetLabel[i]);
        newsPanel.add(newsLabel[i]);
}
```

```
        /*
         * Location Setting
         */

        calHeight = (int)(height * 0.06) ;
        calWidth = (int)width;
        temp=calHeight;
        topPanel.setBackground(Color.GRAY);
        topPanel.setSize(calWidth, calHeight);
        topPanel.setLocation(0, 0);


        calHeight = (int)(height * 0.47) ;
        upperCenterPanel.setBackground(Color.BLACK);
        upperCenterPanel.setSize(calWidth, calHeight);
        upperCenterPanel.setLocation(0,temp);




        lowerCenterPanel.setBackground(Color.LIGHT_GRAY);
        lowerCenterPanel.setSize(calWidth, calHeight);
        lowerCenterPanel.setLocation(0,temp+calHeight);
        upperCenterPanel.add(tweetPanel);
        lowerCenterPanel.add(newsPanel);
        myFrame.add(topPanel);
        myFrame.add(upperCenterPanel);
        myFrame.add(lowerCenterPanel);



        submitButton.addActionListener(mHandler);
}
```

```
private class MyHandler implements ActionListener{

        private TwitterAnalysis ut;
        public void actionPerformed(ActionEvent event){

            if(event.getSource()==submitButton){
                String userId=textBox.getText();
                if(!userId.equals("")){
                    textBox.setText("");
                    ut = new TwitterAnalysis();
                    ut.analyze(userId);


                    int i,num=UserTopTenTweets.tw.size();
                    for(i=0;i<10;i++){
                        String temp="";
                        if(i<num)
                            temp=UserTopTenTweets.tw.poll();
                        tweetLabel[i].setText(temp);

                    }
                    i=0;
                    for(String temp : TwitterAnalysis.newsData){
                        newsLabel[i].setText(temp);
                        final String t=temp;
                        newsLabel[i].setCursor(new Cursor(Cursor.HAND_CURSOR));

                        MouseListener[] arrMouseListeners =
newsLabel[i].getMouseListeners();

                        for ( MouseListener m : arrMouseListeners ) {
                            newsLabel[i].removeMouseListener(m);
                        }


                    newsLabel[i].addMouseListener(new MouseAdapter() {
```

```
                                        @Override
                                        public void mouseClicked(MouseEvent e) {
                                            try {
                                                Desktop.getDesktop().browse(new URI(t));
                                            } catch (URISyntaxException | IOException ex) {
                                                //It looks like there's a problem
                                            }
                                        }
                                    });
                                        i++;
                                    }

                                }
                                else{
                                        JOptionPane.showMessageDialog(null,"Text Box is empty");
                                }
                            }
                        }
                        /*
                        public MyHandler() {
                            //
                            // TODO Auto-generated constructor stub
                        }
                        */


    }

    public static void main(String[] args){
        GUI gui=new GUI();
        gui.createFrame();
    }

}
```

TwitterAnalysis.java :

```
package SearchPart;

import java.io.BufferedReader;
import java.io.File;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.LinkedList;
import java.util.Queue;

public class TwitterAnalysis {
	public static Queue<String> newsData = new LinkedList<String>();
	/*
	public static void showTweet(){
		for(String temp : UserTopTenTweets.tw){
			System.out.println(temp);
		}
	}
	*/
	public void analyze(String userName){

		newsData.clear();
		UserTopTenTweets user = new UserTopTenTweets();
		try {
			user.analyze(userName);
			//System.out.println("-----------------Tweets----------------------");
			//showTweet();
			//System.out.println("-------------------------------------------");

			File file =new File ("src/SearchPart/script.sh");
			String ScriptPath = file.getAbsolutePath();
			//System.out.println(ScriptPath);
			String[] cmd = new String[3];
			cmd[0] = "bash";
			cmd[1] = ScriptPath;
```

```
                        cmd[2] = userName;



                        // create runtime to execute external command
                        Runtime rt = Runtime.getRuntime();
                        Process pr = rt.exec(cmd);

                        // retrieve output from script
                        BufferedReader bfr = new BufferedReader(new InputStreamReader(pr.getInputStream(),
"UTF-8"));

                        String line = "";
                        while((line = bfr.readLine()) != null) {
                        // display each output line form  script
                        newsData.add(line);
                        }
                } catch (IOException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

        }
        /*
         public static void main(String[] args){
                TwitterAnalysis twAnalysis = new TwitterAnalysis();
                twAnalysis.analyze("kapil_chhajer");
        }
         */
}
```

UserTopTenTweets.java :

```
package SearchPart;

import java.io.BufferedReader;
```

```
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.util.LinkedList;
import java.util.Queue;


public class UserTopTenTweets {
	public static Queue<String> tw = new LinkedList<String>();
	public  void analyze(String args) throws IOException {

		tw.clear();
		String username=args;

		File file =new File ("src/SearchPart/twitter.py");
		String pythonScriptPath = file.getAbsolutePath();

		//StopWord stp = new StopWord();
		String[] cmd = new String[3];
		cmd[0] = "python2.7";
		cmd[1] = pythonScriptPath;
		cmd[2] = username;

		// create runtime to execute external command
		Runtime rt = Runtime.getRuntime();
		Process pr = rt.exec(cmd);

		// retrieve output from python script
		BufferedReader bfr = new BufferedReader(new InputStreamReader(pr.getInputStream(), "UTF-
8"));

		String readLine = "";
		while((readLine = bfr.readLine()) != null) {
		// display each output line form python script
```

```
System.out.println(readLine);
}

//
boolean tweetFlag=false,tweetStartFlag=true;
String fileName =username,temp="",line;
int count=0;
try{
	// extracting the 10 latest tweets
	Queue<String> queue = new LinkedList<String>();
	BufferedReader br = new BufferedReader(new FileReader(fileName));
	while ((line = br.readLine()) != null) {
		if(line.equals("<tweet>")){
			tweetFlag=false;
			tweetStartFlag=true;
			continue;
		}
		else if(line.equals("</tweet>")){
			if(temp!=""){
				queue.add(temp);
				temp="";
			}
			continue;
		}
		else if(count>9){
			break;
		}
		else{
				if(tweetStartFlag==true){
					tweetStartFlag=false;
					if(tweetFlag==false
&& !line.trim().toLowerCase().startsWith("rt")){
						tweetFlag=true;
						count++;
					}
				}
```

```
                        if(tweetFlag==true)
                                temp=temp+line+" ";

        }
}
br.close();

// writing last 10 tweets into a file
try {
        /*
         * Set expansion code is placed below in comment section
         */
        /*
        file =new File ("src/SearchPart/final.py");
        String pyScriptPath = file.getAbsolutePath();
        System.out.println(pyScriptPath);
        PrintWriter out = new PrintWriter(fileName);
        String[] c = new String[3];
        c[0] = "python2.7";
        c[1] = pyScriptPath;

        while (!queue.isEmpty()) {
                String l = queue.remove();

                String []split = l.split("\\s+");
                int len=split.length;
                for(int i=0;i<len;i++){
                        String term=split[i];
                        //System.out.println(term);
                        out.print("| "+term+"->");
                        if(stp.stopListFun(term)){
                                c[2]=term;
                                Runtime r = Runtime.getRuntime();
                                Process p = r.exec(c);
```

```
                                                // retrieve output from python script
                                                BufferedReader bfreader = new BufferedReader(new
InputStreamReader(p.getInputStream(), "UTF-8"));
                                                line = "";

                                                while((line = bfreader.readLine()) != null) {

                                                        out.print(" "+line);
                                                }
                                        }

                                }
                                out.println();
                }
                                out.close();
                                */

                                PrintWriter out = new PrintWriter("src/SearchPart/"+fileName);



                                while (!queue.isEmpty()) {
                                        String t=queue.remove();
                                        tw.add(t);
                                        out.println(t);
                }
                                out.close();
                                } catch (IOException e) {
                                        e.printStackTrace();
                                }
                        }
                catch (Exception e)

                        System.err.println(e.getMessage()); // handle exception
                }
        }
}
```

8. CHALLENGES

- Tweets usually contain redundant information

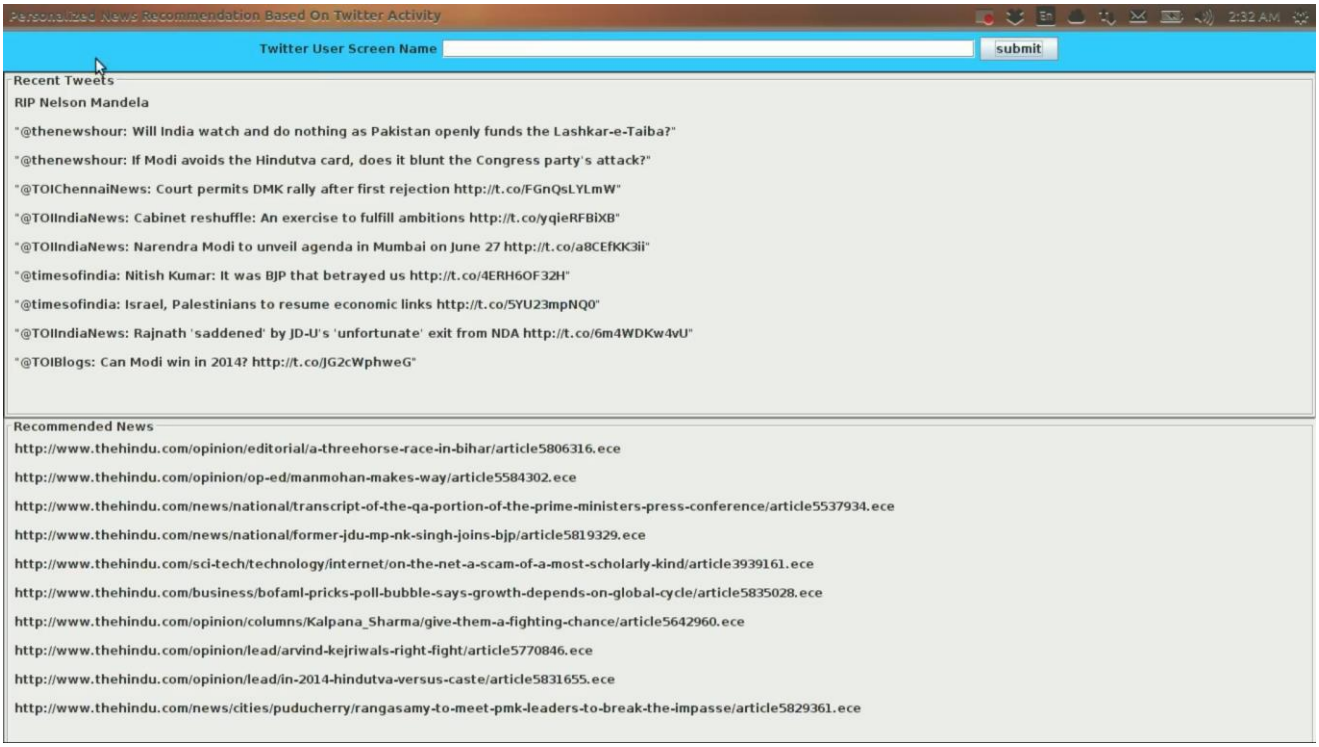
  e.g. hey what's up!!
- Knowing user interest from short tweets of 160 characters
- Handling grammatical mistakes
- If it Contains elongated text

  e.g. grrrrrreat!!

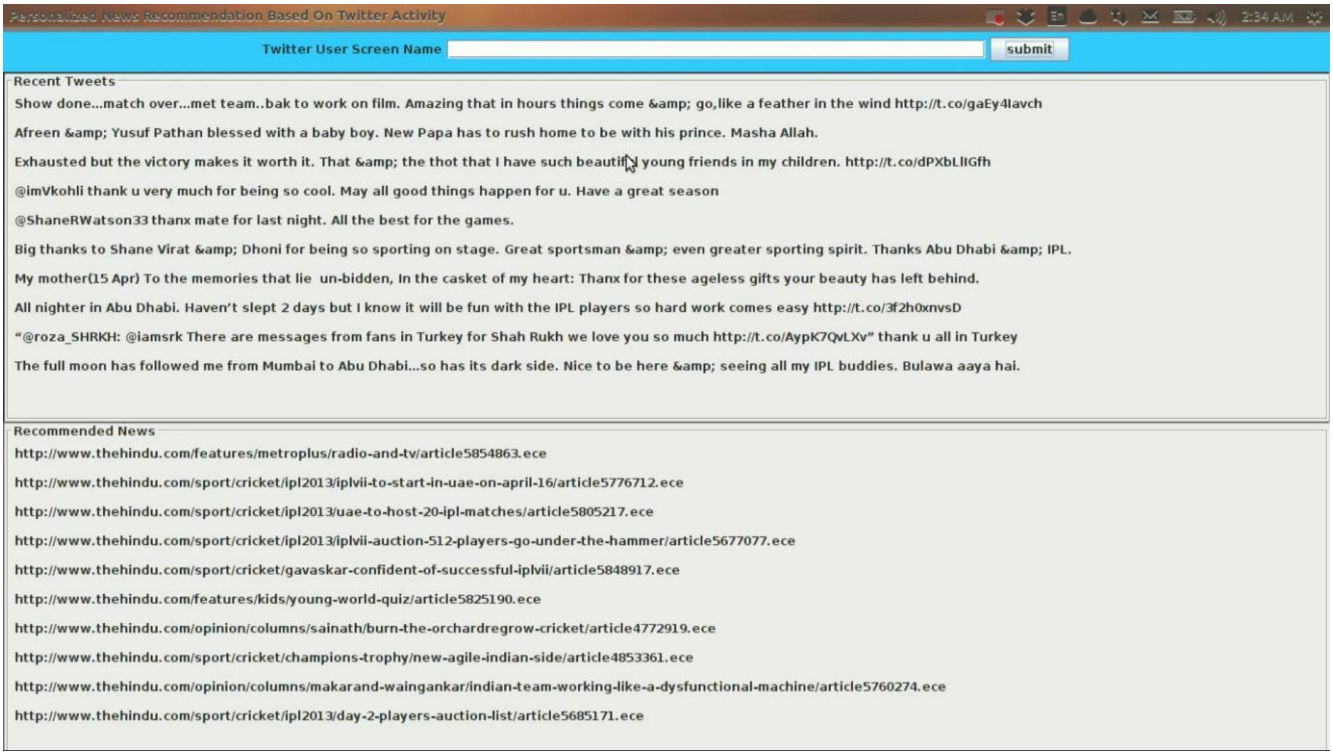
9. RESULTS AND SCREENSHOTS

We conclude that we have successfully implemented a system that recommends news based on user's twitter activity i.e., given a user id, analyse the tweets and recommend the news that are best suited to his interests

Recommended news results for twitter id: ARNABGOSWAMI



Recommended news results for twitter id: iamsrk



10. FUTURE ENHANCEMENTS

1.A feature of sentiment analysis to the news can be added as a future enhancement. This will give the user a clear idea about the type of news and he can judge if he wants to view the news article or not.

2.The positives and negatives of the news article can also be shown through sentiment analysis so that the user can get a summary of the news article.

11. REFERENCES

- http://wis.ewi.tudelft.nl/umap2011/um-twitter-umap-2011.pdf
  Laniado, D., Mika, P.: Making sense of Twitter. In Patel-Schneider et al. (eds.): Proc. of 9th Int. Semantic Web Conference (ISWC), Shanghai, China. Volume 6496 of LNCS. Springer (2010) 470485.

- http://ieeexplore.ieee.org/document/6690687/

- https://www.researchgate.net/publication/261199217_Personalized_News_Recommendation_Using_Twitter

- https://www.researchgate.net/profile/Susan_Gauch/publication/261199217_Personalized_News_Recommendation_Using_Twitter/links/5676d61a08aebcdda0e95590/Personalized-News-Recommendation-Using-Twitter.pdf

- https://dl.acm.org/citation.cfm?id=2569235&dl=ACM&coll=DL&CFID=1003920111&CFTOKEN=16544594