Information Retrieval

NTOULOS PANAGIOTIS

Abstract: Analysis of tweets from Political Parties of Greece. We tried to Categorize their tweets and make a new Dictionary of positive and negative words. Via SVD.

1)

tweets:

We Insert the tweets into the database , with date format (yyy-MM-dd)

2)

Abstract implementation:

Σκεφτόμαστε ότι έχουμε μια λίστα μεγέθους ίσου με το σύνολο των ημερών. Στο data-set μας είναι 11. Και οι τιμές τις λίστας είναι άλλες λίστες που περιέχουν τα tweets .

We got a List with the dates that we want to make the analysis and each item of the list got a list with the tweets of the day that it represents.

Code: `ArrayList<ArrayList<mini_tweet>> data;`

3)

Distribution by day:

We sort the select query by date.

```
public void SortTweetsByDays(ArrayList<mini_tweet> list ){

                ...
                ....}
```

Removing retweets RT.

```
if(list.get(i).gettext().equalsIgnoreCase(list.get(j).gettext())){
                                    list.remove(j);
                            }
```

The idea of our implementation is that we keep into a temp variable the date of each tweet, and when this date changes we change our date index.

4)Μέθοδος Καθαρισμού κείμενου:

H `Cleantext()`is our main function that cleans and categorize the tweets

```
res=temp.replaceAll("@[A-Za-z]+", "");
                         res=res.replaceAll("[\\@\\\"\\'\\{\\}\\;\\?\\\\|\\\\
…\\>\\<\\]\\[\\)\\(\\_\\=\\&\\*\\€\\%\\#\\!\\-\\+\\.\\$\\^:,]","");
                         res=res.replaceAll("[htpps://[A-Za-z]]+", "");
                         res=res.replaceAll("[0-9]+", "");
                         res=res.trim();
                         res=stripAccents(res.toUpperCase());
                         tokens=removestopwords(res);
                         tokens=stem(tokens);
                         category=Category(positive(tokens),negative(tokens));
```

5) Statistics:

SD :
```
double positivemean=pos_per_day();
            for(int i=0;i<this.Positive_day.size();i++){
                   sd = sd+ Math.pow((positivemean-this.Positive_day.get(i)), 2);
            }
            sd=sd/this.Positives;
            sd=Math.sqrt(sd);
```

SD of the whole table is a Bernoulli variable.

με κωδικά:
```
public double SD_table(){
            return Math.sqrt(this.mean_pos()*(1-this.mean_pos()));
      }
```

6)

SVD:

To implement the svd we used jblas

term-doc.

We kept the terms with df>=2
```
public void term_df(){
            ArrayList<String> terms=new ArrayList<String>();

            //exw olous tous orous sto terms//
            for(int i=0;i<this.data.size();i++){
                  for(int j=0;j<this.data.get(i).size();j++){
                        List<String> myList = new
ArrayList<String>(Arrays.asList(this.data.get(i).get(j).gettext().split(" ")));
                        terms.addAll(myList);
                  }
            }
            //UNIQUE TIMES TERM.
            List<String> listDistinct =
terms.stream().distinct().collect(Collectors.toList());
```

```
            //Θα μετρήσουμε το df του καθε unique term//
            List<term_df> ListTerm=new ArrayList<term_df>();

            for (int i=0;i<listDistinct.size();i++){
                  term_df temp=new term_df(listDistinct.get(i));
                  ListTerm.add(temp);
            }
            /*
             * gia kathe tweet an to tweet periexei ton term au3hse tou to df.
             */
            for(int i=0;i<this.data.size();i++){
                  for(int j=0;j<this.data.get(i).size();j++){
                        for(int k=0;k<ListTerm.size();k++){

      if(this.data.get(i).get(j).gettext().contains(ListTerm.get(k).getterm())){
                                    ListTerm.get(k).increasedf();
                              }
                  }
            }
                  }
            List<term_df> term_df = new ArrayList<term_df>();
            for(int i =0;i<ListTerm.size();i++){
                  //an exei df megalutero iso tou 2 valto sthn lista
                  if(ListTerm.get(i).getdf()>=2){
                        term_df.add(ListTerm.get(i));
                  }
            }

            this.term_df=term_df;
      }


public void setTermDoc(){
            this.term_doc=new double[this.term_df.size()][this.n_tweets];
            int doc=0;

            double termidf=0;
            for(int term=0;term<this.term_df.size();term++){
                  doc=0;

      termidf=Math.log(this.n_tweets/this.term_df.get(term).getdf());//vriskw idf
                  this.idf.add(termidf);
                        for(int i=0;i<this.data.size();i++){
                              for(int j=0;j<this.data.get(i).size();j++){
                                    Pattern p=
Pattern.compile(this.term_df.get(term).getterm());//kanei pattern to term
                                    String input=
this.data.get(i).get(j).gettext();//eisodo to tweet.
                                    Matcher m =p.matcher(input);
                                    int counter=0;
                                    while(m.find()){
                                          ++counter;
                                    }//vriskei to tf(term,doc)


                                    this.term_doc[term][doc]=counter;//tf
                                    ++doc;
                                    }//days
                              }//doc
            }//term

      }//function
```

SVD :

```
this.USV=Singular.fullSVD(new DoubleMatrix(this.term_doc));
    Uk(3);//ftiaxnei ton U[m][3] K=3


public void Uk(int k){
double[][] U=this.USV[0].toArray2();
            double[][] _Uk=new double[U.length][k];
            for(int i=0;i<U.length;i++){
                  Arrays.sort(U[i]);//sortarei thn grammh//
            }
            //vriskei ta k prwta//
            for(int i=0;i<U.length;i++){
                  int col=0;
                  for(int j=U[0].length-1;j>U[0].length-k-1;j--){//to sortarisma
ginete se fthinousa seira ara pernoume apo to telos ews kai k prin to telos
                        _Uk[i][col]=U[i][j];
                        ++col;

                  }
            }
```

EUC NORM FOR EACH TERM:

```
            List<Double> ecdnorm=new ArrayList<Double>();
            //vriskei ta norm tou ka8e term//
            for(int i=0;i<_Uk.length;i++){
                  double norm=0;
                  for(int j=0;j<_Uk[i].length;j++){
                        norm=norm+Math.pow(_Uk[i][j], 2);
                  }
                  norm=Math.sqrt(norm);
                  ecdnorm.add(norm);
            }
            for(int i=0;i<_Uk.length;i++){
                  for(int j=0;j<_Uk[i].length;j++){
                        _Uk[i][j]=_Uk[i][j]/ecdnorm.get(i);
                                    }
            }

            this.Uk=_Uk;
      }
```

7)Similarity:

```
public void similarity(){
            DoubleMatrix _Uk=new DoubleMatrix(this.Uk);
            DoubleMatrix _UkT=_Uk.transpose();
            DoubleMatrix similarity=_Uk.mmul(_UkT);
            double[][] arraysim=similarity.toArray2();
            this.similarity=arraysim;
      }
```

8)find nearest p :

.
We find the p nearest terms without searching at the diagonial

```
public void findnearest(int p){
            List<value_index> term_similars = new ArrayList<value_index>();
            List<term> term_and_nearest = new ArrayList<term>();
            this.p=p;
            for(int i=0;i<this.similarity.length;i++){
                  for(int j=0;j<this.similarity[i].length;j++){
                        if(i!=j){
                              term_similars.add(new value_index(this.similarity[i]
[j],j));//value kai index
                        }

                        }
                  //otan topo8etei ola ta similarities prepei na ta sortarei vasei
tou value kai na kratisoume ta prwta p//
                  term_similars=term_similars
                        .stream()
                        .sorted((e1, e2) -> Double.compare(e2.getvalue(),
                                e1.getvalue())).collect(Collectors.toList());


                  //afou exoume tous orous se fthinousa seira kratame tous prwtous
p kai tous topothetoume sthn lista
                  term temp=new term(this.term_df.get(i).getterm());//h seira me
tous orous exei krath8ei.

                  for(int k=0;k<p;k++){
                        //pernei to index twn prwton k kai topothetei sthn lista
                        int index=term_similars.get(k).getindex();
                        temp.addList(this.term_df.get(index).getterm());
                        }
                  term_and_nearest.add(temp);
                        term_similars.clear();//adeiazoume gia kathe term.
                  }
            this.pnearest=term_and_nearest;
```

9)ExPos/ExNeg

```
public void ExPos(){
            List<String> expos=new ArrayList<String>();
            for(int i=0;i<this.pnearest.size();i++){
                  if(this.PositiveWords.contains(this.pnearest.get(i).getname())){
                        //iterate oloi thn lista me tous nearest tou term
                        for(int j=0;j<this.pnearest.get(i).getlist().size();j++){
                              expos.add(this.pnearest.get(i).getlist().get(j));
                        }
                  }
            }
            this.expos=expos;
```

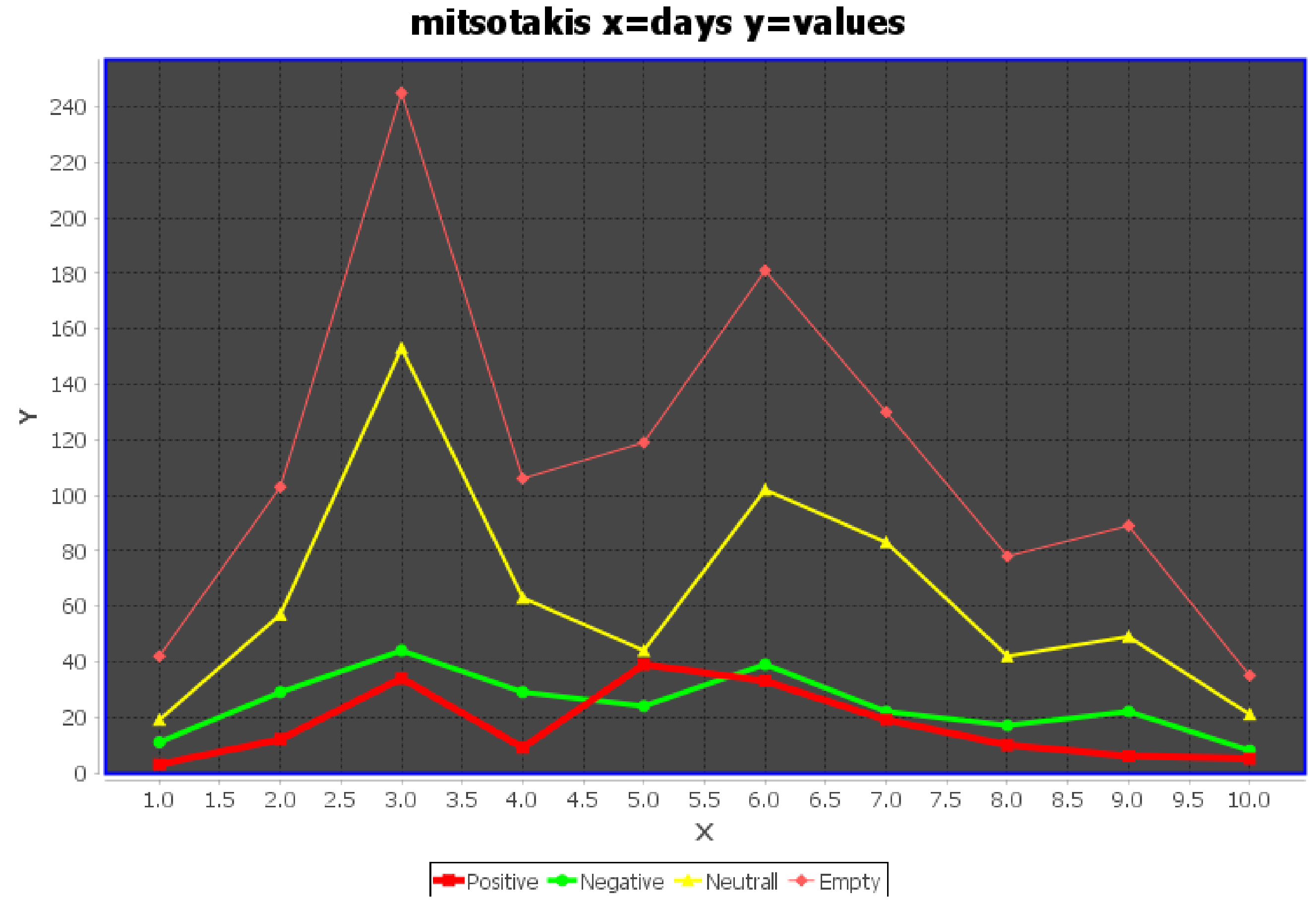
10) meanExpos()/meanExneg

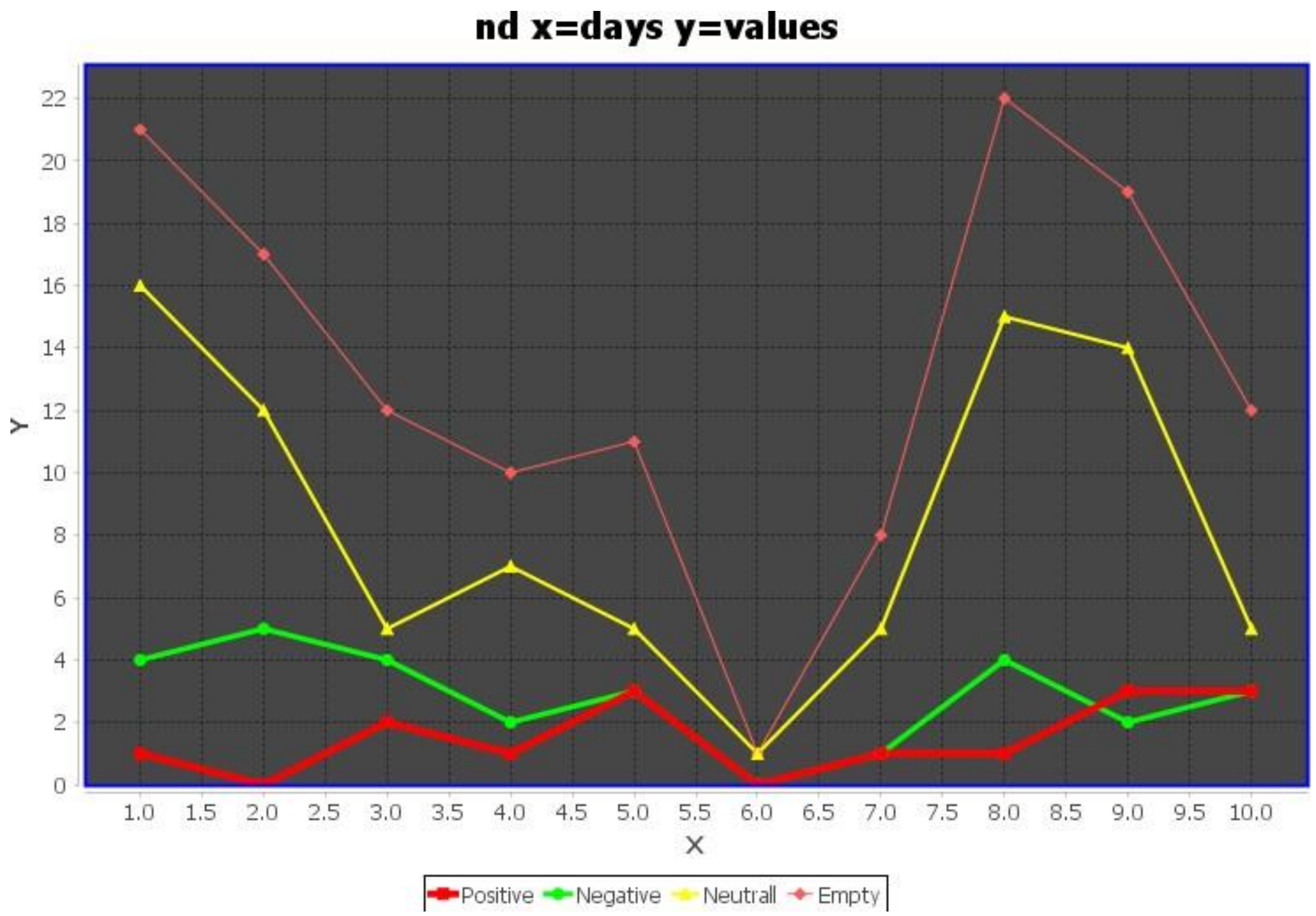
```
public double meanExpos(){

            double counter=0.0;
            for(int i=0;i<this.expos.size();i++){
                  if(this.PositiveWords.contains(this.expos.get(i))){
                        ++counter;
                  }
            }
            return(counter/this.expos.size());
      }
```

11)newPos/newNeg

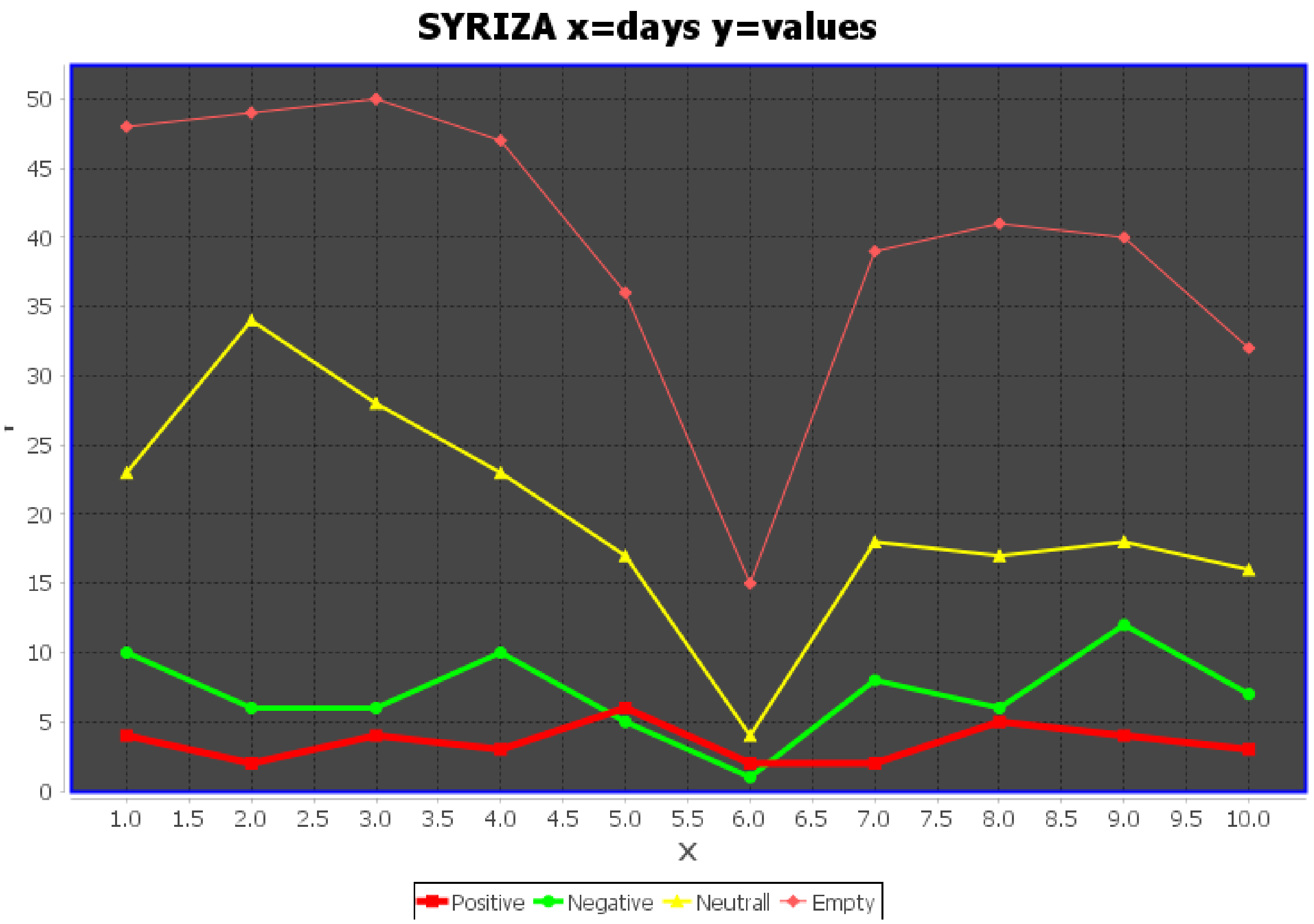
```
public void NewPos(){
                  List<String> newpos=new ArrayList<String>();
                  for(int i=0;i<this.expos.size();i++){
                        if(!(this.PositiveWords.contains(this.expos.get(i)))){
                              newpos.add(this.expos.get(i));
                        }
                  }
                  Writer writer ……......
```

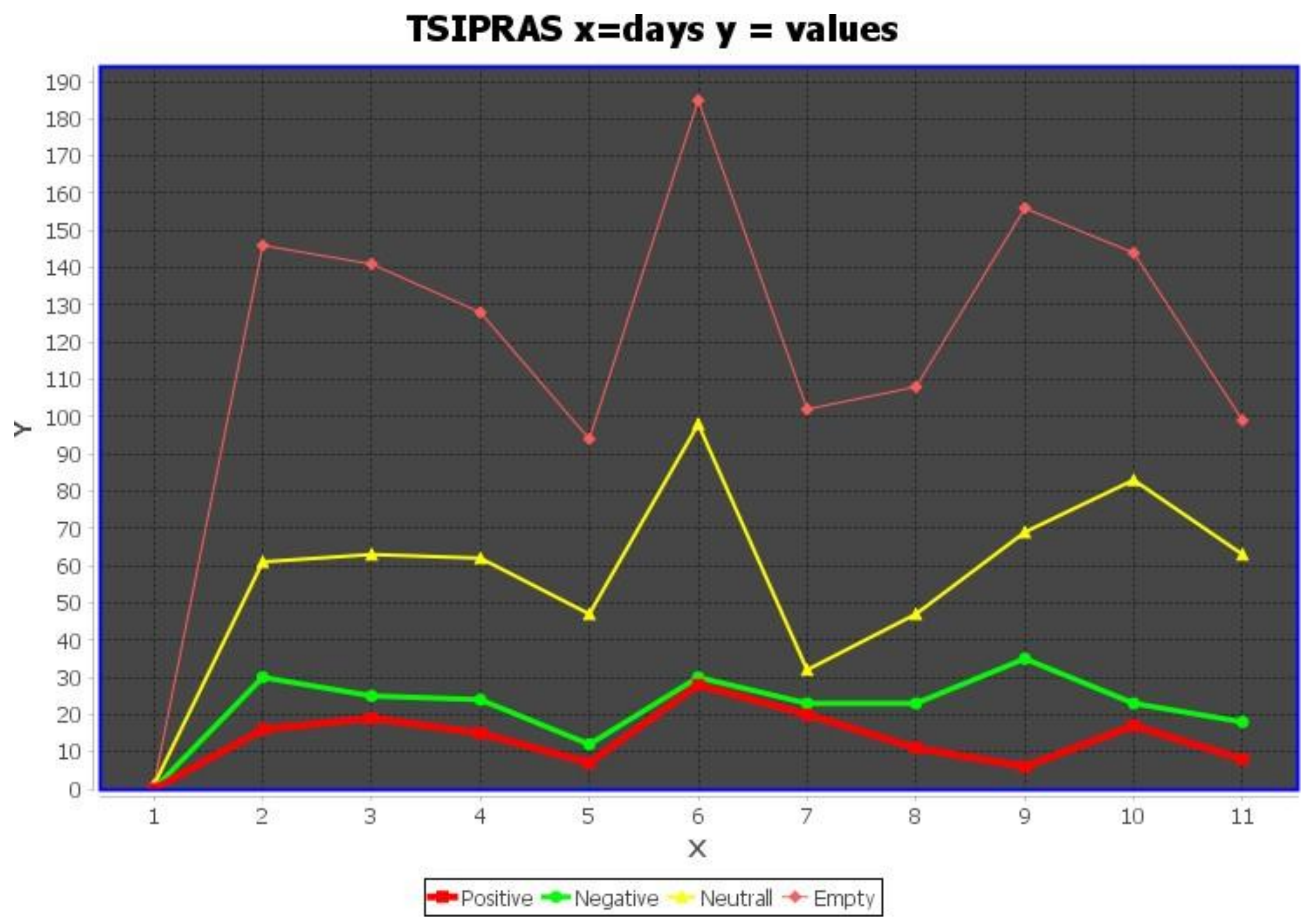
Results:


mitsotakis x=days y=values
Y
240
220
200
180
160
140
120
100
80
60
40
20
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5
9.0
9.5
10.0
X
Positive
Negative
Neutrall
Empty



nd x=days y=values
Y
X
0
2
4
6
8
10
12
14
16
18
20
22
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5
9.0
9.5
10.0
Positive
Negative
Neutrall
Empty


SYRIZA x=days y=values
50
45
40
35
30
25
20
15
10
5
0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
8.5
9.0
9.5
10.0
X
Positive
Negative
Neutrall
Empty



```
Mhtsotakis nearest :1   MeanNeg erwtima 7 = 0.08641975308641975
Mhtsotakis nearest :1   MeanPos erwtima 7 =  0.02564102564102564
Mhtsotakis nearest :2   MeanNeg erwtima 7 = 0.08641975308641975
Mhtsotakis nearest :2   MeanPos erwtima 7 =  0.02564102564102564
Mhtsotakis nearest :4   MeanNeg erwtima 7 = 0.07716049382716049
Mhtsotakis nearest :4   MeanPos erwtima 7 =  0.019230769230769232
Mhtsotakis nearest :5   MeanNeg erwtima 7 = 0.07901234567901234
Mhtsotakis nearest :5   MeanPos erwtima 7 =  0.020512820512820513
Mhtsotakis nearest :10   MeanNeg erwtima 7 = 0.07901234567901234
Mhtsotakis nearest :10   MeanPos erwtima 7 =  0.02564102564102564

Mhtsotakisnumber of tweets = 1128
Mhtsotakis mean positive = 0.40963855421686746
Mhtsotakis mean negative = 0.5903614457831325
Mhtsotakis standar deviation first for positive tweets and second for negative tweets v  = [3.0983866769659336, 2.170770576997759]
Mhtsotakis standar deviation as bernouli = 0.49176702727610966
```

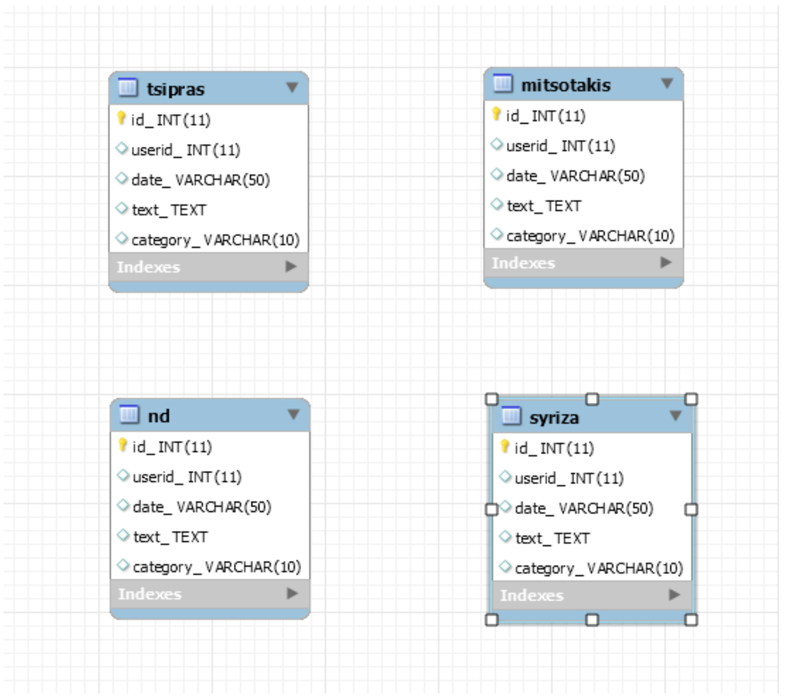
```
Nea-Dhmokratia nearest :1   MeanNeg erwtima 7 = 0.0
Nea-Dhmokratia nearest :1   MeanPos erwtima 7 =  0.3333333333333333
Nea-Dhmokratia nearest :2   MeanNeg erwtima 7 = 0.0
Nea-Dhmokratia nearest :2   MeanPos erwtima 7 =  0.16666666666666666
Nea-Dhmokratia nearest :4   MeanNeg erwtima 7 = 0.0
Nea-Dhmokratia nearest :4   MeanPos erwtima 7 =  0.08333333333333333
Nea-Dhmokratia nearest :5   MeanNeg erwtima 7 = 0.0
Nea-Dhmokratia nearest :5   MeanPos erwtima 7 =  0.1
Nea-Dhmokratia nearest :10   MeanNeg erwtima 7 = 0.01818181818181818
Nea-Dhmokratia nearest :10   MeanPos erwtima 7 =  0.08333333333333333

Nea-Dhmokratianumber of tweets = 133
Nea-Dhmokratia mean positive = 0.3488372093023256
Nea-Dhmokratia mean negative = 0.6511627906976745
Nea-Dhmokratia standar deviation first for positive tweets and second for negative tweets v  = [0.9128709291752769, 0.8783100656536799]
Nea-Dhmokratia standar deviation as bernouli = 0.4766023612074232
```

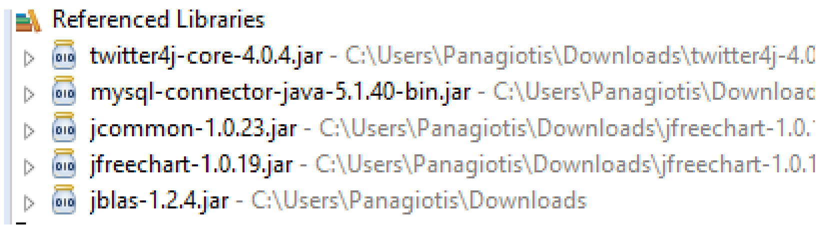
```
Syriza nearest :1   MeanNeg erwtima 7 = 0.05555555555555555
Syriza nearest :1   MeanPos erwtima 7 =  0.0
Syriza nearest :2   MeanNeg erwtima 7 = 0.06944444444444445
Syriza nearest :2   MeanPos erwtima 7 =  0.0
Syriza nearest :4   MeanNeg erwtima 7 = 0.06944444444444445
Syriza nearest :4   MeanPos erwtima 7 =  0.0
Syriza nearest :5   MeanNeg erwtima 7 = 0.07777777777777778
Syriza nearest :5   MeanPos erwtima 7 =  0.01
Syriza nearest :10   MeanNeg erwtima 7 = 0.1
Syriza nearest :10   MeanPos erwtima 7 =  0.04

Syrizanumber of tweets = 397
Syriza mean positive = 0.330188679245283
Syriza mean negative = 0.6698113207547169
Syriza standar deviation first for positive tweets and second for negative tweets v  = [0.6866065623255951, 1.1063198732608175]
Syriza standar deviation as bernouli = 0.4702808898345101
```

```
Tsipras nearest :1   MeanNeg erwtima 7 = 0.08163265306122448
Tsipras nearest :1   MeanPos erwtima 7 =  0.0
Tsipras nearest :2   MeanNeg erwtima 7 = 0.07653061224489796
Tsipras nearest :2   MeanPos erwtima 7 =  0.0
Tsipras nearest :4   MeanNeg erwtima 7 = 0.07908163265306123
Tsipras nearest :4   MeanPos erwtima 7 =  0.00980392156862745
Tsipras nearest :5   MeanNeg erwtima 7 = 0.08571428571428572
Tsipras nearest :5   MeanPos erwtima 7 =  0.01568627450980392
Tsipras nearest :10   MeanNeg erwtima 7 = 0.08163265306122448
Tsipras nearest :10   MeanPos erwtima 7 =  0.021568627450980392

Tsiprasnumber of tweets = 1304
Tsipras mean positive = 0.3769230769230769
Tsipras mean negative = 0.6230769230769231
Tsipras standar deviation first for positive tweets and second for negative tweets v  = [2.054604012935115, 1.9382537332881142]
Tsipras standar deviation as bernouli = 0.4846153846153846
```

<u>Σχόλια:</u>

Έχουν παραδοθεί μαζί με τον κώδικα και τα jars που χρειάζεται για τα jblas, Jfreechart, mysql (χρειάζεται και το twitter4j)

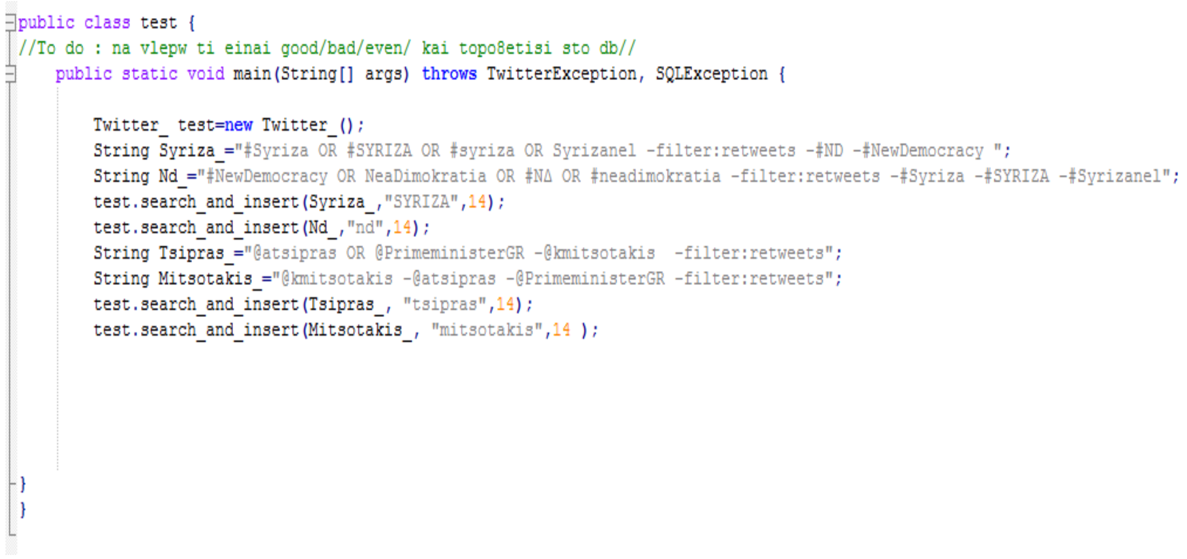




```
Referenced Libraries
  twitter4j-core-4.0.4.jar - C:\Users\Panagiotis\Downloads\twitter4j-4.0
  mysql-connector-java-5.1.40-bin.jar - C:\Users\Panagiotis\Downloac
  jcommon-1.0.23.jar - C:\Users\Panagiotis\Downloads\jfreechart-1.0.
  jfreechart-1.0.19.jar - C:\Users\Panagiotis\Downloads\jfreechart-1.0.1
  jblas-1.2.4.jar - C:\Users\Panagiotis\Downloads
```

The empty Tweets were many.

Tweets insertion



```
public class test {
//To do : na vlepw ti einai good/bad/even/ kai topo8etisi sto db//
    public static void main(String[] args) throws TwitterException, SQLException {

        Twitter_ test=new Twitter_();
        String Syriza_="#Syriza OR #SYRIZA OR #syriza OR Syrizanel -filter:retweets -#ND -#NewDemocracy ";
        String Nd_="#NewDemocracy OR NeaDimokratia OR #NΔ OR #neadimokratia -filter:retweets -#Syriza -#SYRIZA -#Syrizanel";
        test.search_and_insert(Syriza_,"SYRIZA",14);
        test.search_and_insert(Nd_,"nd",14);
        String Tsipras_="@atsipras OR @PrimeministerGR -@kmitsotakis  -filter:retweets";
        String Mitsotakis_="@kmitsotakis -@atsipras -@PrimeministerGR -filter:retweets";
        test.search_and_insert(Tsipras_, "tsipras",14);
        test.search_and_insert(Mitsotakis_, "mitsotakis",14 );




}
}
```