

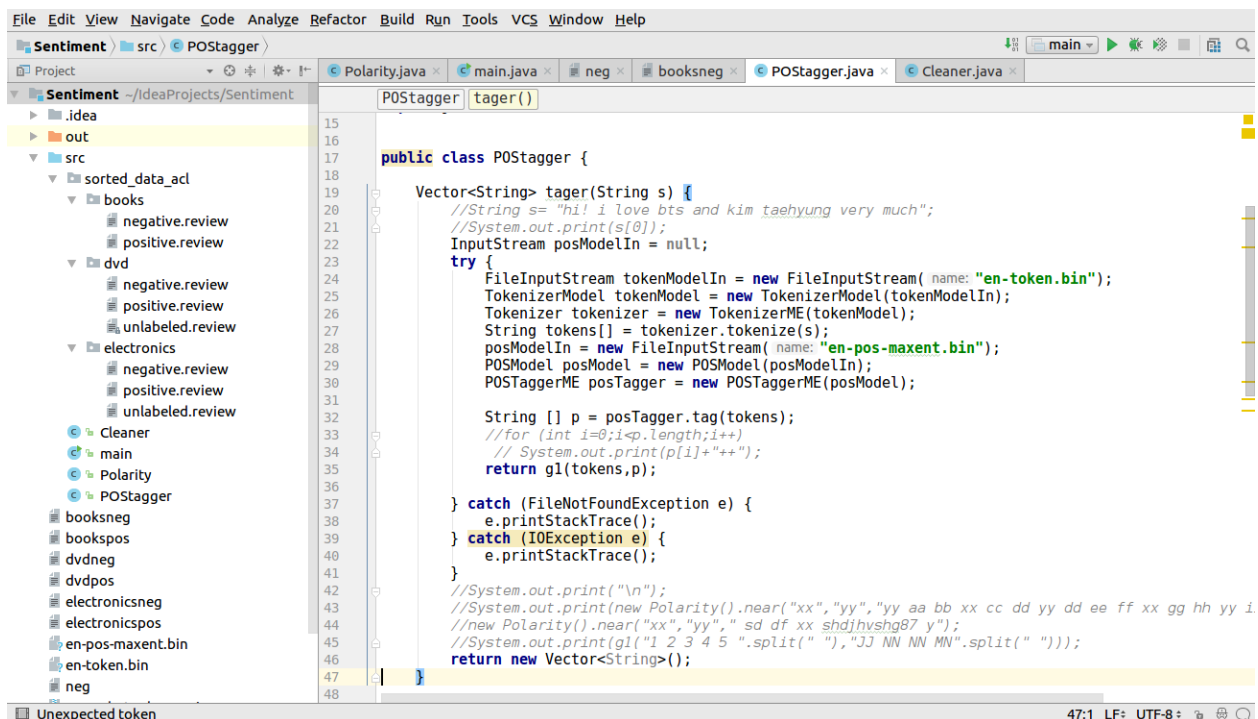
# Sentiment Analysis

بنفشه کریمیان 94521189

طبق اسلاید ابتدا میبایست part of speech tagging انجام داده و جفت کلماتی که نقش به شکل جدول زیر دارند را استخراج کنیم

First Word	Second Word	Third Word (not extracted)
JJ	NN or NNS	anything
RB, RBR, RBS	JJ	Not NN nor NNS
JJ	JJ	Not NN or NNS
NN or NNS	JJ	Nor NN nor NNS
RB, RBR, or RBS	VB, VBD, VBN, VBG	anything

برای استخراج pos از کتابخانه ی opennlp استفاده شده که بعد از توکنایز کردن با استفاده از یک فایل pos کلمات را به ما برمیگرداند.



```
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Sentiment src POStagger
Project
Sentiment ~\IdeaProjects\Sentiment
  .idea
  out
  src
    sorted_data_acl
      books
        negative.review
        positive.review
      dvd
        negative.review
        positive.review
        unlabeled.review
      electronics
        negative.review
        positive.review
        unlabeled.review
    Cleaner
    main
    Polarity
    POStagger
  booksneg
  bookspos
  dvdneg
  dvdpos
  electronicsneg
  electronicspos
  en-pos-maxent.bin
  en-token.bin
  neg

POStagger tager()
15
16
17 public class POStagger {
18
19   Vector<String> tager(String s) {
20     //String s = "hi! i love bts and kim taehyung very much";
21     //System.out.print(s[0]);
22     InputStream posModelIn = null;
23     try {
24       FileInputStream tokenModelIn = new FileInputStream( name: "en-token.bin");
25       TokenizerModel tokenModel = new TokenizerModel(tokenModelIn);
26       Tokenizer tokenizer = new TokenizerME(tokenModel);
27       String tokens[] = tokenizer.tokenize(s);
28       posModelIn = new FileInputStream( name: "en-pos-maxent.bin");
29       POSModel posModel = new POSModel(posModelIn);
30       POSTaggerME posTagger = new POSTaggerME(posModel);
31
32       String [] p = posTagger.tag(tokens);
33       //for (int i=0;i<p.length;i++)
34       // System.out.print(p[i]+" ");
35       return gl(tokens,p);
36     } catch (FileNotFoundException e) {
37       e.printStackTrace();
38     } catch (IOException e) {
39       e.printStackTrace();
40     }
41   }
42   //System.out.print("\n");
43   //System.out.print(new Polarity().near("xx","yy","yy aa bb xx cc dd yy dd ee ff xx gg hh yy i
44   //new Polarity().near("xx","yy"," sd df xx shdjhvshg87 y");
45   //System.out.print(gl("I 2 3 4 5 ".split(" "), "JJ NN NN MN".split(" ")));
46   return new Vector<String>();
47 }
48
```

Unexpected token

47:1 LF: UTF-8

سپس برای استخراج جفت کلمات تابع زیر را استفاده میکنیم.

```

49 Vector<String> g1(String[] s, String[] p){
50
51 Vector<String> o = new Vector<String>();
52 for (int i=0;i<p.length-3;i++){
53
54     if (p[i].equals("JJ")){
55         if (p[i+1].equals("NN")|| p[i+1].equals("NNS")){
56
57             o.add(s[i]+" "+s[i+1]);
58
59         }
60         if(p[i+1].equals("JJ")){
61             if(!p[i+2].equals("NN")&&!p[i+2].equals("NNS")){
62
63                 o.add(s[i]+" "+s[i+1]);
64             }
65         }
66     }
67
68     if (p[i].equals("RB")||p[i].equals("RBR")||p[i].equals("RBS")){
69         if (p[i+1].equals("VB")|| p[i+1].equals("VBD")||p[i+1].equals("VBN")||p[i+1].equals("
70
71             o.add(s[i]+" "+s[i+1]);
72
73         }
74         if(p[i+1].equals("JJ")){
75             if(!p[i+2].equals("NN")&&!p[i+2].equals("NNS")){
76
77                 o.add(s[i]+" "+s[i+1]);
78             }
79         }
80     }
81
82

```

سپس تابع پولاریتی را به صورت زیر تعریف میکنیم :

$$\text{Polarity}(\text{phrase}) = \text{PMI}(\text{phrase}, \text{"excellent"}) - \text{PMI}(\text{phrase}, \text{"poor"})$$

$$= \log_2 \frac{\text{hits}(\text{phrase NEAR "excellent"})}{\text{hits}(\text{phrase})\text{hits}(\text{"excellent"})} - \log_2 \frac{\text{hits}(\text{phrase NEAR "poor"})}{\text{hits}(\text{phrase})\text{hits}(\text{"poor"})}$$

$$= \log_2 \frac{\text{hits}(\text{phrase NEAR "excellent"})}{\text{hits}(\text{phrase})\text{hits}(\text{"excellent"})} \frac{\text{hits}(\text{phrase})\text{hits}(\text{"poor"})}{\text{hits}(\text{phrase NEAR "poor"})}$$

$$= \log_2 \left( \frac{\text{hits}(\text{phrase NEAR "excellent"})\text{hits}(\text{"poor"})}{\text{hits}(\text{phrase NEAR "poor"})\text{hits}(\text{"excellent"})} \right)$$

```

1  import ...
2
3  /**
4   * Created by banafshbts on 18. 7. 8.
5   */
6
7  public class Polarity {
8      Map<String,Double> polarities =new HashMap<>();
9      //Vector<String> data = new Vector<String>();
10     static double log(int x, int base) { return (Math.log(x) / Math.log(base)); }
11     float PMI(String w1, String w2, Vector<String> text){
12         float ret = (float) 0.0;
13         int h1=1;
14         int h2=1;
15         int h3=1;
16         for (int i=0;i<text.size();i++) {
17             h1 += (near(w1, w2, text.get(i))+near(w2, w1, text.get(i)));
18             h2+=hits(w1,text.get(i));
19             h3+=hits(w2,text.get(i));
20         }
21         ret = (float) ( log(h1, base: 2)-log(h2, base: 2)-log(h3, base: 2));
22         //System.out.print("h1"+h1+"h2"+h2+"h3"+h3+"\n");
23         //ret = h1/(h2*h3);
24         return ret;
25     }
26     float PMI2(String w1, String w2, String w3, Vector<String> text){
27         float ret = (float) 0.0;
28         int h1=1;
29         int h2=1;
30         int h3=1;
31         int h4 =1;
32         int h5 =1;
33         for (int i=0;i<text.size();i++) {
34             h1 += hits(w1,text.get(i));
35             h2+=hits(w2,text.get(i));
36         }
37     }
38 }

```

بزای تابع پولاریتی تابع near را به شکل زیر به گونه ای تنظیم کردیم ک اگر تا 6 کلمه در یک کامنت باهم فاصله داشتند نزدیک به هم نزدیک شوند

```

51     return ret;
52 }
53 int near(String w1,String w2,String text){
54
55     //NEAR??
56     int n=0;
57     String[] x = text.split(w1);
58     int limit = 6;
59     if(x.length==1)
60         return 0;
61     for (int i=0;i<x.length;i++){
62         String []y=x[i].split(w2);
63         if(y.length==1||y.length==0) {
64             continue;
65         }
66         // System.out.print(y[0].split(" ").length);
67         if(i!=0&&i!=x.length-1)
68             if(y[0].split( regex: " ").length<limit||y[y.length-1].split( regex: " ").length<limit)
69                 n++;
70         if(i==0)
71             if(y[y.length-1].split( regex: " ").length<limit)
72                 n++;
73         if(i==x.length-1)
74             if(y[0].split( regex: " ").length<limit)
75                 n++;
76     }
77     return n;
78 }
79 int hits(String w,String text){
80
81     return text.split(w).length-1;
82 }
83
84 }

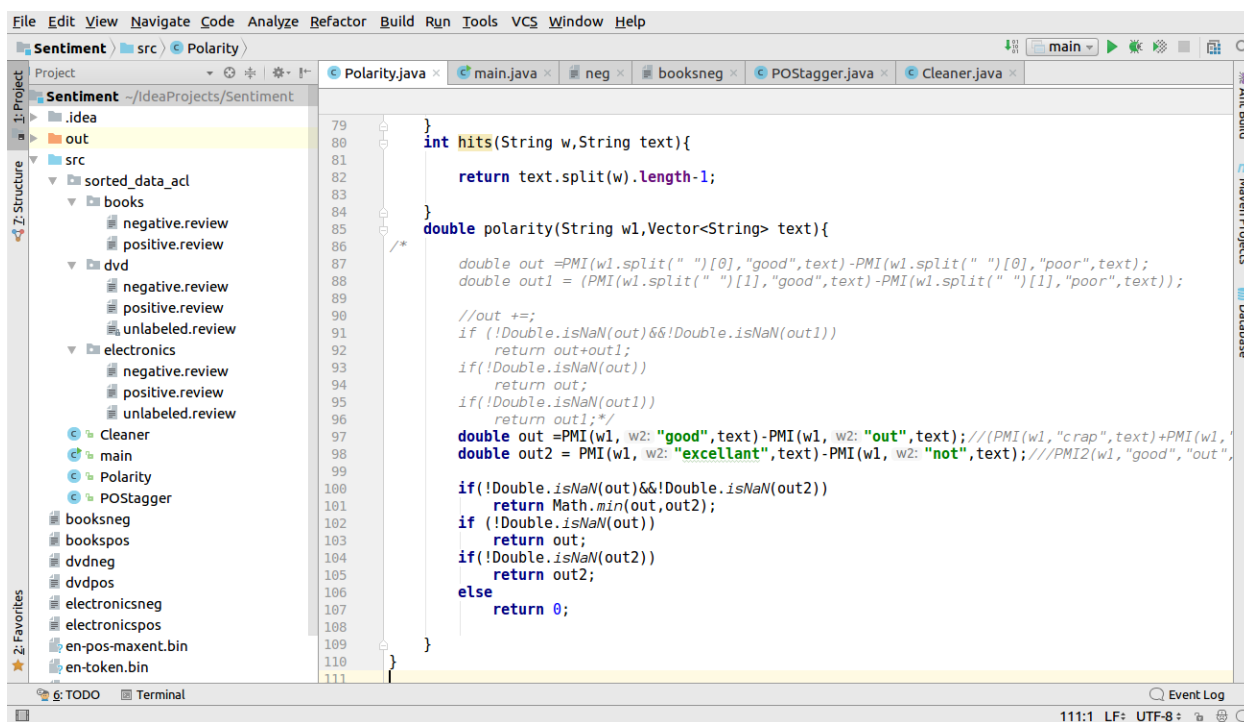
```

سپس برای محاسبه ی پولاریتی ابتدا دو کلمه excellant و poor را به صورت پولاریتی جمع دو کلمه و یا پولاریتی هر دو کلمه حساب کردیم که نتیجه زیاد خوبی به طور کلی نداشت سپس 2جفت کلمه انتخاب کردیم و بین انها مینیم گرفتیم برای جمه پولاریتی دو کلمه که نتیجه بهتر

Precision 52 %

Recall 80%

بدست آمد.

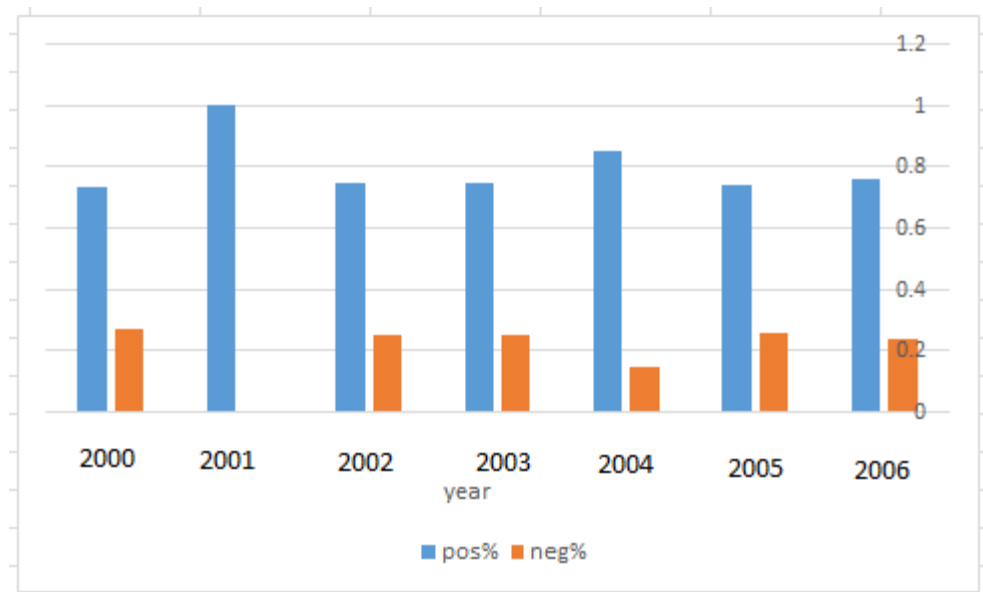


```

79     }
80     int hits(String w,String text){
81
82         return text.split(w).length-1;
83     }
84     double polarity(String w1,Vector<String> text){
85
86         /*
87
88         double out =PMI(w1.split(" ")[0],"good",text)-PMI(w1.split(" ")[0],"poor",text);
89         double out1 = (PMI(w1.split(" ")[1],"good",text)-PMI(w1.split(" ")[1],"poor",text));
90
91         //out +=;
92         if (!Double.isNaN(out)&&!Double.isNaN(out1))
93             return out+out1;
94         if(!Double.isNaN(out))
95             return out;
96         if(!Double.isNaN(out1))
97             return out1;*/
98         double out =PMI(w1, w2: "good",text)-PMI(w1, w2: "out",text);//(PMI(w1,"crap",text)+PMI(w1,
99         double out2 = PMI(w1, w2: "excellant",text)-PMI(w1, w2: "not",text);//(PMI2(w1,"good","out",
100
101         if(!Double.isNaN(out)&&!Double.isNaN(out2))
102             return Math.min(out,out2);
103         if (!Double.isNaN(out))
104             return out;
105         if(!Double.isNaN(out2))
106             return out2;
107         else
108             return 0;
109     }
110 }
111

```

دیتای جمع اوری شده برای این کار کامنت ها درباره ی سه کتگوری در چند سال اند که جدول زیر درصد مثبت و منفی حدس زده شده برای مثال برای کتگوری الکترونیک هست.



که در کل در هر سال میزان مثبت تشخیص داده شده بیشتر از منفی تشخیص داده شده است که البته تا حدی با پریسیژن به دست آمده قابل توجیح است.