Natural Language Processing - II

Contents

1. What Is NLP
2. Sentiment Analysis
3. Sentiment Analysis using R-Package “sentimentr”
4. Sentiment Analysis using R-Package "syuzhet“

What Is NLP

- **Natural Language Processing (NLP)** is the ability of a computer to analyze and process natural language data.
- NLP is used to extract relevant information from a piece of text which is then used for various purposes.
- NLP works on four levels - lexical, syntactic, semantic, pragmatic.
  - **Lexical** - pre-processing of the text, such as removal of stop words, making all text lowercase etc.
  - **Syntax analysis** - It analyses the ‘structure’ of text, ‘correctness’ of a sentence in terms of the grammar of the language of origin.
  - **Semantic assessment** - attempts to study the ‘meaning’ of the text.
  - **Pragmatic** - the analysis is aimed at deciphering the ‘intended’ meaning of the text.

Applications of NLP

Some of the most notable applications of NLP are:

- Search algorithms - when you search for "What is the population of India", the top result shows the actual answer
- General websites - Pop-up windows on websites offering 'chat with their representative' these are chatbots trained to correctly answer commonly asked questions.
- Retail - Assessment of product feedback using text summarization and sentiment analysis; Query resolution with automated responses.
- Personalized services - Email apps predicting next word(s) in an email, tagging emails as important, personal etc.
- Translation Apps

Sentiment Analysis

- Sentiment Analysis is the **process of determining whether a piece of writing is positive, negative or neutral**.
- It’s also known as opinion mining, deriving the opinion or attitude of a speaker.
- Sentiment analysis is performed using natural language processing, text analysis, computational linguistics and, sometimes, biometrics to systematically identify, extract, quantify, and study affective states and subjective information.
- Basic task in sentiment analysis is classifying the **polarity** of a given text at the document, sentence, or feature/aspect level—whether the expressed opinion is positive, negative, or neutral.
- Advanced - "beyond polarity" - sentiment classification looks at emotional states, for instance, "angry", "sad", and "happy".

Sentiment Analysis Using "sentimentr"

- The sentimentr package was developed by Tyler Rinker.
- It adopts a dictionary lookup approach that tries to incorporate weighting for valence shifters.
- It's aim is to improve the polarity recognition performance with respect to the syuzhet package (which does not recognize valence shifters), it does so at the expense of speed.

Sentiment Analysis Using "sentimentr"

```
# Install and Load package "sentimentr"
```

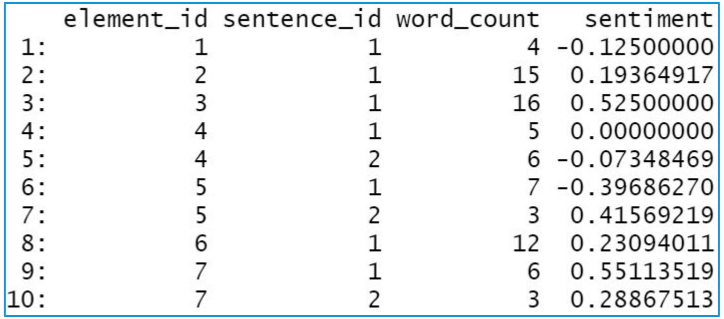
```r
install.packages("sentimentr")
library(sentimentr)
```

```
# Calculate Sentiment Score
```

```r
data<-readLines("HR Appraisal process.txt")
sentiment(data)
```

- ❑ **sentiment()** calculates the sentiment values of each sentence in the data.

```
# Output
```



```
    element_id sentence_id word_count   sentiment
 1:          1           1          4 -0.12500000
 2:          2           1         15  0.19364917
 3:          3           1         16  0.52500000
 4:          4           1          5  0.00000000
 5:          4           2          6 -0.07348469
 6:          5           1          7 -0.39686270
 7:          5           2          3  0.41569219
 8:          6           1         12  0.23094011
 9:          7           1          6  0.55113519
10:          7           2          3  0.28867513
```

- ❑ **element_id** is the id number of the original vector passed to sentiment
- ❑ **sentence_id** is the id number of the sentences within each element_id
- ❑ **word_count** is the count of words in each sentence
- ❑ **sentiment** is the sentiment/polarity score of each sentence

Sentiment Analysis Using "sentimentr"

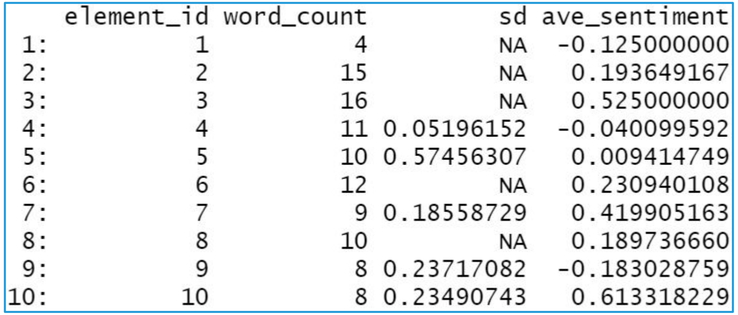
```
# Aggregate sentiment scores”
```

You can also calculate the sentiment scores by aggregating it with respect to different elements. The default value is by=”NULL”, which aggregates the sentiment scores with respect to each line

```
# Calculate Avg Sentiment Score
sentiment_by(data)
```

- ❑ **Sentiment_by()** calculates the aggregate sentiment values

```
# Output
```



```
    element_id word_count         sd ave_sentiment
 1:          1          4         NA  -0.125000000
 2:          2         15         NA   0.193649167
 3:          3         16         NA   0.525000000
 4:          4         11 0.05196152  -0.040099592
 5:          5         10 0.57456307   0.009414749
 6:          6         12         NA   0.230940108
 7:          7          9 0.18558729   0.419905163
 8:          8         10         NA   0.189736660
 9:          9          8 0.23717082  -0.183028759
10:         10          8 0.23490743   0.613318229
```

- ❑ **sd** gives the standard deviation of the sentiment score of the sentences in the review
- ❑ **ave_sentiment** gives the average sentiment score of the sentences in the review

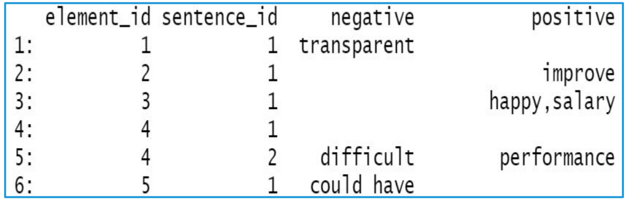
* Note : Sentiment Categories can be derived using Sentiment Scores

Sentiment Analysis Using "sentimentr"

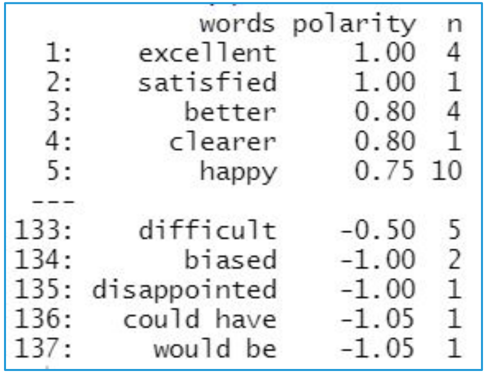
```
# Extract Sentiment words and polarity form each sentence
t<-extract_sentiment_terms(data)
head(t)
attributes(t)$count
```

- ❑ **extract_sentiment_terms()** extracts the sentiment words from a text.
- ❑ **attributes(t)$count** return an aggregated count of the usage of the words and a detailed sentiment score of each word use.

```
# Output
```



```
   element_id sentence_id    negative         positive
1:          1           1 transparent
2:          2           1                      improve
3:          3           1                 happy,salary
4:          4           1
5:          4           2   difficult      performance
6:          5           1  could have
```



```
             words polarity  n
  1:     excellent     1.00  4
  2:     satisfied     1.00  1
  3:        better     0.80  4
  4:       clearer     0.80  1
  5:         happy     0.75 10
 ---
133:     difficult    -0.50  5
134:        biased    -1.00  2
135: disappointed    -1.00  1
136:    could have    -1.05  1
137:      would be    -1.05  1
```

Sentiment Analysis Using "syuzhet"

- Syuzhet is an R package for the extraction of sentiment and sentiment-based plot arcs from text.
- The name “Syuzhet” comes from the Russian Formalists Victor Shklovsky and Vladimir Propp who divided narrative into two components, the “fabula” and the “syuzhet.” Syuzhet refers to the “device” or technique of a narrative whereas fabula is the chronological order of events. Syuzhet, therefore, is concerned with the manner in which the elements of the story (fabula) are organized (syuzhet).
- The package is more suitable for analysis of sentiment trajectory across a text document.

Sentiment Analysis Using "syuzhet"

- Syuzhet incorporates four sentiment lexicons:

“syuzhet” (Default)

- Developed in the Nebraska Literary Lab under the direction of Matthew L. Jockers

“afinn”

- Developed by Finn Arup Nielsen as the AFINN WORD DATABASE

“bing”

- Developed by Minqing Hu and Bing Liu as the OPINION LEXICON

“nrc”

- Developed by Mohammad, Saif M. and Turney, Peter D. as the NRC EMOTION LEXICON

Sentiment Analysis Using "syuzhet"

```
# Install and Load package “syuzhet”
```

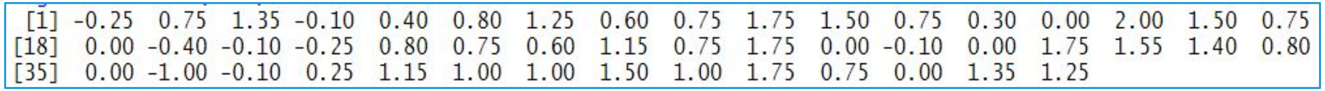
```
install.packages("syuzhet")
library(syuzhet)
```

```
# Calculate Sentiment Values
```

```
get_sentiment(data)
```

- ❑ **get_sentiment()** calculates sentiment of each word or sentence. First argument is a character vector (or sentences or words) and second argument is for method (Which lexicon to be used). The function uses **method="syuzhet"** by default.
- ❑ We have passed data object as it is to the function. This ensures feedback with more than one sentences is not split and considered entirely. However, it is a better practice to 'tokenise' data. Sentences separated by full stops are split.

```
# Output
```



```
 [1] -0.25  0.75  1.35 -0.10  0.40  0.80  1.25  0.60  0.75  1.75  1.50  0.75  0.30  0.00  2.00  1.50  0.75
[18]  0.00 -0.40 -0.10 -0.25  0.80  0.75  0.60  1.15  0.75  1.75  0.00 -0.10  0.00  1.75  1.55  1.40  0.80
[35]  0.00 -1.00 -0.10  0.25  1.15  1.00  1.00  1.50  1.00  1.75  0.75  0.00  1.35  1.25
```

Sentiment Analysis Using "syuzhet"

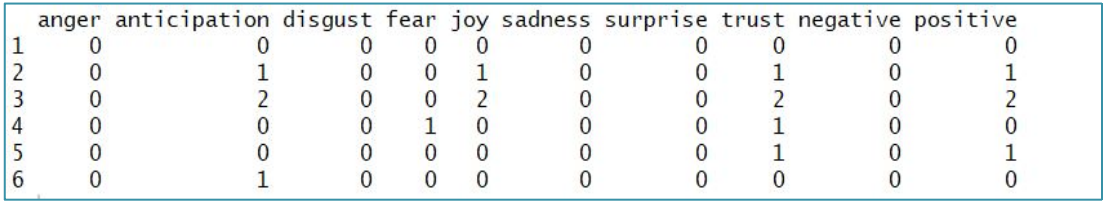
```
# Display emotions and valence from NRC dictionary
```

```
nrcsentiment <- get_nrc_sentiment(data)
head(nrcsentiment)
```

- get_nrc_sentiment() calls the NRC sentiment dictionary to calculate the presence of eight different emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust)and two sentiments (positive and negative).
- It returns a data frame in which each row represents a sentence from the original file. The columns include one for each emotion type as well as the positive or negative sentiment valence.

Sentiment Analysis Using "syuzhet"

```
# Output :
```



```
  anger anticipation disgust fear joy sadness surprise trust negative positive
1     0            0       0    0   0       0        0     0        0        0
2     0            1       0    0   1       0        0     1        0        1
3     0            2       0    0   2       0        0     2        0        2
4     0            0       0    1   0       0        0     1        0        0
5     0            0       0    0   0       0        0     1        0        1
6     0            1       0    0   0       0        0     0        0        0
```

Interpretation:

- Negative score indicates ,negative sentiments.
- Example of how to read output table : second sentence is having joyful sentiments. 4th sentence has fear sentiment.

Sentiment Plot Using "syuzhet"

- It is sometimes useful to plot the values in a graph where the x-axis represents the passage of time from the beginning to the end of the text, and the y-axis measures the degrees of positive and negative sentiment.

- **`plot()`** is a function used for plotting the sequence of sentences in terms of emotions and sentiments.

- This works best when the text being analysed is part of a single document (Like a book, novel, essay, etc.)

Sentiment Plot Using "syuzhet"

```
# Plot the sentiments across time
```

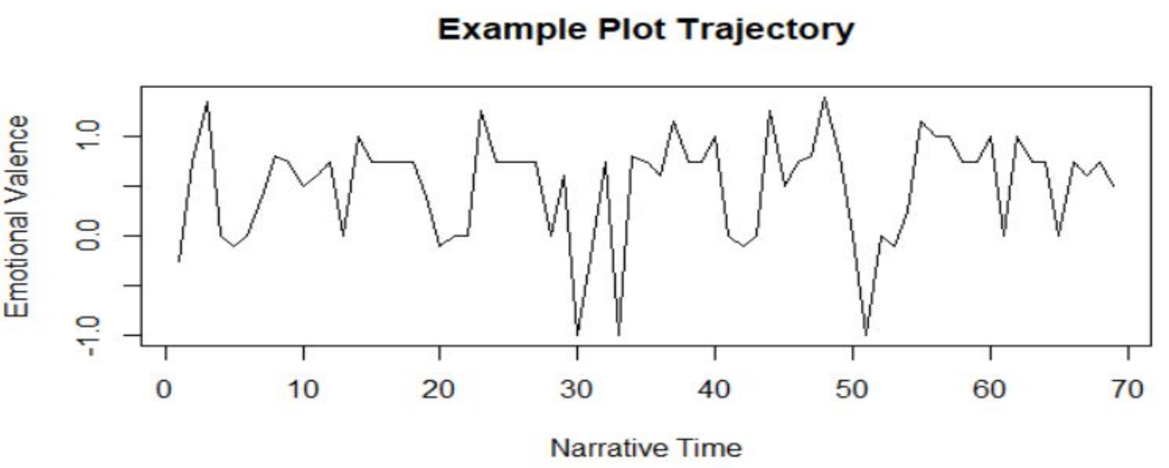
```
read_sentence<- get_sentences(data)
read_sentence_sentiment <- get_sentiment(read_sentence)

plot(
  read_sentence_sentiment,
  type="l",
  main="Example Plot Trajectory",
  xlab = "Narrative Time",
  ylab= "Emotional Valence"
)
```

- **get_sentences()** parses a string into a vector of sentences
- **get_sentiment()** calculates sentiment of each word or sentence. First argument is a character vector (or sentences or words) and second argument is for method (Which lexicon to be used). The function uses method="syuzhet“ by default.
- **type=“1”** is for lines

Sentiment Plot Using "syuzhet"

```
# Output :
```



Interpretation:

- Data mostly varies between neutral and positive sentiments with three negative streaks.

Quick Recap

In this session, we learnt about **NLP & Sentiment Analysis** :

NLP

- Natural Language Processing (NLP) is the ability of a computer to analyze and process natural language data.

Sentiment Analysis

- Sentiment Analysis is the process of determining whether a piece of writing is positive, negative or neutral.
- ‘sentimentr’ & ‘syuzhet’ packages can be used for sentiment analysis.