In the present tutorial, I show an introductory text analysis of a ABC-news news headlines dataset. I will have a look to the most common words therein present and run a sentiment analysis on those headlines by taking advantage of the following sentiment lexicons:

 NRC

 Bing

 AFINN

The NRC sentiment lexicon from Saif Mohammad and Peter Turney categorizes words into categories of positive, negative, anger, anticipation, disgust, fear joy, sadness, surprise and trust.

The Bing sentiment lexicon from Bing Liu and others categorizes words into positive or negative sentiment category.

The AFINN sentiment lexicon from Finn Arup Nielsen assigns words with a score from -5 to 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment.

For more information about those sentiment lexicons, see references listed out at the bottom.

**Packages**

I am going to take advantage of the following R packages.

suppressPackageStartupMessages(library(stringr))

suppressPackageStartupMessages(library(dplyr))

suppressPackageStartupMessages(library(tidytext))

suppressPackageStartupMessages(library(tidyr))

suppressPackageStartupMessages(library(textdata))

suppressPackageStartupMessages(library(widyr))

suppressPackageStartupMessages(library(ggplot2))

Packages versions are herein listed.

packages <- c("stringr", "dplyr", "tidytext", "tidyr", "textdata", "widyr", "ggplot2")

version <- lapply(packages, packageVersion)

version\_c <- do.call(c, version)

data.frame(packages=packages, version = as.character(version\_c))

*## packages version*

*## 1 stringr 1.4.0*

*## 2 dplyr 0.8.4*

*## 3 tidytext 0.2.2*

*## 4 tidyr 1.0.2*

*## 5 textdata 0.3.0*

*## 6 widyr 0.1.2*

*## 7 ggplot2 3.2.1*

Running on Windows-10 the following R language version.

R.version

*## \_*

*## platform x86\_64-w64-mingw32*

*## arch x86\_64*

*## os mingw32*

*## system x86\_64, mingw32*

*## status*

*## major 3*

*## minor 5.3*

*## year 2019*

*## month 03*

*## day 11*

*## svn rev 76217*

*## language R*

*## version.string R version 3.5.3 (2019-03-11)*

*## nickname Great Truth*

**Note**

Before running this code, make sure to have downloaded the lexicon of the sentiments lexicons by executing the following operation:

get\_sentiments("nrc")

get\_sentiments("bing")

get\_sentiments("afinn")

and accepting all prescriptions as asked by the interactive menu showing up.

**Getting Data**

The uncompression produces the abcnews-date-text.csv file. I load it into the news\_data dataset and have a look at.

news\_data <- read.csv("abcnews-date-text.csv", header = TRUE, stringsAsFactors = FALSE)

dim(news\_data)

*## [1] 1103663 2*

head(news\_data)

*## publish\_date headline\_text*

*## 1 20030219 aba decides against community broadcasting licence*

*## 2 20030219 act fire witnesses must be aware of defamation*

*## 3 20030219 a g calls for infrastructure protection summit*

*## 4 20030219 air nz staff in aust strike for pay rise*

*## 5 20030219 air nz strike to affect australian travellers*

*## 6 20030219 ambitious olsson wins triple jump*

tail(news\_data)

*## publish\_date headline\_text*

*## 1103658 20171231 stunning images from the sydney to hobart yacht*

*## 1103659 20171231 the ashes smiths warners near miss liven up boxing day test*

*## 1103660 20171231 timelapse: brisbanes new year fireworks*

*## 1103661 20171231 what 2017 meant to the kids of australia*

*## 1103662 20171231 what the papodopoulos meeting may mean for ausus*

*## 1103663 20171231 who is george papadopoulos the former trump campaign aide*

**Token Analysis**

It is time to extract the tokens from our dataset. Select the column named as *headline\_text* and unnesting the word tokens determine the following.

news\_df <- news\_data %>% select(headline\_text)

news\_tokens <- news\_df %>% unnest\_tokens(word, headline\_text)

head(news\_tokens, 10)

*## word*

*## 1 aba*

*## 1.1 decides*

*## 1.2 against*

*## 1.3 community*

*## 1.4 broadcasting*

*## 1.5 licence*

*## 2 act*

*## 2.1 fire*

*## 2.2 witnesses*

*## 2.3 must*

tail(news\_tokens, 10)

*## word*

*## 1103662.7 ausus*

*## 1103663 who*

*## 1103663.1 is*

*## 1103663.2 george*

*## 1103663.3 papadopoulos*

*## 1103663.4 the*

*## 1103663.5 former*

*## 1103663.6 trump*

*## 1103663.7 campaign*

*## 1103663.8 aide*

It is interesting to generate and inspect a table reporting how many times each token shows up within the headlines and its proportion with respect the total.

news\_tokens\_count <- news\_tokens %>% count(word, sort = TRUE) %>% mutate(proportion = n / sum(n))

The top-10 words which appear most.

head(news\_tokens\_count, 10)

*## # A tibble: 10 x 3*

*## word n proportion*

*##*

*## 1 to 214201 0.0303*

*## 2 in 135981 0.0192*

*## 3 for 130239 0.0184*

*## 4 of 80759 0.0114*

*## 5 on 73037 0.0103*

*## 6 over 50306 0.00711*

*## 7 the 49810 0.00704*

*## 8 police 35984 0.00509*

*## 9 at 31723 0.00449*

*## 10 with 29676 0.00420*

And the ones which appear less frequently:

tail(news\_tokens\_count, 10)

*## # A tibble: 10 x 3*

*## word n proportion*

*##*

*## 1 zweli 1 0.000000141*

*## 2 zwitkowsky 1 0.000000141*

*## 3 zydelig 1 0.000000141*

*## 4 zygar 1 0.000000141*

*## 5 zygiefs 1 0.000000141*

*## 6 zylvester 1 0.000000141*

*## 7 zynga 1 0.000000141*

*## 8 zyngier 1 0.000000141*

*## 9 zz 1 0.000000141*

*## 10 zzz 1 0.000000141*

There is an issue in having doing that way. The issue is that there are words which do not have relevant role in easing the sentiment analysis, the so called *stop words*. Herein below the stop words wihin our dataset are shown.

data(stop\_words)

head(stop\_words, 10)

*## # A tibble: 10 x 2*

*## word lexicon*

*##*

*## 1 a SMART*

*## 2 a's SMART*

*## 3 able SMART*

*## 4 about SMART*

*## 5 above SMART*

*## 6 according SMART*

*## 7 accordingly SMART*

*## 8 across SMART*

*## 9 actually SMART*

*## 10 after SMART*

To remove stop words as required, we take advantage of the *anti\_join* operation.

news\_tokens\_no\_sp <- news\_tokens %>% anti\_join(stop\_words)

head(news\_tokens\_no\_sp, 10)

*## word*

*## 1 aba*

*## 2 decides*

*## 3 community*

*## 4 broadcasting*

*## 5 licence*

*## 6 act*

*## 7 fire*

*## 8 witnesses*

*## 9 aware*

*## 10 defamation*

Then, counting news tokens again after having removed the stop words.

news\_tokens\_count <- news\_tokens\_no\_sp %>% count(word, sort = TRUE) %>% mutate(proportion = n / sum(n))

head(news\_tokens\_count, 10)

*## # A tibble: 10 x 3*

*## word n proportion*

*##*

*## 1 police 35984 0.00673*

*## 2 govt 16923 0.00317*

*## 3 court 16380 0.00306*

*## 4 council 16343 0.00306*

*## 5 interview 15025 0.00281*

*## 6 fire 13910 0.00260*

*## 7 nsw 12912 0.00242*

*## 8 australia 12353 0.00231*

*## 9 plan 12307 0.00230*

*## 10 water 11874 0.00222*

tail(news\_tokens\_count)

*## # A tibble: 6 x 3*

*## word n proportion*

*##*

*## 1 zygiefs 1 0.000000187*

*## 2 zylvester 1 0.000000187*

*## 3 zynga 1 0.000000187*

*## 4 zyngier 1 0.000000187*

*## 5 zz 1 0.000000187*

*## 6 zzz 1 0.000000187*

Then, I filtering out tokens having more than 8,000 counts.

news\_token\_over8000 <- news\_tokens\_count %>% filter(n > 8000) %>% mutate(word = reorder(word, n))

nrow(news\_token\_over8000)

*## [1] 32*

head(news\_token\_over8000, 10)

*## # A tibble: 10 x 3*

*## word n proportion*

*##*

*## 1 police 35984 0.00673*

*## 2 govt 16923 0.00317*

*## 3 court 16380 0.00306*

*## 4 council 16343 0.00306*

*## 5 interview 15025 0.00281*

*## 6 fire 13910 0.00260*

*## 7 nsw 12912 0.00242*

*## 8 australia 12353 0.00231*

*## 9 plan 12307 0.00230*

*## 10 water 11874 0.00222*

tail(news\_token\_over8000, 10)

*## # A tibble: 10 x 3*

*## word n proportion*

*##*

*## 1 day 8818 0.00165*

*## 2 hospital 8815 0.00165*

*## 3 car 8690 0.00163*

*## 4 coast 8411 0.00157*

*## 5 calls 8401 0.00157*

*## 6 win 8315 0.00156*

*## 7 woman 8213 0.00154*

*## 8 killed 8129 0.00152*

*## 9 accused 8094 0.00151*

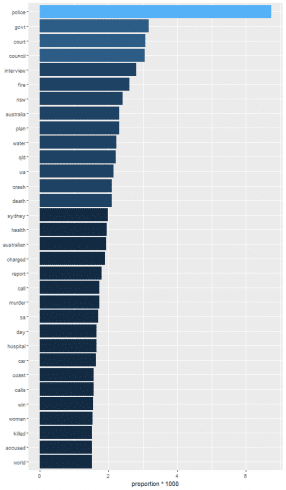
*## 10 world 8087 0.00151*

It is interesting to show the proportion as per-thousands by means of an histogram plot.

news\_token\_over8000 %>%

ggplot(aes(word, proportion\*1000, fill=ceiling(proportion\*1000))) +

geom\_col() + xlab(NULL) + coord\_flip() + theme(legend.position = "none")



**News Sentiment Analysis**

In this paragraph, I focus on each single headline to evaluate its specific sentiment as determined by each lexicon. Hence the output shall determine if each specific headline has got positive or negative sentiment.

head(news\_df, 10)

*## headline\_text*

*## 1 aba decides against community broadcasting licence*

*## 2 act fire witnesses must be aware of defamation*

*## 3 a g calls for infrastructure protection summit*

*## 4 air nz staff in aust strike for pay rise*

*## 5 air nz strike to affect australian travellers*

*## 6 ambitious olsson wins triple jump*

*## 7 antic delighted with record breaking barca*

*## 8 aussie qualifier stosur wastes four memphis match*

*## 9 aust addresses un security council over iraq*

*## 10 australia is locked into war timetable opp*

I will analyse only the first 1000 headlines just for computational time reasons. The token list of such is as follows.

news\_df\_subset <- news\_df[1:1000,,drop=FALSE]

tkn\_l <- apply(news\_df\_subset, 1, function(x) { data.frame(headline\_text=x, stringsAsFactors = FALSE) %>% unnest\_tokens(word, headline\_text)})

Removing the stop words from the token list.

single\_news\_tokens <- lapply(tkn\_l, function(x) {anti\_join(x, stop\_words)})

str(single\_news\_tokens, list.len = 5)

*## List of 1000*

*## $ 1 :'data.frame': 5 obs. of 1 variable:*

*## ..$ word: chr [1:5] "aba" "decides" "community" "broadcasting" ...*

*## $ 2 :'data.frame': 5 obs. of 1 variable:*

*## ..$ word: chr [1:5] "act" "fire" "witnesses" "aware" ...*

*## $ 3 :'data.frame': 4 obs. of 1 variable:*

*## ..$ word: chr [1:4] "calls" "infrastructure" "protection" "summit"*

*## $ 4 :'data.frame': 7 obs. of 1 variable:*

*## ..$ word: chr [1:7] "air" "nz" "staff" "aust" ...*

*## $ 5 :'data.frame': 6 obs. of 1 variable:*

*## ..$ word: chr [1:6] "air" "nz" "strike" "affect" ...*

*## [list output truncated]*

As we can see, to each headline is associated a list of tokens. The sentiment of a headline is computed as based on the sum of positive/negative score of each token of.

single\_news\_tokens[[1]]

*## word*

*## 1 aba*

*## 2 decides*

*## 3 community*

*## 4 broadcasting*

*## 5 licence*

**Bing lexicon**

In this paragraph, the computation of the sentiment associated to the tokens list is shown for *Bing* lexicon. I first define a function named as *compute\_sentiment()* whose purpose is to output the positiveness score of a specific headline.

compute\_sentiment <- function(d) {

if (nrow(d) == 0) {

return(NA)

}

neg\_score <- d %>% filter(sentiment=="negative") %>% nrow()

pos\_score <- d %>% filter(sentiment=="positive") %>% nrow()

pos\_score - neg\_score

}

The inner join on *bing* lexicon of each single headline tokens list is given as input to the *compute\_sentiment()* function to determine the sentiment score of each specific headline.

sentiments\_bing <- get\_sentiments("bing")

str(sentiments\_bing)

*## Classes 'tbl\_df', 'tbl' and 'data.frame': 6786 obs. of 2 variables:*

*## $ word : chr "2-faces" "abnormal" "abolish" "abominable" ...*

*## $ sentiment: chr "negative" "negative" "negative" "negative" ...*

single\_news\_sentiment\_bing <- sapply(single\_news\_tokens, function(x) { x %>% inner\_join(sentiments\_bing) %>% compute\_sentiment()})

The result is a vector of integers each element value at i-th position is the sentiment associated to the i-th news

str(single\_news\_sentiment\_bing)

*## Named int [1:1000] NA -1 1 -1 -1 2 0 NA NA NA ...*

*## - attr(\*, "names")= chr [1:1000] "1" "2" "3" "4" ...*

Here is the summary, please note that:

* the median is negative
* NA's show up

summary(single\_news\_sentiment\_bing)

*## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's*

*## -3.000 -1.000 -1.000 -0.475 1.000 2.000 520*

Collecting the resulting in a data frame as follows.

single\_news\_sentiment\_bing\_df <- data.frame(headline\_text=news\_df\_subset$headline\_text, score = single\_news\_sentiment\_bing)

head(single\_news\_sentiment\_bing\_df, 10)

*## headline\_text score*

*## 1 aba decides against community broadcasting licence NA*

*## 2 act fire witnesses must be aware of defamation -1*

*## 3 a g calls for infrastructure protection summit 1*

*## 4 air nz staff in aust strike for pay rise -1*

*## 5 air nz strike to affect australian travellers -1*

*## 6 ambitious olsson wins triple jump 2*

*## 7 antic delighted with record breaking barca 0*

*## 8 aussie qualifier stosur wastes four memphis match NA*

*## 9 aust addresses un security council over iraq NA*

*## 10 australia is locked into war timetable opp NA*

**NRC lexicon**

In this paragraph, the computation of the sentiment associated to the tokens list is shown for *NRC* lexicon. With respect the previous analysis based on *bing* lexicon, some more pre-processing is needed as explained in what follows. First we get the *NRC* sentiment lexicon and see what are the sentiments threin present.

sentiments\_nrc <- get\_sentiments("nrc")

(unique\_sentiments\_nrc <- unique(sentiments\_nrc$sentiment))

*## [1] "trust" "fear" "negative" "sadness" "anger" "surprise"*

*## [7] "positive" "disgust" "joy" "anticipation"*

To have as output a positive/negative sentiment result, I define a mapping of abovelisted sentiments to a positive/negative string result as follows.

compute\_pos\_neg\_sentiments\_nrc <- function(the\_sentiments\_nrc) {

s <- unique(the\_sentiments\_nrc$sentiment)

df\_sentiments <- data.frame(sentiment = s,

mapped\_sentiment = c("positive", "negative", "negative", "negative",

"negative", "positive", "positive", "negative",

"positive", "positive"))

ss <- sentiments\_nrc %>% inner\_join(df\_sentiments)

the\_sentiments\_nrc$sentiment <- ss$mapped\_sentiment

the\_sentiments\_nrc

}

nrc\_sentiments\_pos\_neg\_scale <- compute\_pos\_neg\_sentiments\_nrc(sentiments\_nrc)

Above function is used to produce the single headline text sentiment results. Such result is given as input to the *compute\_sentiment()* function.

single\_news\_sentiment\_nrc <- sapply(single\_news\_tokens, function(x) { x %>% inner\_join(nrc\_sentiments\_pos\_neg\_scale) %>% compute\_sentiment()})

str(single\_news\_sentiment\_nrc)

*## Named int [1:1000] 1 -4 1 2 -2 2 4 NA 5 -2 ...*

*## - attr(\*, "names")= chr [1:1000] "1" "2" "3" "4" ...*

Here is the summary, please note that:

* the median is equal to zero
* NA's show up

summary(single\_news\_sentiment\_nrc)

*## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's*

*## -9.0000 -2.0000 0.0000 -0.3742 2.0000 9.0000 257*

single\_news\_sentiment\_nrc\_df <- data.frame(headline\_text=news\_df\_subset$headline\_text, score = single\_news\_sentiment\_nrc)

head(single\_news\_sentiment\_nrc\_df, 10)

*## headline\_text score*

*## 1 aba decides against community broadcasting licence 1*

*## 2 act fire witnesses must be aware of defamation -4*

*## 3 a g calls for infrastructure protection summit 1*

*## 4 air nz staff in aust strike for pay rise 2*

*## 5 air nz strike to affect australian travellers -2*

*## 6 ambitious olsson wins triple jump 2*

*## 7 antic delighted with record breaking barca 4*

*## 8 aussie qualifier stosur wastes four memphis match NA*

*## 9 aust addresses un security council over iraq 5*

*## 10 australia is locked into war timetable opp -2*

**AFINN lexicon**

In this paragraph, the computation of the sentiment associated to the tokens list is shown for *AFINN* lexicon.

sentiments\_afinn <- get\_sentiments("afinn")

colnames(sentiments\_afinn) <- c("word", "sentiment")

str(sentiments\_afinn)

*## Classes 'spec\_tbl\_df', 'tbl\_df', 'tbl' and 'data.frame': 2477 obs. of 2 variables:*

*## $ word : chr "abandon" "abandoned" "abandons" "abducted" ...*

*## $ sentiment: num -2 -2 -2 -2 -2 -2 -3 -3 -3 -3 ...*

*## - attr(\*, "spec")=*

*## .. cols(*

*## .. word = col\_character(),*

*## .. value = col\_double()*

*## .. )*

As we can see, the *afinn* lexicon provides a score for each token. We just need to sum up each headline tokens score to obtain the sentiment score of the headline under analysis.

single\_news\_sentiment\_afinn\_df <- lapply(single\_news\_tokens, function(x) { x %>% inner\_join(sentiments\_afinn)})

single\_news\_sentiment\_afinn <- sapply(single\_news\_sentiment\_afinn\_df, function(x) {

ifelse(nrow(x) > 0, sum(x$sentiment), NA)

})

str(single\_news\_sentiment\_afinn)

*## Named num [1:1000] NA -2 NA -2 -1 6 3 NA NA -2 ...*

*## - attr(\*, "names")= chr [1:1000] "1" "2" "3" "4" ...*

Here is the summary, please note that:

* the median is negative
* NA's show up

summary(single\_news\_sentiment\_afinn)

*## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's*

*## -9.000 -3.000 -2.000 -1.148 1.000 7.000 508*

single\_news\_sentiment\_afinn\_df <- data.frame(headline\_text=news\_df\_subset$headline\_text, score = single\_news\_sentiment\_afinn)

head(single\_news\_sentiment\_afinn\_df, 10)

*## headline\_text score*

*## 1 aba decides against community broadcasting licence NA*

*## 2 act fire witnesses must be aware of defamation -2*

*## 3 a g calls for infrastructure protection summit NA*

*## 4 air nz staff in aust strike for pay rise -2*

*## 5 air nz strike to affect australian travellers -1*

*## 6 ambitious olsson wins triple jump 6*

*## 7 antic delighted with record breaking barca 3*

*## 8 aussie qualifier stosur wastes four memphis match NA*

*## 9 aust addresses un security council over iraq NA*

*## 10 australia is locked into war timetable opp -2*

**Comparing results**

Having obtained for each news three potential results as sentiment evaluation, we would like to compare their congruency.  
As congruence we mean the fact that all three lexicons express the same positive or negative result, in other words the same score sign indipendently from its magnitude. If NA values are present, the congruence shall be computed until at least two non NA values are available, otherwise is equal to NA.

Furthermore we compute the final news sentiment as based upon the sum of each lexicon sentiment score.

compute\_congruence <- function(x,y,z) {

v <- c(sign(x), sign(y), sign(z))

# if only one lexicon reports the score, we cannot check for congruence

if (sum([is.na](http://is.na)(v)) >= 2) {

return (NA)

}

# removing NA and zero value

v <- na.omit(v)

v\_sum <- sum(v)

abs(v\_sum) == length(v)

}

compute\_final\_sentiment <- function(x,y,z) {

if ([is.na](http://is.na)(x) && [is.na](http://is.na)(y) && [is.na](http://is.na)(z)) {

return (NA)

}

s <- sum(x, y, z, na.rm=TRUE)

# positive sentiments have score strictly greater than zero

# negative sentiments have score strictly less than zero

# neutral sentiments have score equal to zero

ifelse(s > 0, "positive", ifelse(s < 0, "negative", "neutral"))

}

news\_sentiments\_results <- data.frame(headline\_text = news\_df\_subset$headline\_text,

bing\_score = single\_news\_sentiment\_bing,

nrc\_score = single\_news\_sentiment\_nrc,

afinn\_score = single\_news\_sentiment\_afinn,

stringsAsFactors = FALSE)

news\_sentiments\_results <- news\_sentiments\_results %>% rowwise() %>%

mutate(final\_sentiment = compute\_final\_sentiment(bing\_score, nrc\_score, afinn\_score),

congruence = compute\_congruence(bing\_score, nrc\_score, afinn\_score))

head(news\_sentiments\_results, 40)

*## Source: local data frame [40 x 6]*

*## Groups:*

*##*

*## # A tibble: 40 x 6*

*## headline\_text bing\_score nrc\_score afinn\_score final\_sentiment congruence*

*##*

*## 1 aba decides against community broadcas~ NA 1 NA positive NA*

*## 2 act fire witnesses must be aware of de~ -1 -4 -2 negative TRUE*

*## 3 a g calls for infrastructure protectio~ 1 1 NA positive TRUE*

*## 4 air nz staff in aust strike for pay ri~ -1 2 -2 negative FALSE*

*## 5 air nz strike to affect australian tra~ -1 -2 -1 negative TRUE*

*## 6 ambitious olsson wins triple jump 2 2 6 positive TRUE*

*## 7 antic delighted with record breaking b~ 0 4 3 positive FALSE*

*## 8 aussie qualifier stosur wastes four me~ NA NA NA NA*

*## 9 aust addresses un security council ove~ NA 5 NA positive NA*

*## 10 australia is locked into war timetable~ NA -2 -2 negative TRUE*

*## # ... with 30 more rows*

Is would be useful to replace the numeric score with same {negative, neutral, positive} scale.

replace\_score\_with\_sentiment <- function(v\_score) {

v\_score[v\_score > 0] <- "positive"

v\_score[v\_score < 0] <- "negative"

v\_score[v\_score == 0] <- "neutral"

v\_score

}

news\_sentiments\_results$bing\_score <- replace\_score\_with\_sentiment(news\_sentiments\_results$bing\_score)

news\_sentiments\_results$nrc\_score <- replace\_score\_with\_sentiment(news\_sentiments\_results$nrc\_score)

news\_sentiments\_results$afinn\_score <- replace\_score\_with\_sentiment(news\_sentiments\_results$afinn\_score)

news\_sentiments\_results[,2:5] <- lapply(news\_sentiments\_results[,2:5], as.factor)

head(news\_sentiments\_results, 40)

*## Source: local data frame [40 x 6]*

*## Groups:*

*##*

*## # A tibble: 40 x 6*

*## headline\_text bing\_score nrc\_score afinn\_score final\_sentiment congruence*

*##*

*## 1 aba decides against community broadcas~ positive positive NA*

*## 2 act fire witnesses must be aware of de~ negative negative negative negative TRUE*

*## 3 a g calls for infrastructure protectio~ positive positive positive TRUE*

*## 4 air nz staff in aust strike for pay ri~ negative positive negative negative FALSE*

*## 5 air nz strike to affect australian tra~ negative negative negative negative TRUE*

*## 6 ambitious olsson wins triple jump positive positive positive positive TRUE*

*## 7 antic delighted with record breaking b~ neutral positive positive positive FALSE*

*## 8 aussie qualifier stosur wastes four me~ NA*

*## 9 aust addresses un security council ove~ positive positive NA*

*## 10 australia is locked into war timetable~ negative negative negative TRUE*

*## # ... with 30 more rows*

Tabularizations of each lexicon resulting sentiment and final sentiments are herein shown.

table(news\_sentiments\_results$bing\_score, news\_sentiments\_results$final\_sentiment, dnn = c("bing", "final"))

*## final*

*## bing negative neutral positive*

*## negative 278 15 14*

*## neutral 16 6 11*

*## positive 6 7 127*

table(news\_sentiments\_results$nrc\_score, news\_sentiments\_results$final\_sentiment, dnn = c("nrc", "final"))

*## final*

*## nrc negative neutral positive*

*## negative 353 10 4*

*## neutral 18 13 6*

*## positive 25 16 298*

table(news\_sentiments\_results$afinn\_score, news\_sentiments\_results$final\_sentiment, dnn = c("afinn", "final"))

*## final*

*## afinn negative neutral positive*

*## negative 326 10 12*

*## neutral 3 1 6*

*## positive 4 9 121*

Tabularization of congruence and final sentiments is herein shown.

table(news\_sentiments\_results$congruence, news\_sentiments\_results$final\_sentiment, dnn = c("congruence", "final"))

*## final*

*## congruence negative neutral positive*

*## FALSE 67 33 45*

*## TRUE 292 0 132*

**Conclusions**

We analyzed the news headlines to determine their sentiments while taking advantage of three sentiments lexicons. We show some basics of the methodoloy for such purpose. We also had the chance to compare the results obtained across all three lexicons and set forth a final sentiment evaluation. If you are interested in understanding much more about text analysis, see ref. [4].