AA_Opinion_Mining

Kunal_Ray

June 20, 2018

```
#clear the environment

rm(list = ls(all=TRUE))

## Loading the rvest package

library(rvest)
```

```
## Warning: package 'rvest' was built under R version 3.4.1
```

```
## Loading required package: xml2
```

```
## Warning: package 'xml2' was built under R version 3.4.1
```

```
## Breaking down the URL required to be scraped split by page no.

rev1<-'https://www.g2crowd.com/products/adobe-analytics/reviews?page='
rev2<-'&variant=default'

## Creating an empty data frame to store URLs

url<-data.frame(PgNo=as.integer(),
                   URL=character(),
                stringsAsFactors = FALSE)

## Running a loop to create multiple URLs with varied page numbers

x<-1:3
for (i in seq_along(x)) {
  url[i,1]<-i
  url[i,2]<-paste(rev1,i,rev2,sep = "")
}

url ## View contents of the variable 'url'
```

```
##   PgNo
## 1    1
## 2    2
## 3    3
##                                                                               URL
## 1 https://www.g2crowd.com/products/adobe-analytics/reviews?page=1&variant=default
## 2 https://www.g2crowd.com/products/adobe-analytics/reviews?page=2&variant=default
## 3 https://www.g2crowd.com/products/adobe-analytics/reviews?page=3&variant=default
```

```
rm(x,i,rev1,rev2) ## Removing unrequired variables
```

```
## Reading all reviews & combining into a data frame

reviews<-data.frame(URL=character(),
                    REVIEWS=character(),
                    stringsAsFactors = FALSE) ## Creating empty data frame

x<-1:3
for (i in seq_along(x)) {
  reviews[i,1]<-url[i,2]
  cp1<-read_html(url[i,2]) ## Reading html code from website
  cp2<-html_nodes(cp1,'.formatted-text') ## Get all reviews
  cp3<-html_text(cp2) ## Convert to text
  reviews[i,2]<-paste(unlist(cp3), collapse =" ") ## Merge into single row per page
}

rm(url,cp1,cp2,cp3,i,x) ## Removing unrequired variables
```

```
## Combining every review into a single character variable

fin_review<-paste(unlist(reviews[,2]),collapse = " ")

substr(fin_review, 1, 500) ## View the first 500 characters of review
```

```
## [1] "Adobe Analytics has a robustness that is unparalleled by any other web analytics tool. A
dobe Ad Hoc (formerly Discover) is my favorite analytics tool I've used, despite the poor user e
xperience. Discover allows you to dive into the nitty gritty of the user journey and develop com
plex segments that control for different types of visitors, page names, and page views. On top o
f that, having the Workspace tool allows for easy point and click reporting. In order to run Ado
be Ad Hoc (Discover), you ha"
```

```
## Text Analytics ##

library(tidytext) ## Loading the tidytext library
```

```
## Warning: package 'tidytext' was built under R version 3.4.1
```

```
library(dplyr) ## Loading the dplyr library
```

```
## Warning: package 'dplyr' was built under R version 3.4.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
aareviews<-as.data.frame(fin_review,stringsAsFactors = FALSE) ## Creating a DF

tidyreviews<- aareviews %>%
  unnest_tokens(word,fin_review) ## Tokenizing & converting to tidy structure

head(tidyreviews) ## Viewing the first few rows in the tidy format
```

```
##           word
## 1        adobe
## 1.1  analytics
## 1.2        has
## 1.3          a
## 1.4 robustness
## 1.5       that
```

```
## Removing stop words

data(stop_words) ## Load the stop words

custom_stop_words <- bind_rows(data_frame(word = c("adobe","analytics"),
                                          lexicon = c("custom")),
                               stop_words) ## Customizing list of stop words

tidyreviews <- tidyreviews %>%
  anti_join(custom_stop_words) ## Removing the stop words
```

```
## Joining, by = "word"
```

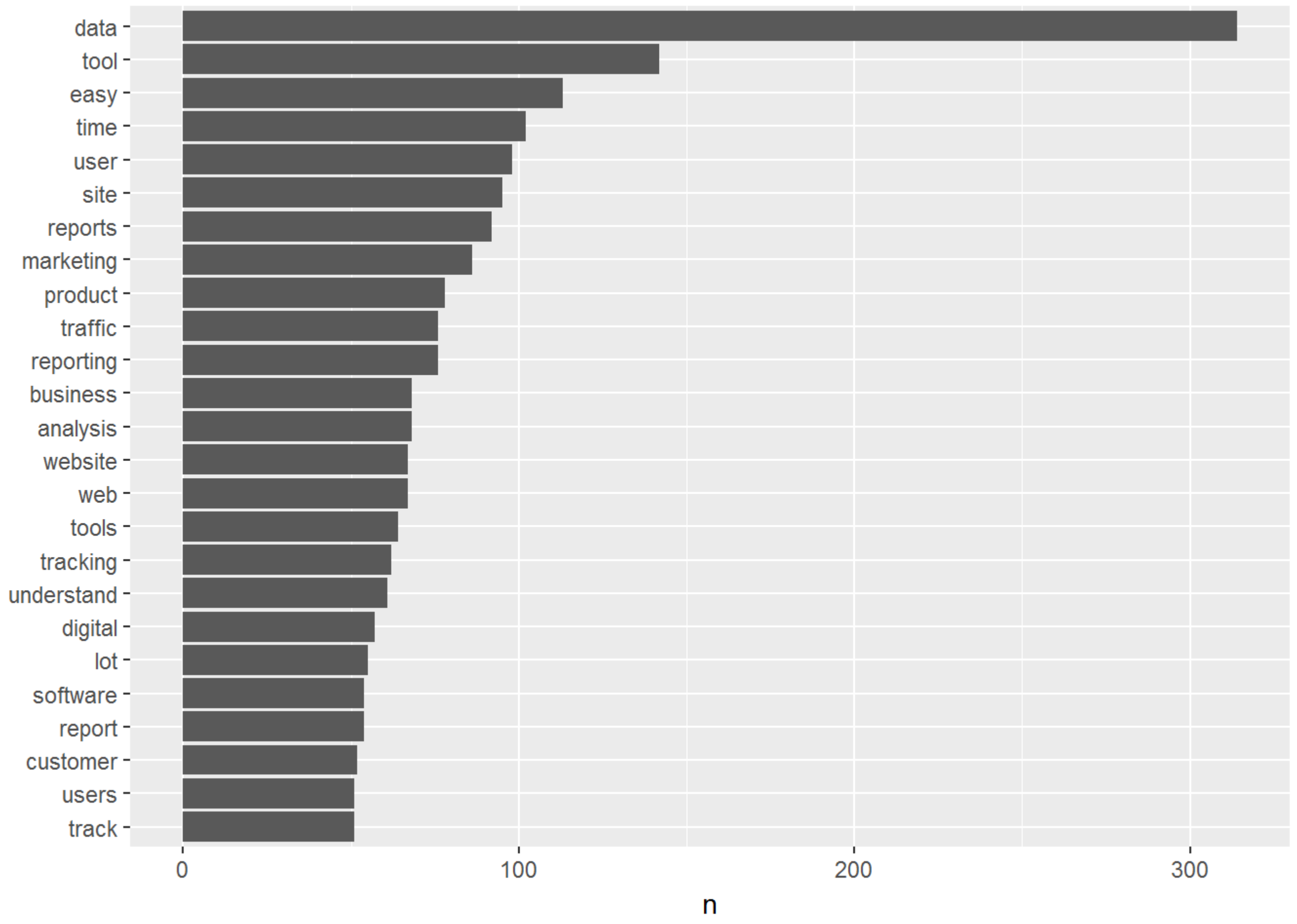
```
tidyreviews %>%
  count(word,sort = TRUE) ## Sorting by most commonly occuring words
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.1
## # A tibble: 2,513 x 2
##    word          n
##    <chr>     <int>
##  1 data        314
##  2 tool        142
##  3 easy        113
##  4 time        102
##  5 user         98
##  6 site         95
##  7 reports      92
##  8 marketing    86
##  9 product      78
## 10 reporting    76
## # ... with 2,503 more rows
```

```
## Plotting the top occuring words

library(ggplot2) ## Loading the ggplot library

tidyreviews%>%
  count(word,sort=TRUE)%>%
  filter(n > 50) %>%
  mutate(word = reorder(word,n)) %>%
  ggplot(aes(word,n))+
  geom_col()+
  xlab(NULL)+
  coord_flip()
```



```
## Using bi-grams instead of words

aareviewsbigram<- aareviews %>%
  unnest_tokens(bigram,fin_review,token = "ngrams",n=2) ## Creating bigrams

aareviewsbigram %>%
  count(bigram,sort = TRUE) ## Checking the top bigrams
```

```
## # A tibble: 16,975 x 2
##    bigram               n
##    <chr>            <int>
##  1 adobe analytics   149
##  2 of the            125
##  3 it is              76
##  4 easy to            74
##  5 able to            72
##  6 to use             72
##  7 can be             71
##  8 you can            66
##  9 the data           60
## 10 is a               57
## # ... with 16,965 more rows
## Need to remove those cases where either word in bigram is a stop word

library(tidyr) ## Loading tidyr library
```

```
## Warning: package 'tidyr' was built under R version 3.4.4
```

```
bigrams_separated <- aareviewsbigram %>%
  separate(bigram,c("word1","word2"),sep = " ") ## separate bigrams into monograms

head(bigrams_separated) ## Checking data structure
```

```
## # A tibble: 6 x 2
##   word1      word2
##   <chr>      <chr>
## 1 adobe      analytics
## 2 analytics  has
## 3 has        a
## 4 a          robustness
## 5 robustness that
## 6 that       is
```

```
bigrams_filtered<-bigrams_separated%>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) ## Filtering out cases where either is a stop word

bigram_counts<-bigrams_filtered%>%
  count(word1,word2,sort = TRUE) ## Getting frequency of occurence

bigram_counts%>%
  filter(n>3)## Getting the top list of bigrams
```

```
## # A tibble: 71 x 3
##    word1    word2         n
##    <chr>    <chr>     <int>
##  1 adobe    analytics   149
##  2 google   analytics    41
##  3 learning curve        35
##  4 user     friendly     34
##  5 web      analytics    26
##  6 ad       hoc          25
##  7 real     time         18
##  8 hoc      analysis     16
##  9 digital  analytics    13
## 10 report   builder      12
## # ... with 61 more rows
## Using trigrams

aareviewstrigram<- aareviews %>%
  unnest_tokens(trigram,fin_review,token = "ngrams",n=3)%>%
  separate(trigram,c("word1","word2","word3"),sep = " ")%>%
  filter(!word1 %in% stop_words$word,
         !word2 %in% stop_words$word,
         !word3 %in% stop_words$word)%>%
  count(word1,word2,word3,sort = TRUE) ## Creating trigram and removing stop words

aareviewstrigram %>%
  filter(n>1) ## Looking at the top trigram occurences
```

```
## # A tibble: 55 x 4
##    word1       word2      word3         n
##    <chr>       <chr>      <chr>     <int>
##  1 ad          hoc        analysis     16
##  2 steep       learning   curve         8
##  3 real        time       data          5
##  4 conversion  variables  success       3
##  5 performance adobe      analytics     3
##  6 pretty      user       friendly      3
##  7 pulls       historical data          3
##  8 user        friendly   easy          3
##  9 variables   success    events        3
## 10 web         analytics  tool          3
## # ... with 45 more rows
```

```
## Analyzing bigrams

negation_words <- c("not", "no", "never", "without")

bigrams_separated%>%
  filter(word1 %in% negation_words) %>%
  count(word1,word2,sort = TRUE) ## Checking those bigrams where first word is negation
```

```
## # A tibble: 125 x 3
##    word1 word2      n
##    <chr> <chr>  <int>
##  1 not   the       11
##  2 not   as         7
##  3 not   have       7
##  4 not   a          6
##  5 not   user       6
##  6 not   always     5
##  7 not   be         5
##  8 not   only       5
##  9 no    longer     4
## 10 not   being      4
## # ... with 115 more rows
## Plotting a network diagram to get a higher level picture

library(igraph) ## Loading the package igraph
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:tidyr':
##
##     %>%, crossing
```

```
## The following objects are masked from 'package:dplyr':
##
##     %>%, as_data_frame, groups, union
```

```
## The following object is masked from 'package:rvest':
##
##     %>%
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

```
bigram_graph <- bigram_counts %>%
  filter(n>5) %>%
  graph_from_data_frame() ## Generate a bigram graph

bigram_graph ## Check the bigram graph
```

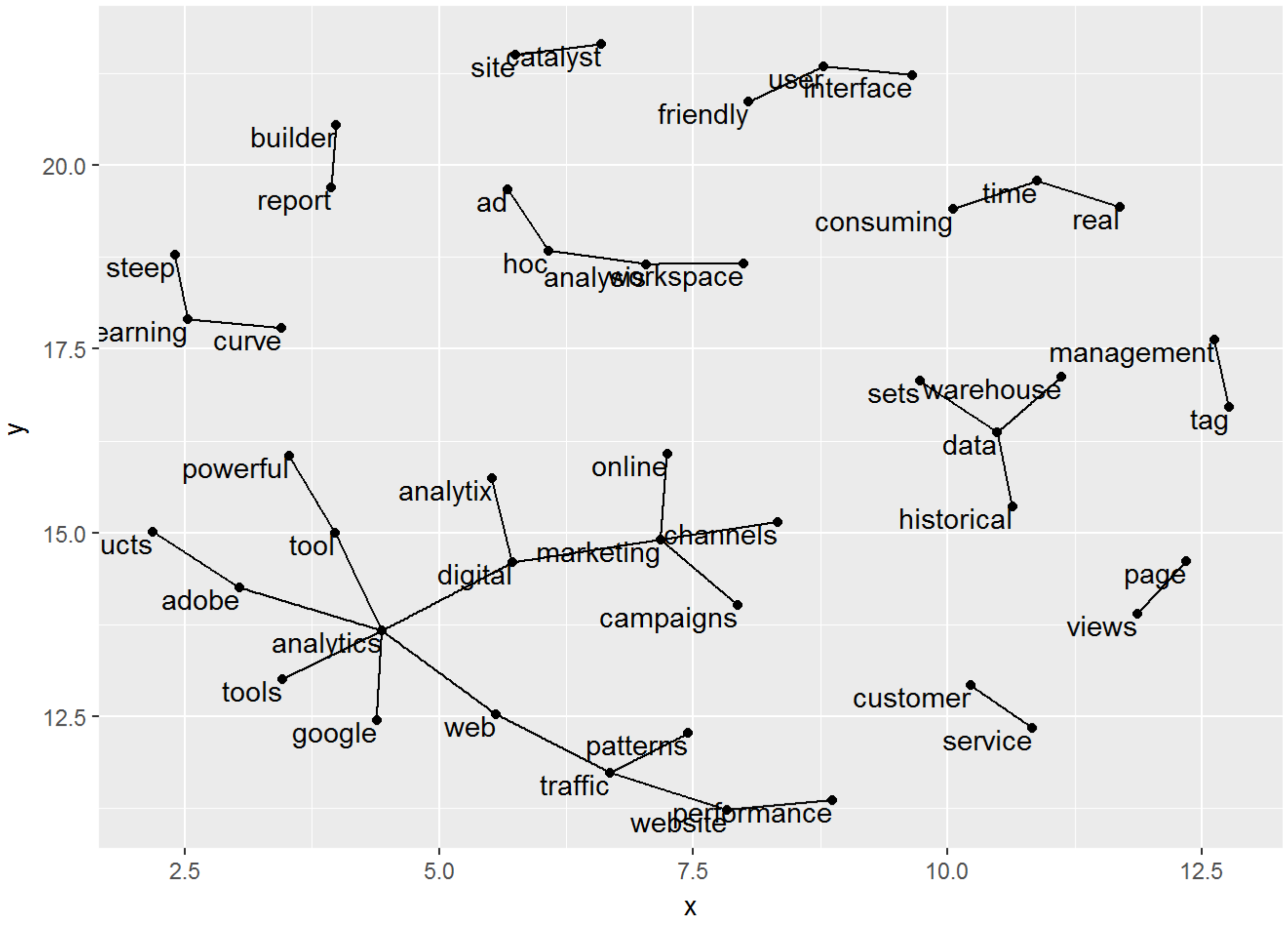
```
## IGRAPH DN-- 45 34 --
## + attr: name (v/c), n (e/n)
## + edges (vertex names):
##  [1] adobe     ->analytics  google    ->analytics  learning  ->curve
##  [4] user      ->friendly   web       ->analytics  ad        ->hoc
##  [7] real      ->time       hoc       ->analysis   digital   ->analytics
## [10] report    ->builder    adobe     ->products   data      ->warehouse
## [13] powerful  ->tool       analytics ->tools      marketing ->campaigns
## [16] analysis  ->workspace  tag       ->management website   ->traffic
## [19] analytics ->tool       digital   ->marketing  historical->data
## [22] site      ->catalyst   steep     ->learning   customer  ->service
## + ... omitted several edges
```

```
library(ggraph) ## Loading the package ggraph
```

```
## Warning: package 'ggraph' was built under R version 3.4.1
```

```
set.seed(13)

ggraph(bigram_graph,layout = "fr") +
  geom_edge_link() +
  geom_node_point() +
  geom_node_text(aes(label = name,vjust = 1, hjust = 1)) ## Graphing word connections
```


site
catalyst
user
interface
friendly
builder
report
ad
time
consuming
real
hoc
analysis
workspace
steep
learning
curve
management
sets
warehouse
tag
data
y
powerful
online
analytix
historical
ucts
tool
channels
marketing
digital
page
adobe
campaigns
views
analytics
tools
customer
google
web
service
patterns
traffic
website
performance
x