R Notebook

#loading libraries  
library(dplyr)

##   
## 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

library(tidytext)  
library(tidyverse)

## -- Attaching packages ------------------------------ tidyverse 1.3.0 --

## <U+2713> ggplot2 3.2.1 <U+2713> purrr 0.3.3  
## <U+2713> tibble 2.1.3 <U+2713> stringr 1.4.0  
## <U+2713> tidyr 1.0.0 <U+2713> forcats 0.4.0  
## <U+2713> readr 1.3.1

## -- Conflicts --------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(twitteR)

##   
## Attaching package: 'twitteR'

## The following objects are masked from 'package:dplyr':  
##   
## id, location

library(tm)

## Loading required package: NLP

##   
## Attaching package: 'NLP'

## The following object is masked from 'package:ggplot2':  
##   
## annotate

#### To get your consumerKey and consumerSecret see the twitteR documentation for instructions

setup\_twitter\_oauth(consumer\_key, consumer\_secret,  
 access\_token=access\_token, access\_secret=access\_secret)

## [1] "Using direct authentication"

tesla <- twitteR::searchTwitter('#Tesla', n = 1000, lang = 'en', since = '2015-06-01', retryOnRateLimit = 1e3)  
d = twitteR::twListToDF(tesla)  
  
ford <- twitteR::searchTwitter('#Ford', n = 1000, lang = 'en', since = '2015-06-01', retryOnRateLimit = 1e3)  
e = twitteR::twListToDF(ford)  
  
mercedes <- twitteR::searchTwitter('#Mercedes', n = 1000, lang = 'en', since = '2015-06-01', retryOnRateLimit = 1e3)  
a = twitteR::twListToDF(mercedes)

#### Cleaning the datasets

##### Remove http and https elements manually

d$text <- gsub("http[^[:space:]]\*","", d$text) # For http  
d$text <- gsub("http[^[:space:]]\*","", d$text) # For https  
  
e$text <- gsub("http[^[:space:]]\*","", e$text) # For http  
e$text <- gsub("http[^[:space:]]\*","", e$text) # For https  
  
a$text <- gsub("http[^[:space:]]\*","", a$text) # For http  
a$text <- gsub("http[^[:space:]]\*","", a$text) # For https

#### Tokenizing all 3 datasets from twitter

tidy\_tesla <- d %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words)

## Joining, by = "word"

tidy\_ford <- e %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words)

## Joining, by = "word"

tidy\_mercedes <- a %>%  
 unnest\_tokens(word, text) %>%  
 anti\_join(stop\_words)

## Joining, by = "word"

#### Combining all 3 tidy data frames and creating correlograms

library(tidyr)  
frequency <- bind\_rows(mutate(tidy\_tesla, author="Tesla"),  
 mutate(tidy\_ford, author= "Ford"),  
 mutate(tidy\_mercedes, author="Mercedes")) %>% #closing bind\_rows  
 mutate(word=str\_extract(word, "[a-z']+")) %>%  
 count(author, word) %>%  
 group\_by(author) %>%  
 mutate(proportion = n/sum(n))%>%  
 select(-n) %>%  
 spread(author, proportion) %>%  
 gather(author, proportion, `Ford`, `Mercedes`)

#### Let’s plot the correlograms:

library(scales)

##   
## Attaching package: 'scales'

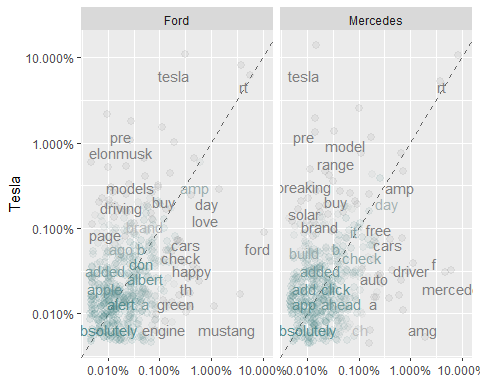
## The following object is masked from 'package:purrr':  
##   
## discard

## The following object is masked from 'package:readr':  
##   
## col\_factor

ggplot(frequency, aes(x=proportion, y=`Tesla`,   
 color = abs(`Tesla`- proportion)))+  
 geom\_abline(color="grey40", lty=2)+  
 geom\_jitter(alpha=.1, size=2.5, width=0.3, height=0.3)+  
 geom\_text(aes(label=word), check\_overlap = TRUE, vjust=1.5) +  
 scale\_x\_log10(labels = percent\_format())+  
 scale\_y\_log10(labels= percent\_format())+  
 scale\_color\_gradient(limits = c(0,0.001), low = "darkslategray4", high = "gray75")+  
 facet\_wrap(~author, ncol=2)+  
 theme(legend.position = "none")+  
 labs(y= "Tesla", x=NULL)

## Warning: Removed 10249 rows containing missing values (geom\_point).

## Warning: Removed 10251 rows containing missing values (geom\_text).

 #### Taking a look at correlation coefficients

cor.test(data=frequency[frequency$author == "Ford",],  
 ~proportion + `Tesla`)

##   
## Pearson's product-moment correlation  
##   
## data: proportion and Tesla  
## t = 12.445, df = 504, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.4152118 0.5488076  
## sample estimates:  
## cor   
## 0.4848327

cor.test(data=frequency[frequency$author == "Mercedes",],  
 ~proportion + `Tesla`)

##   
## Pearson's product-moment correlation  
##   
## data: proportion and Tesla  
## t = 10.749, df = 443, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.3778498 0.5255673  
## sample estimates:  
## cor   
## 0.4548312

#### Sentiment analysis

library(textdata)  
library(tidytext)  
get\_sentiments('afinn') # Show example of the table

## # A tibble: 2,477 x 2  
## word value  
## <chr> <dbl>  
## 1 abandon -2  
## 2 abandoned -2  
## 3 abandons -2  
## 4 abducted -2  
## 5 abduction -2  
## 6 abductions -2  
## 7 abhor -3  
## 8 abhorred -3  
## 9 abhorrent -3  
## 10 abhors -3  
## # … with 2,467 more rows

#### Pulling in sentiment for these 3 tokenized datasets

tidy\_tesla %>%  
 inner\_join(get\_sentiments("afinn"))%>%  
 group\_by(id) %>% #if you remove the group\_by it will calculate sentiment for all the data  
 summarise(sentiment=sum(value)) %>%  
 mutate(method="AFINN")

## Joining, by = "word"

## # A tibble: 272 x 3  
## id sentiment method  
## <chr> <dbl> <chr>   
## 1 1228466360243474432 4 AFINN   
## 2 1228467458727649280 2 AFINN   
## 3 1228467690936778752 -4 AFINN   
## 4 1228467955777843200 -2 AFINN   
## 5 1228469480965201920 -5 AFINN   
## 6 1228470005114834944 3 AFINN   
## 7 1228470388662861826 1 AFINN   
## 8 1228470710806482947 2 AFINN   
## 9 1228470996446916608 -2 AFINN   
## 10 1228471231189401600 2 AFINN   
## # … with 262 more rows

tidy\_ford %>%  
 inner\_join(get\_sentiments("afinn"))%>%  
 group\_by(id) %>% #if you remove the group\_by it will calculate sentiment for all the data  
 summarise(sentiment=sum(value)) %>%  
 mutate(method="AFINN")

## Joining, by = "word"

## # A tibble: 489 x 3  
## id sentiment method  
## <chr> <dbl> <chr>   
## 1 1228164115643813889 4 AFINN   
## 2 1228164283487252480 4 AFINN   
## 3 1228164295189393413 1 AFINN   
## 4 1228164321554788358 6 AFINN   
## 5 1228164567655579650 6 AFINN   
## 6 1228164594687832065 -2 AFINN   
## 7 1228164599054159872 2 AFINN   
## 8 1228164918009987073 4 AFINN   
## 9 1228165235330015239 2 AFINN   
## 10 1228165589992001544 4 AFINN   
## # … with 479 more rows

tidy\_mercedes %>%  
 inner\_join(get\_sentiments("afinn"))%>%  
 group\_by(id) %>% #if you remove the group\_by it will calculate sentiment for all the data  
 summarise(sentiment=sum(value)) %>%  
 mutate(method="AFINN")

## Joining, by = "word"

## # A tibble: 425 x 3  
## id sentiment method  
## <chr> <dbl> <chr>   
## 1 1227659472593653761 0 AFINN   
## 2 1227659671374303232 2 AFINN   
## 3 1227659859757125632 3 AFINN   
## 4 1227663859676258305 2 AFINN   
## 5 1227665540757565440 -4 AFINN   
## 6 1227665625713184774 7 AFINN   
## 7 1227666678370140161 3 AFINN   
## 8 1227670011965296641 -1 AFINN   
## 9 1227670958569246720 -1 AFINN   
## 10 1227671466138669056 -3 AFINN   
## # … with 415 more rows

#### Let’s take a look at the most positive and most negative tokens in the tesla dataset

tidy\_tesla\_sentiment <- tidy\_tesla %>%  
 inner\_join(get\_sentiments("bing")) %>%  
 count(word, sentiment, sort=T)

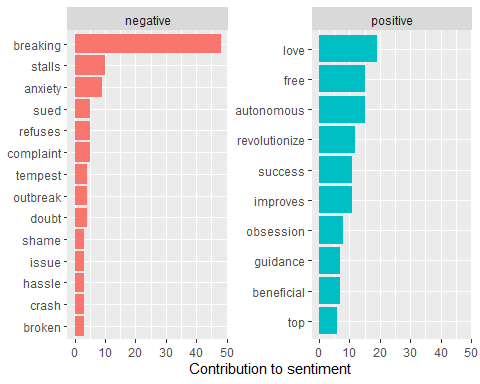
## Joining, by = "word"

print(tidy\_tesla\_sentiment)

## # A tibble: 184 x 3  
## word sentiment n  
## <chr> <chr> <int>  
## 1 breaking negative 48  
## 2 love positive 19  
## 3 autonomous positive 15  
## 4 free positive 15  
## 5 revolutionize positive 12  
## 6 improves positive 11  
## 7 success positive 11  
## 8 stalls negative 10  
## 9 anxiety negative 9  
## 10 obsession positive 8  
## # … with 174 more rows

tidy\_tesla\_sentiment %>%  
 group\_by(sentiment) %>%  
 top\_n(10) %>%  
 ungroup() %>%  
 mutate(word=reorder(word, n)) %>%  
 ggplot(aes(word, n, fill=sentiment)) +  
 geom\_col(show.legend = FALSE) +  
 facet\_wrap(~sentiment, scales = "free\_y")+  
 labs(y="Contribution to sentiment", x=NULL)+  
 coord\_flip()

## Selecting by n

 #### TF-IDF analysis

combined\_cars <- bind\_rows(mutate(d, make="Tesla"),  
 mutate(e, make= "Ford"),  
 mutate(a, make="Mercedes")  
)  
  
tesla\_modif <- combined\_cars %>%  
 unnest\_tokens(word, text) %>%  
 count(make, word, sort=TRUE) %>%  
 ungroup()  
  
tesla\_modif2 <- tesla\_modif %>%  
 group\_by(make) %>%  
 summarize(total=sum(n))  
  
tesla\_leftjoined <- left\_join(tesla\_modif, tesla\_modif2)

## Joining, by = "make"

tidy\_tesla\_tfidf <- tesla\_leftjoined %>%  
 bind\_tf\_idf(word, make, n)  
  
tidy\_tesla\_tfidf # we get all the zeors because we are looking at stop words ... too common

## # A tibble: 8,566 x 7  
## make word n total tf idf tf\_idf  
## <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 Tesla tesla 937 18580 0.0504 0 0  
## 2 Ford ford 930 17525 0.0531 0 0  
## 3 Mercedes mercedes 841 16407 0.0513 0 0  
## 4 Tesla rt 692 18580 0.0372 0 0  
## 5 Ford the 543 17525 0.0310 0 0  
## 6 Tesla to 511 18580 0.0275 0 0  
## 7 Ford rt 491 17525 0.0280 0 0  
## 8 Mercedes rt 470 16407 0.0286 0 0  
## 9 Tesla the 394 18580 0.0212 0 0  
## 10 Tesla is 379 18580 0.0204 0 0  
## # … with 8,556 more rows

tidy\_tesla\_tfidf %>%  
 arrange(desc(tf\_idf))

## # A tibble: 8,566 x 7  
## make word n total tf idf tf\_idf  
## <chr> <chr> <int> <int> <dbl> <dbl> <dbl>  
## 1 Tesla autopilot 253 18580 0.0136 1.10 0.0150   
## 2 Mercedes benz 203 16407 0.0124 1.10 0.0136   
## 3 Tesla neuralnetworks 220 18580 0.0118 1.10 0.0130   
## 4 Tesla 70,000 219 18580 0.0118 1.10 0.0129   
## 5 Tesla gpu 219 18580 0.0118 1.10 0.0129   
## 6 Tesla hours 219 18580 0.0118 1.10 0.0129   
## 7 Tesla output 219 18580 0.0118 1.10 0.0129   
## 8 Tesla tensors 215 18580 0.0116 1.10 0.0127   
## 9 Tesla evankirstel 179 18580 0.00963 1.10 0.0106   
## 10 Mercedes mercedesamgf1 149 16407 0.00908 1.10 0.00998  
## # … with 8,556 more rows

#what can we say about these words?

#### Looking at the graphical apprach:

tidy\_tesla\_tfidf %>%  
 arrange(desc(tf\_idf)) %>%  
 mutate(word=factor(word, levels=rev(unique(word)))) %>%  
 group\_by(make) %>%  
 top\_n(15) %>%  
 ungroup %>%  
 ggplot(aes(word, tf\_idf, fill=make))+  
 geom\_col(show.legend=FALSE)+  
 labs(x=NULL, y="tf-idf")+  
 facet\_wrap(~make, ncol=2, scales="free")+  
 coord\_flip()

## Selecting by tf\_idf

