Code

Twitter Text Analysis-Adobe v. Canva

Introduction, Context and Methodology

The goal of this project was to understand the differences in consumer sentiment in two leading competitors (Adobe and Canva), scraping Twitter text data will be the main source for analysis. Scraping twitter yielded approximately 10,000 tweets for the Canva document and 7,000 tweets for the Adobe document. This imbalance is important to note when comparing the results.

R's text mining library 'tm' will be used to evaluate key text metrics. The first section will evaluate the term frequency (TF) for each word in the document. The term frequency refers to the amount of times a word is used in a document. The second section will evaluate the TF-IDF of words in the two documents. This section will consider the inverse document frequency, and will reflect how important or unique a word is to a specific document over another. This will give a larger score to words which appear frequently in one company's twitter but not the other.

TF information can be applied to all facets of each business. For example, the marketing team can evaluate the popularity of a hashtag or marketing initiative. The product team can find any pain points or common questions that each product receives, finally the strategy team can determine overall consumer sentiment towards the brand. TF-IDF can be used to understand the major differences in consumer perceptions and twitter interactions between Adobe and Canva. This can yield many strategic insights and help understand each companies role in the industry and competitive advantages.

Analysis - Section 1 - Term Frequency

Load the libraries required

Hide

```
library("tm") #text mining library
```

```
Loading required package: NLP
```

Hide

```
library("SnowballC") #For reducing words to their root
```

Canva

Read in document

Hide

```
 myText <- readLines(file.choose())
```

Create a Corpus document

Hide

```
myDocument <- Corpus(VectorSource(myText))
```

Clean the document

Hide

```
myDocument <- tm_map(myDocument, content_transformer(tolower)) #Convert to lower case
```

```
Warning in tm_map.SimpleCorpus(myDocument, content_transformer(tolower)) :
  transformation drops documents
```

Hide

```
myDocument <- tm_map(myDocument, removeWords, stopwords("english")) #Remove stopwords
```

```
Warning in tm_map.SimpleCorpus(myDocument, removeWords, stopwords("english")) :
  transformation drops documents
```

Hide

```
myDocument <- tm_map(myDocument, removeNumbers) #Remove numbers
```

```
Warning in tm_map.SimpleCorpus(myDocument, removeNumbers) :
  transformation drops documents
```

Hide

```
myDocument <- tm_map(myDocument, removePunctuation) #Remove punctuation
```

```
Warning in tm_map.SimpleCorpus(myDocument, removePunctuation) :
  transformation drops documents
```

Hide

```
myDocument <- tm_map(myDocument, stemDocument) #Reduce the words to their root
```

```
Warning in tm_map.SimpleCorpus(myDocument, stemDocument) :
  transformation drops documents
```

Hide

```
myDocument <- tm_map(myDocument, stripWhitespace) #Remove unnecessary white space
```

```
Warning in tm_map.SimpleCorpus(myDocument, stripWhitespace) :
  transformation drops documents
```

Calculate term frequence and store in matrix

Hide

```
termMatrix = as.matrix(TermDocumentMatrix(myDocument))
```

Sort from high to low

Hide

```
sortedTermMatrix <- sort(rowSums(termMatrix), decreasing = TRUE) #sort in decreasing ord
er
```

Save in a dataframe

Hide

```
d <- data.frame("Term" = names(sortedTermMatrix), "Freq." = sortedTermMatrix, row.names
 = NULL) #store in data frame
print(d) #cleaner version of printing. It will still print if you type "d" but the print
(d) restricts to the meaningful part
```

Term <chr>	Freq. <dbl>
canva	9188
use	1020
thank	929
design	833
love	701
creat	578
can	513
make	421
amp	398
oneminutebrief	398

1-10 of 17,068 rows Previous **1** 2 3 4 5 6 ... 100 Next

Adobe

Read in document

Hide

```
 myTextAdobe <- readLines(file.choose())
```

```
Warning in readLines(file.choose()) :
  incomplete final line found on '/Users/justinvoronoff/Desktop/5th Year/1st Sem/Ivey/Co
mpeting with Analytics/Class Deliverables/Komo's Text Analysis/Adobe Tweets.txt'
```

Create a Corpus document

Hide

```
myDocumentAdobe <- Corpus(VectorSource(myTextAdobe))
```

Clean the text

Hide

```
myDocumentAdobe <- tm_map(myDocumentAdobe, content_transformer(tolower)) #Convert to low
er case
```

```
Warning in tm_map.SimpleCorpus(myDocumentAdobe, content_transformer(tolower)) :
  transformation drops documents
```

Hide

```
myDocumentAdobe <- tm_map(myDocumentAdobe, removeWords, stopwords("english")) #Remove st
opwords
```

```
Warning in tm_map.SimpleCorpus(myDocumentAdobe, removeWords, stopwords("english")) :
  transformation drops documents
```

Hide

```
myDocumentAdobe <- tm_map(myDocumentAdobe, removeNumbers) #Remove numbers
```

```
Warning in tm_map.SimpleCorpus(myDocumentAdobe, removeNumbers) :
  transformation drops documents
```

Hide

```
myDocumentAdobe <- tm_map(myDocumentAdobe, removePunctuation) #Remove punctuation
```

```
Warning in tm_map.SimpleCorpus(myDocumentAdobe, removePunctuation) :
  transformation drops documents
```

Hide

```
myDocumentAdobe <- tm_map(myDocumentAdobe, stemDocument) #Reduce the words to their root
```

```
Warning in tm_map.SimpleCorpus(myDocumentAdobe, stemDocument) :
  transformation drops documents
```

Hide

```
myDocumentAdobe <- tm_map(myDocumentAdobe, stripWhitespace) #Remove unnecessary white sp
ace
```

```
Warning in tm_map.SimpleCorpus(myDocumentAdobe, stripWhitespace) :
  transformation drops documents
```

Calculate term frequence and store in matrix

Hide

```
termMatrixAdobe = as.matrix(TermDocumentMatrix(myDocumentAdobe))
```

Sort from high to low

Hide

```
sortedTermMatrixAdobe <- sort(rowSums(termMatrixAdobe), decreasing = TRUE) #sort in decr
easing order
```

Save in a data frame and compare

Hide

```
dAdobe <- data.frame("Term" = names(sortedTermMatrixAdobe), "Freq." = sortedTermMatrixAd
obe, row.names = NULL) #store in data frame

#compare
print(dAdobe) #adobe
```

Term	Freq.
<chr>	<dbl>
adobeccexpress	3300
adobeforedu	706
tanyaavrith	591
creativecloudexpress	586
via	532
adobeeducr	507
use	424
love	398
creativ	381
adob	369

1-10 of 7,801 rows Previous **1** 2 3 4 5 6 ... 100 Next

Analysis - Section 2 - TF-IDF

Create a corpus (document) of the two data sets

Hide

```
myDocumentCombined <- Corpus(VectorSource(c(myDocument, myDocumentAdobe)))
myDocumentCombined
```

```
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 6
```

Hide

```
myDocumentCombined <- tm_map(myDocumentCombined, content_transformer(tolower)) #Convert
 to lower case
```

```
Warning in tm_map.SimpleCorpus(myDocumentCombined, content_transformer(tolower)) :
  transformation drops documents
```

Hide

```
myDocumentCombined <- tm_map(myDocumentCombined, removeWords, stopwords("english")) #Rem
ove stopwords
```

```
Warning in tm_map.SimpleCorpus(myDocumentCombined, removeWords, stopwords("english")) :
  transformation drops documents
```

Hide

```
myDocumentCombined <- tm_map(myDocumentCombined, removeNumbers) #Remove numbers
```

```
Warning in tm_map.SimpleCorpus(myDocumentCombined, removeNumbers) :
  transformation drops documents
```

Hide

```
myDocumentCombined <- tm_map(myDocumentCombined, removePunctuation) #Remove punctuation
```

```
Warning in tm_map.SimpleCorpus(myDocumentCombined, removePunctuation) :
  transformation drops documents
```

Hide

```
myDocumentCombined <- tm_map(myDocumentCombined, stemDocument) #Reduce the words to thei
r root
```

```
Warning in tm_map.SimpleCorpus(myDocumentCombined, stemDocument) :
  transformation drops documents
```

Hide

```
myDocumentCombined <- tm_map(myDocumentCombined, stripWhitespace) #Remove unnecessary wh
ite space
```

```
Warning in tm_map.SimpleCorpus(myDocumentCombined, stripWhitespace) :
  transformation drops documents
```

Hide

```
tdm = TermDocumentMatrix(myDocumentCombined, control = list(weighting = function(x) weig
htTfIdf(x, normalize = FALSE)))
```

```
Warning in TermDocumentMatrix.SimpleCorpus(myDocumentCombined, control = list(weighting
= function(x) weightTfIdf(x,  :
  custom functions are ignored
```

Store in matrix

Hide

```
termMatrix = as.matrix(tdm)
tail(termMatrix)
```

```
                      Docs
Terms                  1 2 3        4 5 6
  ùšùuø¹ù„             0 0 0 2.584963 0 0
  ù…ø§ø°ø§             0 0 0 2.584963 0 0
  ù…ø§ùšùƒø±ù^ø³ù^ùuøª 0 0 0 5.169925 0 0
  ù…ø®ø§ø¨ø±ø§øªùš      0 0 0 2.584963 0 0
  ù…ø¹ù„ù…             0 0 0 5.169925 0 0
  ù…ùšø¯ùšø§          0 0 0 2.584963 0 0
```

Split them

Hide

```
canvaMatrix <- termMatrix[,1] #all rows, column 1
adobeMatrix <- termMatrix[,4] #all rows, column 2
```

Sort them

Hide

```
sortedcanvaMatrix <- sort((canvaMatrix), decreasing = TRUE)
sortedadobeMatrix <- sort((adobeMatrix), decreasing = TRUE)
head(sortedcanvaMatrix)
```

```
          canva             use           thank          design            love oneminutebrie
f
      14562.635        1616.662        1472.430        1320.274        1111.059        1028.81
5
```

Hide

```
head(sortedadobeMatrix)
```

```
      adobeccexpress          tanyaavrith creativecloudexpress           adobeforedu
via      claudiozavalajr
           5230.3763            1527.7128            1514.7880             1118.9835
843.2001             819.4331
```

Sort into data frames

Hide

```
dCanva <- data.frame("Term" = names(sortedcanvaMatrix),"Freq."=sortedcanvaMatrix,
row.names = NULL) #Store as Data Frame

dAdobe <- data.frame("Term" = names(sortedadobeMatrix),"Freq."=sortedadobeMatrix,
row.names = NULL) #Store as Data Frame
```

Final TF-IDF Scores for Adobe and Canva

Hide

```
print (dCanva)
```

Term <chr>	**Freq.** <dbl>
canva	14562.63546
use	1616.66175
thank	1472.43016
design	1320.27376
love	1111.05871
oneminutebrief	1028.81508
creat	916.10833
can	813.08576
make	667.26921
amp	630.81508

1-10 of 21,916 rows Previous 1 2 3 4 5 6 ... 100 Next

Hide

```
print (dAdobe)
```

Term <chr>	Freq. <dbl>
adobeccexpress	5230.37625
tanyaavrith	1527.71284
creativecloudexpress	1514.78803
adobeforedu	1118.98353
via	843.20005
claudiozavalajr	819.43311
adobeeducr	803.57599
use	672.02410
love	630.81508
creativ	603.87071

1-10 of 21,916 rows Previous 1 2 3 4 5 6 ... 100 Next

Recommendations

Adobe’s term frequency data frame highlights consumers reference of their major product lines, such as Adobe Express and Adobe for Education. The frequent term “love” highlights that consumers generally enjoy using these products. However, “challenge” is also a frequent word in Adobe’s tweets. These tweets should be segmented and evaluated further to understand which parts of Adobe’s services are causing challenges for users. Adobe express is the most popular product that they have in their portfolio.The fourth most common word is “tanaavrith”. She is a core voice in Adobe’s marketing strategy for Adobe for Education, therefore this is a sign that she is reaching a significant audience and is an important face of the brand. Adobe should try to utilize Tanya’s reach to deploy more marketing initiatives similar to Cavnva that show quickly users can create with their products. From a strategic lens, Canva generally has stronger consumer sentiment with frequent words such as “love”, “thank”, and “great”. These results also highlight the popularity of Canva’s “one-minute-brief” marketing initiative where they encourage people to create exciting and creative ads to show Canva’s ease of use. Canva’s other marketing initiatives, “canvadesignchallenge’ and”remixwithcanva" are also very popular.