Intro to Natural Language Processing

www.linkedin.com/in/edwardkwartler
@tkwartler

Shameless Plug

Slated for June 2016

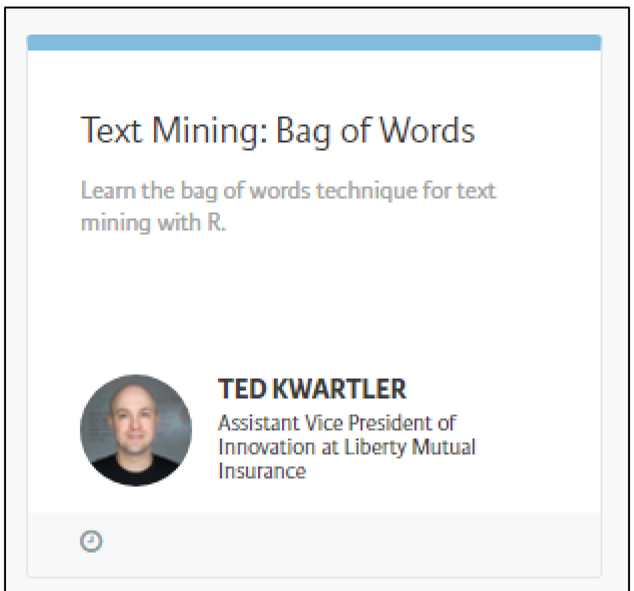
*cover art may change.

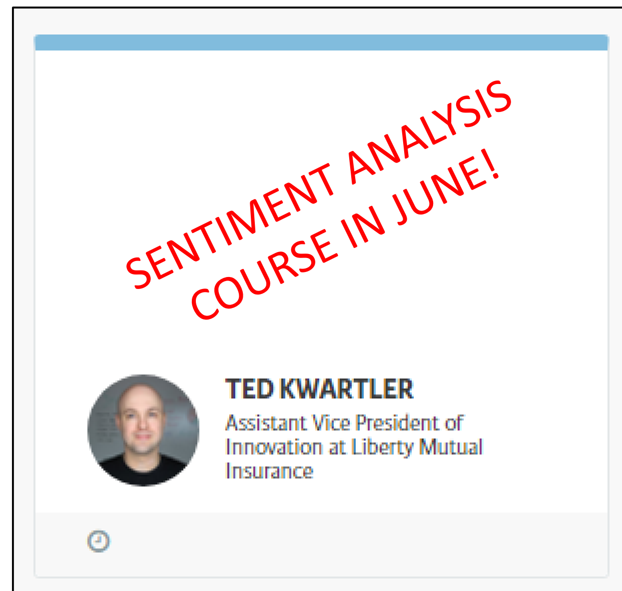
Shameless Plug #1

July!!

Shameless Plug #2

DataCamp





Agenda

What is Text Mining?

String Manipulation
Cleaning

Freq Counts,
Dendrograms
Wordclouds

Sentiment Analysis

Not enough time
Just ask for files

Topic Modeling,
OpenNLP

Agenda

What is Text Mining?

String Manipulation
Cleaning

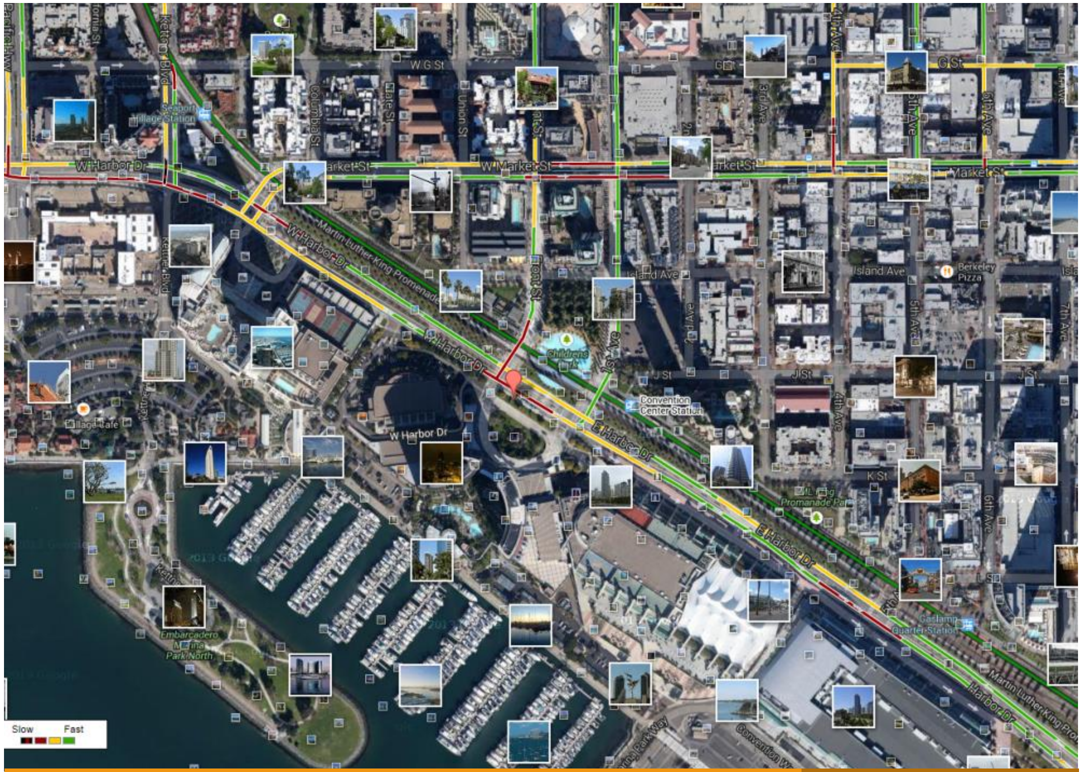
Freq Counts,
Dendrograms
Wordclouds

Sentiment Analysis

Topic Modeling,
OpenNLP

What is text mining?

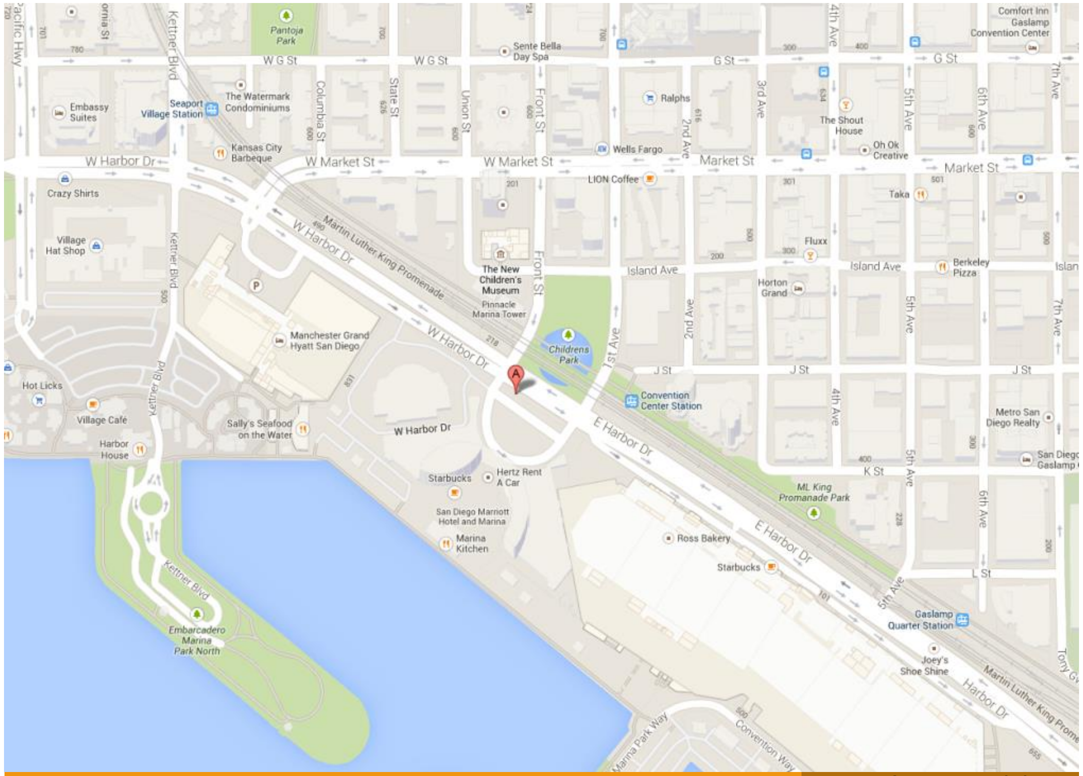
- **Extract new insights from text**
- Let’s you drink from a fire hose of information
- Language is hard; many unsolved problems
    - Unstructured
    - Expression is individualistic
    - Multi-language/cultural implications



Before Text Mining

What is text mining?

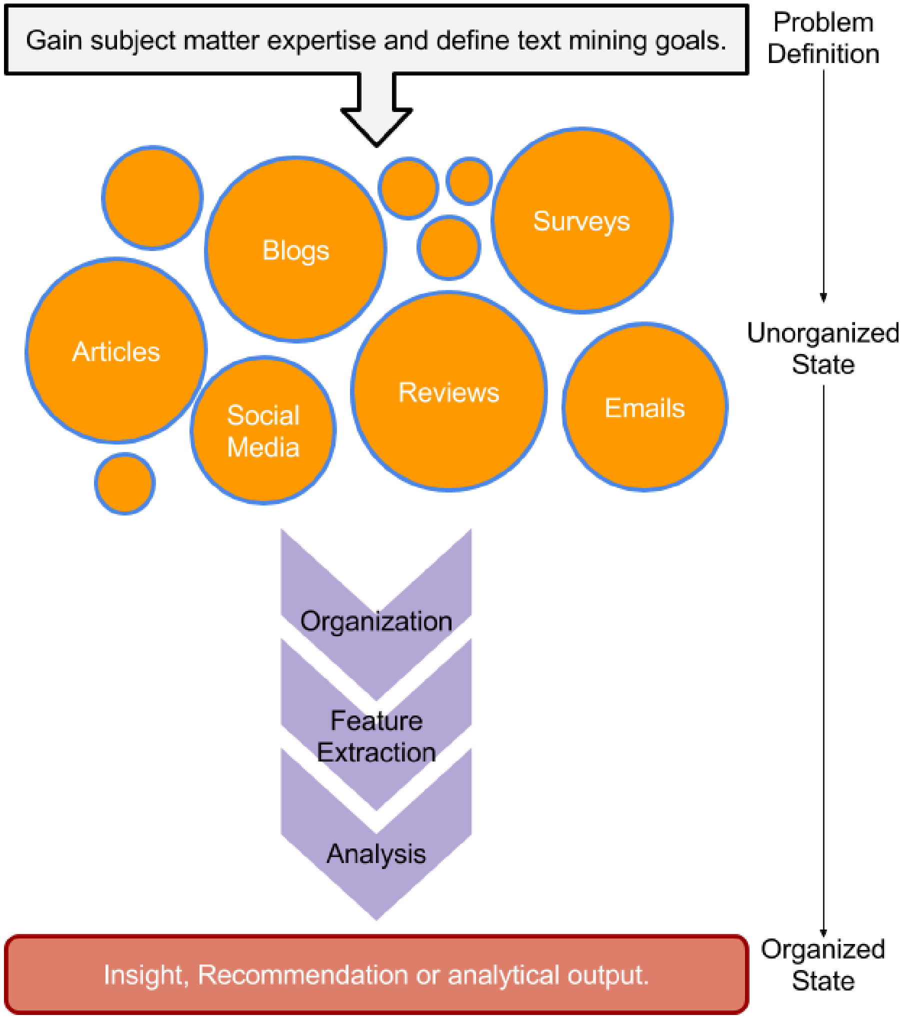
- **Extract new insights from text**
- Let's you drink from a fire hose of information
- Language is hard; many unsolved problems
  - Unstructured
  - Expression is individualistic
  - Multi-language/cultural implications



After Text Mining

What is Text Mining?

Text Mining Workflow



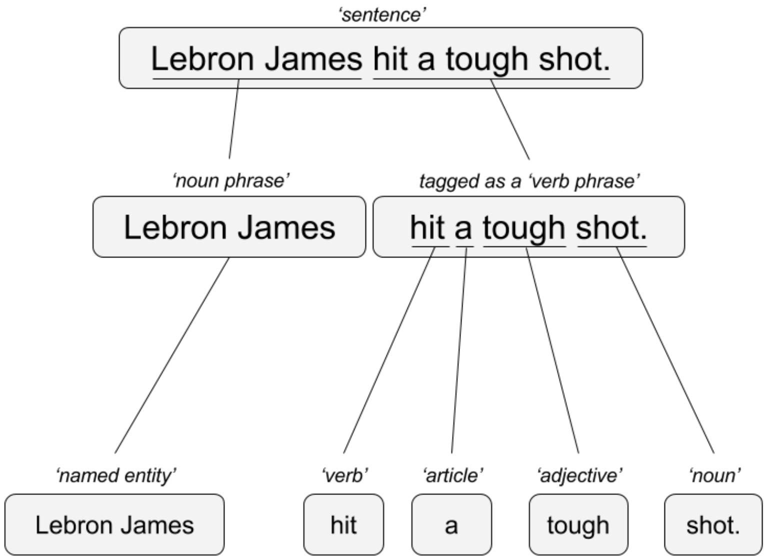
1. Problem Definition
2. Identify Text Sources
3. Text Organization
4. Feature Extraction
5. Analytics
6. Reach Insight/Recommendation

What is Text Mining?

Text Mining Approaches

"Lebron James hit a tough shot."

Semantic Using Syntactic Parsing



Bag of Words

What is Text Mining?

Text Mining Approaches

Some Challenges in Text Mining

- Compound words (tokenization) changes meaning
  - "not bad" versus "bad"
- Disambiguation
- Sarcasm
  - "I like it…NOT!"
- Cultural differences
  - "It's wicked good" (in Boston)

<u>"I made her duck."</u>

- I cooked waterfowl to eat.
- I cooked waterfowl belonging to her.
- I created the (clay?) duck and gave it to her.
- Duck!!

Text Sources

Text can be captured within the enterprise and elsewhere

- Books
- Electronic Docs (PDFs)
- Blogs
- Websites
- Social Media
- Customer Records
- Customer Service Notes
- Notes
- Emails
- Legal Documents
- …

The source and context of the medium is important. It will have a lot of impact on difficulty and data integrity.

Enough of me talking…let's do it for real!

Scripts in this workshop follow a simple workflow

1. Set the Working Directory
2. Load Libraries
3. Make Custom Functions & Specify Options
4. Read in Data & Pre-Process
5. Perform Analysis & Save

Enough of me talking…let's do it for real!

Setup

Install R/R Studio

- http://cran.us.r-project.org/
- http://www.rstudio.com/products/rstudio/download/

Workshop scripts, corpora

(prob best to download at the end)

https://github.com/kwartler/ODSC_Workshop

Install Packages

- Run "1_Install_Packages.R" script
  - An error may occur if the Java version doesn't match and depending on the OS. In that case install packages individually.

What is Text Mining?

Warning: Twitter Profanity

- Twitter demographics skew young and as a result have profanity that appear in the examples.
- **It’s the easiest place to get a lot of messy text fast, if it is offensive feel free to talk to me and I will work to get you other texts for use on your own. No offense is intended.**

Agenda

- What is Text Mining?
- String Manipulation Cleaning
- Freq Counts, Dendrograms Wordclouds
- Sentiment Analysis
- Topic Modeling, OpenNLP

Open the “coffee.csv” to get familiar with the data structure

1000 tweets mentioning “coffee”

	text	favorited	replyToSN	created	truncated	replyToSID	id	replyToUID	statusSource	screenName	retweetCour	retweeted	longitude	latitude
1	@ayyytylerb that is so true drink lots of coffee	FALSE	ayyytylerb	8/9/13 2:43	FALSE	3.6566E+17	3.6566E+17	1637123977	<a href="htt	thejennagibs	0	FALSE	NA	NA
2	RT @bryzy_brib: Senior March tmw morning at 7:25 A.M. in the SENIOR lot. Get up early, make yo coffee/breakfast, cus this will only happen ...	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	carolynicosia	1	FALSE	NA	NA
3	If you believe in #gunsense tomorrow would be a very good day to have your coffee any place BUT @Starbucks Guns+Coffee=#nosense @MomsDemand	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	web	janeCkay	0	FALSE	NA	NA
4	My cute coffee mug. http://t.co/2udvMU6XIG	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	AlexandriaO(	0	FALSE	NA	NA
5	RT @slaredo21: I wish we had Starbucks here... Cause coffee dates in the morning sound perff!	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	Rooosssaaaa	2	FALSE	NA	NA
6	Does anyone ever get a cup of coffee before a cocktail??	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	E_Z_MAC	0	FALSE	NA	NA
7	"I like my coffee like I like my women...black, bitter, and preferably fair trade." I love #Archer	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	Charlie_311¢	0	FALSE	NA	NA
8	@dreamwwediva ya didn't have coffee did ya?	FALSE	dreamwwed	8/9/13 2:43	FALSE	3.6566E+17	3.6566E+17	1316942208	<a href="htt	JessicaSalvat	0	FALSE	NA	NA
9	RT @iDougherty42: I just want some coffee.	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	kaytiekirk	1	FALSE	NA	NA
10	RT @Dorkv76: I can't care before coffee.	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	lissteria	2	FALSE	NA	NA
11	No lie I wouldn't mind coming home smelling like coffee	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	DOPECROOK	0	FALSE	NA	NA
12	RT @JonasWorldFeed: Play Ping Pong with Joe. Take a tour of the stage with Nick. Have coffee with Kevin. Charity auction: https://t.co/VTkK...	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	<a href="htt	TiffCaruso	6	FALSE	NA	NA
13	Have I ever told any of you that Tate Donovan bought my stepmom coffee?	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	web	CurlysCrazyN	0	FALSE	NA	NA
14	RT @JonasWorldFeed: Play Ping Pong with Joe. Take a tour of the stage with Nick. Have coffee with Kevin. Charity auction: https://t.co/VTkK...	FALSE	NA	8/9/13 2:43	FALSE	NA	3.6566E+17	NA	web	JoeJonasVA	6	FALSE	NA	NA
15	@HeatherWhaley I was about 2 joke it takes 2 hands to hold hot coffee...then I read headline! #Don'tDrinkNShoot	FALSE	HeatherWha	8/9/13 2:42	FALSE	3.6565E+17	3.6566E+17	26035764	<a href="htt	AnnaDuleep	0	FALSE	NA	NA
16	RT @MoveTheSticks: Charlie Whitehurst looks like he should be working at a coffee shop in Portland or hosting a renovation show on HGTV.	FALSE	NA	8/9/13 2:42	FALSE	NA	3.6566E+17	NA	<a href="htt	mpr4437	42	FALSE	NA	NA
17	Coffee always makes everything better.	FALSE	NA	8/9/13 2:42	FALSE	NA	3.6566E+17	NA	web	sharkshukri	0	FALSE	NA	NA
18	RT @AdelaideReview: Food For Thought: @Annabelleats shares a delicious Venison and Porcini Mushroom Pie Recipe. http://t.co/N8O7vqFKWN http...	FALSE	NA	8/9/13 2:42	FALSE	NA	3.6566E+17	NA	<a href="htt	thepaulbaker	1	FALSE	NA	NA
19	RT @LittleMelss: lmfao!!!"@bryanlaca: nahhh Melanie u is fa sho like an ummm a Coffee table ;)) yeeeee lmaoo"	FALSE	NA	8/9/13 2:42	FALSE	NA	3.6566E+17	NA	web	bryanlaca	1	FALSE	NA	NA
20	I wonder if Christian Colon will get a cup of coffee once the rosters expand to 40 man in September. Really nothing to lose by doing so.	FALSE	NA	8/9/13 2:42	FALSE	NA	3.6566E+17	NA	<a href="htt	Shauncore	0	FALSE	NA	NA

“text$text”
is the vector of tweets that we are interested in.

All other attributes are automatically returned from the twitter API

Keyword scanning, Cleaning & Freq Counts

2_Keyword_Scanning.R

Basic R Unix Commands

grepl returns a vector of T/F if the pattern is present at least once

```
grepl("pattern", searchable object, ignore.case=TRUE)
```

```
 [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[18] FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

grep returns the position of the pattern in the document

```
grep("pattern", searchable object, ignore.case=TRUE)
```

```
[1]   4 214 276 366 479 534 549 620
```

"library(stringi)" Functions

stri_count counts the number of patterns in a document

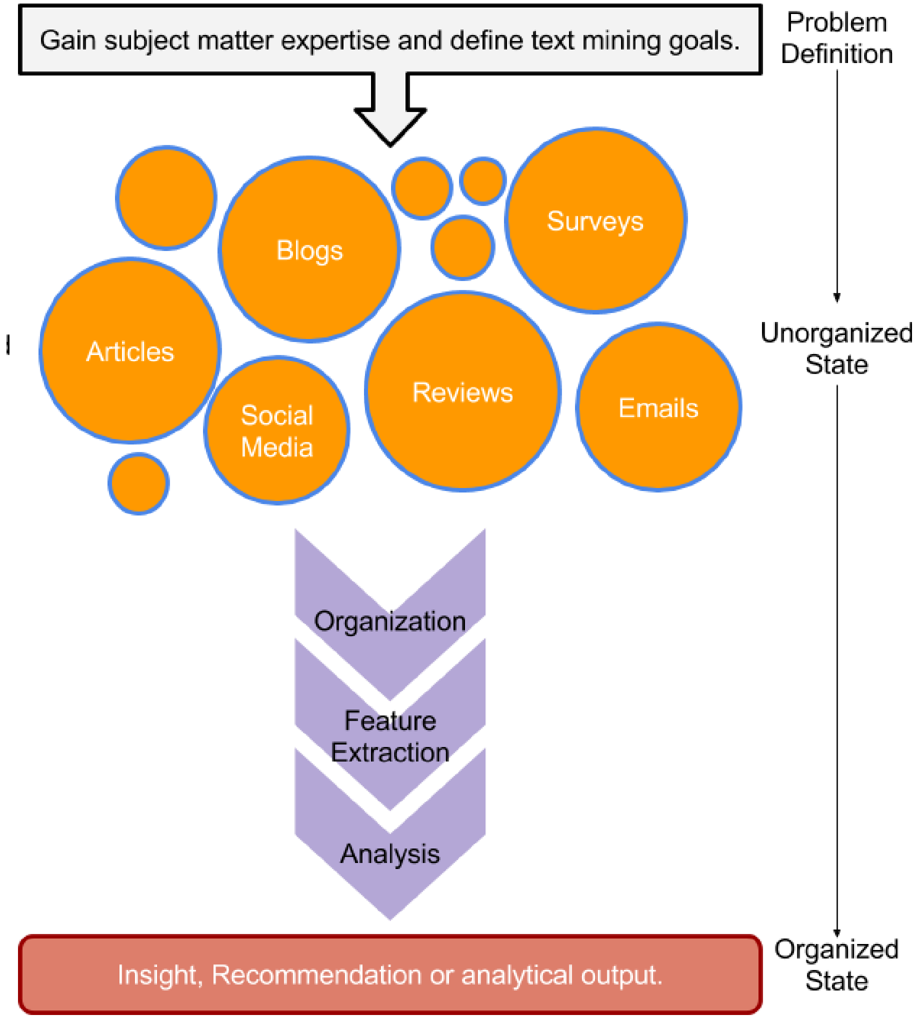
```
stri_count(searchable object, fixed="pattern")
```

```
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[54] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
```

Keyword scanning, Cleaning & Freq Counts

2_String Manipulation.R

Remember This?



What is Text Mining?

R for our Cleaning Steps

Tomorrow I'm going to have a nice glass of Chardonnay and wind down with a good book in the corner of the county :-)

1. Remove Punctuation
2. Remove extra white space
3. Remove Numbers
4. Make Lower Case
5. Remove “stop” words

tomorrow going nice glass chardonnay wind down good book corner county

Keyword scanning, Cleaning & Freq Counts

3_Cleaning and Frequency Count.R

"library(tm)" Functions

VCorpus creates a corpus held in memory.

VCorpus(source)

tm_map applies the transformations for the cleaning

tm_map(corpus, function)

getTransformations() will list all standard tm corpus transformations
We can apply standard R ones too. Sometimes it makes sense to perform all of these or a subset or even other transformations not listed like "stemming"

tm_map(corpus, removePunctuation) - removes the punctuation from the documents
tm_map(corpus, stripWhitespace) - extra spaces, tabs are removed
tm_map(corpus, removeNumbers) - removes numbers
tm_map(corpus, content_transformer(tolower)) – makes all case lower
tm_map(corpus, removeWords) - removes specific "stopwords"

New Text Mining Concepts

Corpus- A collection of documents that analysis will be based on.
Stopwords – are common words that provide very little insight, often articles like "a", "the". Customizing them is sometimes key in order to extract valuable insights.

Keyword scanning, Cleaning & Freq Counts

3_Cleaning and Frequency Count.R

"library(qdap)" Functions

Multiple Global Substitution

mgsub("search pattern", "replacement pattern", text object)

Family of Replace Functions

replace_abbreviation()- Replace Abbreviations
replace_contraction()- Replace Contractions
replace_number()- Replace Numbers With Text Representation
replace_ordinal()- Replace Mixed Ordinal Numbers With Text Representation
replace_symbol()- Replace Symbols with Word Equivalents

To use on a corpus you need to apply content_transformer

tm_map(corpus, content_transformer(replace_abbreviation))

New Text Mining Concepts

Lemmatization in linguistics, is the process of grouping together the different inflected forms of a word so they can be analyzed as a single item.

Keyword scanning, Cleaning & Freq Counts

Poor Man's lemmatization

library(lexicon)

```
> hash_lemmas
             token     lemma
    1:   furtherst   further
    2:     skilled     skill
    3:      'cause   because
    4:          'd     would
    5:         'em      them
   ---
41529:        zoos       zoo
41530:   zoospores  zoospore
41531:   zucchinis  zucchini
41532:       zulus      zulu
41533:     zygotes    zygote
```

Qdap's mgsub() lets you easily aggregate words

```
#Poor Man's Lemmatization
data(hash_lemmas)
text$text <- mgsub(hash_lemmas$token,hash_lemmas$lemma,text$text)
```

Warning, takes awhile

3_Cleaning and Frequency Counts.R

3_Cleaning and Frequency Count.R

“tryTolower”is poached to account for errors when making lowercase.

```
tryTolower <- function(x){
 # return NA when there is an error
 y = NA
 # tryCatch error
 try_error = tryCatch(tolower(x), error = function(e) e)
 # if not an error
 if (!inherits(try_error, 'error'))
  y = tolower(x)
 return(y)}
```

“clean.corpus” makes applying all transformations easier.

```
clean.corpus<-function(corpus){
 corpus <- tm_map(corpus, removePunctuation)
 corpus <- tm_map(corpus, stripWhitespace)
 corpus <- tm_map(corpus, removeNumbers)
 corpus <- tm_map(corpus, content_transformer(str_to_lower))
 corpus <- tm_map(corpus,
content_transformer(replace_contraction))
 corpus <- tm_map(corpus, removeWords, custom.stopwords)
 return(corpus)}
```

Base: tolower (basic)
Stringr: str_to_lower (wrapper)
Custom: tryTolower (handles errors)

Keyword scanning, Cleaning & Freq Counts

3_Cleaning and Frequency Count.R

"custom.stopwords" combines vectors of words to remove from the corpus

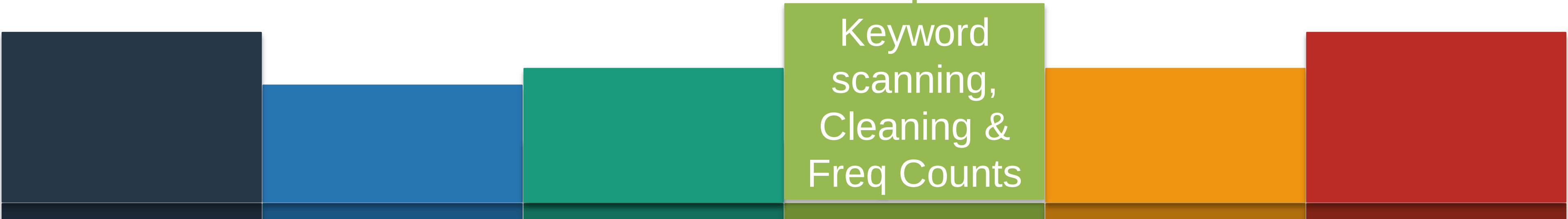
```
#Create custom stop words
custom.stopwords <- c(stopwords('english'), 'lol', 'smh')
```

Add channel specific stop words. E.g. Twitter abbreviations

"custom.reader" keeps the meta data (tweet ID) with the original document

```
#bring in some text
text<-read.csv('coffee.csv', header=TRUE)

#Keep the meta data, apply the functions to make a clean corpus
custom.reader <- readTabular(mapping=list(content="text", id="id"))
corpus <- VCorpus(DataframeSource(text), readerControl=list(reader=custom.reader))
corpus<-clean.corpus(corpus)
```



3_Cleaning and Frequency Count.R

Bag of Words means creating a Term Document Matrix or Document Term Matrix*

Term Document Matrix

	Tweet1	Tweet 2	Tweet3	Tweet4	…	Tweet_n
Term1	0	0	0	0	0	0
Term2	1	1	0	0	0	0
Term3	1	0	0	2	0	0
…	0	0	3	0	1	1
Term_n	0	0	0	1	1	0

Document Term Matrix

	Term1	Term2	Term3	…	Term_n
Tweet1	0	1	1	0	0
Tweet2	0	1	0	0	0
Tweet3	0	0	0	3	0
…	0	0	0	1	1
Tweet_n	0	0	0	1	0

"as.matrix" makes the tm's version of a matrix into a simpler version

```
dtm<-DocumentTermMatrix(corpus)
tdm<-TermDocumentMatrix(corpus)
dtm.tweets.m<-as.matrix(dtm)
tdm.tweets.m<-as.matrix(tdm)
```

These matrices are often very sparse and large therefore some special steps may be needed and will be covered in subsequent scripts.

*Depends on analysis, both are transpositions of the other

Keyword scanning, Cleaning & Freq Counts

Agenda

- What is Text Mining?
- String Manipulation Cleaning
- Freq Counts, Dendrograms Wordclouds
- Sentiment Analysis
- Topic Modeling, OpenNLP

4_dendrogram.R script builds on the matrices

First let's explore simple frequencies

```
#Summed Vector
tdm.m <- as.matrix(tdm)
tdm.v <- sort(rowSums(tdm.m),decreasing=TRUE)
tdm.df <- data.frame(word = names(tdm.v),freq=tdm.v, row.names=NULL)
```

Term Document Matrix

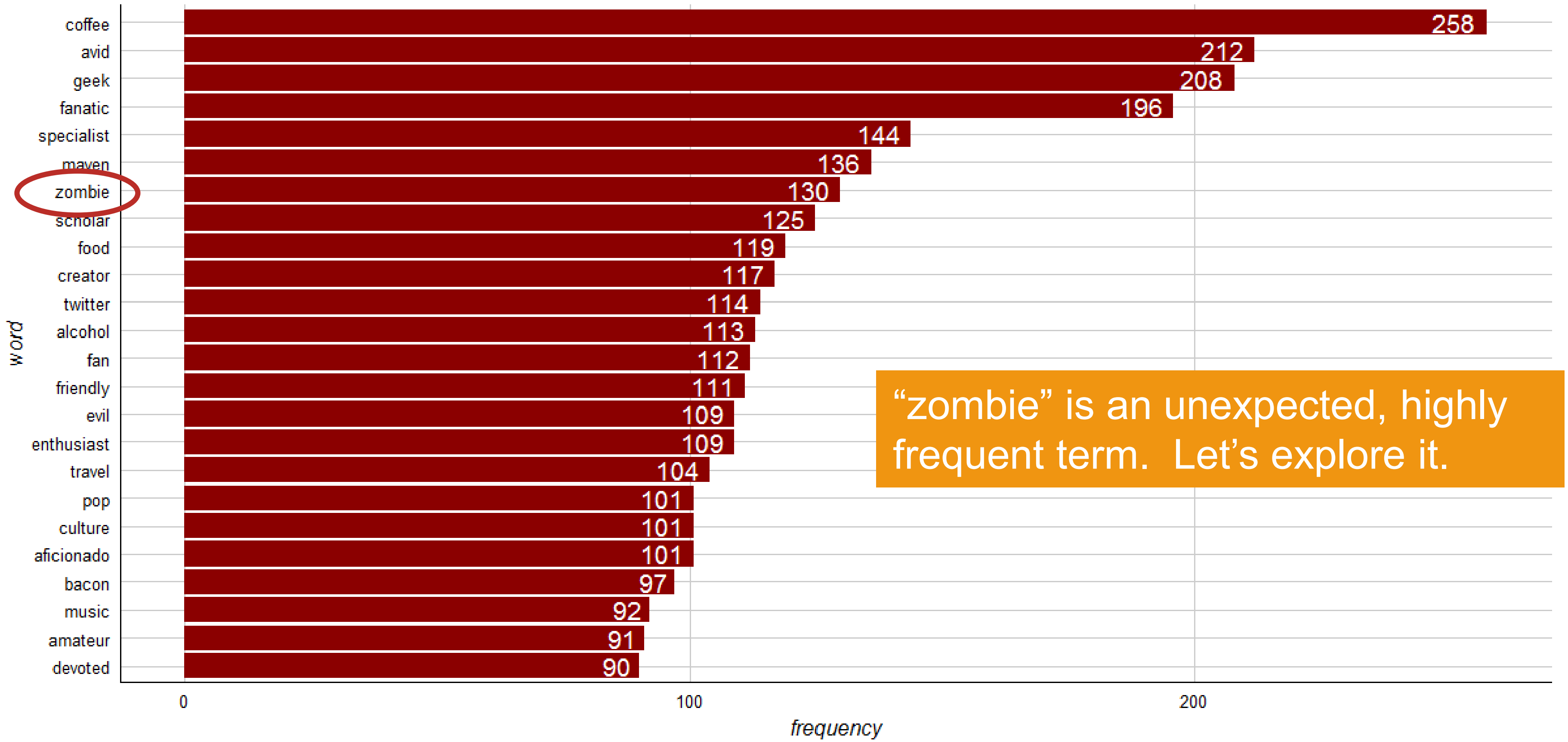
	Tweet1	Tweet 2	Tweet3	Tweet4	…	Tweet_n
Term1	0	0	0	0	0	0
Term2	1	1	0	0	0	0
Term3	1	0	0	2	0	0
…	0	0	3	0	1	1
Term_n	0	0	0	1	1	0

word	freq
Term1	0
Term2	2
Term3	3
…	5
Term_n	2

Freq Counts, Dendrograms Wordclouds

4_dendrograms.R

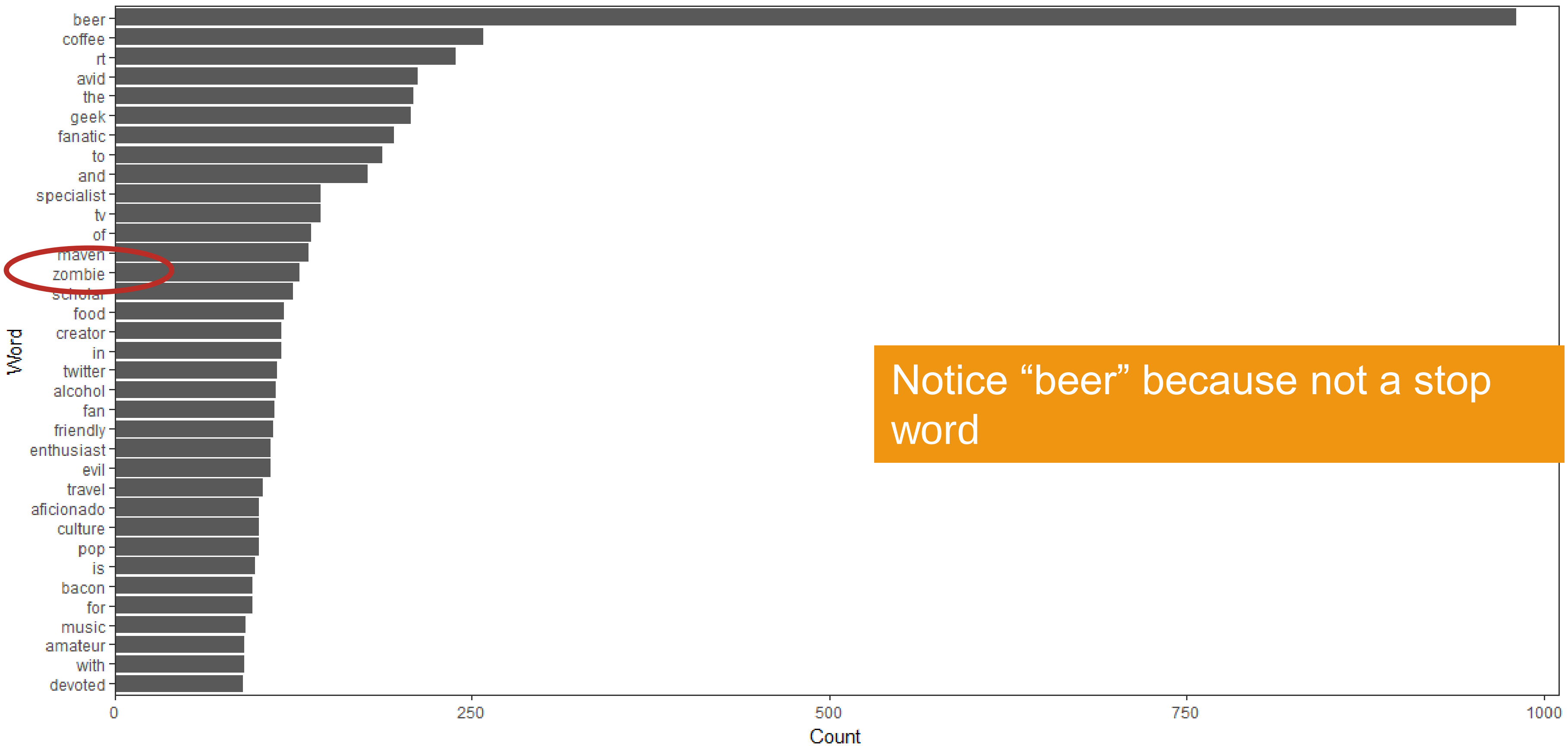
4_dendrogram.R script –ggplot2



"zombie" is an unexpected, highly frequent term. Let's explore it.

Freq Counts, Dendrograms Wordclouds

4_dendrogram.R script –qdap's freq_terms



Notice "beer" because not a stop word

Freq Counts, Dendrograms Wordclouds

4_dendrogram.R script

Next let's explore word associations, similar to correlation

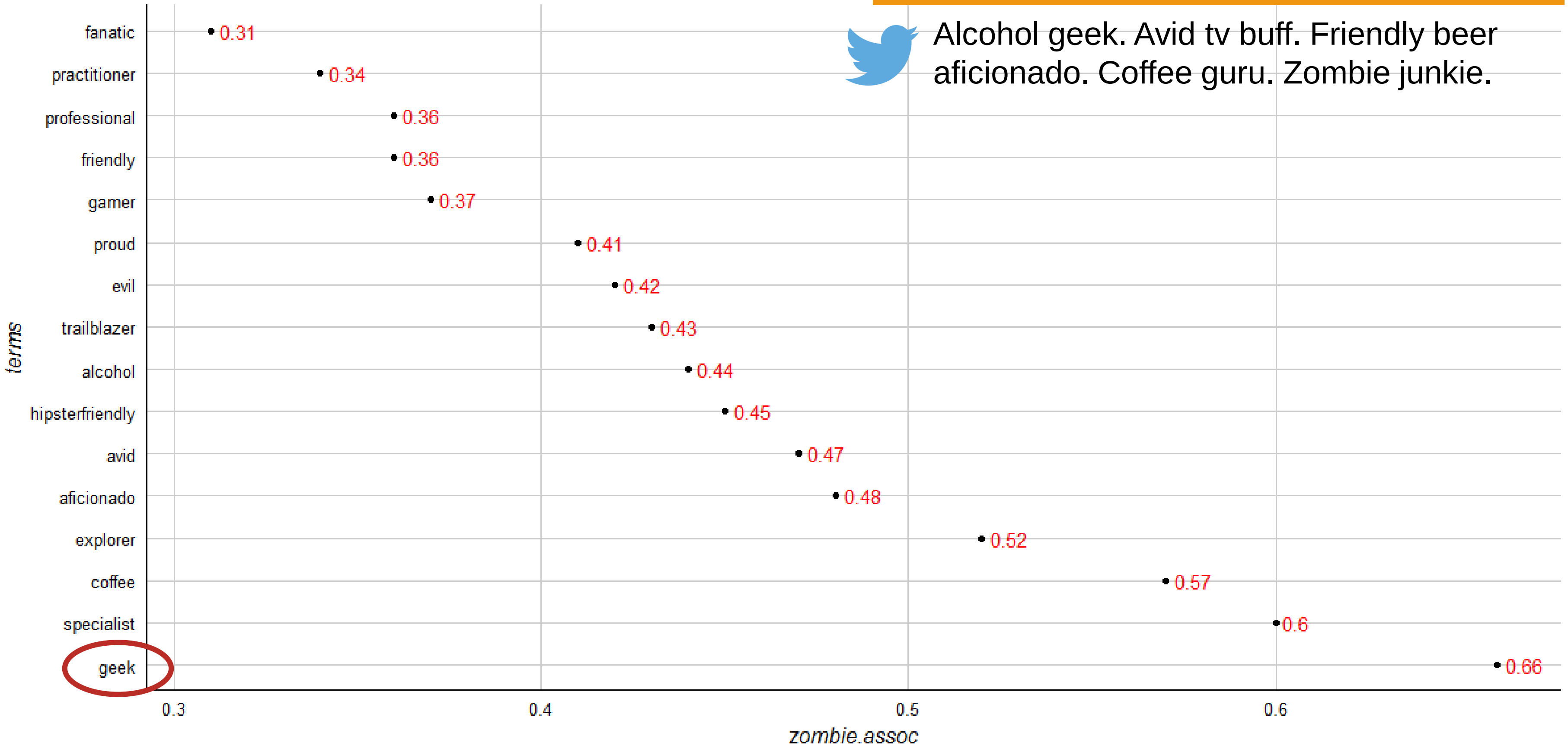
```
associations<-findAssocs(tdm, 'zombie', 0.30)
a.df<-do.call(cbind,associations)
a.df<-data.frame(terms=row.names(a.df),a.df, row.names=NULL)
a.df$terms<-factor(a.df$terms, levels=a.df$terms)
ggplot(a.df, aes(y=terms)) + geom_point(aes(x=zombie), data=a.df)+
  theme_gdocs()+geom_text(aes(x=zombie,label=zombie),
colour="red",hjust=-.25)
```

- Adjust 0.30 to get the terms that are associated .30 or more with the 'zombie' term.
- Treating the terms as factors lets ggplot2 sort them for a cleaner look.

4_dendrogram.R script

"geek" has the highest word association with "zombie".

Alcohol geek. Avid tv buff. Friendly beer aficionado. Coffee guru. Zombie junkie.



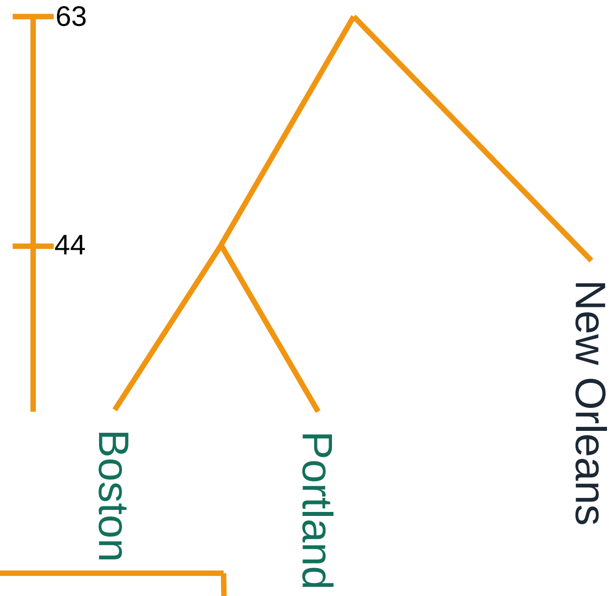
Freq Counts, Dendrograms Wordclouds

Extracting Meaning using dendrograms

Dendrograms visualize hierarchical clusters based on frequencies distances.

- Reduces information much like average is a reduction of many observations' values
- Word clusters emerge often showing related terms
- Term frequency is used to construct the word cluster. Put another way, term A and term B have similar frequencies in the matrix so they are considered a cluster.

City	Annual Rainfall
Portland	43.5
Boston	43.8
New Orleans	62.7



Boston & Portland are a cluster at height 44. You lose some of the exact rainfall amount in order to cluster them.

Freq Counts, Dendrograms Wordclouds

4_dendrogram.R script

Weird associations! Maybe a dendrogram will help us more

```
#Hierarchical Clustering
tdm2 <- removeSparseTerms(tdm, sparse=0.95) #shoot for ~40 terms
tdm2.df<-as.data.frame(inspect(tdm2))
hc <- hclust(dist(tdm2.df))
hcd <- as.dendrogram(hc)
clusMember <- cutree(hc, 4)
labelColors <- c("#CDB380", "#036564", "#EB6841", "#EDC951")
clusDendro <- dendrapply(hcd, colLab)
plot(clusDendro, main = "Hierarchical Dendrogram", type = "triangle")
```

% of zeros allowed
e.g. higher means more words in TDM/DTM

```
> tdm
<<TermDocumentMatrix (terms: 2511, documents: 1000)>>
Non-/sparse entries: 10299/2500701
Sparsity           : 100%
Maximal term length: 65
Weighting          : term frequency (tf)
```

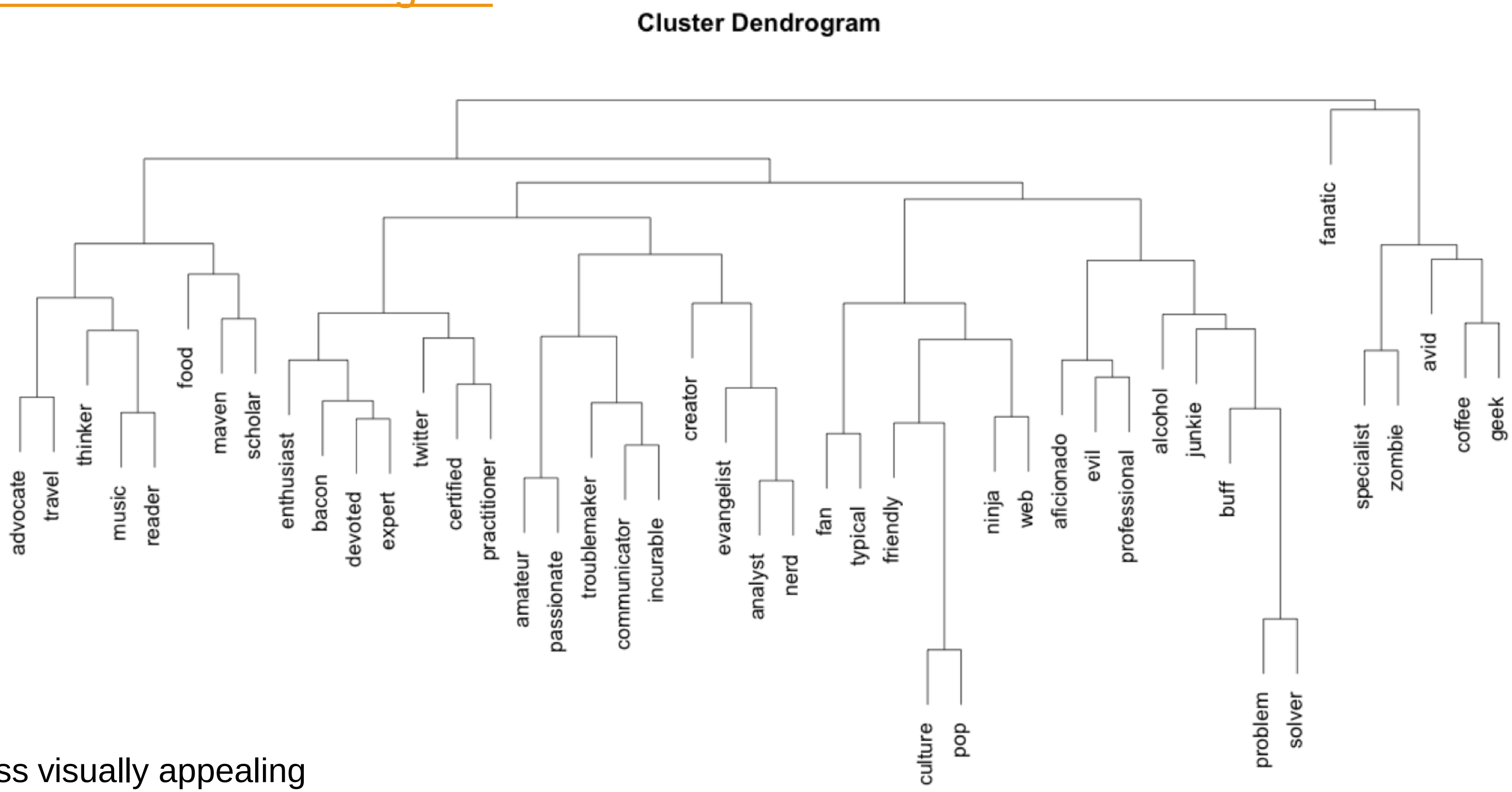
```
> tdm2
<<TermDocumentMatrix (terms: 45, documents: 1000)>>
Non-/sparse entries: 4491/40509
Sparsity           : 90%
Maximal term length: 12
Weighting          : term frequency (tf)
```

New Text Mining Concept

Sparse- Term Document Matrices are often extremely sparse. This means that any document (column) has mostly zero's. Reducing the dimensions of these matrices is possible by specifying a sparse cutoff parameter.   Higher sparse parameter will bring in more terms.

4_dendrogram.R script

Base Plot of a Dendrogram

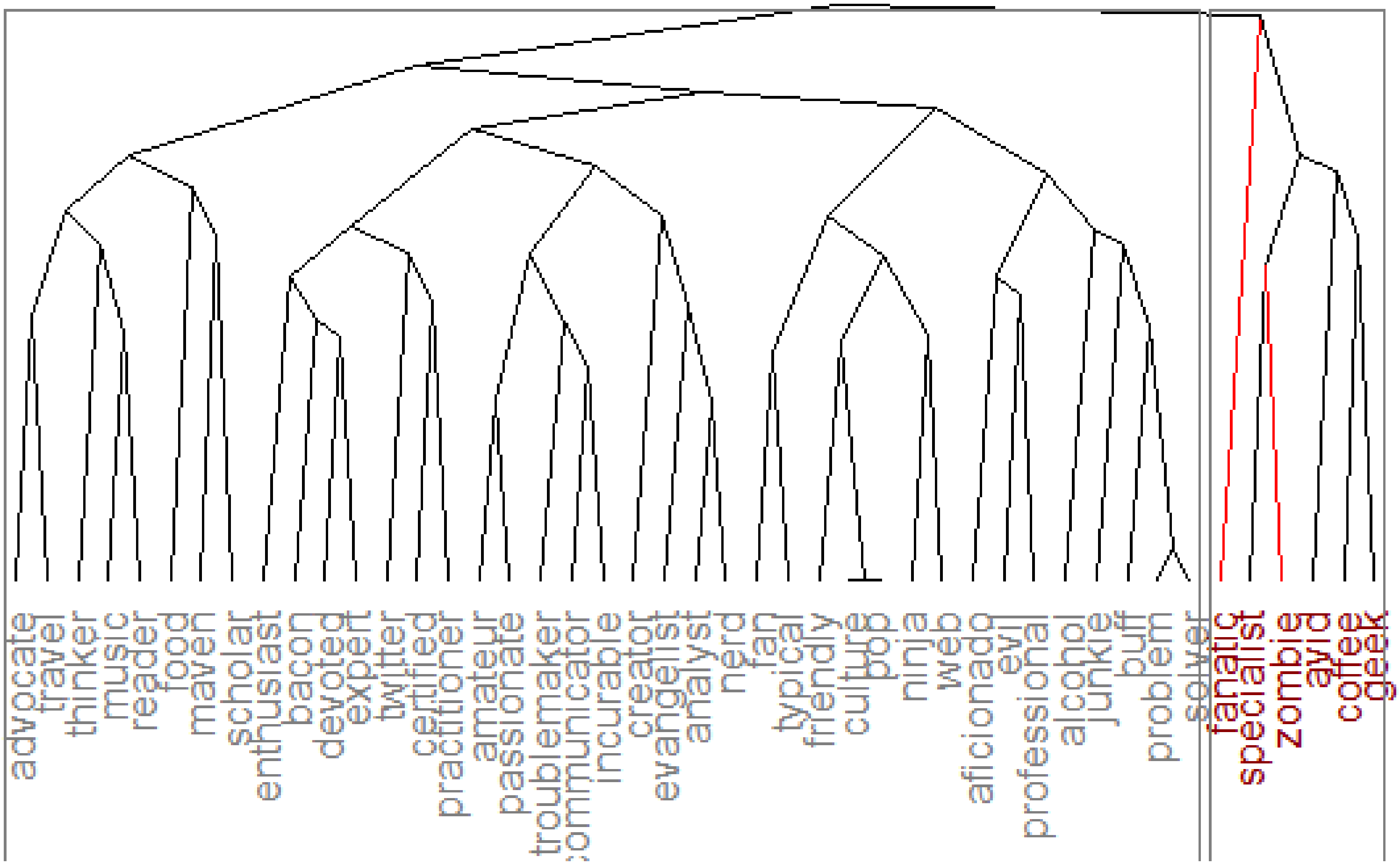


- Less visually appealing
- Clusters can be hard to read given the different heights

Freq Counts, Dendrograms Wordclouds

4_dendrogram.R script

Dendextend offers more flexiblity



Freq Counts, Dendrograms Wordclouds

5_Simple_Wordcloud.R

5_Simple_Wordcloud.R script

```
bigram.tokenizer <-function(x){
  unlist(lapply(ngrams(words(x), 2), paste, collapse = " "), use.names = FALSE)
  }
```

```
tdm<-TermDocumentMatrix(corpus,
control=list(tokenize=bigram.tokenizer))
```

Text Mining is so fun. So do Text Mining!

Unigram

```
           Docs
Terms       1
   fun.     1
   mining   2
   text     2
```

with common stopwords

Bigram

```
                  Docs
Terms              1
   do text         1
   fun so          1
   is so           1
   mining is       1
   so do           1
   so fun          1
   text mining     2
```

New Text Mining Concept

Tokenization- So far we have created single word n-grams. We can create multi word “tokens” like bigrams, or trigrams with this line function. It is applied when making the term document matrix.

Freq Counts, Dendrograms Wordclouds

5_Simple_Wordcloud.R script

To make a wordcloud we follow the previous steps and create a data frame with the word and the frequency.

```
#Summed Vector
tdm.m <- as.matrix(bigram_tdm)
tdm.v <- sort(rowSums(tdm.m),decreasing=TRUE)
tdm.df <- data.frame(word =
names(tdm.v),freq=tdm.v)
```

Term Document Matrix

	Tweet1	Tweet 2	Tweet3	Tweet4	…	Tweet_n
Term1	0	0	0	0	0	0
Term2	1	1	0	0	0	0
Term3	1	0	0	2	0	0
…	0	0	3	0	1	1
Term_n	0	0	0	1	1	0

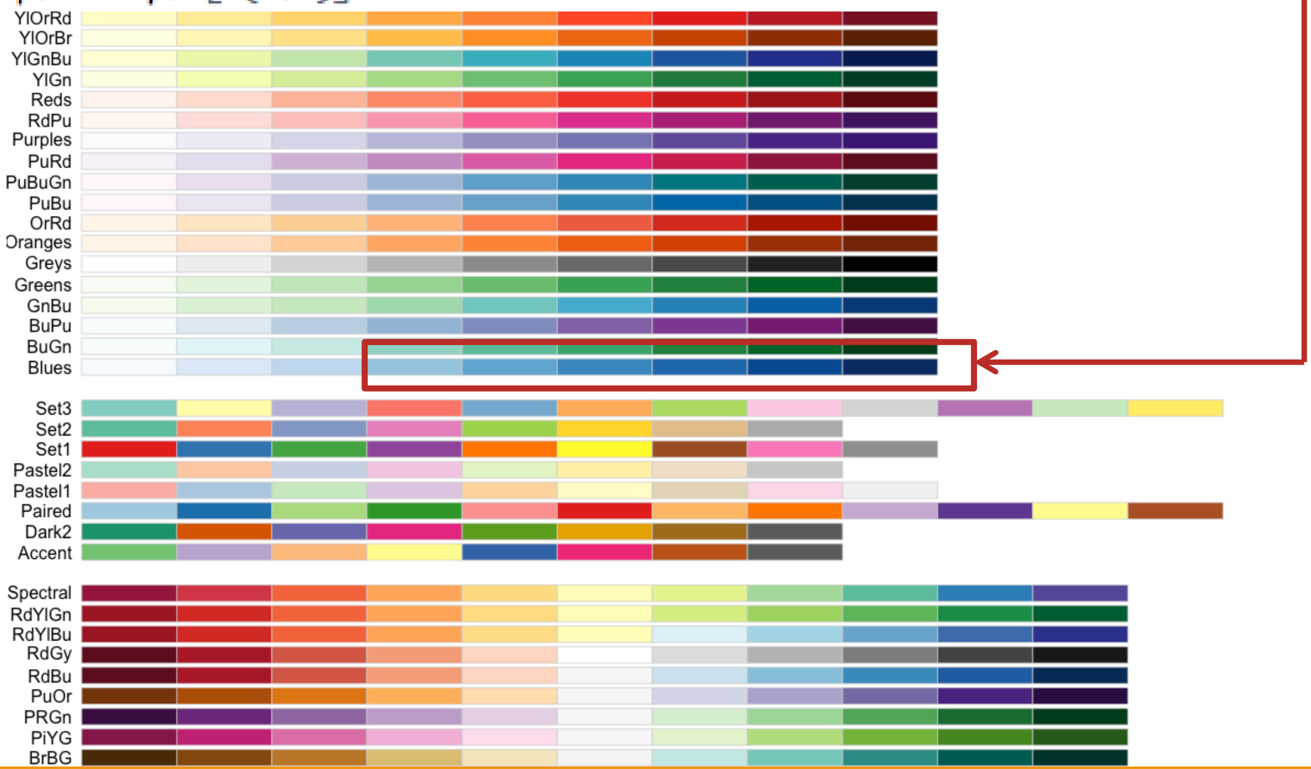
word	freq
Term1	0
Term2	2
Term3	3
…	5
Term_n	2

Freq Counts, Dendrograms Wordclouds

5_Simple_Wordcloud.R script

Next we need to select the colors for the wordcloud.

```
#look at all available color pallettes & choose one
display.brewer.all()
pal <- brewer.pal(8, "Blues")
pal <- pal[-(1:2)]
```



Freq Counts, Dendrograms Wordclouds

5_Simple_Wordcloud.R script

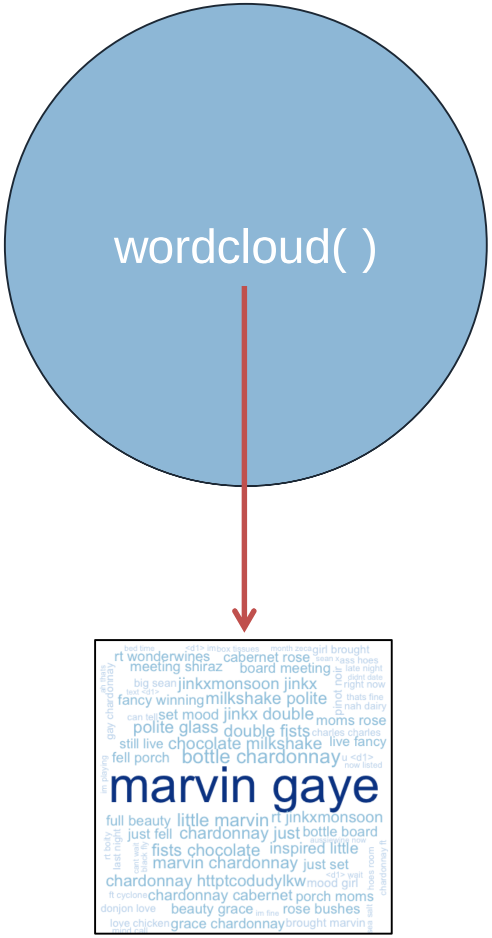
```
set.seed(2016)
wordcloud(tdm.df$word,tdm.df$freq,max.words=50, random.order=FALSE, colors=pal)
```

- Bigram Tokenization has captured “marvin gaye”
- A word cloud is a frequency visualization. The larger the term (or bigram here) the more frequent the term.
- You may get warnings if certain tokens are to large to be plotted in the graphics device.

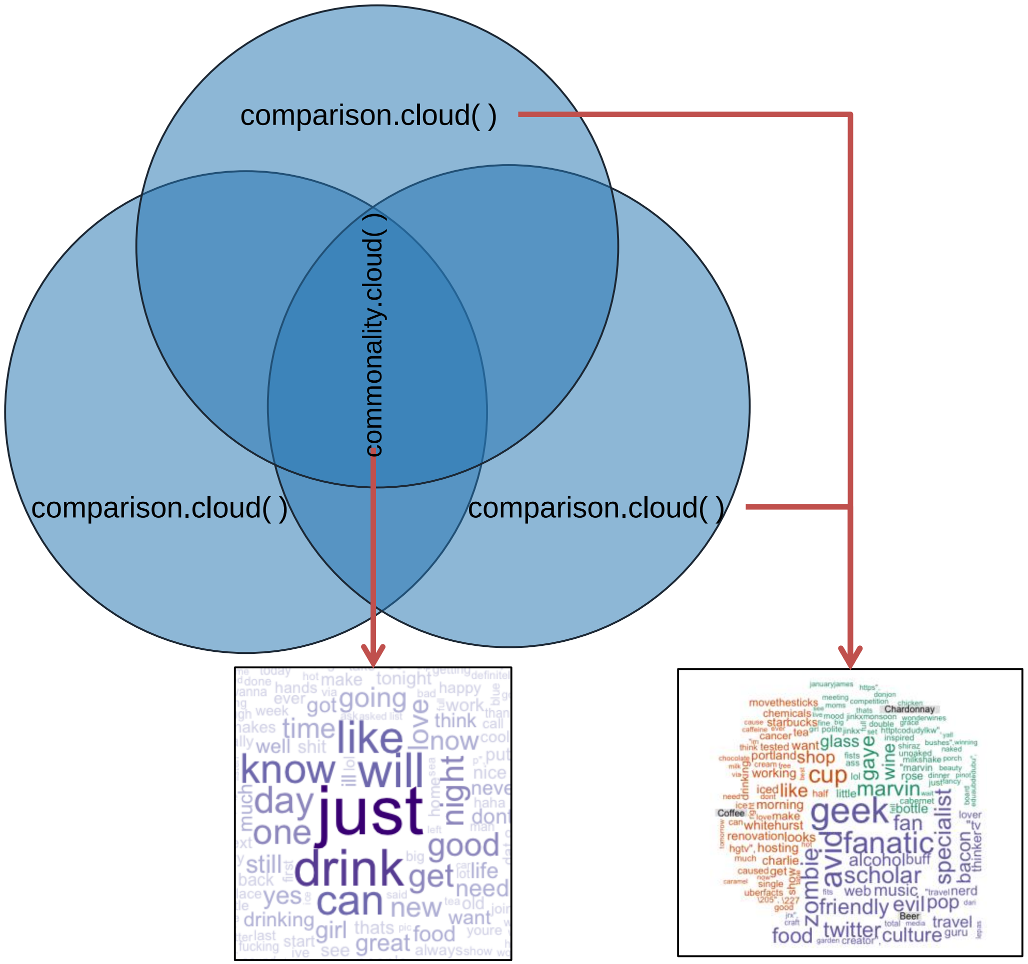
Freq Counts, Dendrograms Wordclouds

Types of Wordclouds

Single Corpus



Multiple Corpora



Freq Counts, Dendrograms Wordclouds

6_Other_Wordclouds.R

6_Other_Wordcloud.R

Bring in more than one corpora.

```
corp.collapse<-function(csv.name, text.column.name){
  x <- read.csv(file=csv.name, head=TRUE, sep=",")
  x <-iconv(x[,text.column.name], "latin1","ASCII",sub='')
  x <- VCorpus(VectorSource(x))
  x <- tm_map(x, removePunctuation)
  x <- tm_map(x, removeNumbers)
  x <- tm_map(x, tryTolower)
  x <- tm_map(x, removeWords, custom.stopwords)
  x <- paste(x, collapse=" ")
}
```

This function accepts the CSV, creates a corpus, has the "clean.corpus" functions embedded and finally collapses all 1000 individual tweets into a single document.

```
chardonnay<-corp.collapse('chardonnay.csv','text')
coffee<-corp.collapse('coffee.csv','text')
beer<-corp.collapse('beer.csv','text')
```

Freq Counts, Dendrograms Wordclouds

Commonality Cloud

- The tweets mentioning “chardonnay” “beer”, and “coffee” have these words in common.
- Again size is related to frequency.
- Not helpful in this but in diverse corpora it may be more helpful e.g. political speeches.


```
#Common Words
commonality.cloud(tdm, max.words=300, random.order=FALSE,colors=pal)
```

Freq Counts, Dendrograms Wordclouds

Comparison Cloud

- The tweets mentioning "chardonnay" "beer", and "coffee" have these dissimilar words.
- Again size is related to frequency.
- Beer drinkers in this snapshot are passionate (fanatics, geeks, specialists) on various subjects while Chardonnay drinkers mention Marvin Gaye. Coffee mentions up & working.


```
comparison.cloud(all.tdm, max.words=150,
random.order=FALSE,
title.size=1.0,
colors=brewer.pal(ncol(all.tdm),"Dark2"))
```

Agenda

- What is Text Mining?
- String Manipulation Cleaning
- Freq Counts, Dendrograms Wordclouds
- Sentiment Analysis
- Topic Modeling, OpenNLP

Simple Sentiment Polarity

Scoring

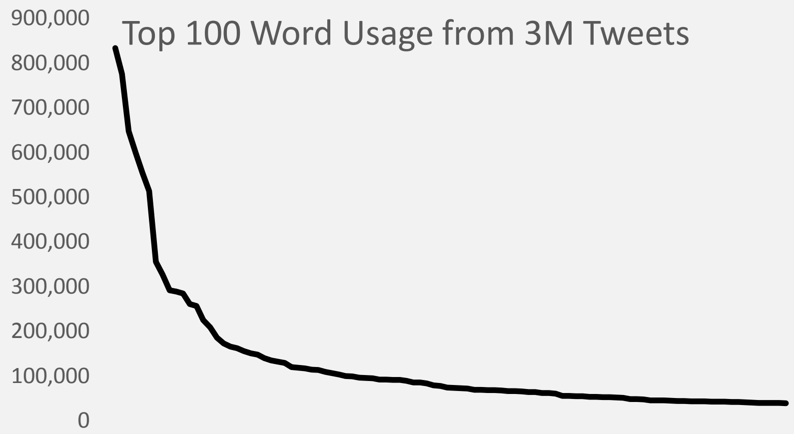
Surprise is a sentiment.
Hit by a bus! – Negative Polarity
Won the lottery!- Positive Polarity

- I loathe BestBuy Service -1
- I love BestBuy Service. They are the best. +2
- I like shopping at BestBuy but hate traffic. 0

R's QDAP polarity function scans for positive words, and negative words as defined by MQPA Academic Lexicon research. It adds positive words and subtracts negative ones along with valence shifters. The final score represents the polarity of the social interaction.

Zipf's Law

Many words in natural language but there is steep decline in everyday usage. Follows a predictable pattern.



Simple Sentiment Polarity

Scoring

```
library(qdap)

text1<-'i love St Peters University'
text2<-'this lecture is good'
text3<-'this lecture is very good'
text4<-'data science is hard I like it a little'
text5<-'data science is hard'

polarity(text1)
polarity(text2)
polarity(text3)
polarity(text4)
polarity(text5)
```

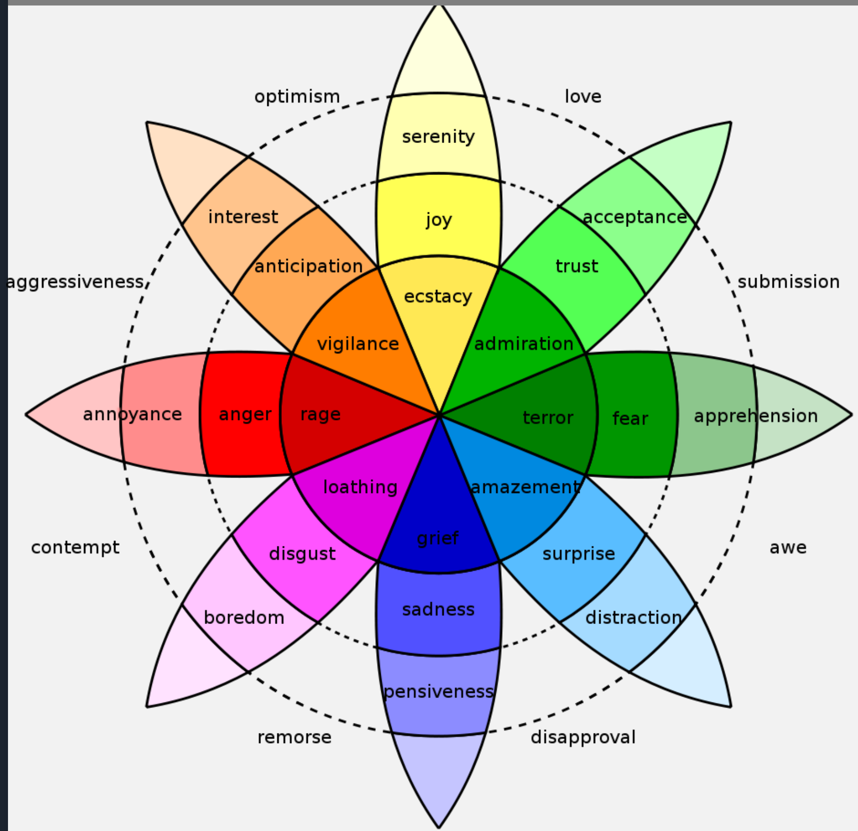
- Text 1: "love" was identified as positive. The text has 5 words and so 1/sqrt(5) = .447
- Text 2: "good" was identified positively. So 1/sqrt(4)=.5
- Text 3: "good" was found along with the amplifier "very". So (.8+1)/sqrt(5)=.805
- Text 4: hard and like cancel each other out so the polarity is zero. 1-1/sqrt(9)=0
- Text 5: "hard" is -1/sqrt(4)=-.50

First it looks for the polarized word. Then identifies valence shifters (default 4 words before and two words after) Amplifiers are assigned +.8 and de-amplifiers weight is constrained to -1. Lastly the sum is divided by the square root of the total number of words in the passage.

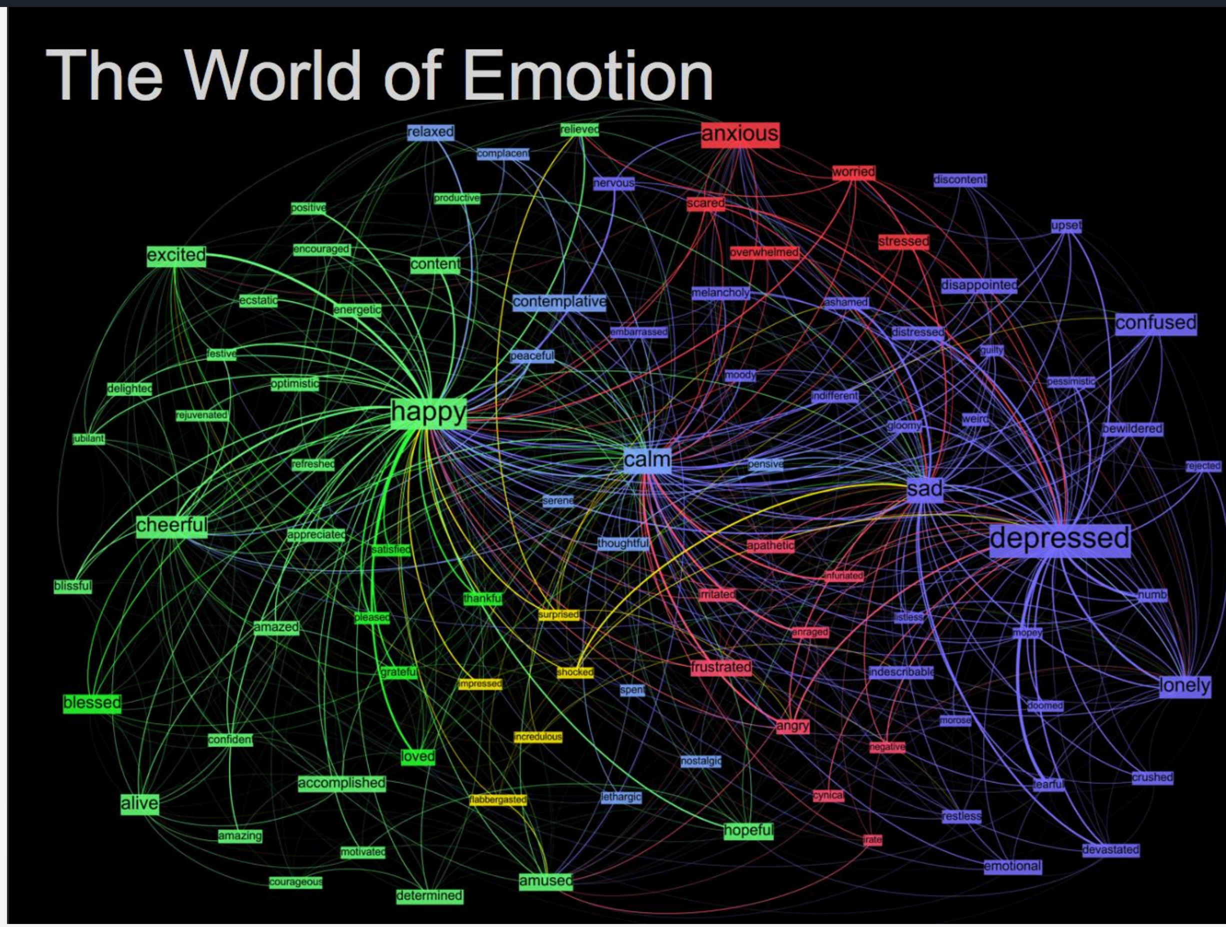
In reality sentiment is more complex.

Many Many Emoji

Plutchik's Wheel



Kanjoya's Experience Corpus



Sentiment the Tidy Way!

```
##Tidy Sentiment Analysis
data(sentiments)
sentiments

#Stopwords
data(stop_words)
stop_words

#Add stopwords
custom.stopwords<-data.frame(word=c('amp','beer'),
lexicon='custom')

stop_words<-rbind(stop_words,custom.stopwords)
```

```
> sentiments
# A tibble: 23,165 x 4
          word sentiment lexicon score
         <chr>     <chr>   <chr> <int>
1       abacus     trust     nrc    NA
2      abandon      fear     nrc    NA
3      abandon  negative     nrc    NA
4      abandon   sadness     nrc    NA
5    abandoned     anger     nrc    NA
6    abandoned      fear     nrc    NA
7    abandoned  negative     nrc    NA
8    abandoned   sadness     nrc    NA
9  abandonment     anger     nrc    NA
10 abandonment      fear     nrc    NA
# ... with 23,155 more rows
```

```
> stop_words
# A tibble: 1,151 x 2
          word lexicon
*        <chr>   <chr>
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
# ... with 1,141 more rows
```


Ceci n'est pas une pipe.

Tidy Functions

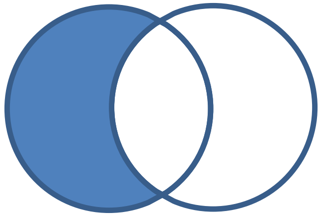
The pipe operator

%>%

Forwards an object so the code is easy to understand & concise.

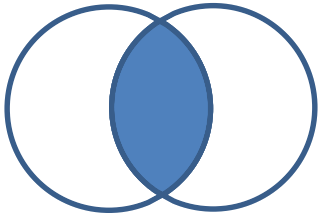
```
all.tidy <- all.tidy %>%
  anti_join(stop_words)
```

Tweet words   Stop words



`anti_join()`

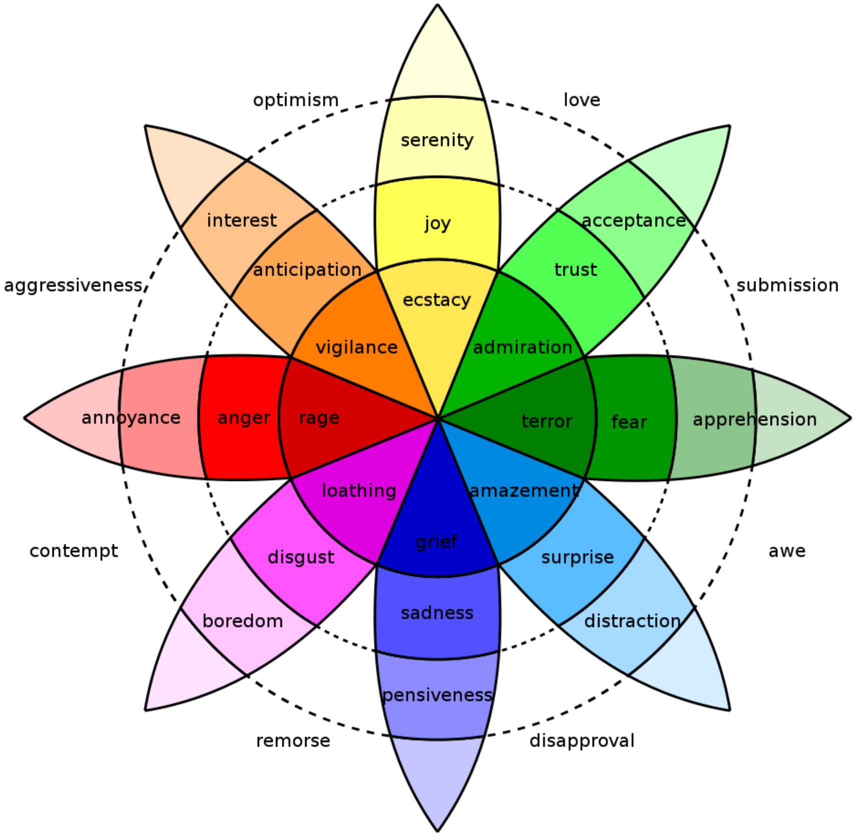
Sentiment words   text



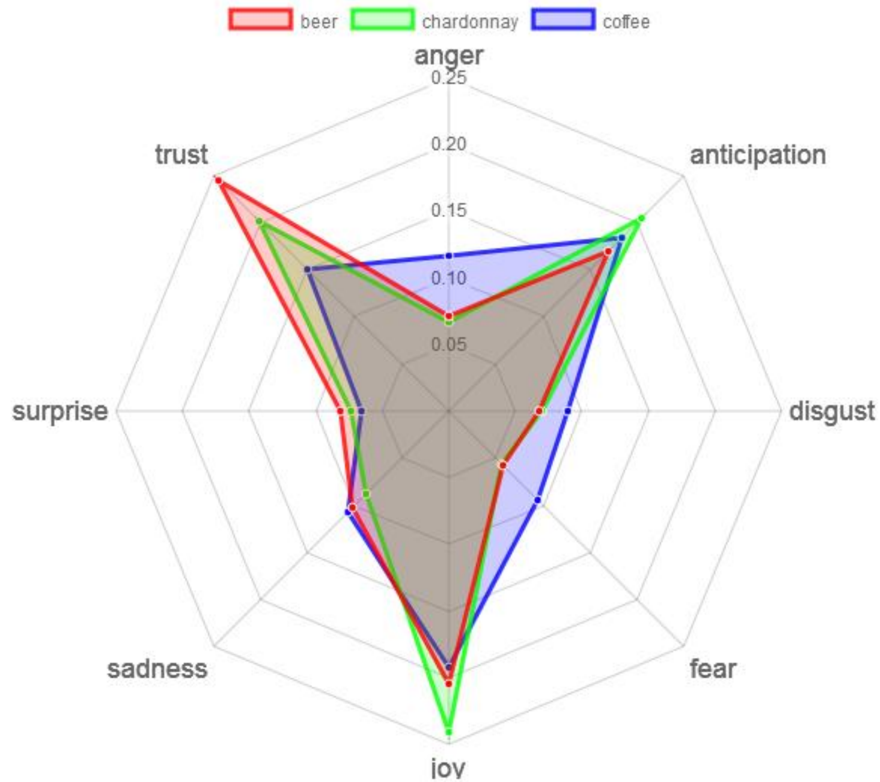
`inner_join()`

```
all.sentiment <- all.tidy %>%
  inner_join(nrc.lexicon) %>%
  count(tweet,sentiment) %>%
  spread(tweet, n, fill = 0)
```

7_Sentiment_analysis.R



optimism
love
serenity
interest
acceptance
joy
anticipation
trust
aggressiveness
submission
ecstacy
vigilance
admiration
annoyance
anger
rage
terror
fear
apprehension
loathing
amazement
contempt
awe
grief
disgust
surprise
sadness
boredom
distraction
pensiveness
remorse
disapproval



beer
chardonnay
coffee
anger
0.25
0.20
0.15
0.10
0.05
trust
anticipation
surprise
disgust
sadness
fear
joy

Agenda

- What is Text Mining?
- String Manipulation Cleaning
- Freq Counts, Dendrograms Wordclouds
- Sentiment Analysis
- Topic Modeling, OpenNLP

Not enough time
Just ask for files

Questions?

https://github.com/kwartler/ODSC_Workshop

@tkwartler

www.linkedin.com/in/edwardkwartler