DUKE
FUQUA
SCHOOL OF BUSINESS

Sudipta Dasmohapatra
sd345@duke.edu

Natural Language Processing

Fall 2019

Introduction

"Unstructured Data"

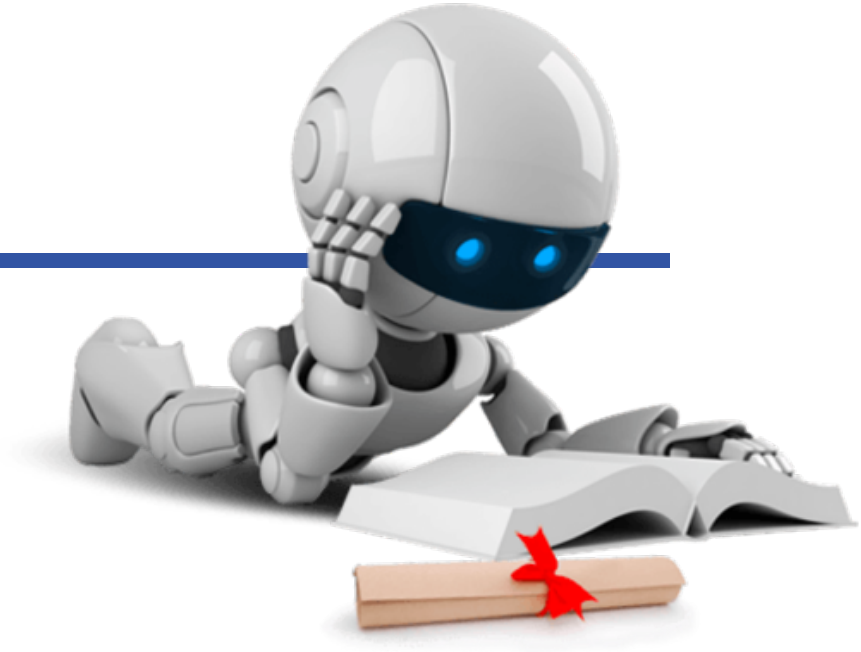
Processing and Analysis

DUKE
FUQUA
SCHOOL OF BUSINESS

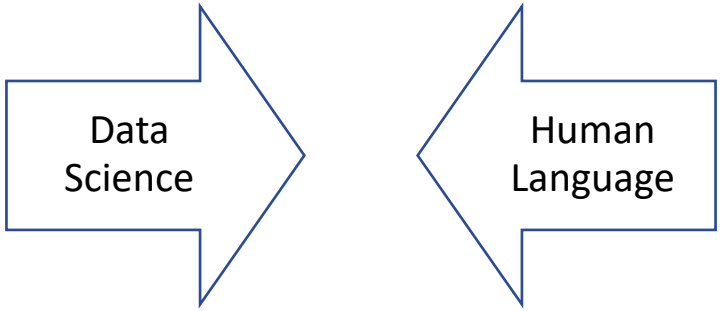
Natural Language Processing (NLP)

- Natural Language Understanding:
  - Taking some spoken/typed sentence and working out what it means
- Natural Language Generation:
  - Taking some formal representation of what you want to say and working out a way to express it in a natural (human) language (e.g., English)

NLP-what is it?



- Field of machine learning where machines can read, understand and analyze the human languages



DUKE FUQUA SCHOOL OF BUSINESS

Natural Language Processing

- Computation to analyze text
  - Classification of text
  - Text mining – Processing and Understanding
  - Topic modeling
  - Sentiment analysis

Hands-on examples for data retrieval from web, text wrangling and pre-processing, parsing, sentiment analysis

DUKE FUQUA SCHOOL OF BUSINESS

Regular Expresssions

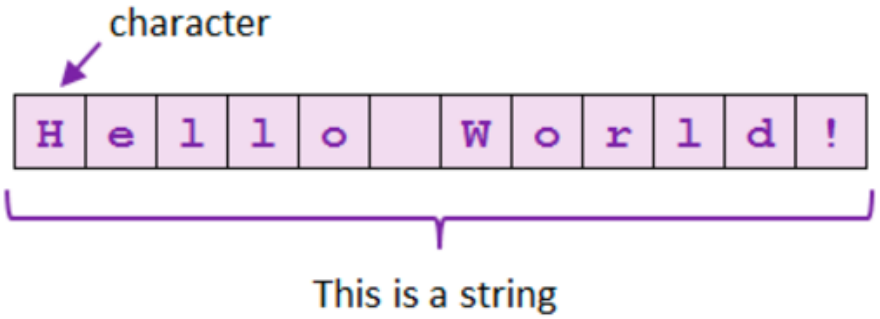
- What is it?
- Types
- Examples

What is a Regular Expression?

- RegEx: Sequence of characters to search text
- Text data is messy
  - We need some methods to extract useful information from the noisy data
  - String manipulation functions and regular expressions will prepare you for text mining
- Regular expressions are set of pattern matching commands used to detect string sequences in a large text data (Using regex you can get more out of text data by writing shorter codes)

String Manipulation

- String Manipulation is a series of functions to extract information from text variables



- Widely used for doing feature engineering (creating new features from existing string features)
  - In R: stringr and stringi are two packages

String Manipulation Functions in R

- String is any value in “ “

```
#1. String Manipulation
#a. Basic string manipulation using base R
text <- "north carolina"
typeof(text)
[1] "character”
num<- c(”2018", ”2019", ”2020")
typeof(num)
[1] "character"
```


DUKE FUQUA SCHOOL OF BUSINESS

String Manipulation Functions in R

```
ml<- paste("nlp", "deepl", sep="-")
ml
[1] "nlp-deepl”

paste(1:5, c("nlp", "deepl"), sep="-")
[1] "1-nlp"   "2-deepl" "3-nlp"   "4-deepl" "5-nlp”

#Print and Concatenate strings without quotes
cat(text, "Durham", sep="-")
north carolina-Durham
```

String Manipulation Functions in R

```
cat(month.name[1:5], sep=" ")
```

January February March April May

```
#convert non-character value to a string using toString
toString(1:10)
```

Functions	Description
nchar()	Counts no. of characters in a string or vector
tolower()	Coverts string to lower case
toupper()	Converts string to upper case
substr()	Used to extract parts of a string. Start and end positions to be specified
strsplit()	Split a string based on a criterion. Gives a list

String Functions

```
library(stringr)
string <- "Duke U in North Carolina was created in 1924 by James Buchanan Duke as a memorial to his father, Washington Duke. The Dukes, a Durham family that built a worldwide financial empire in the manufacture of tobacco products and developed electricity production in the Carolina, long had been interested in Trinity College. In December 1924, the provisions of indenture by Benjamin's brother, James B. Duke, created the family philanthropic foundation, The Duke Endowment, which provided for the expansion of Trinity College into Duke U."."
```

String Functions

strwrap(string)

[1] "Duke U was created in 1924 by James Buchanan Duke as a memorial to his father, Washington Duke. The Dukes, a Durham family"

[2] "that built a worldwide financial empire in the manufacture of tobacco products and developed electricity production in the"

[3] "Carolinas, long had been interested in Trinity College. In December 1924, the provisions of indenture by Benjamin's brother,"

[4] "James B. Duke, created the family philanthropic foundation, The Duke Endowment, which provided for the expansion of Trinity"

[5] "College into Duke U."

String Functions

nchar(string)

[1] 536

str_length(string)

[1] 536

tolower(string)

str_to_lower(string)

[[1] "duke u in north carolina was created in 1924 by james buchanan duke as a memorial to his father, washington duke. \nthe dukes, a durham family that built a worldwide financial empire in the manufacture of tobacco products \nand developed electricity production in the carolina, long had been interested in trinity college. in december 1924, \nthe provisions of indenture by benjamin's brother, james b. duke, created the family philanthropic foundation, \nthe duke endowment, which provided for the expansion of trinity college into duke u."

String Functions

#convert to upper

toupper(string)

str_to_upper(string)

[1] "DUKE U IN NORTH CAROLINA WAS CREATED IN 1924 BY JAMES BUCHANAN DUKE AS A MEMORIAL TO HIS FATHER, WASHINGTON DUKE. \nTHE DUKES, A DURHAM FAMILY THAT BUILT A WORLDWIDE FINANCIAL EMPIRE IN THE MANUFACTURE OF TOBACCO PRODUCTS \nAND DEVELOPED ELECTRICITY PRODUCTION IN THE CAROLINA, LONG HAD BEEN INTERESTED IN TRINITY COLLEGE. IN DECEMBER 1924, \nTHE PROVISIONS OF INDENTURE BY BENJAMIN'S BROTHER, JAMES B. DUKE, CREATED THE FAMILY PHILANTHROPIC FOUNDATION, \nTHE DUKE ENDOWMENT, WHICH PROVIDED FOR THE EXPANSION OF TRINITY COLLEGE INTO DUKE U."

String Functions

```
#replace strings
str_replace_all(string=string, pattern=c("U"),
replacement="University") # this is case sensitive
```

```
[1] "Duke University in North Carolina was created in
1924 by James Buchanan Duke as a memorial to his father,
Washington Duke. \nThe Dukes, a Durham family that
built a worldwide financial empire in the manufacture of
tobacco products \nand developed electricity production
in the Carolina, long had been interested in Trinity
College. In December 1924, \nthe provisions of indenture
by Benjamin's brother, James B. Duke, created the family
philanthropic foundation, \nThe Duke Endowment,
which provided for the expansion of Trinity College into
Duke University."
```

String Functions

```
#extract parts of a string
substr(x=string, start=5, stop=25)
[1] " U in North Carolina "
#get difference between two vectors
setdiff(c("ml", "deepl", "ai"), c("ml", "iot", "cloudcomp"))
[1] "deepl" "ai"
#check if strings are equal
setequal(c("ml", "deepl", "ai"), c("ml", "deepl", "ai"))
[1] TRUE
setequal(c("ml", "deepl", "ai"), c("ml", "deepl", "iot"))
[1] FALSE
```

String Functions

#abbreviate strings

```
abbreviate(c("university", "college", "department"), minlength=4)
```

```
university    college department
    "unvr"     "cllg"     "dprt"
```

#split strings

```
strsplit(x=c("Nov 10", "Dec 21", "Jan 6", "Feb 18"), split=" ")
```

```
str_split(string=c("Nov 10", "Dec 21", "Jan 6", "Feb 18"), pattern=" ", simplify=T)
```

```
 [,1]  [,2]
[1,] "Nov" "10"
[2,] "Dec" "21"
[3,] "Jan" "6"
[4,] "Feb" "18"
```

String Functions

```
#find and replace matches

sub(pattern="C", replacement="K", x=string, ignore.case=T)
#replace first match only
```

[1] "Duke U in North Karolina was created in 1924 by James Buchanan Duke as a memorial to his father, Washington Duke. \nThe Dukes, a Durham family that built a worldwide financial empire in the manufacture of tobacco products \nand developed electricity production in the Carolina, long had been interested in Trinity College. In December 1924, …

```
gsub(pattern="Carolina", replacement="Karolina", x=string,
ignore.case=T) #replaces all matches
```

[1] "Duke U in North Karolina was created in 1924 by James Buchanan Duke as a memorial to his father, Washington Duke. \nThe Dukes, a Durham family that built a worldwide financial empire in the manufacture of tobacco products \nand developed electricity production in the Karolina, long had been

List of RegEx Commands

Function	Description
grep	Returns the index or value of the matched string
grepl	Returns the Boolean (true/false) of the matched string
regexpr	Returns the index of the first match
gregexpr	Returns the index of all matches
regexec	Is a hybrid of regexpr and gregexpr
regmatches	Returns the matched string at a specified index. Used in conjunction with regexpr and gregexpr

Regular Expressions

- Metacharacters
- Sequences
- Quantifiers
- Character classes
- POSIX character classes

1. Metacharacters

\ | ( ) [ ] { } $ * + ? (regex doesn't detect them, you will need \\ to detect them)

```
#c.1 Metacharacters (use both grep and gsub function)
ws<-c("percent%", "percent")
grep(pattern="percent\\%", x=ws, value=T)
[1] "percent%"
ws<-c("nlp?", "ai$", "ml&")
grep(pattern="[a-z][\\?-\\$-\\&]", x=ws, value=T)
[1] "nlp?" "ai$"  "ml&"
gsub(pattern="[\\?-\\$-\\&]", replacement=" ", x=ws)
[1] "nlp " "ai "  "ml "
```

2. Quantifiers

Helps to detect patterns in text

Changes output values entirely with slight position change

- Determine length of resulting match
- Is exercised on items to the immediate left of it

Quantifier	Description
.	It matches everything except a newline
?	Item to the left is optional and is matched at most once
*	Item to its left will be matched zero or more times
+	Item to its left will be matched one or more times
{n}	Item to the left is matched exactly n times
{n,}	Item to the left is matched n or more times
{n,m}	Items to the left is matched at least n times but not more than m times

2. Quantifiers

```
number <- "1010000000010"
regmatches(number, gregexpr(pattern="1.*1", text=number))
[1] "101000000001"
regmatches(number, gregexpr(pattern="1.?1", text=number))
[1] "101”
states <- c("virginia", "florida", "pennsylvania", "tennessee",
"texas")
grep(pattern= "i+", x=states, value=T)
[1] "virginia"     "florida"      "pennsylvania"
grep(pattern="n{2}", x=states, value=T)
[1] "pennsylvania" "tennessee"
```

3. Sequences

Sequences contain special characters used to describe a pattern in a given string.

Sequences	Description
\d	Matches a digit character
\D	Matches a non-digit character
\s	Matches a space character
\S	Matches a non-space character
\w	Matches a word character
\W	Matches a non-word character
\b	Matches a word boundary
\B	Matches a non-word boundary

3. Sequences

```
mqm<- "The MQM program began 2 years ago”
#match a digit
gsub(pattern="\\d+", replacement="_", x=mqm)
[1] "The MQM program began _ years ago"
regmatches(mqm, regexpr(pattern="\\d+", text=mqm))
[1] "2”
#match a non-digit
gsub(pattern="\\D+", replacement="_", x=mqm)
[1] "_2_"
regmatches(mqm, regexpr(pattern="\\D+", text=mqm))
[1] "The MQM program began "
```

3. Sequences

```
mqm<- "The MQM program began 2 years ago”
#match a space - returns positions
 gregexpr(pattern="\\s+", text=mqm)
[1]  4  8 16 22 24 30
attr(,"match.length")
[1] 1 1 1 1 1 1
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
#match a non space
gsub(pattern="\\S+", replacement="app", x=mqm)
[1] "app app app app app app app"
```

```
#match a word character
gsub(pattern="\\w",
replacement="k", x=mqm)
[1] "kkk kkk kkkkkkk kkkkk k
kkkkk kkk"

#match a non-word character
gsub(pattern="\\W",
replacement="k", x=mqm)
[1]
"ThekMQMkprogramkbegan
k2kyearskago"
```


DUKE FUQUA SCHOOL OF BUSINESS

4. Character Classes

Set of characters enclosed in a square bracket [ ]

These classes match only the characters enclosed in [ ]

Characters	Description
[aeiou]	Matches lower case vowels
[AEIOU]	Matches upper case vowels
[0123456789]	Matches any digit
[0-9]	Matches any digit
[a-z]	Match any lower case letter
[A-Z]	Match any upper case letter
[a-zA-Z0-9]	Matches any of the above classes
[^aeiou]	Matches everything except letters
[^0-9]	Matches everything except digits

4. Character Classes

```
mqm <- "The MQM program is for 10 months. Students can choose 4 different tracks."

#extract numbers

regmatches(x=mqm, gregexpr("[0-9]+", text=mqm))

[1] "10" "4"

#extract without digits

regmatches(x=mqm, gregexpr("[^0-9]+", text=mqm))

[1] "The MQM program is for "       " months. Students can choose " " different tracks."
```

5. POSIX Character Classes

Classes enclosed within double square brackets ([[ ]])

POSIX characters	Description
[[:lower:]]	Matches lower case letters
[[:upper:]]	Matches upper case letters
[[:alpha:]]	Matches letters
[[:digit:]]	Matches digits
[[:space:]]	Matches space chrs, e.g., tab, newline, vertical tab, space
[[:blank:]]	Matches blank chrs, e.g., space, tab
[[:alnum:]]	Matches alphanumeric chrs, e.g., AB12, ID101
[[:cntrl:]]	Matches control chras (non printable) \t (tab), \e escape
[[:punct:]]	Matches punctuation chrs
[[:graph:]]	Matches graphical chrs. These are [[:alpha:]] and [[:punct:]]
[[:print:]]	Matches printable chrs. ([[:alpha:]], {{:punct:]] and space

5. POSIX Character Classes

```
mqm <- c("There are over 200 students\n, in the MQM program",
"The students\n can choose 4 tracks.", "The program is for\t 10
months?")

#get digits

unlist(regmatches(mqm, gregexpr("[[:digit:]]+", text=mqm)))

[1] "200" "4"   "10"

#remove punctuations

gsub(pattern="[[:punct:]]+", replacement=" ", x=mqm)

[1] "There are over 200 students\n  in the MQM program" "The
students\n can choose 4 tracks "

[3] "The program is for\t 10 months "
```

5. POSIX Character Classes

```
#remove spaces
gsub(pattern="[[:blank:]]", replacement="-", x=mqm)
[1] "There-are-over-200-students\n,-in-the-MQM-program" "The-students\n-can-choose-4-tracks."
[3] "The-program-is-for--10-months?
#remove control characters
gsub(pattern="[[:cntrl:]]+", replacement="", x=mqm)
[1] "There are over 200 students, in the MQM program" "The students can choose 4 tracks."
[3] "The program is for 10 months?"
#remove nongraphical characters
gsub(pattern="[^[:graph:]]+", replacement="", x=mqm)
[1] "Thereareover200students,intheMQMprogram" "Thestudentscanchoose4tracks."  "Theprogramisfor10months?"
```

Summary of Regular Expressions

- Crucial aspect of text mining and NLP
- Used for feature engineering on text data
- Basics of string manipulations and regular expressions
  - Base R functions to handle strings and regex

Exercise 1

1. Extract digits from a string of characters

#extract digits

exercise <- “My id number is 1006781”

2. Remove punctuation from a line of text

exercise_4<- "a1~!@#$%^&*bcd()_+:efg\"<>?,./;'[]-="

3. Extract email addresses from a given string

email <- c(“My email address is sudipta.dasmohapatra@duke.edu”, my email address is sd@work.edu”, “esther koeif”, “paul donneg”

Solutions Exercise 1

#1. Extract digits from a string of characters (any three work)

```
exercise<- "My id number is 1006781"

gsub(pattern="[^0-9]", replacement="", x=exercise)
[1] "1006781"
regmatches(exercise, regexpr("[0-9]+", exercise))
[1] "1006781"
regmatches(exercise, regexpr("[[:digit:]]+", exercise))
[1] "1006781"
```

Solutions Exercise 1

#2. Remove punctuation from a line of text

```
exercise_2<- "a1~!@#$%^&*bcd(){}_+:efg\"<>?,./;'[]-="
gsub(pattern="[[:punct:]]+", replacement="", x=exercise_2)
[1] "a1bcdefg"
```

#3. Extract email addresses from a given string

```
email <-c("My email address is sudipta.dasmohapatra@duke.edu", "my email address is sd@work.edu", "esther koeif", "paul donegg")
unlist(regmatches(x=email, gregexpr(pattern = "[[:alnum:]]+\\@[[:alpha:]]+\\.edu", text=email)))
[1] "dasmohapatra@duke.edu" "sd@work.edu"
```


DUKE FUQUA SCHOOL OF BUSINESS

Text Mining and Feature Engineering

- What is TM and NLP?
- Steps in TM
- Packages used in R

- Example

Text Mining and Feature Engineering

- What is Text Mining and NLP?
- Steps in Text Mining
- Feature Engineering techniques in Text Mining
- Practical Example

Text Mining or NLP

Set of ML techniques that automates text processing to derive useful insights from unstructured data

Resulting data is structural and are high dimensional (e.g., large rows and columns)

Generally, algorithms such as *naiive bayes, deep learning, glmnet* work well with text data

Algorithms

- Naiive Bias: Classification technique that assumes that the presence of a particular feature in a class is unrelated to any other feature
- Deep learning: Use Neural Networks with layers to classify
- Glmnet: Generalized linear model (class of models where response variable is assumed to follow an exponential family distribution) with fit based on penalized maximum likelihood

Text Mining

- **Text mining** methods allow us to highlight the most frequently used keywords in a paragraph of texts. One can create a **word cloud**, also referred as *text cloud* or *tag cloud*, which is a visual representation of text data.
- The procedure of creating word clouds is very simple in R if you know the different steps to execute. The text mining package (*tm*) and the word cloud generator package (*wordcloud*) are available in R for helping us to analyze texts and to quickly visualize the keywords as a word cloud.

Bag of Words Model

- Model that allows you to count all words in a piece of text
- Creates an occurrence matrix for the sentence or document disregarding grammar or word order
- These word frequencies are used then as features for a training a classifier

Steps in Text Mining

Process involves cleaning phase to transform the text into tabular format for analysis

- Corpus Development
- Text Cleaning
- Feature Engineering
- Model Building

1. Corpus Development

- Create a matrix (table) comprising documents and terms (tokens)
- A document is each row having the text (e.g., customer feedback) and each column having terms (words)
  - Terms are each word in the description
  - Number of documents in the corpus is equal to the number of rows in the given data

2. Text Cleaning

- *Remove words*: e.g., remove html tags
- *Remove stopwords*: e.g., words such as “a”, “an”, ”the”, ”they”, “where”, …

(Stopwords are a set of words which helps in sentence construction and don’t have any real information)

- *Convert to lower case*: Maintains standardization across all text and get rid of case differences
- *Remove punctuation*: remove punctuation since they don’t deliver any information

2. Text Cleaning

- *Remove number*: Similarly remove numerical information from text
- *Remove whitespaces*: Remove used spaces in the text
- *Stemming and Lemmetization*: Convert the terms into their root. E.g., Words such as smile, smiles, smiling, smiled gets converted to the root word "smile". It helps capture the intent of terms precisely

3. Feature Engineering

- Popular feature engineering methods:

  1. *n-grams* (explore when one or two or more words that occur together gives more information to the model):
     - In the document corpus, 1 word is called 1-gram
     - Bi-gram is 2 words, etc.

  2. *TF-IDF*: Term Frequency – Inverse Document Frequency (from a document corpus, a learning algorithm gets more information from rarely occurring terms than frequently occurring terms)

3. Feature Engineering

- Popular feature engineering methods:
  2. *TF-IDF*: Term Frequency – Inverse Document Frequency
     - TF: Frequency of term in a document/ all terms in a document
     - IDF: ratio of log(total documents in the corpus/number of documents with the term in the corpus)
     - TF-IDF = TF x IDF

Inverse Document Frequency

- Approaches for obtaining topics from text:
  - Term Frequency: Measure of commonness of a term. Measures how many times a term occurs in the corpus
  - Inverse Document Frequency: Measure of rareness of a term (uniqueness)

$$idf = \log\left(\frac{n}{df}\right)$$

Term of Interest	No. of Documents Containing the term	Total Docs/Docs Containing Terms	IDF Values for these Terms
a	100,000,000	1	-
health	1,000,000	100	2
diagnosis	100,000	1000	3
Cancer	1000	100,000	5

3. Feature Engineering

- Popular feature engineering methods:

  3. *Cosine Similarity*: This helps to find similar documents (commonly used distance metric used in text analysis.)

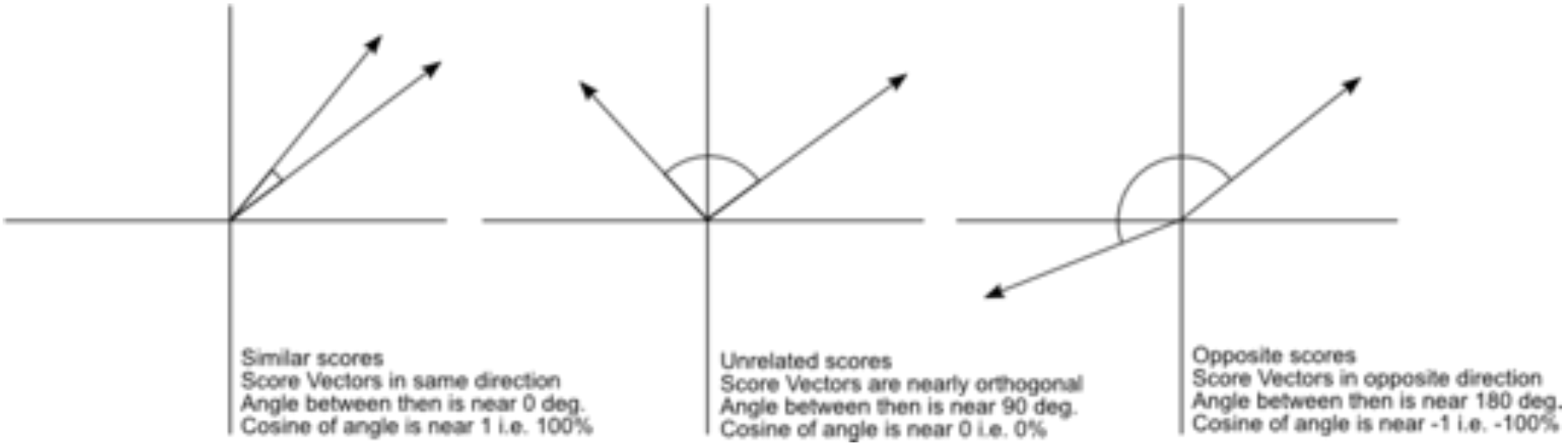
     - For 2 vectors A and B of length n each, cosine similarity can be calculated as a dot product of two unit vectors. The two vectors are arrays containing the word counts of two documents.

$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos\theta$$

$$\cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$

3. Feature Engineering

- Popular feature engineering methods:
  3. *Cosine Similarity*:



Each dimension corresponds to a word in the document, the cosine similarity captures the orientation (the angle) of the documents, not the magnitude. Smaller angle = more similarity

3. Feature Engineering

- Popular feature engineering methods:
  3. *Cosine Similarity*:

Document	team	coach	hockey	baseball	soccer	penalty	score	win	loss	season
Document1	5	0	3	0	2	0	0	2	0	0
Document2	3	0	2	0	1	1	0	1	0	1
Document3	0	7	0	2	1	0	0	3	0	0
Document4	0	1	0	0	1	2	2	0	3	0

Cosine Similarity between first two term frequency vectors (x.y)
=5x3 + 0x0 + 3x2 + 0x0 + 2x1 + 0x1 + 0x0 +2x1 + 0x0 + 0x1 = 25

$||x|| = \sqrt{5^2 + 0^2 + 3^2 + \cdots} = 6.48$

$||y|| = \sqrt{3^2 + 0^2 + 2^2 + \cdots} = 4.12$

Cosine(x,y)=0.94

3. Feature Engineering

- Popular feature engineering methods:

  4. *Jaccard Similarity*: Distance metric

  - For a given two vectors (A and B), it can be calculated as ratio of (terms which are available in both vectors / terms which are available in either of the vectors).
  - It's formula is: $(A \cap B)/(A \cup B)$. To create features using distance metrics, first create cluster of similar documents and assign a unique label to each document in a new column.

3. Feature Engineering

- Popular feature engineering methods:

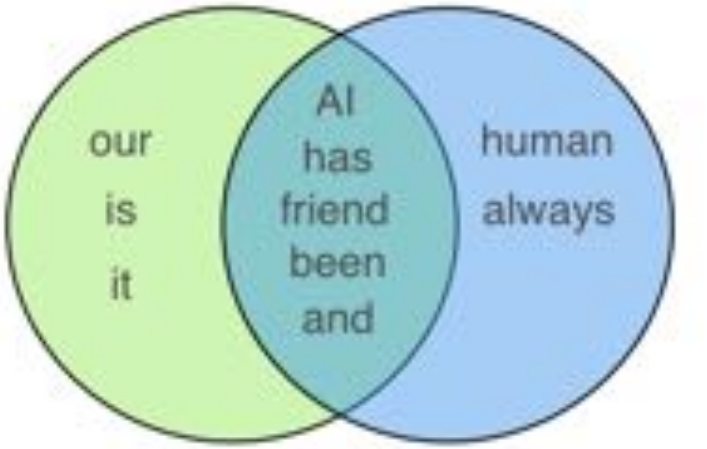
  4. *Jaccard Similarity*:

  Sentence 1: AI is our friend and it has been friendly

  Sentence 2: AI and humans have always been friendly

  We first do lemmatization (convert words to their roots: friend and friendly – "friend", has and have = "has"

J= 5/ (5+3+2)
=0.5



DUKE FUQUA SCHOOL OF BUSINESS

3. Feature Engineering

- Popular feature engineering methods:

  5. *Levenshtein Distance* : Distance measure that tells you how different two strings are. Higher number = more difference.

  Distance between KITTEN and SITTING is 3 (min of 3 edits are required to change to another):

  - kitten > **s**itten (substitution of s for k)
  - sitten > sitt**i**n (substitution of i for e)
  - sittin > sittin**g** (insertion of g at the end)

An edit = insertion of character, a deletion of character, a replacement of character

3. Feature Engineering

- Popular feature engineering methods:

  6. *Feature Hashing* : This technique implements the 'hashing trick' which helps in reducing the dimension of document matrix (lesser columns). It doesn't use the actual data, instead it uses the indexes[i,j] of the data, thus it processes data only when needed.

- *John likes to watch movies.*
- *Mary likes movies too.*
- *John also likes football*

Term	Index
John	1
likes	2
to	3
watch	4
movies	5
Mary	6
too	7
also	8
football	9

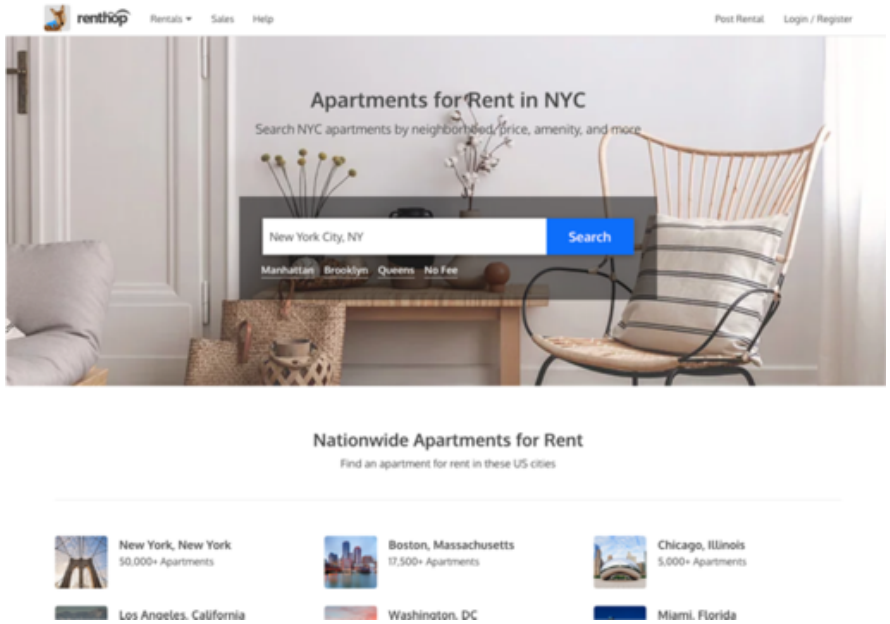
4. Model Building

- After raw data is cleaned and feature engineering is done, it is ready for model building
- Not all ML algorithms work well on text data. Some algorithms perform well:
  - Naiive Bayes
  - Deep Neural Network

Example: Kaggle

- Predict the popularity of apartment rental listing based on given features

Finding the perfect place to call your new home should be more than browsing through endless listings. RentHop makes apartment search smarter by using data to sort rental listings by quality.

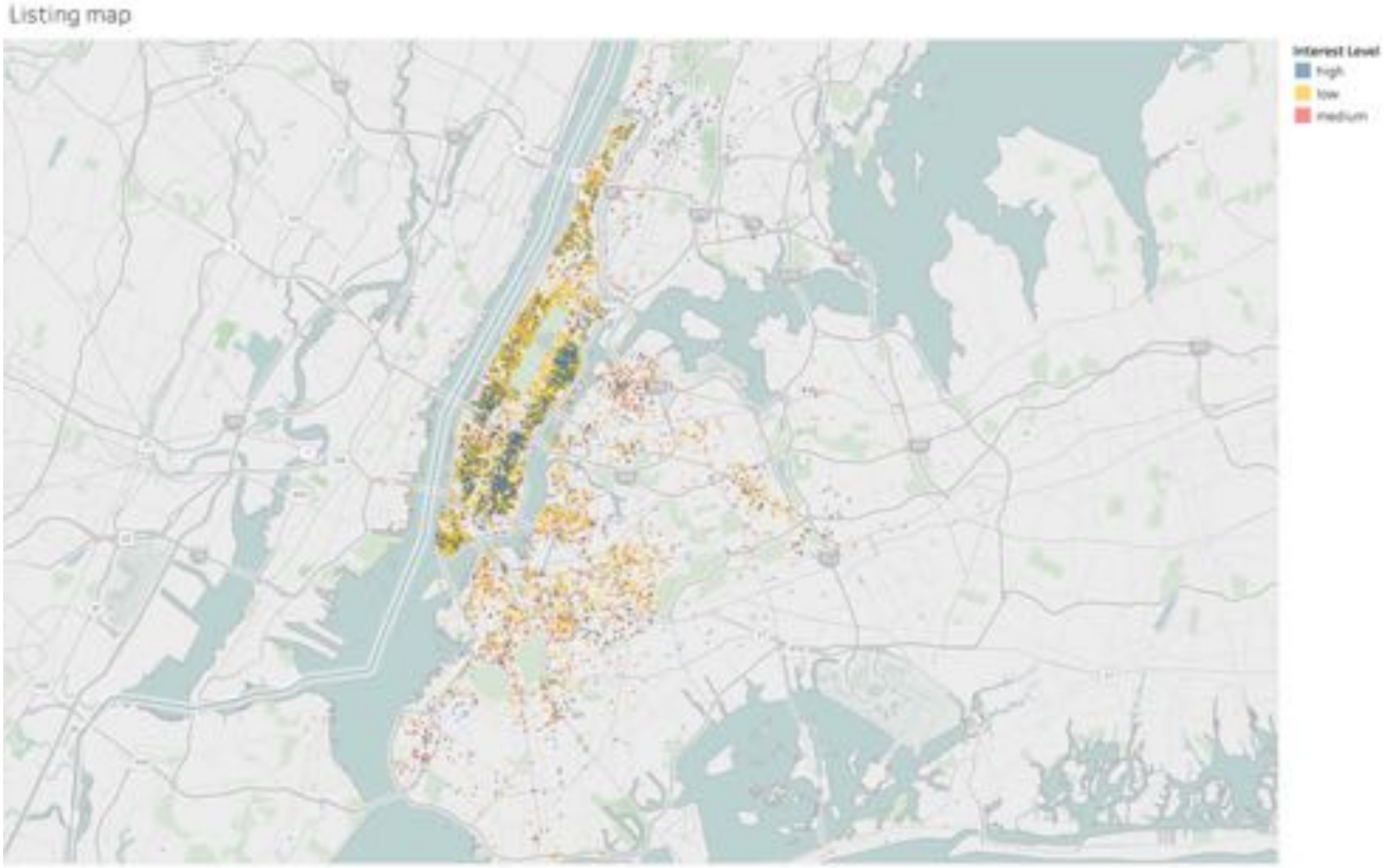


DUKE FUQUA SCHOOL OF BUSINESS

Problem

- Our challenge is to predict the interest level in an apartment rental listing.
- Rental listings in and around NYC between April and June 2016
  - 49,000 labeled entries in training set
  - Test data = 74,000.
  - Target = interest level = *high (7.8%)*, *medium* (22.7%) and *low (69.5%) of training set*
- The scope of this problem represents a typical machine learning challenge faced by many organizations. Data consists of numerical, continuous, ordinal, and categorical variables. It has pictures, descriptions of the units and geospatial information.

Target



FUQUA
SCHOOL OF BUSINESS

R Packages

```
#Load Libraries
path <- "~/Desktop/NLP/"
setwd(path)
library(data.table)
library(jsonlite)
library(rjson)
library(purrr)
library(RecordLinkage)
library(stringr)
library(tm)
library(NLP)
```

Data Table

```
#load data
traind <- fromJSON(file="train.json")
test <- fromJSON(file="test.json")

#convert json to data table (using map_at function from purrr package)
vars <-setdiff(names(traind), c("photos", "features"))
train <- map_at(traind, vars, unlist) %>% as.data.table()
test <- map_at(test, vars, unlist) %>% as.data.table()
```

Extract Text Features

```
train <- train[,.(listing_id,features,
description,street_address,display_address,interest_level)]

test <-
test[,.(listing_id,features,street_address,display_address,desc
ription)]

#understanding data

dim(train)

[1] 49352     6

dim(test)

[1] 74659     5
```

Understand Data

#understanding data

head(train)

```
   listing_id                                                                features
1:    7170325 Dining Room,Pre-War,Laundry in Building,Dishwasher,Hardwood Floors,Dogs Allowed,...
2:    7092344              Doorman,Elevator,Laundry in Building,Dishwasher,Hardwood Floors,No Fee
3:    7158677     Doorman,Elevator,Laundry in Building,Laundry in Unit,Dishwasher,Hardwood Floors
4:    7211212                                                                          <list>
5:    7225292                                Doorman,Elevator,Fitness Center,Laundry in Building
6:    7226687                              Doorman,Elevator,Loft,Dishwasher,Hardwood Floors,No Fee

description
1:
Spacious 1 Bedroom 1 Bathroom in Williamsburg!Apartment Features:- Renovated Eat in Kitchen With Dishwasher- Renovated Bathroom- Beautiful Hardwood
Floors- Lots of Sunlight- Great Closet Space- Freshly Painted- Heat and Hot Water Included- Live in Super Nearby L, J, M & G Trains !<br /><br />Con
tact Information:Kenneth BeakExclusive AgentC: 864-692-8838Email: kagglemanager@renthop.com, Text or Email to schedule a private viewing!<br /><br /
><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><p><a  website_redacted
2:                                                                                BRAND NEW GUT RENOVATED TRUE 2 BEDROOMFind yourself and your home
in the center of it all. Steps from Grand Central Station, at the epicenter of Manhattan, The Centra combines convenience and luxury to create a per
fectly balanced living experience. Offering newly renovated over sized apartment layouts.<br /><br />Full Time DoormanElevatorNewly Renovated Hallwa
ysLaundry in BuildingOn-Site Parking Garage<br /><br />I operate with the utmost care and integrity. The client is my #1 priority. Contact me for a
viewing of the great apartment, I'm more than confident we'll find a place for you to call home.Call/Text Keon: Email: If you require a move within
```

Understand Data

#understanding data

head(train)

```
5:
Over-sized Studio w abundant closets. Available Immediately.<br /><br />Rent Stabilized. Shares: Case By Case. Minimum Term: 12 months. Income Requi
red: 45x Rent. Guarantors Allowed.Guarantors Income Required: 75x Rent. No Corporate Guarantors.<br /><br />DERRICK OMANE 232-463-6268 kagglemanager
@renthop.com<br /><br /><p><a  website_redacted
6: This spectacular converted 3 bed apartment allows  features include fully equipped kitchen with tons of cabinetry space, dishwasher and microwave
. Huge layout with ample closet space. breath taking views and tons of sunlight. Building amenities include gracious lobby with 24 hour Doorman and
concierge service. Laundry service valet is provided as well as in building laundry room--LOW FEE UNIT--<br /><br />Union Square is the location of
several wonderful shops, theaters, and restaurants. Some of the many noteworthy dining establishments include The Grey Dog?s Caf?, Blue Water Grill,
Coffee Shop, Republic, and Max Brenner. Another valued attraction is the neighborhood?s green market that features dozens of local farmers, bakers,
brewers, artists, and businesses.<br /><br />For a private tour call Jamie fields at 832-568-9993  or email kagglemanager@renthop.com<br /><br /><p>
<a  website_redacted
           street_address     display_address interest_level
1:     145 Borinquen Place 145 Borinquen Place         medium
2:          230 East 44th           East 44th            low
3:    405 East 56th Street    East 56th Street         medium
4: 792 Metropolitan Avenue Metropolitan Avenue         medium
5:    340 East 34th Street    East 34th Street            low
6:    145 East 16th Street    East 16th Street            low
```

Understand Data

#understanding data

head(test)

	listing_id	features	street_address	display_address
1:	7142618	Elevator,Laundry in Building,Laundry in Unit,Dishwasher,Hardwood Floors,Outdoor Space	99 Suffolk Street	Suffolk Street
2:	7210040	Pre-War,Dogs Allowed,Cats Allowed	176 Thompson Street	Thompson Street
3:	7174566	Pre-War,Dogs Allowed,Cats Allowed	115 Sullivan Street	Sullivan Street
4:	7191391	Hardwood Floors,Dogs Allowed,Cats Allowed	23 Jones Street	Jones Street
5:	7171695	Roof Deck,Doorman,Elevator,Fitness Center,Pre-War,Laundry in Building,...	20 Exchange Place	Exchange Place
6:	7225206	Cats Allowed,Dogs Allowed,No Fee,Doorman,Elevator,Fitness Center,...	650 W 42nd St.	W 42nd St.

description

1:

Large with awesome terrace--accessible via bedroom and living room. Unique find in the LES.Apartment Features:-Large terrace via bedroom and living room-Hardwood floors-Newly renovated -Granite counter top-Breakfast Bar-Ample counter space and storage-Dishwasher-Great Lighting Neighborhood Features:-A few blocks from Whole Foods-1 block from the J, Z and M subway-All the restaurants and night life the Lower East Side is known for (Hotel Chantel, DL, Pianos)Call/txt/Email James to set up a showing:kagglemanager@renthop.com

<p><a website_redacted

2:

Understand Data

sapply(train,class)

```
 listing_id        features     description  street_address display_address  interest_level
  "numeric"          "list"     "character"     "character"     "character"     "character"
```

sapply(test,class)

```
listing_id        features  street_address display_address     description
 "numeric"          "list"     "character"     "character"     "character"
```

Understanding Data

1.The train data has 49352 rows and 6 columns.
2.The test data has 74659 rows and 5 columns.
3. *interest_level* is the dependent variable
4. listing_id variable has unique value for every listing. It is the identifier variable.
5. *features* comprises of a list of features for every listing_id
6. *description* refers to the description of a listing_id provided by the agent
7. *Street address* and display address refers to the address of the listed apartment

Feature Engineering

- Count of number of features per listing
- Count of number of words in the description
- Count of total length of description
- Similarity between street and display address using Levenshtein distance

```
#Join Train and Test data
test[,interest_level := "None"]
tdata <- rbindlist(list(train,test))
```

Feature Engineering

```
#fill empty values in the list
tdata[,features := ifelse(map(features,
is_empty),"aempty",features)]
#count number of features per listing
tdata[,feature_count := unlist(lapply(features, length))]
#count number of words in description
tdata[,desc_word_count := str_count(description,pattern =
"\\w+")]
#count total length of description
tdata[,desc_len := str_count(description)]
#similarity between address
tdata[,lev_sim :=
levenshteinDist(street_address,display_address)]
```

Extract Variables from Features

```
dim(tdata)
[1] 124011     10
#extract variables from features
fdata <- data.table(listing_id = rep(unlist(tdata$listing_id), lapply(tdata$features, length)), features = unlist(tdata$features))
head(fdata)
```

```
   listing_id           features
1:    7170325        Dining Room
2:    7170325           Pre-War
3:    7170325 Laundry in Building
4:    7170325         Dishwasher
5:    7170325    Hardwood Floors
6:    7170325       Dogs Allowed
```

Features Transformation

#convert features to lower case

fdata[,features := unlist(lapply(features, tolower))]

#calculate count for every feature

fdata[,count := .N, features]

fdata[order(count)][1:20]

	features	count
1:	Private Entrances	1
2:	Top Floor Unit	1
3:	XXL Windows	1
4:	Sleep Loft	1
5:	own entrance	1
6:	oak cabinets	1
7:	** HOLY NO FEE DEAL BATMAN! * OVERSIZED 2BR HOME * SPARKLING CLEAN & BRITE * HEART OF GREENPOINT * NEAR THE PARK & TRAINS **	1
8:	Recreational Room	1
9:	Washer	1
10:	!!!!LOW FEE!!!!	1
11:	** CHELSEA BABY! * MASSIVE 2BR SUPER SHARE * ALL MODERN & NEW * ELEV /LNDRY BLDG **	1
12:	Loft Area	1
13:	Flex two bed	1
14:	** SPRAWLING STUDIO HOME * DISHWASHER * EXPOSED BRICK * GUT RENOVATED * PETS OK * STEPS TO THE PARK **	1
15:	Control & pay for your own heat with your own thermostat	1
16:	NO alternate side parking rules in neighborhood	1
17:	Previous tenant found easy on-street parking.	1
18:	Citi Bike Station	1
19:	Garage Parking!	1
20:	** COURT SQUARE GEM! * SPRAWLING SUNDRENCHED 2BR HOME * CUSTOM FINISHES * DISHWASHER * FIREPLACES * EAT-IN KITCHEN * BAY WINDOWS **	1

Features Transformation

```
#keep features which occur 100 or more times
fdata <- fdata[count >= 100]
#Convert each feature into a separate column to use those as variables in model training (dcast function)
fdata <- dcast(data = fdata, formula = listing_id ~ features, fun.aggregate = length, value.var = "features")
dim(fdata)
[1] 123764     96
```

Extract Features from Description

```
#create a corpus of descriptions
text_corpus <- Corpus(VectorSource(tdata$description))
#check first 4 documents
inspect(text_corpus[1:4])
```

```
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 4

[1] Spacious 1 Bedroom 1 Bathroom in Williamsburg!Apartment Features:- Renovated Eat in Kitchen With Dishwasher- Renovated Bathroom- Beautiful Hardw
ood Floors- Lots of Sunlight- Great Closet Space- Freshly Painted- Heat and Hot Water Included- Live in Super Nearby L, J, M & G Trains !<br /><br /
>Contact Information:Kenneth BeakExclusive AgentC: 864-692-8838Email: kagglemanager@renthop.com, Text or Email to schedule a private viewing!<br /><
br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><p><a  website_redacted
[2] BRAND NEW GUT RENOVATED TRUE 2 BEDROOMFind yourself and your home in the center of it all. Steps from Grand Central Station, at the epicenter of
Manhattan, The Centra combines convenience and luxury to create a perfectly balanced living experience. Offering newly renovated over sized apartmen
t layouts.<br /><br />Full Time DoormanElevatorNewly Renovated HallwaysLaundry in BuildingOn-Site Parking Garage<br /><br />I operate with the utmos
t care and integrity. The client is my #1 priority. Contact me for a viewing of the great apartment, I'm more than confident we'll find a place for
you to call home.Call/Text Keon: Email: If you require a move within 30 days write "URGENT" in the subject email or text message to be taken with hi
gh priority.<br /><br />One Month Free - net effective rent listed<p><a  website_redacted
[3] **FLEX 2 BEDROOM WITH FULL PRESSURIZED WALL**Looking for the perfect apartment in Midtown East - Sutton Place? Come check out the beautiful apar
tment in this prime location! **Mid 50's and 1st Ave** Elevator Building with 24-hour doorman, laundry room, and bike room!LARGE living space with K
```

Corpus

#the corpus is a list object in R of type CORPUS

print(lapply(text_corpus[1:2], as.character))

```
[[1]]
[1] "Spacious 1 Bedroom 1 Bathroom in Williamsburg!Apartment Features:- Renovated Eat in Kitchen With Dishwasher- Renovated Bathroom- Beautiful Hard
wood Floors- Lots of Sunlight- Great Closet Space- Freshly Painted- Heat and Hot Water Included- Live in Super Nearby L, J, M & G Trains !<br /><br
/>Contact Information:Kenneth BeakExclusive AgentC: 064-692-8838Email: kagglemanager@renthop.com, Text or Email to schedule a private viewing!<br />
<br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><br /><p><a  website_redacted "

[[2]]
[1] "BRAND NEW GUT RENOVATED TRUE 2 BEDROOMFind yourself and your home in the center of it all. Steps from Grand Central Station, at the epicenter o
f Manhattan, The Centra combines convenience and luxury to create a perfectly balanced living experience. Offering newly renovated over sized apartm
ent layouts.<br /><br />Full Time DoormanElevatorNewly Renovated HallwaysLaundry in BuildingOn-Site Parking Garage<br /><br />I operate with the utm
ost care and integrity. The client is my #1 priority. Contact me for a viewing of the great apartment, I'm more than confident we'll find a place fo
r you to call home.Call/Text Keon: Email: If you require a move within 30 days write \"URGENT\" in the subject email or text message to be taken wit
h high priority.<br /><br />One Month Free - net effective rent listed<p><a  website_redacted "
```

Cleaning the Corpus

```
#Clean the data of "br" - it it just noise
dropword <- "br"
#remove br
text_corpus <-
tm_map(text_corpus,removeWords,dropword)
print(as.character(text_corpus[[1]]))
```

```
[1] "Spacious 1 Bedroom 1 Bathroom in Williamsburg!Apartment Features:- Renovated Eat in Kitchen With Dishwasher- Renovated Bathroom- Beautiful Hard
wood Floors- Lots of Sunlight- Great Closet Space- Freshly Painted- Heat and Hot Water Included- Live in Super Nearby L, J, M & G Trains !< /><  />Co
ntact Information:Kenneth BeakExclusive AgentC: 064-692-8838Email: kagglemanager@renthop.com, Text or Email to schedule a private viewing!< /><  /><
/><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><p><a  website_redacted "
```

Clean Corpus

```
#convert to lower case
text_corpus <- tm_map(text_corpus, tolower)
print(as.character(text_corpus[[1]]))
```

```
[1] "spacious 1 bedroom 1 bathroom in williamsburg!apartment features:- renovated eat in kitchen with dishwasher- renovated bathroom- beautiful hard
wood floors- lots of sunlight- great closet space- freshly painted- heat and hot water included- live in super nearby l, j, m & g trains !< /><  />co
ntact information:kenneth beakexclusive agentc: 064-692-8838email: kagglemanager@renthop.com, text or email to schedule a private viewing!< /><  /><
/><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><  /><p><a  website_redacted "
```

Clean Corpus

```
#remove punctuation
text_corpus <- tm_map(text_corpus, removePunctuation)
print(as.character(text_corpus[[1]]))
```

```
[1] "spacious 1 bedroom 1 bathroom in williamsburgapartment features renovated eat in kitchen with dishwasher renovated bathroom beautiful hardwood
floors lots of sunlight great closet space freshly painted heat and hot water included live in super nearby l j m  g trains   contact informationken
neth beakexclusive agentc 0646928838email kagglemanagerrenthopcom text or email to schedule a private viewing                pa  websiteredacted "
```

```
#remove numbers
text_corpus <- tm_map(text_corpus, removeNumbers)
print(as.character(text_corpus[[1]]))
```

```
[1] "spacious  bedroom  bathroom in williamsburgapartment features renovated eat in kitchen with dishwasher renovated bathroom beautiful hardwood fl
oors lots of sunlight great closet space freshly painted heat and hot water included live in super nearby l j m  g trains   contact informationkenne
th beakexclusive agentc email kagglemanagerrenthopcom text or email to schedule a private viewing                pa  websiteredacted "
```

Clean Corpus

```
#remove whitespaces
text_corpus <- tm_map(text_corpus, stripWhitespace)
print(as.character(text_corpus[[1]]))
```

```
[1] "spacious bedroom bathroom in williamsburgapartment features renovated eat in kitchen with dishwasher renovated bathroom beautiful hardwood floor
s lots of sunlight great closet space freshly painted heat and hot water included live in super nearby l j m g trains contact informationkenneth beak
exclusive agentc email kagglemanagerrenthopcom text or email to schedule a private viewing pa websiteredacted "
```

Clean Corpus

#remove stopwords

text_corpus <- tm_map(text_corpus, removeWords, c(stopwords('english')))

print(as.character(text_corpus[[1]]))

```
[1] "spacious bedroom bathroom  williamsburgapartment features renovated eat  kitchen  dishwasher renovated bathroom beautiful hardwood floors lots
sunlight great closet space freshly painted heat  hot water included live  super nearby l j m g trains contact informationkenneth beakexclusive agent
c email kagglemanagerrenthopcom text  email  schedule  private viewing pa websiteredacted "
```

Stopwords

#The “stop_words” dictionary

```
data("stop_words")
head(stop_words)
```

```
# A tibble: 6 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
```

Word Stemming and Stem Completion

You may need to perform word stemming and word completion in text mining during data preprocessing

Word stemming: reduces words to unify across document (e.g., the stem of *computational*, *computers* and *computation* is **comput**)

Stem completion: Reconstruct the words: Since **comput** is not a word, we reconstruct it to *computer*

The *tm* package provides a stemDocument() function to get to a word's root. This function either takes a character vector and returns a character vector or takes a PlainTextDocument and returns a PlainTextDocument

E.g., stemDocument(c('computational', 'computers', 'computation'))

returns 'comput' 'comput' 'comput'

stemCompletion(character vector, completion dictionary)

#stemCompletion accepts a character vector and a completion dictionary or a Corpus. The completion dictionary must contain the word computer to reconstruct all instances of comput

DUKE FUQUA SCHOOL OF BUSINESS

Word Stemming and Stem Completion

Example:

```
complicate <- c('complicated', 'complication', 'complicatedly')
stem_doc <- stemDocument(complicate)
#the above word stemming code stems everything in complicate to "complic" which is the root word

#create the completion dictionary: comp_dict
comp_dict <- c('complicate')

#perform stem completion: complete_text
complete_text <-stemCompletion(stem_doc, comp_dict)

#Print complete_text
```

```
> complete_text
     complic      complic      complic
"complicate" "complicate" "complicate"
>
```

DUKE FUQUA SCHOOL OF BUSINESS

Stemming

```
#convert to text document
text_corpus <- tm_map(text_corpus, PlainTextDocument)

#perform stemming - this should always be performed after text doc conversion
text_corpus <- tm_map(text_corpus, stemDocument,language = "english")
print(as.character(text_corpus[[1]]))
text_corpus[[1]]$content
```

Stemming

[2] "brand new gut renov true bedroomfind home center step grand central station epicent manhattan centra combin conveni luxuri creat perfect bal anc live experi offer newli renov size apart layout full time doormanelevatornewli renov hallwayslaundri buildingonsit park garag oper utmost care i ntegr client prioriti contact view great apart im confid well find place call homecalltext keon email requir move within day write urgent subject em ail text messag taken high prioriti one month free net effect rent listedpa websiteredact"

[3] "flex bedroom full pressur walllook perfect apart midtown east sutton place come check beauti apart prime locat mid s st ave elev build hour doorman laundri room bike roomlarg live space king bedroombeauti larg kitchen stainless steel applianc includ dishwash stun modern bathroom ampl amo unt closet space throughout entir apart enough space fit everyth need apart short distanc shop restaur public transport em train dont miss great apa rt act quick calltext edan schedul privat view today pa websiteredact"

[4] "brand new bedroom bath apartmentenjoy follow apart featur rent modern design bathroom w deep spa soak tub room room acheat real oak hardwood floor rain forest shower head ss steel applianc w chef gas cook oven lg fridg washer dryer apt cabl internet readi granit counter top kitchen w lot cabinet storag spaceit just block l train dont miss sever great apart immedi area addit inform pa websiteredact"

[5] "overs studio w abund closet avail immedi rent stabil share case case minimum term month incom requir x rent guarantor allowedguarantor incom requir x rent corpor guarantor derrick oman kagglemanagerrenthopcom pa websiteredact"

[6] "spectacular convert bed apart allow featur includ fulli equip kitchen ton cabinetri space dishwash microwav huge layout ampl closet space br eath take view ton sunlight build amen includ gracious lobbi hour doorman concierg servic laundri servic valet provid well build laundri roomlow fee unit union squar locat sever wonder shop theater restaur mani noteworthi dine establish includ grey dog caf blue water grill coffe shop republ max b renner anoth valu attract neighborhood green market featur dozen local farmer baker brewer artist busi privat tour call jami field email kagglemanag errenthopcom pa websiteredact"

[7] "amaz deal brand new renov huge true bed apart door space locat fabul east villag unit offer ton light washerdry unit huge live room larg bed room expos brick interior separ kitchen granit counter build offer tight secur virtual doorman beauti expos brick well maintain lobbi contact imani

...

[997] "current offer one month free broker fee paid list like one schedul view pleas visit wwwnycrentalmarketcom featur elev build hr doorman apart balconi splendid view unit featur lot window throughout entir apart kitchen finish includ white granit countertop applianc builtin microwav custom c abinetri wireless internet avail spacious closet parquet wood floor full kitchen includ stainless steel applianc includ dishwash allwhit tile bathro om includ marbl tub larg mirror build amen build doorman elev laundri room basement locat perfect area real block away central park whole food duan read associ supermarket chase capit one bankshom good cvs rite aid sephora tjmax michael starbuck crumb petco train express b c"

[998] "prime lic locat step queensborough plaza conveni subway n q train one stop away lexington ave th street easi access time squar hell kitchen via cross town th train ultra luxuri fantast jr bedroom resid featur larg live room great bedroom full marbl bathroom massiv open chef kitchen stain less steel applianc oak strip floor privat washer dryer balconi floortoceil window overs window h doorman laundri elev park garag gym loung pool res id loung net effect rent list pa websiteredact"

[999] ""

[1000] "sparkl kingsbridg build brand new renovationsemail call asap check amaz deal wont lastkagglemanagerrenthopcom nofeetransport block trainapar t detailsgut renov throughoutexpos brickstainless steel applianc dishwasherwash amp dryer unitton natur lightbik storag buildingheat amp hot water i ncludedclos best shop park restaur kingsbridgethi unit jam problem can show apart nyc work agent simplifi movei work car less walk find perfect apar t"

[reached getOption("max.print") -- omitted 123011 entries]

Convert to Term-Document Matrix

- A corpus (collection of documents) can be converted into a Term Document Matrix

	W1	W2	W3	Wn
D1	0	2	1	3
D2	1	4	0	0
D3	0	2	3	1
Dm	1	1	3	0

n is the size of vocabulary (words/terms); m=number of documents

Corpus to Matrix

```
#convert to document term matrix
docterm_corpus <- DocumentTermMatrix(t_corpus)
dim(docterm_corpus)
[1] 124011  52816

#remove terms that are 95% sparse or more
new_docterm_corpus <-
removeSparseTerms(docterm_corpus,sparse = 0.95)
dim(new_docterm_corpus)
[1] 124011     87
```

Explore and Visualize High Frequency Features

```
#Find frequent terms
colS <- colSums(as.matrix(new_docterm_corpus))
length(colS)
[1] 87

#Count of the terms
doc_features <- data.table(name = attributes(colS)$names,
count = colS)
```

Explore and Visualize High Frequency Features

#most frequent and least frequent words

doc_features[order(-count)][1:10] #top 10 most frequent words

	name	count
1:	apart	46682
2:	street	44986
3:	websiteredact	35409
4:	build	34821
5:	bedroom	34299
6:	floor	32532
7:	kitchen	31925
8:	new	27283
9:	locat	24920
10:	view	24901

doc_features[order(count)][1:10] #least 10 frequent words

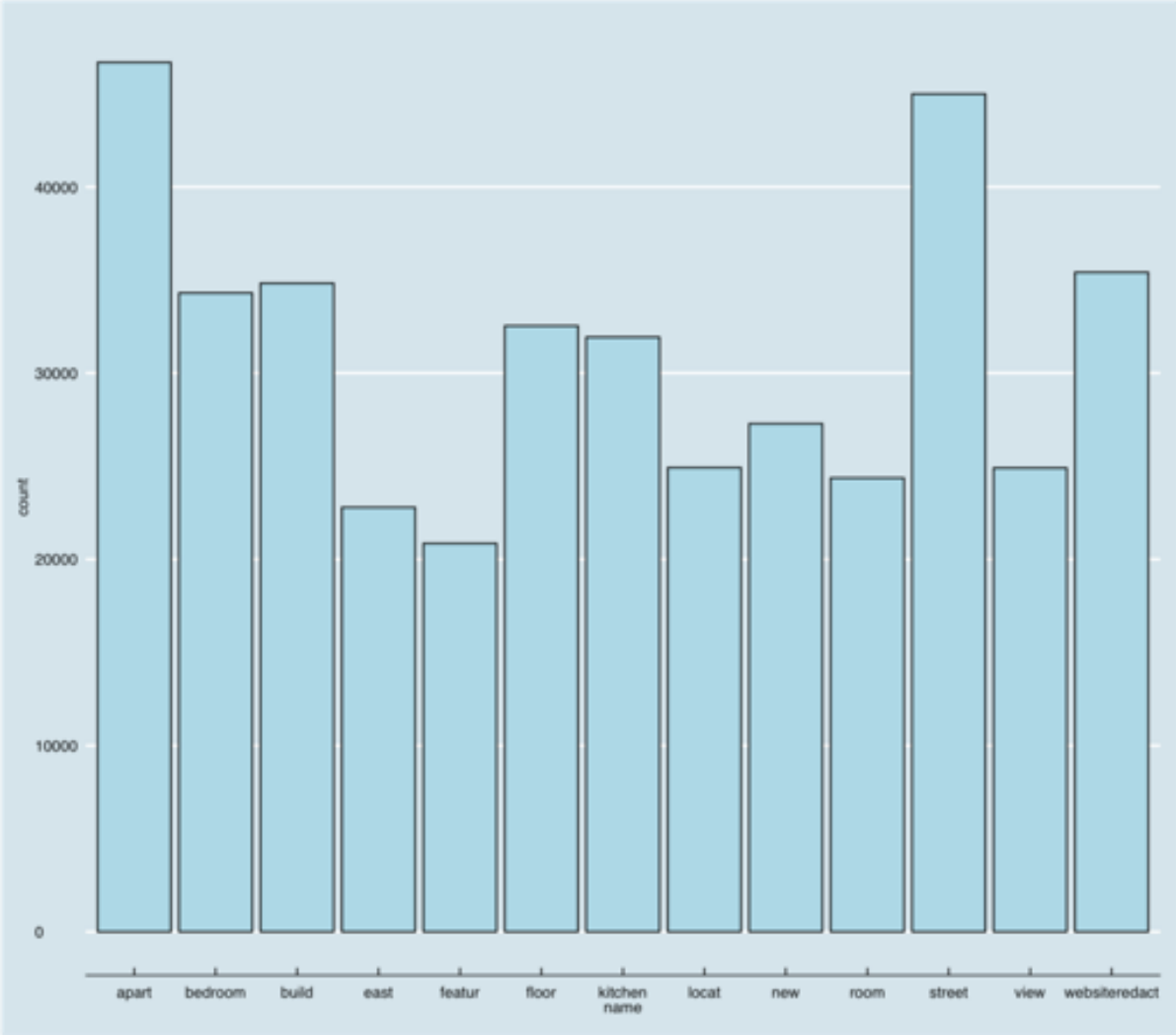
	name	count
1:	show	6772
2:	deck	6799
3:	hour	6878
4:	hous	6917
5:	close	6983
6:	pleas	7032
7:	ave	7047
8:	walk	7110
9:	text	7172
10:	step	7211

Visualize the terms

```
#Visualize the terms with simple plots
library(ggplot2)
library(ggthemes)

ggplot(doc_features[count>20000],aes(name, count)) +
  geom_bar(stat = "identity",fill='lightblue',color='black')+
  theme(axis.text.x = element_text(angle = 45, hjust = 1))+
  theme_economist()+
  scale_color_economist()
```

Plot of Item Frequency



DUKE FUQUA SCHOOL OF BUSINESS

Association between terms

```
findAssocs(new_docterm_corpus,"street",corlimit = 0.5)
$street
numeric(0)

findAssocs(new_docterm_corpus,"new",corlimit = 0.5)
$new
york
0.63
```

Word Clouds

```
library(wordcloud)
wordcloud(names(colS), colS, min.freq = 100, scale = c(6,.1), colors
= brewer.pal(6, 'Dark2'))
```

DUKE FUQUA SCHOOL OF BUSINESS

Word Cloud

wordcloud(names(colS), colS, min.freq = 5000, scale = c(6,.1), colors = brewer.pal(6, 'Dark2'))

DUKE FUQUA SCHOOL OF BUSINESS

Modeling: Preparing Data

```
#Convert matrix into data frame and merge with new features from "feature" column

#create data set for training

processed_data <- as.data.table(as.matrix(new_docterm_corpus))


#data1: combining the data using listing_id

data_one <- cbind(data.table(listing_id = tdata$listing_id, interest_level = tdata$interest_level),processed_data)


#merging the features

data_one <- fdata[data_one, on="listing_id"]
```

Modeling: Preparing the Data

```
#split the data set into train and test
train_one <- data_one[interest_level != "None"]
test_one <- data_one[interest_level == "None"]
test_one[,interest_level := NULL]
rm(data_one,processed_data)
```

XG Boost Algorithm

XGBOOST Algorithm is used for improving a model's prediction accuracy

Boosting: Family of techniques that provide power to Machine Learning Models to improve their accuracy of prediction

- Converts weak learners to strong learners

Boosting Example

Classify Emails as SPAM
or NOT SPAM

Criteria:

- Email has only one link (SPAM)
- Email has only one promotional image (SPAM)
- Email body has a sentence "you have won a prize…." (SPAM
- Email from an official domain (NOT SPAM)
- Email from known source (NOT SPAM)
- …

These rules are individually not powerful enough
to classify emails: weak learners

Boosting Example

Classify Emails as SPAM or NOT SPAM

These rules are individually not powerful enough to classify emails: weak learners

Converting weak to strong learners may require a combination of prediction of each weak learner using methods like:

- Average or weighted average of all rules
- Considering prediction has higher vote

We have five weak learners. 3 of these are voted as SPAM and 2 are "NOT SPAM". By default an email is SPAM because we have higher vote for SPAM

Boosting: How it Works?

Classify Emails as SPAM or NOT SPAM

These rules are individually not powerful enough to classify emails: weak learners

How does Boosting identify weak rules?

- We apply base ML algorithms to sample with a different distribution each time.
- Each time base learning algorithm is applied, it generates a weak prediction rule
- After many iterations, boosting algorithm combines weak rules into a strong prediction rule

DUKE FUQUA SCHOOL OF BUSINESS

Boosting: How it Works?

Classify Emails as SPAM or NOT SPAM

More focus on examples which are misclassified or have higher errors by preceding weak rules

How do we choose different sampling distributions each round?

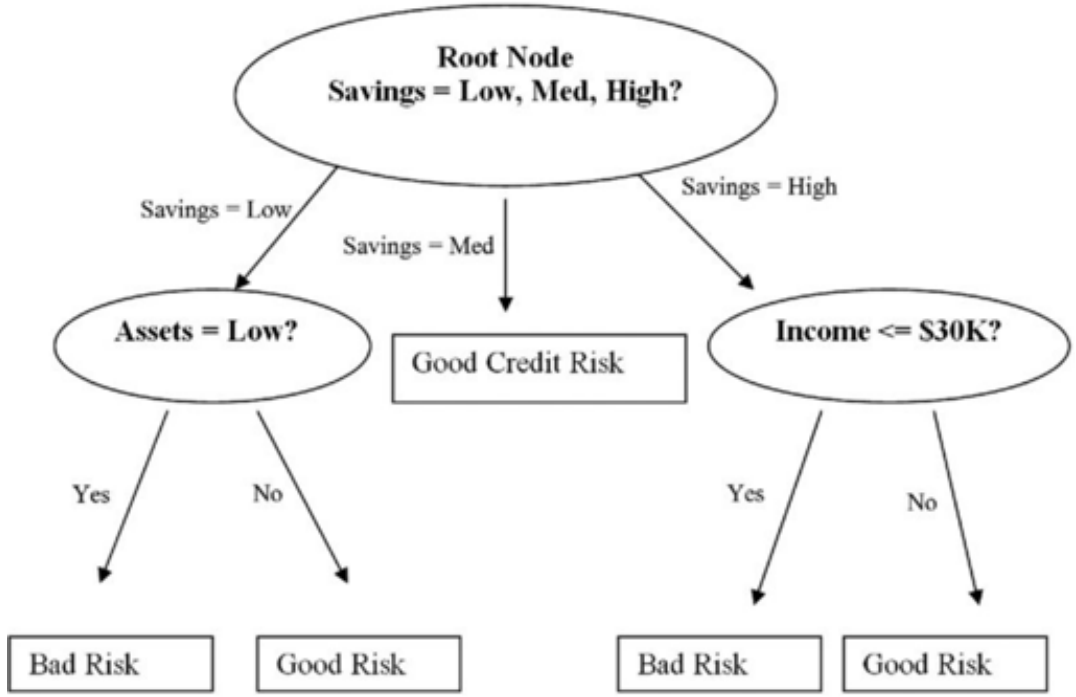
- Step1: The base ML learner takes all the distributions and assigns equal weight to each observation
- Step2: If there is any prediction error caused by first base learning algorithm, then we pay more attention to observations having prediction error
- Step3: Then we apply the next base learning algorithm
- Step4: Iterate the above steps until the accuracy doesn't change
- Step5: Combine the output from weak learners and create a strong learner which improves the prediction power of the model

DUKE FUQUA SCHOOL OF BUSINESS

Boosting Algorithms

- AdaBoost (Adaptive Boosting)
- Gradient Tree Boosting
- XGBoost

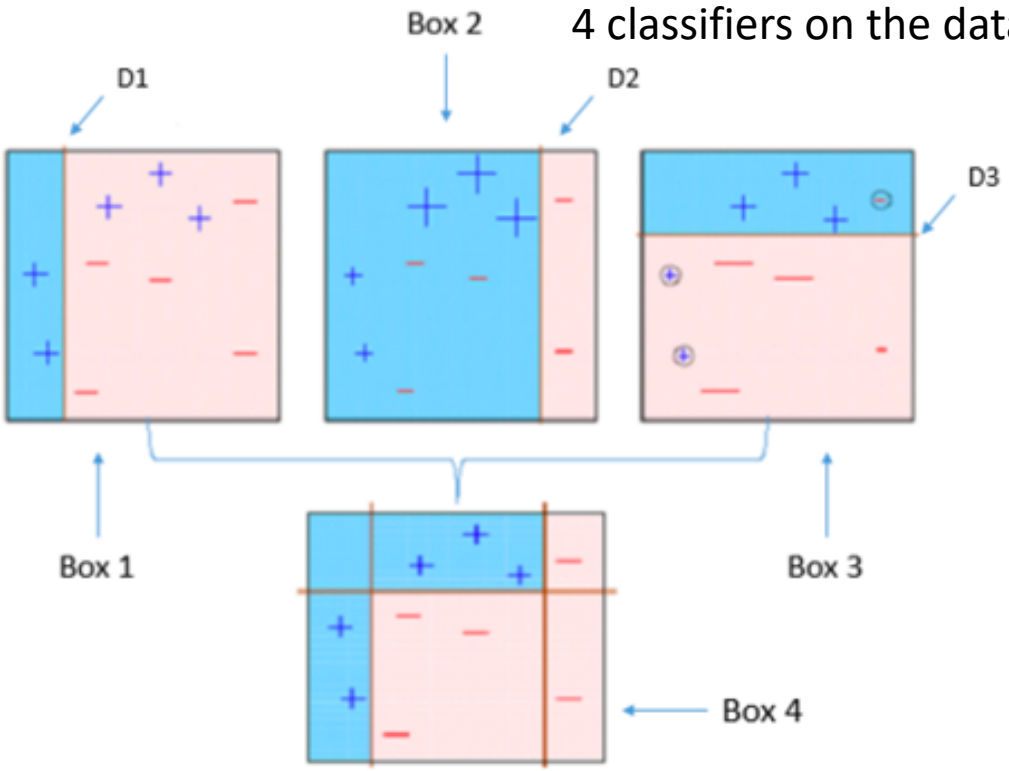
Works on the concept of Decision Trees: A **decision tree** is a decision support tool that uses a tree-like graph or model of decisions



DUKE FUQUA SCHOOL OF BUSINESS

Boosting Algorithms

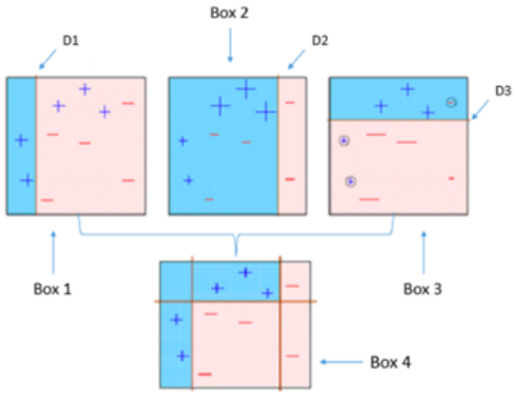
First classifier creates vertical split at D1
Anything to left of D1 is + and right is –
Misclassifies three + and those get more weight in the next split
….



Box 4= Weighted combination of all three weak classifiers

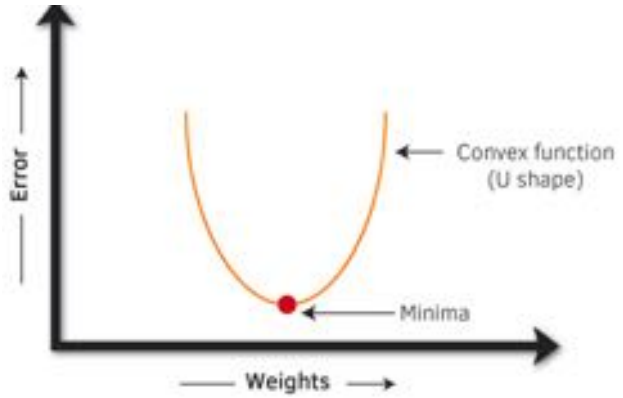
DUKE FUQUA SCHOOL OF BUSINESS

AdaBoost vs. Gradient Tree



AdaBoost identifies weak learners using high weight data points

- Gradient Boosting trains many models in sequential manner
- GB identifies weak learners same by using gradients in the loss function ($y=ax+b+e$, *e needs a special mention as it is the error term)*.
- The loss function is a measure indicating how good are model's coefficients are at fitting the underlying data.



DUKE FUQUA SCHOOL OF BUSINESS

XGBoost

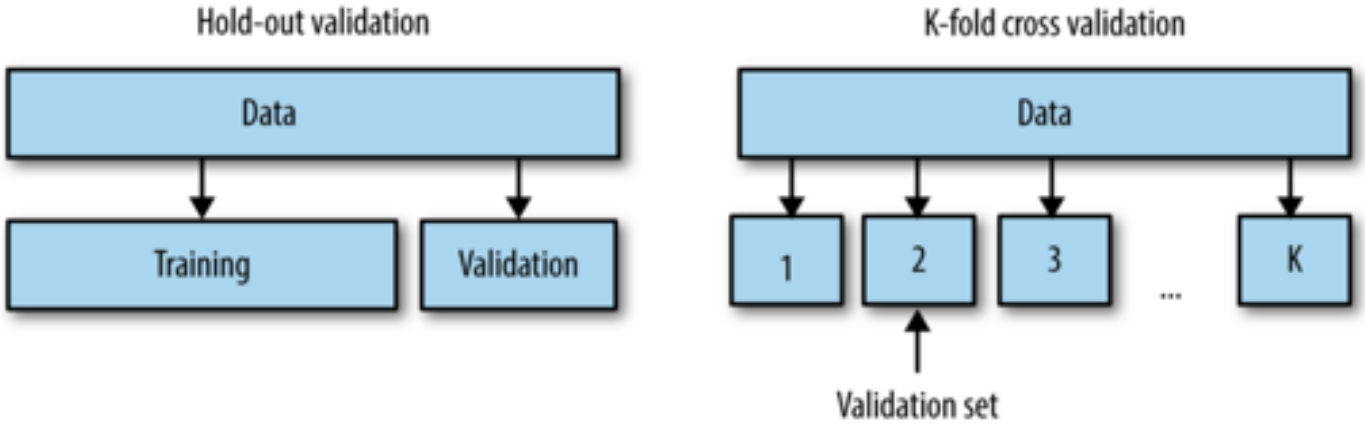
Extreme Gradient Boosting: is optimized distributed gradient boosting

- Faster and Quicker than GBM
- Enabled with parallel processing
- Enabled with regularization (technique that prevents overfitting in linear and tree models)
- Enabled cross validation
- Handles missing values internally
- Flexible: Handles regression (Booster=gblinear/gbtree), classification (booster=gbtree), ranking, and user defined evaluation metrics

About XGBoost

- It accepts dependent variable in integer format
- The data should be in matrix format
- To check model performance, we can specify a validation set such that we can check validation errors during model training (both k-fold and hold-out validation strategy can be used) For faster training, we'll use hold-out validation strategy
- The training data must not have dependent variable. It should be removed. Neither, it should have the identifier variable (listing_id)

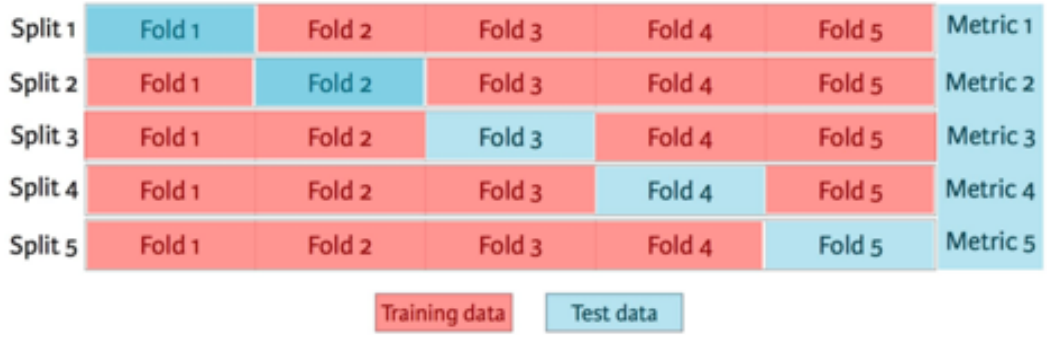
Model Performance



- Hold-Out Validation:
  - Split your dataset into "train" and "test"
  - Model is trained on "train" data and scored on "test" data
  - Generally dataset is split 60:40, 70:30, 80:20

Model Performance

- Cross validation or K-fold validation:
  - Dataset is randomly split into k groups
  - One group is used as test set and rest is training set
  - Model is trained on training set and scored on test set and repeated for each fold (until each unique group has been used as test set)
  - Five fold cross validation



DUKE FUQUA SCHOOL OF BUSINESS

Preparing the Target Variable

```
train_one[,interest_level := as.factor(interest_level)]
train_one[,interest_level :=
as.integer(as.factor(interest_level))-1]
#high = 0, low = 1, medium = 2

library(caTools)
library(xgboost)
```

Set Data for xgboost

```
#returns 60% indexes from train data (stratified splitting data)
sp <- sample.split(Y = train_one$interest_level,SplitRatio = 0.6)
#create data for xgboost
xg_val <- train_one[sp]
listing_id <- train_one$listing_id
target <- train_one$interest_level
xg_val_target <- target[sp]

d_train <- xgb.DMatrix(data = as.matrix(train_one[,-
c("listing_id","interest_level"),with=F]),label = target)
d_val <- xgb.DMatrix(data = as.matrix(xg_val[,-
c("listing_id","interest_level"),with=F]), label = xg_val_target)
d_test <- xgb.DMatrix(data = as.matrix(test_one[,-c("listing_id"),with=F]))
```

Set Data for xgboost

```
param <- list(booster="gbtree",
              objective="multi:softprob",
              eval_metric="mlogloss",
              #nthread=13,
              num_class=3,
              eta = .02,
              gamma = 1,
              max_depth = 4,
              min_child_weight = 1,
              subsample = .7,
              colsample_bytree = .5
)
```

Eta=Learning rate (default 0.3) – rate at which model learns patterns in data
Gamma= Min loss reduction required to further split lead node
Max depth=depth of tree
Child weight= no. of instances in each child node required to split
Subsample: randomly sample 0.70 of the training data for each boost
Colsample: Fraction of columns to be subsampled

Run Model

```
watch <- list(val=d_val, train=d_train)
xgb2 <- xgb.train(data = d_train,
                  params = param,
                  watchlist=watch, #monitor evaluation result
                    on all data in the list
                  # nrounds = no. of decision trees in model
                  nrounds = 500,
                  print_every_n = 10)
```

Model Validation Errors

```
[1]     val-mlogloss:1.088589   train-mlogloss:1.088610
[11]    val-mlogloss:1.003316   train-mlogloss:1.003304
[21]    val-mlogloss:0.940318   train-mlogloss:0.940283
[31]    val-mlogloss:0.893040   train-mlogloss:0.893008
[41]    val-mlogloss:0.856689   train-mlogloss:0.856630
[51]    val-mlogloss:0.828960   train-mlogloss:0.828901
[61]    val-mlogloss:0.807652   train-mlogloss:0.807596
[71]    val-mlogloss:0.791047   train-mlogloss:0.791015
[81]    val-mlogloss:0.777708   train-mlogloss:0.777666
[91]    val-mlogloss:0.767020   train-mlogloss:0.766962
[101]   val-mlogloss:0.758523   train-mlogloss:0.758461
[111]   val-mlogloss:0.751466   train-mlogloss:0.751436
[121]   val-mlogloss:0.745742   train-mlogloss:0.745721
[131]   val-mlogloss:0.740981   train-mlogloss:0.741008
[141]   val-mlogloss:0.736962   train-mlogloss:0.737004
[151]   val-mlogloss:0.733551   train-mlogloss:0.733583
[161]   val-mlogloss:0.730490   train-mlogloss:0.730523
[171]   val-mlogloss:0.727976   train-mlogloss:0.727995
[451]   val-mlogloss:0.695954   train-mlogloss:0.696213
[461]   val-mlogloss:0.695299   train-mlogloss:0.695577
[471]   val-mlogloss:0.694600   train-mlogloss:0.694898
[481]   val-mlogloss:0.693964   train-mlogloss:0.694271
[491]   val-mlogloss:0.693278   train-mlogloss:0.693573
[500]   val-mlogloss:0.692673   train-mlogloss:0.692980
```

Validation Error= 0.6929
Multigloss metric was used

Model Accuracy and Prediction

- Validation Error: 0.6929. Not good, right?
- Remember we used only two variables from the dataset:
  - Features
  - Description
- Use the non-text variables and run the model
- Also use other data mining techniques

Predicting on Test Data

```
#Create Predictions on the test data and save this as xgb_textmining.csv file

xg_pred <- as.data.table(t(matrix(predict(xgb2, d_test), nrow=3, ncol=nrow(d_test))))

colnames(xg_pred) <- c("high","low","medium")

xg_pred <- cbind(data.table(listing_id = test$listing_id),xg_pred)

fwrite(xg_pred, "xgb_textmining.csv")
```

Exercise2 – Cleaning and Creating a Word Cloud

- MLK: I have a dream speech
- The text mining package (*tm*) and the word cloud generator package (*wordcloud*) are available in R to analyze texts and to quickly visualize the keywords as a word cloud.
- Analyze **"I have a dream speech**" from "**Martin Luther King"**
- Data=**dream.txt**

Solution Exercise 2

```
# Load libraries
library("tm")
library("NLP")
library("SnowballC")
library("wordcloud")
library("RColorBrewer")
#The text is loaded using Corpus() function from text mining (tm) package. #Import the text file created in
text <- readLines(file.choose())
#You can read a text document from the web
filePath <- "http://www.sthda.com/sthda/RDoc/example-files/martin-luther-king-i-have-a-dream-speech.txt"
text <- readLines(filePath)
```

Solution Exercise 2

```
# 1. Load the data as a corpus
docs <- Corpus(VectorSource(text))
#2. Inspect content of the document
inspect(docs)
```

```
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 46

 [1]
 [2] And so even though we face the difficulties of today and tomorrow, I still have a dream. It is a dream deeply rooted in the American dream.
 [3]
 [4] I have a dream that one day this nation will rise up and live out the true meaning of its creed:
 [5]
 [6] We hold these truths to be self-evident, that all men are created equal.
 [7]
 [8] I have a dream that one day on the red hills of Georgia, the sons of former slaves and the sons of former slave owners will be able to sit down
together at the table of brotherhood.
 [9]
[10] I have a dream that one day even the state of Mississippi, a state sweltering with the heat of injustice, sweltering with the heat of oppressior
, will be transformed into an oasis of freedom and justice.
[11]
[12] I have a dream that my four little children will one day live in a nation where they will not be judged by the color of their skin but by the co
ntent of their character.
```

Solution Exercise 2

#Transformation is performed using **tm_map()** function to replace, for example, special characters from the text. Replacing "/", "@" and "|" with space:

```
toSpace <- content_transformer(function (x , pattern )
gsub(pattern, " ", x))

docs <- tm_map(docs, toSpace, "/")

docs <- tm_map(docs, toSpace, "@")

docs <- tm_map(docs, toSpace, "\\|")
```

Exercise 2: Cleaning the Text

```
# Convert the text to lower case docs
<- tm_map(docs, content_transformer(tolower))
# Remove numbers docs
<- tm_map(docs, removeNumbers)
# Remove english common stopwords docs
<- tm_map(docs, removeWords, stopwords("english"))
# Remove your own stop word
# specify your stopwords as a character vector docs
<- tm_map(docs, removeWords, c("blabla1", "blabla2"))
# Remove punctuations docs
<- tm_map(docs, removePunctuation)
# Eliminate extra white spaces docs
<- tm_map(docs, stripWhitespace)
# Text stemming # docs
<- tm_map(docs, stemDocument)
```

DUKE FUQUA SCHOOL OF BUSINESS

Exercise 2: Build a Term-Document Matrix

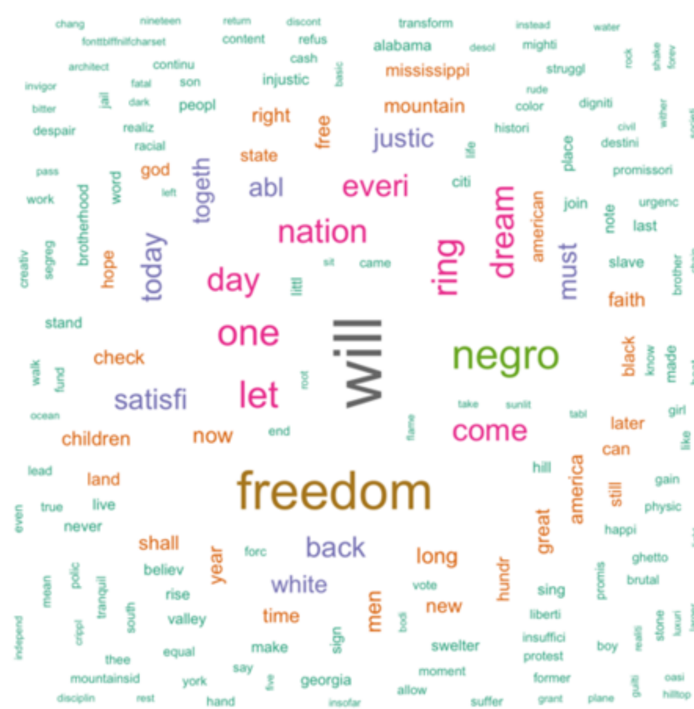
```
> dtm <- TermDocumentMatrix(docs)
> m <- as.matrix(dtm)
> v <- sort(rowSums(m),decreasing=TRUE)
> d <- data.frame(word = names(v),freq=v)
> head(d, 10)
```

	word	freq
will	will	26
freedom	freedom	20
negro	negro	15
one	one	13
let	let	13
ring	ring	12
nation	nation	11
day	day	11
dream	dream	11
come	come	10

DUKE FUQUA SCHOOL OF BUSINESS

Exercise 2: Generate the Word Cloud

```
set.seed(1234) wordcloud(words =
d$word, freq = d$freq, min.freq = 1,
max.words=200, random.order=FALSE,
rot.per=0.35, colors=brewer.pal(8,
"Dark2"))
```



DUKE FUQUA SCHOOL OF BUSINESS

nGrams in this Exercise

```
bigrams <- docs %>%
  unnest_tokens(bigram, text, token = "ngrams", n=2)
bigrams %>%
  count(bigram, sort=TRUE)
```

```
# A tibble: 435 x 2
   bigram           n
   <chr>        <int>
 1 will be      51744
 2 freedom ring 40656
 3 a dream      36960
 4 let freedom  36960
 5 have a       33264
 6 ring from    33264
 7 able to      29568
 8 be able      29568
 9 i have       29568
10 one day      29568
# … with 425 more rows
```

Bigrams

```
#A lot of uninteresting words
bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)
#new bigram counts
bigram_counts<-bigrams_filtered %>%
  count(word1, word2, sort=TRUE)
bigram_counts
```

```
# A tibble: 38 x 3
   word1     word2       n
   <chr>     <chr>   <int>
 1 freedom   ring    40656
 2 join      hands    7392
 3 american  dream    3696
 4 beautiful symphony 3696
 5 black     boys     3696
 6 black     girls    3696
 7 capped    rockies  3696
 8 children  black    3696
 9 country   tis      3696
10 created   equal    3696
# … with 28 more rows
```

Bigrams

```
bigrams_united <- bigrams_filtered %>%
  unite(bigram, word1, word2, sep=" ")
```

```
                    bigram
1              dream deeply
2             deeply rooted
3            american dream
4              true meaning
5             created equal
6                 red hills
7              slave owners
8        injustice sweltering
9                 day live
10           vicious racists
11            lips dripping
12               black boys
13              black girls
14               join hands
15               white boys
16              white girls
17 jangling discords
```

```
483            freedom ring
484            freedom ring
485          children black
486    gentiles protestants
487               join hands
488          negro spiritual
489           spiritual free
490             god almighty
491             dream deeply
492            deeply rooted
493           american dream
494             true meaning
495            created equal
496                red hills
497             slave owners
498    injustice sweltering
499               day live
500          vicious racists
 [ reached getOption("max.print") -- omitted 180604 rows ]
```

Topic Modeling

- What is Topic Modeling?
- Latent Dirichlet Allocation
- Mixture Models

- Example

Topic Models

- Statistical Modeling for discovering the abstract "topics" that occur in a collection of documents
- It is the process to automatically identify topics present in a text object and to derive hidden patterns exhibited by the corpus (collection of documents)

- It is different from rule-based text mining approaches that use dictionary based keyword search techniques
- We use unsupervised approach to find a group of words that we call "topics" in large clusters of texts

DUKE FUQUA SCHOOL OF BUSINESS

What are Topics?

- Topics are "repeating pattern of co-occurring terms in a corpus".
- A good topic model should result in specifically grouped words that should be homogeneous

Health
Doctor
Patient
Hospital

Farm
Crops
Wheat

Approaches for Topic Modeling

- Approaches for obtaining topics from text:
  - Term Frequency: Measure of commonness of a term. Measures how many times a term occurs in the corpus
  - Inverse Document Frequency: Measure of rareness of a term (uniqueness)

$$idf = \log\left(\frac{n}{df}\right)$$

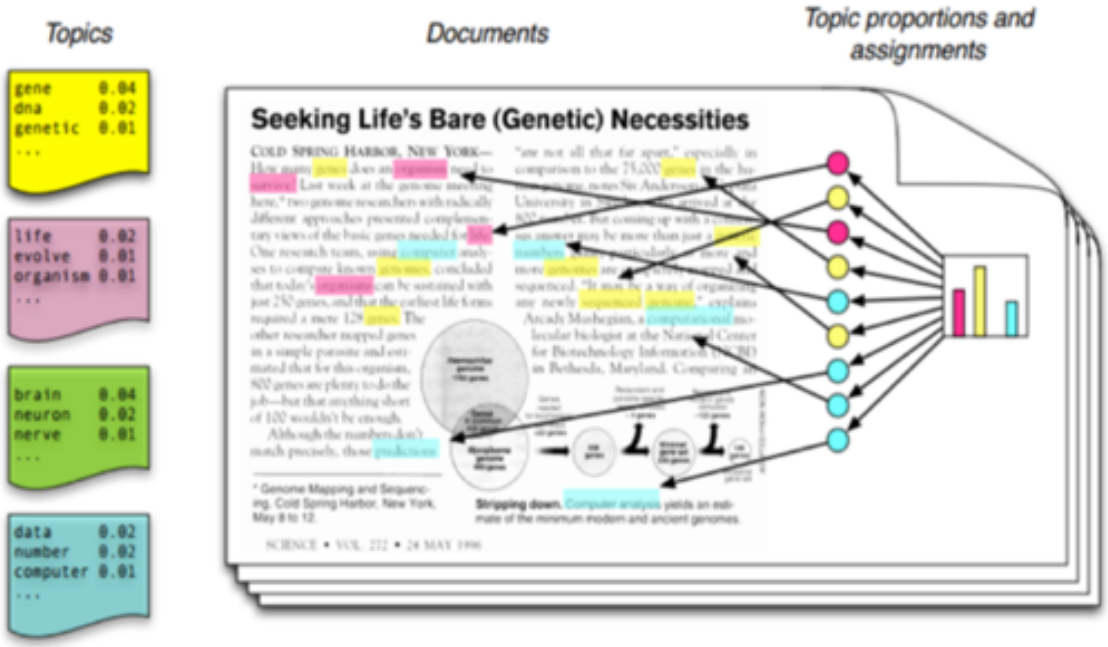
Term of Interest	No. of Documents Containing the term	Total Docs/Docs Containing Terms	IDF Values for these Terms
a	100,000,000	1	-
health	1,000,000	100	2
diagnosis	100,000	1000	3
Cancer	1000	100,000	5

Approaches for Topic Modeling

- Approaches for obtaining topics from text:
  - Non-Negative Matrix Factorization Techniques – automatically extracts sparse and meaningful features from a set of nonnegative data vectors
  - Latent Dirichlet Allocation (LDA):
    - Assumes documents are produced from a mixture of topics
    - These topics generate words based on their probability distribution
    - Given a dataset of documents, LDA tries to figure out what topics will create those documents in the first place
  - Correlated Topic Model (CTM)

LDA: Model Definition

- Each *topic* is a distribution of words
- Each *document* is a mixture of corpus-wide topics
- Each *word* is drawn from one of these topics
- We only care about the words within the documents



DUKE FUQUA SCHOOL OF BUSINESS

LDA

The LDA algorithm asks that you select a number of classes (topics) and input a corpus of documents. The result is a list of topics, each topic is a probability distribution over words. The LDA model will also be able to classify a document, and assign probabilities for each topic.

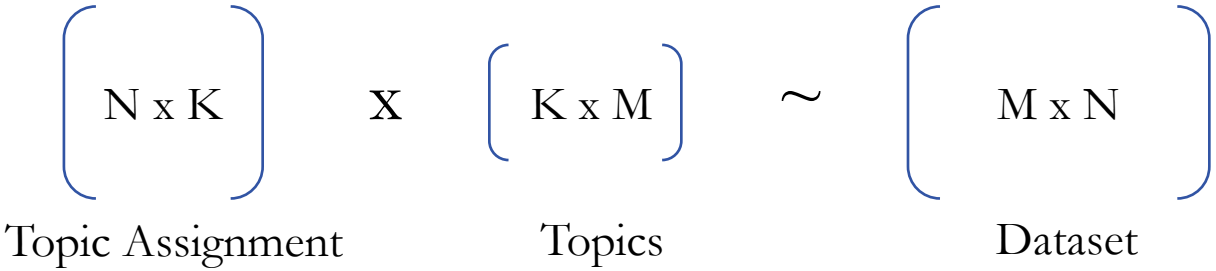
It's important to note that the algorithm makes no high level determination of what the topic *should* be or that the number of topics is adequate for your corpus. That's where we apply human intuition and some mathematical optimization to determine what the topics are and if they are adequately represented by the model.

LDA steps

- A document can, and generally does, belong to multiple topics
- Each topic is represented as a probabilistic distribution over terms in a FIXED vocabulary. What does that mean? For any topic X, we'll have words (w_1..w_n) with associated probabilities that sum to 1. This also means that we'll have to agree to a fixed vocabulary in advance.
- For topic modeling, we are also going to analyze our text data as a bag of words. Order doesn't matter here.

Latent Dirichlet Allocation

- LDA is a matrix factorization technique



M documents in N words, K topics

- Generative model
  - How are data came to be in a sequence of probabilistic steps
  - Use posterior inference to uncover the latent variables (topics) that best explain the data
  - Latent Dirichlet Allocation

Latent Dirichlet Allocation

- A corpus (collection of documents) can be represented as a document-term matrix:

	W1	W2	W3	Wn
D1	0	2	1	3
D2	1	4	0	0
D3	0	2	3	1
Dm	1	1	3	0

n is the size of vocabulary (words/terms); m=number of documents

- LDA converts the DTM into two lower dimensions (M1 and M2)

LDA Matrices

	K1	K2	K3	K
D1	1	0	0	1
D2	1	1	0	0
D3	1	0	0	1
Dm	1	0	1	0

M1= Document-Topics Matrix

	W1	W2	W3	Wn
K1	0	1	1	1
K2	1	1	1	0
K3	1	0	0	1
K	1	1	0	0

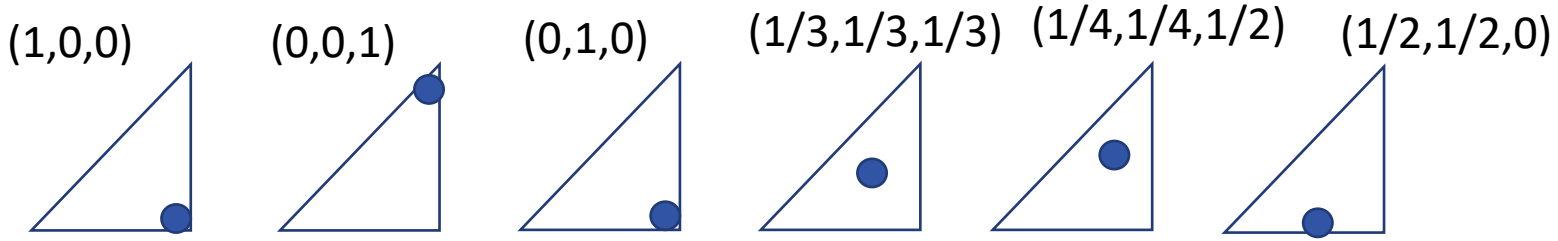
M2= Topic-Terms Matrix

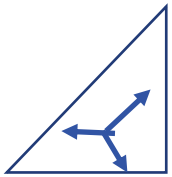
K is the number of topics

- The two matrices provide document-topic and topic-word distributions
- Aim of LDA is the improve these distributions using sampling techniques

LDA: Matrix Factorization Approach

- Dirichlet process: Distribution over a multinomial distribution
  - If distributions live in triangular spaces
  - Distribution over outcomes

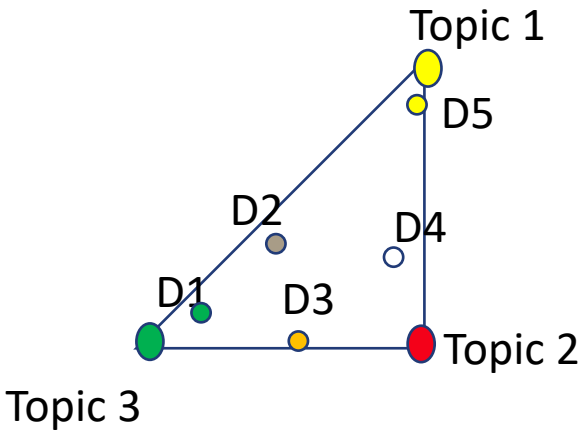




$$p(\emptyset|\alpha, w) \propto p(w|\emptyset) * p(\emptyset|\alpha)$$

LDA: Generative Model

- How our data came to be?
  - We have some topics
  - $\theta_d \sim Dirichlet\ (\alpha)$
  - $W \sim Mult(\theta z)$



We draw our topics from a Dirichlet distribution (theta)of words with some prior, and then for every document we draw a distribution over topics for that document

LDA: Topic Models

- Topics to word types: multinomial distribution
- Documents to topics: multinomial distribution
- Model: story of how your data came to be
- Latent variables: missing data
- Statistical inference: how to fill in the missing data

Statistical Inference

- We are interested in something called the posterior distribution $p(Z|X, \theta)$

We use MCMC (markov chain monte carlo) Bayesian modeling

Gibbs Sampling Equation

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,w_{d,n}} + \eta_{w_{d,n}}}{\sum_i v_{k,i} + \eta_i}$$

Number of times document d uses topic k

Number of times topic k uses word type $w_{d,n}$

Dirichlet parameter for document to topic distribution

Dirichlet parameter for topic to word distribution

How much this document likes topic k (whole left term)

How much this topic likes word $w_{d,n}$ (right term)

Statistical Inference

We have a document, we have five words, we start by assigning topics randomly (four topics) and do this for all documents.

Z ↓ W

3	2	1	3	1
Health	Diagnostic	Cancer	Heart	Patient

	1	2	3	4
Health	1	0	35	…
Diagnostic	50	0	1	…
Cancer	42	1	0	…
Heart	0	1	20	…
Patient	10	8	1	…

Total count from all docs →

Statistical Inference

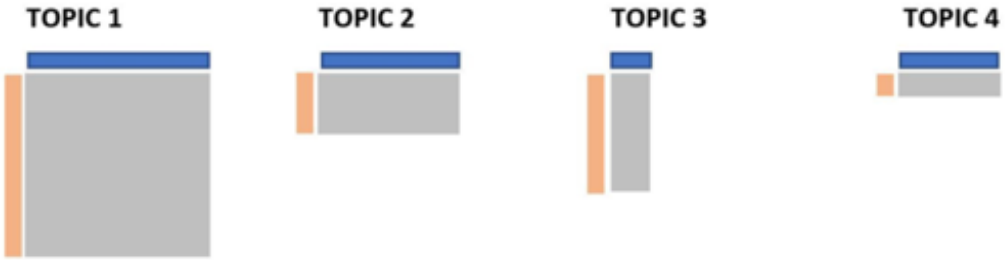
Lets sample the word patient. We take the word patient occurring in Topic 2, we unassign it.

Z ↓ W					
	3	2	1	3	1
	Health	Patient	Cancer	Heart	Diagnostic

Lets represent the matrix n(m,k) in the following way to show how much a document uses each topic

Then lets represent how many times each topic is assigned to this word

Now we multiply these two matrices to get conditional probabilities



Statistical Inference

. We will reduce the count of the number of times the topic 2 uses word patient from 8 to 7.

Z	3	2	1	3	1
↓					
W	Health	Patient	Cancer	Heart	Diagnostic

	1	2	3	4
Health	1	0	35	…
Diagnostic	50	0	1	…
Cancer	42	1	0	…
Heart	0	1	20	…
Patient	10	7 (from 8)	1	…

Total count from all docs →

Statistical Inference

We randomly pick any of the other topics and will assign that topic to patient and this step is repeated for all other words as well. Ideally, topic with highest conditional probability should be selected (using Gibbs Sampler).

Z ↓ W

3	?	1	3	1
Health	Patient	Cancer	Heart	Diagnostic

Total count from all docs →

	1	2	3
Health	1	0	35
Diagnostic	50	0	1
Cancer	42	1	0
Heart	0	1	20
Patient	10	7 (from 8)	1

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \frac{v_{k,wd_n} + \eta_{w_{dn}}}{\sum_i v_{k,i} + \eta_i}$$

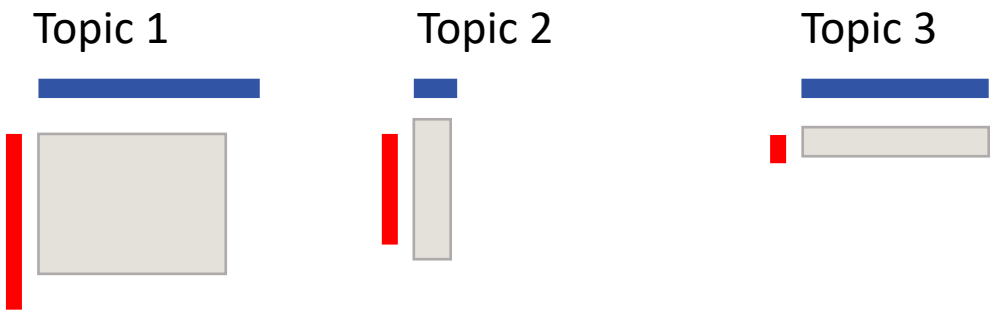
Statistical Inference

We sample with a probability equal to the grey area and we up the count of patient in topic 1. So, the blue bar gets wider and red bar gets longer

z ↓ w

3	1	1	3	1
Health	Patient	Cancer	Heart	Diagnostic

$$\frac{n_{d,k} + \alpha_k}{\sum_i^K n_{d,i} + \alpha_i} \quad \frac{v_{k,wd_n} + \eta_{w_{d,n}}}{\sum_i v_{k,i} + \eta_i}$$



	1	2	3
Patient	11	7	1

LDA Process

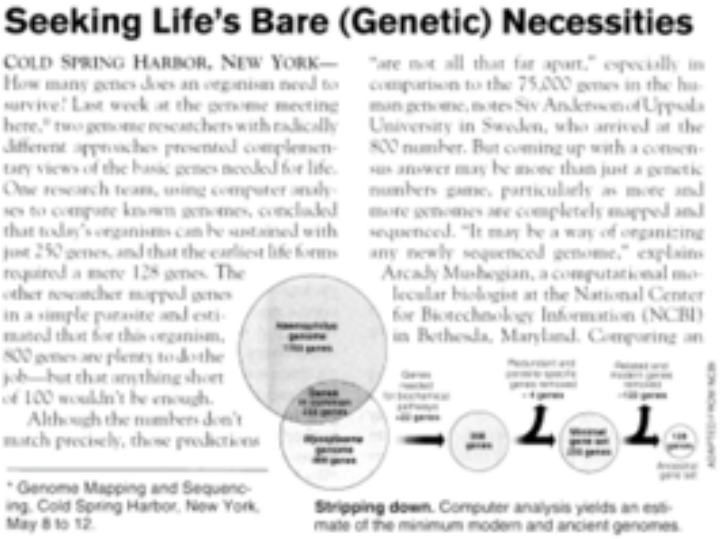
- A new topic "k" is assigned to word "w" with a probability (P) which is a product of two probabilities p1 and p2
- For every topic, two probabilities p1 and p2 are calculated.
  - *P1 – p(topic / document d) = the proportion of words in document d that are currently assigned to topic k.*
  - *P2 – p(word w / topic) = the proportion of assignments to topic k over all documents that come from this word w.*
- The current topic – word assignment is updated with a new topic with the probability (Beta values), product of p1 and p2
- After a number of iterations, a steady state is achieved where the document topic and topic term distributions are fairly good. This is the convergence point of LDA.

LDA: Example Output and Simulation

- Data: Collection of Science from 1990-2000
- 17K documents
- 11M words
- 20K unique terms (redundant words removed)

Model: 100-topic LDA model

Source: Science, 1996, Vol 272, Genome Mapping and Sequencing



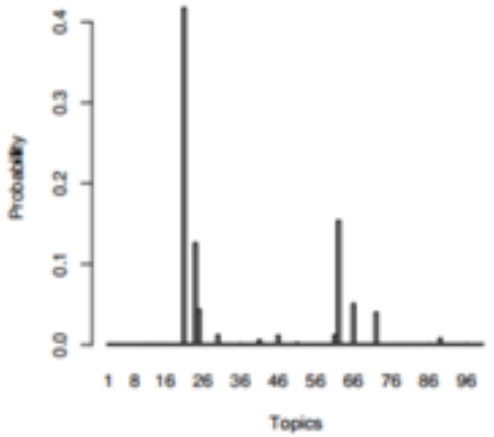
Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.



DUKE FUQUA SCHOOL OF BUSINESS

LDA: Example Output and Simulation

- Most probable words in four of the topics:

Probability ↑

Topic 1	Topic 2	Topic 3	Topic 4
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Values of K (no. of topics)

- Values of K has to be identified apriori
- The stm package in r provides a function search where you should be able to provide a range of k and look at output for each k
- Diagnostics such as residuals can only be observed if you are using a supervised model with topics as explanatory variables
- Any measures in the literature currently are imperfect to human validation of the topic models by carefully inspecting not only the top words associated with each document

Improving Results of Topic Modeling

- The result of topic models are completely dependent on the features (terms) present in the corpus.
- Reducing the dimensionality of the matrix can improve the results of topic modeling
  - **Frequency Filter**: Arrange all terms according to frequency (get rid of all weak features)
  - **Batch-wise LDA**: A corpus can be divided into batches of fixed-sizes. Running LDA multiple times on these batches will provide different results, however, the best topic terms will be the intersection of all batches

Example: NBC News Topic Modeling

```
#NBC Health Tweet Data Topic Modeling
library(data.table)
nbc <- read.csv(file="~/Desktop/NBCHealthData.csv", head=T, stringsAsFactors = FALSE)
ls()
str(nbc)
```

```
'data.frame':   4165 obs. of  1 variable:
 $ Health_Tweets: chr  "Ebola Exposure?: CDC Worker 'Remains Well' " "Pregnant Woman Taken Off Life Support in Ireland " "Money from #IceBucketChall
nge still flowing to ALS Association where leaders remain awestruck by gush of goodwill: " "Caramel Apples Recalled After Listeria Deaths " ...
```

```
#Loading libraries "dplyr" and "tidytext" to tokenize
#You can also use library "stringr" when working with text strings
#You can use library "tidyr" for data reshaping
library(tidyr)
library(dplyr)
library(tidytext)
library(stringr)
```

Tokenization

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called *tokens* , perhaps at the same time throwing away certain characters, such as punctuation.

"Tokens" are usually individual words and "**tokenization**" is taking a text or set of text and breaking it up into its individual words

Un-nesting Tokens

#use the unnest_tokens function to get the words from the "Health_Tweets" column of pcd

#unnest_tokens also converts all words into lower case

```
tidy_text <- nbc %>% unnest_tokens(word, Health_Tweets)
```

```
str(tidy_text)
```

```
'data.frame':   38455 obs. of  1 variable:
 $ word: chr  "ebola" "exposure" "cdc" "worker" ...
```

```
head(tidy_text)
```

```
      word
1    ebola
1.1 exposure
1.2     cdc
1.3  worker
1.4 remains
1.5    well
```

Removing Stopwords

```
#Removing stopwords
tidy_text <- tidy_text %>% anti_join(stop_words)
#Sorting by frequency of tokens
tidy_text %>% count(word, sort = TRUE)
```

```
# A tibble: 6,395 x 2
   word              n
   <chr>         <int>
 1 http            555
 2 nbcnews.to      424
 3 study           331
 4 ebola           291
 5 fda             238
 6 health          235
 7 cancer          187
 8 kids            142
 9 flu             139
10 on.today.com    125
# … with 6,385 more rows
```

Create own Stop-words Dictionary

```
#create own stopword dictionary
my_stopwords <- tibble(word=c(as.character(1:10),
                              "http", "nbcnews.to", "on.today.com",
"u.s.", "study"))

tidy_text <- tidy_text %>% anti_join(my_stopwords)
```

Frequency of Words

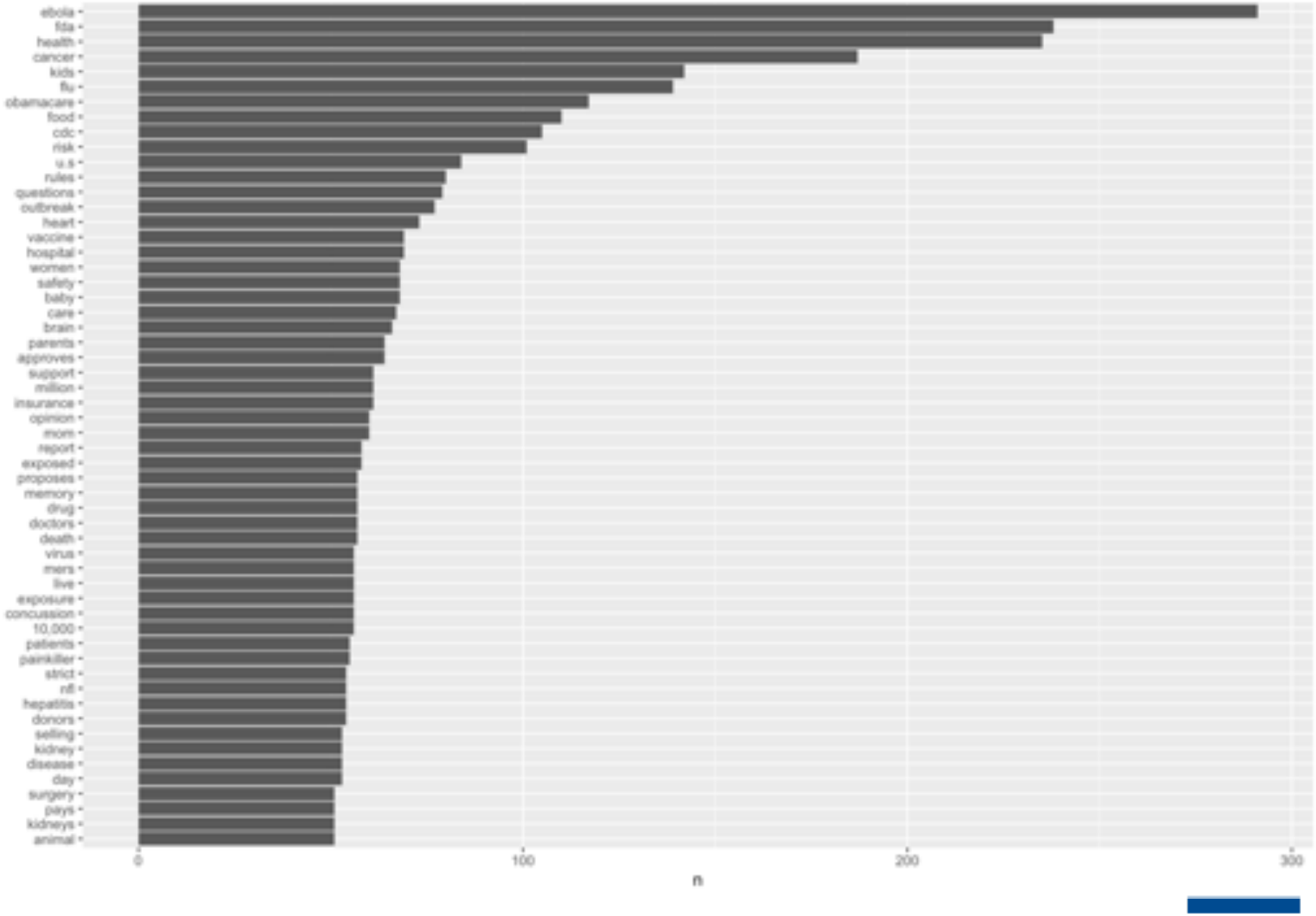
#Sorting by frequency of tokens again

tidy_text %>% count(word, sort = TRUE)

```
# A tibble: 6,381 x 2
   word          n
   <chr>     <int>
 1 ebola       291
 2 fda         238
 3 health      235
 4 cancer      187
 5 kids        142
 6 flu         139
 7 obamacare   117
 8 food        110
 9 cdc         105
10 risk        101
# … with 6,371 more rows
```

Visualizing Tokens

```
#Visualizing sorted tokens with min freq of 50
library(ggplot2)
tidy_text %>%
  count(word, sort = TRUE) %>%
  filter( n > 50) %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n)) +
  geom_bar(stat = "identity") +
  xlab(NULL) +
  coord_flip()
```


ebola
fda
health
cancer
kids
flu
obamacare
food
cdc
risk
u.s
rules
questions
outbreak
heart
vaccine
hospital
women
safety
baby
care
brain
parents
approves
support
million
insurance
opinion
mom
report
exposed
proposes
memory
drug
doctors
death
virus
mers
live
exposure
concussion
10,000
patients
painkiller
strict
nfl
hepatitis
donors
selling
kidney
disease
day
surgery
pays
kidneys
animal
0
100
200
300
n

Sentiment Analysis

Many scholars view sentiment analysis as "the computational treatment of opinion, sentiment, evaluation, attitudes, appraisal, views, emotions, and subjectivity in text"

Text = reviews, blogs, discussions, news, comments, feedback

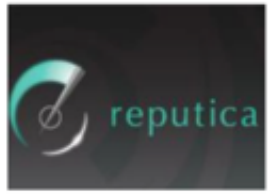
Also called opinion mining

Typical Sentiment Analysis Usage

- Extract how people feel about a particular topic/product/brand
- Sentiment analysis can be tricky
  - Honda Accords and Toyota Camrys are nice sedans
  - Honda Accords and Toyota Camrys are nice sedans but hardly the best car on the road

Sentiment Analysis is one of the most Popular areas of Analytics

Dozens of companies and over thousand papers



Sentiment Analysis

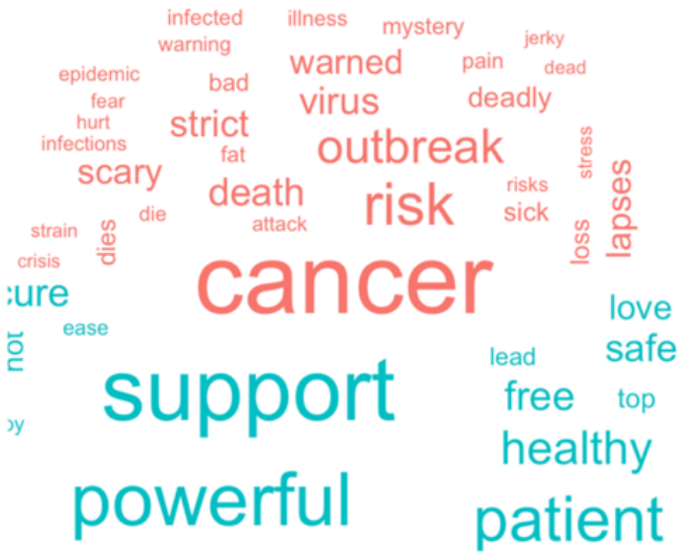
#Let's do sentiment analysis to tag +ve and -ve words using inner join function

```
x <- tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
x
```

```
# A tibble: 831 x 3
   word      sentiment     n
   <chr>     <chr>     <int>
 1 cancer    negative    187
 2 risk      negative    101
 3 outbreak  negative     77
 4 support   positive     61
 5 death     negative     57
 6 virus     negative     56
 7 strict    negative     54
 8 powerful  positive     50
 9 scary     negative     50
10 genuine   positive     48
# … with 821 more rows
```

Color coded Word Cloud using Sentiments

```
library(reshape2)
tidy_text %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("#F8766D", "#00BFC4"),
                   max.words = 70)
```



DUKE FUQUA SCHOOL OF BUSINESS

Another Type of Sentiment Analysis

```
#You can also use the nrc method to get an expanded sentiment group for each word

y <- tidy_text %>%

  inner_join(get_sentiments("nrc")) %>%

  count(word, sentiment, sort = TRUE) %>%

  ungroup()

y
```

```
# A tibble: 2,819 x 3
   word    sentiment     n
   <chr>   <chr>     <int>
 1 cancer  anger       187
 2 cancer  disgust     187
 3 cancer  fear        187
 4 cancer  negative    187
 5 cancer  sadness     187
 6 flu     fear        139
 7 flu     negative    139
 8 food    joy         110
 9 food    positive    110
10 food    trust       110
# … with 2,809 more rows
```

Color Coded Word Cloud

```
##color-coded word cloud based
on sentiment

tidy_text %>%


inner_join(get_sentiments("bing"))
%>%

  count(word, sentiment, sort =
TRUE) %>%

  acast(word ~ sentiment, value.var
= "n", fill = 0) %>%

  comparison.cloud(colors =
c("#F8766D", "#00BFC4"),

                   max.words = 100)
```

DUKE FUQUA SCHOOL OF BUSINESS

Topic Modeling

```
library(RTextTools)
library(tm)
library(wordcloud)
library(topicmodels)
library(slam)
library(NLP)
library(RColorBrewer)
```

Corpus: Cleaning Data

```
nbc_corpus <- Corpus(VectorSource(nbc$Health_Tweets))

nbc_corpus <- tm_map(nbc_corpus, removeNumbers)
nbc_corpus <- tm_map(nbc_corpus, removePunctuation)
nbc_corpus <- tm_map(nbc_corpus, stripWhitespace)
nbc_corpus <- tm_map(nbc_corpus, tolower)
nbc_corpus <- tm_map(nbc_corpus, removeWords,
stopwords("english"))
nbc_corpus <- tm_map(nbc_corpus, stemDocument, language =
"english")

dtm_nbc <-DocumentTermMatrix(nbc_corpus)
```

Document Term Matrix

dtm_nbc

<<DocumentTermMatrix (documents: 4165, terms: 5063)>>

Non-/sparse entries: 27382/21060013

Sparsity : 100%

Maximal term length : 38

Weighting : term frequency (tf)

Document-Term Matrix

#see the first 10 rows and first 20 columns from the document term matrix)

dtm_nbc <-as.matrix(dtm_nbc)

dtm_nbc[1:10, 1:20]

```
    Terms
Docs cdc ebola exposur remain well worker ireland life pregnant support taken woman associ
   1   1     1       1      1    1      1       0    0        0       0     0     0      0
   2   0     0       0      0    0      0       1    1        1       1     1     1      0
   3   0     0       0      1    0      0       0    0        0       0     0     0      1
   4   0     0       0      0    0      0       0    0        0       0     0     0      0
   5   0     0       0      0    0      0       0    0        0       0     0     0      0
   6   1     1       1      0    0      0       0    0        0       0     0     0      0
   7   1     1       1      0    0      0       0    0        0       0     0     0      0
   8   0     0       0      0    0      0       0    0        0       0     0     0      0
   9   0     0       0      0    0      0       0    0        0       0     0     0      0
   10  0     0       0      0    0      0       0    0        0       0     0     0      0
    Terms
Docs awestruck flow goodwil gush icebucketchalleng leader money
   1         0    0       0    0                 0      0     0
   2         0    0       0    0                 0      0     0
   3         1    1       1    1                 1      1     1
   4         0    0       0    0                 0      0     0
   5         0    0       0    0                 0      0     0
   6         0    0       0    0                 0      0     0
   7         0    0       0    0                 0      0     0
   8         0    0       0    0                 0      0     0
   9         0    0       0    0                 0      0     0
   10        0    0       0    0                 0      0     0
```

LDA

```
rowTotals <- apply(dtm_nbc , 1, sum) #Find the sum of words in each Document

dtm.new   <- dtm_nbc[rowTotals> 0, ] #remove all docs without words

#Run the LDA model

lda <- LDA(dtm.new, k = 4) # k is the number of topics to be found.

lda
```

A LDA_VEM topic model with 4 topics.

LDA

```
library(tidytext)
lda_td <- tidy(lda)
lda_td
```

```
# A tibble: 20,252 x 3
   topic term        beta
   <int> <chr>      <dbl>
 1     1 cdc     0.00258
 2     2 cdc     0.00479
 3     3 cdc     0.00416
 4     4 cdc     0.00372
 5     1 ebola   0.0155
 6     2 ebola   0.0146
 7     3 ebola   0.000696
 8     4 ebola   0.00907
 9     1 exposur 0.000527
10     2 exposur 0.00238
# … with 20,242 more rows
```

```
top_terms <- lda_td %>%
  group_by(topic) %>%
  top_n(10, beta) %>%
  ungroup() %>%
  arrange(topic, -beta)
top_terms
```

```
# A tibble: 40 x 3
   topic term      beta
   <int> <chr>    <dbl>
 1     1 may    0.0217
 2     1 find   0.0211
 3     1 new    0.0176
 4     1 ebola  0.0155
 5     1 studi  0.0130
 6     1 flu    0.00880
 7     1 risk   0.00874
 8     1 rule   0.00833
 9     1 cancer 0.00824
10     1 can    0.00788
# … with 30 more rows
```

Plots

```
top_terms %>%
  ggplot(aes(term, beta, fill=factor(topic))) +
  geom_col(show.legend=FALSE) +
  facet_wrap(~topic,scales="free")+
  coord_flip()
```

Sentiment Analysis

- R packages for Sentiment Analysis
- Extracting social data
- Cleaning Corpus
- Word Cloud
- Sentiment Analysis

Text Mining and Sentiment Analysis- R packages

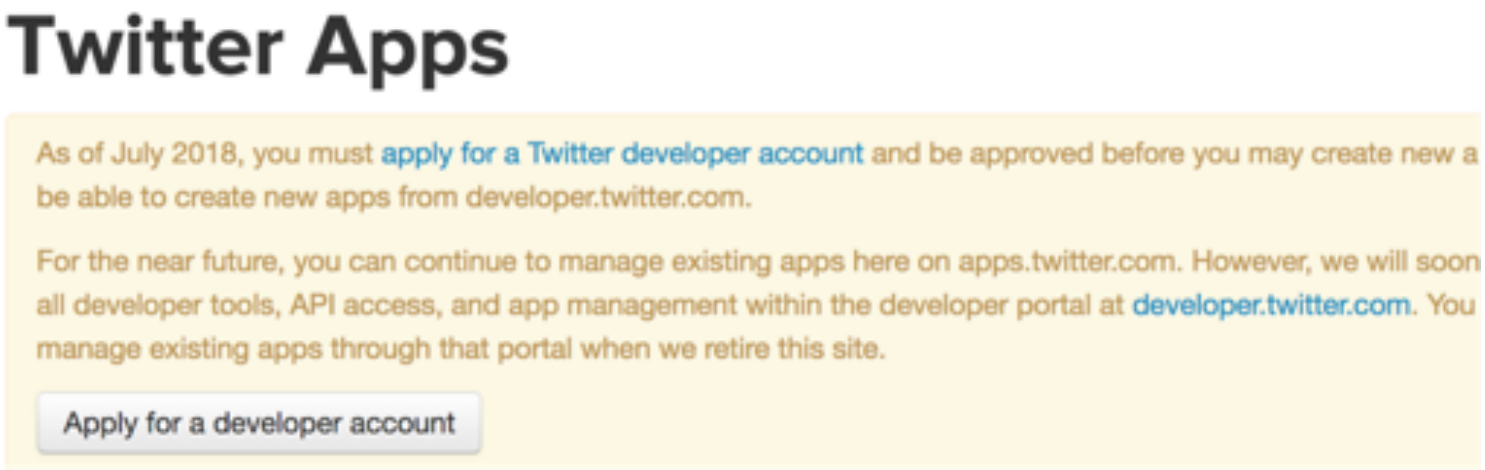
- twitteR: Extract tweets from the Twitter website
- *tm* & *NLP* : create corpus, clean text (remove punctuations, numbers, hyperlinks and stop words, stemming)
  - Build a "term document matrix"
- wordcloud: Analyze topics with a wordcloud
- syuzhet: Conduct sentiment analysis
- lubridate, dplyr: Perform data manipulation using *lubridate* (working with dates), *dplyr* (data management) package
- ggplot2, igraph, scale: Analyze term and tweet relationships with the *ggplot2*, *igraph,* and do title scale manipulation using *scale* package

DUKE FUQUA SCHOOL OF BUSINESS

Extracting Twitter Data

1. Create a Twitter application to extract data from Twitter

   You must first have a twitter account, as well as a twitter developer account

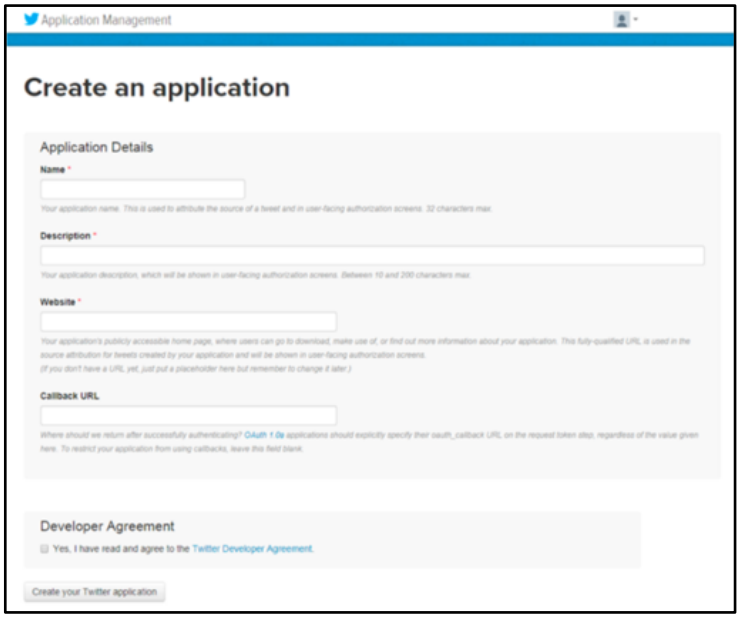


DUKE FUQUA SCHOOL OF BUSINESS

Extracting Twitter Data

2. Once you have a developer account, you can create a twitter app (click on *create an app*)

Fill in the application form (shown below) with relevant details. Note that the name should be unique and should not have been used by anyone else before. After you have read the *Twitter Developer Agreement*, tick the check box and '*Create your Twitter application*'. Oh! And, you will need to have your phone linked to twitter to create an app



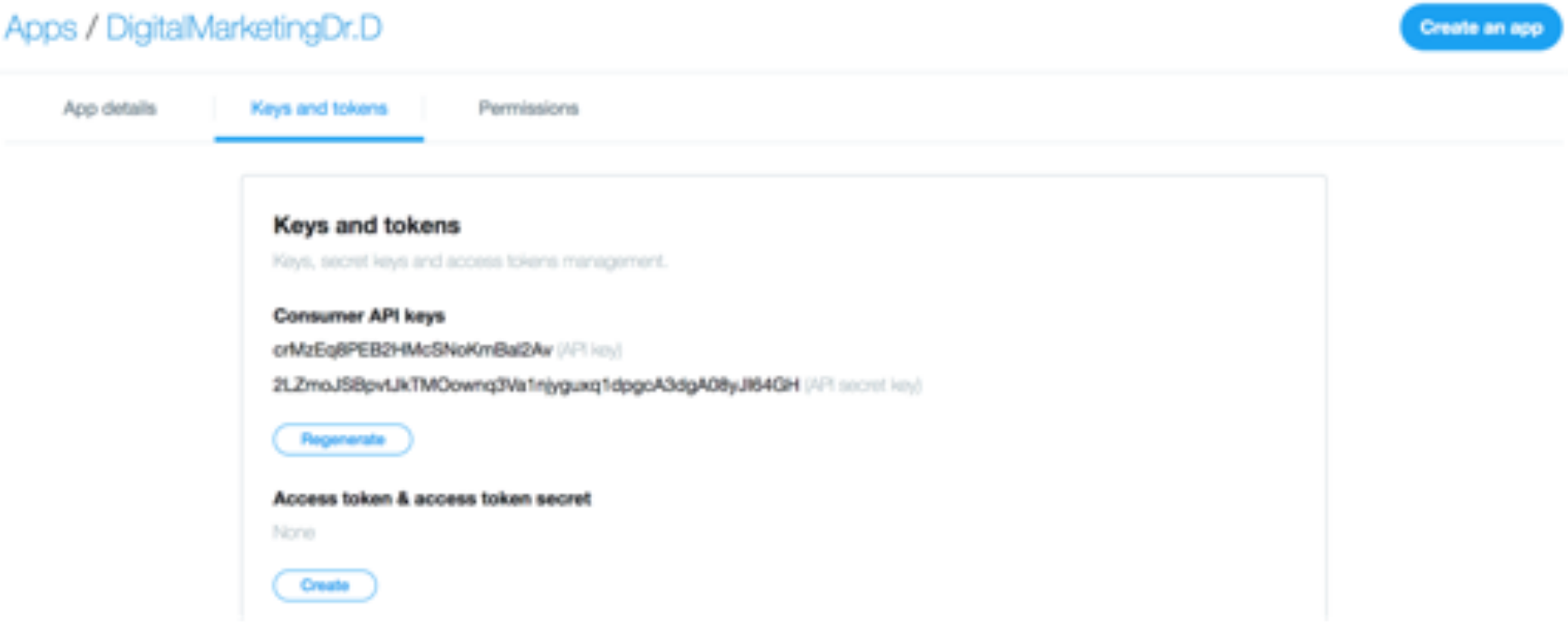
DUKE FUQUA SCHOOL OF BUSINESS

New App: DigitalMarketingDr.D

DUKE FUQUA SCHOOL OF BUSINESS

Extracting Twitter Data

3. You will lend on application details page; move to '*Keys and Access Tokens*' tab, scroll down and click '*Create my access token*'. Note the values of *API Key* and *API Secret* for future use. Thou shan't share these with anyone, one can access your account if they get the keys.



Extracting Twitter Data

4. In order to extract tweets, you will need to establish a secure connection between R and Twitter as follows:

- ROAuth: R interface for OAuth, the open standard for token-based authorization on the internet and then setup twitter authorization code with api and access information shown below

```
#Load required libraries
install.packages("twitteR")
install.packages("ROAuth")
library("twitteR")
library(”ROAuth")

setup_twitter_oauth(api_key,
                    api_secret,
                    access_token,
                    access_token_secret)
```

```
#Insert Values
api_key <-’xx1'
api_secret <- ‘xx2'
access_token <- ‘xx3'
access_token_secret <-’xx4'
```

DUKE FUQUA SCHOOL OF BUSINESS

Extract Tweets

- Now that we are all done with setting up gateways to reach Twitter, use Function searchTwitter - it lets you search through Twitter and return a list of tweets consisting the searched text.

```
tweets <-searchTwitter("#tesla", n=1000, lang='en')

length(tweets)

#[1] 1000

tweets #Look at the extracted tweets
```

```
[1] RT @vincent13031925: The CupHead from the latest Tesla's V10 update is definitely my new favorite game!! \n\
nThanks @elonmusk &amp;amp; @Tesla!! Spen…
[2] RT @exploreplanets: We've teamed up with @ZelectricBug and @omaze again to give you the chance to win a one-
of-a-kind 1965 VW Convertible B…
[3] RT @EliBurton_: I captured a hit a hit and run on my #Tesla dash cam. Watch how this ends ♂  https://t.co
/YccButCZMK
[4] RT @hasanihunter: Hey Nebraska legislators.. prohibiting Tesla from selling its cars in this state is like t
elling customers the only way y…
[5] RT @TeslaSono: Worked out today that driving a Tesla Model 3 from Sussex to Inverness (608 miles) will cost
a #Tesla owner £30, if using th…
```

…

DUKE FUQUA SCHOOL OF BUSINESS

Pre-processing Data

```
#convert into list to data frame
tesla<-twListToDF(tweets)
#Create CSV file and save it
write.csv(tesla, file = '~/Desktop/tesla.csv', row.names = F)
#Reading Data File
tesla<-read.csv(file.choose(), header=T)
```

DUKE FUQUA SCHOOL OF BUSINESS

Structure of Twitter File

str(tesla)

```
'data.frame':   1000 obs. of  16 variables:
 $ text         : Factor w/ 466 levels ".@elonmusk - Love love LOVE the new rapid TACC acceleration in V10! I use TACC regularly on city
highways and s"| __truncated__,..: 267 269 261 242 202 316 266 292 127 373 ...
 $ favorited    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ favoriteCount: int  0 0 0 0 2 0 0 0 0 0 ...
 $ replyToSN    : Factor w/ 35 levels "_youhadonejob1",..: NA NA NA NA NA NA NA NA NA NA ...
 $ created      : Factor w/ 978 levels "2019-09-30 12:09:21",..: 978 977 976 975 974 973 972 971 970 969 ...
 $ truncated    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ replyToSID   : num  NA NA NA NA NA NA NA NA NA NA ...
 $ id           : num  1.18e+18 1.18e+18 1.18e+18 1.18e+18 1.18e+18 ...
 $ replyToUID   : num  NA NA NA NA NA NA NA NA NA NA ...
 $ statusSource : Factor w/ 46 levels "<a href=\"http://app.sendblur.com\" rel=\"nofollow\">Social Media Publisher App </a>",..: 13 12 9
13 12 10 32 32 13 13 ...
 $ screenName   : Factor w/ 760 levels "_baldtires","_ClimateXChange",..: 378 281 41 179 520 347 97 73 553 420 ...
 $ retweetCount : int  432 16 11 3 0 1 46 148 0 10 ...
 $ isRetweet    : logi  TRUE TRUE TRUE TRUE FALSE TRUE ...
 $ retweeted    : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ longitude    : num  NA NA NA NA NA NA NA NA NA NA ...
 $ latitude     : num  NA NA NA NA NA NA NA NA NA NA ...
```

DUKE FUQUA SCHOOL OF BUSINESS

Building Corpus

```
#clean the text of special characters such as symbols and emoticons
tesla$text <- sapply(tesla$text,function(row) iconv(row, "latin1", "ASCII", sub=""))
#Building Corpus
library(tm)
library(NLP)
  corpus <-iconv(tesla$text, to='utf-8-mac') #need only the first col text from file
  corpus <- Corpus(VectorSource(corpus)) #corpus is a collection of texts
  inspect(corpus[1:5]) #inspect the first five tweets
```

```
<<SimpleCorpus>>
Metadata:  corpus specific: 1, document level (indexed): 0
Content:  documents: 5

[1] RT @vincent13031925: The CupHead from the latest Teslas V10 update is definitely my new favorite game!! \n\n
Thanks @elonmusk &amp;amp; @Tesla!! Spen
[2] RT @exploreplanets: Weve teamed up with @ZelectricBug and @omaze again to give you the chance to win a one-o
f-a-kind 1965 VW Convertible B
[3] RT @EliBurton_: I captured a hit a hit and run on my #Tesla dash cam. Watch how this ends  https://t.co/YccB
utCZMK
[4] RT @hasanihunter: Hey Nebraska legislators.. prohibiting Tesla from selling its cars in this state is like t
elling customers the only way y
[5] RT @TeslaSono: Worked out today that driving a Tesla Model 3 from Sussex to Inverness (608 miles) will cost
a #Tesla owner 30, if using th
```

Clean Data (Corpus)

#convert data to lower case for analysis

corpus <-tm_map(corpus, tolower) #convert all alphabet to lower case

inspect(corpus[1:5]) #inspect the first five tweets

```
[1] rt @vincent13031925: the cuphead from the latest teslas v10 update is definitely my new favorite game!! \n\n
thanks @elonmusk &amp;amp; @tesla!! spen
[2] rt @exploreplanets: weve teamed up with @zelectricbug and @omaze again to give you the chance to win a one-o
f-a-kind 1965 vw convertible b
[3] rt @eliburton_: i captured a hit a hit and run on my #tesla dash cam. watch how this ends  https://t.co/yccb
utczmk
[4] rt @hasanihunter: hey nebraska legislators.. prohibiting tesla from selling its cars in this state is like t
elling customers the only way y
[5] rt @teslasono: worked out today that driving a tesla model 3 from sussex to inverness (608 miles) will cost
a #tesla owner 30, if using th
```

#remove punctuations

corpus <-tm_map(corpus, removePunctuation)

inspect(corpus[1:5]) #inspect the first five tweets

```
[1] rt vincent13031925 the cuphead from the latest teslas v10 update is definitely my new favorite game \n\nthan
ks elonmusk amp tesla spen
[2] rt exploreplanets weve teamed up with zelectricbug and omaze again to give you the chance to win a oneofakin
d 1965 vw convertible b
[3] rt eliburton i captured a hit a hit and run on my tesla dash cam watch how this ends  httpstcoyccbutczmk
[4] rt hasanihunter hey nebraska legislators prohibiting tesla from selling its cars in this state is like telli
ng customers the only way y
[5] rt teslasono worked out today that driving a tesla model 3 from sussex to inverness 608 miles will cost a te
sla owner 30 if using th
```

DUKE FUQUA SCHOOL OF BUSINESS

Clean Corpus

```
#remove numbers
corpus <-tm_map(corpus, removeNumbers)
inspect(corpus[1:5]) #inspect the first five tweets
```

```
[1] rt vincent the cuphead from the latest teslas v update is definitely my new favorite game \n\nthanks elonmus
k amp tesla spen
[2] rt exploreplanets weve teamed up with zelectricbug and omaze again to give you the chance to win a oneofakin
d  vw convertible b
[3] rt eliburton i captured a hit a hit and run on my tesla dash cam watch how this ends  httpstcoyccbutczmk
[4] rt hasanihunter hey nebraska legislators prohibiting tesla from selling its cars in this state is like telli
ng customers the only way y
[5] rt teslasono worked out today that driving a tesla model  from sussex to inverness  miles will cost a tesla
owner  if using th
```

```
#select stopwords(english) to see what words are removed
cleanset <-tm_map(corpus, removeWords, stopwords('english'))
inspect(cleanset[1:5])
```

```
[1] rt vincent  cuphead  latest teslas v update  definitely  new favorite game \n\nthanks elonmusk amp tesla sp
en
[2] rt exploreplanets weve teamed  zelectricbug  omaze  give  chance  win  oneofakind  vw convertible b
[3] rt eliburton  captured  hit  hit  run  tesla dash cam watch  ends  httpstcoyccbutczmk
[4] rt hasanihunter hey nebraska legislators prohibiting tesla  selling  cars  state  like telling customers
way y
[5] rt teslasono worked  today  driving  tesla model  sussex  inverness  miles will cost  tesla owner  using t
h
```

Clean Corpus (Stopwords Dictionary)

```
> stopwords('english')
  [1] "i"          "me"         "my"         "myself"     "we"         "our"        "ours"       "ourselves"  "you"
 [10] "your"       "yours"      "yourself"   "yourselves" "he"         "him"        "his"        "himself"    "she"
 [19] "her"        "hers"       "herself"    "it"         "its"        "itself"     "they"       "them"       "their"
 [28] "theirs"     "themselves" "what"       "which"      "who"        "whom"       "this"       "that"       "these"
 [37] "those"      "am"         "is"         "are"        "was"        "were"       "be"         "been"       "being"
 [46] "have"       "has"        "had"        "having"     "do"         "does"       "did"        "doing"      "would"
 [55] "should"     "could"      "ought"      "i'm"        "you're"     "he's"       "she's"      "it's"       "we're"
 [64] "they're"    "i've"       "you've"     "we've"      "they've"    "i'd"        "you'd"      "he'd"       "she'd"
 [73] "we'd"       "they'd"     "i'll"       "you'll"     "he'll"      "she'll"     "we'll"      "they'll"    "isn't"
 [82] "aren't"     "wasn't"     "weren't"    "hasn't"     "haven't"    "hadn't"     "doesn't"    "don't"      "didn't"
 [91] "won't"      "wouldn't"   "shan't"     "shouldn't"  "can't"      "cannot"     "couldn't"   "mustn't"    "let's"
[100] "that's"     "who's"      "what's"     "here's"     "there's"    "when's"     "where's"    "why's"      "how's"
[109] "a"          "an"         "the"        "and"        "but"        "if"         "or"         "because"    "as"
[118] "until"      "while"      "of"         "at"         "by"         "for"        "with"       "about"      "against"
[127] "between"    "into"       "through"    "during"     "before"     "after"      "above"      "below"      "to"
[136] "from"       "up"         "down"       "in"         "out"        "on"         "off"        "over"       "under"
[145] "again"      "further"    "then"       "once"       "here"       "there"      "when"       "where"      "why"
[154] "how"        "all"        "any"        "both"       "each"       "few"        "more"       "most"       "other"
[163] "some"       "such"       "no"         "nor"        "not"        "only"       "own"        "same"       "so"
[172] "than"       "too"        "very"
>
```

DUKE FUQUA SCHOOL OF BUSINESS

Clean Corpus

#remove URLs (https://etc.); make use of function http

removeURL <- function(x) gsub("http[[:alnum:]]*", '', x)

cleanset <-tm_map(cleanset, content_transformer(removeURL))

```
[1] rt vincent  cuphead  latest teslas v update  definitely  new favorite game \n\nthanks elonmusk amp tesla sp
en
[2] rt exploreplanets weve teamed  zelectricbug  omaze  give  chance  win  oneofakind  vw convertible b
[3] rt eliburton  captured  hit  hit  run  tesla dash cam watch  ends
[4] rt hasanihunter hey nebraska legislators prohibiting tesla  selling  cars  state  like telling customers
way y
[5] rt teslasono worked  today  driving  tesla model  sussex  inverness  miles will cost  tesla owner  using t
h
```

#tweets were pulled using tesla or tsla so we can clean it from the text

cleanset <-tm_map(cleanset, removeWords, c('tesla', 'tsla', 'teslas'))

inspect(cleanset[1:5])

```
[1] rt vincent  cuphead  latest  v update  definitely  new favorite game \n\nthanks elonmusk amp  spen
[2] rt exploreplanets weve teamed  zelectricbug  omaze  give  chance  win  oneofakind  vw convertible b
[3] rt eliburton  captured  hit  hit  run  dash cam watch  ends
[4] rt hasanihunter hey nebraska legislators prohibiting  selling  cars  state  like telling customers  way y
[5] rt teslasono worked  today  driving  model  sussex  inverness  miles will cost  owner  using th
```

DUKE FUQUA SCHOOL OF BUSINESS

Clean Corpus

```
#remove white spaces
cleanset <- tm_map(cleanset, stripWhitespace)
inspect(cleanset[1:5])
```

```
[1] rt vincent cuphead latest v update definitely new favorite game thanks elonmusk amp spen
[2] rt exploreplanets weve teamed zelectricbug omaze give chance win oneofakind vw convertible b
[3] rt eliburton captured hit hit run dash cam watch ends
[4] rt hasanihunter hey nebraska legislators prohibiting selling cars state like telling customers way y
[5] rt teslasono worked today driving model sussex inverness miles will cost owner using th
```

```
#lets now provide some structure to tweets by creating a matrix of
rows/columns #Create term document matrix (tdm)
tdm <- TermDocumentMatrix(cleanset)
tdm
```

```
<<TermDocumentMatrix (terms: 2243, documents: 1000)>>
Non-/sparse entries: 9948/2233052
Sparsity           : 100%
Maximal term length: 34
Weighting          : term frequency (tf)
```

DUKE FUQUA SCHOOL OF BUSINESS

Clean Corpus and Visualize Data

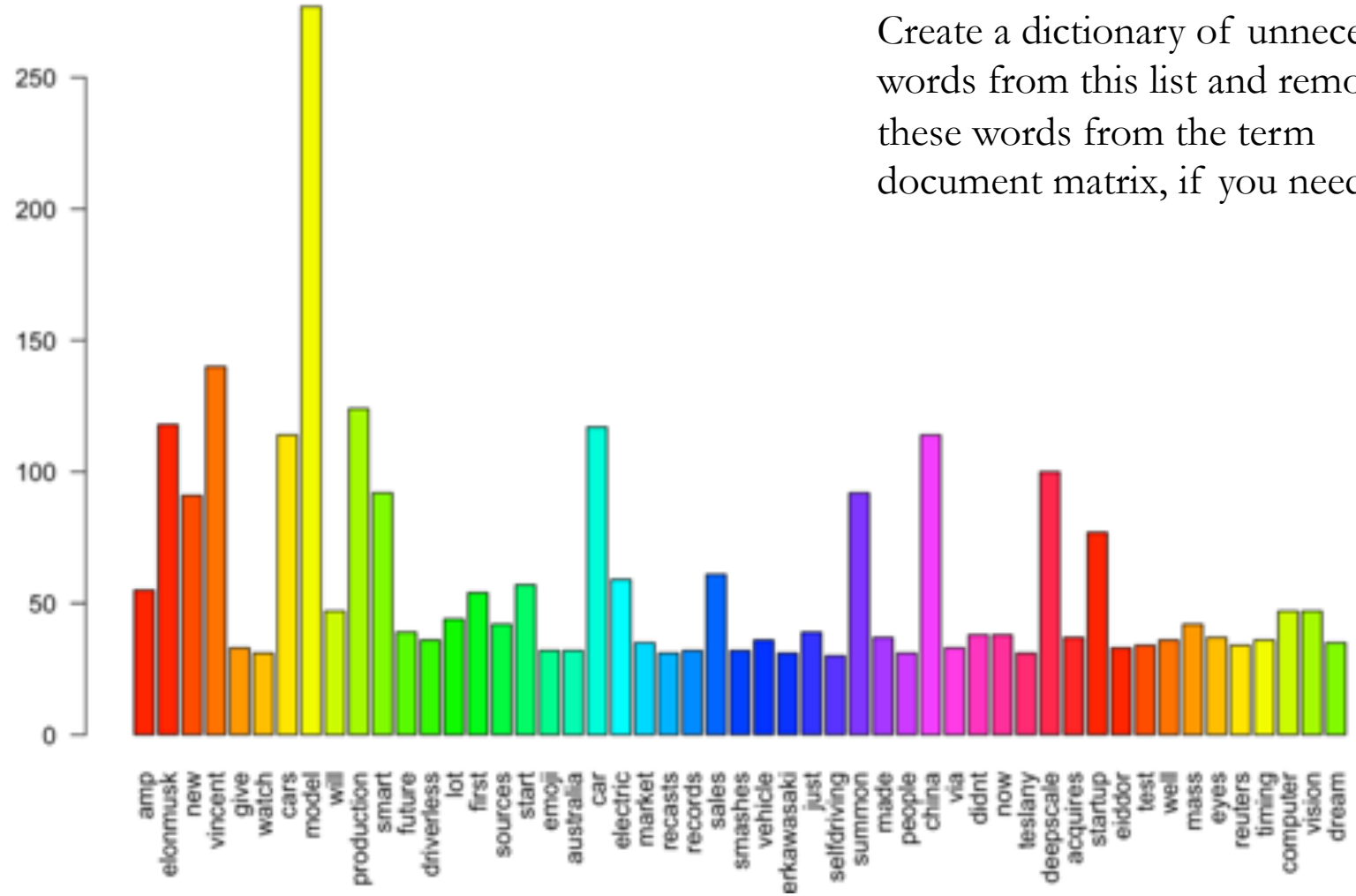
```
tdm <- as.matrix(tdm)
tdm[1:10, 1:20] #look at first 10 rows/terms and 20 tweets
```

	Docs																			
Terms	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
amp	1	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	1	0
cuphead	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
definitely	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
elonmusk	1	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1
favorite	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
game	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
latest	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
new	1	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	1	0
spen	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
thanks	1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0

```
#Sum the rows (how many times a term appears), Too many terms so we can create a subset of w where row sum is >30
w <- rowSums(tdm)
w <- subset(w, w>=30) #run "w" to see which words appear how many times
barplot(w, las = 2, col=rainbow(40))
```

DUKE FUQUA SCHOOL OF BUSINESS

Bar Plot of terms in Subset

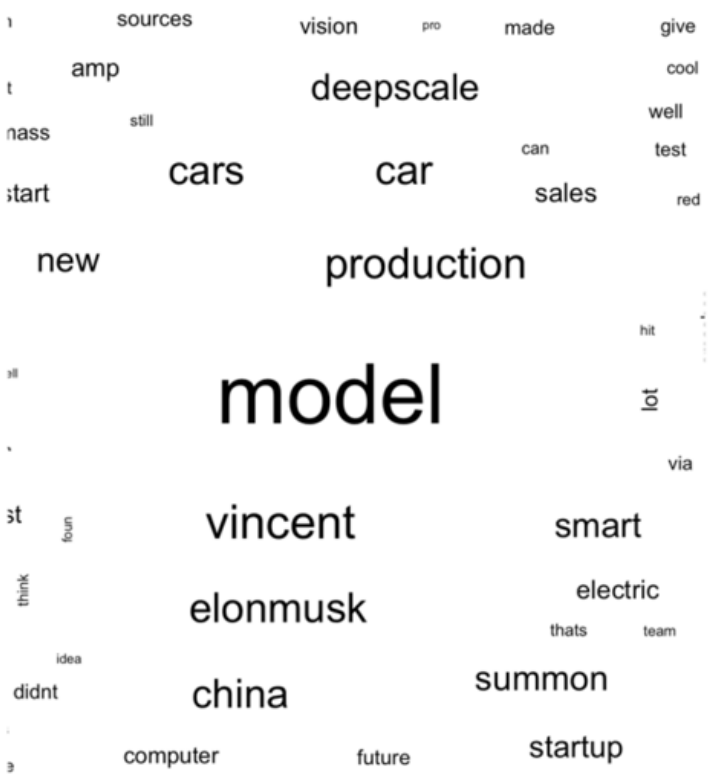


Create a dictionary of unnecessary words from this list and remove these words from the term document matrix, if you need to

DUKE FUQUA SCHOOL OF BUSINESS

Word Clouds

```
# Word Cloud
library(wordcloud)
library(RColorBrewer)
w <- sort(rowSums(tdm), decreasing=TRUE)
set.seed(9999)
wordcloud(words = names(w),
          freq=w,
          random.order =FALSE)  #words are specified in names in w dataframe, frequency is stored in w, random order=false
```

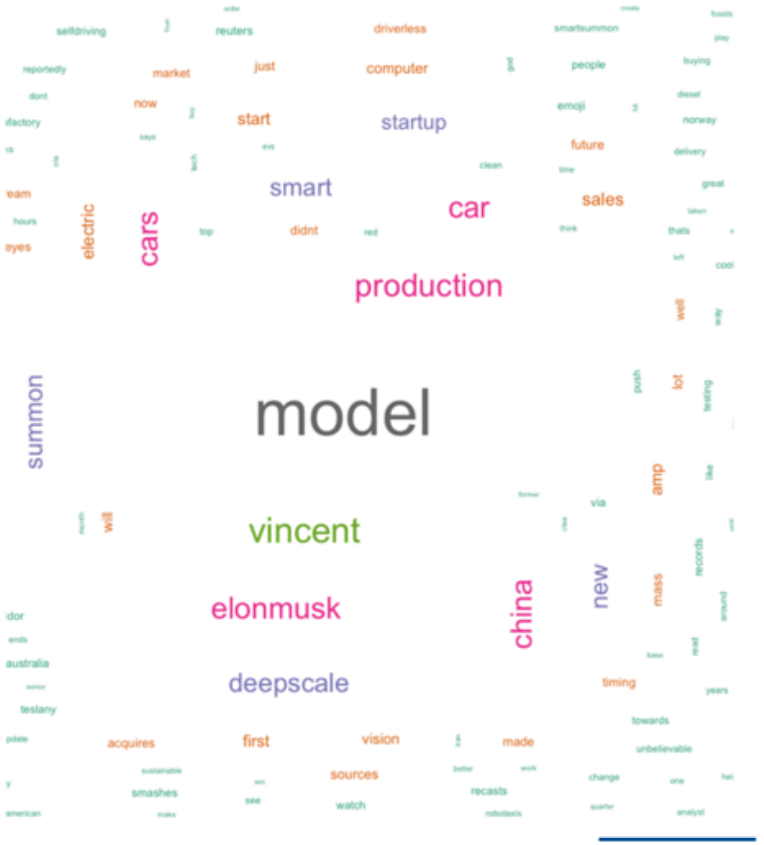


DUKE FUQUA SCHOOL OF BUSINESS

Word Clouds

#Specify that max words be no more than say, 200; Freq for terms (say they have to appear 5 times to be included); color words, specify scale (bigger words max size =3, smaller =0.2)
#rotate some words (rotation percentage = 30%)

```
wordcloud(words = names(w),
        freq=w,
        random.order =FALSE,
        max.words = 200,
        min.freq = 5,
        colors = brewer.pal(8, 'Dark2'),
        scale = c(3, 0.2),
        rot.per = .3)
```



Sentiment Analysis

```
#load packages
library(syuzhet)
library(lubridate)
library(ggplot2)
library(scales)
library(dplyr)

#reading Files
#take the initial apple tweet file (1000 obs and 16 vars for this)
#take the first column, text and put it into tweets dataframe
tweets <- iconv(tesla$text, to="utf-8-mac")
```

DUKE FUQUA SCHOOL OF BUSINESS

Sentiment Analysis

#obtain sentiment scores for each 1000 tweets; nrc_sentiment dictionary is called to calculate presence of eight emotions in their text file

s <-get_nrc_sentiment(tweets)

head(s)

	anger	anticipation	disgust	fear	joy	sadness	surprise	trust	negative	positive
1	0	0	0	0	1	0	0	1	0	1
2	0	0	0	0	1	0	1	1	0	1
3	1	1	0	1	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	1
6	0	0	0	0	0	0	0	0	0	0

```
> tail(s)
```

	anger	anticipation	disgust	fear	joy	sadness	surprise	trust	negative	positive
995	0	0	0	0	0	0	0	0	0	0
996	0	0	0	0	0	1	0	0	1	2
997	0	0	0	0	0	0	0	0	0	1
998	0	2	0	0	2	0	2	2	0	4
999	0	0	0	0	0	0	0	0	0	0
1000	0	0	0	0	0	0	0	0	0	1

DUKE FUQUA SCHOOL OF BUSINESS

Sentiment Analysis

tweets[996] #look at tweet number 996

```
[1] "Buy a #Tesla #Model3 and join the family. You won't regret it. https://t.co/lH4ImMIdXC"
```

```
996      0            0       0    0   0       1        0     0        1        2
```

#you could also look at phrases or words in these tweets to see if they lead to positive or negative')

get_nrc_sentiment('ridiculous')

```
anger anticipation disgust fear joy sadness surprise trust negative positive
    1            0       1    0   0       0        0     0        1        0
```

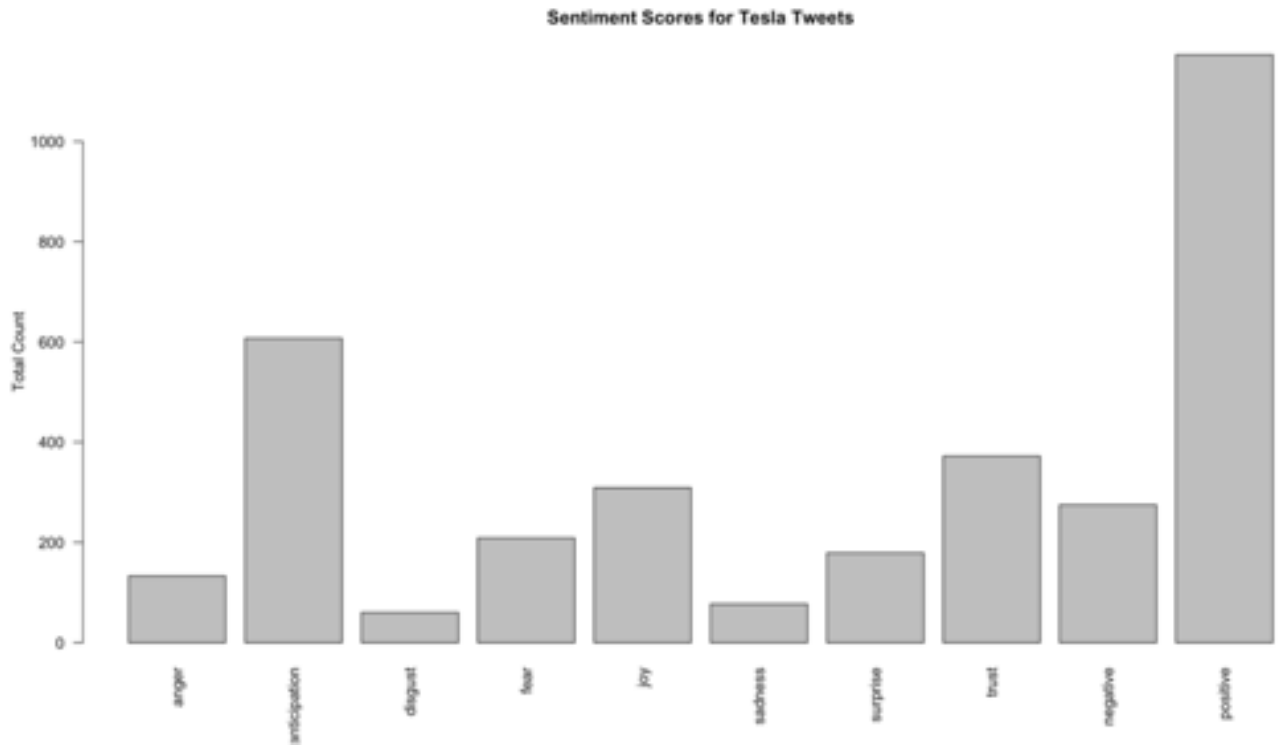
get_nrc_sentiment('finally tested summon feature')

```
anger anticipation disgust fear joy sadness surprise trust negative positive
    0            1       1    0   1       0        1     1        0        2
```

DUKE FUQUA SCHOOL OF BUSINESS

Sentiment Analysis

```
#lets sum the column scores across tweets for the plot
barplot(colSums(s),
        las = 2,
        ylab = 'Total Count',
        main ='Sentiment Scores for Tesla Tweets')
```



DUKE FUQUA SCHOOL OF BUSINESS