Code ▾

Step by Step Text Analytics on My Own Learning Data Log

Ariful Mondal (ariful dot mondal [at] gmail dot com)

30 November 2017

This is a quick text analysis using 3 months of my learning completion data on Lynda and LinkedIn Learning using *R*. Some of the required packages are listed below.

Hide

```
setwd("D:\\RProgramming")
## Load Packages
library("tm")
library("stringi")
library("wordcloud")
library("clue")
library("ggplot2")
library("RColorBrewer")
library("SnowballC")
library("RWeka")
```

- **tm:** (https://cran.r-project.org/web/packages/tm/tm.pdf) for text mining and natural language processing(NLP)
- **wordcloud:** (https://cran.r-project.org/web/packages/wordcloud/wordcloud.pdf) to create word clouds
- **stringi:** (https://cran.r-project.org/web/packages/stringi/stringi.pdf) to leverage character string processing facilities
- **clue:** (https://cran.r-project.org/web/packages/clue/clue.pdf) for Cluster Ensembles
- **SnowballC:** (https://cran.r-project.org/web/packages/SnowballC/SnowballC.pdf) for stemming text
- **RColorBrewer:** (https://cran.r-project.org/web/packages/RColorBrewer/RColorBrewer.pdf) color palettes for nice colours of the graphs
- **RWeka:** (https://cran.r-project.org/web/packages/RWeka/RWeka.pdf) text mining and N-Gram analysis
- **ggplot2** (https://cran.r-project.org/package=ggplot2/ggplot2.pdf) to Create Elegant Data Visualizations Using the Grammar of Graphics

1 Read data and print few lines

You may downlaod this data from my google drive here (https://drive.google.com/open?id=1KudIYfcJ7KbGvrhcssA4pRpIA3tMswHOw_mwxDTs9xQ)

Hide

```
lnd <- readLines("file:///D:/RProgramming/Ariful_Islam_Mondal_Training_Log_Nov_2017.csv",  en
coding = "UCS-2LE", skipNul = TRUE)
lnd[1:10]
```

```
##  [1] "COURSE, Skills"                                                              

##  [2] "Learning Information Governance,Document Management"                         

##  [3] "Open Data: Unleashing Hidden Value,\"Big Data, Data Analysis\""             

##  [4] "Learning Public Data Sets,\"Data Analysis, Microsoft Excel\""               

##  [5] "Smarter Cities: Using Data to Drive Urban Innovation,\"Big Data, Data Management\""

##  [6] "Blockchain Basics,\"Corporate Finance, Databases\""                          

##  [7] "Online Marketing Foundations: Digital Marketing Research,Lead Generation"    

##  [8] "Marketing Analytics: Presenting Digital Marketing Data,\"Google Analytics, Web Analy
tics\""
##  [9] "Calculating Gross Profit with Google Analytics,\"Google Analytics, E-commerce\""

## [10] "Microsoft Azure for Developers,\"Microsoft Azure, Cloud Development\""
```

2 Basic text processing and cleaning

- Remove non-English characters, letters etc. using `iconv()` and option `latin1`
- Remove special characters with spaces using `gsub()` and regular expression `[^0-9a-z]`
- Remove duplicate characters using `gsub()` and regular expression
- Remove special numbers with spaces using `gsub()` and regular expression
- Remove multiple spaces to one using `gsub()` and regular expression

To know more on `incon()` click here (https://www.rdocumentation.org/packages/base/versions/3.4.1/topics/iconv), for `gsub()` click here (https://www.r-bloggers.com/regular-expression-and-associated-functions-in-r/) and click here (https://en.wikipedia.org/wiki/Regular_expression) to know more on `regular expression` , also view `regex` (https://stat.ethz.ch/R-manual/R-devel/library/base/html/regex.html).

Hide

```
# Remove non-English characters, letters etc.
# Help ?inconv
lnd<-iconv(lnd, "latin1", "ASCII", sub="")
# Remove special characters with spaces
# Help ?gsub
lnd <- gsub("[^0-9a-z]", " ", lnd, ignore.case = TRUE)
# Remove duplicate characters
lnd <- gsub('([[:alpha:]])\\1+', '\\1\\1', lnd)
# Remove special numbers with spaces
lnd <- gsub("[^a-z]", " ", lnd, ignore.case = TRUE)
# Remove multiple spaces to one
lnd <- gsub("\\s+", " ", lnd)
lnd <- gsub("^\\s", "", lnd)
lnd <- gsub("\\s$", "", lnd)
```

Print after clean up…

Hide

```
# Summary
lnd[1:10]
```

```
##  [1] "COURSE Skills"

##  [2] "Learning Information Governance Document Management"

##  [3] "Open Data Unleashing Hidden Value Big Data Data Analysis"

##  [4] "Learning Public Data Sets Data Analysis Microsoft Excel"

##  [5] "Smarter Cities Using Data to Drive Urban Innovation Big Data Data Management"

##  [6] "Blockchain Basics Corporate Finance Databases"

##  [7] "Online Marketing Foundations Digital Marketing Research Lead Generation"

##  [8] "Marketing Analytics Presenting Digital Marketing Data Google Analytics Web Analytic
s"
##  [9] "Calculating Gross Profit with Google Analytics Google Analytics E commerce"

## [10] "Microsoft Azure for Developers Microsoft Azure Cloud Development"
```

Hide

```
summary(lnd)
```

```
##    Length     Class      Mode 
##       115 character character
```

Hide

```
str(lnd)
```

```
##  chr [1:115] "COURSE Skills" ...
```

3 Create a corpus

Create a virtual corpus using `Vcorpus()` function.

Hide

```
# create Corpus
# Help ??VCorpus
myCorpus <- VCorpus(VectorSource(lnd))
myCorpus
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 115
```

4 Transformation of text

[Optional for already cleaned data]

Perform necessary transformation/preprocessing activities using `tm_map()` from `tm` package. The objective is to have clean texts by removing stop words, punctuation, multiple white spaces etc. We will perform the following transformations

- **tolower:** (https://www.rdocumentation.org/packages/quanteda/versions/0.99.12/topics/toLower) normalize text to small cases. For example `My name Is Ariful` will be converted to small case my name is ariful .

- **stopwords:** (https://www.rdocumentation.org/packages/tm/versions/0.7-1/topics/stopwords) remove stop words using English dictionary. For example remove words like “a”, “and”, “but”, “how”, “or”, and “what”
- **removePunctuation:** (https://www.rdocumentation.org/packages/tm/versions/0.7-1/topics/removePunctuation) remove punctuation marks such as ! " # $ % & ’ ( ) * + , - . / : ; < = > ? @ [ ] ^ _ ` { | } ~.
- **stripWhitespace:** (https://www.rdocumentation.org/packages/tm/versions/0.7-1/topics/stripWhitespace) Strip extra white space from a text document. Multiple white space characters are collapsed to a single blank
- **PlainTextDocument:** (https://www.rdocumentation.org/packages/tm/versions/0.7-1/topics/PlainTextDocument) convert document to plain text format

Hide

```
# Help ??tm_map'

# Normalize to small cases
myCorpus <- tm_map(myCorpus, content_transformer(tolower))

# Remove Stop Words
myCorpus <- tm_map(myCorpus, removeWords, stopwords("english"))

# Remove Punctuation
myCorpus <- tm_map(myCorpus, removePunctuation)

# Remove Numbers
 myCorpus <- tm_map(myCorpus, removeNumbers)

# Create plain text documents
myCorpus <- tm_map(myCorpus, PlainTextDocument)

# Stem words in a text document using Porter's stemming algorithm.
myCorpus <- tm_map(myCorpus, stemDocument, "english")

# Strip White Spaces
myCorpus <- tm_map(myCorpus, stripWhitespace)
```

5 Term Document Matrix & N-Gram Analysis

Now we will use `TermDocumentMatrix()` to create a `document-term matrix` or `term-document matrix` which is a mathematical matrix that describes the frequency of terms/words/strings that occur in a collection of documents. In a document-term matrix, rows correspond to documents in the collection and columns correspond to terms. There are various schemes for determining the value that each entry in the matrix should take. Read more on wiki (https://en.wikipedia.org/wiki/Document-term_matrix).

In the fields of `computational linguistics` and `probability` , an `n-gram` is a contiguous sequence of n items from a given sequence of text or speech. The items can be `phonemes` , `syllables` , `letters` , `words` or `base pairs` according to the application.

The n-grams typically are collected from a text or speech corpus. When the items are words, `n-grams` may also be called shingles.

- An n-gram of size 1 is referred to as a `"unigram"` ;
- An n-gram of size 2 is a `"bigram"` (or, less commonly, a “digram”);
- An n-gram of size 3 is a `"trigram"` .

Larger sizes are sometimes referred to by the value of n in modern language, e.g., “*four-gram*”, “*five-gram*”, and so on. [wiki (https://en.wikipedia.org/wiki/N-gram)].

5.1 Crate Unigram

Hide

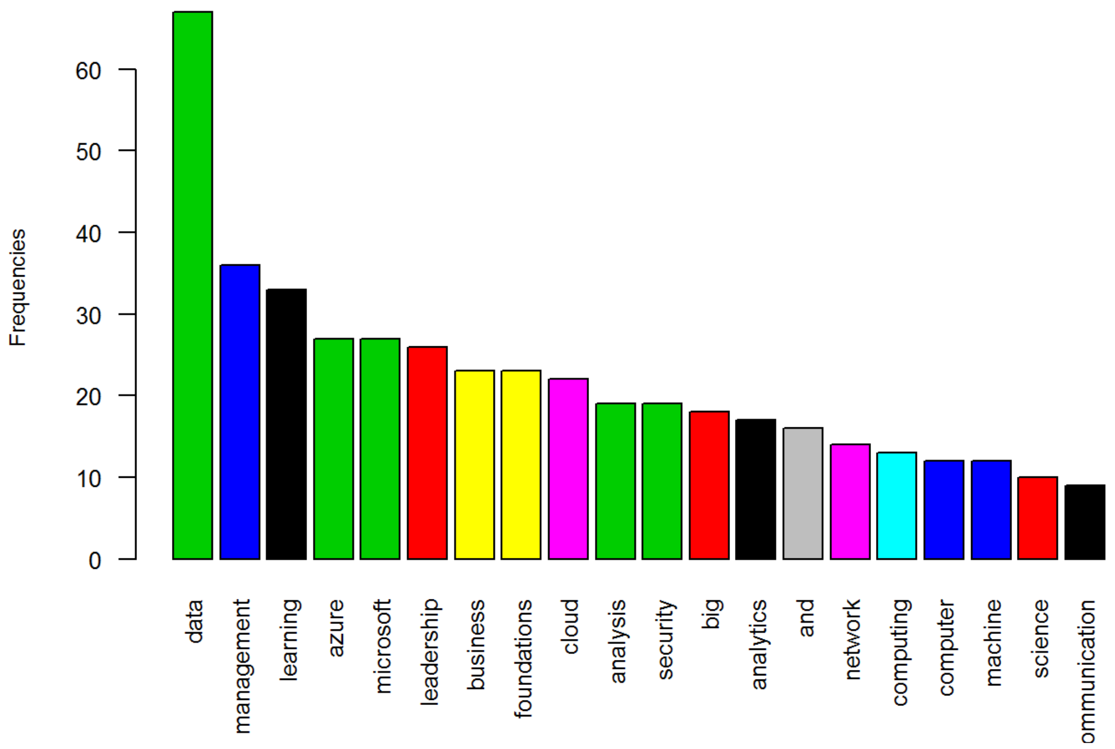
```
unitdm <- TermDocumentMatrix(myCorpus)
mat <- as.matrix(unitdm)
wf <- sort(rowSums(mat),decreasing=TRUE)
df <- data.frame(word = names(wf),freq=wf)
head(df, 10)
```

```
##                    word freq
## data               data   67
## management   management   36
## learning       learning   33
## azure             azure   27
## microsoft     microsoft   27
## leadership   leadership   26
## business       business   23
## foundations foundations   23
## cloud             cloud   22
## analysis       analysis   19
```

5.1.1 Ploting unigram

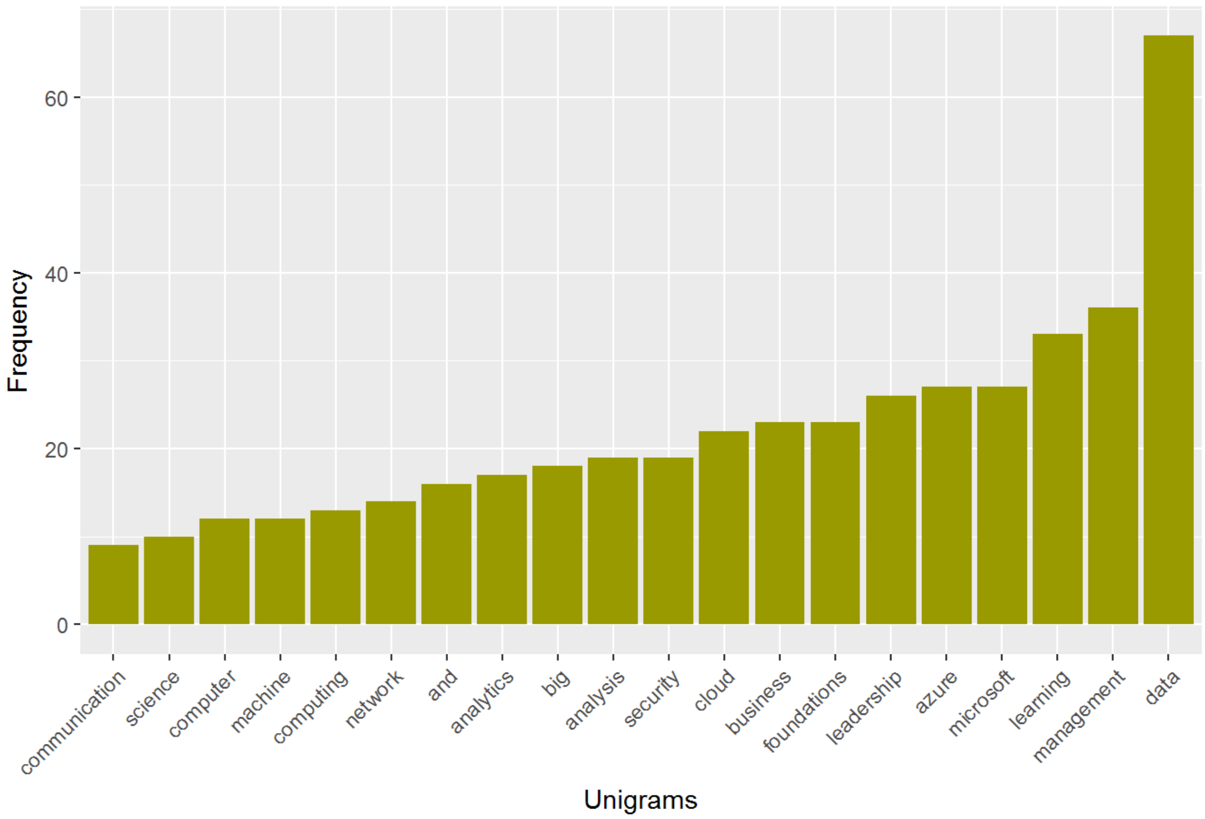
Hide

```
barplot(df[1:20,]$freq, las = 2, names.arg = df[1:20,]$word,
        col =df[1:20,]$freq, main ="",
        ylab = "Frequencies", cex.axis=.8, cex = .8, cex.lab=0.75, cex.main=.75)
```



Hide

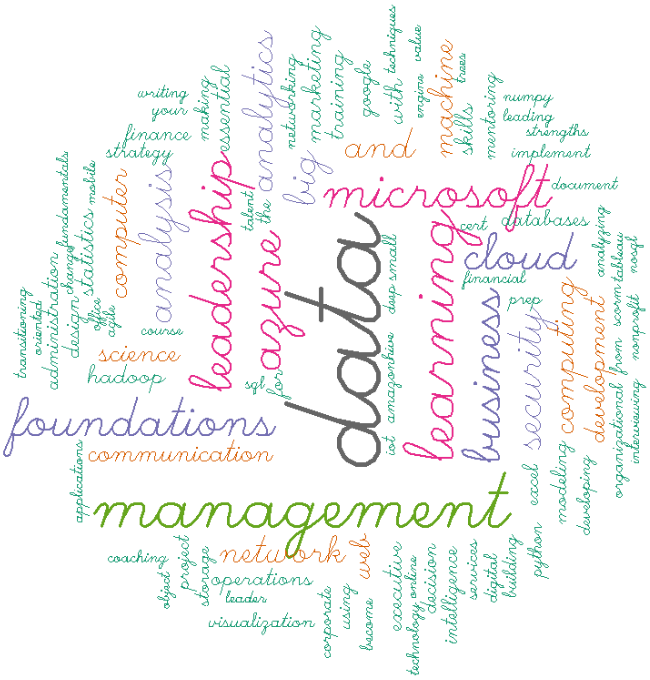
```
ggplot(df[1:20,], aes(x = reorder(df[1:20,]$word, df[1:20,]$freq), y = df[1:20,]$freq)) +
    geom_bar(stat = "identity", fill = "#999900") +
    labs(title = " ") +
    xlab("Unigrams") +
    ylab("Frequency")+
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



5.1.2 Create Word cloud with unigram

Hide

```
set.seed(1234)
wordcloud(words = df$word, freq = df$freq, min.freq = 1,
          max.words=100, random.order=FALSE, rot.per=0.75, las=3,
          colors=brewer.pal(8, "Dark2"), c(5,.5), vfont=c("script","plain"))
```



5.1.3 Find frequently occured words

Hide

```
findFreqTerms(unitdm, lowfreq = 5)
```

```
##  [1] "administration" "analysis"       "analytics"      "and"
##  [5] "azure"          "big"            "business"       "cloud"
##  [9] "communication"  "computer"       "computing"      "data"
## [13] "databases"      "decision"       "design"         "development"
## [17] "essential"      "excel"          "executive"      "finance"
## [21] "for"            "foundations"    "google"         "hadoop"
## [25] "intelligence"   "leadership"     "learning"       "machine"
## [29] "management"     "marketing"      "microsoft"      "modeling"
## [33] "network"        "operations"     "science"        "security"
## [37] "skills"         "statistics"     "training"       "web"
## [41] "with"
```

5.1.4 Find Associations between Words

Find associations in document-term or term-document matrix using function `findAssocs(x, terms, corlimit)` from *tm* package.

- **x**: A DocumentTermMatrix or a TermDocumentMatrix.
- **terms**: a character vector holding terms.
- **corlimit**: a numeric vector (of the same length as terms; recycled otherwise) for the (inclusive) lower correlation limits of each term in the range from zero to one.

1. Find associated words with *data*…

Hide

```
findAssocs(unitdm, terms = "data", corlimit = 0.35)
```

```
## $data
##            big        science       modeling       analysis         career 
##           0.71           0.56           0.52           0.48           0.48 
## certifications       database          paths          steps   intelligence 
##           0.48           0.48           0.48           0.48           0.40 
##      analytics  visualization 
##           0.38           0.35
```

2. Find associated words with *machine*…

Hide

```
findAssocs(unitdm, terms = "machine", corlimit = 0.35)
```

```
## $machine
##    learning       trees estimations mathematica      python 
##        0.68        0.49        0.47        0.47        0.38
```

3. Find associated words with *management*…

Hide

```
findAssocs(unitdm, terms = "management", corlimit = 0.35)
```

```
## $management
## operations      small  nonprofit    finding       high potentials 
##       0.62       0.51       0.41       0.35       0.35       0.35 
##    records  retaining 
##       0.35       0.35
```

4. Find associated words with *azure*…

Hide

```
findAssocs(unitdm, terms = "azure", corlimit = 0.35)
```

```
## $azure
##      microsoft      computing      implement          cloud     networking 
##           0.85           0.58           0.53           0.52           0.52 
## administration         active      directory        virtual    development 
##           0.51           0.40           0.40           0.36           0.35
```

5. Find associated words with *security*…

Hide

```
findAssocs(unitdm, terms = "security", corlimit = 0.35)
```

```
## $security
##      computer       network          cert    compliance       comptia 
##          0.78          0.68          0.53          0.53          0.53 
##   operational          prep         asset         cissp  cryptography 
##          0.53          0.53          0.40          0.40          0.40 
## investigation      response 
##          0.40          0.40
```

5.2 Crate Bigram

Hide

```
BigramTokenizer <- function(x) NGramTokenizer(x, Weka_control(min = 2, max = 2)) # Create bigram tokenizer using RWeka
bitdm <- TermDocumentMatrix(myCorpus, control = list(tokenize = BigramTokenizer)) # Create bigram
inspect(bitdm[15:30,1:20]) # Inspect few bigrams
```

```
## <<TermDocumentMatrix (terms: 16, documents: 20)>>
## Non-/sparse entries: 6/314
## Sparsity           : 98%
## Maximal term length: 20
## Weighting          : term frequency (tf)
## Sample             :
##                     Docs
## Terms                1 14 19 2 3 4 5 6 7 8
##   ai advanced        0  0  0 0 0 0 0 0 0 0
##   ai foundations     0  0  0 0 0 0 0 0 0 0
##   amazon web         0  0  0 0 0 0 0 0 0 0
##   analysis business  0  0  0 0 0 0 0 0 0 0
##   analysis data      0  1  0 0 0 0 0 0 0 0
##   analysis microsoft 0  0  0 0 0 1 0 0 0 0
##   analysis office    0  0  1 0 0 0 0 0 0 0
##   analysis web       0  1  0 0 0 0 0 0 0 0
##   analytics business 0  1  0 0 0 0 0 0 0 0
##   analytics career   0  1  0 0 0 0 0 0 0 0
```

Hide

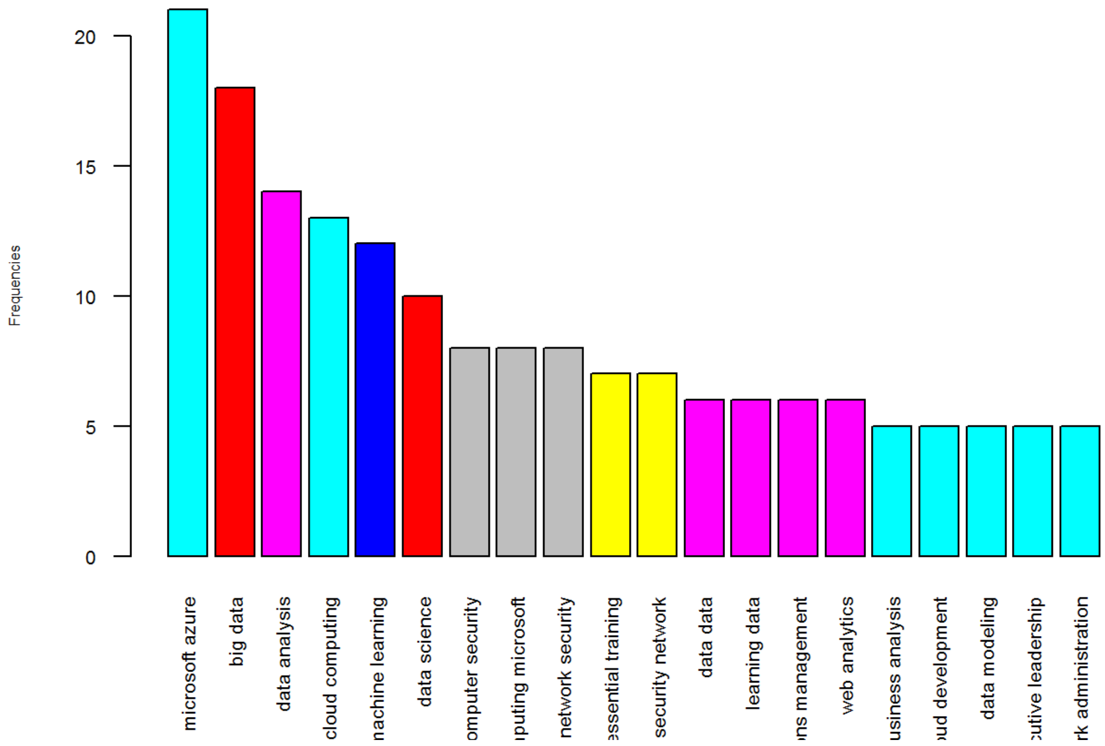
```
mat_bigram <- as.matrix(bitdm)
wf_bigram <- sort(rowSums(mat_bigram),decreasing=TRUE)
df1 <- data.frame(word = names(wf_bigram),freq=wf_bigram)
head(df1, 10)
```

```
##                                    word freq
## microsoft azure         microsoft azure   21
## big data                       big data   18
## data analysis             data analysis   14
## cloud computing         cloud computing   13
## machine learning       machine learning   12
## data science               data science   10
## computer security     computer security    8
## computing microsoft computing microsoft    8
## network security       network security    8
## essential training   essential training    7
```

5.2.1 Ploting Bigram

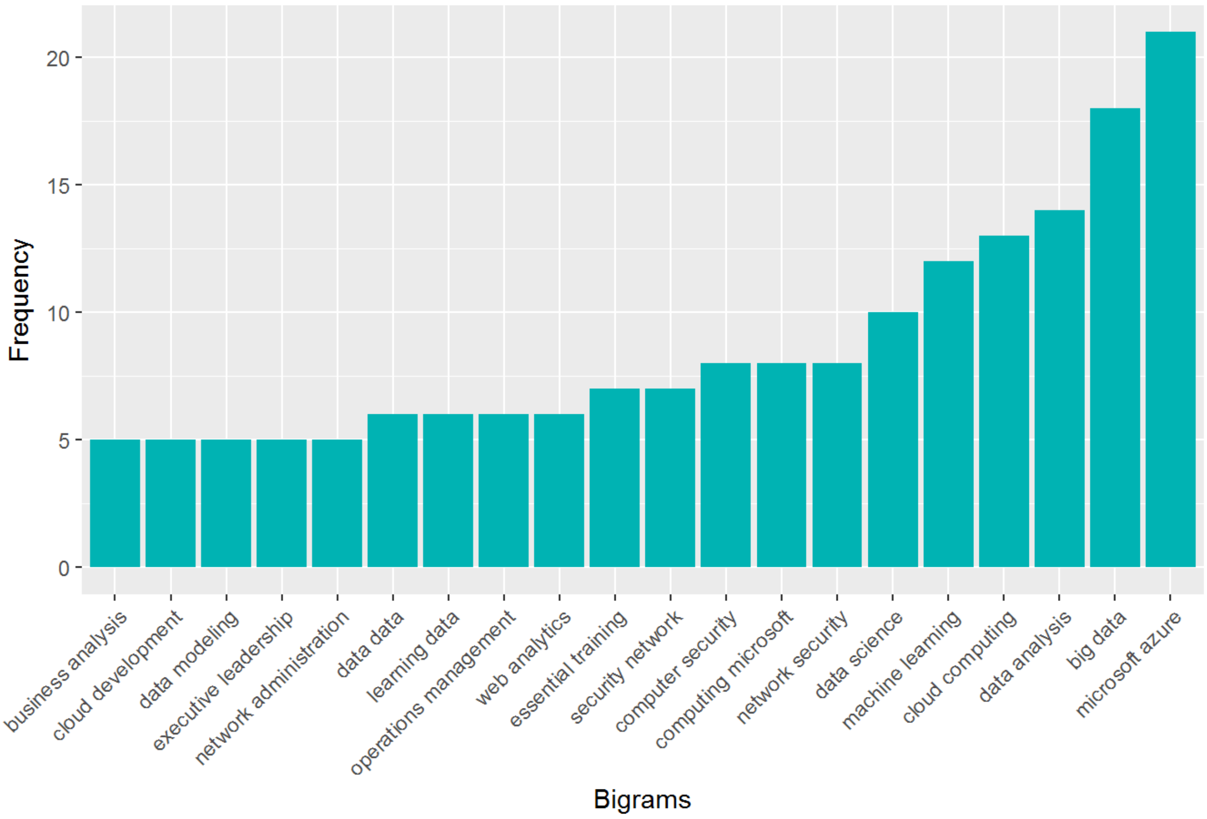
Hide

```
biplot<-barplot(df1[1:20,]$freq, las = 2, names.arg = df1[1:20,]$word,
        col = df1[1:20,]$freq, main ="",
        ylab = "Frequencies", cex.axis=.65, cex = .65, cex.lab=0.5, cex.main=.75)
```



Hide

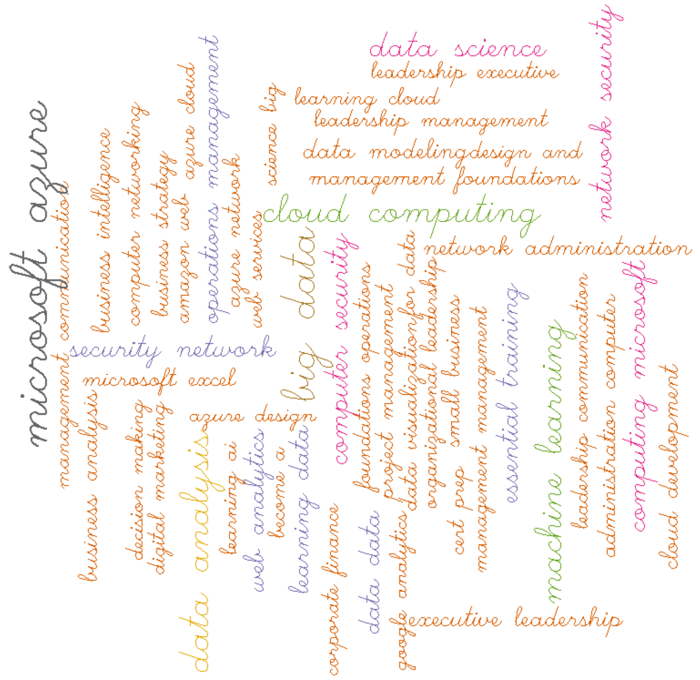
```
ggplot(df1[1:20,], aes(x = reorder(df1[1:20,]$word, df1[1:20,]$freq), y = df1[1:20,]$freq)) +
    geom_bar(stat = "identity", fill = "#00b3b3") +
    labs(title = " ") +
    xlab("Bigrams") +
    ylab("Frequency")+
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



5.2.2 Create Word cloud with Bigram

Hide

```
set.seed(1234)
wordcloud(words = df1$word, freq = df1$freq, min.freq = 3,
          max.words=100, random.order=T, rot.per=0.75,
          colors=brewer.pal(8, "Dark2"), c(2,.7), vfont=c("script","plain"))
```



5.3 Crate Trigram

Hide

```
TrigramTokenizer <- function(x) NGramTokenizer(x, Weka_control(min = 3, max = 3)) # Create tr
igram tokenizer using RWeka
tritdm <- TermDocumentMatrix(myCorpus, control = list(tokenize = TrigramTokenizer)) # Create
 trigram
inspect(tritdm[15:30,1:20]) # Inspect few trigrams
```

```
## <<TermDocumentMatrix (terms: 16, documents: 20)>>
## Non-/sparse entries: 6/314
## Sparsity           : 98%
## Maximal term length: 29
## Weighting          : term frequency (tf)
## Sample             :
##                               Docs
## Terms                          1 14 19 2 3 4 5 6 7 8
##   ai advanced decision         0  0  0 0 0 0 0 0 0 0
##   ai foundations decision      0  0  0 0 0 0 0 0 0 0
##   ai foundations value         0  0  0 0 0 0 0 0 0 0
##   amazon web services          0  0  0 0 0 0 0 0 0 0
##   analysis data management     0  1  0 0 0 0 0 0 0 0
##   analysis microsoft excel     0  0  0 0 0 1 0 0 0 0
##   analysis office web          0  0  1 0 0 0 0 0 0 0
##   analysis web analytics       0  1  0 0 0 0 0 0 0 0
##   analytics business analysis 0  1  0 0 0 0 0 0 0 0
##   analytics career paths       0  1  0 0 0 0 0 0 0 0
```

Hide

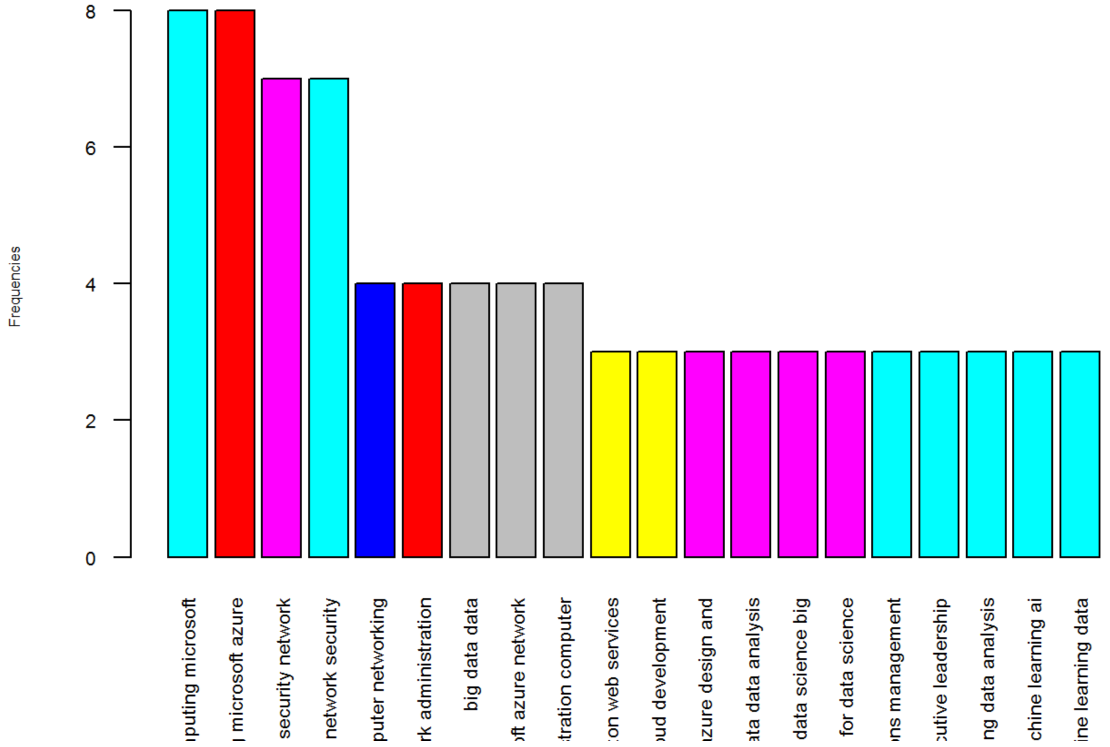
```
mat_trigram <- as.matrix(tritdm)
wf_trigram <- sort(rowSums(mat_trigram),decreasing=TRUE)
df2 <- data.frame(word = names(wf_trigram),freq=wf_trigram)
head(df2, 10)
```

```
##                                                                   word freq
## cloud computing microsoft                    cloud computing microsoft    8
## computing microsoft azure                    computing microsoft azure    8
## computer security network                    computer security network    7
## security network security                    security network security    7
## administration computer networking administration computer networking    4
## azure network administration              azure network administration    4
## big data data                                            big data data    4
## microsoft azure network                        microsoft azure network    4
## network administration computer    network administration computer    4
## amazon web services                                amazon web services    3
```

5.3.1 Ploting Trigram

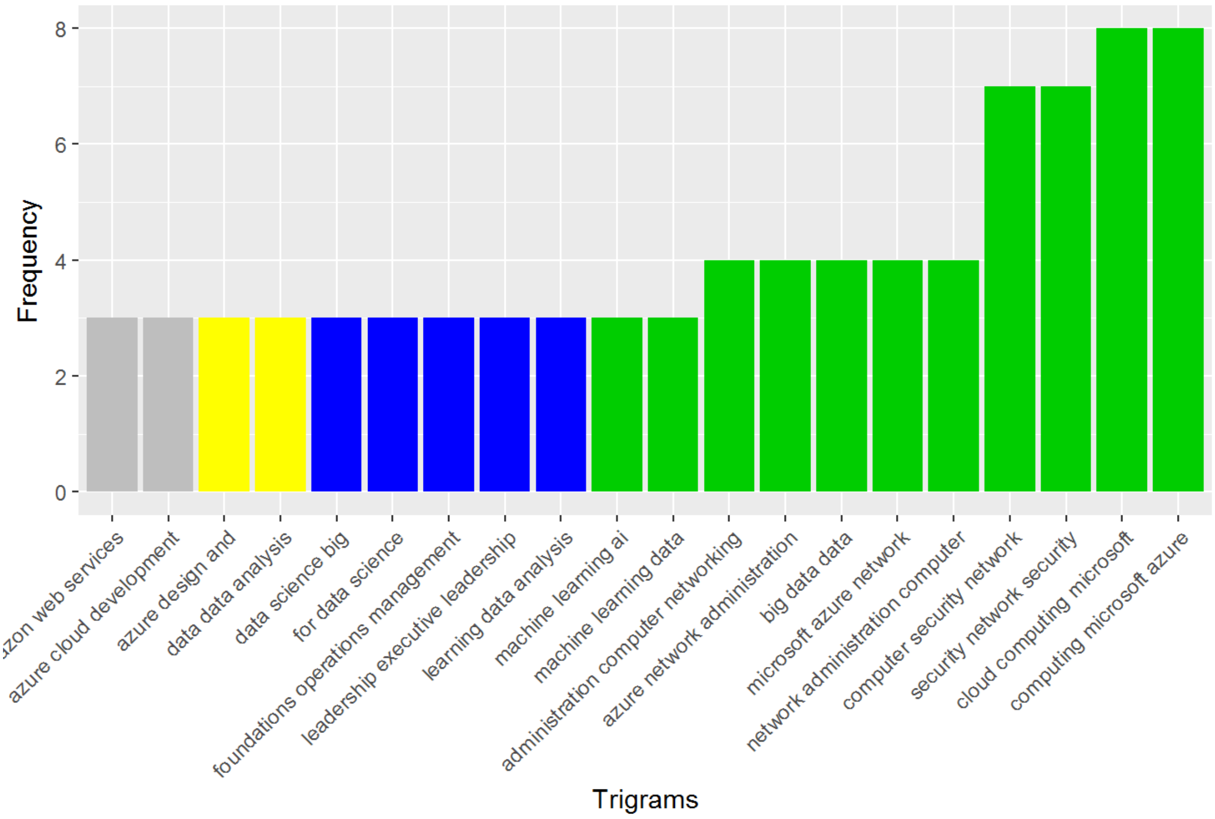
Hide

```
triplot<-barplot(df2[1:20,]$freq, las = 2, names.arg = df2[1:20,]$word,
        col = df1[1:20,]$freq, main ="",
        ylab = "Frequencies", cex.axis=.65, cex = .65, cex.lab=0.5, cex.main=.75)
```



Hide

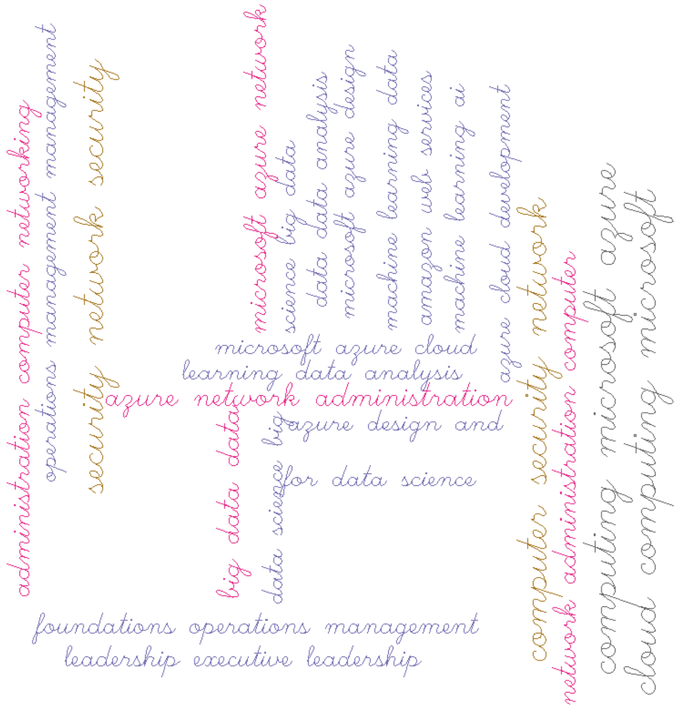
```
ggplot(df2[1:20,], aes(x = reorder(df2[1:20,]$word, df2[1:20,]$freq), y = df2[1:20,]$freq)) +
    geom_bar(stat = "identity", fill=df2[1:20,]$freq) +
    labs(title = " ") +
    xlab("Trigrams") +
    ylab("Frequency")+
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



5.3.2 Create Word cloud with Trigram

Hide

```
set.seed(1234)
wordcloud(words = df2$word, freq = df2$freq, min.freq = 3,
          max.words=100, random.order=T, rot.per=0.75,
          colors=brewer.pal(8, "Dark2"), c(1.7,.7), vfont=c("script","plain"))
```



More coming soon….

6 Appendix

6.1 About R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

6.1.1 Including summary

Hide

```
summary(cars)
```