Text as Data HW 1

Chenjie Su – cs5998

3/04/2020

```r
rm(list = ls())
# import libraries
library(quanteda)
```

```
## Package version: 1.5.2

## Parallel computing: 2 of 8 threads used.

## See https://quanteda.io for tutorials and examples.

##
## Attaching package: 'quanteda'

## The following object is masked from 'package:utils':
##
##     View
```

```r
library(quanteda.corpora)
library(gutenbergr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(stylest)
library(corpus)
library(sophistication)
library(pbapply)
```

Q1a

```r
inaugural <- data_corpus_inaugural
#the inaugural addresses given by Richard Nixon in 1969
rn1969 <- corpus_subset(inaugural, Year=='1969') #get the corpus subset
rn1969T <- texts(rn1969) #get the texts
rn1969tokens <- tokens(rn1969T, remove_punct = TRUE) #tokenizing
TTR1969 <- textstat_lexdiv(rn1969tokens, measure = "TTR") #get the TTR
TTR1969
```

```
##     document       TTR
## 1 1969-Nixon 0.3333333
```

```r
#the inaugural addresses given by Richard Nixon in 1973
rn1973 <- corpus_subset(inaugural, Year=='1973')
rn1973T <- texts(rn1973)
rn1973tokens <- tokens(rn1973T, remove_punct = TRUE)
TTR1973 <- textstat_lexdiv(rn1973tokens, measure = "TTR")
TTR1973
```

```
##     document       TTR
## 1 1973-Nixon 0.2815103
```

For the inaugural addresses (of president Richard Nixon) in 1969, TTR=0.3333333, in 1973, TTR=0.2815103.

Q1b

```r
# create dfm, remove the punctuation without other prepossesing
n1969_n1973_dfm <- dfm(c(rn1969T, rn1973T), remove_punct = TRUE)
n1969_n1973_dfm[,1:6]
```

```
## Document-feature matrix of: 2 documents, 6 features (8.33% sparse).
## 2 x 6 sparse Matrix of class "dfm"
##             features
## docs          senator dirksen mr chief justice vice
##   1969-Nixon       1       1  2     1       1    2
##   1973-Nixon       1       0  3     1       1    1
```

```r
# Calculate similarity
similarity_nixon1969_nixon1973 <- textstat_simil(n1969_n1973_dfm,
                                                 margin = "documents",
                                                 method = "cosine")
as.matrix(similarity_nixon1969_nixon1973)
```

```
##            1969-Nixon 1973-Nixon
## 1969-Nixon  1.0000000  0.9223596
## 1973-Nixon  0.9223596  1.0000000
```

In this question, we need to create a document feature matrix of the two speeches. And calculate the cosine similarity which is 0.9223596.

Q2a

```r
# stemming the words
stemmed_1969 <- tokens_wordstem(rn1969tokens)
stemmed_1973 <- tokens_wordstem(rn1973tokens)
# redo 1a
TTR1969a <- textstat_lexdiv(stemmed_1969, measure = "TTR")
TTR1973a <- textstat_lexdiv(stemmed_1973, measure = "TTR")
TTR1969a
```

```
##     document       TTR
## 1 1969-Nixon 0.2947269
```

```r
TTR1973a
```

```
##     document       TTR
## 1 1973-Nixon 0.2476402
```

```r
# redo 1b
n1969_n1973_dfm_2a <- dfm(c(stemmed_1969, stemmed_1973))
similarity_nixon1969_nixon1973_2a <- textstat_simil(n1969_n1973_dfm_2a,
                                                    margin = "documents",
                                                    method = "cosine")
as.matrix(similarity_nixon1969_nixon1973_2a)
```

```
##            1969-Nixon 1973-Nixon
## 1969-Nixon  1.0000000  0.9193894
## 1973-Nixon  0.9193894  1.0000000
```

Q2a: Results: TTR decrease, similarity decrease.

After stemming the words, the TTR will decrease. TTR = total types/total tokens, when we stem the words, the tokens keep the same, however the types get smaller since types are the sets of unique tokens. The tokens who have the same root will become one type. So TTR get smaller.

And for the similarity, cosine similarity captures the style or topic in document. When we stem the words, if their styles are different, there will be less common words in two documents. The direction of the documents is getting further, their difference become larger.

Q2b

```r
# removing stop words
nostop_1969 <- tokens_select(rn1969tokens, pattern = stopwords('en'), selection = 'remove')
nostop_1973 <- tokens_select(rn1973tokens, pattern = stopwords('en'), selection = 'remove')
# redo 1a
TTR1969b <- textstat_lexdiv(nostop_1969, measure = "TTR")
TTR1973b<- textstat_lexdiv(nostop_1973, measure = "TTR")
print(TTR1969b)
```

```
##     document       TTR
## 1 1969-Nixon 0.6040856
```

```
print(TTR1973b)
```

```
##     document       TTR
## 1 1973-Nixon 0.4947121
```

```
#redo 1b
n1969_n1973_dfm_2b <- dfm(c(nostop_1969, nostop_1973))
similarity_nixon1969_nixon1973_2b <- textstat_simil(n1969_n1973_dfm_2b,
                                                    margin = "documents",
                                                    method = "cosine")
as.matrix(similarity_nixon1969_nixon1973_2b)
```

```
##            1969-Nixon 1973-Nixon
## 1969-Nixon   1.000000   0.630754
## 1973-Nixon   0.630754   1.000000
```

Q2b: Results: TTR increase, similarity decrease.

After removing stop words, TTR will rise up. The tokens decreased, types are decreasing too but not that much. Because the major words still exist, those unique sets(types) keep the same. So the type-token ratio increased. For the similarity: generally, stop words are very common in the documents, when we remove the stop words, the difference between two documents is larger than before. So the cosine similarity is getting smaller.

Q2c

```
# Converting all words to lowercase
lowercase_1969 <- tokens_tolower(rn1969tokens)
lowercase_1973 <- tokens_tolower(rn1973tokens)
# redo 1a
TTR1969c <- textstat_lexdiv(lowercase_1969, measure = "TTR")
TTR1973c<- textstat_lexdiv(lowercase_1973, measure = "TTR")
print(TTR1969c)
```

```
##     document       TTR
## 1 1969-Nixon 0.3333333
```

```
print(TTR1973c)
```

```
##     document       TTR
## 1 1973-Nixon 0.2815103
```

```
#redo 1b
n1969_n1973_dfm_2c <- dfm(c(lowercase_1969, lowercase_1973))
similarity_nixon1969_nixon1973_2c <- textstat_simil(n1969_n1973_dfm_2c,
                                                    margin = "documents",
                                                    method = "cosine")
as.matrix(similarity_nixon1969_nixon1973_2c)
```

```
##             1969-Nixon 1973-Nixon
## 1969-Nixon  1.0000000  0.9223596
## 1973-Nixon  0.9223596  1.0000000
```

Q2c: Results: TTR and similarity keep the same.

After converting all words to lowercase, TTR and similarity will not change. Probably because there are no special uppercase words in two documents. So lowercase the words won't change the type-token ratio and similarity between two documents. But there might be some words like the first word of the sentence -> lowercase, less types and more tokens, TTR will increase a little bit.

Q2d

```
# use tfidf based on the whole corpus
full_dfm <- dfm(inaugural, remove_punct = TRUE)
topfeatures(full_dfm)
```

```
##   the    of   and    to    in     a   our  that    we    be
## 10082  7103  5310  4526  2785  2246  2181  1789  1739  1481
```

```
# Frequency weighting
weighted_dfm <- dfm_tfidf(full_dfm) # uses the absolute frequency of terms in each document
topfeatures(weighted_dfm)
```

```
##      america        union       should constitution     congress      freedom
##     52.68044     51.14846     42.10689     40.21661     39.13390     38.31822
##          you      revenue         upon       public
##     35.99430     34.11779     33.88348     33.74429
```

```
# Relative frequency weighting
proportional <- dfm_weight(full_dfm, scheme = "prop")
normalized <- dfm_tfidf(full_dfm, scheme_tf = "prop")
topfeatures(normalized)
```

```
##    america      union       your    freedom  americans        you      today
## 0.02790843 0.02005852 0.01908852 0.01775268 0.01743558 0.01725678 0.01702545
##  democracy     should        let
## 0.01669462 0.01610641 0.01577398
```

Q2b: Results: TTR and similarity keep the same. I just used the codes from the lab recitation to analyze the whole corpus. There is no prepossessing based on the documents, so the TTR and similarity won't change. The tf-idf is term frequency-inverse document frequency. I think tf-idf is just a way to demonstrate if a given word is indicative of difference in the documents. In this question, we only have two documents. Tfidf does not make sense in this question.

Q3

```
t1 <- texts("Trump Says He’s ‘Not Happy’ With Border Deal, but Doesn’t Say if He Will Sign It.")
t2 <- texts("Trump ‘not happy’ with border deal, weighing options for building wall.")
t1dfm <- dfm(t1,tolower=T,remove_punct = TRUE)
t2dfm <- dfm(t2,tolower=T,remove_punct = TRUE)
# covert to vectors
zeros = c(0,0,0,0,0)
v1 = as.vector(t1dfm)
v2 = append(as.vector(t2dfm),zeros)
```

In pre-processing, I only choose to cover the words to lowercase since there are many capital words in the first sentence which are not important. Some negative words are very important for understanding the sentence. So we can’t simply stem the words. Also, stop words like ‘doesn’t’ are essential, so I choose to keep the stop words.

Q3a,b,c–do calculate

```
calculate_distance <- function(vec1, vec2) {
  euc_dist = sqrt(sum((vec1 - vec2)^2))
  man_dist = sum(abs(vec1 - vec2))
  nominator <- vec1%*% vec2
  denominator <- sqrt(vec1 %*% vec1)*sqrt(vec2 %*% vec2)
  cos_dist = nominator/denominator
  return(list(Euclidean = euc_dist, Manhattan = man_dist,Cosine=cos_dist))
}
calculate_distance(v1,v2)
```

```
## $Euclidean
## [1] 2.236068
##
## $Manhattan
## [1] 5
##
## $Cosine
##           [,1]
## [1,] 0.8291562
```

The Euclidean distance of the two sentences is 2.236068, the Manhattan distance between them is 5, the cosine similarity between them is 0.8291562.

Q4a & Q4b

Get the data (download the novels) and create a data frame.

```
#get the gutenberg_id which we need
gbid <- c()
for (i in c("Austen, Jane","Dickens, Charles","Alcott, Louisa May","Brontë, Charlotte")){
   gid = gutenberg_works(author == i)[1:4,1] #id from 1-4 rows, the first column
   gbid = rbind(gbid,gid)
}
gbid
```

```
## # A tibble: 16 x 1
##    gutenberg_id
##           <int>
##  1          105
##  2          121
##  3          141
##  4          158
##  5           46
##  6           98
##  7          564
##  8          580
##  9          163
## 10          514
## 11         2726
## 12         2786
## 13         1028
## 14         1260
## 15         9182
## 16        30486
```

```
# covert gbid, remove the name of column
book= c()
for (i in gbid){
  book <- rbind(book,i)
}

#download books, extract 500 lines of each text, create the dataframe
books = data.frame()
for (b in book){
  book_ <- gutenberg_download(b,meta_fields = c('title','author'))
  lines <- book_[100:600,]
  books <- rbind(books,lines)
}
```

```
## Determining mirror for Project Gutenberg from http://www.gutenberg.org/robot/harvest
```

```
## Using mirror http://aleph.gutenberg.org
```

```
# remove the special characters that is non-utf-8
df <- data.frame(apply(books, 2, function(x) {x <- gsub("\xa0", "", x)}))
str(df)
```

```
## 'data.frame':    8016 obs. of  4 variables:
##  $ gutenberg_id: Factor w/ 16 levels "   46","   98",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ text        : Factor w/ 6492 levels "",""                                * * * * *",..: 6231 5161
##  $ title       : Factor w/ 16 levels "A Christmas Carol in Prose; Being a Ghost Story of Christmas",
##  $ author      : Factor w/ 4 levels "Alcott, Louisa May",..: 2 2 2 2 2 2 2 2 2 2 ...
```

```
df[1:2,]
```

```
##   gutenberg_id
## 1          105
## 2          105
##                                                                    text
## 1           which he had not been very much tempted to do.  Elizabeth had
## 2 succeeded, at sixteen, to all that was possible, of her mother's rights
##        title       author
## 1 Persuasion Austen, Jane
## 2 Persuasion Austen, Jane
```

Q4c

Use the stylest select vocab function to select the terms in my model.

```
filter <- corpus::text_filter(drop_punct = TRUE, drop_number = TRUE, drop=stopwords_en)
set.seed(2020L)
# fits n-fold cross-validation, I choose n=10
vocab_custom <- stylest_select_vocab(df$text, df$author,
                                     filter = filter, smooth = 1, nfold = 10,
                                     cutoff_pcts = c(10, 20, 30, 40, 50, 60, 70, 80, 90))
# percentile with best prediction rate, the best speaker classification rate
vocab_custom$cutoff_pct_best
```

```
## [1] 10
```

```
# rate of incorrectly predicted speakers of held-out texts,
#matrix of the mean percentage of incorrectly identified speakers for each cutoff percent and fold
vocab_custom$miss_pct
```

```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
##  [1,] 39.52618 39.52618 39.52618 39.52618 39.52618 41.27182 43.51621 44.51372
##  [2,] 44.13965 44.13965 44.13965 44.13965 44.13965 44.63840 44.63840 47.13217
##  [3,] 43.14214 43.14214 43.14214 43.14214 43.14214 45.01247 45.01247 47.13217
##  [4,] 43.07116 43.07116 43.07116 43.07116 43.07116 45.06866 45.69288 47.06617
##  [5,] 41.27182 41.27182 41.27182 41.27182 41.27182 43.26683 44.63840 44.76309
##  [6,] 40.07491 40.07491 40.07491 40.07491 40.07491 43.57054 44.06991 45.94257
##  [7,] 39.57553 39.57553 39.57553 39.57553 39.57553 41.19850 42.32210 44.44444
##  [8,] 42.51870 42.51870 42.51870 42.51870 42.51870 42.14464 43.14214 46.75810
##  [9,] 41.27182 41.27182 41.27182 41.27182 41.27182 43.26683 44.51372 45.51122
## [10,] 43.44569 43.44569 43.44569 43.44569 43.44569 44.44444 45.94257 47.69039
##           [,9]
##  [1,] 47.50623
##  [2,] 50.37406
##  [3,] 50.24938
##  [4,] 50.93633
##  [5,] 48.00499
##  [6,] 50.93633
##  [7,] 48.43945
##  [8,] 49.12718
##  [9,] 46.88279
## [10,] 50.68664
```

For the pre-processing, I choose to drop punctuation which is quite normal. And drop number, I think numeric characters are not important in literature works. Plus, remove the stop words since there are so many irrelevant stop words in novels.

So, 10 percentile of term frequency has the best prediction rate. And the above shows the matrix of the mean percentage of incorrectly identified speakers for each cutoff percent and fold

Q4d

```r
# subset features # USE SAME FILTER, get the terms
vocab_subset <- stylest_terms(df$text,df$author, vocab_custom$cutoff_pct_best , filter = filter)
# fit the model
style_model <- stylest_fit(df$text,df$author, terms = vocab_subset, filter = filter)
# explore output # influential terms, compute the influence of the terms
head(stylest_term_influence(style_model, df$text, df$author))
```

```
##     term    infl_avg    infl_max
## 1     mr 0.035482906 0.090069081
## 2   said 0.005615025 0.011282474
## 3 little 0.010893635 0.033373774
## 4    one 0.002078975 0.003626721
## 5   like 0.005670873 0.009769696
## 6    mrs 0.015539628 0.034009247
```

```r
# report the top 5 terms
authors <- unique(df$author)
term_usage <- style_model$rate
lapply(authors, function(x) head(term_usage[x,][order(-term_usage[x,][1:5])])) %>% setNames(authors)
```

```
## $`Austen, Jane`
##          mr         one      little        said        like
## 0.006582373 0.004127929 0.003830420 0.003458535 0.001822239
##
## $`Dickens, Charles`
##          mr        said         one        like      little
## 0.011539716 0.004770837 0.004444626 0.002242701 0.001998043
##
## $`Alcott, Louisa May`
##       little         said         like          one           mr
## 0.0074285714 0.0059047619 0.0040000000 0.0039238095 0.0002666667
##
## $`Brontë, Charlotte`
##          mr        said        like      little         one
## 0.005229008 0.003854962 0.003473282 0.003396947 0.003320611
```

I think these words don't make sense.

Q4e

```r
# choose: "Austen, Jane","Dickens, Charles"
AJ = term_usage[1,]
DC = term_usage[2,]
sort(AJ,decreasing=TRUE)[1:5]
```

```
##      little        said        like         one     flowers
## 0.007428571 0.005904762 0.004000000 0.003923810 0.003542857
```

```
sort(DC,decreasing=TRUE)[1:5]
```

```
##          mr         sir         mrs        must         one
## 0.006582373 0.006061733 0.005020454 0.004276683 0.004127929
```

I think the ratio shows the words that the specific author would like to use in his/her novel. For instance, Dickens, Charles used the word "mr" more often than Austen, Jane.

Q4f

```
mystery_ex = readRDS("/Users/chenjiesu/Desktop/Text_as_Data_Assignments/mystery_excerpt.rds")
pred = stylest_predict(style_model,mystery_ex)
pred$predicted
```

```
## [1] Brontë, Charlotte
## 4 Levels: Alcott, Louisa May Austen, Jane ... Dickens, Charles
```

According to the model, Bronte Charlotte is the most likely author to the mystery excerpt.

Q4g

From the results below, I think the 10 collocations with the largest lambda value can perform better to be multi-word expressions (lambda is the n-way interaction term from a saturated log-linear model).

```
# bigrams
g1<-toString(df[1:500,2])#Austen, Jane: Persuasion
textstat_collocations(g1, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##       collocation count count_nested length    lambda         z
## 1   kellynch hall    11            0      2 10.190807  6.615570
## 2        mrs clay     5            0      2  9.791404  6.093425
## 3      sir walter    34            0      2  9.565090 10.376344
## 4    lady russell    13            0      2  9.350647  6.333661
## 5     mr shepherd     8            0      2  9.127125  8.386012
## 6  lady russell's     7            0      2  8.273312  5.580280
## 7            i am     5            0      2  7.283054  4.893742
## 8        they are     5            0      2  5.780373  8.676038
## 9         did not     5            0      2  5.706705  7.271226
## 10      any other     7            0      2  5.696508  9.371931
```

```
textstat_collocations(g1, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##       collocation count count_nested length    lambda         z
## 1      sir walter    34            0      2  9.565090 10.376344
## 2          of the    25            0      2  1.329002  5.821480
## 3           to be    22            0      2  2.686310  9.782908
## 4          in the    19            0      2  1.984030  7.340554
## 5        had been    16            0      2  4.021952 11.126688
## 6          he had    13            0      2  3.119070  9.097969
## 7    lady russell    13            0      2  9.350647  6.333661
## 8         she was    12            0      2  3.093401  8.837347
## 9         she had    12            0      2  2.798950  8.163984
## 10 kellynch hall    11            0      2 10.190807  6.615570
```

```
g2<-toString(df[501:1000,2])#Austen, Jane: Northanger Abbey
textstat_collocations(g2, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##        collocation count count_nested length    lambda        z
## 1     miss morland     5            0      2 10.109295 5.989508
## 2          my dear     6            0      2  8.236179 5.501675
## 3  without having     5            0      2  7.314594 8.426271
## 4          said he     5            0      2  7.162166 7.332625
## 5       young lady     5            0      2  6.651041 8.022154
## 6             i am    10            0      2  5.891231 6.679257
## 7           i wish    11            0      2  5.486946 7.734530
## 8        a partner     6            0      2  5.084947 5.573912
## 9        have been     7            0      2  5.065840 9.167753
## 10       they were     9            0      2  4.993150 9.882636
```

```
textstat_collocations(g2, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length    lambda         z
## 1        to be    22            0      2 3.5839844 11.312515
## 2       of the    16            0      2 1.3842775  4.938792
## 3        it is    13            0      2 4.4060400 10.984466
## 4       in the    13            0      2 1.9016516  5.970445
## 5         of a    12            0      2 1.4650306  4.595542
## 6      she had    11            0      2 3.8154724  9.380070
## 7       i wish    11            0      2 5.4869457  7.734530
## 8         i am    10            0      2 5.8912308  6.679257
## 9       of her    10            0      2 1.6588185  4.739767
## 10      to the    10            0      2 0.9699149  2.878026
```

```
g3<-toString(df[1001:1500,2])#Austen, Jane: Mansfield Park
textstat_collocations(g3, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##     collocation count count_nested length    lambda        z
## 1    sir thomas    16            0      2 11.448654 6.932206
## 2 lady bertram     8            0      2  9.688388 8.042401
## 3    great deal     6            0      2  7.810218 8.864364
## 4          i am     9            0      2  7.311755 5.016777
## 5       my dear     6            0      2  6.393238 8.198243
## 6       used to     5            0      2  5.585125 3.775836
## 7     the world     5            0      2  5.566850 3.763569
## 8      at least     5             0      2  5.441477 8.123234
## 9       you are     9             0      2  5.218256 9.703512
## 10      did not     7             0      2  5.110451 8.297657
```

```
textstat_collocations(g3, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length     lambda         z
## 1       of her    28            0      2  1.7692447  7.933199
## 2       in the    24            0      2  2.4818362  9.677401
## 3        to be    19            0      2  2.4018454  8.499476
## 4   sir thomas    16            0      2 11.4486536  6.932206
## 5       of the    15            0      2  0.9102902  3.269665
## 6      she had    14            0      2  3.6448413 10.135423
## 7       to the    13            0      2  0.7074067  2.402834
## 8     would be    11            0      2  4.2034106  9.908631
## 9      and the    11            0      2  0.4199131  1.335620
## 10       it is    10            0      2  4.2954499 10.079675
```

```
g4<-toString(df[1501:2000,2])#Austen, Jane: Emma
textstat_collocations(g4, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1  miss taylor    18            0      2 9.172963 6.301774
## 2      my dear     5            0      2 6.803505 8.886790
## 3   every body     7            0      2 6.662885 8.615424
## 4         i am     8            0      2 6.089192 6.791049
## 5    enough to     7            0      2 6.029725 4.121204
## 6       do not     5            0      2 5.303395 7.503686
## 7    poor miss     7            0      2 5.288242 9.205918
## 8     you know     5            0      2 5.182857 7.359534
## 9   her father     7            0      2 5.116481 7.724677
## 10     said mr     6            0      2 4.916511 8.409037
```

```
textstat_collocations(g4, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda         z
## 1       of the    21            0      2 1.645267  6.563113
## 2       it was    20            0      2 3.941992 12.365761
## 3  miss taylor    18            0      2 9.172963  6.301774
## 4      he had    17            0      2 3.986063 11.855397
## 5        to be    15            0      2 2.649798  8.188032
## 6       in the    15            0      2 2.212685  7.332701
## 7       of her    15            0      2 2.055670  6.817212
## 8        was a    14            0      2 1.855151  6.137176
## 9         of a    14            0      2 1.097146  3.783666
## 10      a very    13            0      2 2.981033  8.243411
```

```
#Dickens, Charles: A Christmas Carol in Prose; Being a Ghost Story of Christmas
g5<-toString(df[2001:2500,2])
textstat_collocations(g5, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##         collocation count count_nested length   lambda         z
## 1    good afternoon     6            0      2 7.470984  7.567174
## 2             as if     6            0      2 7.322173  4.938098
## 3   merry christmas     7            0      2 6.530053  8.288696
## 4      returned the     6            0      2 5.350684  3.640037
## 5      said scrooge    16            0      2 5.216085 11.462667
## 6     the gentleman     6            0      2 4.114968  4.525910
## 7           was not     6            0      2 3.754809  7.050846
## 8         the clerk     7            0      2 3.752585  5.108974
## 9            a time     5            0      2 3.698741  5.725482
## 10           it was    13            0      2 3.590081  9.624842
```

```
textstat_collocations(g5, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##         collocation count count_nested length   lambda         z
## 1            in the    23            0      2 2.246782  8.329408
## 2            of the    20            0      2 1.774746  6.573600
## 3      said scrooge    16            0      2 5.216085 11.462667
## 4            it was    13            0      2 3.590081  9.624842
## 5             to be     9            0      2 3.273498  7.333339
## 6           that it     8            0      2 2.853261  6.831949
## 7              in a     8            0      2 1.656290  4.284379
## 8   merry christmas     7            0      2 6.530053  8.288696
## 9          the door     7            0      2 3.415707  5.238117
## 10        the clerk     7            0      2 3.752585  5.108974
```

```
g6<-toString(df[2501:3000,2])#Dickens, Charles: A Tale of Two Cities
textstat_collocations(g6, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##     collocation count count_nested length   lambda         z
## 1     dover mail     6            0      2 6.302471  8.683258
## 2          as if     6            0      2 4.916548  7.730045
## 3          it is     6            0      2 4.606811  7.925819
## 4      the guard    18            0      2 4.264844  6.305187
## 5      they were     5            0      2 3.988754  7.404072
## 6      the dover     8            0      2 3.967378  4.470205
## 7         it was    12            0      2 3.840880 10.003065
## 8      the coach    11            0      2 3.767681  5.380451
## 9         out of     8            0      2 3.546447  6.403845
## 10        he had     5            0      2 3.425038  6.667414
```

```
textstat_collocations(g6, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##     collocation count count_nested length    lambda         z
## 1         of the    49            0      2 1.7666377  9.543104
## 2         in the    26            0      2 1.6203944  6.694067
## 3        and the    25            0      2 0.7299606  3.243697
## 4         to the    20            0      2 0.8894542  3.527858
## 5      the guard    18            0      2 4.2648444  6.305187
## 6         on the    15            0      2 2.2255078  6.359326
## 7         it was    12            0      2 3.8408800 10.003065
## 8      the coach    11            0      2 3.7676807  5.380451
## 9     from the    10            0      2 2.5992492  5.583360
## 10    the mail    10            0      2 2.5733952  5.343167
```

```r
g7<-toString(df[3001:3500,2])#Dickens, Charles: The Mystery of Edwin Drood
textstat_collocations(g7, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##     collocation count count_nested length   lambda        z
## 1  young fellow     6            0      2 7.052872 8.667448
## 2          i am     5            0      2 6.248770 4.213614
## 3      has been     5            0      2 5.020773 8.043962
## 4      a little     8            0      2 4.732846 6.499321
## 5      the dean    13            0      2 4.519991 6.518051
## 6       you are     6            0      2 3.978965 7.299049
## 7      you know     5            0      2 3.912434 6.707749
## 8         it is    10            0      2 3.756180 9.042435
## 9         as he     6            0      2 3.716635 7.395655
## 10    the young     5            0      2 3.235031 4.698692
```

```r
textstat_collocations(g7, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1       with a    14            0      2 2.873287 8.439314
## 2       on the    14            0      2 2.572348 7.271553
## 3     the dean    13            0      2 4.519991 6.518051
## 4       in the    11            0      2 1.469121 4.327221
## 5       to the    11            0      2 1.292158 3.862566
## 6        it is    10            0      2 3.756180 9.042435
## 7       of the    10            0      2 1.012214 2.955894
## 8     a little     8            0      2 4.732846 6.499321
## 9       at the     8            0      2 3.007459 5.919879
## 10        in a     8            0      2 1.748006 4.504913
```

```r
g8<-toString(df[3501:4000,2])#Dickens, Charles: The Pickwick Papers
textstat_collocations(g8, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##         collocation count count_nested length   lambda         z
## 1  this association     7            0      2 8.536222  5.704896
## 2     samuel weller     7            0      2 8.437686  8.096619
## 3        account of     6            0      2 5.467284  3.719923
## 4          had been     8            0      2 5.230485  9.003194
## 5     pickwick club     7            0      2 5.226713  8.735288
## 6           said mr     9            0      2 4.669127  8.429435
## 7            he had    13            0      2 4.415727 10.749138
## 8         the first     5            0      2 3.706649  4.014385
## 9           and how     9            0      2 3.706210  6.823586
## 10            to be     7            0      2 3.509080  6.985594
```

```r
textstat_collocations(g8, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda          z
## 1       of the    52            0      2 1.877711 10.2869594
## 2        to the    22              0       2 1.516047  5.9138863
## 3        in the    14              0       2 1.231645  4.0278941
## 4       and the    14              0       2 0.266749  0.9360299
## 5        he had    13              0       2 4.415727 10.7491378
## 6        of his    10              0       2 1.355364  3.9034808
## 7       said mr     9              0       2 4.669127  8.4294350
## 8       and how     9              0       2 3.706210  6.8235863
## 9        with a     9              0       2 2.599738  6.6339362
## 10       on the     9              0       2 2.028722  4.8020641
```

```
g9<-toString(df[4001:4500,2])#Alcott, Louisa May: Flower Fables
textstat_collocations(g9, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##           collocation count count_nested length   lambda         z
## 1           at length     5            0      2 8.607290  5.656549
## 2             go back     5            0      2 6.660575  8.590651
## 3        golden light     8            0      2 6.506227  9.860453
## 4        gentle words     5            0      2 5.932490  8.131144
## 5        green leaves     6            0      2 5.726868  9.257542
## 6           among the     9            0      2 5.572502  3.836699
## 7            she went     6            0      2 4.958255  7.810400
## 8  the frost-spirits     5            0      2 4.882947  3.302920
## 9             back to     7            0      2 4.749614  6.438475
## 10      little violet    10            0      2 4.614141 10.260989
```

```
textstat_collocations(g9, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##       collocation count count_nested length    lambda          z
## 1          to the    20            0      2 1.0851093  4.3064545
## 2          in the    19            0      2 1.7863015  6.4282400
## 3         and the    19            0      2 0.1539401  0.6275987
## 4          of the    14            0      2 1.2192290  4.0613646
## 5          on the    12            0      2 2.1396071  5.8648336
## 6        from the    11            0      2 3.0614787  6.4601157
## 7  little violet    10            0      2 4.6141409 10.2609893
## 8     through the     9            0      2 3.6257297  5.7671924
## 9      the golden     9            0      2 3.8334836  5.3727964
## 10     the gentle     9            0      2 2.3960765  5.3008564
```

```
g10<-toString(df[4501:5000,2])#Alcott, Louisa May: Little Women
textstat_collocations(g10, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1      her own     5            0      2 6.271536 4.232216
## 2       no one     5            0      2 5.922425 8.646276
## 3      i shall     6            0      2 4.947506 6.555223
## 4     said meg     8            0      2 4.636534 9.493670
## 5       i know     5            0      2 4.428308 6.371264
## 6    said beth     6            0      2 4.356567 8.282341
## 7      you can     5            0      2 4.281095 6.903735
## 8      who was     6            0      2 4.169556 7.665423
## 9      i don't     7            0      2 4.149575 7.535792
## 10     you are     7            0      2 3.985308 7.964846
```

```
textstat_collocations(g10, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length    lambda         z
## 1        to be    18            0      2 3.7527232 10.572083
## 2       with a    18            0      2 3.0577337  9.798129
## 3      and the    17            0      2 0.6216656  2.366915
## 4       of the    16            0      2 1.7209246  5.916971
## 5       in the    14            0      2 1.9512797  6.144686
## 6       at the    10            0      2 2.4010782  6.037543
## 7        was a    10            0      2 2.0771878  5.793718
## 8       it was     9            0      2 3.8993794  8.977369
## 9       in her     9            0      2 2.3246175  6.165687
## 10    said meg     8            0      2 4.6365336  9.493670
```

```
g11<-toString(df[5001:5500,2])#Alcott, Louisa May: Eight Cousins
textstat_collocations(g11, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda         z
## 1      i don't     6            0      2 5.510056  6.012342
## 2        as if     7            0      2 5.497179  8.130687
## 3       i hope     5            0      2 5.324134  5.716255
## 4      did you     6            0      2 4.976913  7.291704
## 5      i shall     5            0      2 4.812974  6.209158
## 6        he is     5            0      2 4.535401  7.852149
## 7    said rose     5            0      2 4.337524  7.449619
## 8       it was    11            0      2 4.239122 10.053677
## 9       as she     9            0      2 3.818807  8.863490
## 10     she had     7            0      2 3.690384  7.664421
```

```
textstat_collocations(g11, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length    lambda         z
## 1      and the    13            0      2 0.5700873  1.919106
## 2       with a    12            0      2 2.8710802  7.907593
## 3        and i    12            0      2 2.2339342  6.288182
## 4       it was    11            0      2 4.2391216 10.053677
## 5       in the    11            0      2 2.1694943  5.970030
## 6       to the    10            0      2 0.8915389  2.635935
## 7       as she     9            0      2 3.8188067  8.863490
## 8         in a     8            0      2 2.2904290  5.663026
## 9        was a     8            0      2 2.2904290  5.663026
## 10      of the     8            0      2 1.0433025  2.768972
```

```
g12<-toString(df[5501:6000,2])#Alcott, Louisa May: Jack and Jill
textstat_collocations(g12, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda         z
## 1      i guess     5            0      2 6.820132  4.592260
## 2     had been     5            0      2 6.198108  6.582930
## 3        do it     6            0      2 4.999353  7.852468
## 4        it is    10            0      2 4.857555 10.192665
## 5      who had     9             0       2 4.833826  9.867022
## 6       out of     6             0       2 4.403372  6.546544
## 7        as if     7             0       2 4.048974  7.948353
## 8      as they     5             0       2 3.698409  6.673098
## 9     into the     5             0       2 3.526699  4.604213
## 10      like a     6             0       2 3.339943  6.130065
```

```
textstat_collocations(g12, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length      lambda          z
## 1       in the    25            0      2  2.68092748  9.5700372
## 2       on the    17            0      2  3.21216767  8.3142107
## 3       of the    16            0      2  1.33204640  4.7053619
## 4       with a    14            0      2  2.84452615  8.4505081
## 5       at the    12            0      2  2.94292574  6.9310231
## 6     with the    11            0      2  1.71836000  4.9045187
## 7        it is    10            0      2  4.85755475 10.1926653
## 8      for the    10            0      2  1.28229039  3.6678323
## 9      and the    10            0      2 -0.02007263 -0.0615227
## 10     who had     9            0      2  4.83382606  9.8670220
```

```
g13<-toString(df[6001:6500,2])#Brontë, Charlotte: The Professor
textstat_collocations(g13, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1      he said     5            0       2 5.328650 6.837197
## 2     did not     7            0       2 5.310942 8.519485
## 3    my uncles     5            0       2 5.296328 5.699354
## 4      you are     6            0       2 5.020518 8.501077
## 5     know not     5            0       2 4.425639 7.521738
## 6        i saw     5            0       2 4.391583 4.749620
## 7      to take     6            0       2 4.366562 5.822132
## 8       when i     7            0       2 3.699269 7.356640
## 9       it was     9            0       2 3.586988 8.525486
## 10     i could     5            0       2 3.543727 5.147385
```

```
textstat_collocations(g13, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length    lambda        z
## 1       in the    28            0       2 2.2722081 9.450907
## 2       of the    24            0       2 1.4351427 6.090245
## 3        i had    16            0       2 2.9902974 8.676845
## 4       to the    15            0       2 0.9724464 3.452475
## 5        of my    13            0       2 2.2163743 6.796140
## 6        i was    12            0       2 1.8099577 5.501946
## 7       on the    11            0       2 3.1349296 7.047652
## 8        to me    11            0       2 2.1596158 6.232567
## 9       with a    10            0       2 2.5353586 6.847449
## 10      it was     9            0       2 3.5869878 8.525486
```

```r
g14<-toString(df[6501:7000,2])#Brontë, Charlotte: Jane Eyre: An Autobiography
textstat_collocations(g14, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1   miss abbot     7            0      2 8.855806 8.083248
## 2      you are     8            0      2 7.537582 5.140970
## 3    could not     6            0      2 5.135942 8.598688
## 4   would have     5            0      2 4.556280 7.977474
## 5    because i     5            0      2 4.427288 6.798099
## 6        to be     9            0      2 3.841845 7.410211
## 7      i never     5            0      2 3.763296 5.455056
## 8      the bed     5            0      2 3.576739 4.668265
## 9       like a     8            0      2 3.429066 7.239547
## 10      it was     9            0      2 3.061365 7.637147
```

```r
textstat_collocations(g14, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1       of the    23            0      2 1.553967 6.391921
## 2        i was    19            0      2 2.528235 8.841587
## 3       in the    16            0      2 1.684369 5.823669
## 4       to the    16            0      2 1.157897 4.175477
## 5        i had    12            0      2 2.437100 7.016979
## 6       by the    10            0      2 2.694631 6.435080
## 7       it was     9            0      2 3.061365 7.637147
## 8        to be     9            0      2 3.841845 7.410211
## 9       like a     8            0      2 3.429066 7.239547
## 10      with a     8            0      2 2.537355 6.184434
```

```r
g15<-toString(df[7001:7500,2])#Brontë, Charlotte: Villette
textstat_collocations(g15, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda        z
## 1  little girl     6            0      2 7.646035 5.148586
## 2      i found     5            0      2 5.876935 3.968795
## 3      you are     5            0      2 5.869224 7.374544
## 4       in its     5            0      2 4.059637 6.562753
## 5    which she     5            0      2 3.719289 6.760173
## 6     into the     6            0      2 3.692496 5.522514
## 7    the child     7            0      2 3.469714 5.764312
## 8     her hand     5            0      2 3.320991 5.640263
## 9       it was    10            0      2 3.131164 7.984879
## 10      i have     6            0      2 3.107247 5.978442
```

```r
textstat_collocations(g15, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length    lambda        z
## 1     from the    14            0      2 2.9398084 8.029563
## 2       of the    14            0      2 1.7019411 5.559563
## 3       in the    13            0      2 1.8958897 5.880726
## 4      she was    12            0      2 2.3919810 7.073662
## 5        it was    10            0      2 3.1311640 7.984879
## 6        to the     9            0      2 0.7563777 2.153163
## 7          in a     8            0      2 1.7827541 4.611112
## 8       was not     7            0      2 2.5328607 5.859372
## 9     the child     7            0      2 3.4697138 5.764312
## 10       on the     7            0      2 1.7373245 4.141471
```

```
g16<-toString(df[7501:8000,2])#Brontë, Charlotte: Shirley
textstat_collocations(g16, size = 2, min_count = 5) %>% arrange(-lambda) %>% slice(1:10)
```

```
##    collocation count count_nested length   lambda         z
## 1    there was     5            0      2 4.467198  7.545182
## 2        it is    13            0      2 4.060107 10.625564
## 3     they are     7            0      2 3.920830  7.944400
## 4    they were     7            0      2 3.837151  7.888309
## 5       i have     7            0      2 3.822517  8.074974
## 6      what is     5            0      2 3.761470  6.916782
## 7     a little     5            0      2 3.589989  5.584115
## 8       is not     5            0      2 3.517192  6.709462
## 9  the curates     6            0      2 3.248099  5.256077
## 10      like a     5            0      2 3.069024  5.388871
```

```
textstat_collocations(g16, size = 2, min_count = 5) %>% arrange(-count) %>% slice(1:10)
```

```
##    collocation count count_nested length    lambda         z
## 1       in the    19            0      2 1.9956926  7.221378
## 2       of the    15            0      2 0.8434918  3.005408
## 3         of a    14            0      2 1.2194202  4.145036
## 4        it is    13            0      2 4.0601066 10.625564
## 5       of his    11            0      2 1.9482736  5.623797
## 6         in a    10            0      2 1.6078692  4.630024
## 7       to the    10            0      2 0.8365762  2.487985
## 8        to be     9            0      2 3.0112449  7.218753
## 9       i have     7            0      2 3.8225168  8.074974
## 10    they are     7            0      2 3.9208297  7.944400
```

Q5a

```
# make the snippets of one sentence, between 150-350 chars in length
# data(data_corpus_ukmanifestos, package = "quanteda.corpora")
snippetData <- snippets_make(data_corpus_ukmanifestos, nsentence = 1, minchar = 150, maxchar = 350)
```

```
## Warning: '[[<-.corpus' is deprecated.
## Use 'docvars' instead.
## See help("Deprecated")
```

```
# clean up the snippets and get the top 10
snippetData <- snippets_clean(snippetData)
```

```
## Cleaning 14,501 snippets...
##    removed 442 snippets containing numbers of at least 1,000

##    removed 5 snippets containing long elipses ....

##    removed 1,651 snippets containing ALL CAPS titles

##    ...finished.
```

```r
head(snippetData,10)
```

```
##                   docID snippetID
## 1  UK_natl_1945_en_Con    100002
## 2  UK_natl_1945_en_Con    100009
## 3  UK_natl_1945_en_Con    100010
## 4  UK_natl_1945_en_Con    100015
## 5  UK_natl_1945_en_Con    100016
## 6  UK_natl_1945_en_Con    100019
## 7  UK_natl_1945_en_Con    100020
## 8  UK_natl_1945_en_Con    100031
## 9  UK_natl_1945_en_Con    100032
## 10 UK_natl_1945_en_Con    100045
##
## 1
## 2                                                                                             It will
## 3                                                                                                  Ha
## 4
## 5                                               We have, during the years of our history, gained the
## 6
## 7
## 8  The prowess of the Indian Army must not be overlooked in the framing of plans for granting India
## 9
## 10
```

Q5b

```r
set.seed(2020)
# generate more pairs from a larger sample of data
snippetPairsAll <- pairs_regular_make(snippetData[sample(1:nrow(snippetData), 1000), ])
# Make some "Gold" questions -- for use with CrowdFlower workers
gold_questions <- pairs_gold_make(snippetPairsAll, n.pairs = 10)
```

```
## Starting the creation of gold questions...

##    computing Flesch readability measure

##    selecting top different 10 pairs

##    applying min.diff.quantile thresholds of 3.13, 40.74
##    creating gold_reason text

##    ...finished.
```

```
gold_questions
```

```
##                 docID1 snippetID1
## 1   UK_natl_2001_en_Lab    5900275
## 2   UK_natl_1945_en_Lab     200036
## 3    UK_natl_2005_en_FW    8000067
## 4  UK_natl_2005_en_SDLP    9200363
## 5  UK_natl_2005_en_UKIP    9800231
## 6   UK_natl_2005_en_EIP    7800140
## 7   UK_natl_1987_en_Lab    3400099
## 8   UK_natl_1974_en_Lab    2500191
## 9   UK_natl_2005_en_SSP    9600652
## 10  UK_natl_1983_en_Lab    3100506
##
## 1
## 2                                                         What is true is that the anti-controllers
## 3
## 4                                                              Sustainable Development · Urgently i
## 5
## 6                                                                     5.3  Mixed employment opportu
## 7
## 8
## 9
## 10 We shall increase the number of disablement resettlement officers; extend capital grants to adapt
##                 docID2 snippetID2
## 1    UK_natl_2005_en_FW    8000010
## 2    UK_natl_2001_en_SF    6500415
## 3   UK_natl_2001_en_PCy    6300187
## 4   UK_natl_1979_en_Con    2702960
## 5   UK_natl_2001_en_Lab    5900275
## 6   UK_natl_2005_en_DUP    7600094
## 7   UK_natl_2005_en_PVP    8900073
## 8    UK_natl_2005_en_Gr    8100073
## 9  UK_natl_2005_en_SDLP    9200363
## 10  UK_natl_1974_en_Lab    2500128
##
## 1
## 2
## 3     Without the greater capacity to examine public objectives that a truly democratic Economic and
## 4
## 5                                                                                                  I
## 6
## 7  Giving you the right to vote "None of the Above" Protest Vote Party Manifesto 7 of 13  www.Protest
## 8
## 9                               Sustainable Development · Urgently implement a Sustainable Developme
## 10
##        read1     read2  readdiff _golden easier_gold
## 1  -28.34800  41.64742 -69.99542    TRUE           2
## 2   27.61020 -27.99429  55.60449    TRUE           1
## 3   47.94231 -14.57239  62.51470    TRUE           1
## 4  -34.19853  21.28500 -55.48353    TRUE           2
## 5   46.10500 -28.34800  74.45300    TRUE           1
## 6   17.59930  75.29655 -57.69725    TRUE           2
## 7   57.55786   3.01500  54.54286    TRUE           1
## 8   44.69667 -11.88000  56.57667    TRUE           1
## 9   46.66500 -34.19853  80.86353    TRUE           1
## 10 -20.48409  49.04138 -69.52547    TRUE           2
##
## 1  Text B is "easier" to read because it contains some combination of shorter sentences, more common
## 2  Text A is "easier" to read because it contains some combination of shorter sentences, more common
## 3  Text A is "easier" to read because it contains some combination of shorter sentences, more common
## 4  Text B is "easier" to read because it contains some combination of shorter sentences, more common
## 5  Text A is "easier" to read because it contains some combination of shorter sentences, more common
## 6  Text B is "easier" to read because it contains some combination of shorter sentences, more common
## 7  Text A is "easier" to read because it contains some combination of shorter sentences, more common
## 8  Text A is "easier" to read because it contains some combination of shorter sentences, more common
## 9  Text A is "easier" to read because it contains some combination of shorter sentences, more common
## 10 Text B is "easier" to read because it contains some combination of shorter sentences, more common
```

pairs_regular_make: create test questions for CrowdFlower from sample of snippet pairs, where the readability is the most different between pairs.

My classification: **B, A, A, B, A, B, B, A, A, B**

Machine Classific: **B, A, A, B, A, B, A, A, A, B**

So, 90% of the ten gold pairs we reach the agreement. For the number 7 pairs, there is an url link in the text B, so I think the text A is easier to understand. But machine might think there are more common words in text B.

Q6

```r
# download the texts
T1232 = gutenberg_download(1232)$text
L3207= gutenberg_download(3207)$text
# Zipf's law as a feature selection tool
dfm1232<- dfm(T1232, tolower=TRUE, remove_punct = TRUE, remove=stopwords("english")) # pre-processing
plot(log10(1:100), log10(topfeatures(dfm1232, 100)),
     xlab = "log10(rank)", ylab = "log10(frequency)", main = "Zipf's Law",col = "red")
# Fits a linear regression
regression <- lm(log10(topfeatures(dfm1232, 100)) ~ log10(1:100))
abline(regression, col = "red")
confint(regression)
```

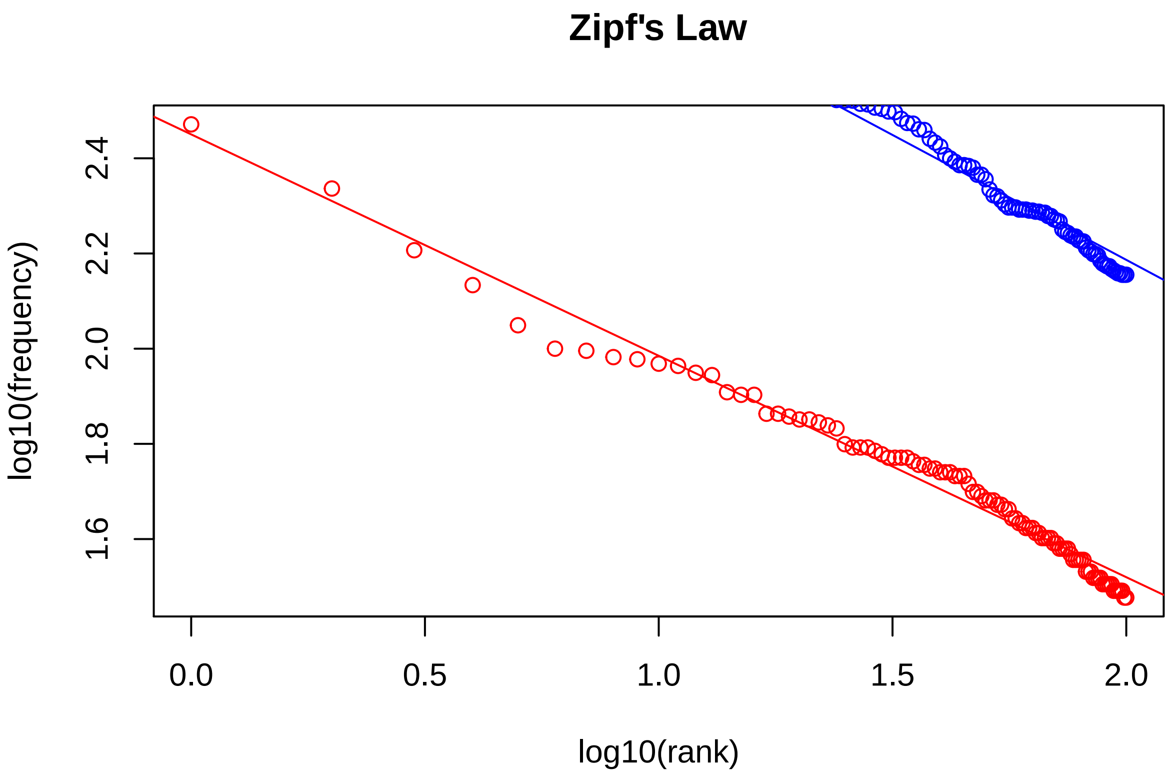
```
##                    2.5 %     97.5 %
## (Intercept)   2.4274431  2.4730838
## log10(1:100) -0.4791861 -0.4511823
```

```r
summary(regression)
```

```
##
```

```
## Call:
## lm(formula = log10(topfeatures(dfm1232, 100)) ~ log10(1:100))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.088280 -0.022026  0.002352  0.022051  0.051179
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.450263   0.011499  213.08   <2e-16 ***
## log10(1:100) -0.465184   0.007056  -65.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0283 on 98 degrees of freedom
## Multiple R-squared:  0.978,  Adjusted R-squared:  0.9777
## F-statistic:  4347 on 1 and 98 DF,  p-value: < 2.2e-16
```

```r
# for leviathan
dfm3207 <- dfm(L3207, tolower=TRUE,remove_punct = TRUE,remove=stopwords("english"))
points(log10(1:100), log10(topfeatures(dfm3207, 100)),
     xlab = "log10(rank)", ylab = "log10(frequency)", main = "Zipf's Law",col="blue")
# regression
regression_ <- lm(log10(topfeatures(dfm3207, 100)) ~ log10(1:100))
abline(regression_, col = "blue")
```



```r
confint(regression_)
```

```
##                    2.5 %     97.5 %
## (Intercept)   3.2097435  3.2630453
## log10(1:100) -0.5412665 -0.5085621
```

```r
summary(regression_)
```

```
##
## Call:
## lm(formula = log10(topfeatures(dfm3207, 100)) ~ log10(1:100))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.207823 -0.015973  0.001437  0.018635  0.050610
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.23639    0.01343   241.0   <2e-16 ***
## log10(1:100) -0.52491    0.00824   -63.7   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03305 on 98 degrees of freedom
## Multiple R-squared:  0.9764, Adjusted R-squared:  0.9762
## F-statistic:  4058 on 1 and 98 DF,  p-value: < 2.2e-16
```

For the pre-processing, I converted all the words to lowercase, removed punctuation and stop words. From the graph, we can know the difference between two novels. And also, the graph of log(corpus frequency) function is almost linear which validate the zipf's law.

Q7

```r
k <- 44
M1232<- nfeat(dfm1232) # M: total number of types(features)
M3207<- nfeat(dfm3207)
TT1232<- sum(ntoken(dfm1232)) # T: get the number of tokens (the text size)
TT3207<- sum(ntoken(dfm3207))
#caculate b
# M = k*(T)^b => logM = b*log(k*(T)) => b = logM / log(k*(T))
b1232 <- log(M1232/k,TT1232)
b3207 <- log(M3207/k,TT3207)

#report the result
print (paste0("The value of b in Machiavelli's "The Prince" is ",b1232))
```

```
## [1] "The value of b in Machiavelli's "The Prince" is 0.475578684701218"
```

```r
print (paste0("The value of b in Hobbes's "Leviathan" is ",b3207))
```

```
## [1] "The value of b in Hobbes's "Leviathan" is 0.47207615642333"
```

As I used the dfm from Q6, I did the same preprocessing as Q6. The value of b in Machiavelli's "The Prince" is 0.475578684701218. And the value of b in Hobbes's "Leviathan" is 0.47207615642333.

Q8

```r
# use kwic() to get the info of key words
# choose politic*, environment, events, administration, government, public, affair, power
#all the words are political words first come to my mind .
kw_1232 <- kwic(T1232, pattern = c('politic*','environment', 'events', 'administration',
                                   'government', 'public', 'affair','power'), window = 7)
kw_3207 <- kwic(L3207, pattern = c('politic*','environment', 'events', 'administration',
                                   'government', 'public', 'affair', 'power'), window = 7)
head(kw_1232,10)
```

```
##
##   [text26, 2]                                      Italian |    power
##   [text28, 4]                       Machiavelli entered the |   public
##   [text29, 6]                   Florence was free under the | government
##   [text30, 9]              1512, when the Medici returned to |    power
##  [text43, 12]                is so well known that the early | environment
##  [text48, 12]          although at one time he wielded immense |    power
##   [text85, 8]               serving four years in one of the |   public
##  [text87, 14]             on firm ground when dealing with the |   events
##  [text111, 2]                                Machiavelli's |   public
##  [text111, 8] Machiavelli's public life was largely occupied with |   events
##
##  | under the guidance of Lorenzo de'
##  | service. During his official career
##  | of a Republic, which lasted
##  | , and Machiavelli lost his
##  | of
##  | over the
##  | offices he was appointed
##  | of
##  | life was largely occupied with events arising
##  | arising out
```

```r
head(kw_3207,10)
```

```
##
##    [text8, 9]                   OR THE MATTER, FORME,& |   POWER    |
##   [text103, 3]                              the Civill |   Power    |
##  [text103, 11]               , should not be by the Civill |   Power    |
##  [text104, 12] reprehending it, declare they think that |   Power    |
##   [text106, 3]                                 Seat of |   Power    |
##   [text114, 6]            whence they impugne the Civill |   Power    |
##   [text160, 4]                                  10. OF |   POWER    |
##   [text187, 4]                         TO THE SOVERAIGN |   POWER    |
##   [text193, 7]                22. OF SYSTEMES SUBJECT, | POLITICALL |
##   [text195, 9] . OF THE PUBLIQUE MINISTERS OF SOVERAIGN |   POWER    |
##
##  OF A COMMON-WEALTH
##  , should not be by the Civill
##  condemned; nor
##  too
##   ,( like to those simple and
##   . If notwithstanding this, you find
##   , WORTH, DIGNITY, HONOUR,
##
##   , AND PRIVATE
##
```

```r
nrow(kw_1232)
```

```
## [1] 86
```

```r
nrow(kw_3207)
```

```
## [1] 958
```

For the words I selected, there are 958 rows of data about those words in The Prince, only 86 rows in Leviathan. In both works, major are 'power' and 'government'. That's very interesting.

```r
# how about war related words?
kw_1232_1 <- kwic(T1232, pattern = c('princedom','kingdom', 'conquer','fortune',
                                     'arms','military','war'), window = 7)
kw_3207_1 <- kwic(L3207, pattern = c('princedom','kingdom', 'conquer','fortune',
                                     'arms','military','war'), window = 7)
nrow(kw_1232_1)
```

```
## [1] 219
```

```r
nrow(kw_3207_1)
```

```
## [1] 105
```

I briefly choose the words from The Prince, which I think those might be more common in the book. There are 219 rows in The Prince and 105 rows in Leviathan. So I guess there might be more contents about war in The Prince.

```r
# words about religion
kw_1232_2 <- kwic(T1232, pattern = c('wealth', 'right','christ', 'authority',
                                     'right', 'church', 'religon*'), window = 7)
kw_3207_2 <- kwic(L3207, pattern = c('wealth', 'right','christ', 'authority',
                                     'right', 'church', 'religon*'), window = 7)
nrow(kw_1232_2)
```

```
## [1] 50
```

```r
nrow(kw_3207_2)
```

```
## [1] 1220
```

I selected the words about religions. From the size of key words data frame, we can know there are much more information about religions in Leviathan.

Q9a

```
head(docvars(data_corpus_ukmanifestos), 10)
```

```
##                      Country Type Year Language Party
## UK_natl_1945_en_Con      UK natl 1945       en   Con
## UK_natl_1945_en_Lab      UK natl 1945       en   Lab
## UK_natl_1945_en_Lib      UK natl 1945       en   Lib
## UK_natl_1950_en_Con      UK natl 1950       en   Con
## UK_natl_1950_en_Lab      UK natl 1950       en   Lab
## UK_natl_1950_en_Lib      UK natl 1950       en   Lib
## UK_natl_1951_en_Con      UK natl 1951       en   Con
## UK_natl_1951_en_Lab      UK natl 1951       en   Lab
## UK_natl_1951_en_Lib      UK natl 1951       en   Lib
## UK_natl_1955_en_Con      UK natl 1955       en   Con
```

```
CorpSub <- corpus_subset(data_corpus_ukmanifestos, Party=='Lab')
sentences = corpus_reshape(CorpSub, to = "sentences")
head(docvars(CorpSub), 10) # see the selected data
```

```
##                      Country Type Year Language Party
## UK_natl_1945_en_Lab      UK natl 1945       en   Lab
## UK_natl_1950_en_Lab      UK natl 1950       en   Lab
## UK_natl_1951_en_Lab      UK natl 1951       en   Lab
## UK_natl_1955_en_Lab      UK natl 1955       en   Lab
## UK_natl_1959_en_Lab      UK natl 1959       en   Lab
## UK_natl_1964_en_Lab      UK natl 1964       en   Lab
## UK_natl_1966_en_Lab      UK natl 1966       en   Lab
## UK_natl_1970_en_Lab      UK natl 1970       en   Lab
## UK_natl_1974_en_Lab      UK natl 1974       en   Lab
## UK_natl_1979_en_Lab      UK natl 1979       en   Lab
```

```
# convert corpus to df
uklabdf <- sentences$documents %>% select(texts, Party, Year) %>% mutate(Year = as.integer(Year))
# Let's filter out any NAs
uklabdf <- na.omit(uklabdf)
# mean Flesch statistic per year
flesch_point <- uklabdf$texts %>% textstat_readability(measure = "Flesch") %>%
  group_by(uklabdf$Year) %>%
  summarise(mean_flesch = mean(Flesch)) %>%
  setNames(c("Year", "mean")) %>% arrange(Year)
flesch_point
```

```
## # A tibble: 16 x 2
##     Year  mean
##    <int> <dbl>
##  1  1945  55.2
##  2  1950  50.2
##  3  1951  59.5
##  4  1955  49.8
##  5  1959  48.6
## 6  1964  40.6
## 7  1966  50.2
## 8  1970  44.3
## 9  1974  42.2
## 10 1979  42.7
## 11 1983  41.5
## 12 1987  40.5
## 13 1992  42.2
## 14 1997  44.1
## 15 2001  48.2
## 16 2005  38.8
```

```
# We will use a loop to bootstrap a sample of texts and subsequently calculate standard errors
iters <- 10

library(pbapply)
Years = unique(uklabdf$Year)
# build function to be used in bootstrapping
boot_flesch <- function(Year_data){
  N <- nrow(Year_data)
  bootstrap_sample <- sample_n(Year_data, N, replace = TRUE)
  readability_results <- textstat_readability(bootstrap_sample$texts, measure = "Flesch")
  return(mean(readability_results$Flesch))
}

# apply function to each year
boot_flesch_by_Year <- pblapply(Years, function(x){
  sub_data <- uklabdf %>% filter(Year == x)
  output_flesch <- lapply(1:iters, function(i) boot_flesch(sub_data))
  return(unlist(output_flesch))
})
names(boot_flesch_by_Year) <- Years

# compute mean and std.errors
Year_means <- lapply(boot_flesch_by_Year, mean) %>% unname() %>% unlist()
Year_ses <- lapply(boot_flesch_by_Year, sd) %>% unname() %>% unlist()
# bootstrap standard error = sample standard deviation bootstrap distribution
Year_ses
```

```
##  [1] 1.4833240 1.7266025 1.8453322 2.4509111 1.1650446 1.5175465 1.3112333
##  [8] 0.9825483 1.9587855 0.3886173 0.9682493 1.2860997 1.2088081 0.6884571
## [15] 0.7489051 1.0112467
```

```
#report in a table
cbind(flesch_point,Year_ses)
```

```
##    Year     mean  Year_ses
## 1  1945 55.20642 1.4833240
## 2  1950 50.21797 1.7266025
## 3  1951 59.46630 1.8453322
## 4  1955 49.80629 2.4509111
## 5  1959 48.63661 1.1650446
## 6  1964 40.63344 1.5175465
## 7  1966 50.20094 1.3112333
## 8  1970 44.26262 0.9825483
## 9  1974 42.15639 1.9587855
## 10 1979 42.68944 0.3886173
## 11 1983 41.49276 0.9682493
## 12 1987 40.54144 1.2860997
## 13 1992 42.18025 1.2088081
## 14 1997 44.07867 0.6884571
## 15 2001 48.16852 0.7489051
## 16 2005 38.84223 1.0112467
```

9b

```r
# FRE
readability <- textstat_readability(CorpSub, "Flesch") #%>% head()
readability
```

```
##              document   Flesch
## 1  UK_natl_1945_en_Lab 51.86038
## 2  UK_natl_1950_en_Lab 50.70109
## 3  UK_natl_1951_en_Lab 58.91667
## 4  UK_natl_1955_en_Lab 50.10691
## 5  UK_natl_1959_en_Lab 49.34173
## 6  UK_natl_1964_en_Lab 38.34323
## 7  UK_natl_1966_en_Lab 44.81326
## 8  UK_natl_1970_en_Lab 42.59430
## 9  UK_natl_1974_en_Lab 40.71085
## 10 UK_natl_1979_en_Lab 42.79773
## 11 UK_natl_1983_en_Lab 41.35974
## 12 UK_natl_1987_en_Lab 41.48275
## 13 UK_natl_1992_en_Lab 42.13868
## 14 UK_natl_1997_en_Lab 44.64706
## 15 UK_natl_2001_en_Lab 48.28391
## 16 UK_natl_2005_en_Lab 38.00418
```

```r
# textstat_readability(texts(CorpSub, groups=="Year"), "Flesch"), there is no comparison in CorbSub
# but we can get the FRE in the whole corpus by using:
# textstat_readability(texts(data_corpus_ukmanifestos, groups = "Year"), "Flesch")
```

```r
cbind(flesch_point,readability)
```

```
##    Year     mean            document   Flesch
## 1  1945 55.20642 UK_natl_1945_en_Lab 51.86038
## 2  1950 50.21797 UK_natl_1950_en_Lab 50.70109
## 3  1951 59.46630 UK_natl_1951_en_Lab 58.91667
## 4  1955 49.80629 UK_natl_1955_en_Lab 50.10691
## 5  1959 48.63661 UK_natl_1959_en_Lab 49.34173
## 6  1964 40.63344 UK_natl_1964_en_Lab 38.34323
## 7  1966 50.20094 UK_natl_1966_en_Lab 44.81326
## 8  1970 44.26262 UK_natl_1970_en_Lab 42.59430
## 9  1974 42.15639 UK_natl_1974_en_Lab 40.71085
## 10 1979 42.68944 UK_natl_1979_en_Lab 42.79773
## 11 1983 41.49276 UK_natl_1983_en_Lab 41.35974
## 12 1987 40.54144 UK_natl_1987_en_Lab 41.48275
## 13 1992 42.18025 UK_natl_1992_en_Lab 42.13868
## 14 1997 44.07867 UK_natl_1997_en_Lab 44.64706
## 15 2001 48.16852 UK_natl_2001_en_Lab 48.28391
## 16 2005 38.84223 UK_natl_2005_en_Lab 38.00418
```

Bootstapping is via random sampling with replacement from our sample. Form the table above, I think the values are very similar.