**##################################################################################**

**####################### ANALYSIS (SECTIONS 4.2 AND 4.3) ##############################**

**################### Reading the data from file PrelimData.RData #########################**

**##################################################################################**

**rm(list = ls())**

**library(tm)**

**library(slam)**

**library(wordcloud)**

**## LOADING DATA (39th Congress)**

**load("C:\\Johannes Ledolter\\2020March01Book\\Chapter4WEB\\PrelimData.RData")**

**################################################################################**

**## CREATING THE CORPUS AND THE RESULTS IN CHAPTER 4 ##############################**

**################################################################################**

**## creating corpus**

**corpus <- VCorpus(VectorSource(data),readerControl = list(reader = readPlain))**

**corpus1 <- tm\_map(corpus, stripWhitespace)**

**corpus2 <- tm\_map(corpus1, content\_transformer(tolower))**

**corpus3 <- tm\_map(corpus2, removePunctuation)**

**corpus4 <- tm\_map(corpus3, removeNumbers)**

**corpus5 <- tm\_map(corpus4, removeWords, stopwords("english"))**

**corp.dtm <- DocumentTermMatrix(corpus5,control=list(stemming=FALSE)) ## no stemming is the default**

**dim(corp.dtm)**

**## in case you want to omit certain terms from the corpus such as "globe" or "unitedstates"**

**## dim(corp.dtm)**

**## corp.dtm=corp.dtm[,!labels(corp.dtm)$Terms=="globe"]**

**## dim(corp.dtm)**

**## corp.dtm=corp.dtm[,!labels(corp.dtm)$Terms=="unitedstates"]**

**## dim(corp.dtm)**

**## corp.dtm=corp.dtm[,!labels(corp.dtm)$Terms=="gentfrom"]**

**## dim(corp.dtm)**

**## corp.dtm=corp.dtm[,!labels(corp.dtm)$Terms=="senfrom"]**

**## dim(corp.dtm)**

**## corp.dtm=corp.dtm[,!labels(corp.dtm)$Terms=="friendfrom"]**

**## dim(corp.dtm)**

**## corp.dtm=corp.dtm[,!labels(corp.dtm)$Terms=="sensreps"]**

**## dim(corp.dtm)**

**as.matrix(corp.dtm[1:20,1:5])**

**## dtm and tdm matrices are going to be of very large dimensions**

**## they are stored in special format (triplet matrices)**

**## use the as.matrix function to display smaller parts of these matrices**

**Bcorp.dtm=weightBin(corp.dtm)**

**## creates truncated frequencies (binary occurrences)**

**TFIDFcorp.dtm=weightTfIdf(corp.dtm,normalize=TRUE)**

**## creates tfidf frequencies**

**## normalize=FALSE uses counts for term-frequencies**

**## normalize=TRUE uses relative frequencies for term-frequencies**

**## cutting out short letters**

**quantile(len)**

**indexlen=len>=32 ## can pick a length to omit short letters**

**nnn=dim(corp.dtm)[1]**

**number=dim(nnn)**

**for (i in 1:nnn) {**

**number[i]=i**

**}**

**corp.dtm=corp.dtm[number[indexlen],]**

**dim(corp.dtm)**

**meta2=meta2[number[indexlen]]**

**meta3=meta3[number[indexlen]]**

**meta4=meta4[number[indexlen]]**

**meta5=meta5[number[indexlen]]**

**meta6=meta6[number[indexlen]]**

**group=group[number[indexlen]]**

**len=len[number[indexlen]]**

**## cutting out short letters**

**## simplifying the corpus and its dtm**

**dim(corp.dtm)**

**## corpStripped.dtm=removeSparseTerms(corp.dtm,0.99) ## experiment with sparcity**

**## terms that show up in at least 1 percent of documents (for 50,000 documents: 50,000 x 0.01 = 500)**

**## dim(corpStripped.dtm)**

**corpStripped.dtm=removeSparseTerms(corp.dtm,0.999)**

**## terms that show up in at least 0.1 percent of documents (for 50,000 documents: 50,000 x 0.001 = 50)**

**dim(corpStripped.dtm)**

**## Check for documents with zero words as stripping words may lead a document with no words**

**## empty documents need to be omitted**

**dim(corpStripped.dtm)**

**index=row\_sums(corpStripped.dtm)==0**

**nnn=dim(corpStripped.dtm)[1]**

**number=dim(nnn)**

**for (i in 1:nnn) {**

**number[i]=i**

**}**

**number[index]**

**## do the following only if there are empty documents**

**corpStripped.dtm=corpStripped.dtm[-number[index],]**

**dim(corpStripped.dtm)**

**meta2=meta2[-number[index]]**

**meta3=meta3[-number[index]]**

**meta4=meta4[-number[index]]**

**meta5=meta5[-number[index]]**

**meta6=meta6[-number[index]]**

**group=group[-number[index]]**

**len=len[-number[index]]**

**## WORD FREQUENCIES AND WORD CLOUDS**

**## PROGRAM SHOWN BELOW AVOIDS PROBLEMS IF DTM MATRIX INCLUDES MANY TERMS**

**## lists all words that are mentioned more than 2,000 times**

**findFreqTerms(corpStripped.dtm,2000)**

**## plotting the frequency distribution**

**ffff=rollup(t(as.DocumentTermMatrix(corpStripped.dtm)),2,na.rm=TRUE,FUN=sum)**

**freq=as.vector(ffff)**

**labels=rownames(ffff)**

**wf=data.frame(labels,freq)**

**wf=wf[order(-wf$freq),]**

**library(ggplot2)**

**p=ggplot(subset(wf,freq>15000),aes(labels,freq))**

**p=p+geom\_bar(stat="identity") + xlab("Word") + ylab("Frequency")**

**p=p+theme(axis.text.x=element\_text(angle=45,hjust=1))**

**p**

**## displaying word clouds**

**set.seed(142)**

**dark2 <- brewer.pal(6,"Dark2")**

**wordcloud(labels,freq,max.words=50,rot.per=0.2,colors=dark2)**

**## WORD FREQUENCIES AND WORD CLOUDS**

**## colSums MAY NOT WORK IF THERE ARE TOO MANY TERMS IN DTM MATRIX**

**## TO MAKE THIS WORK WE EDUCE THE TERMS IN DTM (through removeSparseTerms)**

**## lists all words that are mentioned more than 2,000 times**

**findFreqTerms(corpStripped.dtm,2000)**

**## plotting the frequency distribution**

**freq=colSums(as.matrix(corpStripped.dtm))**

**freq=sort(colSums(as.matrix(corpStripped.dtm)),decreasing=TRUE)**

**wf=data.frame(word=names(freq),freq=freq)**

**library(ggplot2)**

**p=ggplot(subset(wf,freq>15000),aes(word,freq))**

**p=p+geom\_bar(stat="identity")**

**p=p+theme(axis.text.x=element\_text(angle=45,hjust=1))**

**p**

**## displaying word clouds**

**set.seed(142)**

**dark2 <- brewer.pal(6,"Dark2")**

**wordcloud(names(freq),freq,max.words=50,rot.per=0.2,colors=dark2)**

**## CHECKING ON SOME SPECIFIC FREQUENCIES**

**sum(as.matrix(corpStripped.dtm[,"slave"]))**

**sum(as.matrix(corpStripped.dtm[,"slaveholder"]))**

**sum(as.matrix(corpStripped.dtm[,"rfg"]))**

**sum(as.matrix(corpStripped.dtm[,"sec"]))**

**sum(as.matrix(corpStripped.dtm[,"abife"]))**

**sum(as.matrix(corpStripped.dtm[,"postoffice"]))**

**sum(as.matrix(corpStripped.dtm[,"glo"]))**

**sum(as.matrix(corpStripped.dtm[,"horep"]))**

**sum(as.matrix(corpStripped.dtm[,"sensreps"]))**

**sum(as.matrix(corpStripped.dtm[,"civilwar"]))**

**sum(as.matrix(corpStripped.dtm[,"declarindep"]))**

**sum(as.matrix(corpStripped.dtm[,"globe"]))**

**sum(as.matrix(corpStripped.dtm[,"unitedstates"]))**

**sum(as.matrix(corpStripped.dtm[,"gentfrom"]))**

**sum(as.matrix(corpStripped.dtm[,"senfrom"]))**

**sum(as.matrix(corpStripped.dtm[,"friendfrom"]))**

**sum(as.matrix(corpStripped.dtm[,"capitalist"]))**

**## WORD ASSOCIATIONS**

**## may be slow if there are many terms in the dtm matrix**

**findAssocs(corpStripped.dtm, "white", 0.5)**

**findAssocs(corpStripped.dtm, "negro", 0.5)**

**findAssocs(corpStripped.dtm, "war", 0.5)**

**findAssocs(corpStripped.dtm, "slave", 0.5)**

**findAssocs(corpStripped.dtm, "labor", 0.5)**

**## weightBin creates indicator variables for absence/presence of a term**

**BcorpStripped.dtm=weightBin(corpStripped.dtm)**

**findAssocs(BcorpStripped.dtm, "white", 0.3)**

**findAssocs(BcorpStripped.dtm, "negro", 0.3)**

**findAssocs(BcorpStripped.dtm, "war", 0.3)**

**findAssocs(BcorpStripped.dtm, "slave", 0.3)**

**findAssocs(BcorpStripped.dtm, "labor", 0.3)**

**## combining terms such as "slave" and "slaveholder" and adding a column to the .dtm matrix**

**y1=as.matrix(corpStripped.dtm[,"slave"])**

**y2=as.matrix(corpStripped.dtm[,"slavery"])**

**corpTEMP.dtm=corpStripped.dtm[,1] ## using any column as placeholder**

**dimnames(corpTEMP.dtm)$Terms="slave slaveholder"**

**corpTEMP.dtm[,1]=y1+y2**

**corpStrippedExp.dtm=as.DocumentTermMatrix(cbind(corpStripped.dtm,corpTEMP.dtm),weighting=weightTf)**

**dim(corpStrippedExp.dtm)**

**labels(corpStrippedExp.dtm)$Terms**

**## WORD ASSOCIATIONS WITH THE NEW EXPANDED DTM**

**findAssocs(corpStrippedExp.dtm, "white", 0.5)**

**findAssocs(corpStrippedExp.dtm, "negro", 0.5)**

**findAssocs(corpStrippedExp.dtm, "war", 0.5)**

**findAssocs(corpStrippedExp.dtm, "slave", 0.5)**

**findAssocs(corpStrippedExp.dtm, "labor", 0.5)**

**## weightBin creates indicator variables for presence of term**

**BcorpStrippedExp.dtm=weightBin(corpStrippedExp.dtm)**

**findAssocs(BcorpStrippedExp.dtm, "white", 0.3)**

**findAssocs(BcorpStrippedExp.dtm, "negro", 0.3)**

**findAssocs(BcorpStrippedExp.dtm, "war", 0.3)**

**findAssocs(BcorpStrippedExp.dtm, "slave", 0.3)**

**findAssocs(BcorpStrippedExp.dtm, "labor", 0.3)**

**## WORD ASSOCIATIONS (TWO CLUSTERS): Explore the relationships on your own**

**findAssocs(corpStripped.dtm, "race", 0.5)**

**findAssocs(corpStripped.dtm, "white", 0.5)**

**findAssocs(corpStripped.dtm, "black", 0.5)**

**findAssocs(corpStripped.dtm, "negro", 0.5)**

**findAssocs(corpStripped.dtm, "slave", 0.5)**

**findAssocs(corpStripped.dtm, "treason", 0.5)**

**findAssocs(corpStripped.dtm, "traitor", 0.5)**

**findAssocs(corpStripped.dtm, "secession", 0.5)**

**findAssocs(corpStripped.dtm, "betrayal", 0.5)**

**## weightBin creates indicator variables for presence of term**

**BcorpStripped.dtm=weightBin(corpStripped.dtm)**

**findAssocs(BcorpStripped.dtm, "race", 0.3)**

**findAssocs(BcorpStripped.dtm, "white", 0.3)**

**findAssocs(BcorpStripped.dtm, "black", 0.3)**

**findAssocs(BcorpStripped.dtm, "negro", 0.3)**

**findAssocs(BcorpStripped.dtm, "slave", 0.3)**

**findAssocs(BcorpStripped.dtm, "treason", 0.3)**

**findAssocs(BcorpStripped.dtm, "traitor", 0.3)**

**findAssocs(BcorpStripped.dtm, "secession", 0.3)**

**findAssocs(BcorpStripped.dtm, "betrayal", 0.3)**

**## Zipf's law: applied to ALL speeches of the 39th Congress**

**library(VGAM)**

**## needed for the Riemann zeta function**

**## to calculate the sum of the general harmonic series (p-series)**

**library(poweRlaw)**

**## creating corpus from ALL documents**

**corpus <- VCorpus(VectorSource(data),readerControl = list(reader = readPlain))**

**corpus1 <- tm\_map(corpus, stripWhitespace)**

**corpus2 <- tm\_map(corpus1, content\_transformer(tolower))**

**corpus3 <- tm\_map(corpus2, removePunctuation)**

**corpus4 <- tm\_map(corpus3, removeNumbers)**

**corpus5 <- tm\_map(corpus4, removeWords, stopwords("english"))**

**corp.dtm <- DocumentTermMatrix(corpus5,control=list(stemming=FALSE)) ## no stemming is the default**

**dim(corp.dtm)**

**ffff=rollup(t(as.DocumentTermMatrix(corp.dtm)),2,na.rm=TRUE,FUN=sum)**

**wordfreq=as.vector(ffff) ## word frequencies**

**labels=rownames(ffff)**

**dd=data.frame(labels,wordfreq)**

**dd=dd[order(-dd$wordfreq),]**

**dd[1:50,]**

**hist(wordfreq)**

**hist(log(wordfreq),main="Histogram: Logarithm of word frequencies")**

**tt=data.frame(table(wordfreq))**

**tt[1:10,]**

**xval=as.numeric(levels(tt[,1]))**

**freq=tt[,2]**

**plot(freq~xval,ylab="number of words of given frequency",xlab="frequency of occurrence")**

**plot(freq~xval,xlim=c(0,50),ylab="number of words of given frequency",xlab="frequency of occurrence")**

**lnfreq=log(freq)**

**lnxval=log(xval)**

**plot(lnfreq~lnxval,ylab="logarithm of the number of words of given frequency",xlab="log frequency of occurrence")**

**fit=lm(lnfreq~lnxval)**

**summary(lm(lnfreq~lnxval))**

**alphaLS=-fit$coef[2] ## LS estimate of alpha**

**alphaLS**

**plot(lnfreq~lnxval,ylab="logarithm of the number of words of given frequency",xlab="log frequency of occurrence")**

**abline(fit)**

**## omitting word frequencies that occur less than k times**

**k=2 ## this omits word frequencies that occur just once**

**ind=freq<2**

**yy=lnfreq[!ind]**

**xx=lnxval[!ind]**

**fit=lm(yy~xx)**

**summary(lm(yy~xx))**

**alphaLS=-fit$coef[2] ## LS estimate of alpha**

**alphaLS**

**plot(yy~xx,ylab="logarithm of the number of words of given frequency",xlab="log frequency of occurrence")**

**abline(fit)**

**## omitting word frequencies that occur less than k times**

**## maximum likelihood estimation**

**x=wordfreq**

**nuobs=length(x)**

**nuobs**

**sus=sum(log(x))**

**sus**

**obj=dim(200)**

**alpha=dim(200)**

**par=1.00**

**for (jjj in 1:200) {**

**par=par+0.01**

**alpha[jjj]=par**

**sum=zeta(par) ## calculates the Riemann zeta function**

**obj[jjj]=-nuobs\*log(sum)-par\*sus**

**}**

**alpha**

**obj**

**plot(obj~alpha,ylab="log-likelihood",xlab="alpha")**

**gg=obj==max(obj)**

**alpha[gg==TRUE] ## MLE of alpha**

**## using the R library poweRlaw**

**library(poweRlaw)**

**m = displ$new(x)**

**## estimate\_xmin(m)**

**## m$setXmin(1)**

**estimate\_pars(m)**

**## using the R library poweRlaw**

**## estimating xmin**

**library(poweRlaw)**

**m = displ$new(x)**

**estimate\_xmin(m)**

**m$setXmin(3)**

**estimate\_pars(m)**

**################################################################################################################## ZIPF LAW: Program Zipf.docx ################################################### #################################################################################################**

**## following program is stored in Zipf.docx**

**library(VGAM)**

**## needed for the Riemann zeta function**

**## to calculate the sum of the general harmonic series (p-series)**

**l**

**## generating data from a discrete Pareto distribution**

**set.seed(1) ## seed for the random number generator (for repeatability)**

**n=10000 ## consider max word frequency n**

**xx=dim(n)**

**prob=dim(n)**

**xx=c(1:n)**

**xx**

**alpha=1.25 ## selected alpha**

**for (i in 1:n) {**

**prob[i]=xx[i]^(-alpha)**

**}**

**prob=prob/sum(prob)**

**sum(prob)**

**plot(prob)**

**sampleDist = function(n) {**

**sample(xx, n, replace = T, prob)**

**}**

**N=100000 ## number of words in corpus: feel free to experiment with different numbers**

**x=sampleDist(N)**

**x**

**t=table(x)**

**dim(t)**

**freq=dim(dim(t))**

**for (i in 1 :dim(t)) {**

**freq[i]=t[[i]]**

**}**

**freq**

**xval=as.numeric(levels(factor(x)))**

**freq**

**xval**

**plot(freq~xval,ylab="number of words of given frequency",xlab="frequency of occurrence")**

**plot(freq~xval,xlim=c(0,50),ylab="number of words of given frequency",xlab="frequency of occurrence")**

**lnfreq=log(freq)**

**lnxval=log(xval)**

**plot(lnfreq~lnxval)**

**fit=lm(lnfreq~lnxval)**

**summary(lm(lnfreq~lnxval))**

**alphaLS=-fit$coef[2] ## LS estimate of alpha**

**alphaLS**

**plot(lnfreq~lnxval,ylab="logarithm of the number of words of given frequency",xlab="log frequency of occurrence")**

**abline(fit)**

**## omitting word frequencies that occur less than k times**

**k=2 ## this omits word frequencies that occur just once**

**ind=freq<2**

**yy=lnfreq[!ind]**

**xx=lnxval[!ind]**

**fit=lm(yy~xx)**

**summary(lm(yy~xx))**

**alphaLS=-fit$coef[2] ## LS estimate of alpha**

**alphaLS**

**plot(yy~xx,ylab="logarithm of the number of words of given frequency",xlab="log frequency of occurrence")**

**abline(fit)**

**## omitting word frequencies that occur less than k times**

**## maximum likelihood estimation**

**x**

**nuobs=length(x)**

**nuobs**

**sus=sum(log(x))**

**sus**

**obj=dim(200)**

**alpha=dim(200)**

**par=1.00**

**for (jjj in 1:200) {**

**par=par+0.01**

**alpha[jjj]=par**

**sum=zeta(par) ## calculates the Riemann zeta function**

**obj[jjj]=-nuobs\*log(sum)-par\*sus**

**}**

**alpha**

**obj**

**plot(obj~alpha,ylab="log-likelihood",xlab="alpha")**

**gg=obj==max(obj)**

**alpha[gg==TRUE] ## MLE of alpha**

**## using the R library poweRlaw**

**library(poweRlaw)**

**m = displ$new(x)**

**## estimate\_xmin(m)**

**## m$setXmin(1)**

**estimate\_pars(m)**

**## using the R library poweRlaw**

**## estimating xmin**

**estimate\_xmin(m)**

**m$setXmin(3)**

**estimate\_pars(m)**