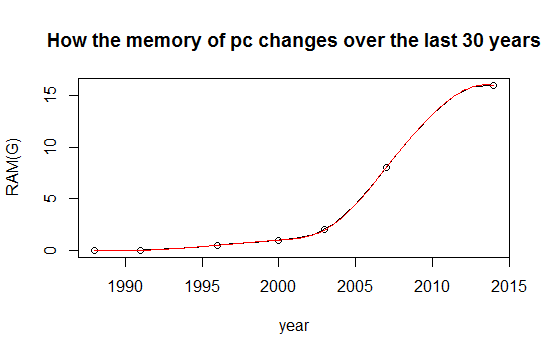
Qiuxing Yu 27720161153029

Homework1

Question1

This is the development path of memory of PCs



Corresponding code

year<c(1988,1991,1996,2000,2003,2007,2014)

RAM<-c(0.002,0.004,0.5,1,2,8,16)

plot(year,RAM,ylab = "")

title(main ="How the memory of pc changes over the last 30 years",ylab = "RAM(G)")

lines(spline(year,RAM))

lines(spline(year,RAM, n = 201), col = 2)

Twenty years ago,the memory of pc is too small to support the analysis of FMRY

Question2

Logistic regression:

In statistics, logistic regression, or logit regression, or logit model[1] is a regression model where the dependent variable (DV) is categorical. This article covers the case of a binary dependent variable—that is, where the output can take only two values, "0" and "1", which represent outcomes such as pass/fail, win/lose, alive/dead or healthy/sick. Cases where the dependent variable has more than two outcome categories may be analysed in multinomial logistic regression, or, if the multiple categories are ordered, in ordinal logistic regression.[2] In the terminology of economics, logistic regression is an example of a qualitative response/discrete choice model.

A example:Probability of passing an exam versus hours of study

Suppose we wish to answer the following question:

A group of 20 students spend between 0 and 6 hours studying for an exam. How does the number of hours spent studying affect the probability that the student will pass the exam?

The reason for using logistic regression for this problem is that the dependent variable pass/fail represented by "1" and "0" are not cardinal numbers. If the problem was changed so that pass/fail was replaced with the grade 0–100 (cardinal numbers), then simple regression analysis could be used.

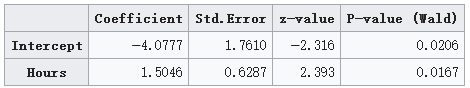
The table shows the number of hours each student spent studying, and whether they passed (1) or failed (0)



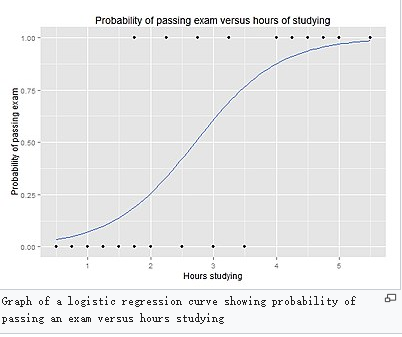
The graph shows the probability of passing the exam versus the number

of hours studying, with the logistic regression curve fitted to the data.

The logistic regression analysis gives the following output.







Homework2

Q1

Code:

x = 6

n = 1000

lambda = 2

p = lambda / n

dbinom (x,2\*n,p) # binomial probability mass function

dpois (x, 2\*lambda ) # Poisson probability mass function

dpois (0, 5 )

Q2

Code:

plot(year,number,type = "b",

col="black",main = "The history of computer memory",

sub = "This is the change in the number of types of computer memory",

xlab = "year",ylab = "The number of memory")

barplot(number,

xlab = "year",ylab ="The number of memory ")

Q3

Code:

lambda=2

x=seq(0:6)

P<-data.frame(dpois(x,lambda))

sum<-(P[1,]+P[7,]+P[2,]+P[6,]+P[3,]+P[5,]+P[4,]+P[4,])

sum

lambda=5

x=0

dpois(x,lambda)

Homework3

Q1

library("digest")

# now do the hash code calculation

digest("I learn a lot from this class when I am proper listening to the professor")

digest("I do not learn a lot from this class when I am absent and playing on my Iphone")

Q2

What Is DSA (Digital Signature Algorithm)?

Digital signatures are essential to verify the sender of a document’s identity. The signature is computer using a set of rules and algorithm such that the identity of the person can be verified.

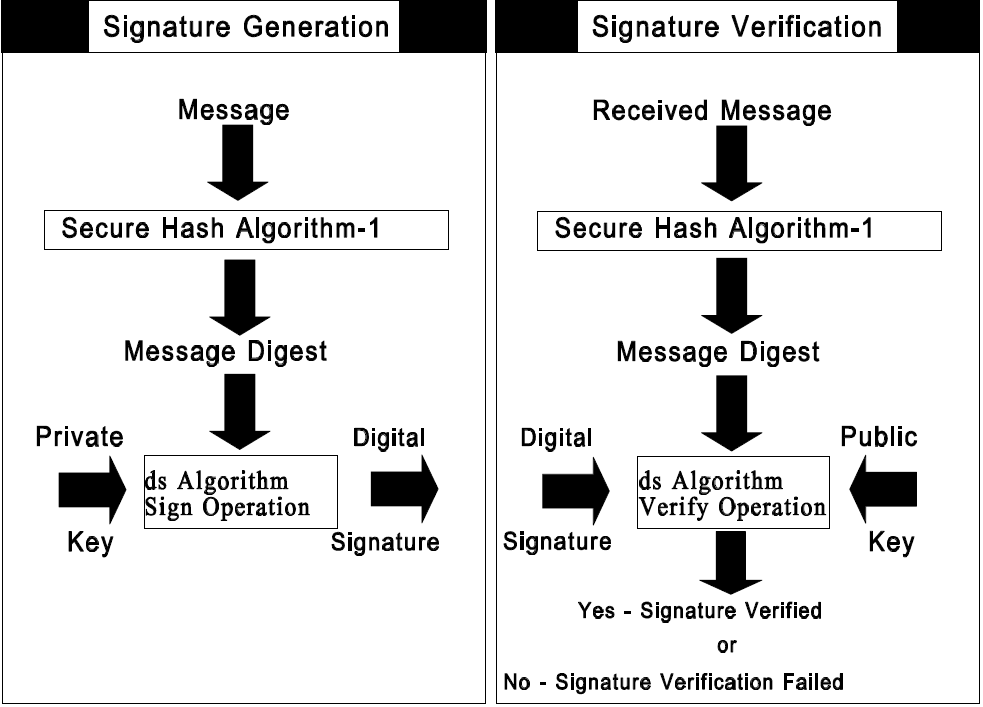
The signature is generated by the use of a private key that known only to the user. The signature is verified when a public key is corresponds to the private key. With every user having a public/private key pair, this is an example of public-key cryptography.

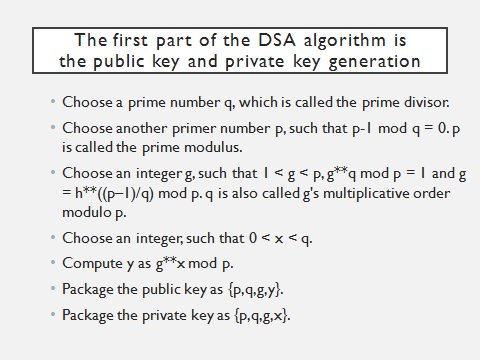
Public keys, which are known by everyone, can be used to verify the signature of a user. The private key, which is never shared, is used in signature generation, which can only be done by the user.

What can DSA do?

Digital signatures are used to detect unauthorized modifications to data. Also, the recipient of a digitally signed document in proving to a third party that the document was indeed signed by the person who it is claimed to be signed by. This is known as nonrepudiation, because the person who signed the document cannot repudiate the signature at a later time.

Digital signature algorithms can be used in e-mails, electronic funds transfer, electronic data interchange, software distribution, data storage, and just about any application that would need to assure the integrity and originality of dat





Q3

R-code:

>library(RJSONIO)

> letter<-LETTERS[1:10]

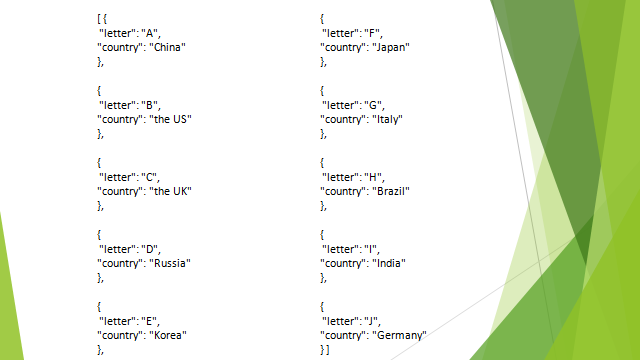
>country<-c("China","the US","the UK","Russia",

"Korea","Japan","Italy","Brazil","India","Germany")

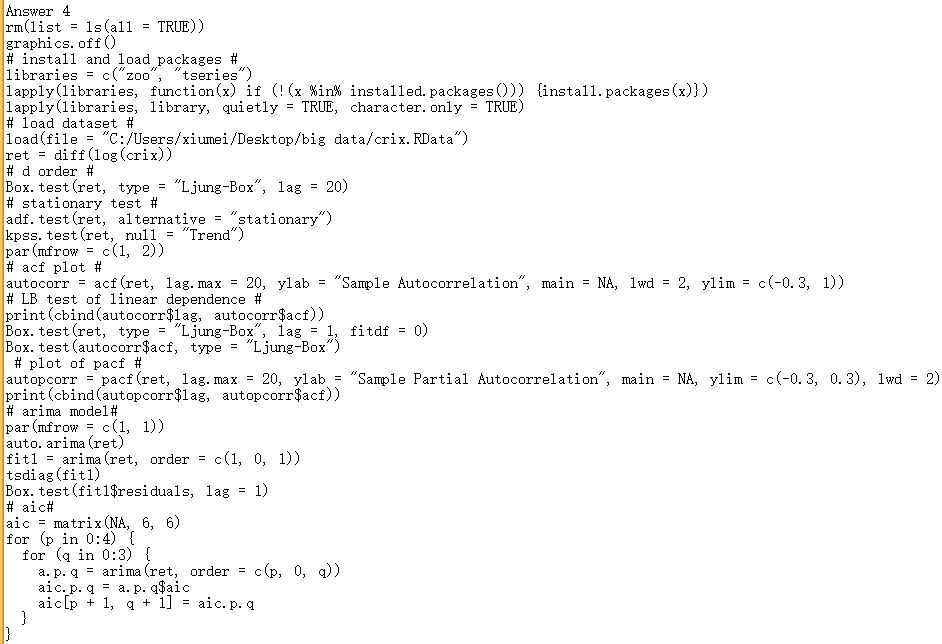
> data<-data.frame(letter,country)

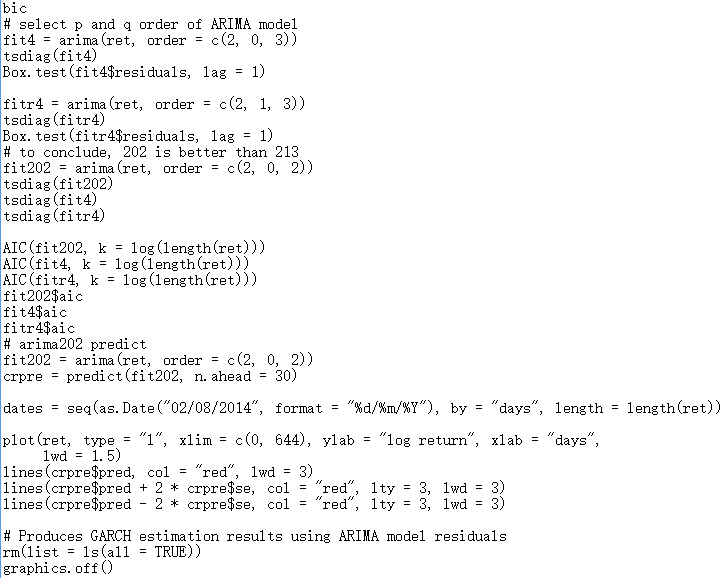
> da<-as.matrix(data)

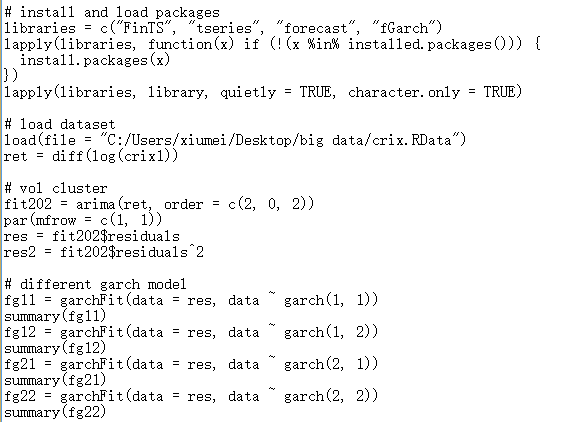
>cat(toJSON(da))

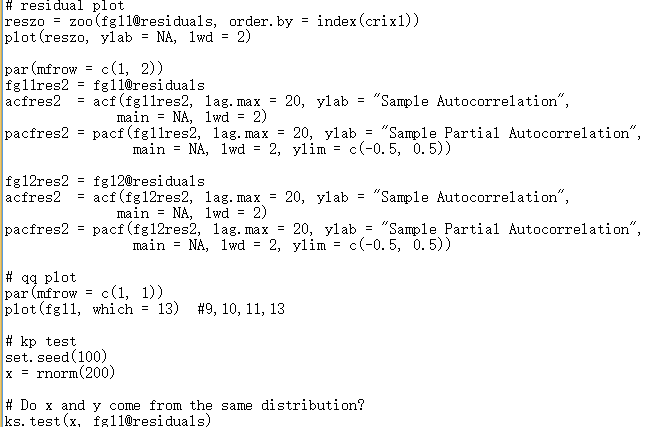


Q4



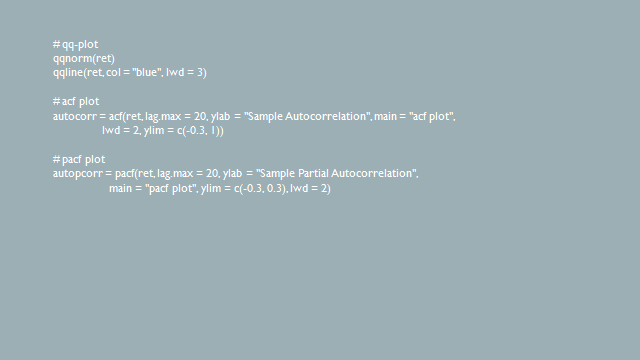
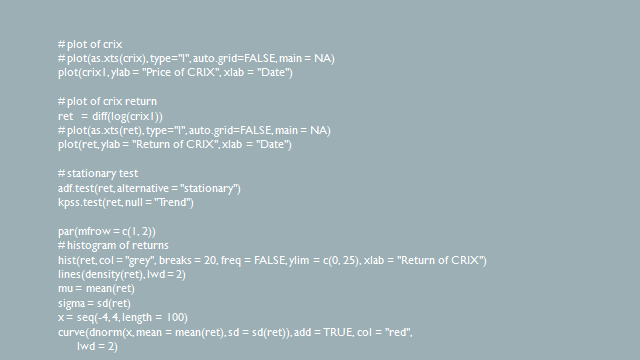
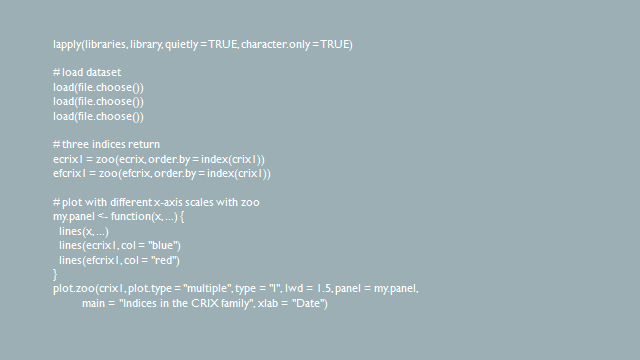
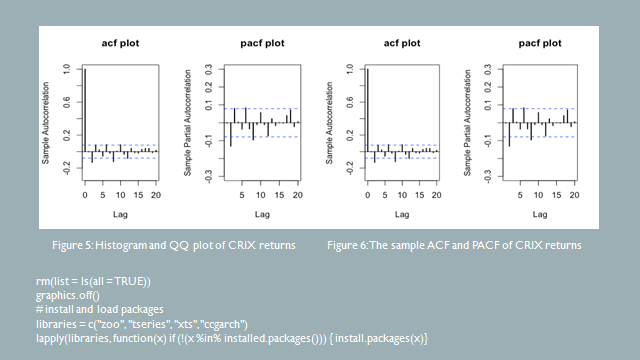
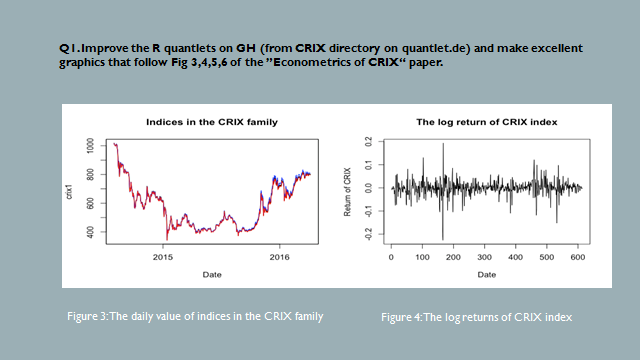


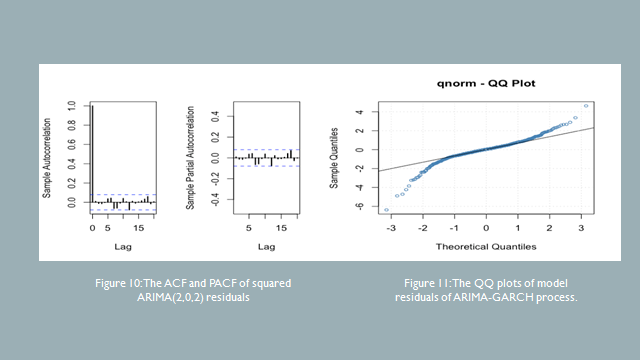
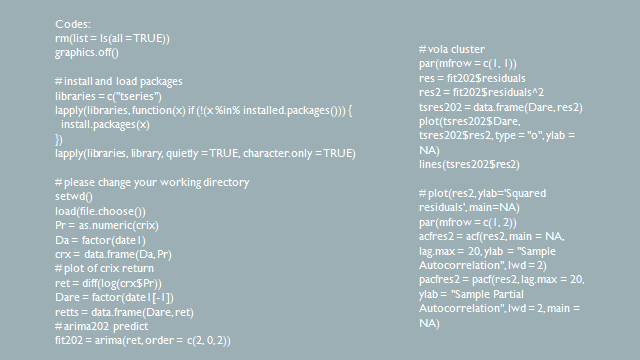
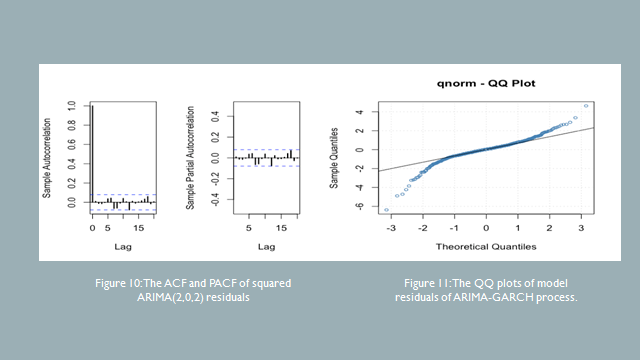
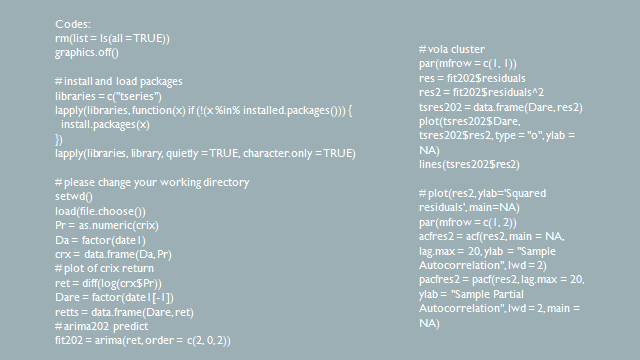
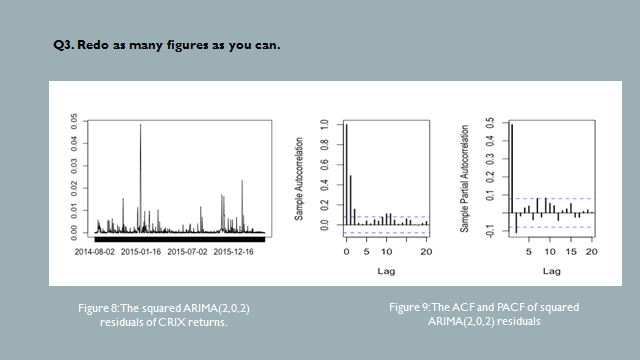
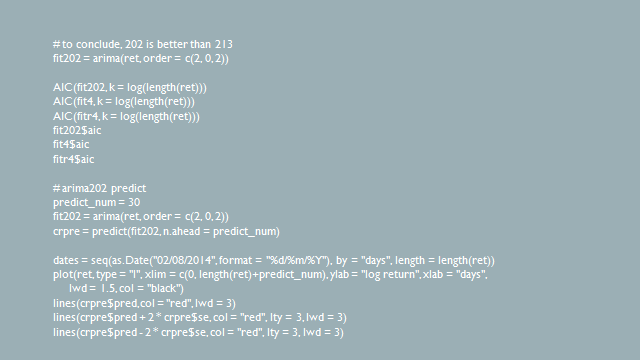
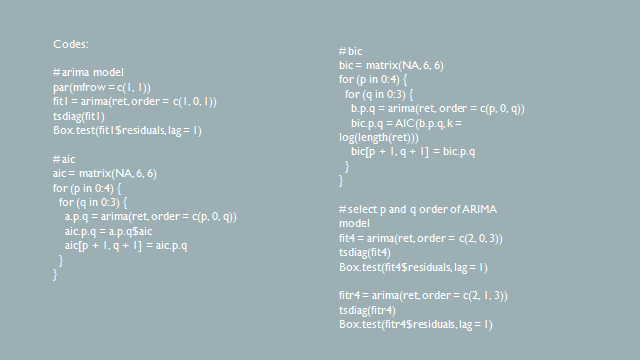
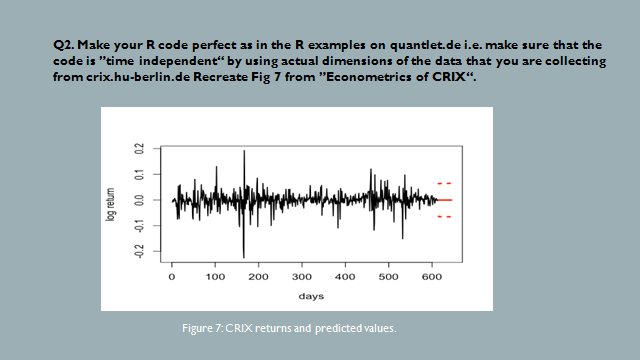




Homewor4

Q1

Q2



**Homework 5**

Q1. do a word cloud for Shakesspeare’s dramas. Romeo and Julia, Julius Caesar, Hamlet.

rm(list = ls())

#install.packages("RCurl")

#install.packages("XML")

library(RCurl)

library(XML)

url1 = "http://shakespeare.mit.edu/romeo\_juliet/full.html"

url2 = "http://shakespeare.mit.edu/julius\_caesar/full.html"

url3 = "http://shakespeare.mit.edu/hamlet/full.html"

html1 = readLines(url1, encoding = "UTF-8")

html2 = readLines(url2, encoding = "UTF-8")

html3 = readLines(url3, encoding = "UTF-8")

html1 = htmlParse(html1, encoding = "UTF-8")

html2 = htmlParse(html2, encoding = "UTF-8")

html3 = htmlParse(html3, encoding = "UTF-8")

#install.packages("bitops")

#install.packages("stringr")

library(bitops)

library(stringr)

abs1 = lapply(url1, FUN = function(x) htmlParse(x, encoding = "Latin-1"))

abs2 = lapply(url2, FUN = function(x) htmlParse(x, encoding = "Latin-1"))

abs3 = lapply(url3, FUN = function(x) htmlParse(x, encoding = "Latin-1"))

clean\_txt = function(x) {

cleantxt = xpathApply(x, "//body//text()

[not(ancestor :: script)][ not(ancestor :: style)]

[not(ancestor :: noscript)] " ,xmlValue)

cleantxt = paste(cleantxt, collapse="\n")

cleantxt = str\_replace\_all(cleantxt, "\n", " ")

cleantxt = str\_replace\_all(cleantxt, "\r", "")

cleantxt = str\_replace\_all(cleantxt, "\t", "")

cleantxt = str\_replace\_all(cleantxt, "<br>", "")

return(cleantxt)

}

cleantxt1 = lapply(abs1,clean\_txt)

cleantxt2 = lapply(abs2,clean\_txt)

cleantxt3 = lapply(abs3,clean\_txt)

vec\_abs1 = unlist(cleantxt1)

vec\_abs2 = unlist(cleantxt2)

vec\_abs3 = unlist(cleantxt3)

#install.packages("tm")

#install.packages("SnowballC")

library(tm)

library(SnowballC)

abs1 = Corpus(VectorSource(vec\_abs1))

abs2 = Corpus(VectorSource(vec\_abs2))

abs3 = Corpus(VectorSource(vec\_abs3))

abs\_dtm1 = DocumentTermMatrix(abs1, control = list(

stemming = TRUE, stopwords = TRUE, minWordLength = 3,

removeNumbers = TRUE, removePunctuation = TRUE))

abs\_dtm2 = DocumentTermMatrix(abs2, control = list(

stemming = TRUE, stopwords = TRUE, minWordLength = 3,

removeNumbers = TRUE, removePunctuation = TRUE))

abs\_dtm3 = DocumentTermMatrix(abs3, control = list(

stemming = TRUE, stopwords = TRUE, minWordLength = 3,

removeNumbers = TRUE, removePunctuation = TRUE))

##WordCloud

instal.packages("ggplot2")

install.packages("wordcloud")

library(ggplot2)

library(wordcloud)

freq1 = colSums(as.matrix(abs\_dtm1))

freq2 = colSums(as.matrix(abs\_dtm2))

freq3 = colSums(as.matrix(abs\_dtm3))

wf1 = data.frame(word=names(freq1), freq=freq1)

wf2 = data.frame(word=names(freq2), freq=freq2)

wf3 = data.frame(word=names(freq3), freq=freq3

#Romeo and Juliet

plot1 = ggplot(subset(wf1, freq>15), aes(word, freq1))

plot1 = plot1 + geom\_bar(stat="identity")

plot1 = plot1 + theme(axis.text.x=element\_text(angle=45, hjust=1))

plot1

freq1 = colSums(as.matrix(abs\_dtm1))

dark2\_1 = brewer.pal(6, "Dark2")

wordcloud(names(freq1), freq1, max.words=100, rot.per=0.2, colors=dark2\_1)

#Julius Caeser

plot2 = ggplot(subset(wf2, freq>15), aes(word, freq2))

plot2 = plot2 + geom\_bar(stat="identity")

plot2 = plot2 + theme(axis.text.x=element\_text(angle=45, hjust=1))

plot2

freq2 = colSums(as.matrix(abs\_dtm2))

dark2\_2 = brewer.pal(6, "Dark2")

wordcloud(names(freq2), freq2, max.words=100, rot.per=0.2, colors=dark2\_2)

#Hamlet

plot3 = ggplot(subset(wf3, freq>15), aes(word, freq3))

plot3 = plot3 + geom\_bar(stat="identity")

plot3 = plot3 + theme(axis.text.x=element\_text(angle=45, hjust=1))

plot3

freq3 = colSums(as.matrix(abs\_dtm3))

dark2\_3 = brewer.pal(6, "Dark2")

wordcloud(names(freq3), freq3, max.words=100, rot.per=0.2, colors=dark2\_3)

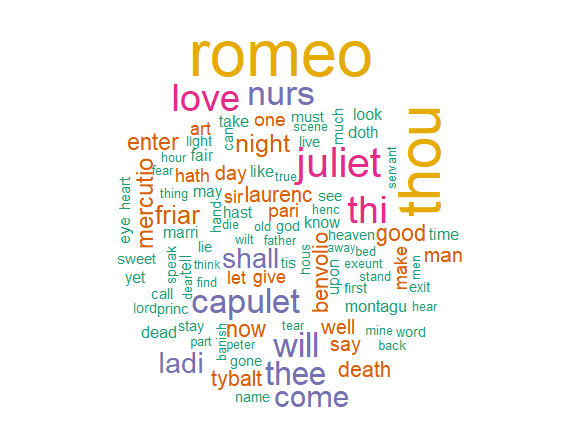


Figure 1: Romeo and Julia

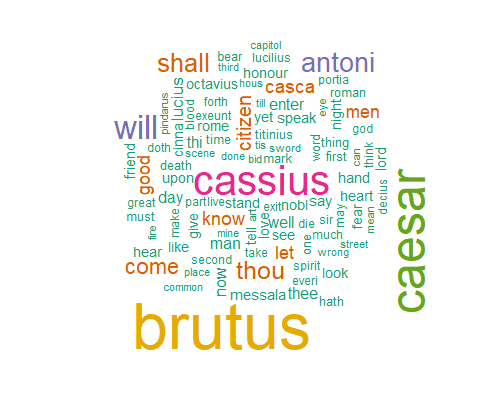


Figure 2: Julius Caesar

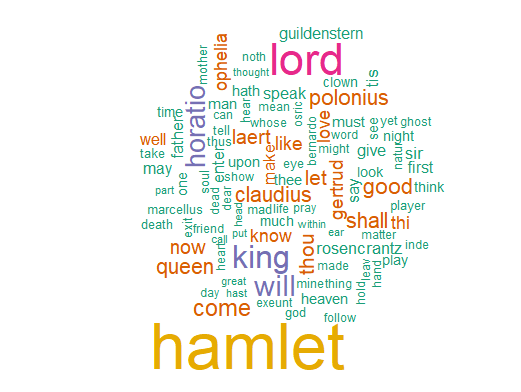


Figure 3: Hamlet

Q2. calculate the histogram of words

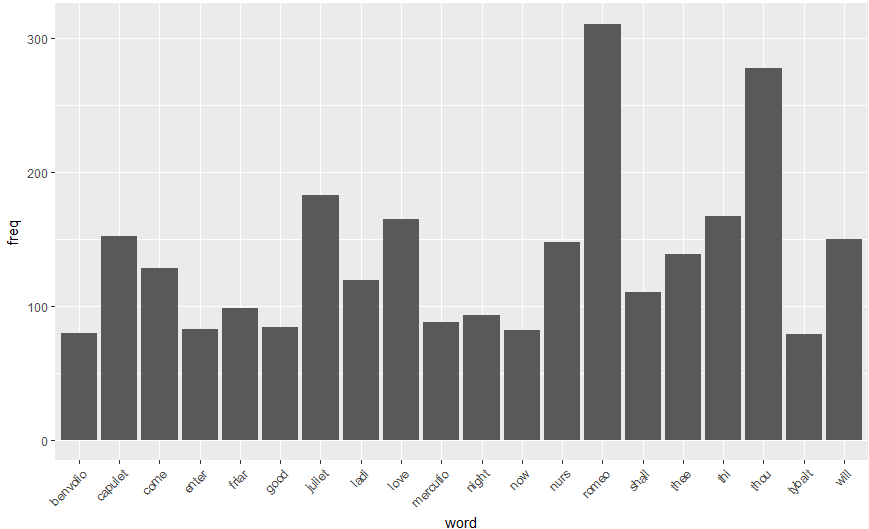


Figure 4: Romeo and Julia

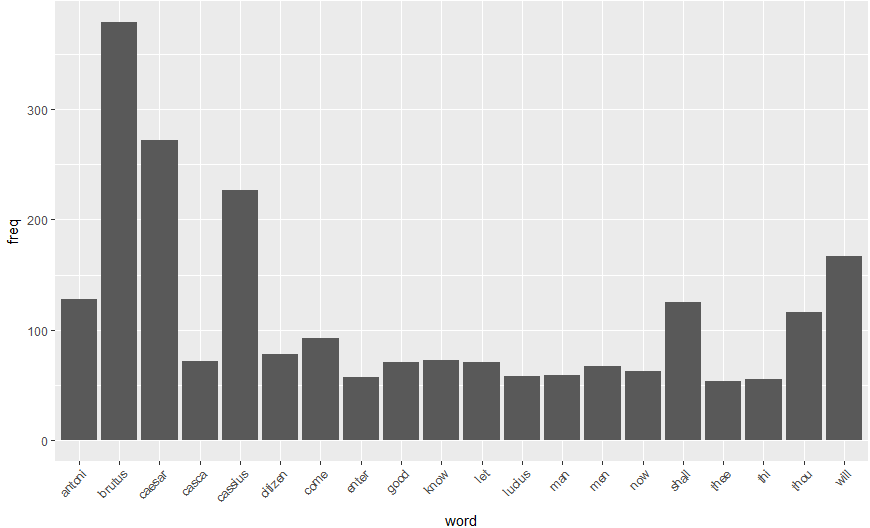


Figure 5: Julius Caesar

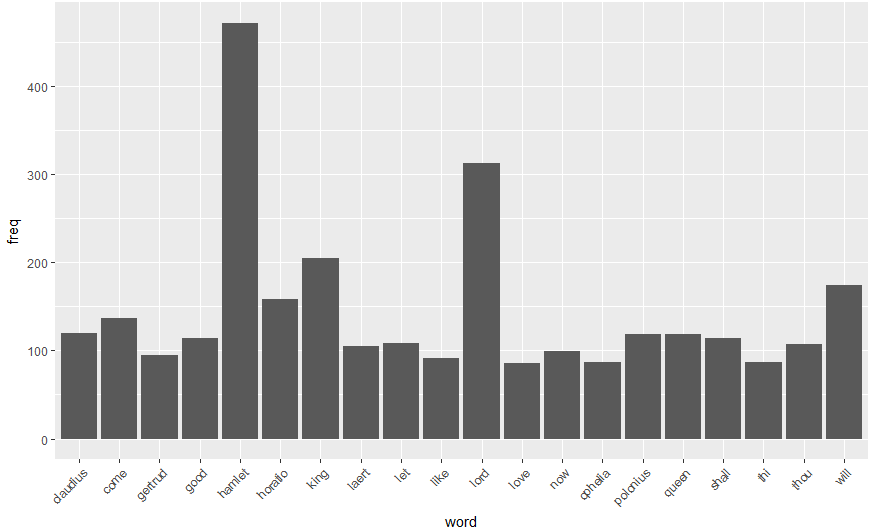


Figure 6: Hamlet