HW7_IS457_8

Part 1: Simulation

Let X and Y be two random variables following normal distribution.

We will use the simulation techniques to find the distribution of X+Y.

(a)

As we will generate random number, to ensure reproducibility, please set the seed as 457. (1 pt)

Your answer here

```
set.seed(457)
```

(b)

Generate 1000 samples from normal distribution with mean=10, standard deviation=2 as X and

the other 1000 samples from normal distribution with mean=5, standard deviation=3 as Y.

Then find the mean and standard deviation of X+Y. (1 pt)

Your answer here

```
x = rnorm(1000,mean = 10,sd=2)
set.seed(457)
y= rnorm(1000,mean=5,sd=3)
mean(x+y)
```

```
## [1] 15.048
```

```
sd(x+y)
```

```
## [1] 5.081643
```

(c)

Now use the simulation to estimate the distribution of X+Y and create the confidence intervals.

(1)

Form a set of Xs and Ys by repeating individual experiment for B = 2000 times,

each experiment has n = 1000 samples. You may want to write a for loop and create two

matrices "sample_X" and "sample_Y" to save those values (see class notes). (2 pts)

Your answer here

```
sample_x= matrix(nrow = 2000,ncol=1000)
set.seed(457)

for(i in 1:2000){
  sample_x[i,]=rnorm(1000,mean=10,sd=2)
}

sample_y=matrix(nrow=2000,ncol=1000)
set.seed(457)
for(i in 1:2000){
  sample_y[i,]=rnorm(1000,mean=5,sd=3)
}
```

(2)

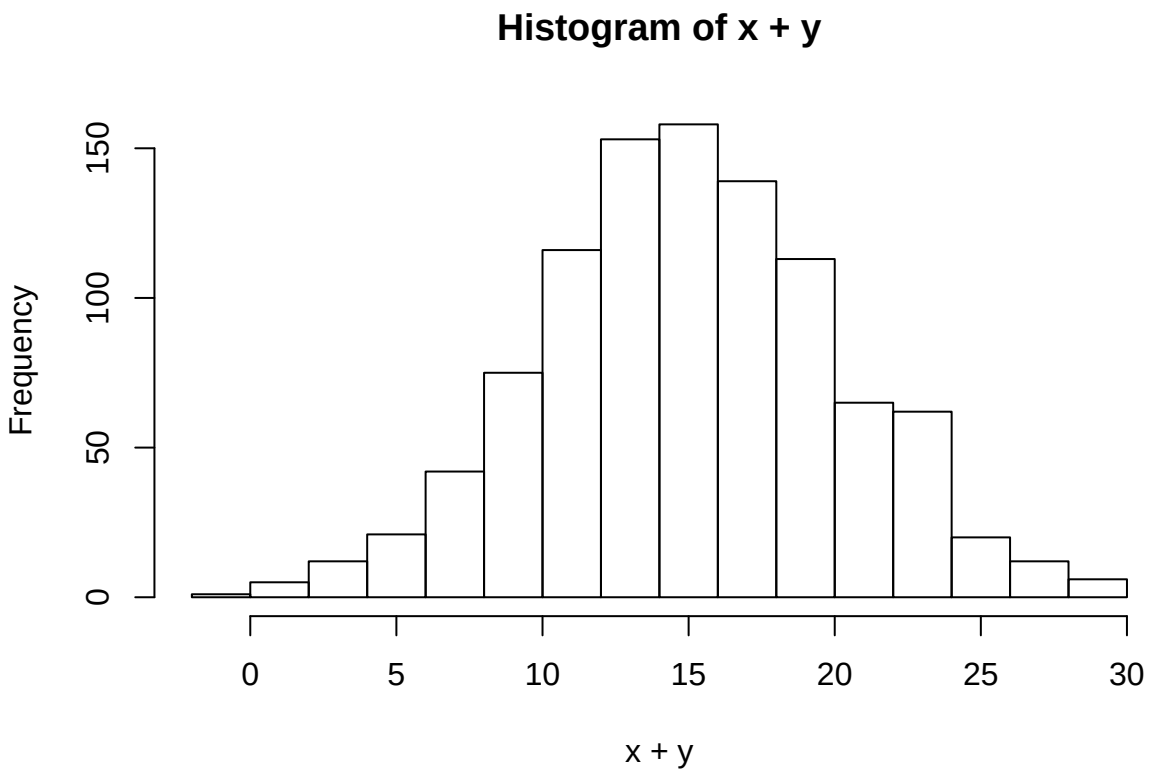
Calculate mean of X+Y for each experiment and save it

to a vector which has a length of B, and plot a histogram of

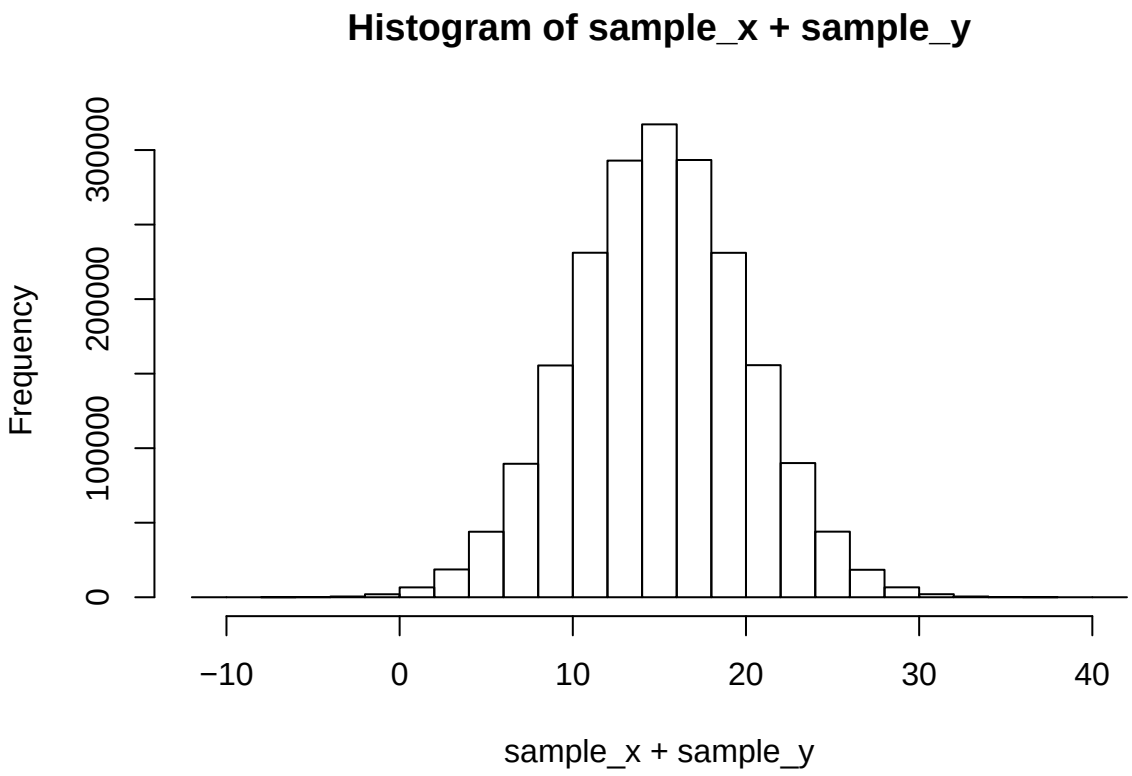
these means. (2 pts)

Your answer here

```
new_vector_1= mean(x+y)
new_vector_2= mean(sample_x+sample_y)
hist(x+y)
```




```
hist(sample_x + sample_y)
```



(3)

Now as we have a simulated sampling distribution of X+Y, calculate the 95% confidence interval for mean of X+Y (this can be done empirically). (2 pts)

Your answer here

```
xplusy=sample_x+sample_y
SE_xplusy = sd(xplusy)/sqrt(length(xplusy))
soln = qt(0.975,(length(xplusy)-1))
mean(xplusy)- (soln * SE_xplusy)
```

```
## [1] 14.99504
```

```
mean(xplusy)+(soln * SE_xplusy)
```

```
## [1] 15.0089
```

the 95% confidence interval for mean of X+Y is (14.995,15.009)

(d)

In the above example, we have fixed the sample size n and number of experiments B.

Next, we want to change B and n, and see how confidence interval will change.

Please write a function to create a confidence interval for any B and n. (3 pts)

```
MC_CI <- function(n, B){

  ## your answer here

  new_x= matrix(nrow = B,ncol=n)
  set.seed(457)

  for(i in 1:B){
    new_x[i,]=rnorm(n,mean=10,sd=2)
  }

  new_y=matrix(nrow=B,ncol=n)
  set.seed(457)
  for(i in 1:B){
    new_y[i,]=rnorm(n,mean=5,sd=3)
  }

  new_xplusy=new_x+new_y
  SE_new_xplusy = sd(new_xplusy)/sqrt(length(new_xplusy))
  new_soln = qt(0.975,(length(new_xplusy)-1))
  lower_bound=mean(new_xplusy)- (new_soln * SE_new_xplusy)
  upper_bound=mean(new_xplusy)+(new_soln * SE_new_xplusy)

  return(c(lower_bound,upper_bound))
}
```

(e)

Suppose the sample size n varies (100, 200, 300, ... , 1000) (fix B=2000) and

the number of experiments B varies (1000, 2000, ... , 10000) (fix n=500).

Plot your confidence intervals to compare the effect of changing the sample size n and

changing the number of simulation replications B (2 plots).

What do you conclude? (4 pts)

(Hint: Check function errbar() in Hmisc package for plot)

```
#install.packages("Hmisc")
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 3.4.4
## Loading required package: lattice
## Warning: package 'lattice' was built under R version 3.4.4
## Loading required package: survival
## Loading required package: Formula
## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 3.4.4
##
## Attaching package: 'Hmisc'
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

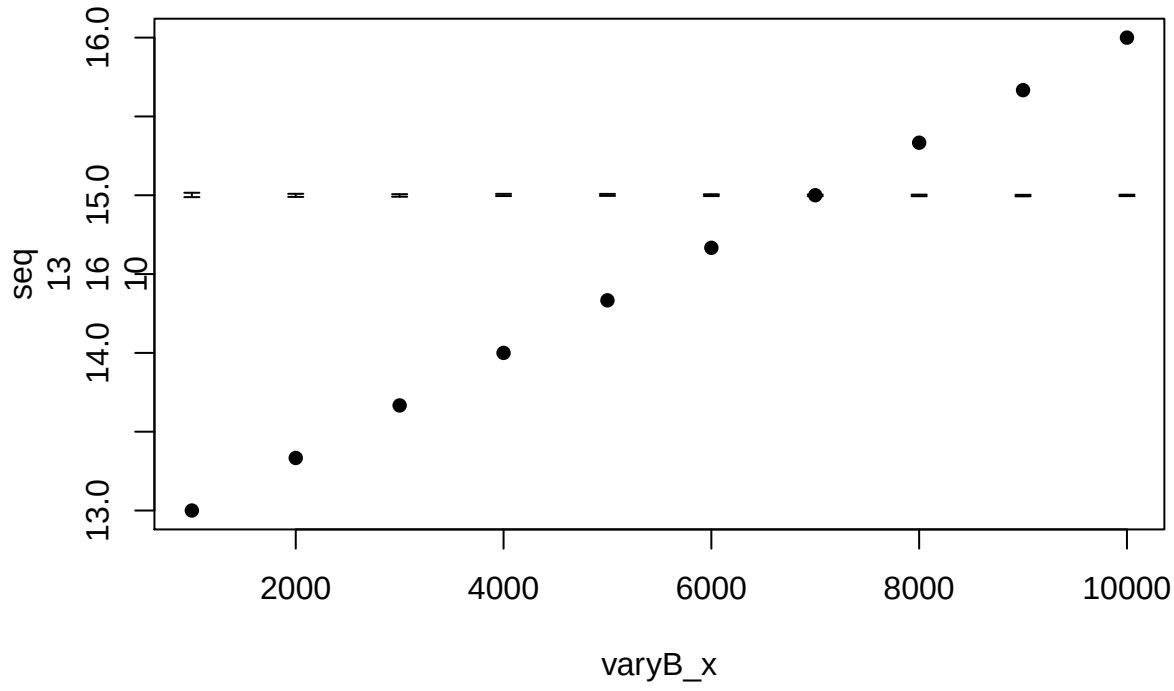
```
# fix n, B vary
varyB_x =c(1000,2000,3000,4000,5000,6000,7000,8000,9000,10000)

CI_vary_b1=MC_CI(500,1000)
CI_vary_b2=MC_CI(500,2000)
CI_vary_b3=MC_CI(500,3000)
CI_vary_b4=MC_CI(500,4000)
CI_vary_b5=MC_CI(500,5000)
CI_vary_b6=MC_CI(500,6000)
CI_vary_b7=MC_CI(500,7000)
```

```
CI_vary_b8=MC_CI(500,8000)
CI_vary_b9=MC_CI(500,9000)
CI_vary_b10=MC_CI(500,10000)

lower_bound_varyB=rbind(CI_vary_b1[1],CI_vary_b2[1],CI_vary_b3[1],CI_vary_b4[1],CI_vary_b5[1],
                        CI_vary_b6[1],
              CI_vary_b7[1],CI_vary_b8[1],CI_vary_b9[1],CI_vary_b10[1])
upper_bound_varyB=rbind(CI_vary_b1[2],CI_vary_b2[2],CI_vary_b3[2],CI_vary_b4[2],CI_vary_b5[2],
                        CI_vary_b6[2],
                    CI_vary_b7[2],CI_vary_b8[2],CI_vary_b9[2],CI_vary_b10[2])


errbar(x=varyB_x,y=seq(13,16,length.out = 10), yplus=upper_bound_varyB, yminus=lower_bound_varyB,
       main ="PLOT OF CONFIDENCE INTERVAL")
```

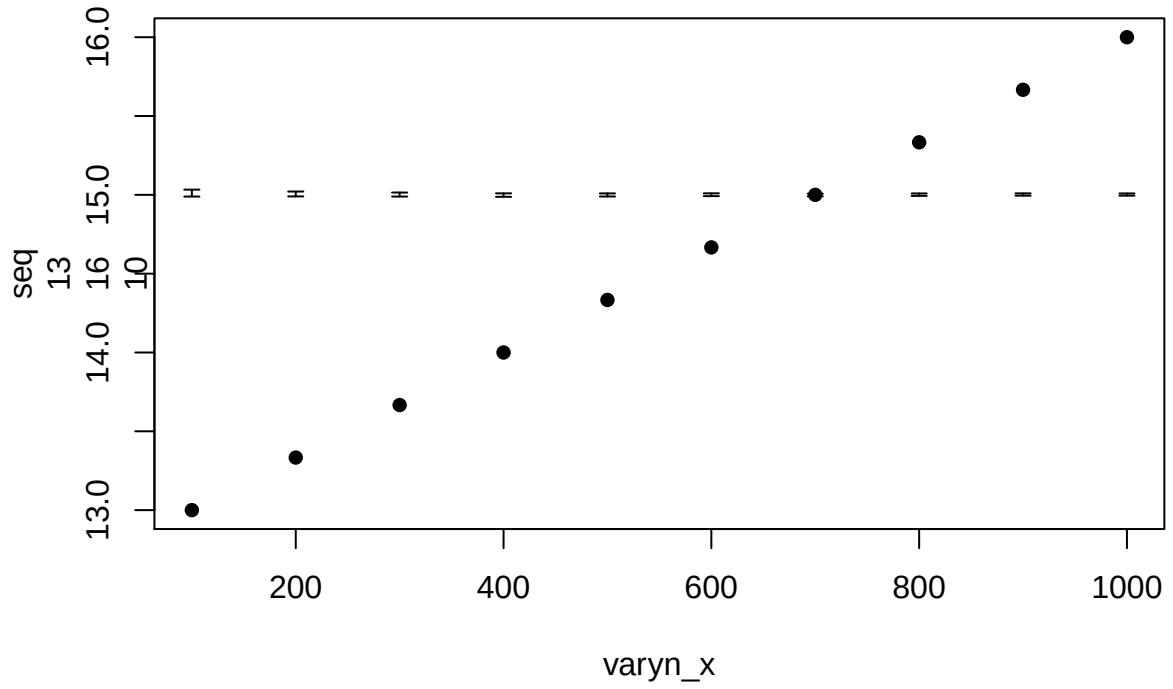



```
# fix B, n vary
varyn_x =c(100,200,300,400,500,600,700,800,900,1000)

CI_vary_n1=MC_CI(100,2000)
CI_vary_n2=MC_CI(200,2000)
CI_vary_n3=MC_CI(300,2000)
CI_vary_n4=MC_CI(400,2000)
CI_vary_n5=MC_CI(500,2000)
CI_vary_n6=MC_CI(600,2000)
CI_vary_n7=MC_CI(700,2000)
CI_vary_n8=MC_CI(800,2000)
```

```r
CI_vary_n9=MC_CI(900,2000)
CI_vary_n10=MC_CI(1000,2000)
## Your answer here
lower_bound_varyn=rbind(CI_vary_n1[1],CI_vary_n2[1],CI_vary_n3[1],CI_vary_n4[1],CI_vary_n5[1],
                        CI_vary_n6[1],
                        CI_vary_n7[1],CI_vary_n8[1],CI_vary_n9[1],CI_vary_n10[1])
upper_bound_varyn=rbind(CI_vary_n1[2],CI_vary_n2[2],CI_vary_n3[2],CI_vary_n4[2],CI_vary_n5[2],
                        CI_vary_n6[2],
                        CI_vary_n7[2],CI_vary_n8[2],CI_vary_n9[2],CI_vary_n10[2])


errbar(x=varyn_x,y=seq(13,16,length.out = 10), yplus=upper_bound_varyn, yminus=lower_bound_varyn
       ,main="PLOT OF CONFIDENCE INTERVAL")
```



Your answer here

changing the sample size n and changing the number of simulation replications B

has no effect on the confidence interval plots.There is a 95% confidence interval that the mean of

x+y lies between 14.9 and 15

Part 2: Regular Expressions

(a)

Use grep() to find which words in test set match the following requirements.

(return locations of the matched words in test set)

(1)

Words start with 's' (1 pt)

```
test_1 = c("introduction", "to", "data", "science", "457", "stats","snack")
```

Your answer here

```
grep("^[s]",test_1)
```

```
## [1] 4 6 7
```

(2)

A string that would be our netID (lower case letters + number) (1 pt)

```
test_2 = c("mxian111", "ANSA111", "Jeffery", "199", "Jeff333", "linz22",
           "wod123")
```

Your answer here

```
grep("^[[:lower:]]+[[:digit:]]+",test_2)
```

```
## [1] 1 6 7
```

(3)

An email address in form of example@example.xxx (1 pt)

```
test_3 = c("aaa@illinois.edu", "nothing@nothing", "bbb@gmail.com",
           "maybe@now.net","data @ gmail.com", "ee123@illinois.edu",
           "abc_123@gmail.com","abc_123@gmail.com.net")
```

Your answer here

```
grep("[[:alnum:][:punct:]]+[@][[:lower:]]+[.][[:lower:]]{3}$",test_3)
```

```
## [1] 1 3 4 6 7
```

(4)

An email address that ends with .com or .edu (1 pt)

```
test_4 = c("aaa@illinois.edu", "nothing@nothing", "bbb@gmail.com",
           "maybe@now.net","data @ gmail.com", "ee123@illinois.edu",
           "abc_123@gmail.com","abc_123@gmail.com.net")
```

Your answer here

```
grep("[[:alnum:][:punct:]]+[@][[:lower:]]+[.][com|edu]{3}$",test_4)
```

```
## [1] 1 3 6 7
```

(b)

Carry out the following exercises on the Womens Clothing E-Commerce Reviews dataset

(available in moodle, Womens Clothing E-Commerce Reviews.txt).

(1)

Use readLines() to read text data (it is in moodle) (encoding = "UTF-8") (1 pt)

Your answer here

```
womens_clothing=readLines(con=file("C:/Users/Womens Clothing E-Commerce Reviews.txt",open="r"))
#close(con)
```

(2)

Use regular expressions to return the number of reviews in the dataset (1 pt)

(Hint: find the pattern for each review)

Your answer here

```
length(grep("[R][[:lower:]]+[[:blank:]][T]", womens_clothing))
```

```
## [1] 23486
```

(3)

extract information of "Title", "Review text", "Rating"

and "Department name", and save them into 4 vectors, named

"title", "review", "rating" and "department".

Please print the first 3 elements and the length of each vector.

(3 pts)

(hint: split each string by some patterns, for list manipulation

you may want to use funtions in apply family. and unname())

Your answer here

```
old_review=regmatches(womens_clothing,
                      regexpr("xt:[\t][[:punct:]]*[[:alnum:][:blank:][:punct:][:digit:]]+[\t][R]"
                              ,womens_clothing))
review = gsub("^xt:|R$","",old_review)

old_title=regmatches(womens_clothing,regexpr("le[:][\t][[:alpha:][:blank:][:punct:]]+[\t]Re"
                                             ,womens_clothing))
title=gsub("^le[:]|Re$","",old_title)

rating=regmatches(womens_clothing,regexpr("[\t][[:digit:]][\t]",womens_clothing))

old_department=regmatches(womens_clothing,regexpr("me[:][\t][[:alpha:]]+",womens_clothing))
department=gsub("^me|[[:punct:]]","",old_department)

review[1:3]
```

```
## [1] "\tAbsolutely wonderful - silky and sexy and comfortable\t"
## [2] "\t\"Love this dress!  it's sooo pretty.  i happened to find it in a store, and i'm glad i did b
## [3] "\t\"I had such high hopes for this dress and really wanted it to work for me. i initially ordere
```

```
title[1:3]
```

```
## [1] "\tSome major design flaws\t" "\tMy favorite buy!\t"
## [3] "\tFlattering shirt\t"
```

```
rating[1:3]
```

```
## [1] "\t4\t" "\t5\t" "\t3\t"
```

```
department[1:3]
```

```
## [1] "\tIntimate" "\tDresses"  "\tDresses"
```

```
length(review)
```

```
## [1] 20591
```

```
length(title)
```

```
## [1] 19569
```

```
length(rating)
```

```
## [1] 23486
```

```
length(department)
```

```
## [1] 23472
```

(4)

When checking the text in 'title', we find some additional symbols, for example the in:

“"Shimmer, surprisingly goes with lots"”

“"Nice, but not for my body"”

use regular expressions to find all the titles which have that pattern of “"xxxxx"”

print out the first three incorrect titles and the total number of incorrect titles.

(2pts)

Your answer here

```
incorrect_title=regmatches(title,
                           regexpr("[\t][[:punct:]]{1}[[:alpha:][:punct:][:blank:]]+[[:punct:]]+"
                                   ,title))
incorrect_title[1:3]
```

```
## [1] "\t\"Shimmer, surprisingly goes with lots\""
## [2] "\t\"Nice, but not for my body\""
## [3] "\t\"You need to be at least average height, or taller\""
```

```
length(incorrect_title)
```

```
## [1] 1937
```

(5)

Change the format of all the incorrect titles, follow this rule:

“"xxxxxxx"” —-> “xxxxxxx” (2 pts)

for example:

“"Shimmer, surprisingly goes with lots"”

—-> “Shimmer, surprisingly goes with lots”

Your answer here

```
new_incorrect_title=gsub( "^[\t][[:punct:]]|[[:punct:]]$", "", incorrect_title)
new_incorrect_title[1]
```

```
## [1] "Shimmer, surprisingly goes with lots"
```

(6)

Let’s look at the reviews, and do some sentimental analysis.

We want to compare the reviews between Dresses and Tops department.

So first select reviews from these two department and save them as

two vectors review_dress and review_top. (2 pts)

(Hint: there are 5695 reviews in dresses department and 9656 reviews in tops department)

Your answer here

```
new_review=review
new_department=department

max.len = max(length(new_review), length(new_department))
new_review = c(new_review, rep(NA, max.len - length(new_review)))
new_department = c(new_department, rep(NA, max.len - length(new_department)))

my_df=data.frame(new_review,new_department)
my_df[] <- lapply(my_df, as.character)

review_dress= my_df$new_review[my_df$new_department=="\tDresses"]
review_dress=na.omit(review_dress)


review_top = my_df$new_review[my_df$new_department=="\tTops"]
review_top= na.omit(review_top)
```

(7)

Eliminate punctuations, numbers, and change all characters into lowercase

for review_dress and review_top. (3 pts)

For example:

"Finally a dress that is not too short! i'm 5'11 and ordered two sizes."

—-> "finally a dress that is not too short im and ordered two sizes"

Your answer here

```
review_dress=gsub("[[:punct:]]|[[:digit:]]|[\t]","",review_dress)
review_dress=tolower(review_dress)

review_top=gsub("[[:punct:]]|[[:digit:]]|[\t]","",review_top)
review_top=tolower(review_top)
```

(8)

Split the reviews that you have cleaned in Q(7) by blanks and drop any empty words.

Save all the split words into one list for each department,

named them as token_top and token_dress.

print the length of token_top and token_dress. (3 pts)

Your answer here

```
token_dress=strsplit(review_dress, "[[:blank:]]+")
token_top=strsplit(review_top, "[[:blank:]]+")

length(token_dress)
```

```
## [1] 5487
```

```
length(token_top)
```

```
## [1] 9240
```

(9)

Based on results in Q(8), calculate the token frequency for each token in each department.(2 pts)

Your answer here

```
library(plyr)
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:Hmisc':
##
##     is.discrete, summarize
```

```
token_dress_freq=count(unlist(token_dress))
token_top_freq = count(unlist(token_top))

head(token_dress_freq)
```

```
##                                  x  freq
## 1                                    3
## 2                                a 10013
## 3                               aa     3
## 4 aaaaandidontwanttopayforshipping     1
## 5                           aaahed     1
## 6                               ab     1
```

```
head(token_top_freq)
```

```
##               x  freq
## 1                 10
## 2             a 16979
## 3            aa    10
## 4 aaaaaaamazing     1
## 5  aaaaannnnnnd     1
## 6        aaaahs     1
```

(10)

Carry out some exploratory analysis of the data and term frequencies.

For example, find the words associated with positive and negative reviews. What are distribution patterns for the term frequencies?

Plot and interpret your result. What are your observations? (4 pts)

Your answer here

Some popular words in token_dress associated with positive reviews are love,

great,like,perfect,soft,flattering,cute,comfortable,beautfiul,nice,pretty,good,

perfectly

Some popular words in token_dress associted with negative reviews are return,

unfortunately,disappointed,returned,returning,never,worried

Some popular words in token_top associted with positive reviews are love,perfect,

flattering

,soft,comfortable,cute,beautiful,nice

perfectly,gorgeous,lovely,recommend,compliments

some popular words in token_top associted with negative reviews are return,unfortunately

,disappointed,returned,returning,sadly

unflattering

```r
max(token_dress_freq$freq)
```

```
## [1] 17592
```

```r
max(token_top_freq$freq)
```

```
## [1] 30051
```