HomeWork 1

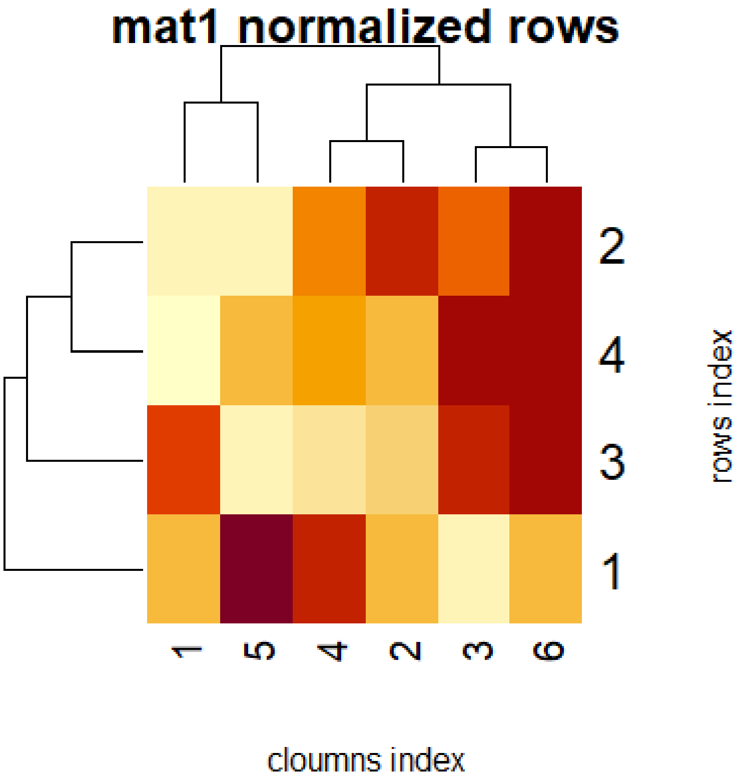
Adi Falach and Assaf Lovton

4/21/2021

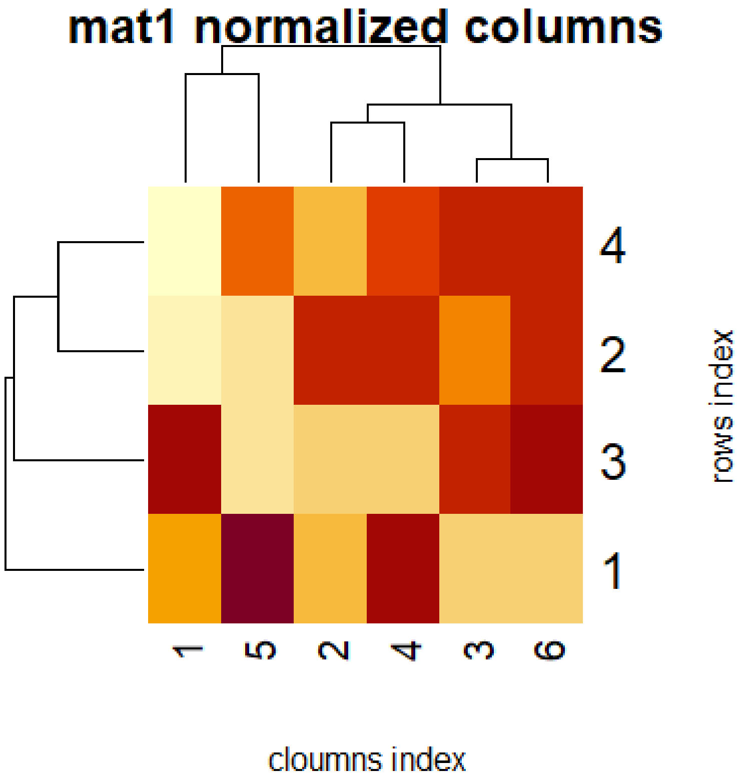
Question 1:

```
#1.1 mat1:
mat1<-matrix(runif(n=24, min = 1, max=20), nrow = 4, ncol = 6)
mat1

##           [,1]      [,2]     [,3]      [,4]      [,5]      [,6]
## [1,]  7.405404  7.160817  5.84026  9.532885 10.685010  7.191468
## [2,]  2.706162 16.812753 12.78590 10.966565  2.428084 17.973565
## [3,] 15.370687  6.233613 16.50371  4.953124  3.674777 18.912985
## [4,]  1.154070  9.003355 17.64817 10.330125  8.304781 18.459290

#1.2 The sum of each Row:
result<-apply(mat1, 1, sum)
result

## [1] 47.81584 63.67303 65.64889 64.89979

#1.3 The tentative.normalization() function:
tentative.normalization<- function(mat, npar) {
  max_vec<-apply(mat, npar, max)
  sweep(mat, npar, max_vec, FUN = '/')
}

#1.4 Calling for rows and columns:
mat1.norm.rows<-tentative.normalization(mat1,1)
mat1.norm.rows

##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.69306480 0.6701741 0.5465844 0.8921737 1.0000000 0.6730427
## [2,] 0.15056346 0.9354156 0.7113723 0.6101497 0.1350920 1.0000000
## [3,] 0.81270553 0.3295943 0.8726125 0.2618901 0.1942992 1.0000000
## [4,] 0.06251977 0.4877412 0.9560591 0.5596166 0.4498971 1.0000000

mat1.norm.columns<-tentative.normalization(mat1,2)
mat1.norm.columns

##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.48178745 0.4259158 0.3309272 0.8692681 1.0000000 0.3802397
## [2,] 0.17605993 1.0000000 0.7244884 1.0000000 0.2272421 0.9503294
## [3,] 1.00000000 0.3707669 0.9351510 0.4516568 0.3439189 1.0000000
## [4,] 0.07508256 0.5355075 1.0000000 0.9419654 0.7772366 0.9760115
```

```
#1.5 Hitmap for rows and columns:
print("Hitmap for normalized rows of mat1")

## [1] "Hitmap for normalized rows of mat1"

heatmap(mat1.norm.rows,main = "mat1 normalized rows", xlab = "cloumns index",
ylab="rows index")
```



```
print("Hitmap for normalized columns of mat1")

## [1] "Hitmap for normalized columns of mat1"

heatmap(mat1.norm.columns, main = "mat1 normalized columns", xlab = "cloumns
index", ylab="rows index")
```



## Question 2:

```
#2.1 Loading data and reading it into 2 variables:

data1.file <- file.path("date1.csv", fsep="/")
data2.file <- file.path("date2.csv", fsep="/")

data1<-read.csv("data1.csv")
data2<-read.csv("data2.csv")

#2.2 Finding out the type of data1:
class(data1)

## [1] "data.frame"

#As you can see the type is data frame

#2.3 Separating data2 into control and treatment groups, each group contains
Blood Sugar Levels:
control_data2<- dplyr::filter(data2, Treatment == "Placebo")
control_data2

##   Treatment Blood_Sugar_Levels
## 1   Placebo              21.51
## 2   Placebo              28.14
## 3   Placebo                 NA
## 4   Placebo              23.45
## 5   Placebo              23.68
## 6   Placebo              19.79
## 7    Placebo              28.40
## 8    Placebo              20.98
## 9    Placebo              22.51
## 10   Placebo              20.10
## 11   Placebo              26.91
## 12   Placebo              26.25
```

```
treatment_data2<- dplyr::filter(data2, Treatment == "HF123")
treatment_data2
```

```
##    Treatment Blood_Sugar_Levels
## 1      HF123              25.71
## 2      HF123              26.37
## 3      HF123              22.80
## 4      HF123              25.34
## 5      HF123              24.97
## 6      HF123              28.14
## 7      HF123              29.58
## 8      HF123              30.92
## 9      HF123              34.02
## 10     HF123              21.90
## 11     HF123              31.53
## 12     HF123              20.73
```

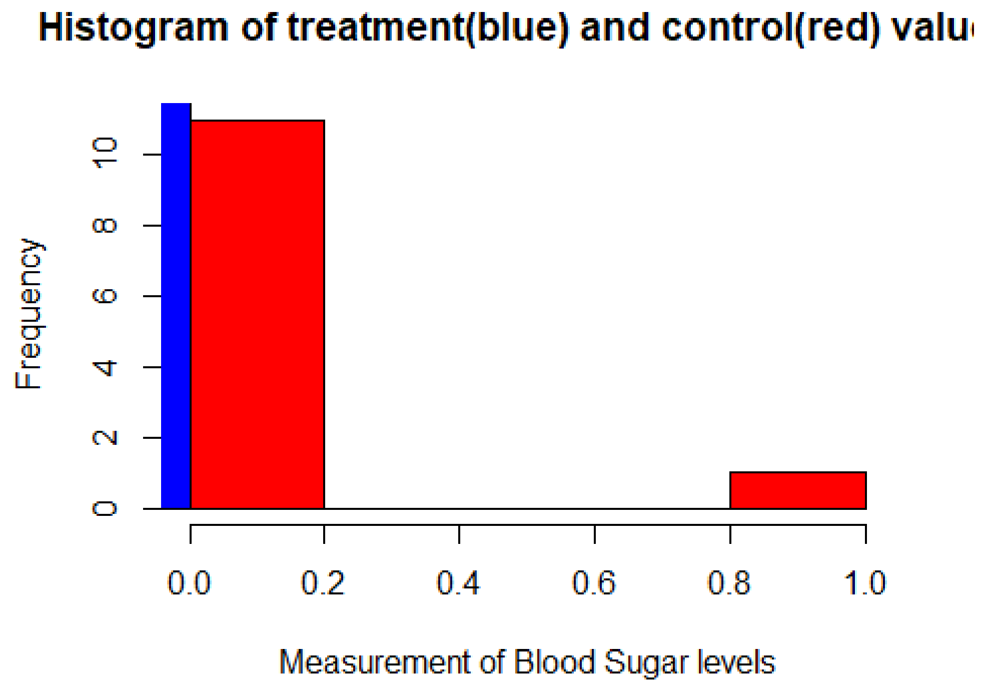
```
#2.4 Calculating the mean difference between the control and treatment
groups:
mean_diff <- mean(control_data2$Blood_Sugar_Levels, na.rm=TRUE)-
mean(treatment_data2$Blood_Sugar_Levels, na.rm=TRUE)
mean_diff#Since it is the difference between the means there is not much
insight from this value
```

```
## [1] -3.041439
```

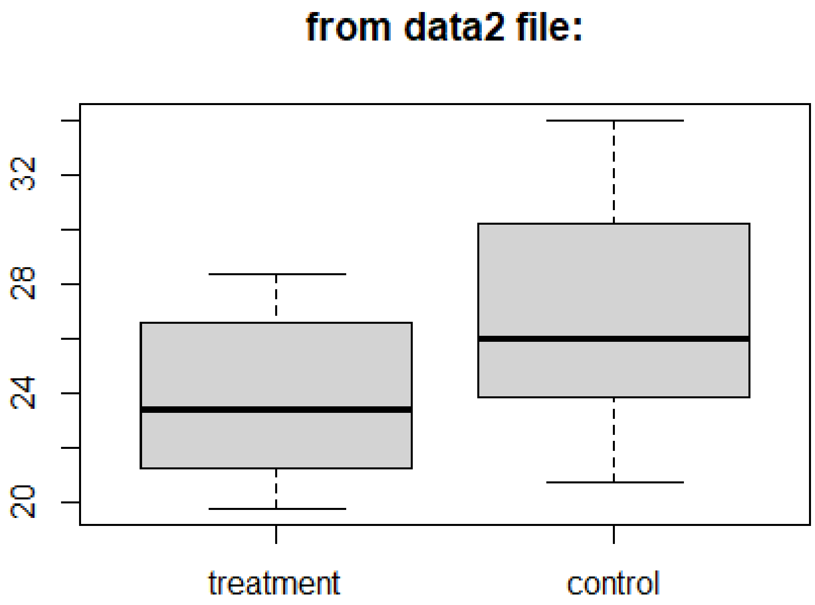
```
#we saw better ways to check our data reliability e.g. the p value
#we can calculate the approximate error but there is not much insight from
this...
approx_error<-
abs(mean_diff/mean(control_data2$Blood_Sugar_Levels,na.rm=TRUE))*100
approx_error #the approximated error in percents
```

```
## [1] 12.78306
```

```
#We can calculate the approximate error which is 12% but again there are
better ways to measure this.

#2.5 Showing histogram for both treatment and control groups:
hist(as.numeric(is.na(control_data2[2])),col = "red",
     main="Histogram of treatment(blue) and control(red) values",
     xlab="Measurement of Blood Sugar levels")
hist(as.numeric(is.na(treatment_data2[2])),col = "blue", add = T)
```

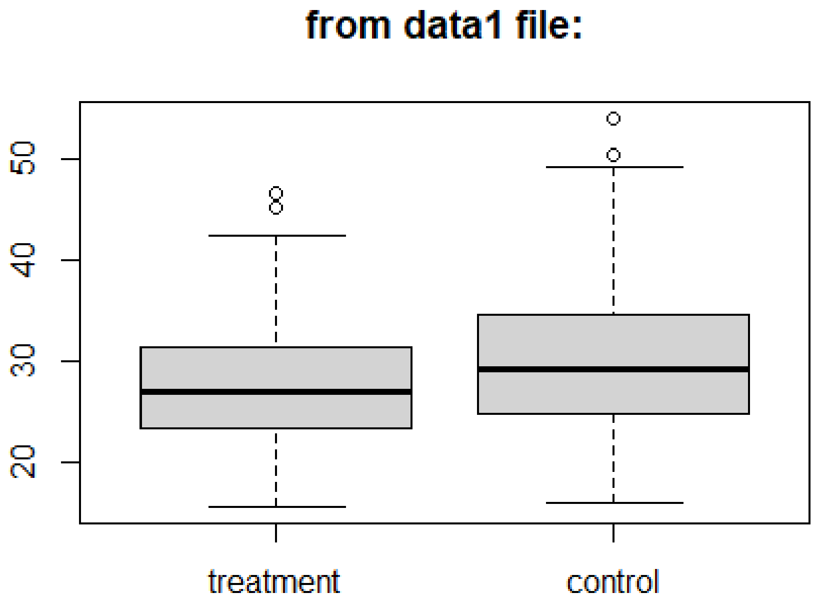


```
#2.6 Showing boxplot for both treatment and control groups:
study<-cbind(control_data2[2],treatment_data2[2])
colnames(study)<-c("treatment", "control")
boxplot(study, main = "from data2 file:")
```



```
#2.7 Preforming T test:
print(t.test(treatment_data2[2],control_data2[2]))

## 
##  Welch Two Sample t-test
## 
## data:  treatment_data2[2] and control_data2[2]
## t = 2.0001, df = 20.458, p-value = 0.05894
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1259596  6.2088384
## sample estimates:
## mean of x mean of y 
##  26.83417  23.79273

#the T test P value result is above 0.05 therefore there is a no significant
difference
#between the treatment group and the control group

#2.7 Showing boxplot for data1 and T test results:
control_data1<- dplyr::filter(data1, Treatment == "Placebo")
treatment_data1<- dplyr::filter(data1, Treatment == "HF123")
study1<-data.frame(cbind(control_data1[,3],treatment_data1[,3]))

## Warning in cbind(control_data1[, 3], treatment_data1[, 3]): number of rows
of
## result is not a multiple of vector length (arg 2)
```

```
colnames(study1)<-c("treatment", "control")
boxplot(study1, main = "from data1 file:")
```



```
#2.8 Preforming T test:
print(t.test(treatment_data1[,3],control_data1[,3]))

## 
##  Welch Two Sample t-test
## 
## data:  treatment_data1[, 3] and control_data1[, 3]
## t = 7.179, df = 727.79, p-value = 1.737e-12
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  2.232920 3.913876
## sample estimates:
## mean of x mean of y 
##  30.46951  27.39611

#the T test P value result is very close to zero (smaller than 0.05)
therefore there is a true difference. And there is a true difference between
the treatment and the control group.

#To sum up we will trust data1 more than data2 based on the p value results,
so the treatment in #data1 is providing better results.
```

Question 3:

```
suppressMessages(library("GenomicRanges"))

#3 Loading data and relevant libraries:
CLIP_data<-read.table(file.path("CLIP_data.BED"), header = TRUE)

#3.1 Creating GRange object that contains the data:
gr<- GRanges(seqnames = Rle(c(CLIP_data$chromosome)) ,
             ranges = IRanges (start=CLIP_data$start, end = CLIP_data$end),
             strand =CLIP_data$strand,
             names = CLIP_data$name,
             score = CLIP_data$score)
gr

## GRanges object with 44 ranges and 2 metadata columns:
##        seqnames              ranges strand |       names     score
##           <Rle>           <IRanges>  <Rle> | <character> <numeric>
##     [1]     chr1       567412-567473      + |       G10.1  0.653213
##     [2]     chr1   10231530-10231560      + |      G100.1 -0.124939
##     [3]     chr1 161433585-161433627      + |     G1000.1  1.237894
##     [4]    chr16   69782164-69782223      - |    G10009.1  0.778151
##     [5]    chr16   69789483-69789507      - |    G10010.1  0.698970
##     ...      ...                 ...    ... .         ...       ...
##    [40]     chr1 161510012-161510135      + |     G1010.1  0.361096
##    [41]    chr16   87417585-87417708      - |    G10100.1  1.531156
##    [42]    chr16   87866215-87866241      - |    G10103.1  0.301030
##    [43]    chr16   87903029-87903088      - |    G10104.1  0.522879
##    [44]    chr16   87915229-87915289      - |    G10105.1  0.845098
##    -------
##    seqinfo: 2 sequences from an unspecified genome; no seqlengths

#3.2 Flanking regions of the end of each range (width-200):
flanked_gr <- flank(gr, width=200, start=FALSE, both=FALSE)
flanked_gr

## GRanges object with 44 ranges and 2 metadata columns:
##        seqnames              ranges strand |       names     score
##           <Rle>           <IRanges>  <Rle> | <character> <numeric>
##     [1]     chr1       567474-567673      + |       G10.1  0.653213
##     [2]     chr1   10231561-10231760      + |      G100.1 -0.124939
##     [3]     chr1 161433628-161433827      + |     G1000.1  1.237894
##     [4]    chr16   69781964-69782163      - |    G10009.1  0.778151
##     [5]    chr16   69789283-69789482      - |    G10010.1  0.698970
##     ...      ...                 ...    ... .         ...       ...
##    [40]     chr1 161510136-161510335      + |     G1010.1  0.361096
##    [41]    chr16   87417385-87417584      - |    G10100.1  1.531156
##    [42]    chr16   87866015-87866214      - |    G10103.1  0.301030
##    [43]    chr16   87902829-87903028      - |    G10104.1  0.522879
##    [44]    chr16   87915029-87915228      - |    G10105.1  0.845098
##   -------
##   seqinfo: 2 sequences from an unspecified genome; no seqlengths
```