MVE510 - Introduction to Bioinformatics
Computer Exercise 3
(Revised)

Yankun Xu (940630-0237)
yankun@student.chalmers.se

Duygu Ayalp (931006-C701)
ayalp@student.chalmers.se

Exercise 1

The counts matrix is representing the counts for 58,037 genes in 80 distinct samples (patients). The metadata contains further information of each patient in terms of age, sex and the state of health. There are 29 female and 51 male patients. 40 of the 80 patients have the Chron's disease. In the next step genes with low expression are filtered. Therefore, genes that have non-zero entries in less than 25% of total samples were removed. The number of genes left is 38558.

Exercise 2

In the next step the counts data was normalized and logarithmically transformed. Normalizing the data across samples is a critical step for data interpretation since it corrects systemic errors such as sequencing errors or G/C-bias. This enables a comparison of gene expression levels across different samples. The log-transformation of the data aims to achieve a normal distribution of the data. [1]

Exercise 3

The unnormalized counts show a higher variability among the samples as shown in figure 1. The distribution of the data is clearly skewed. Figure 2 shows the boxplots for normalized counts. Normalization of the counts data resulted in a more stabilized and normal distribution with lower/ adjusted variability of the counts across the samples. The medians of the genes in different samples are almost at the same level.

Figure 3 shows the scatter plot of sample 1 vs sample 41. We add a line y=x to make the plot more clear. We can see that the expression of sample 41 is higher than sample 1. The bands in left and bottom mean that the zero counts in sample1 have the same small values after taking logarithm with adding offset 1, for these genes, sample 41 doesn't have zero count, so

these scatters are shown as a band in the left. Based on the same principle, the bottom bands are associated with the zero counts in the sample 41 and non-zero counts in sample 1.

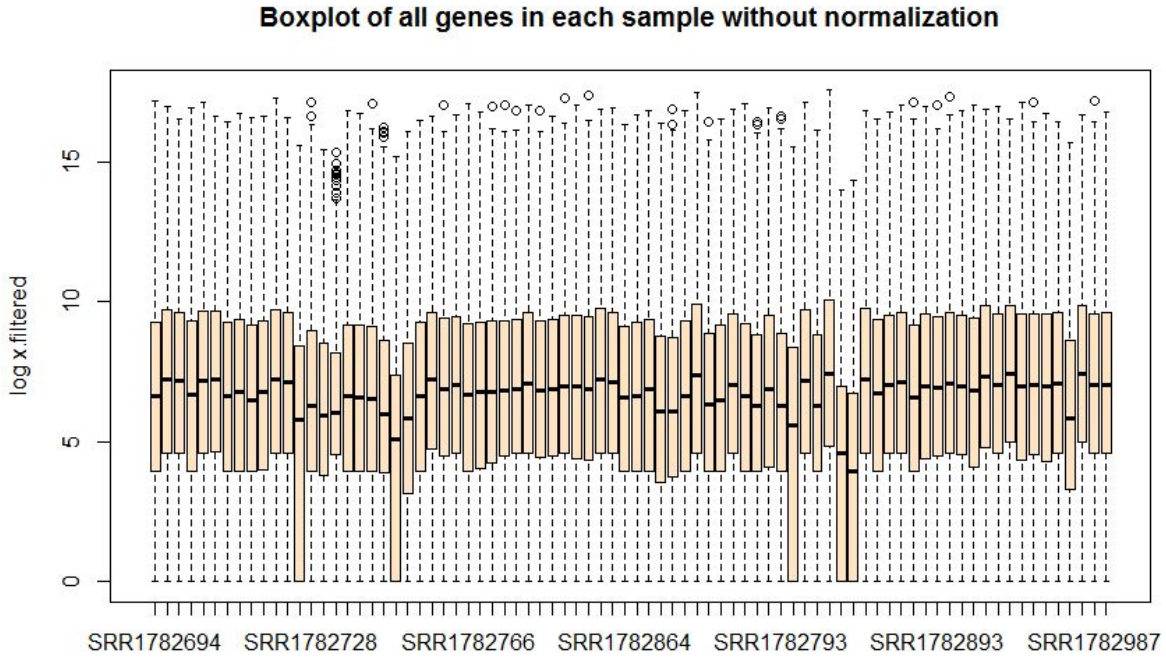

Figure 1: Unnormalized counts data

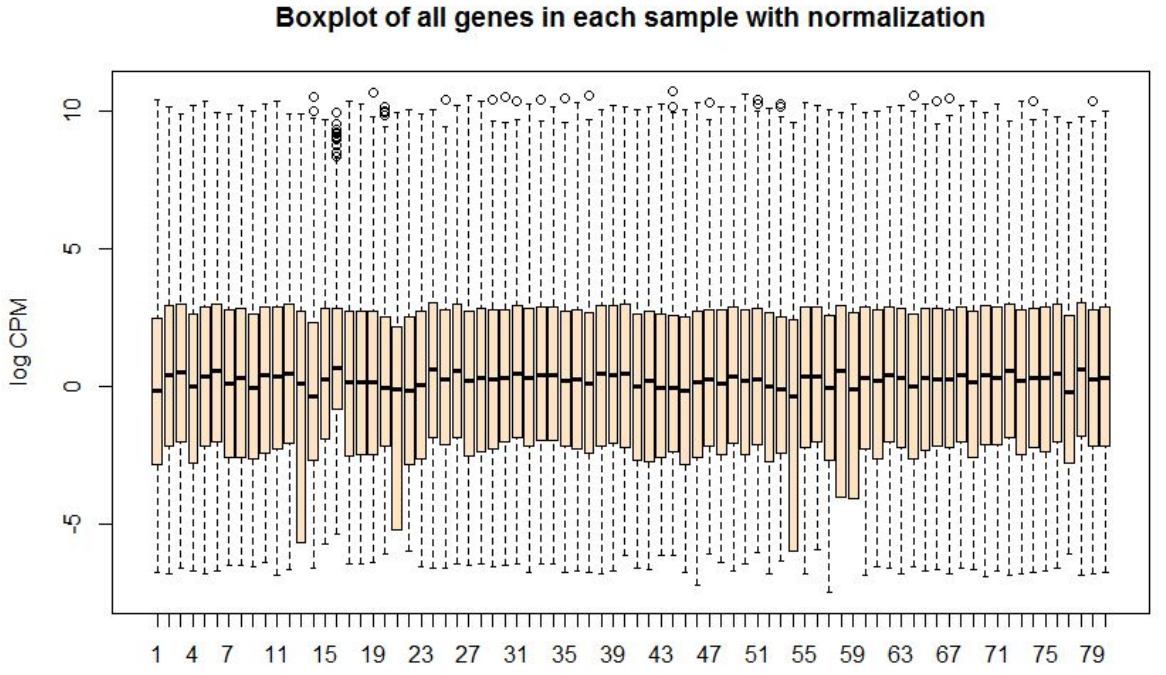

Figure 2: Normalized counts data

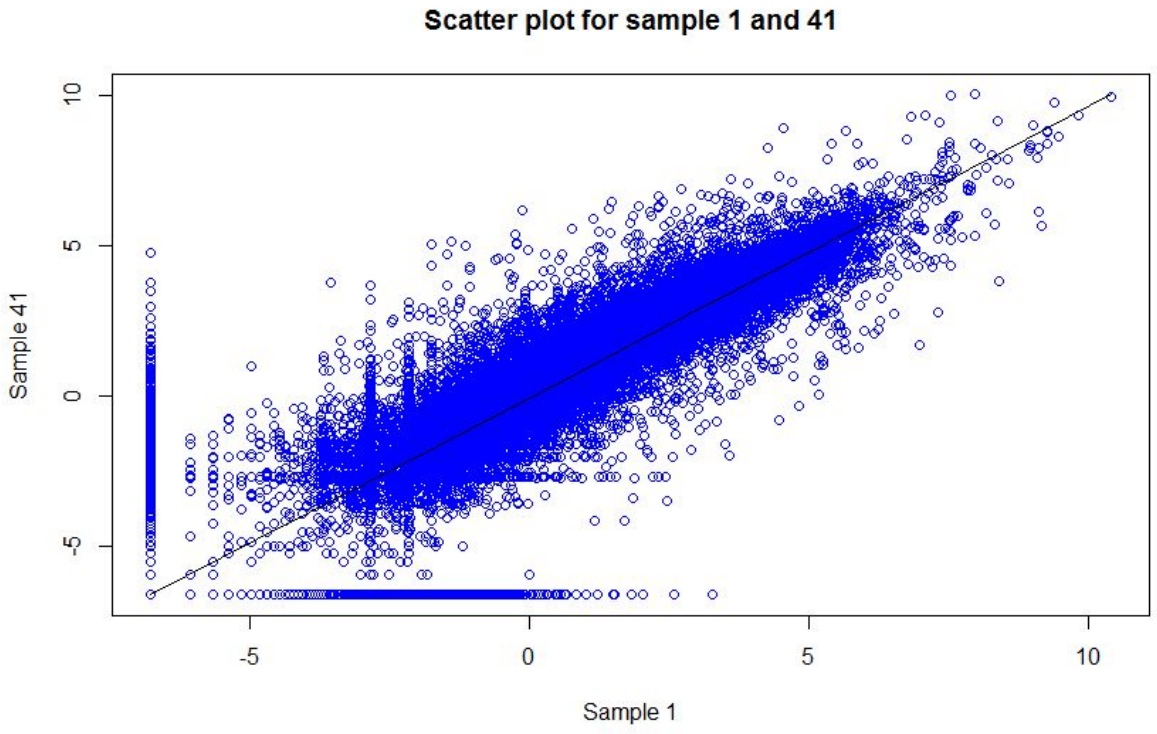

Figure 3. Scatter plot of sample 1 vs sample 41

Exercise 4

Figure 4 and 5 illustrates the gene expression level of gene 1 (ENSG00000000003) in both diagnosis groups (IBD: inflammatory bowel disease; CD: Chron's disease). Comparing the medians as well as the upper and lower quartiles of both boxplots it is evident that the gene 1 is higher expressed in the diseased group. For both groups outliers could be detected.

After performing linear modeling with the first model (fit1) which considers only the two disease groups (CD, not IBD). Using this model a p-value of 9.06e-08 for IBD variable is calculated. This indicates that gene1 is differentially expressed when comparing the disease groups. In the second model (fit2) further factors such as age and gender were taken into account. The p-value of IBD variable is 7.9e-08 and therefore gene1 is also differentially expressed in fit2. After investigating the summary of fit1 and fit2, we can see that the value of the Estimate column is positive, which means gene1 is up-regulated for IBD variable. The effect size is about 0.39 based on two models. The expression of this gene is not influenced by age or gender. Compared to fit1, fit2 consider the two more variables, age and gender, but these two variables have high p-value which means we don't need consider it. Furthermore, when we look at the value of Adjusted R-squared, fit2 has higher value than fit1, which means fit1 is better to describe the data than fit2.

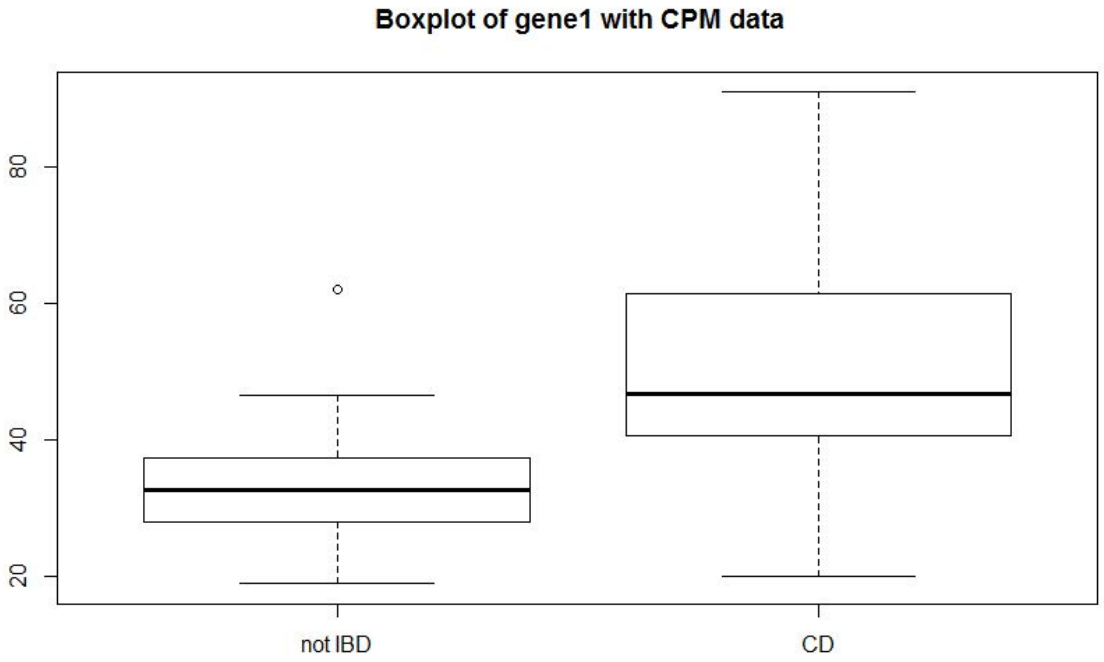

Figure 4. Boxplot of gene1 with CPM data

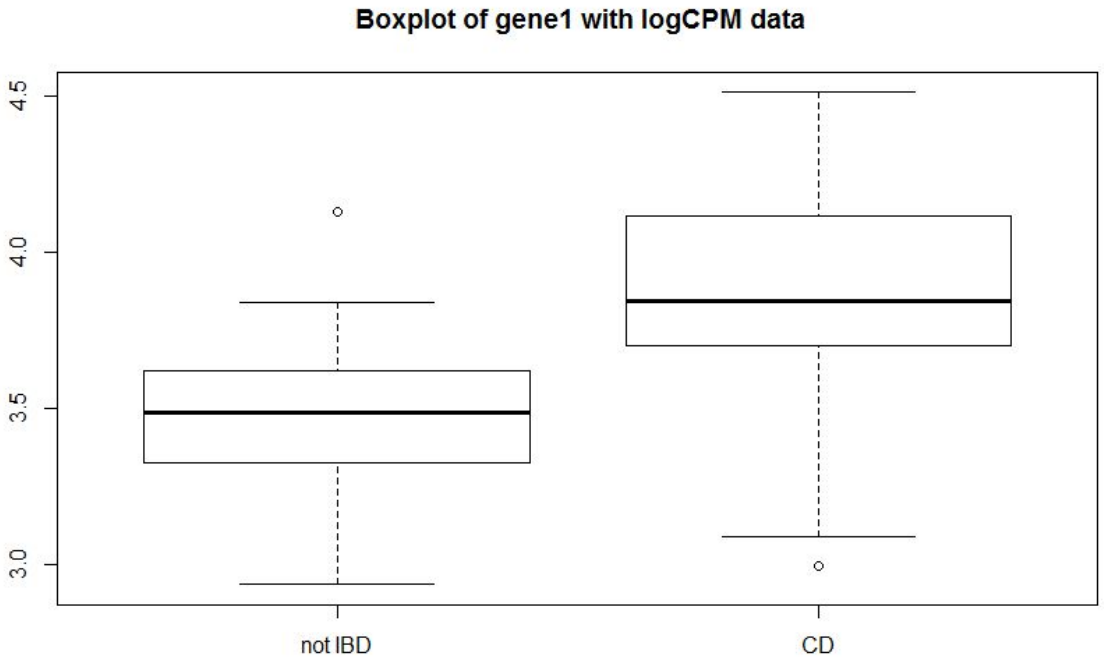

Figure 5. Boxplot of gene1 with logCPM data

Exercise 5

In the next step, both linear models (fit1, fit2) were used for all the genes . Using the first model 5702 genes show significant differential expression whereby 2499 genes are up-regulated and 3203 genes are down-regulated. The cutoff for the FDR was set as 0.05. With the second linear model, 5464 genes have shown differential expression. Comparing the diagnosis groups, 2338 genes were up-regulated and 3126 genes were down-regulated. By the way, all numbers is based on the comparison of disease to control.

The most significant gene was the *mucin 1* gene (ENSG00000185499) which is down-regulated in the disease group, and the log fold change is 0.73. The Muc 1 gene encodes for a glycosylated protein that is part of mucous barriers surfacing tissues such as stomach or pancreas. The expression of this protein is especially high on gastric mucosal surfaces and is associated with limitation of bacterial infection or inflammation. Therefore, Muc 1 might play a critical role in the gastrointestinal homeostasis. Regarding this, Sheng et al. (2013) [3] describes the influence of the differential expression of Muc 1 as follows: *"Inappropriate MUC1 or MUC13 expression may be responsible, at least in part, for progression from local inflammation to more severe gastrointestinal disease, such as in IBD and cancers; an hypothesis that is consistent with the altered expression profiles of mucins in gastrointestinal inflammatory and malignant diseases."* [2,3]

Revised top6 significant genes. "ENSG00000185499" "ENSG00000203747" "ENSG00000183010" "ENSG00000150337" "ENSG00000162747" "ENSG00000182240"

The 5 other most significant genes that are up-regulated and encoding for:

> *Fc fragment of IgG, low affinity IIIa, receptor (CD16a)*
> *Fc fragment of IgG, low affinity IIIb, receptor (CD16b)*
> *Fc fragment of IgG, high affinity Ia, receptor (CD64)*
> *Pyrroline-5-carboxylate reductase 1*
> *beta-site APP-cleaving enzyme 2 (BACE2)*

The first three proteins are Fc receptors that play a critical role in the immune response against pathogens. For instance CD16a and CD16b are expressed on natural killer cells and polymorphonuclear neutrophils, respectively. A change in the expression of these genes can be a contributing factor in Crohn's disease that is associated with an impaired immune response since they are affecting the susceptibility to pathogens or play a role in inflammatory diseases. Pyrroline-5-carboxylate reductase 1 plays a role in the reduction of pyrroline-5-carboxylate to L-proline and the NADP+-generation in some cells. Beta-site APP-cleaving enzyme 2 is an aspartic protease cleaving amyloid precursor protein into amyloid beta peptide. [4, 5, 6, 7, 8, 10]

In terms of variables age and gender, we calculate the p-values of age and gender again. We get none gene significantly associated with age (all adjusted p-value for age is 1) and 54 genes are found associated with gender (we consider all genes with adjusted p-values less than 1 as significant ones). The *eukaryotic translation initiation factor 1A* gene (ENSG00000198692) is found as the most significant for the gender factor because this gene has lowest adjusted p-value. This gene is located on the Y chromosome and plays a role in the binding of the initiator-tRNA (with Methionine) to the ribosome subunit. [8]

Exercise 6

Figure 6 shows the hierarchical clustering of the most 100 significant genes, the row is genes and the column is samples. We can see the clustering of column that the samples cluster very good enough, just few mistakes happened. By the way, The orange is for the group not IBD and purple represents IBD.

The clustering of genes also looks good enough. Firstly, there are few genes with negative values located in the bottom. The genes with values around zero are located in the lower middle area of the plot, and the middle part is the darkest part which means the genes located in here have highest value. And top part of the plot shows the grouped genes clearly as well.

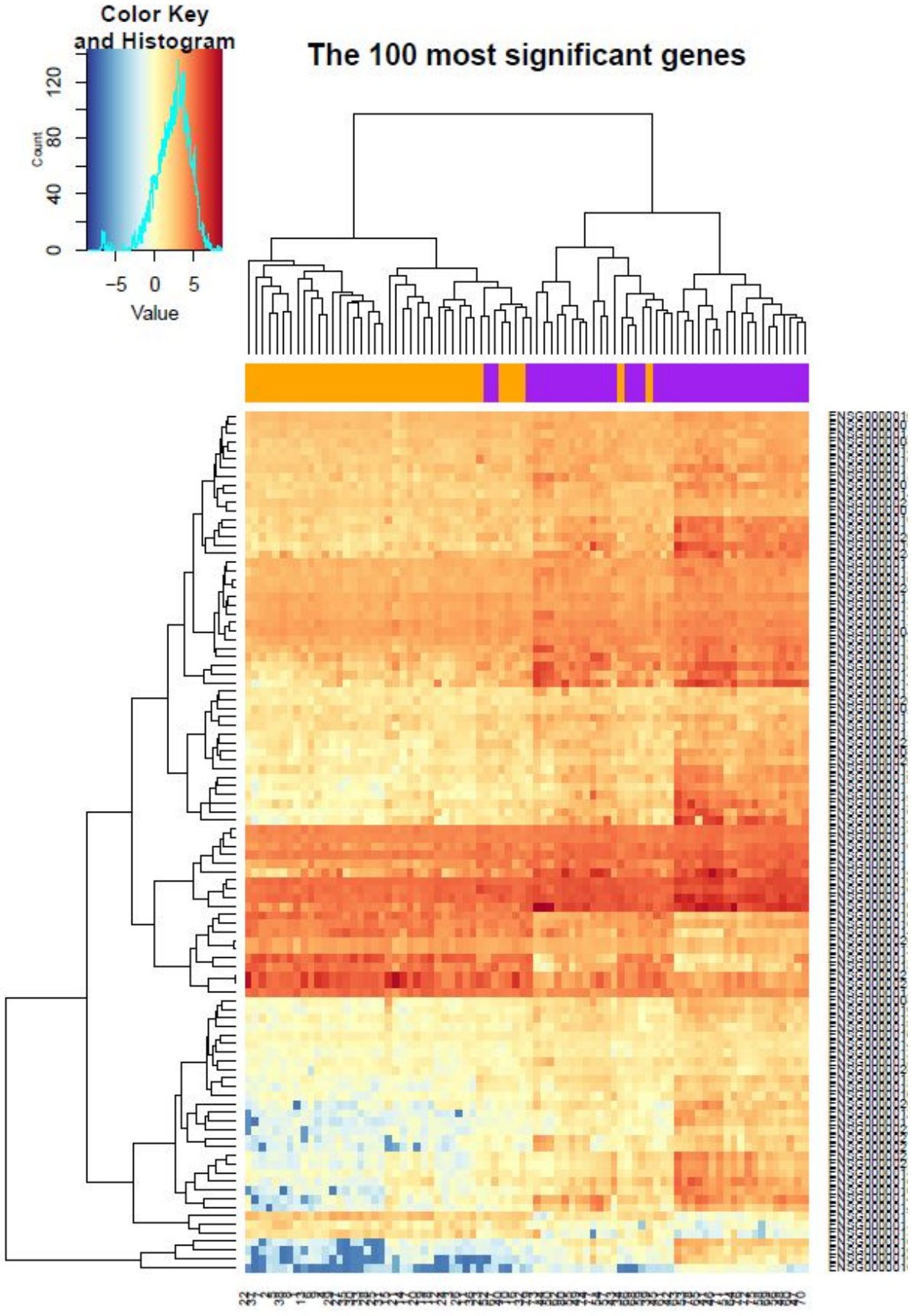

Figure 6. Hierarchical clustering of the most 100 significant genes

Exercise 7

When we do the principle component analysis(PCA) for our data, the standard deviation of the first two principal components is 82.7 and 59.2 respectively. The we take the scatter plot of PC1, PC2 and PC3 against each other to investigate the relationships among them. (The colour definition is same as exercise 6)

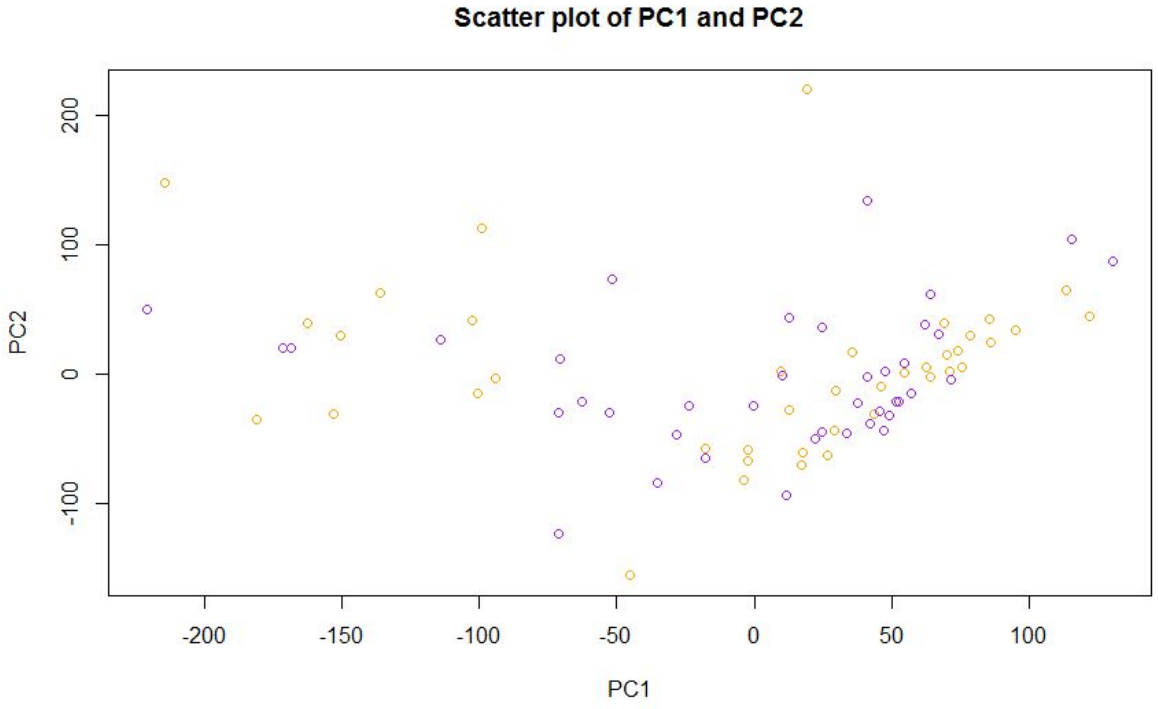

Figure 7. Scatter plot of PC1 vs PC2

We can see Figure 7 that the first two components are not sufficient to separate the Chron's disease samples from the controls even if we could see there is a little bit cluster among the data.

When we use PC1 against PC3 and PC1 against PC3 to represent the data, we can both see separation from the Figure 8, most orange cluster in the top part of figure and most purple cluster in the bottom part. However, the separation is not perfect, there are still some mixture data amone them. We think the reason is that we just take two principal components to represent data even if two is also good, and obviously we miss the maximal component. If we could use more large principal components, the better separation for the data we will achieve.

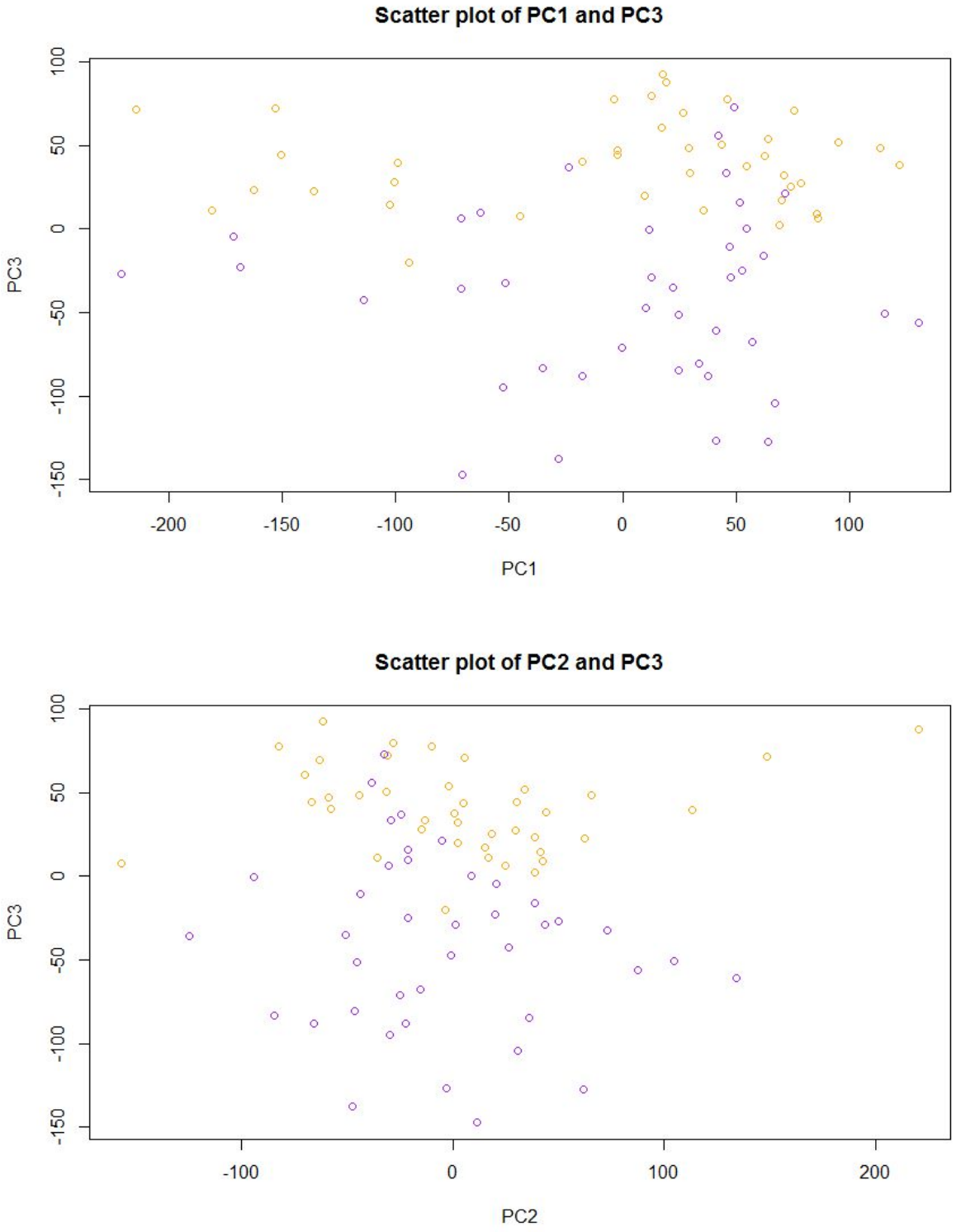

Figure 8. Scatter plot of PC1 vs PC3 and PC2 vs PC3

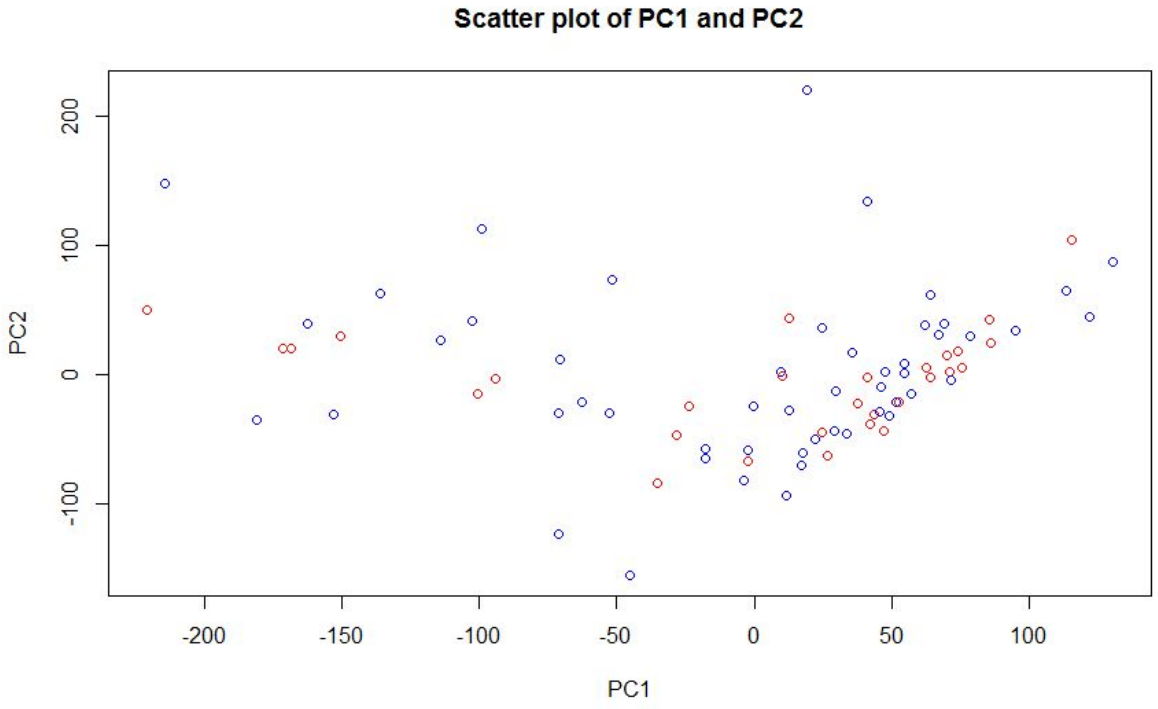

Figure 9. The scatter plot of PC1 and PC2 for gender group

Figure 9 shows the scatter plot of PC1 and PC2 when we colour the gender group. We cannot see any separation among the data, it is a totally mess! Actually, we have tried PC1 vs PC3 and PC2 vs PC3, the results are also messy!

References:

[1] https://www.ncbi.nlm.nih.gov/pubmed/25240000
[2] https://www.ncbi.nlm.nih.gov/gene?Db=gene&Cmd=DetailsSearch&Term=4582
[3] https://www.nature.com/articles/mi201298
[4] https://www.ncbi.nlm.nih.gov/gene/2214
[5] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4075408/
[6] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4209745/
[7] http://omim.org/entry/179035
[8] http://omim.org/entry/612772
[9] https://www.ncbi.nlm.nih.gov/gene?Db=gene&Cmd=DetailsSearch&Term=9086
[10] https://www.ncbi.nlm.nih.gov/gene/25825

Appendix: Code in R language

```
x = read.table('counts_matrix.txt')
metadata = read.table('metadata.txt',sep='\t',header=TRUE)

rownames(x) # gene
colnames(x) # 80 samples
metadata$patient.id # same as the last command
colnames(x) == metadata$patient.id # all same

# ex1
count.nonzero = function(x){
  x.length = nrow(x)
  counts = vector(length=x.length,mode="double")
  for (i in 1:x.length) {
    num = sum(x[i,]>0)
    counts[i]= num
  }
  return(counts)
}

counts = count.nonzero(x)

x.filtered = x[counts>=20,]  # 38558 left
nrow(x.filtered)/nrow(x) > 0.25 # Ture

# ex2
CPM = function(x){
  x = x+1
  x.col = ncol(x)
  x.row = nrow(x)
  cpm = matrix(0,x.row,x.col)
  for (i in 1:x.col) {
    sum.col = sum(x[,i])

    for (j in 1:x.row) {
      cpm[j,i] = (x[j,i]/sum.col) * 1000000
    }
  }
  return(cpm)
```

```
}

x.filtered.cpm = CPM(x.filtered)
x.filtered.logcpm = log(x.filtered.cpm)

# ex3

x.filtered.log = log(1 + x.filtered)

boxplot(x.filtered.logcpm,col = "bisque",ylab="log CPM" ,add = FALSE)
title("Boxplot of all genes in each sample with normalization")

boxplot(x.filtered.log, col = "bisque",ylab="log x.filtered" ,add = FALSE)
title("Boxplot of all genes in each sample without normalization")

plot(x.filtered.logcpm[,1], x.filtered.logcpm[,41], col="blue",
    main="Scatter plot for sample 1 and 41",
    xlab = "Sample 1", ylab = "Sample 41")
par(new=TRUE)
plot(1:10, 1:10,"l",axes = FALSE,xlab = "", ylab = "")

# ex4

gene1.cpm = t(x.filtered.cpm[1,])
gene1.logcpm = t(x.filtered.logcpm[1,])

boxplot(gene1.cpm[1:40],gene1.cpm[41:80],names = c("not IBD","CD"))
title(" Boxplot of gene1 with CPM data")
boxplot(gene1.logcpm[1:40], gene1.logcpm[41:80],names = c("not IBD","CD"))
title(" Boxplot of gene1 with logCPM data")

diagnosis = as.factor(metadata[,"diagnosis"])
diagnosis = relevel(diagnosis, ref = "Not IBD")
sex = as.factor(metadata[,"Sex"])
age = metadata[,"age.at.diagnosis"]

fit1 = lm(x.filtered.logcpm[1,]~diagnosis)
fit2 = lm(x.filtered.logcpm[1,]~age+sex+diagnosis)

# ex5

gene.length = nrow(x.filtered.logcpm)
```

```
fit1.pval = vector(length = gene.length,mode = "double")
fit1.coeff = vector(length = gene.length,mode = "double")
fit2.pval = vector(length = gene.length,mode = "double")
fit2.coeff = vector(length = gene.length,mode = "double")

for (i in 1:gene.length) {
  fit1.i = lm(x.filtered.logcpm[i,]~diagnosis)
  fit2.i = lm(x.filtered.logcpm[i,]~age+sex+diagnosis)
  fit1.pval[i]= summary(fit1.i)$coefficients["diagnosisCD","Pr(>|t|)"]
  fit1.coeff[i]= summary(fit1.i)$coefficients["diagnosisCD","Estimate"]
  fit2.pval[i]= summary(fit2.i)$coefficients["diagnosisCD","Pr(>|t|)"]
  fit2.coeff[i]= summary(fit2.i)$coefficients["diagnosisCD","Estimate"]
}

fit1.padjust = data.frame(cbind(p.adjust(fit1.pval, method = "fdr"), fit1.coeff))
fit2.padjust = data.frame(cbind(p.adjust(fit2.pval, method = "fdr"), fit2.coeff))

rownames(fit1.padjust) = c(rownames(x.filtered))
rownames(fit2.padjust) = c(rownames(x.filtered))

rownames(x.filtered.logcpm) = c(rownames(x.filtered))
rownames(x.filtered.cpm) = c(rownames(x.filtered))

colnames(fit1.padjust) = c("p-adjust","coefficients")
colnames(fit2.padjust) = c("p-adjust","coefficients")

cutoff = 0.05

fit1.diff.coeff= fit1.padjust[fit1.padjust[,1] < cutoff,2]
fit2.diff.coeff= fit2.padjust[fit2.padjust[,1] < cutoff,2]

num.fit1.diff = length(fit1.diff.coeff)
num.fit2.diff = length(fit2.diff.coeff)

fit1.coeff.up.num = sum(fit1.diff.coeff>0)
fit1.coeff.down.num = num.fit1.diff - fit1.coeff.up.num
fit2.coeff.up.num = sum(fit2.diff.coeff>0)
fit2.coeff.down.num = num.fit2.diff - sum(fit2.diff.coeff>0)

fit1.most.gene = rownames(fit1.padjust)[fit1.padjust[,1] == min(fit1.padjust[,1])]
fit2.most.gene = rownames(fit2.padjust)[fit2.padjust[,1] == min(fit2.padjust[,1])]
# most significant gene: "ENSG00000185499"
```

```
coeff.most.gene.fit1 = fit1.padjust["ENSG00000185499",2]
coeff.most.gene.fit2 = fit2.padjust["ENSG00000185499",2]
# almost same

# effect size (log fold-change)
log.fold.change = log(sum(x.filtered.logcpm["ENSG00000185499",41:80]) /
  sum(x.filtered.logcpm["ENSG00000185499",1:40]))
# 0.73
log.fold.change = log(sum(x.filtered.cpm["ENSG00000185499",41:80])/
  sum(x.filtered.cpm["ENSG00000185499",1:40]))
# 2.49

fit1.padjust.order=order(fit1.padjust[,1], decreasing=FALSE)
fit1.padjust.sorted = fit1.padjust[fit1.padjust.order,]
fit2.padjust.order=order(fit2.padjust[,1], decreasing=FALSE)
fit2.padjust.sorted = fit2.padjust[fit2.padjust.order,]

rownames(fit1.padjust.sorted)[1:6]
fit1.padjust.sorted[1:6,2] > 0
# gene
# "ENSG00000185499" "ENSG00000203747" "ENSG00000183010" "ENSG00000150337"
"ENSG00000162747" "ENSG00000182240"
# coefficients (up or down - regulate)
# TRUE TRUE TRUE TRUE TRUE TRUE
rownames(fit2.padjust.sorted)[1:6]
fit2.padjust.sorted[1:6,2] > 0
# "ENSG00000185499" "ENSG00000203747" "ENSG00000183010" "ENSG00000162747"
"ENSG00000140274" "ENSG00000150337"
# "ENSG00000185499" "ENSG00000203747" "ENSG00000183010" "ENSG00000150337"
"ENSG00000162747" "ENSG00000182240"
# TRUE TRUE TRUE TRUE TRUE TRUE


fit2.age.pval = vector(length = gene.length,mode = "double")
fit2.sex.pval = vector(length = gene.length,mode = "double")

for (i in 1:gene.length) {
  fit2.i = lm(x.filtered.logcpm[i,]~age+sex+diagnosis)
  fit2.age.pval[i]= summary(fit2.i)$coefficients["age","Pr(>|t|)"]
  fit2.sex.pval[i] = summary(fit2.i)$coefficients["sexMale","Pr(>|t|)"]
}
```

```
fit2.age.padjust = p.adjust(fit2.age.pval)
# all pvalue adjust is 1. none is related to age

fit2.sex.padjust = p.adjust(fit2.sex.pval)
rownames(x.filtered.logcpm)[fit2.sex.padjust==min(fit2.sex.padjust)]
# "ENSG00000067048" most significant for gender


# ex6

xsig = x.filtered.logcpm[rownames(fit1.padjust.sorted)[1:100],]
xsig = as.matrix(xsig)

mypalette = brewer.pal(11,"RdYlBu")
morecols = colorRampPalette(mypalette)
mycols=rev(morecols(255))
column.cols=c("purple","orange")[metadata$diagnosis]
pdf("top100sigGenesHeatmap.pdf",height=9,width=6)
heatmap.2(xsig,trace='none',col=mycols,main='The 100 most significant
genes',ColSideColors=column.cols)
dev.off()
# orange is not IBD

# ex7

pca=prcomp(t(x.filtered.logcpm))

PC1 = pca$x[,1]
PC2 = pca$x[,2]
PC3 = pca$x[,3]

plot(PC1,PC2,main = "Scatter plot of PC1 and PC2", xlab = "PC1",
    ylab = "PC2", col=column.cols)

plot(PC1,PC3,main = "Scatter plot of PC1 and PC3", xlab = "PC1",
    ylab = "PC3", col=column.cols)

plot(PC2,PC3,main = "Scatter plot of PC2 and PC3", xlab = "PC2",
    ylab = "PC3", col=column.cols)

# based on gender
```

```
sex.cols=c("red","blue")[metadata$Sex]
# red for female, blue for male

plot(PC2,PC3,main = "Scatter plot of PC2 and PC3", xlab = "PC2",
    ylab = "PC3", col=sex.cols)

plot(PC1,PC3,main = "Scatter plot of PC1 and PC3", xlab = "PC1",
    ylab = "PC3", col=sex.cols)

plot(PC1,PC2,main = "Scatter plot of PC1 and PC2", xlab = "PC1",
    ylab = "PC2", col=sex.cols)
```