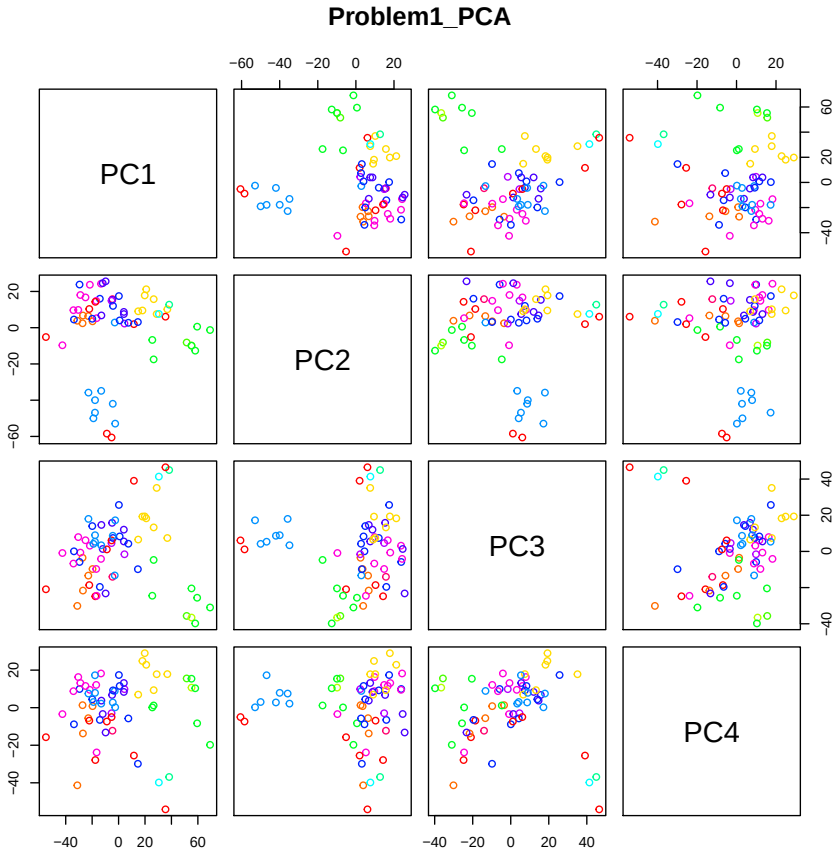
BINF702 Assignment 3

Samiksha Borkar

May 1, 2016

Question 1

```
library(knitr)
opts_chunk$set(fig.path='figure/minimal-', fig.align='centre', fig.show='hold')
options(formatR.arrow=TRUE,width=90)

library(ISLR)

## Warning:  package 'ISLR' was built under R version 3.2.5

data("NCI60")
nci.labs=NCI60$labs
nci.data=NCI60$data
pr.out =prcomp (nci.data , scale=TRUE)
Cols=function(vec){
cols = rainbow(length(unique(vec)))
return(cols[as.numeric(as.factor(vec))])
}
pairs(pr.out$x[,1:4], main = "Problem1_PCA",pch = 21, col= Cols(nci.labs))
```



The scatterplot matrix shows the projection of NCI60 cancer cell line on first 4 principal components. The score of first four principal component on a whole shows that cell lines from the same cancer type tend to have similar values hence their gene expression levels are quite similar.

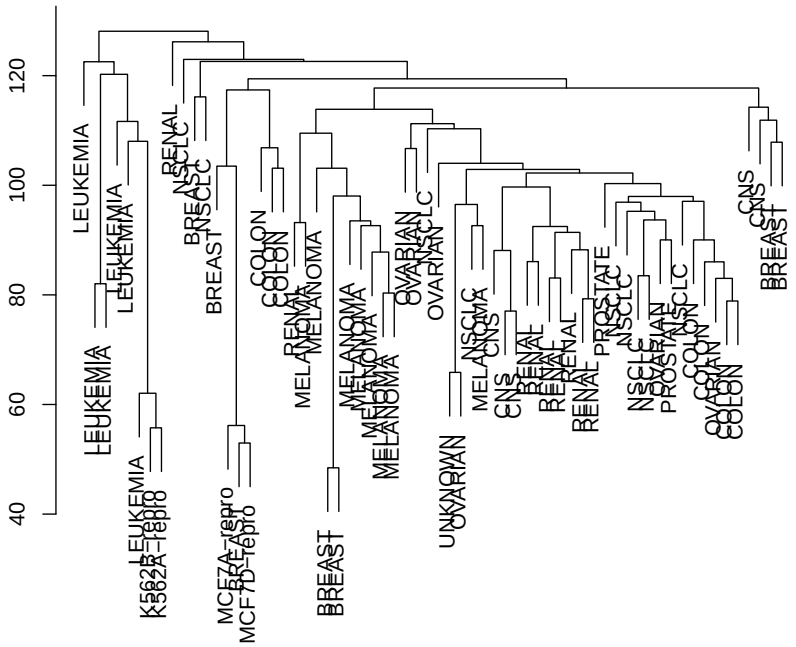
Question 2

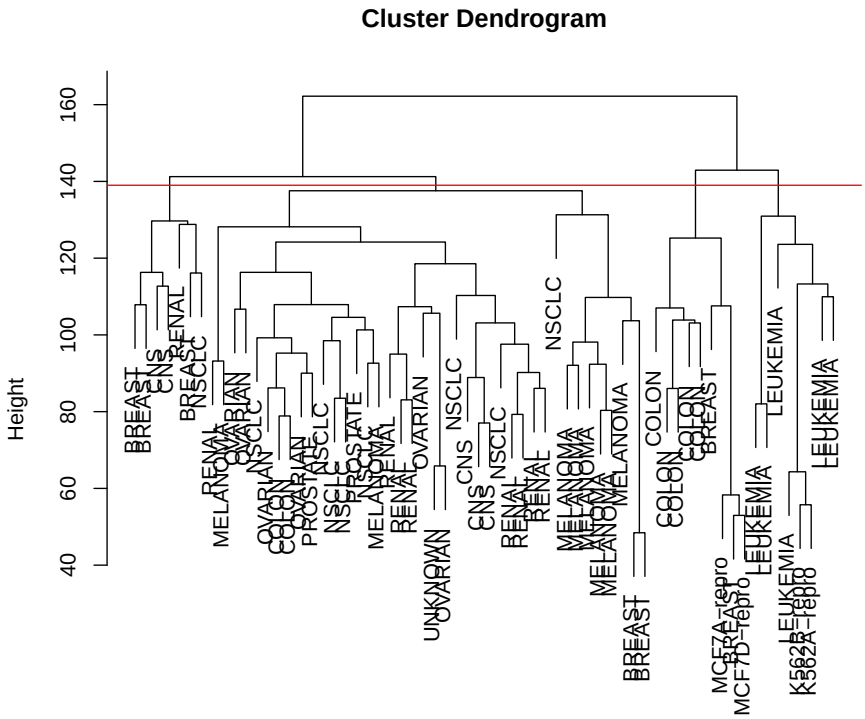
```
sd.data=scale(nci.data)
data.dist=dist(sd.data)
plot(hclust (data.dist , method ="average"),
labels =nci.labs ,main=" Average Linkage ", xlab ="", sub ="", ylab ="")
hc.out =hclust (dist(sd.data))
hc.clusters =cutree (hc.out ,4)
plot(hc.out , labels =nci.labs)
abline (h=139, col =" red ")
```

```
table(hc.clusters ,nci.labs)
```

```
##             nci.labs
## hc.clusters BREAST CNS COLON K562A-repro K562B-repro LEUKEMIA MCF7A-repro MCF7D-repro
##           1      2   3     2           0           0        0           0           0
##           2      3   2     0           0           0        0           0           0
##           3      0   0     0           1           1        6           0           0
##           4      2   0     5           0           0        0           1           1
##             nci.labs
## hc.clusters MELANOMA NSCLC OVARIAN PROSTATE RENAL UNKNOWN
##           1        8     8       6        2     8       1
##           2        0     1       0        0     1       0
##           3        0     0       0        0     0       0
##           4        0     0       0        0     0       0
```

Average Linkage





In problem 2, hierarchical clustering is performed on standardized variables using average linkage. From the dendogram it could be inferred that the average linkage shows clusters evenly distributed and cell lines within a single cancer type do tend to cluster together. After we cut the dendogram, we see some clear patterns, like all the leukemia cell lines falls in the single cluster on right most side and while the breast cancer cell lines are spread out over all other clusters.

Question 3

```
set.seed(0)
km.out =kmeans(sd.data ,4)
km.clusters =km.out$cluster
table(km.clusters, nci.labs)

##             nci.labs
```

```
## km.clusters BREAST CNS COLON K562A-repro K562B-repro LEUKEMIA MCF7A-repro MCF7D-repro
##           1      2   0     0           0           0        0           0           0
##           2      3   5     0           0           0        0           0           0
##           3      2   0     7           0           0        0           1           1
##           4      0   0     0           1           1        6           0           0
##            nci.labs
## km.clusters MELANOMA NSCLC OVARIAN PROSTATE RENAL UNKNOWN
##           1        7     0       0        0     0       0
##           2        1     4       3        1     9       1
##           3        0     5       3        1     0       0
##           4        0     0       0        0     0       0
```

```
set.seed(0)
km.out1 = kmeans(pr.out$x, 4)
km.clusters1 = km.out1$cluster
table(km.clusters1, nci.labs)
```

```
##             nci.labs
## km.clusters1 BREAST CNS COLON K562A-repro K562B-repro LEUKEMIA MCF7A-repro MCF7D-repro
##            1      2   0     0           0           0        0           0           0
##            2      3   5     0           0           0        0           0           0
##            3      2   0     7           0           0        0           1           1
##            4      0   0     0           1           1        6           0           0
##             nci.labs
## km.clusters1 MELANOMA NSCLC OVARIAN PROSTATE RENAL UNKNOWN
##            1        7     0       0        0     0       0
##            2        1     4       3        1     9       1
##            3        0     5       3        1     0       0
##            4        0     0       0        0     0       0
```

In the problem 3 we perform k-mean clustering on the same NCI data. From the clustering table we can see that the clustering from k-mean and hierarchical differs quite a bit like the leukemia cell line falls in cluster 3 in hierarchical but it falls in cluster 4 in kmean. However the kmean clustering analysis performed on the principal components shows similar clustering as that on the original variables.

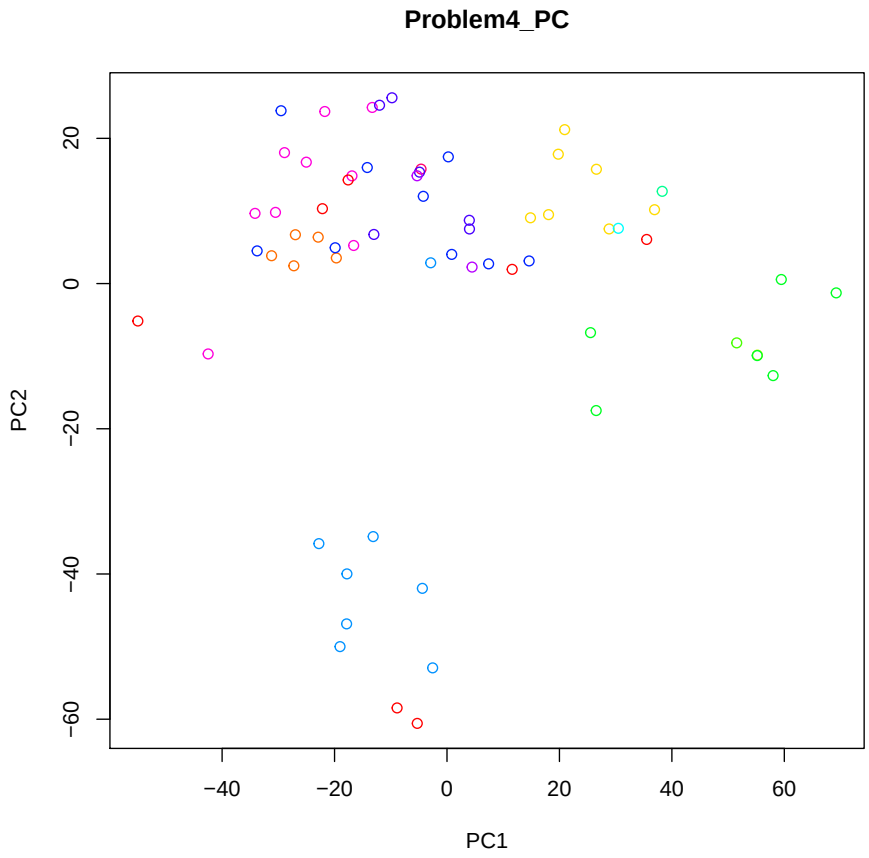
Question 4

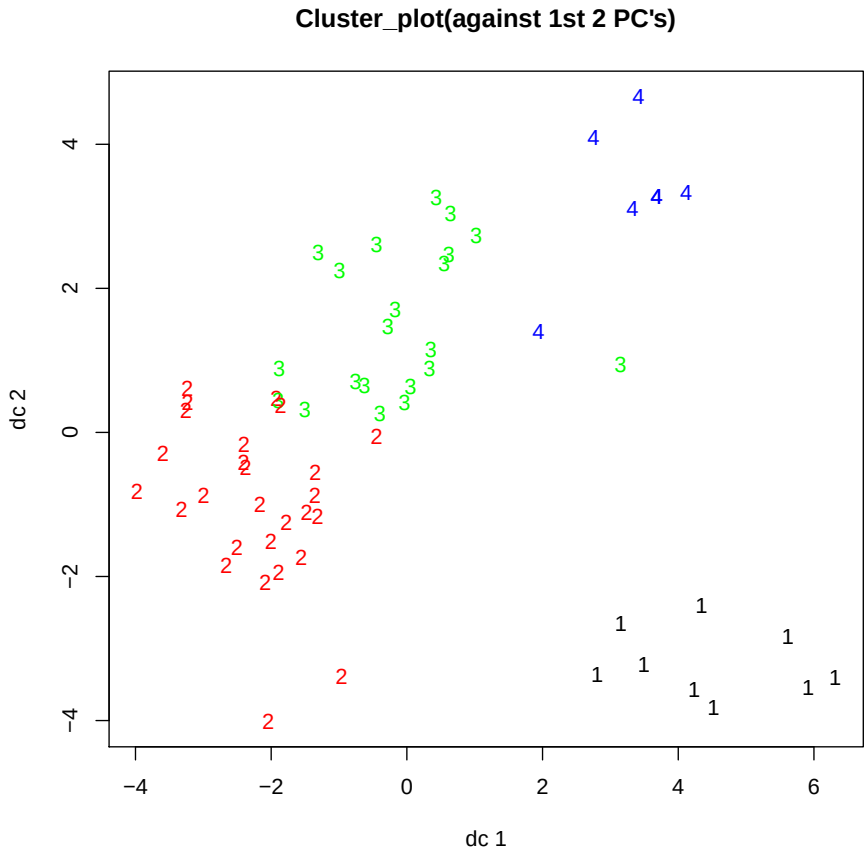
```
library(cluster)
library(fpc)
```

```
## Warning:  package 'fpc' was built under R version 3.2.5
```

```
plot(pr.out$x[,1:2], main = "Problem4_PC",pch = 21, col= Cols(nci.labs))
```

```r
set.seed(0)
fit= kmeans(pr.out$x[,1:4], 4)
plotcluster(pr.out$x[,1:2], fit$cluster, main = "Cluster_plot(against 1st 2 PC's)")
```





The first plot shows the projection of NCI60 cancer cell line on first 2 principal components. The second plot shows kmean clustering of first 4 principal components plotted onto the first two principal components. It shows that cell line from same cancer type does shows similar expression levels.

Question 5

```
library(MASS)
library(ISLR)
data("Khan")

trainx <- as.data.frame(Khan$xtrain)
trainy<- (Khan$ytrain)
khan.pca = prcomp (trainx, scale = TRUE)
khan.cl = qda(trainy~khan.pca$x[,1:4])
```

```
khan.cl
```

```
## Call:
## qda(trainy ~ khan.pca$x[, 1:4])
##
## Prior probabilities of groups:
##         1         2         3         4
## 0.1269841 0.3650794 0.1904762 0.3174603
##
## Group means:
##   khan.pca$x[, 1:4]PC1 khan.pca$x[, 1:4]PC2 khan.pca$x[, 1:4]PC3 khan.pca$x[, 1:4]PC4
## 1            19.293771            -7.078198            -4.386389           23.6725946
## 2            -4.928023            -1.588148             9.308908           -0.6028703
## 3            12.609426             4.270438           -11.323943           -8.1816065
## 4            -9.615937             2.095386            -2.156323           -3.8667731
```

```
dat=data.frame(x=Khan$xtrain, y=as.factor(Khan$ytrain))
cl = predict(khan.cl, dat)
table(cl$class, dat$y)
```

```
##
##      1  2  3  4
##   1  8  0  0  0
##   2  0 19  0  5
##   3  0  0 12  0
##   4  0  4  0 15
```

```
library(MASS)
data("Khan")
trainx <- (Khan$xtrain)
trainy<- (Khan$ytrain)
fit = qda(trainy~trainx[,1:4])
fit
```

```
## Call:
## qda(trainy ~ trainx[, 1:4])
##
## Prior probabilities of groups:
##         1         2         3         4
## 0.1269841 0.3650794 0.1904762 0.3174603
##
## Group means:
##   trainx[, 1:4]1 trainx[, 1:4]2 trainx[, 1:4]3 trainx[, 1:4]4
## 1    -1.61763028     -2.5599246    -0.03645743     -0.8083104
## 2     0.52197824     -2.1051780    -0.11078146     -1.4101961
```

```
## 3    -0.03725562     -2.0683272     0.22719767     -0.2844974
## 4     0.53196338     -0.7919321    -0.77764231     -1.2802346
```

```
dat=data.frame(x=Khan$xtrain, y=as.factor(Khan$ytrain))
cl1 = predict(fit, dat)
table(cl1$class, dat$y)
```

```
##
##      1  2  3  4
##   1  8  0  0  0
##   2  0 16  1  3
##   3  0  5 11  0
##   4  0  2  0 17
```

Quadratic discriminate analysis were performed on Khan dataset. QDA is classified on first four principal components in the first part and then on the first four genes and it was tested on the training data. QDA provides a less direct approach to modeling the predicted probabilities given some set of predictor(X). We can see that the QDA model predicts probability of gene expression level in the training data, corresponds to the tissue type . We also see the group means; these are the average of each predictor within each class.

Question 6

```
library(tree)
```

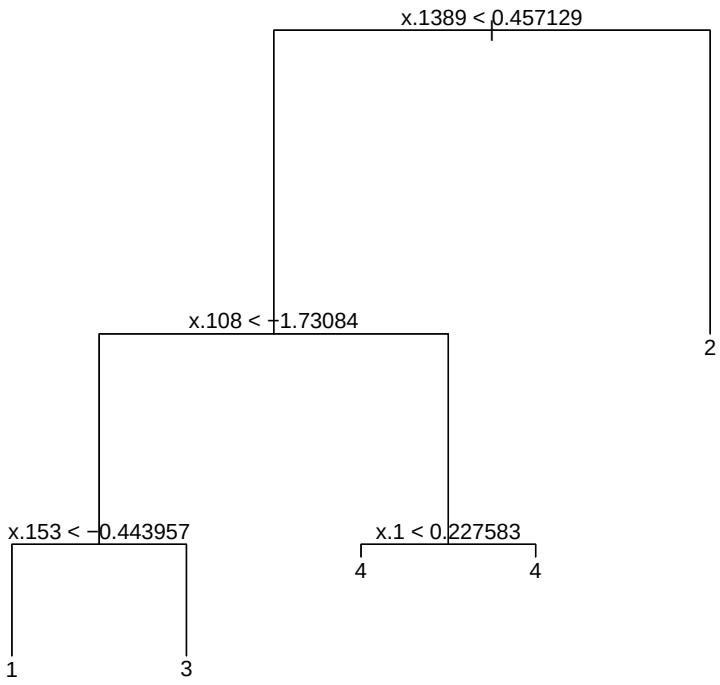
```
## Warning: package 'tree' was built under R version 3.2.5
```

```
library(ISLR);data("Khan")

dat=data.frame(x=Khan$xtrain , y=as.factor(Khan$ytrain))
tree.khan =tree(y~., dat)
summary(tree.khan)
```

```
##
## Classification tree:
## tree(formula = y ~ ., data = dat)
## Variables actually used in tree construction:
## [1] "x.1389" "x.108"  "x.153"  "x.1"
## Number of terminal nodes:  5
## Residual mean deviance:  0.1902 = 11.03 / 58
## Misclassification error rate: 0.03175 = 2 / 63
```

```
dat1=data.frame(x=Khan$xtest , y=as.factor(Khan$ytest))
pred.treetest = predict(tree.khan, dat1)
table(pred.treetest)
```

```
## pred.treetest
##     0 0.125   0.2   0.8 0.875     1
##    54     4     2     2     4    14

plot(tree.khan)
text(tree.khan)
```



In problem 6, we use classification tree to analysis the Khan dataset. On running the tree, the summary lists the variables that are used as internal nodes in the tree, the number of terminal nodes are 5 and the error rate(training data) is 3 percent. The deviance is also reported in the summary of classification tree, which is 0.19. A small deviance indicates that tree provides a good fit to the training data. The plot function displays the tree structure.

Question 7

```
library(e1071)
```

```
## Warning:  package 'e1071' was built under R version 3.2.5
```

```
dat=data.frame(x=Khan$xtrain , y=as.factor(Khan$ytrain))
output =svm(y~.,dat , kernel ="radial",cost =10)
summary(output)
```

```
##
## Call:
## svm(formula = y ~ ., data = dat, kernel = "radial", cost = 10)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##       gamma:  0.0004332756
##
## Number of Support Vectors:  62
##
##  ( 22 20 12 8 )
##
##
## Number of Classes:  4
##
## Levels:
##  1 2 3 4
```

```
table(output$fitted,dat$y)
```

```
##
##      1  2  3  4
##   1  8  0  0  0
##   2  0 23  0  0
##   3  0  0 12  0
##   4  0  0  0 20
```

```
pred.svmtrain = predict(output,dat)
table(pred.svmtrain, dat$y)
```

```
##
## pred.svmtrain  1  2  3  4
##             1  8  0  0  0
##             2  0 23  0  0
##             3  0  0 12  0
##             4  0  0  0 20
```

```
dat1=data.frame(x=Khan$xtest , y=as.factor(Khan$ytest))
pred.svmtest=predict(output, dat1)
table(pred.svmtest, dat1$y)

##
## pred.svmtest 1 2 3 4
##            1 0 0 0 0
##            2 1 4 0 0
##            3 0 0 1 0
##            4 2 2 5 5
```

In problem7, we use the support vector regression on Khan dataset. We use support vector approach to predict cancer subtype using gene expression measurements. The radial kernel provided additional flexibility. We see there are no training errors.We tested it on test data as well as training data.Svm, function will perform multi-class classification using the one-versus-one approach.

Question 8

```
dat1 = Khan$xtrain[Khan$ytrain == 2,]
dat2 = Khan$xtrain[Khan$ytrain == 4,]
xvals = rbind(dat1,dat2)
yvals = c(rep(0,23),rep(1,20))
dat = data.frame(x=xvals,y=yvals)
resLogit <- glm(formula = yvals~xvals,family = binomial, data = dat)
pred.res = predict(resLogit)
summary(pred.res)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -26.570 -26.570 -26.570  -1.853  26.570  26.570
```

In problem8, we estimate a logistic regression model using the glm (generalized linear model) function. The summary of logistic regression is not printed as it was very long but the output of logistic regression shows residual deviance as 59.2. However coefficients were computed only for few variables and some of them showes N/A.

Question 9

```
library(randomForest)

## Warning: package 'randomForest' was built under R version 3.2.5
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

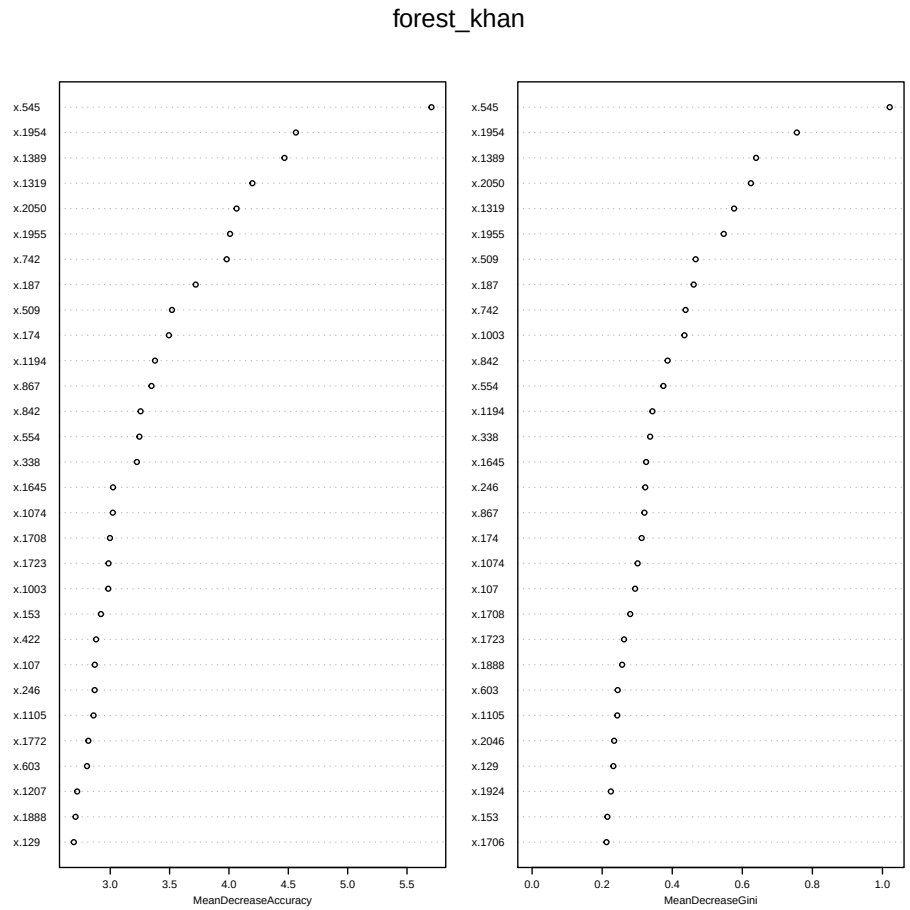
```
library(ISLR)
data("Khan")
set.seed(0)
khan_data=data.frame(x=Khan$xtrain , y=as.factor(Khan$ytrain))
forest_khan = randomForest(y~.,khan_data, mtry = 50, importance = TRUE)
forest_khan

##
## Call:
##  randomForest(formula = y ~ ., data = khan_data, mtry = 50, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 50
##
##         OOB estimate of  error rate: 0%
## Confusion matrix:
##    1  2  3  4 class.error
## 1 8  0  0  0           0
## 2 0 23  0  0           0
## 3 0  0 12  0           0
## 4 0  0  0 20           0

pred.fortrain = predict(forest_khan, dat)
table(pred.fortrain, dat$y)

##
## pred.fortrain  0  1
##             1  0  0
##             2 23  0
##             3  0  0
##             4  0 20

dat_khan1=data.frame(x=Khan$xtest , y=as.factor(Khan$ytest))
pred.fortest = predict(forest_khan, dat_khan1)
table(pred.fortest, dat_khan1$y)

##
## pred.fortest 1 2 3 4
##            1 3 0 0 0
##            2 0 6 0 0
##            3 0 0 5 0
##            4 0 0 1 5
```

Random forest classifier is applied on the Khan dataset and the tree is fitted on to the khan training data. The summary shows the number of trees are 500 and number of variable tried at each split

is 50. The error rate is 0 percent. Random forest classifier is tested on training as well as test data.

Question 10

```
varImpPlot(forest_khan, cex = 0.5)
```



From the plot it could be inferred that the top two genes are x.545 and x.1954 respectively as the topmost variables are the most significant one.
Based on the Gini plot, a score with a greater Gini index produces a greater separation among the observations. In case of khan dataset a higher Gini index indicates greater ability to distinguish the expression level. Therefore, the expression scores of genes with the highest Gini index are preferred. Hence we might keep the genes with greater Gini index in this case first 2 genes.