**Chapter 10: Unsupervised Learning**

**1(a)**

证明：

****

**1(b)**

减小每个类中每个观测与类中心欧式距离的平方之和与减小每个类中各个观测之间所有成对的欧式距离的平方之和除以类中观测总数是等价的。

**2(a)**

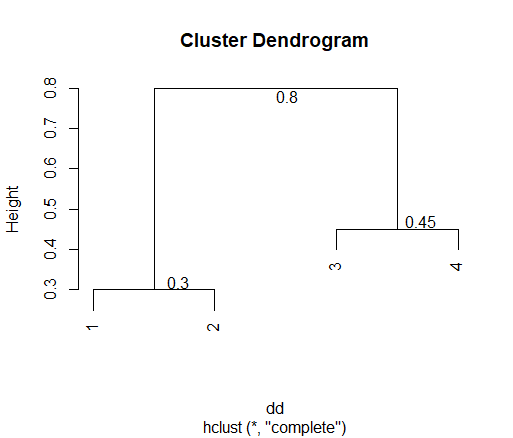
> dd = as.dist(matrix(c(0, 0.3, 0.4, 0.7, 0.3, 0, 0.5, 0.8, 0.4, 0.5, 0, 0.45, 0.7, 0.8, 0.45, 0), nrow=4))

> plot(hclust(dd, method="complete"))

> text(x = 1.7, y = 0.32, labels = "0.3")

> text(x = 3.7, y = 0.47, labels = "0.45")

> text(x = 2.6, y = 0.78, labels = "0.8")



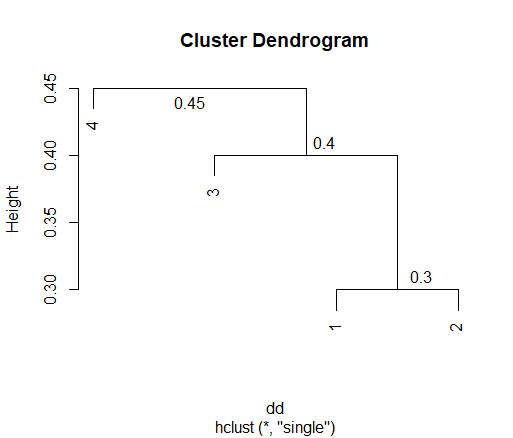
**2(b)**

> plot(hclust(dd, method="single"))

> text(x = 3.7, y = 0.31, labels = "0.3")

> text(x = 2.9, y = 0.41, labels = "0.4")

> text(x = 1.8, y = 0.44, labels = "0.45")



**2(c)**

第1个类有观测1和观测2，第2个类有观测3和观测4。

**2(d)**

第1个类有观测1，观测2和观测3，第2个类有观测4。

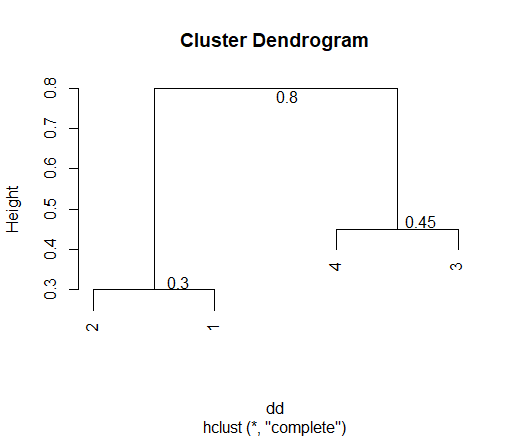
**2(e)**

> plot(hclust(dd, method="complete"), labels=c(2,1,4,3))

> text(x = 1.7, y = 0.32, labels = "0.3")

> text(x = 3.7, y = 0.47, labels = "0.45")

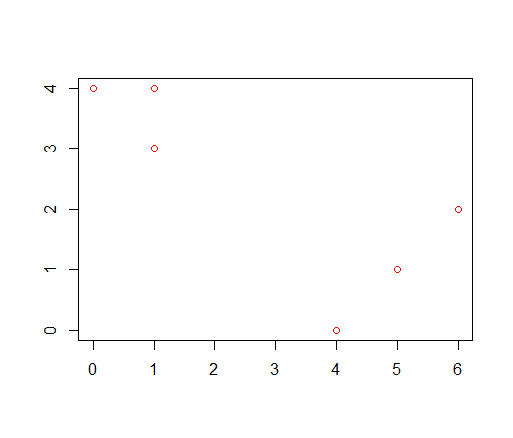
> text(x = 2.6, y = 0.78, labels = "0.8")



**3(a)**

> X = matrix(c(1, 1, 0, 5, 6, 4, 4, 3, 4, 1, 2, 0), ncol = 2)

> plot(X[,1],X[,2],xlab = "",ylab = "",col = "red")



**3(b)**

> label = sample(2, nrow(X), replace=T)

> label

[1] 2 1 2 2 2 1

**3(c)**

> center1 = c(mean(X[label == 1, 1]), mean(X[label == 1, 2]))

> center2 = c(mean(X[label == 2, 1]), mean(X[label == 2, 2]))

>

> center1

[1] 2.5 1.5

> center2

[1] 3.00 2.75

**3(d)**

> Eculidean = function(x,y){

+ return(sqrt((x[1]-y[1])^2+(x[2]-y[2])^2))

+ }

> k2 = function(X,center1,center2){

+ label = rep(NA, nrow(X))

+ for (i in 1:nrow(X)) {

+ if(Eculidean(X[i,],center1) <= Eculidean(X[i,],center2)){

+ label[i] = 1

+ }

+ else{

+ label[i] = 2

+ }

+ }

+ return(label)

+ }

> k2(X,center1,center2)

[1] 2 2 2 1 2 1

**3(e)**

> finallabel = rep(0, 6)

> while (!all(finallabel == label)) {

+ finallabel = label

+ center1 = c(mean(X[label == 1, 1]), mean(X[label == 1, 2]))

+ center2 = c(mean(X[label == 2, 1]), mean(X[label == 2, 2]))

+ label = k2(X, center1, center2)

+ }

>

> label

[1] 2 2 2 1 1 1

> center1 = c(mean(X[label == 1, 1]), mean(X[label == 1, 2]))

> center2 = c(mean(X[label == 2, 1]), mean(X[label == 2, 2]))

> center1

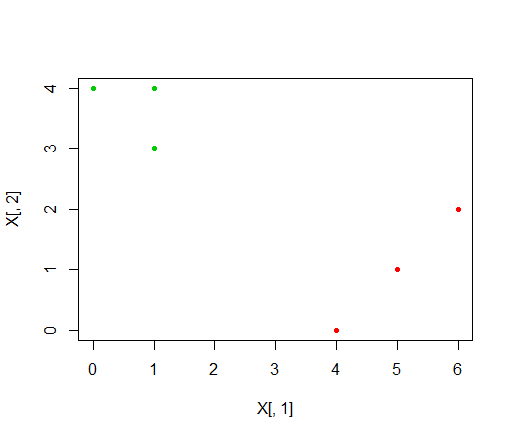
[1] 5 1

> center2

[1] 0.6666667 3.6666667

**3(f)**

> plot(X[,1], X[,2], col=(label+1), pch=20)



**4(a)**

信息不充分。如果这两个类的最大类间相异度与最小类间相异度相等，那么汇合点就在相同的高度，如果最大类间相异度大于最小类间相异度，那么用complete linkage方法得到的汇合点就比用single linkage方法得到的汇合点高。

**4(b)**

在相同的高度。两个观测只有一个相异度，用哪种方法得到的汇合点都是一样的。

**5**

如图10.14

没有进行标准化时，{1,2,3,7,8}会被归为一类，因为这几个网购者买了较多的袜子或电脑，或者同时买了较多的袜子和电脑，其中袜子的购买数量对相异度指标影响较大。{4,5,6}会被归为另一类，他们买了较少的袜子和电脑。

进行标准化后，{1,2,3,4}会被归为一类，因为此时电脑的购买数量对相异度指标影响较大，而这几个网购者没有买电脑。{5,6,7,8}会被归为另一类，他们都买了电脑。

第三种情况下，{1,2,3,4}会被归为一类，因为此时网购产品的购买金额对相异度指标影响较大，而这几个网购者没有买电脑，花费的金额较少。{5,6,7,8}会被归为另一类，他们都买了电脑，花费的金额较多。

**6(a)**

将观测投影到第一主成分上损失了90%的信息，即数据的方差有90%是不在第一主成分的解释范围之内的。

**6(b)**

第一主成分的PVE较小，可以将病人样本是在机器A或机器B中处理的作为一个属性加到数据集中，然后做主成分分析，再进行双样本t检验。

**6(c)**

> set.seed(1000)

> Control = matrix(rnorm(50\*1000), ncol=50)

> Treatment = matrix(rnorm(50\*1000), ncol=50)

> G = cbind(Control, Treatment)

> G[1,] = seq(-10,0.21,10.79)

> pr.out = prcomp(scale(G))

> pr.var = pr.out$sdev^2

> pve = pr.var/sum(pr.var)

> pve[1]

[1] 0.1011156

为原数据集增加一行，设病人样本用机器A处理的取值为10，用机器B处理的取值为0，代码如下：

> set.seed(10)

> A = sample(c(0,10),50,replace = T,prob = c(0.2,0.8))

> B = sample(c(0,10),50,replace = T,prob = c(0.8,0.2))

> G = rbind(G,c(A,B))

> pr.out = prcomp(scale(G))

> pr.var = pr.out$sdev^2

> pve = pr.var/sum(pr.var)

> pve[1]

[1] 0.1197798

可以看到，PVE有所增大。

**7**

> library(ISLR)

> scale\_USArrests = scale(USArrests)

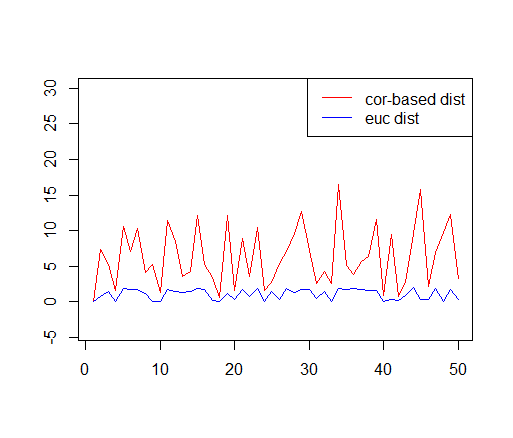
> r = as.matrix(as.dist(1 - cor(t(scale\_USArrests))))

> d = as.matrix(dist(scale\_USArrests)^2)

> plot(c(1:50),d[1,],type = "l",ylim = c(-4,30),col = "red",xlab = "",ylab = "")

> lines(c(1:50),r[1,],col = "blue")

> legend("topright",c("cor-based dist","euc dist"),col = c("red","blue"),cex = 1, lty = 1)



以欧几里得距离的平方矩阵的第一行和矩阵的第一行为例，作图可以看出，二者是成比例的。

**8(a)**

> pr.out = prcomp(USArrests, center=T, scale=T)

> pr.var = pr.out$sdev^2

> pve = pr.var / sum(pr.var)

> pve

[1] 0.62006039 0.24744129 0.08914080 0.04335752

**8(b)**

> cal\_pve = 0

> for (m in 1:4) {

+ a = 0

+ for (i in 1:50) {

+ b = 0

+ for (j in 1:4) {

+ b = b + rotation[j,m] \* scale\_USArrests[i,j]

+ }

+ a = a + b^2

+ }

+ cal\_pve[m] = a

+ }

> sumxij2 = 0

> for (j in 1:4) {

+ for (i in 1:50) {

+ sumxij2 = sumxij2 + scale\_USArrests[i,j]^2

+ }

+ }

> cal\_pve = cal\_pve/sumxij2

> cal\_pve

[1] 0.62006039 0.24744129 0.08914080 0.04335752

**9(a)**

> hc.complete = hclust(dist(USArrests), method="complete")

> plot(hc.complete)

**9(b)**

> cutree(hc.complete, 3)



> table(cutree(hc.complete, 3))

1 2 3

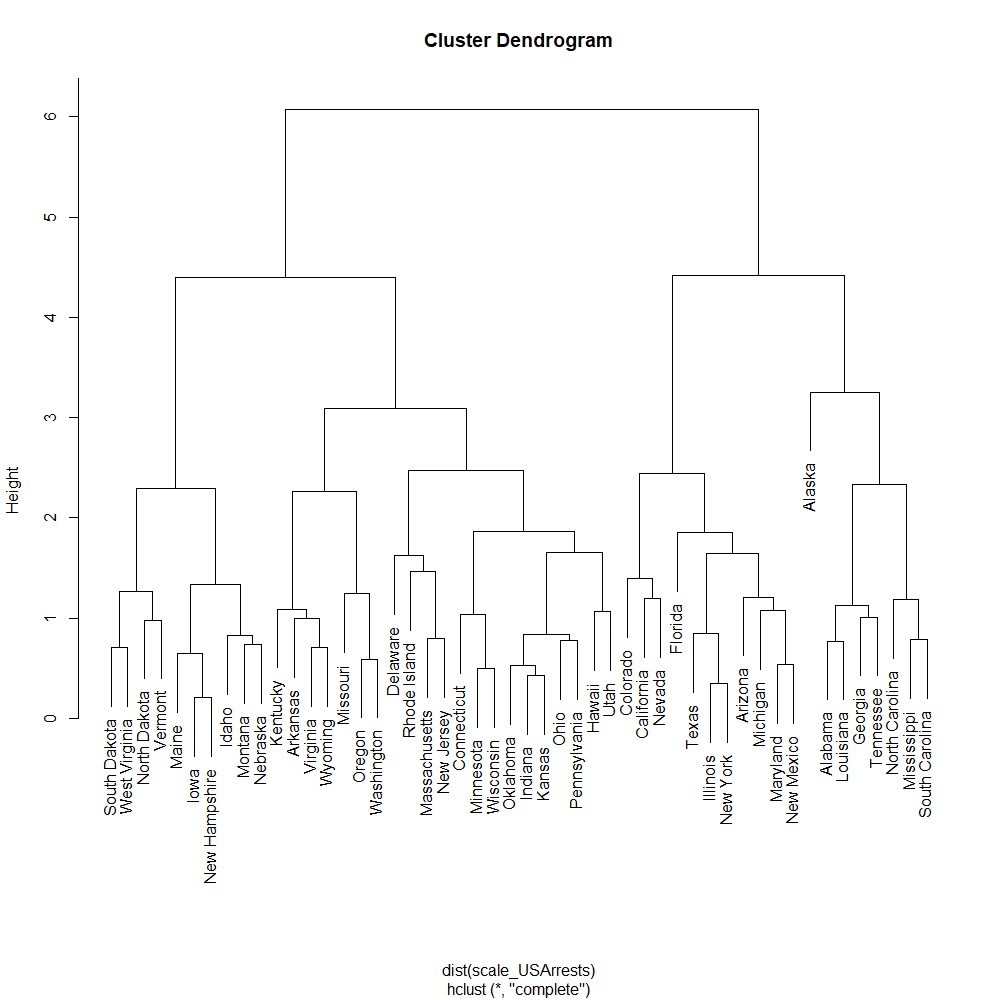
16 14 20

在1,2,3类的州的数量分别为16,14,20。

**9(c)**

> scale\_hc.complete = hclust(dist(scale\_USArrests), method="complete")

> plot(scale\_hc.complete)



**9(d)**

> cutree(scale\_hc.complete, 3)



> table(cutree(scale\_hc.complete, 3))

1 2 3

8 11 31

> table(cutree(scale\_hc.complete, 3), cutree(hc.complete, 3))

1 2 3

1 6 2 0

2 9 2 0

3 1 10 20

**10(a)**

> set.seed (2)

> x=matrix (rnorm (60\*50) , ncol = 50)

> x[1:20 ,1]= x[1:20 ,1]+5

> x[1:20 ,2]= x[1:20 ,2]+5

> x[21:40 ,1]=x[21:40 ,1] -5

> x[21:40 ,2]=x[21:40 ,2] -5

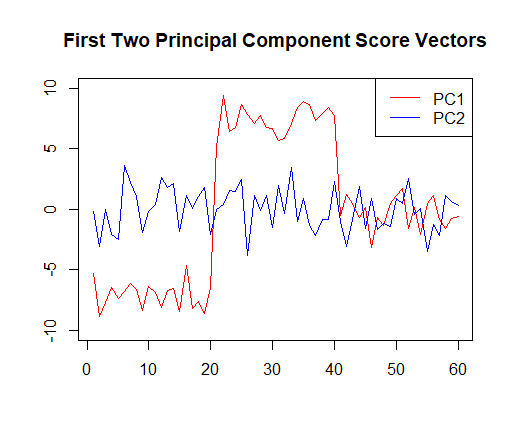
**10(b)**

> pr.out = prcomp(x)

> plot(c(1:60),pr.out$x[,1],col = "red",type = "l",xlab = "",ylab = "",main = "First Two Principal Component Score Vectors",ylim = c(-10,10))

> lines(c(1:60),pr.out$x[,2],col = "blue")

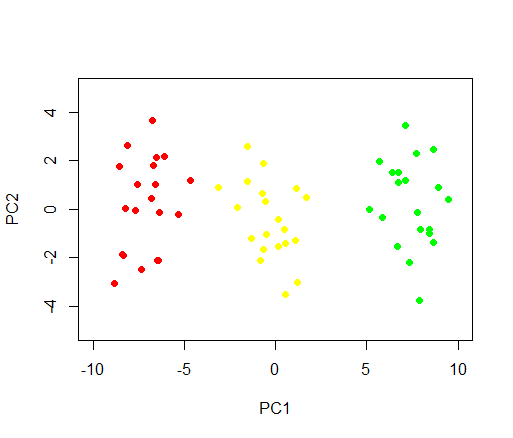
> legend("topright",c("PC1","PC2"),col = c("red","blue"),cex = 1, lty = 1)



> plot(pr.out$x[1:20,1:2], col="red", xlab="PC1", ylab="PC2", pch=19,xlim = c(-10,10),ylim = c(-5,5))

> points(pr.out$x[21:40,1:2], col="green",pch=19)

> points(pr.out$x[41:60,1:2], col="yellow",pch=19)



**10(c)**

> km.out$cluster

[1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3

[59] 3 3

> table(km.out$cluster, c(rep(1,20), rep(2,20), rep(3,20)))

1 2 3

1 0 20 0

2 20 0 0

3 0 0 20

因为K-means方法会随机为类进行编号，所以K-means得到的第1类，第2类和第3类分别对应原数据的第2类，第1类和第3类，从而，K-means的聚类效果很好，成功将原数据聚成3类没有误差。

**10(d)**

> km.out = kmeans(x, 2, nstart=20)

> km.out$cluster

[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1

[59] 1 1

> table(km.out$cluster, c(rep(1,20), rep(2,20), rep(3,20)))

1 2 3

1 20 0 19

2 0 20 1

第1类和第2类的聚类结果是完全正确的。对于第3类，K-means将19个第3类的数据聚为第1类，将1个第3类数据聚为第2类。

**10(e)**

> km.out = kmeans(x, 4, nstart=20)

> km.out$cluster

[1] 1 1 1 1 1 2 1 2 1 1 2 2 2 2 1 2 1 2 2 1 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4

[59] 4 4

> table(km.out$cluster, c(rep(1,20), rep(2,20), rep(3,20)))

1 2 3

1 11 0 0

2 9 0 0

3 0 20 0

4 0 0 20

对于第1类数据，K-means将11个第1类的数据聚为第1类，将9个第1类数据聚为第2类。K-means得到的第3类和第4类分别对应原数据的第2类和第3类，是完全正确的。

**10(f)**

> km.out = kmeans(pr.out$x[,1:2], 3, nstart=20)

> km.out$cluster

[1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

[59] 1 1

> table(km.out$cluster, c(rep(1,20), rep(2,20), rep(3,20)))

1 2 3

1 0 0 20

2 0 20 0

3 20 0 0

K-means得到的第1类，第2类和第3类分别对应原数据的第3类，第2类和第1类。 K-means的聚类效果很好，成功将原数据聚成3类没有误差。

**10(g)**

> km.out = kmeans(scale(x), 3, nstart=20)

> km.out$cluster

[1] 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 1 3 3 2 1 1 1 1 1 1 1 1 1 1 1 2 1 2 1 2 1 1 2 1 3 2 3 1 2 3 2 3 3 1 1 3 2 3 2 2 2 3

[59] 2 2

> table(km.out$cluster, c(rep(1,20), rep(2,20), rep(3,20)))

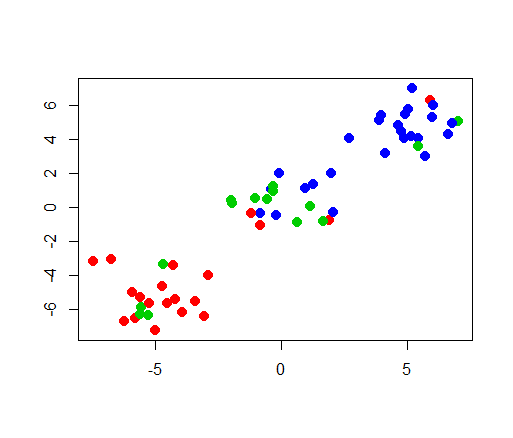
1 2 3

1 1 16 3

2 2 4 9

3 17 0 8

> plot(x, col = (km.out$cluster +1), xlab ="", ylab="", pch =20, cex =2)



K-means聚类效果很差，有很多数据被聚到错误的类中。

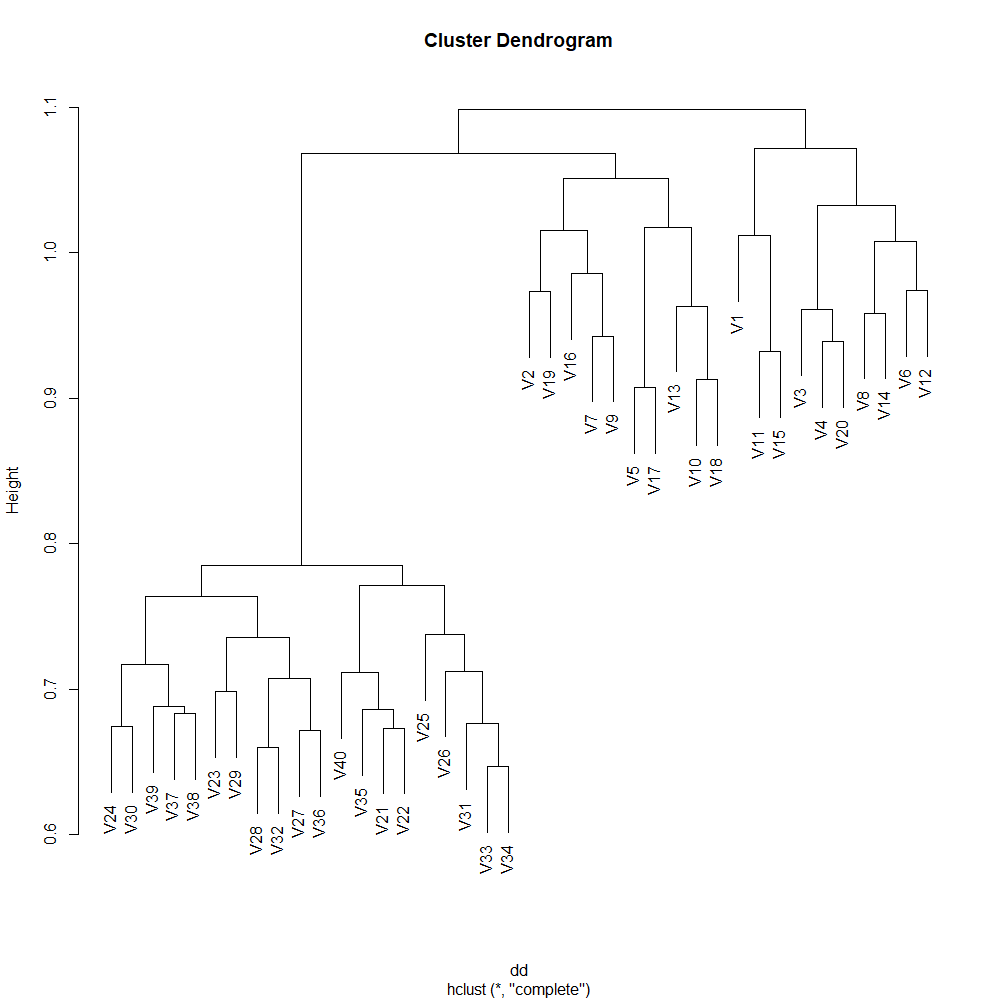
**11(a)**

> Ch10Ex11<-read.csv("D:\\研究生\\ISLR\\ILS课后练习\\Ch10Ex11.csv",header = F)

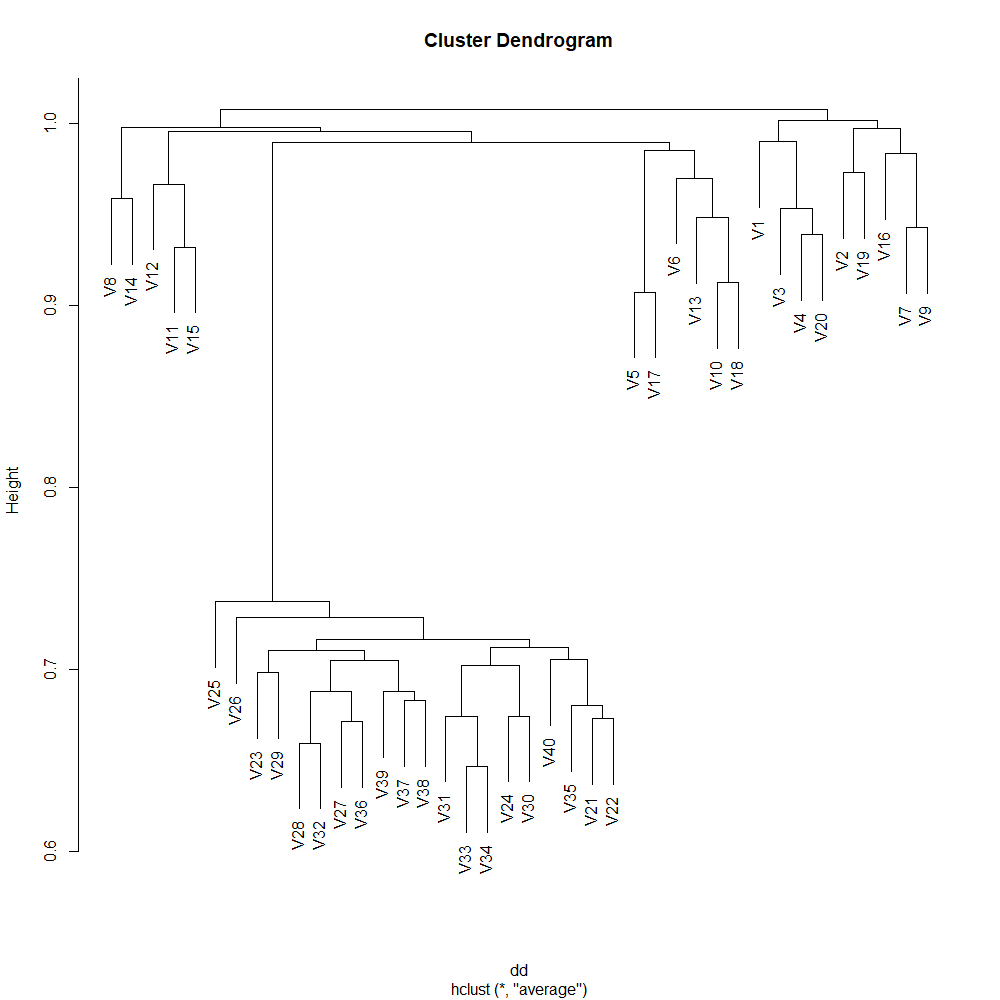
**11(b)**

> dd = as.dist(1 - cor(Ch10Ex11))

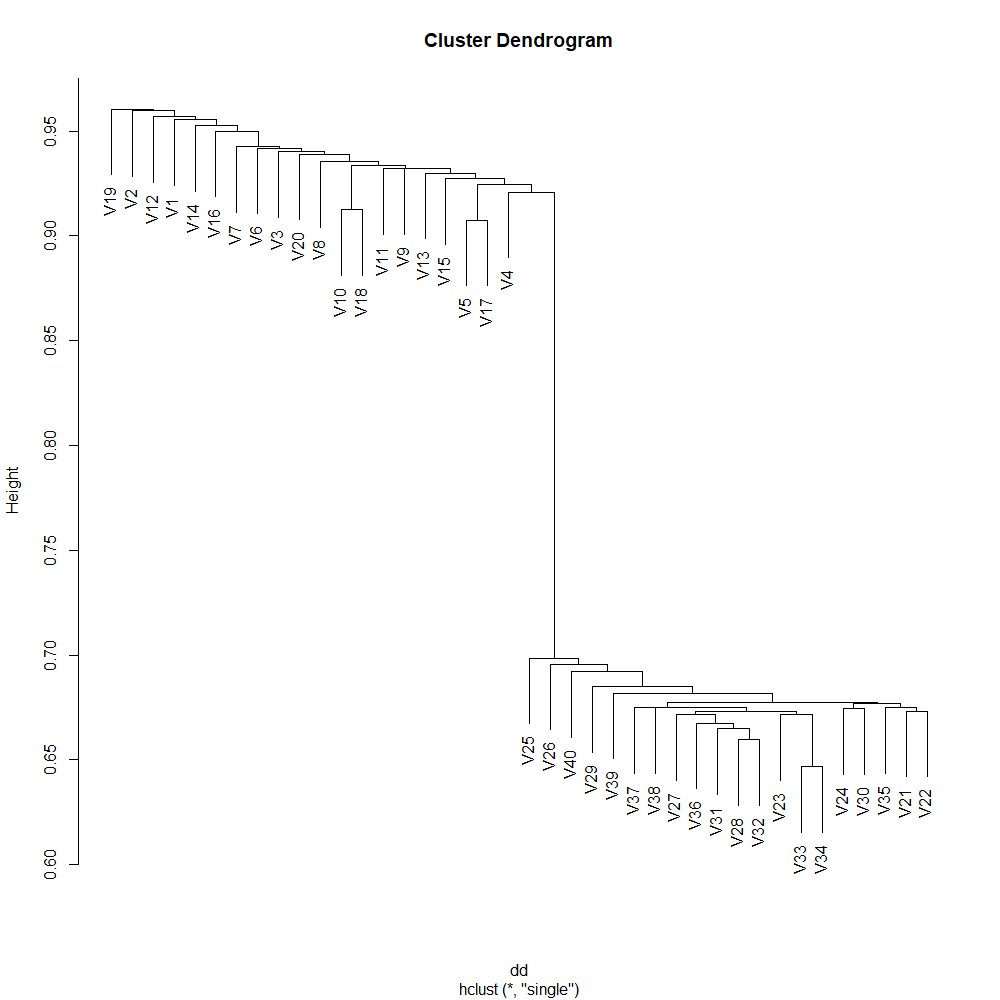
> plot(hclust(dd, method="complete"))



> plot(hclust(dd, method="average"))



> plot(hclust(dd, method="single"))



对于最长距离和平均距离法，是可以将样本分到两个类中的，用最短距离法得到的结果不是很适合将样本分成两类。显然分类结果与距离类型有关。

**11(c)**

将数据集分成healthy和diseased 两部分，将数据集标准化后，分别计算两个数据集各个基因的均值，然后计算哪个基因在两组的差距最大。

> Ch10Ex11 = scale(Ch10Ex11)

> healthy = Ch10Ex11[,1:20]

> diseased = Ch10Ex11[,21:40]

> healthy\_dist = apply(healthy, 1, mean)

> diseased\_dist = apply(diseased, 1, mean)

> which.max(abs(healthy\_dist - diseased\_dist))

[1] 600

> abs(healthy\_dist - diseased\_dist)[600]

[1] 2.215968

由此可知，第600个基因在两组（healthy和diseased）中差得最多。