Class 08 Mini-project

Andre Modolo

##Unsupervised Learning Analysis of Human Breast Cancer Cells

#Data Import Save the file to you computer in the Class 08 BIMM143 folder

wisc.df <- read.csv("WisconsinCancer.csv", row.names=1)  
head(wisc.df)

diagnosis radius\_mean texture\_mean perimeter\_mean area\_mean  
842302 M 17.99 10.38 122.80 1001.0  
842517 M 20.57 17.77 132.90 1326.0  
84300903 M 19.69 21.25 130.00 1203.0  
84348301 M 11.42 20.38 77.58 386.1  
84358402 M 20.29 14.34 135.10 1297.0  
843786 M 12.45 15.70 82.57 477.1  
 smoothness\_mean compactness\_mean concavity\_mean concave.points\_mean  
842302 0.11840 0.27760 0.3001 0.14710  
842517 0.08474 0.07864 0.0869 0.07017  
84300903 0.10960 0.15990 0.1974 0.12790  
84348301 0.14250 0.28390 0.2414 0.10520  
84358402 0.10030 0.13280 0.1980 0.10430  
843786 0.12780 0.17000 0.1578 0.08089  
 symmetry\_mean fractal\_dimension\_mean radius\_se texture\_se perimeter\_se  
842302 0.2419 0.07871 1.0950 0.9053 8.589  
842517 0.1812 0.05667 0.5435 0.7339 3.398  
84300903 0.2069 0.05999 0.7456 0.7869 4.585  
84348301 0.2597 0.09744 0.4956 1.1560 3.445  
84358402 0.1809 0.05883 0.7572 0.7813 5.438  
843786 0.2087 0.07613 0.3345 0.8902 2.217  
 area\_se smoothness\_se compactness\_se concavity\_se concave.points\_se  
842302 153.40 0.006399 0.04904 0.05373 0.01587  
842517 74.08 0.005225 0.01308 0.01860 0.01340  
84300903 94.03 0.006150 0.04006 0.03832 0.02058  
84348301 27.23 0.009110 0.07458 0.05661 0.01867  
84358402 94.44 0.011490 0.02461 0.05688 0.01885  
843786 27.19 0.007510 0.03345 0.03672 0.01137  
 symmetry\_se fractal\_dimension\_se radius\_worst texture\_worst  
842302 0.03003 0.006193 25.38 17.33  
842517 0.01389 0.003532 24.99 23.41  
84300903 0.02250 0.004571 23.57 25.53  
84348301 0.05963 0.009208 14.91 26.50  
84358402 0.01756 0.005115 22.54 16.67  
843786 0.02165 0.005082 15.47 23.75  
 perimeter\_worst area\_worst smoothness\_worst compactness\_worst  
842302 184.60 2019.0 0.1622 0.6656  
842517 158.80 1956.0 0.1238 0.1866  
84300903 152.50 1709.0 0.1444 0.4245  
84348301 98.87 567.7 0.2098 0.8663  
84358402 152.20 1575.0 0.1374 0.2050  
843786 103.40 741.6 0.1791 0.5249  
 concavity\_worst concave.points\_worst symmetry\_worst  
842302 0.7119 0.2654 0.4601  
842517 0.2416 0.1860 0.2750  
84300903 0.4504 0.2430 0.3613  
84348301 0.6869 0.2575 0.6638  
84358402 0.4000 0.1625 0.2364  
843786 0.5355 0.1741 0.3985  
 fractal\_dimension\_worst  
842302 0.11890  
842517 0.08902  
84300903 0.08758  
84348301 0.17300  
84358402 0.07678  
843786 0.12440

Create a new df without the expert diagnosis so you don’t have the “answer” to whether the cells are malignant or benign

wisc.data <- wisc.df[,-1]  
head(wisc.data)

radius\_mean texture\_mean perimeter\_mean area\_mean smoothness\_mean  
842302 17.99 10.38 122.80 1001.0 0.11840  
842517 20.57 17.77 132.90 1326.0 0.08474  
84300903 19.69 21.25 130.00 1203.0 0.10960  
84348301 11.42 20.38 77.58 386.1 0.14250  
84358402 20.29 14.34 135.10 1297.0 0.10030  
843786 12.45 15.70 82.57 477.1 0.12780  
 compactness\_mean concavity\_mean concave.points\_mean symmetry\_mean  
842302 0.27760 0.3001 0.14710 0.2419  
842517 0.07864 0.0869 0.07017 0.1812  
84300903 0.15990 0.1974 0.12790 0.2069  
84348301 0.28390 0.2414 0.10520 0.2597  
84358402 0.13280 0.1980 0.10430 0.1809  
843786 0.17000 0.1578 0.08089 0.2087  
 fractal\_dimension\_mean radius\_se texture\_se perimeter\_se area\_se  
842302 0.07871 1.0950 0.9053 8.589 153.40  
842517 0.05667 0.5435 0.7339 3.398 74.08  
84300903 0.05999 0.7456 0.7869 4.585 94.03  
84348301 0.09744 0.4956 1.1560 3.445 27.23  
84358402 0.05883 0.7572 0.7813 5.438 94.44  
843786 0.07613 0.3345 0.8902 2.217 27.19  
 smoothness\_se compactness\_se concavity\_se concave.points\_se  
842302 0.006399 0.04904 0.05373 0.01587  
842517 0.005225 0.01308 0.01860 0.01340  
84300903 0.006150 0.04006 0.03832 0.02058  
84348301 0.009110 0.07458 0.05661 0.01867  
84358402 0.011490 0.02461 0.05688 0.01885  
843786 0.007510 0.03345 0.03672 0.01137  
 symmetry\_se fractal\_dimension\_se radius\_worst texture\_worst  
842302 0.03003 0.006193 25.38 17.33  
842517 0.01389 0.003532 24.99 23.41  
84300903 0.02250 0.004571 23.57 25.53  
84348301 0.05963 0.009208 14.91 26.50  
84358402 0.01756 0.005115 22.54 16.67  
843786 0.02165 0.005082 15.47 23.75  
 perimeter\_worst area\_worst smoothness\_worst compactness\_worst  
842302 184.60 2019.0 0.1622 0.6656  
842517 158.80 1956.0 0.1238 0.1866  
84300903 152.50 1709.0 0.1444 0.4245  
84348301 98.87 567.7 0.2098 0.8663  
84358402 152.20 1575.0 0.1374 0.2050  
843786 103.40 741.6 0.1791 0.5249  
 concavity\_worst concave.points\_worst symmetry\_worst  
842302 0.7119 0.2654 0.4601  
842517 0.2416 0.1860 0.2750  
84300903 0.4504 0.2430 0.3613  
84348301 0.6869 0.2575 0.6638  
84358402 0.4000 0.1625 0.2364  
843786 0.5355 0.1741 0.3985  
 fractal\_dimension\_worst  
842302 0.11890  
842517 0.08902  
84300903 0.08758  
84348301 0.17300  
84358402 0.07678  
843786 0.12440

Store the diagnosis values as a factor vector

diagnosis <- factor(wisc.df[,1])  
head(diagnosis)

[1] M M M M M M  
Levels: B M

Get familiar with the data set: Q1) How many observations are in this dataset? ie. How many people?

nrow(wisc.data)

[1] 569

There are 569 different people/ observations in this data set

Q2) How many of the observations have a malignant diagnosis?

table(wisc.df$diagnosis)

B M   
357 212

There are 212 malignant diagnosis

Q3) How many variables/features in the data are suffixed with \_mean? Use colnames() to find the column names

colname <- colnames(wisc.data)  
colname

[1] "radius\_mean" "texture\_mean"   
 [3] "perimeter\_mean" "area\_mean"   
 [5] "smoothness\_mean" "compactness\_mean"   
 [7] "concavity\_mean" "concave.points\_mean"   
 [9] "symmetry\_mean" "fractal\_dimension\_mean"   
[11] "radius\_se" "texture\_se"   
[13] "perimeter\_se" "area\_se"   
[15] "smoothness\_se" "compactness\_se"   
[17] "concavity\_se" "concave.points\_se"   
[19] "symmetry\_se" "fractal\_dimension\_se"   
[21] "radius\_worst" "texture\_worst"   
[23] "perimeter\_worst" "area\_worst"   
[25] "smoothness\_worst" "compactness\_worst"   
[27] "concavity\_worst" "concave.points\_worst"   
[29] "symmetry\_worst" "fractal\_dimension\_worst"

Then you search for “\_mean” pattern using the grep() function

grep("\_mean",colname)

[1] 1 2 3 4 5 6 7 8 9 10

To find how many times we found them you can use the length() function

length(grep("\_mean",colname))

[1] 10

There are 10 variables/features that end with “\_mean”

How many dimensions are in this data set?

dim(wisc.data)

[1] 569 30

569 rows and 30 columns

#Principal Component Analysis

First we need to see if the data needs to be scaled. We start by checking the column means colMeans() and apply it to find the standard deviations for each component

colMeans(wisc.data)

radius\_mean texture\_mean perimeter\_mean   
 1.412729e+01 1.928965e+01 9.196903e+01   
 area\_mean smoothness\_mean compactness\_mean   
 6.548891e+02 9.636028e-02 1.043410e-01   
 concavity\_mean concave.points\_mean symmetry\_mean   
 8.879932e-02 4.891915e-02 1.811619e-01   
 fractal\_dimension\_mean radius\_se texture\_se   
 6.279761e-02 4.051721e-01 1.216853e+00   
 perimeter\_se area\_se smoothness\_se   
 2.866059e+00 4.033708e+01 7.040979e-03   
 compactness\_se concavity\_se concave.points\_se   
 2.547814e-02 3.189372e-02 1.179614e-02   
 symmetry\_se fractal\_dimension\_se radius\_worst   
 2.054230e-02 3.794904e-03 1.626919e+01   
 texture\_worst perimeter\_worst area\_worst   
 2.567722e+01 1.072612e+02 8.805831e+02   
 smoothness\_worst compactness\_worst concavity\_worst   
 1.323686e-01 2.542650e-01 2.721885e-01   
 concave.points\_worst symmetry\_worst fractal\_dimension\_worst   
 1.146062e-01 2.900756e-01 8.394582e-02

round(apply(wisc.data,2,sd),2)

radius\_mean texture\_mean perimeter\_mean   
 3.52 4.30 24.30   
 area\_mean smoothness\_mean compactness\_mean   
 351.91 0.01 0.05   
 concavity\_mean concave.points\_mean symmetry\_mean   
 0.08 0.04 0.03   
 fractal\_dimension\_mean radius\_se texture\_se   
 0.01 0.28 0.55   
 perimeter\_se area\_se smoothness\_se   
 2.02 45.49 0.00   
 compactness\_se concavity\_se concave.points\_se   
 0.02 0.03 0.01   
 symmetry\_se fractal\_dimension\_se radius\_worst   
 0.01 0.00 4.83   
 texture\_worst perimeter\_worst area\_worst   
 6.15 33.60 569.36   
 smoothness\_worst compactness\_worst concavity\_worst   
 0.02 0.16 0.21   
 concave.points\_worst symmetry\_worst fractal\_dimension\_worst   
 0.07 0.06 0.02

You can see that the sd for each variable is quite different so the data is measured with different units and therefore should be scaled.

How we can try prcomp() with scaling

wisc.pr <- prcomp(wisc.data, scale=T)  
summary(wisc.pr)

Importance of components:  
 PC1 PC2 PC3 PC4 PC5 PC6 PC7  
Standard deviation 3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172  
Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251  
Cumulative Proportion 0.4427 0.6324 0.72636 0.79239 0.84734 0.88759 0.91010  
 PC8 PC9 PC10 PC11 PC12 PC13 PC14  
Standard deviation 0.69037 0.6457 0.59219 0.5421 0.51104 0.49128 0.39624  
Proportion of Variance 0.01589 0.0139 0.01169 0.0098 0.00871 0.00805 0.00523  
Cumulative Proportion 0.92598 0.9399 0.95157 0.9614 0.97007 0.97812 0.98335  
 PC15 PC16 PC17 PC18 PC19 PC20 PC21  
Standard deviation 0.30681 0.28260 0.24372 0.22939 0.22244 0.17652 0.1731  
Proportion of Variance 0.00314 0.00266 0.00198 0.00175 0.00165 0.00104 0.0010  
Cumulative Proportion 0.98649 0.98915 0.99113 0.99288 0.99453 0.99557 0.9966  
 PC22 PC23 PC24 PC25 PC26 PC27 PC28  
Standard deviation 0.16565 0.15602 0.1344 0.12442 0.09043 0.08307 0.03987  
Proportion of Variance 0.00091 0.00081 0.0006 0.00052 0.00027 0.00023 0.00005  
Cumulative Proportion 0.99749 0.99830 0.9989 0.99942 0.99969 0.99992 0.99997  
 PC29 PC30  
Standard deviation 0.02736 0.01153  
Proportion of Variance 0.00002 0.00000  
Cumulative Proportion 1.00000 1.00000

We captured 100% of the variance after 29 principal component analysis iterations

Q4) From your results, what proportion of the original variance is captured by the first principal components (PC1)?

44.27%

Q5)How many principal components (PCs) are required to describe at least 70% of the original variance in the data?

You need at least 3 PCs

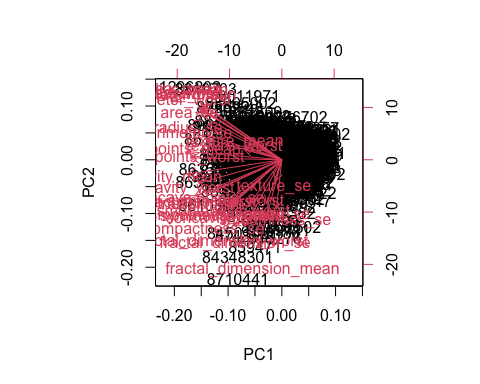
Q6) How many principal components (PCs) are required to describe at least 90% of the original variance in the data?

You need 7 PCs

#Interpreting PCA results

Lets try interpreting PCA results using biplot()

biplot(wisc.pr)

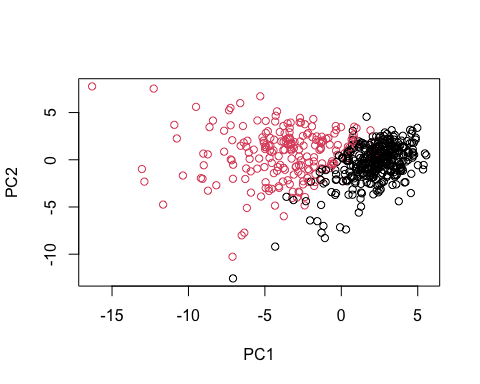


Q7) What stands out to you about this plot? Is it easy or difficult to understand? Why?

This plot is dogwater. Rubbish.

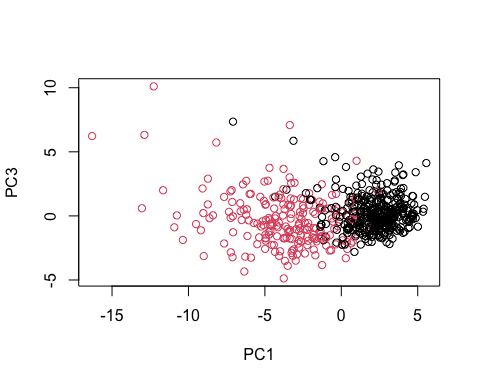
Lets try plotting it with a regular scatter plot colored by diagnosis

plot(wisc.pr$x[,1], wisc.pr$x[,2], col=diagnosis, xlab="PC1", ylab="PC2")



Q8) Generate a similar plot for principal components 1 and 3. What do you notice about these plots?

plot(wisc.pr$x[,1], wisc.pr$x[,3], col = diagnosis,   
 xlab = "PC1", ylab = "PC3")



These plots are easier to see some sort of pattern with regards to the diagnosis. The plot with PC2 looks a little cleaner cut and seems to separate the 2 diagnosis variables a little better. Also because the difference is seen among the x axis, it shows that PC1 is capturing the diagnosis variation

#Lets try and look at this on ggplot

df <- as.data.frame(wisc.pr$x)  
df$diagnosis <- diagnosis

library(ggplot2)

ggplot(df) + aes(PC1, PC2, col=df$diagnosis) + geom\_point()



#Variance explained

get the standard deviations from the wisc.pr output

pr.var <- wisc.pr$sdev^2  
head(pr.var)

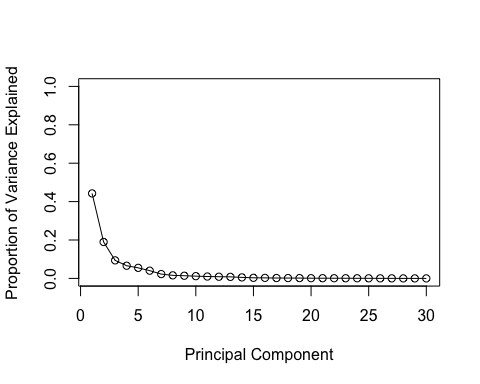
[1] 13.281608 5.691355 2.817949 1.980640 1.648731 1.207357

pve <- pr.var/ sum(pr.var)  
head(pve)

[1] 0.44272026 0.18971182 0.09393163 0.06602135 0.05495768 0.04024522

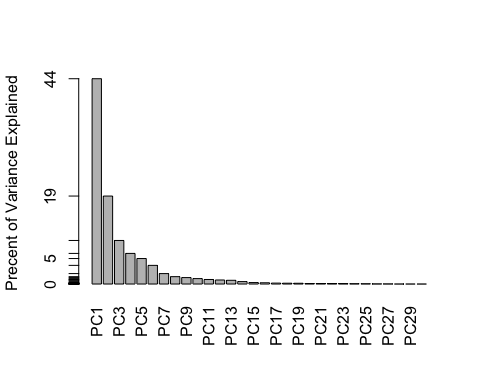
Plot variance explained for each principal component

plot(pve, xlab = "Principal Component",   
 ylab = "Proportion of Variance Explained",   
 ylim = c(0, 1), type = "o")



Alternative scree plot of the same data, note data driven y-axis

barplot(pve, ylab = "Precent of Variance Explained",  
 names.arg=paste0("PC",1:length(pve)), las=2, axes = FALSE)  
axis(2, at=pve, labels=round(pve,2)\*100 )



# Communicating PCA results

Q9) For the first principal component, what is the component of the loading vector (i.e. wisc.pr$rotation[,1]) for the feature concave.points\_mean?

wisc.pr$rotation[,1]

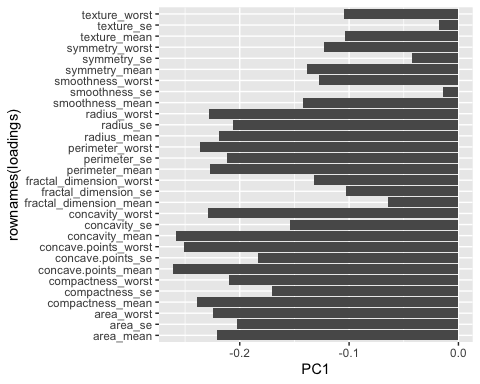
radius\_mean texture\_mean perimeter\_mean   
 -0.21890244 -0.10372458 -0.22753729   
 area\_mean smoothness\_mean compactness\_mean   
 -0.22099499 -0.14258969 -0.23928535   
 concavity\_mean concave.points\_mean symmetry\_mean   
 -0.25840048 -0.26085376 -0.13816696   
 fractal\_dimension\_mean radius\_se texture\_se   
 -0.06436335 -0.20597878 -0.01742803   
 perimeter\_se area\_se smoothness\_se   
 -0.21132592 -0.20286964 -0.01453145   
 compactness\_se concavity\_se concave.points\_se   
 -0.17039345 -0.15358979 -0.18341740   
 symmetry\_se fractal\_dimension\_se radius\_worst   
 -0.04249842 -0.10256832 -0.22799663   
 texture\_worst perimeter\_worst area\_worst   
 -0.10446933 -0.23663968 -0.22487053   
 smoothness\_worst compactness\_worst concavity\_worst   
 -0.12795256 -0.21009588 -0.22876753   
 concave.points\_worst symmetry\_worst fractal\_dimension\_worst   
 -0.25088597 -0.12290456 -0.13178394

Using concave.points\_mean you get:

wisc.pr$rotation["concave.points\_mean",1]

[1] -0.2608538

loadings <- as.data.frame(wisc.pr$rotation)  
ggplot(loadings)+ aes(PC1, rownames(loadings))+ geom\_col()



Q10) What is the minimum number of principal components required to explain 80% of the variance of the data?

5 PCs

# Hierarchical clustering

Scale the wisc.data data using the “scale()” function

data.scaled <- scale(wisc.data)

Calculate the (Euclidean) distances between all pairs of observations in the new scaled dataset

data.dist <- dist(data.scaled)

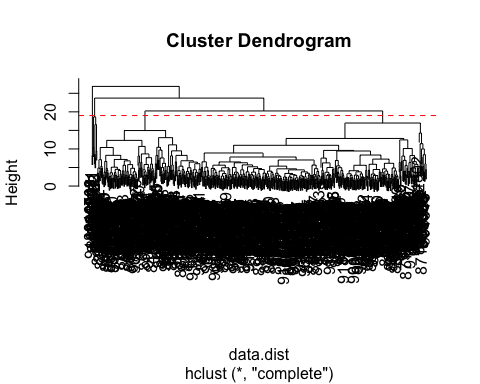
Create a hierarchical clustering model using complete linkage. Manually specify the method argument to hclust()

wisc.hclust <- hclust(data.dist)

Now we can plot this data Q11) Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters?

h=19 gives 4 clusters

plot(wisc.hclust)  
abline(h=19, col="red", lty=2)



#Selecting number of clusters

wisc.hclust.clusters <- cutree(wisc.hclust, k=4)  
table(wisc.hclust.clusters, diagnosis)

diagnosis  
wisc.hclust.clusters B M  
 1 12 165  
 2 2 5  
 3 343 40  
 4 0 2

Q12) Can you find a better cluster vs diagnoses match by cutting into a different number of clusters between 2 and 10?

Cutting into clusters that are higher than 2, doesn’t help our case because ideally we want 2 clusters, one that matches B and one that matches M.

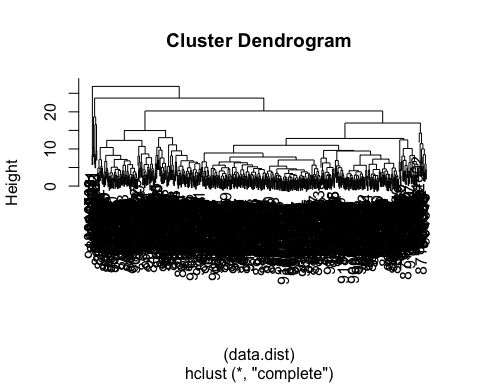
wisc.hclust.clusters <- cutree(wisc.hclust, k=10)  
table(wisc.hclust.clusters, diagnosis)

diagnosis  
wisc.hclust.clusters B M  
 1 12 86  
 2 0 59  
 3 0 3  
 4 331 39  
 5 0 20  
 6 2 0  
 7 12 0  
 8 0 2  
 9 0 2  
 10 0 1

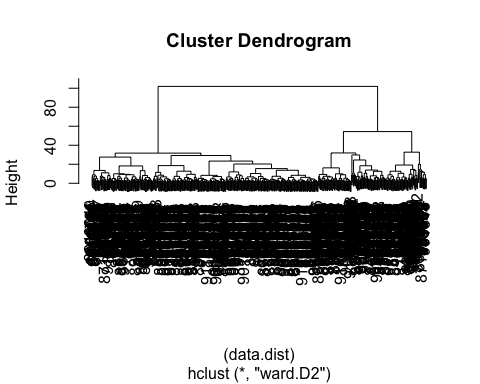
#Combine Methods

Q13. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning. Lets try out some methods and see which one looks best:

hclust.compete <- hclust((data.dist), method="complete")  
plot(hclust.compete)



hclust.ward.D2 <- hclust((data.dist), method="ward.D2")  
plot(hclust.ward.D2)

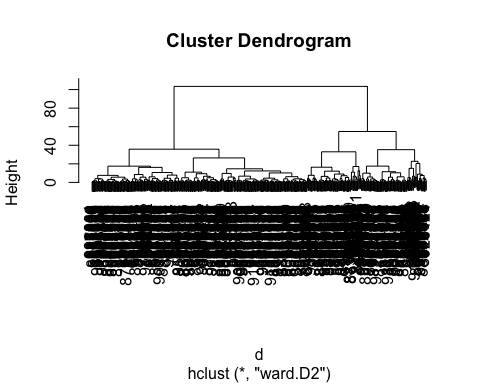


I like the ward.D2 because it gives me the biggest goal posts and more separation between clusters.

My PCA results were interesting as they showed a separation of M and B samples along PC1

I want to cluster my PCA results - that is use the wisc.pr$x as input to my hclust() You can try just taking the first 3 PCs because those are encompassing a lot of the variance. Also you can use method=“ward.D2”

d <- dist(wisc.pr$x[,1:3])  
wisc.pr.hclust <- hclust(d,method="ward.D2")  
plot(wisc.pr.hclust)



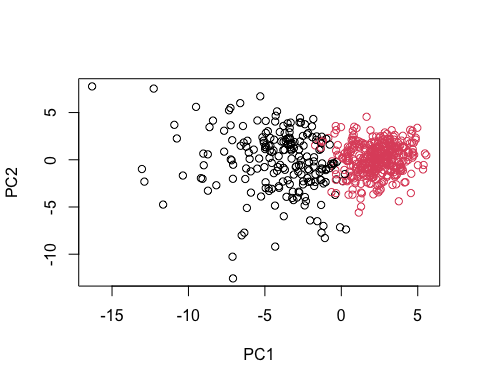
Lets cut into 2 groups/clusters

grps <- cutree(wisc.pr.hclust, k=2)  
table(grps)

grps  
 1 2   
203 366

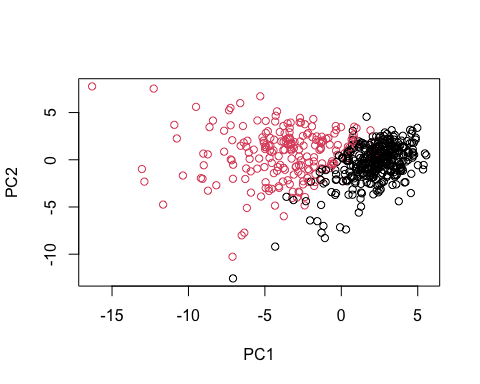
Now we can plot using this grps variable

plot(wisc.pr$x[,1:2], col=grps)



Now compare to the plot we made before

plot(wisc.pr$x[,1:2], col=diagnosis)



Q15) How well does the newly created model with four clusters separate out the two diagnoses?

Lets relevel the B and M for groups though so black is malignant and red is benign

g <- as.factor(grps)  
levels(g)

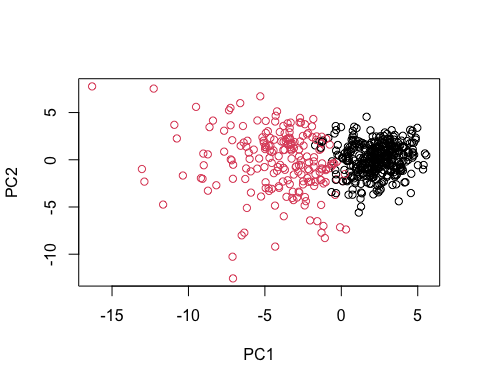
[1] "1" "2"

g <- relevel(g,2)  
levels(g)

[1] "2" "1"

Now we can replot

plot(wisc.pr$x[,1:2], col=g)



Q15) How well does the newly created model with four clusters separate out the two diagnoses?

table(g, diagnosis)

diagnosis  
g B M  
 2 333 33  
 1 24 179

We can text the accuracy by checking for false positive Malignant cases you get: For our prediction we get in group 2 (Benign) 33 cases that were actually scored as Malignant by the experts so there is a 6.4% chance of giving a false positive.

33/(333+179)

[1] 0.06445312