mini-project class 8

```
# Save your input data file into your Project directory
fna.data <- "WisconsinCancer.csv"

# Complete the following code to input the data and store as wisc.df
wisc.df <- read.csv(fna.data, row.names=1)

# We can use -1 here to remove the first column
wisc.data <- wisc.df[,-1]

# Create diagnosis vector for later
diagnosis <- factor(wisc.df[,1])
```

Q1. How many observations are in this dataset?

```
nrow(wisc.data)
```

```
[1] 569
```

569 observations (rows). ## Q2. How many of the observations have a malignant diagnosis?

```
length(grep("M", diagnosis))
```

```
[1] 212
```

212 observations. ## Q3. How many variables/features in the data are suffixed with _mean?

```
table(endsWith(colnames(wisc.data), "_mean"))
```

```
FALSE  TRUE
   20    10
```

10 variables.

```
# Check column means and standard deviations
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
```

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

```
            radius_mean            texture_mean          perimeter_mean
           3.524049e+00            4.301036e+00            2.429898e+01
              area_mean         smoothness_mean        compactness_mean
           3.519141e+02            1.406413e-02            5.281276e-02
         concavity_mean     concave.points_mean           symmetry_mean
           7.971981e-02            3.880284e-02            2.741428e-02
 fractal_dimension_mean               radius_se              texture_se
           7.060363e-03            2.773127e-01            5.516484e-01
           perimeter_se                 area_se           smoothness_se
           2.021855e+00            4.549101e+01            3.002518e-03
         compactness_se            concavity_se       concave.points_se
           1.790818e-02            3.018606e-02            6.170285e-03
            symmetry_se    fractal_dimension_se            radius_worst
           8.266372e-03            2.646071e-03            4.833242e+00
          texture_worst         perimeter_worst              area_worst
           6.146258e+00            3.360254e+01            5.693570e+02
       smoothness_worst       compactness_worst         concavity_worst
           2.283243e-02            1.573365e-01            2.086243e-01
   concave.points_worst          symmetry_worst fractal_dimension_worst
           6.573234e-02            6.186747e-02            1.806127e-02
```

Some values differ by several orders of 10. Scaling is needed.

```
scaledData <- scale(wisc.data)

# Perform PCA on wisc.data by completing the following code
wisc.pr <- prcomp(scaledData)

# Look at summary of results
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
```

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

44.27% of the variance is captured by PC1.

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

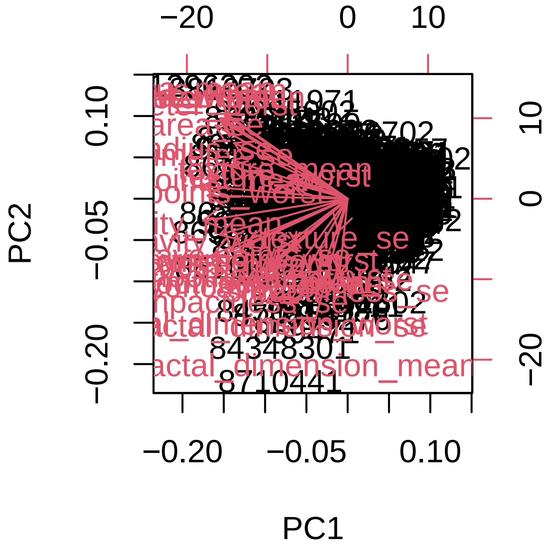
3– PC1, PC2, and PC3. They describe 72.64% of the original variance. PC1 and PC2 only comprise 63.24%.

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

7– PC1 through PC7. They comprise 91.01% of the original variance.

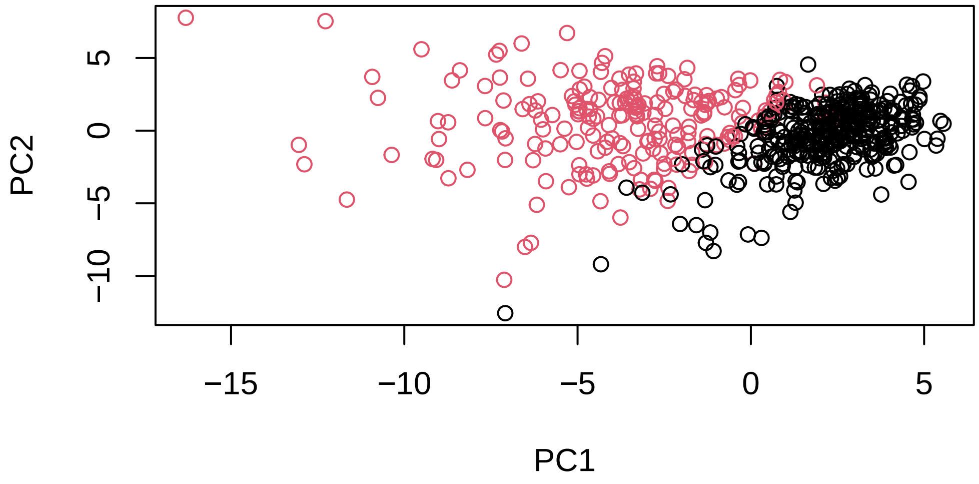
Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

```
biplot(wisc.pr)
```



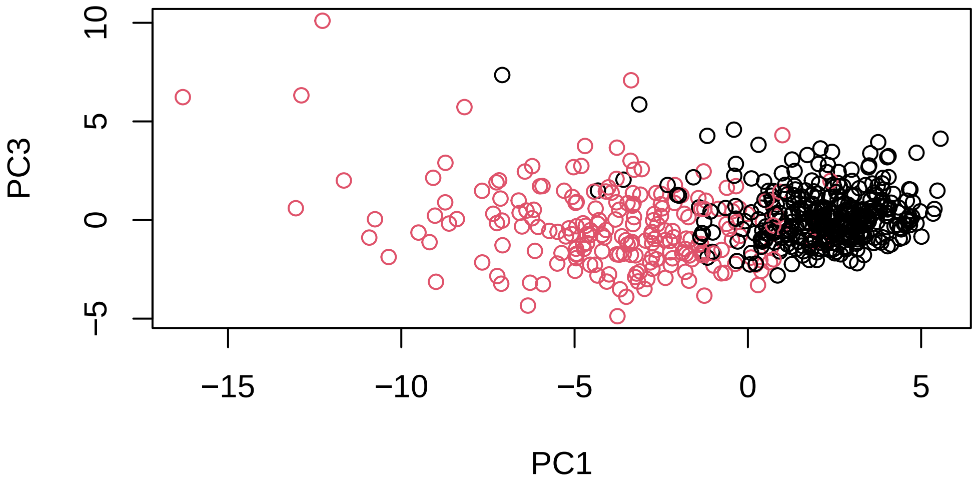
Yeah, this plot sucks. It’s extremely difficult to read with all the labels and data overlapping.I would have no idea what I was looking at out of context.

```
# Scatter plot observations by components 1 and 2
plot( wisc.pr$x , col = diagnosis ,
     xlab = "PC1", ylab = "PC2")
```



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

```
# Repeat for components 1 and 3
plot(wisc.pr$x[, 1], wisc.pr$x[,3], col = diagnosis,
     xlab = "PC1", ylab = "PC3")
```

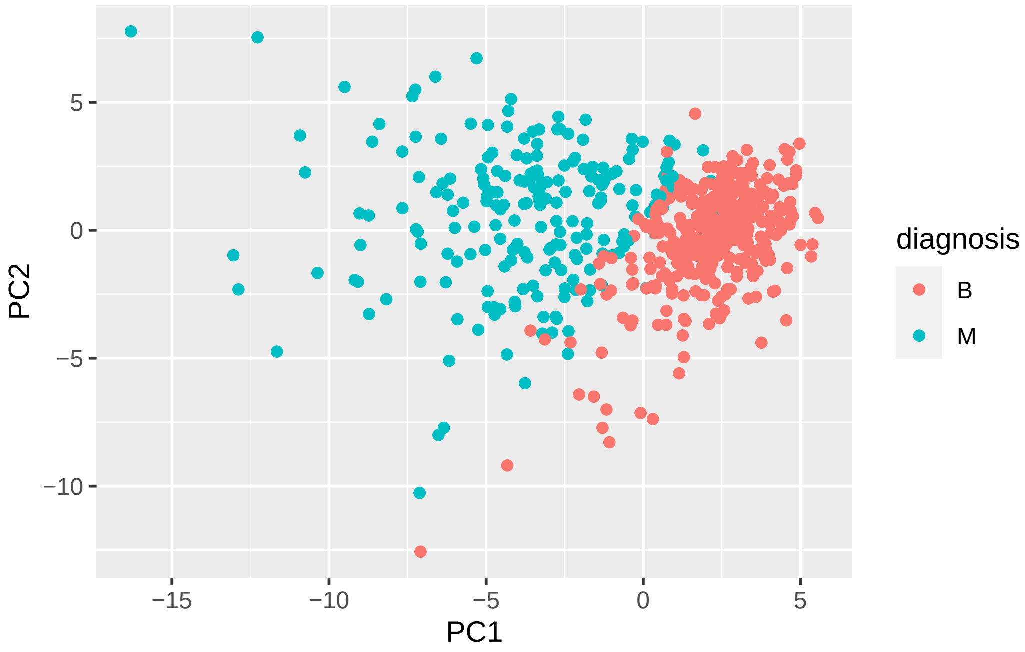


There's a really clear line of distinction between the red and black in PC1 vs. PC2, but against PC3 there starts to be a lot more overlap, with the red much more spread out underneath black. They're both still ugly though.

```
# Create a data.frame for ggplot
df <- as.data.frame(wisc.pr$x)
df$diagnosis <- diagnosis

# Load the ggplot2 package
library(ggplot2)

# Make a scatter plot colored by diagnosis
ggplot(df) +
  aes(PC1, PC2, col=diagnosis) +
  geom_point()
```

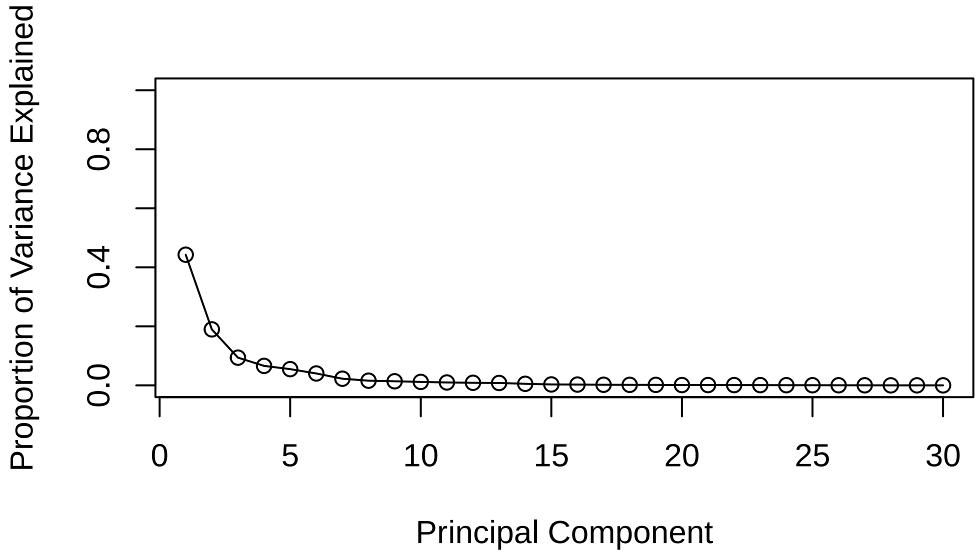


```r
# Calculate variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

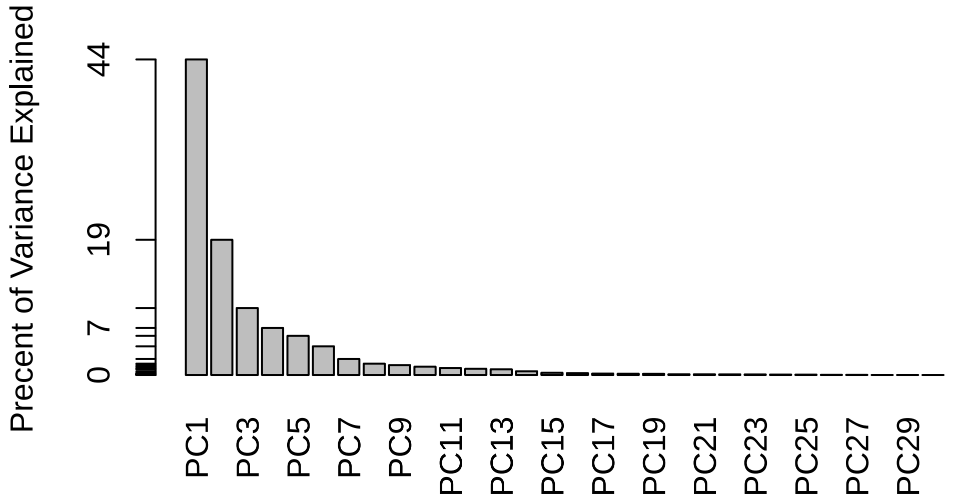
```
[1] 13.281608  5.691355  2.817949  1.980640  1.648731  1.207357
```

```r
# Variance explained by each principal component: pve
pve <- pr.var / 30

# 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 )
```



```
## OPTIONAL: factoextra package; ggplot based graph
#install.packages("factoextra")
#library(factoextra)
#fviz_eig(wisc.pr, addlabels = TRUE)
```

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
```

concave.points_mean = -0.26085376. ## Q10. What is the minimum number of principal components required to explain 80% of the variance of the data? 5

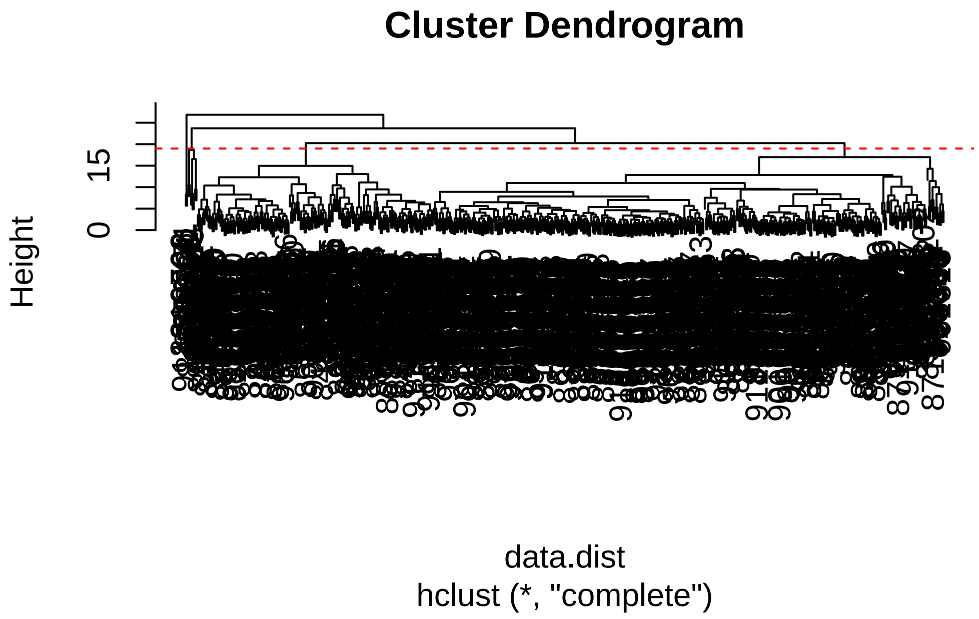
```
# Scale the wisc.data data using the "scale()" function
data.scaled <- scale(wisc.data)

data.dist <- dist(data.scaled)

wisc.hclust <- hclust(data.dist, method="complete")
```

Q11. Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters?

```
plot(wisc.hclust)
abline(a=19, b=0, col="red", lty=2)
```



Between 18 and 19, closer to 19. About 18.7 (closest to 18.637)

```
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?

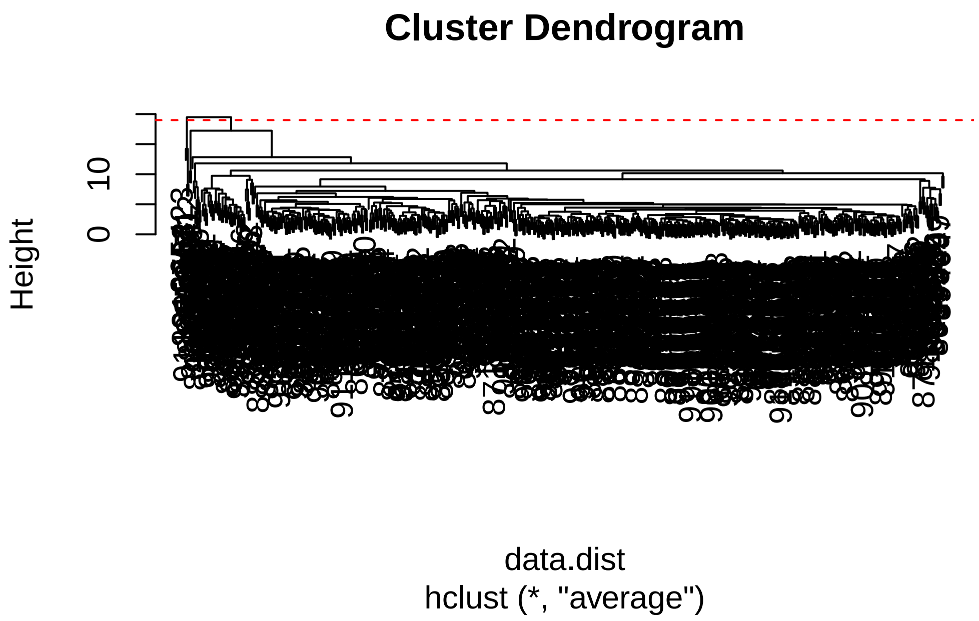
```
wisc.hclust.clustersTemp <- cutree(wisc.hclust, k=7)
table(wisc.hclust.clustersTemp, diagnosis)
```

```
                        diagnosis
wisc.hclust.clustersTemp   B   M
```

```
1  12 165
2   0   3
3 331  39
4   2   0
5  12   1
6   0   2
7   0   2
```

Q13. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning.

```r
wisc.hclustTemp <- hclust(data.dist, method="average")
plot(wisc.hclustTemp)
abline(a=19, b=0, col="red", lty=2)
```



I prefer “average” because it allows for easy visualization of the relationships over cluster number– going downwards in height, the graph extends with new clusters and can be read left to right. It’s intuitive.

```
wisc.km <- kmeans(wisc.data, centers=2, nstart= 20)
table(wisc.km$cluster, diagnosis)
```

```
   diagnosis
      B   M
  1 356  82
  2   1 130
```

```
table(wisc.hclust.clusters, wisc.km$cluster)
```

```
wisc.hclust.clusters   1   2
                   1  68 109
                   2   5   2
                   3 365  18
                   4   0   2
```

Q14. How well does k-means separate the two diagnoses? How does it compare to your hclust results?

It does a much better job than just the hclust alone.

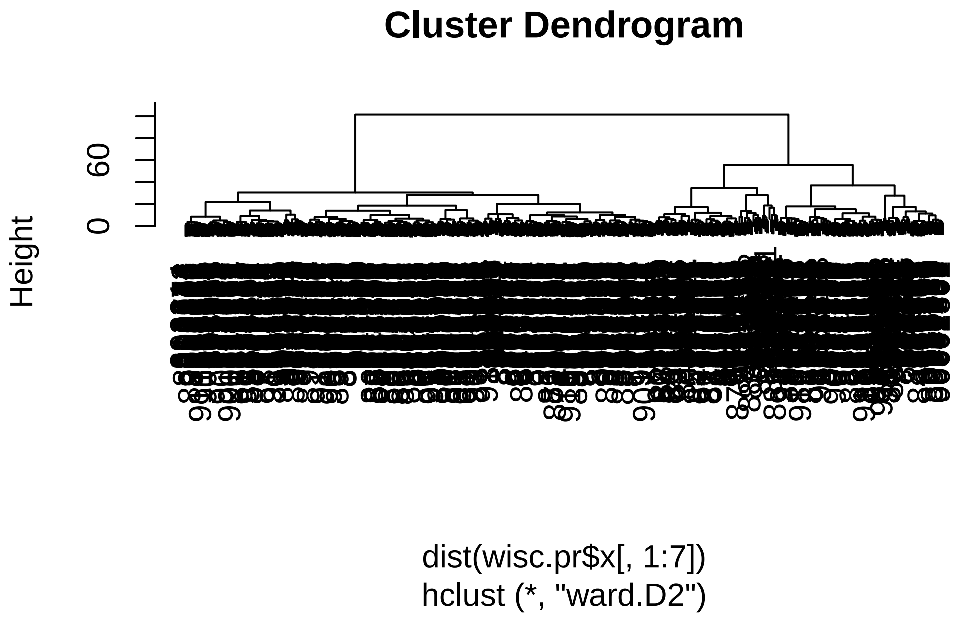
```
wisc.pr.hclust <- hclust(dist(wisc.pr$x[,1:7]), method="ward.D2")
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
grps
  1   2
216 353
```

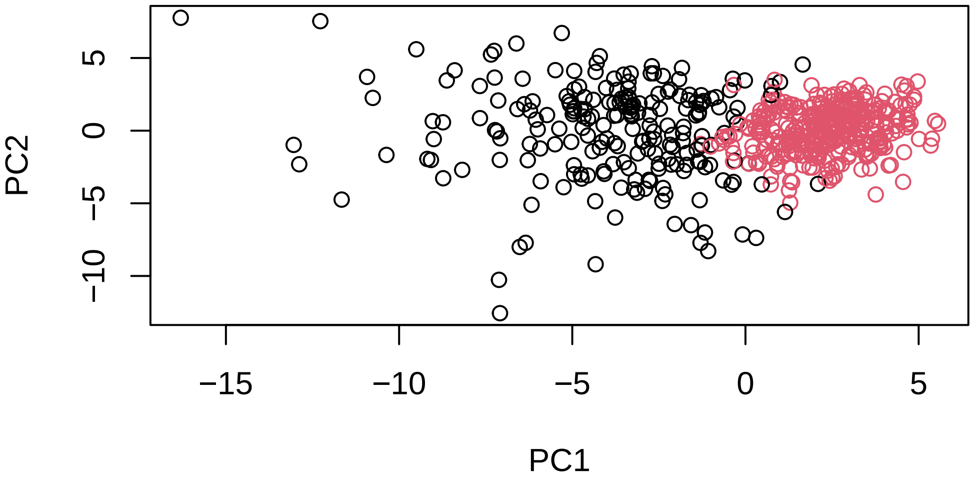
```
table(grps, diagnosis)
```

```
    diagnosis
grps   B   M
   1  28 188
   2 329  24
```

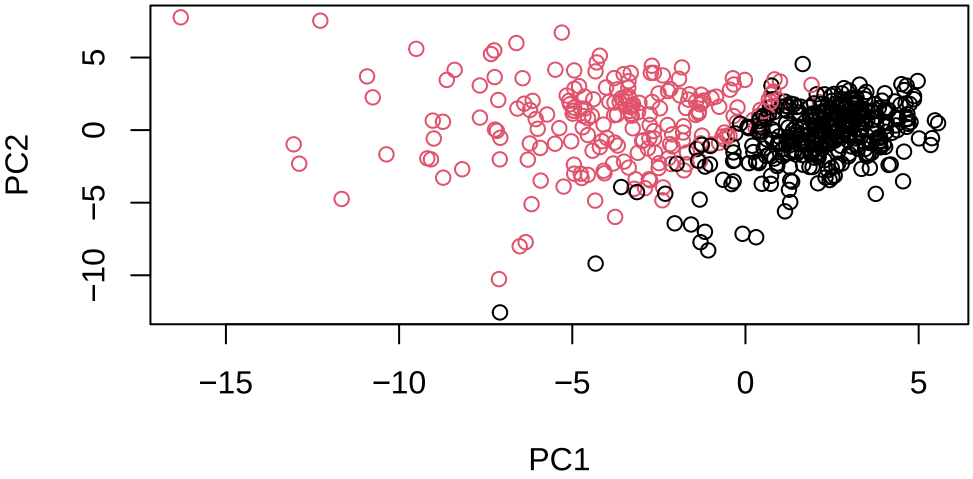
```
plot(wisc.pr.hclust)
```



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



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



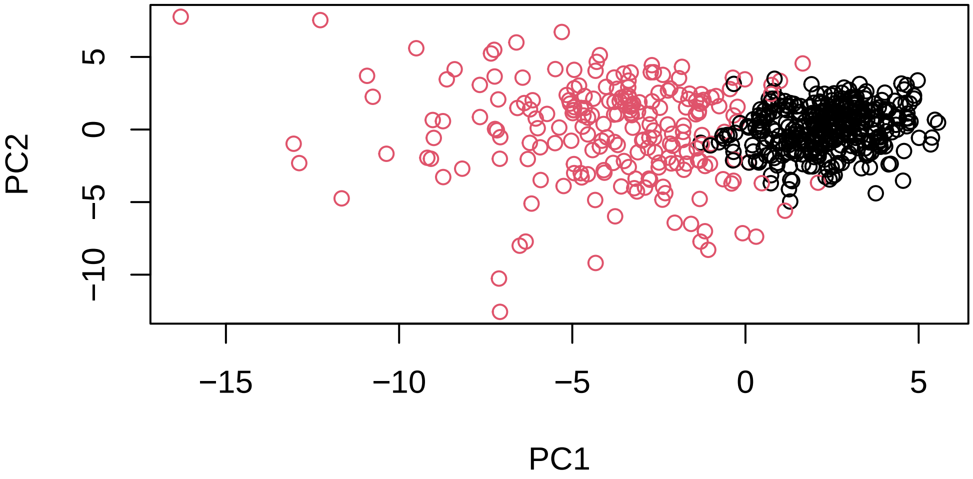
```
g <- as.factor(grps)
levels(g)
```

```
[1] "1" "2"
```

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

```
[1] "2" "1"
```

```
# Plot using our re-ordered factor
plot(wisc.pr$x[,1:2], col=g)
```



```
## Use the distance along the first 7 PCs for clustering i.e. wisc.pr$x[, 1:7]
wisc.pr.hclust <- hclust(dist(wisc.pr$x[,1:7]), method="ward.D2")
wisc.pr.hclust.clusters <- cutree(wisc.pr.hclust, k=2)
```

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

```
# Compare to actual diagnoses
table(wisc.pr.hclust.clusters, diagnosis)
```

```
                       diagnosis
wisc.pr.hclust.clusters   B   M
                      1  28 188
                      2 329  24
```

Very well!! We still have some to go, though. The malignant cluster could be better separated.

Q16. How well do the k-means and hierarchical clustering models you created in previous sections (i.e. before PCA) do in terms of separating the diagnoses? Again, use the table() function to compare the output of each model (wisc.km$cluster and wisc.hclust.clusters) with the vector containing the actual diagnoses.

```
table(wisc.km$cluster, diagnosis)
```

```
   diagnosis
      B   M
  1 356  82
  2   1 130
```

```
table(wisc.hclust.clusters, diagnosis)
```

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

I prefer wisc.km$cluster. It has the best separation for the malignant and benign clusters.

Q17. Which of your analysis procedures resulted in a clustering model with the best specificity? How about sensitivity?

```
#model 1: wisc.km$cluster
sens1 <- 130/(130+82)
sens1
```

```
[1] 0.6132075
```

```
spec1 <- 356/(356+1)
spec1
```

```
[1] 0.9971989
```

```
# model 2: wisc.hclust.clusters
sens2 <- (165+5+2)/((165+5+2)+40)
sens2
```

```
[1] 0.8113208
```

```
spec2 <- 343/(343+40)
spec2
```

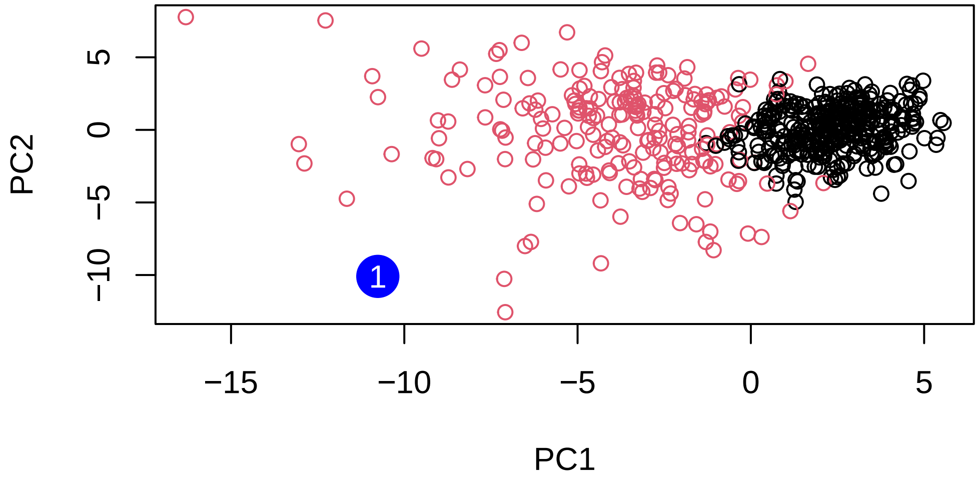
```
[1] 0.8955614
```

wisc.km$cluster produces the highest specificity, but wisc.hclust.clusters has the higher sensititivity.

```
#url <- "new_samples.csv"
url <- "https://tinyurl.com/new-samples-CSV"
new <- read.csv(url)
npc <- predict(wisc.pr, newdata=new)
npc
```

```
           PC1        PC2        PC3       PC4      PC5       PC6        PC7
[1,] -10.76452 -10.093978 -0.5897994 -4.164748 10.61922 -1.630738 0.03566861
[2,] -18.09606  -9.967098 -2.1549431 -4.006848  6.69687 -2.034714 1.25088149
           PC8       PC9     PC10       PC11     PC12       PC13      PC14
[1,] 0.7308658 -1.580861 3.166451 -0.7167150 3.850569 -0.8259764 1.0195729
[2,] 0.6308585 -1.155629 3.608207 -0.3405375 2.288732 -0.3976672 0.1347203
         PC15      PC16      PC17      PC18     PC19      PC20      PC21
[1,] 3.735687 -4.068783 1.0877034 0.9985959 1.022760 -2.430215 -1.295749
[2,] 3.543905 -3.749616 0.7613603 1.1763217 1.366702 -2.609643 -1.541050
          PC22       PC23      PC24       PC25      PC26      PC27       PC28
[1,] -1.348026 -0.7388274 -1.083000 -0.4220831 -1.892993 -1.176056 0.05527974
[2,] -1.424290 -0.7591376 -1.439202 -0.6508838 -1.981711 -1.397390 0.18112357
          PC29       PC30
[1,] 0.2658028 0.05162840
[2,] 0.2842191 0.02734355
```

```
plot(wisc.pr$x[,1:2], col=g)
points(npc[,1], npc[,2], col="blue", pch=16, cex=3)
text(npc[,1], npc[,2], c(1,2), col="white")
```



I don’t know why it’s labelled itself that way.

Q18. Which of these new patients should we prioritize for follow up based on your results?

Patient group 2, with the more extreme PC values,