Class09 Mini-Project

Leyna Nguyen (PID: A15422197)

10/26/2021

1. Exploratory Data Analysis

```
# 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 that contains the diagnoses
wisc.data <- wisc.df[,-1]

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

```
##   [1] M M M M M M M M M M M M M M M M M B B B M M M M M M M M M M M M M
##  [38] B M M M M M M M B M B B B B B M M B M M B B B B M B M M B B B B M B M M
##  [75] B M B M M B B B M M B M M M B B B M B B M M B B B M M B B B B M B B M B B
## [112] B B B B B B M M M B M M B B B M M B M B M M B M M B B M B B M B B B B M B
## [149] B B B B B B B B M B B B B M M B M B B M M B B M M B B B B M B B M M M B M
## [186] B M B B B M B B M M B M M M M B M M M B M B M B B M B M M M M B B M M B B
## [223] B M B B B B B M M B B M B B M M B M B B B B M B B B B B M B M M M M M M M
## [260] M M M M M M M B B B B B B M B M B B M B B M B M M B B B B B B B B B B B
## [297] B M B B M B M B B B B B B B B B B B B B B M B B B M B M B B B B M M M B B
## [334] B B M B M B M B B B M B B B B B B B M M M B B B B B B B B B B M M B M M
## [371] M B M M B B B B B M B B B B B M B B B M B B M B B M M B B B B B B M B B B B B B
## [408] B M B B B B B M B B M B B B B B B B B B B B B M B M M B M B B B B B M B B
## [445] M B M B B M B M B B B B B B B B M M B B B B B B M B B B B B B B B B M B
## [482] B B B B B B M B M B B M B B B B B B M M B M B M B B B B B M B B M B M B M M
## [519] B B B M B B B B B B B B B B B M B M M B B B B B B B B B B B B B B B B B B
## [556] B B B B B B B M M M M M M B
## Levels: B M
```

Q1. How many observations are in this dataset?

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

```
## [1] 569
```

There are 569 observations.

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

```
table(diagnosis)
```

```
## diagnosis
##   B   M
## 357 212
```

212 of the 569 observations have a malignant diagnosis.

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

```
length(grep("_mean", colnames(wisc.df)))
```

```
## [1] 10
```

There are 10 variables/columns with the suffix "_mean".

2. Principal Component Analysis

```
# 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
```

```
# Perform PCA on wisc.data. We need to do the scale=TRUE argument in this case as the column data are o
wisc.pr <- prcomp(wisc.data, scale.=TRUE)
```

```
# 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 original variance is captured by PC1.

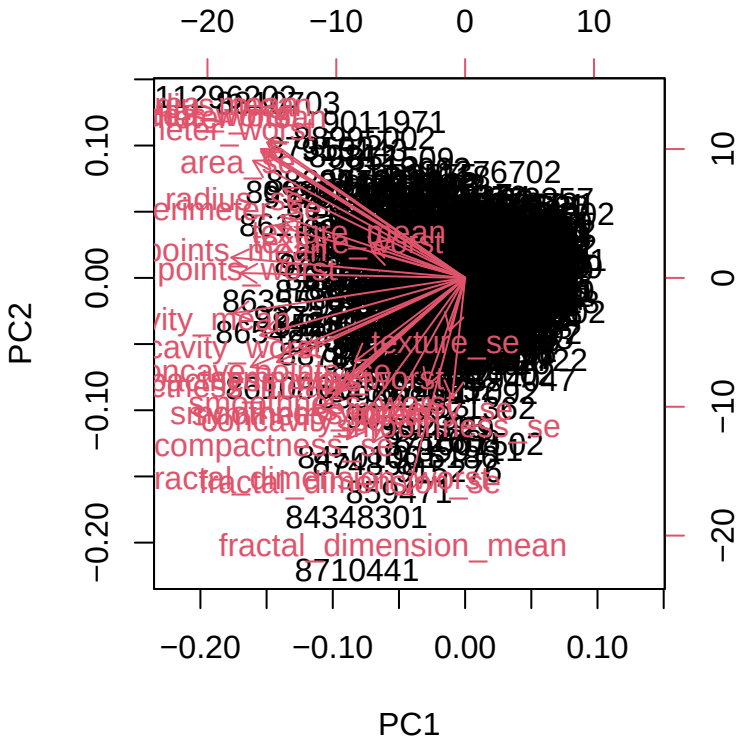
> Q5. How many principal components (PCs) are required to describe at least 70% of the original variance in the data?

3 PCs are needed to describe at least 70% of the original variance in the data. You would need PC1 (44.27%), PC2 (18.97%), and PC3 (9.393%).

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

7 PCs are required to describe at least 90% of the original variance in the data. You would need PC1 (44.27%), PC2 (18.97%), PC3 (9.393%), PC4 (6.602%), PC5 (5.496%), PC6 (4.025%), and PC7 (2.251%).

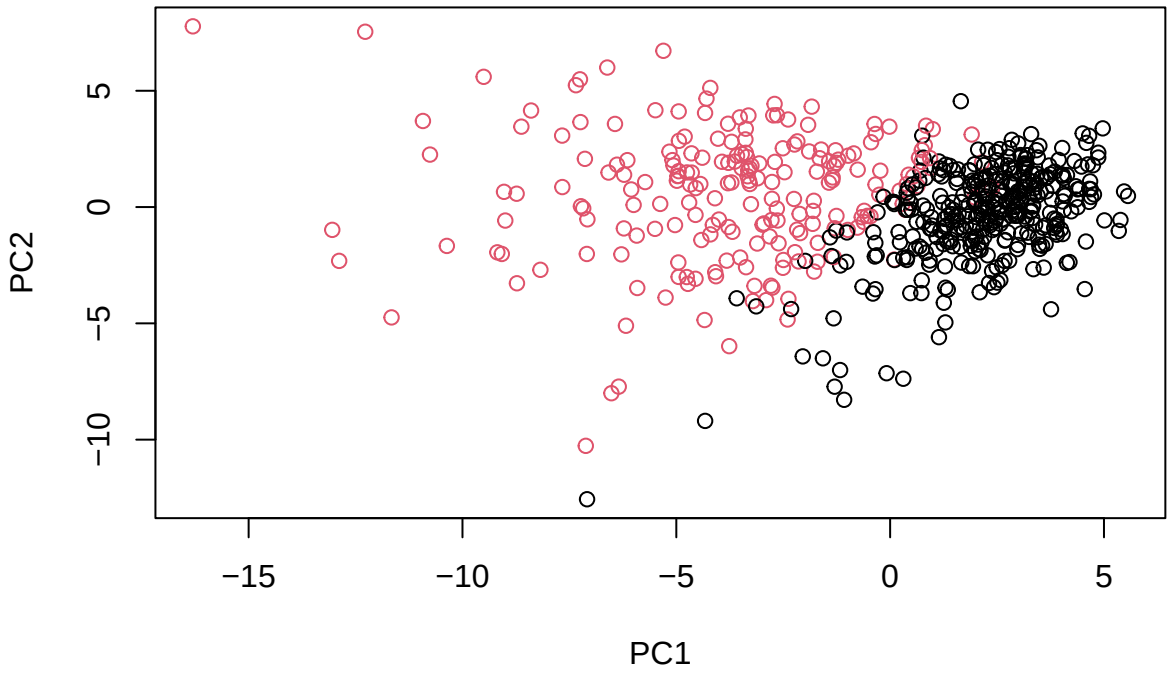
```
# Let's make a biplot for wisc.pr
biplot(wisc.pr)
```



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

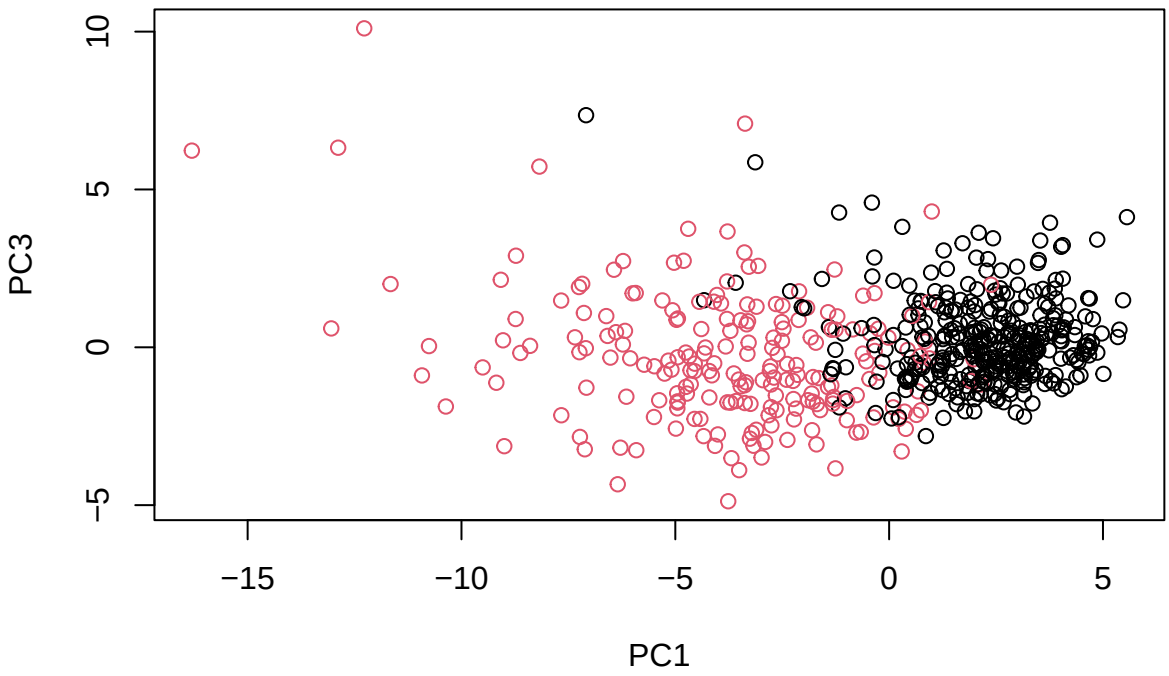
What stands out to me is that there are so many names that are overlapping each other which makes it hard to discern what label is for what. This plot includes the row names, or the patients' IDs, in the plot and they all are stacked on top of one another, making it very hard to interpret the graph.

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



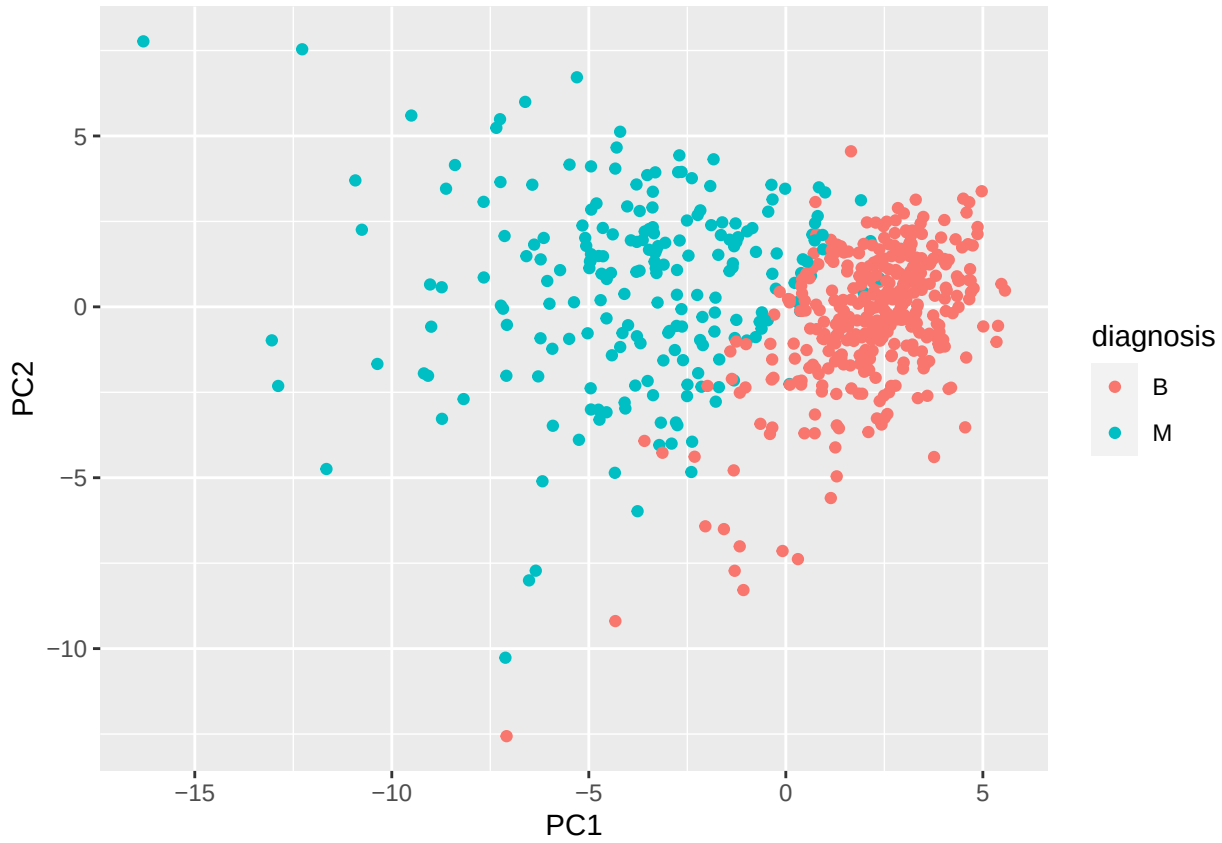
Q8. Generate a similar plot for principal components 1 and 3. What do you notice about these plots?

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



Comparing these plots to each other, the second plot of PC1 and PC3 has more overlap between the subgroups since PC3 accounts for less variance than PC2, meanwhile the first plot of PC1 and PC2 has less overlap between the subgroups.

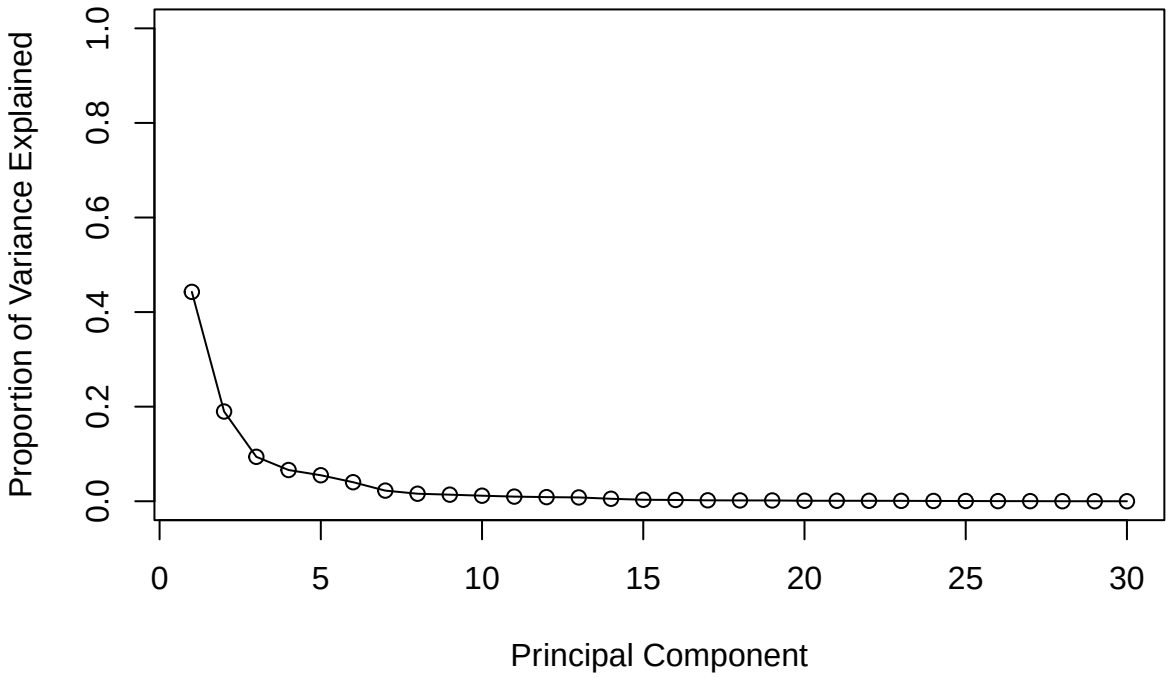
```
# Create a data.frame for ggplot
df <- as.data.frame(wisc.pr$x)
df$diagnosis <- diagnosis

# Load the ggplot2 package
library(ggplot2)

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



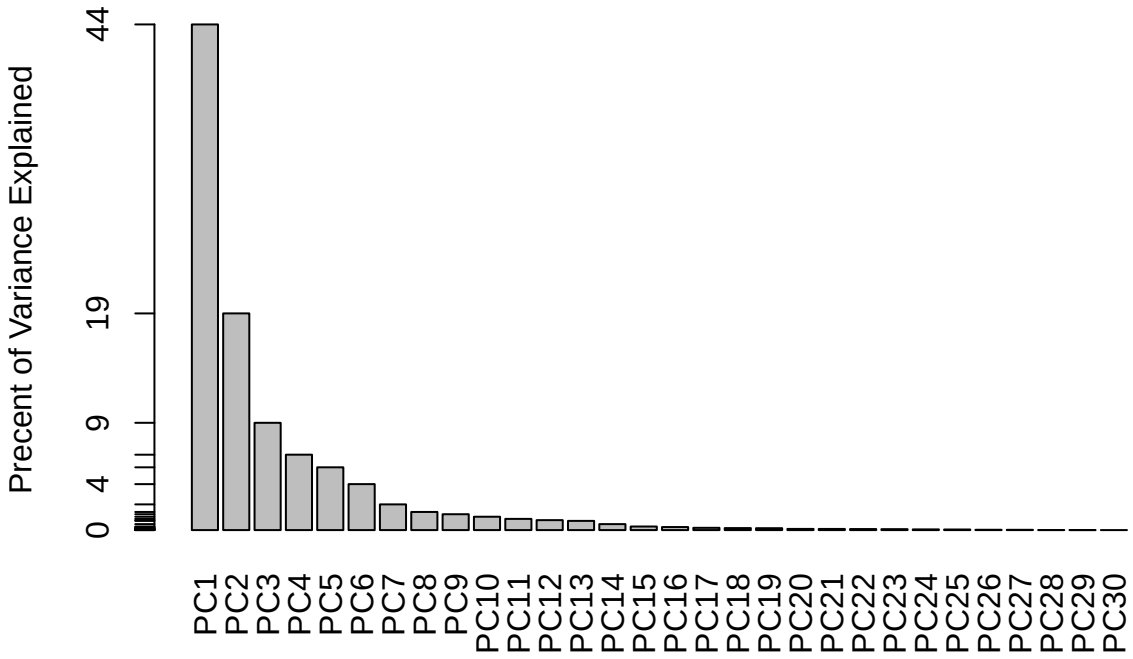

```
# Calculate variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

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

```
# Variance explained by each principal component: pve
pve <- pr.var/sum(pr.var)

# Plot variance explained for each principal component
plot(pve, xlab = "Principal Component",
     ylab = "Proportion of Variance Explained",
     ylim = c(0, 1), type = "o")
```



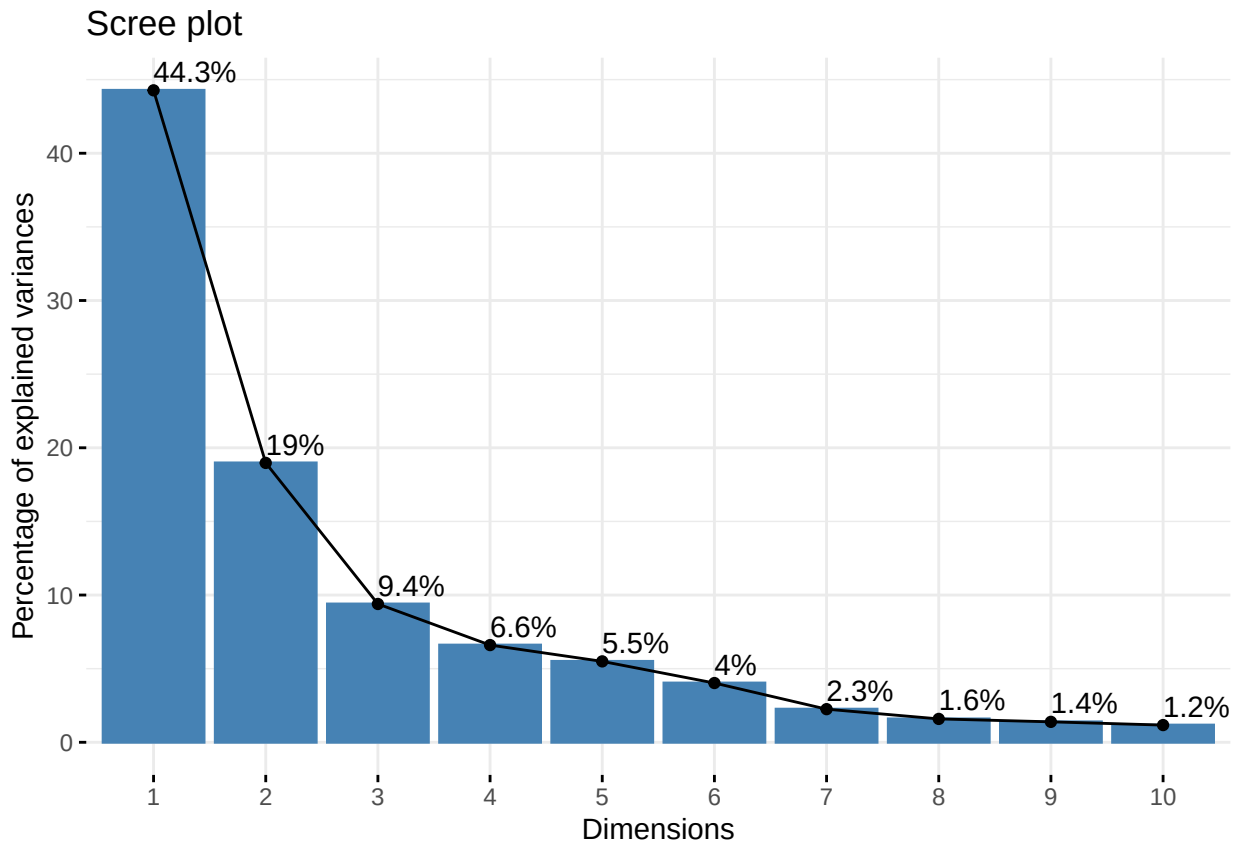

```
# Alternative scree plot of the same data, note data driven y-axis
barplot(pve, ylab = "Precent of Variance Explained",
     names.arg=paste0("PC",1:length(pve)), las=2, axes = FALSE)
axis(2, at=pve, labels=round(pve,2)*100 )
```




```
## ggplot based graph
#install.packages("factoextra")
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
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["concave.points_mean", 1]
```

```
## [1] -0.2608538
```

The component of the loading vector for the feature concave.points_mean is -0.26085376.

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

```
summary(wisc.pr)
```

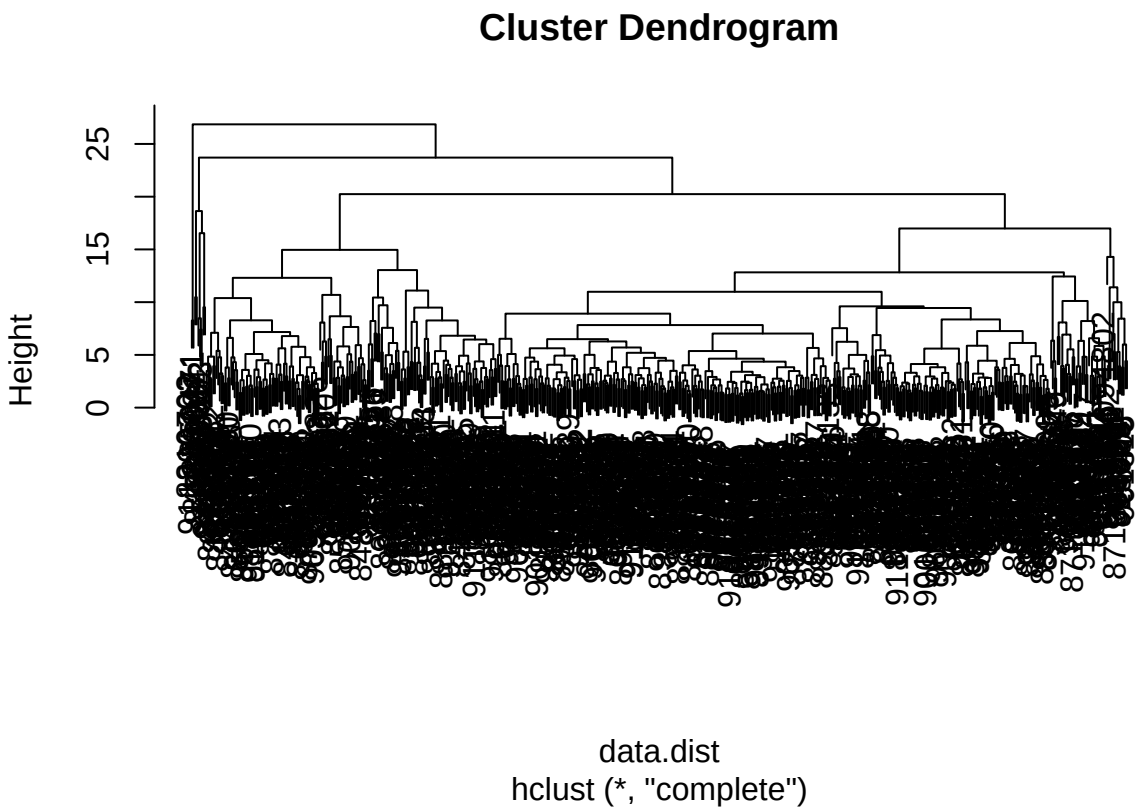
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172
## Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251
## Cumulative Proportion  0.4427 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
```

The minimum number is 5. You would need PC1 (44.27%), PC2 (18.97%), PC3 (9.393%), PC4 (6.602%), and PC5 (5.496%).

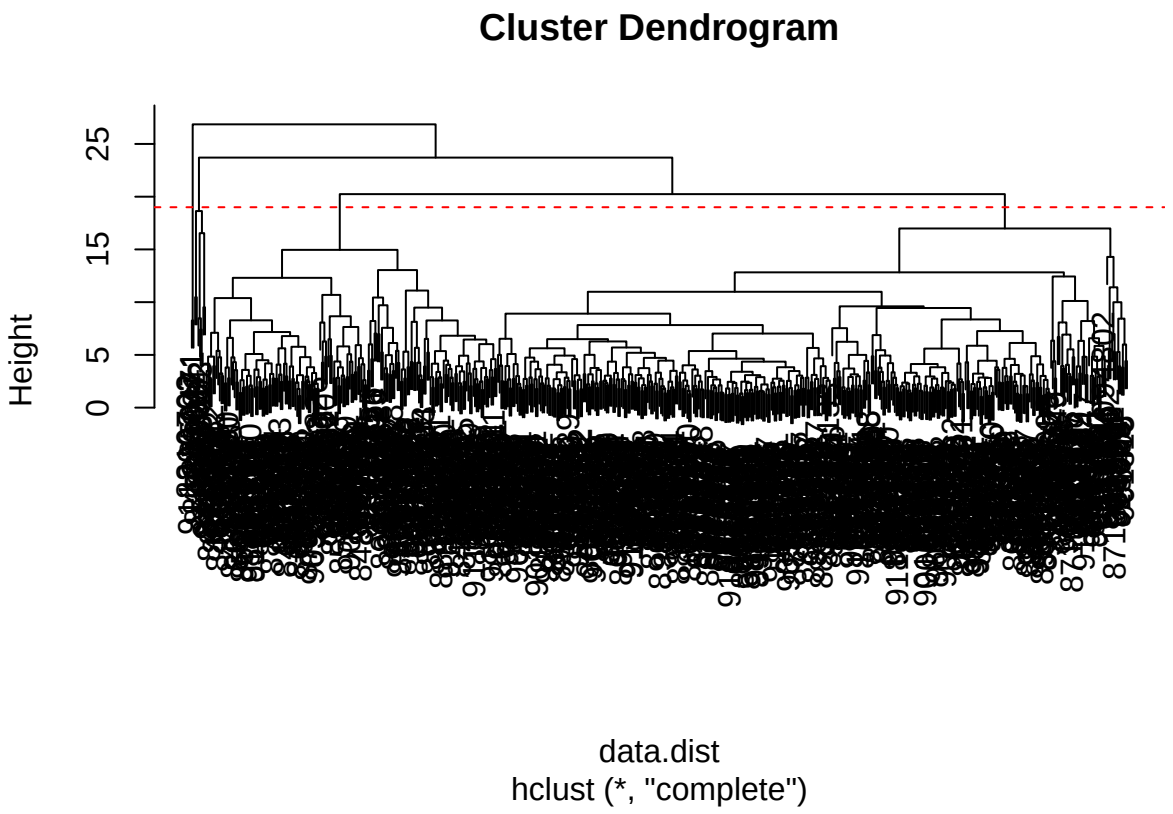
3. Hierarchical Clustering

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

# Calculate the (Euclidean) distances between all pairs of observations
data.dist <- dist(data.scaled)

# Create a hierarchical clustering model using complete linkage
wisc.hclust <- hclust(data.dist)
plot(wisc.hclust)
```



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

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



The clustering model has 4 clusters at height h=19.

```
wisc.hclust.clusters <- cutree(wisc.hclust, h=19)

# Compare to diagnosis results
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?

```
grps2 <- cutree(wisc.hclust, k=2)
table(grps2, diagnosis)
```

```
##      diagnosis
## grps2   B   M
##     1 357 210
##     2   0   2
```

```
grps3 <- cutree(wisc.hclust, k=3)
table(grps3, diagnosis)
```

```
##      diagnosis
## grps3   B   M
##     1 355 205
##     2   2   5
##     3   0   2
```

```
grps5 <- cutree(wisc.hclust, k=5)
table(grps5, diagnosis)
```

```
##      diagnosis
## grps5   B   M
##     1  12 165
##     2   0   5
##     3 343  40
##     4   2   0
##     5   0   2
```

```
grps6 <- cutree(wisc.hclust, k=6)
table(grps6, diagnosis)
```

```
##      diagnosis
## grps6   B   M
##     1  12 165
##     2   0   5
##     3 331  39
##     4   2   0
##     5  12   1
##     6   0   2
```

```
grps7 <- cutree(wisc.hclust, k=7)
table(grps7, diagnosis)
```

```
##      diagnosis
## grps7   B   M
##     1  12 165
##     2   0   3
##     3 331  39
##     4   2   0
##     5  12   1
##     6   0   2
##     7   0   2
```

```
grps8 <- cutree(wisc.hclust, k=8)
table(grps8, diagnosis)
```

```
##      diagnosis
## grps8   B   M
```

```
##     1  12  86
##     2   0  79
##     3   0   3
##     4 331  39
##     5   2   0
##     6  12   1
##     7   0   2
##     8   0   2
```

```
grps9 <- cutree(wisc.hclust, k=9)
table(grps9, diagnosis)
```

```
##      diagnosis
## grps9   B   M
##     1  12  86
##     2   0  79
##     3   0   3
##     4 331  39
##     5   2   0
##     6  12   0
##     7   0   2
##     8   0   2
##     9   0   1
```

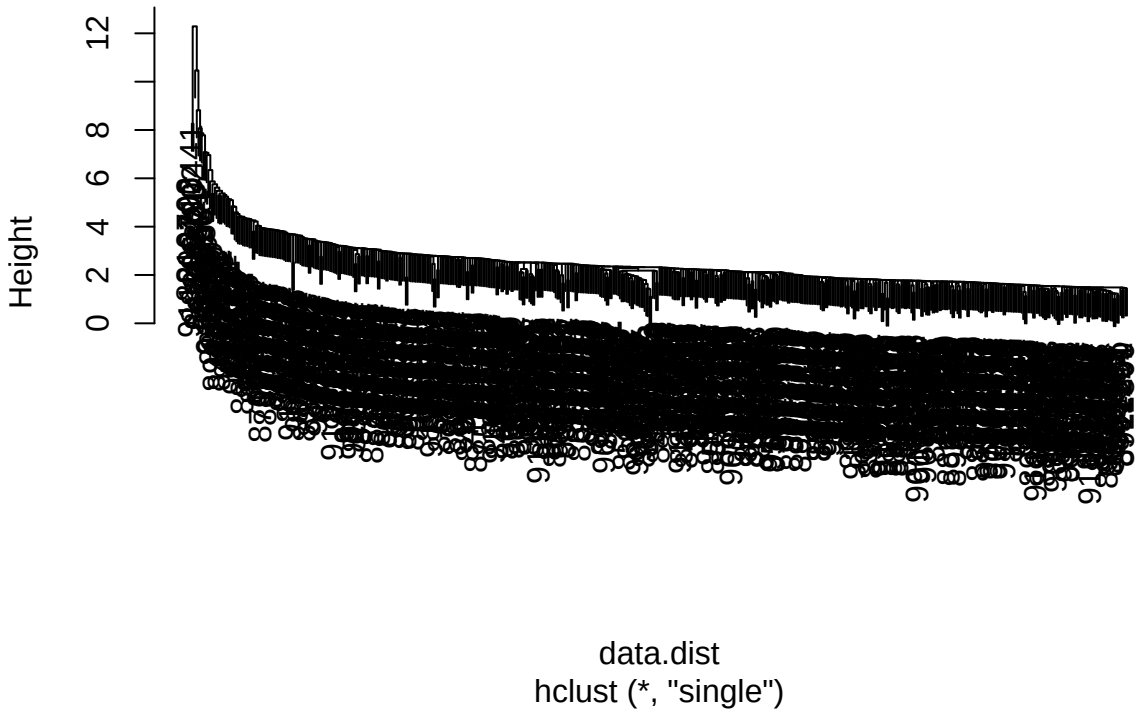
```
grps10 <- cutree(wisc.hclust, k=10)
table(grps10, diagnosis)
```

```
##       diagnosis
## grps10   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
```

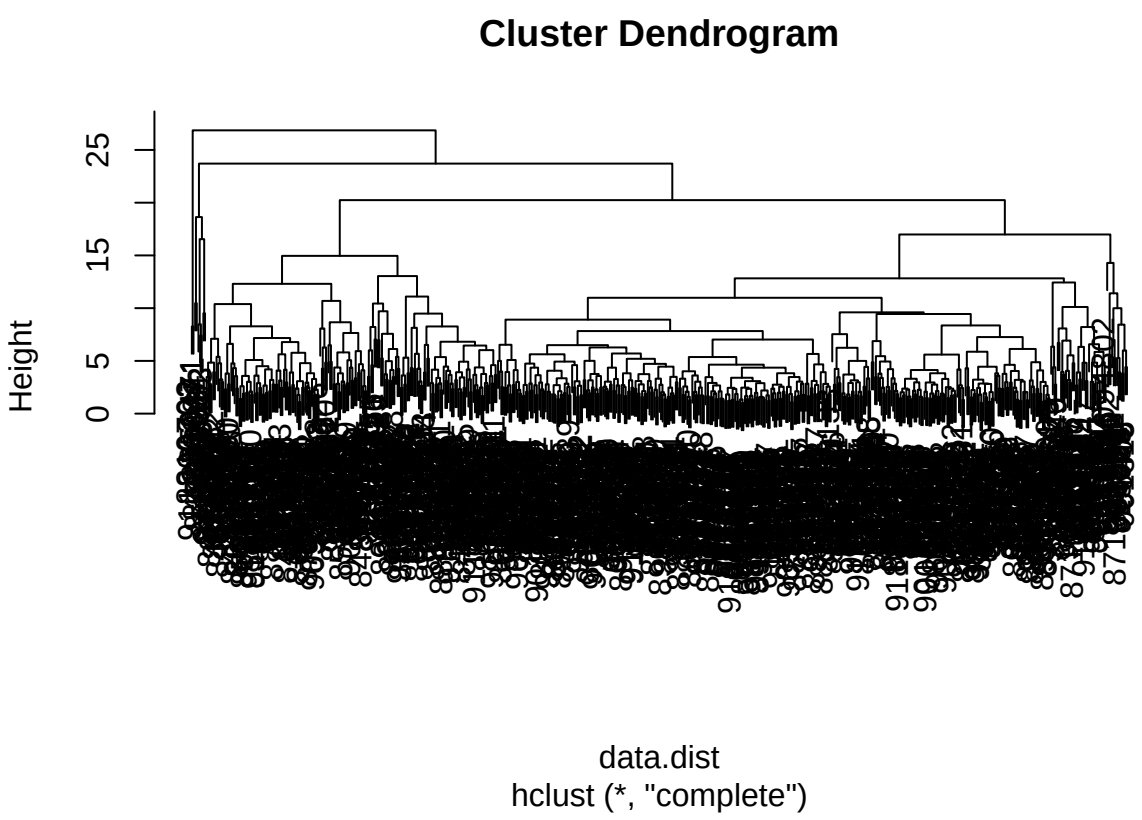
Cutting the model into 4 clusters seems to work best, but cutting the model into 5, 6, and 7 clusters work almost just as well.

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

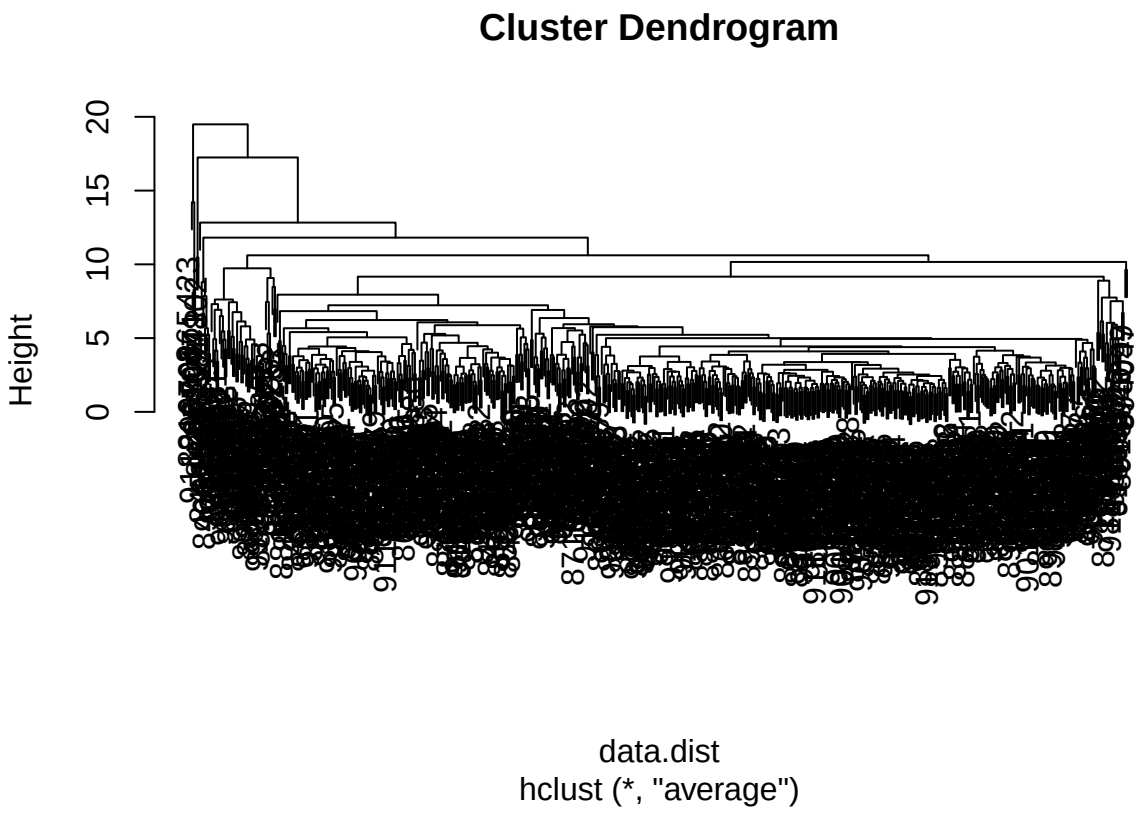
```
# Using the "single" method
wisc.hclust.single <- hclust(data.dist, method="single")
plot(wisc.hclust.single)
```



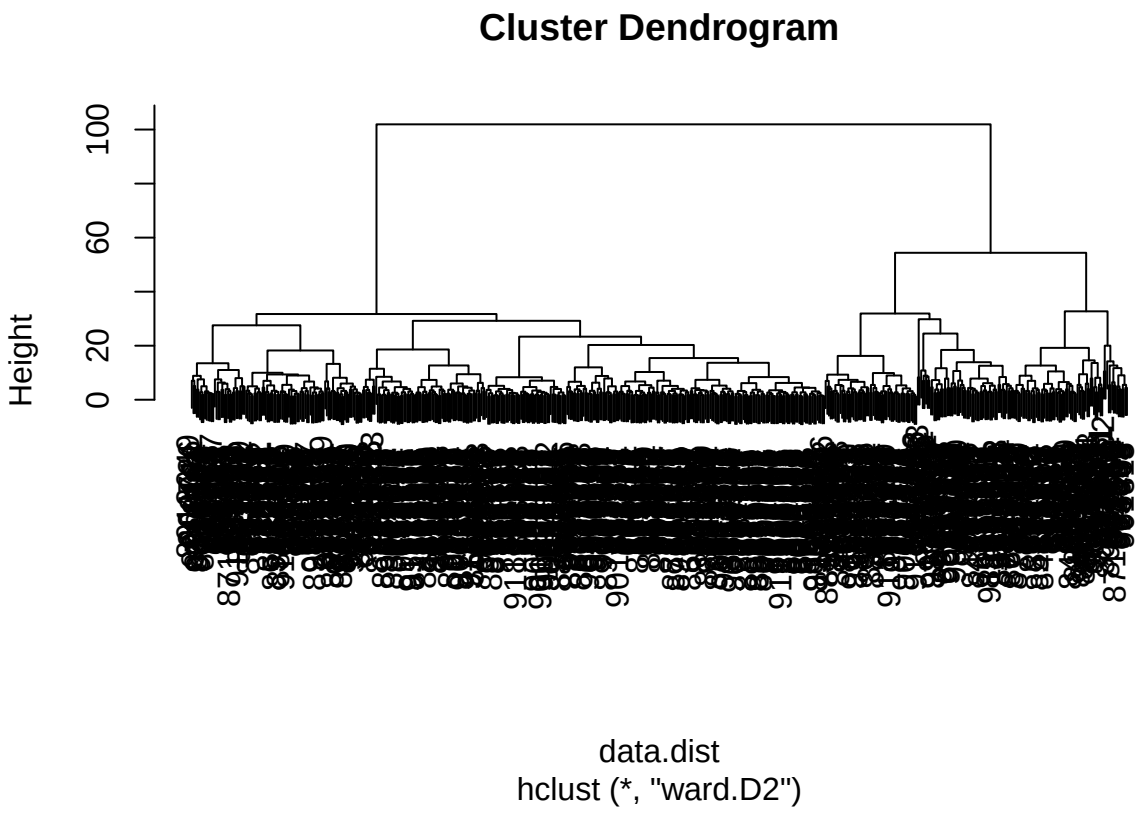

```
# Using the "complete" method
wisc.hclust.complete <- hclust(data.dist, method="complete")
plot(wisc.hclust.complete)
```




```
# Using the "average" method
wisc.hclust.average <- hclust(data.dist, method="average")
plot(wisc.hclust.average)
```




```
# Using the "ward.D2" method
wisc.hclust.ward.D2 <- hclust(data.dist, method="ward.D2")
plot(wisc.hclust.ward.D2)
```



The "ward.D2" method is my favorite. Although each cluster dendrogram appears to be messy and difficult to interpret, the dendrogram that uses the "ward.D2" method is the neatest and is a tiny bit easier to understand.

5. Combining Methods

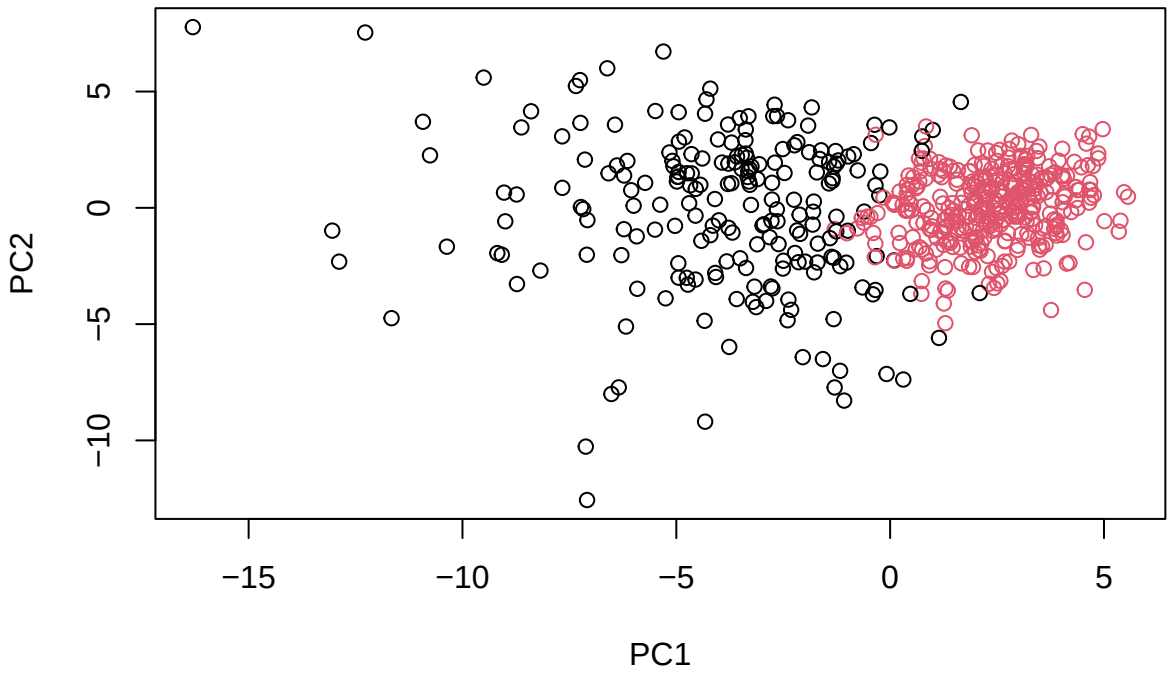
```
wisc.pr.hclust <- hclust(dist((wisc.pr$x[, 1:7])), method="ward.D2")
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
## grps
##   1   2
## 216 353
```

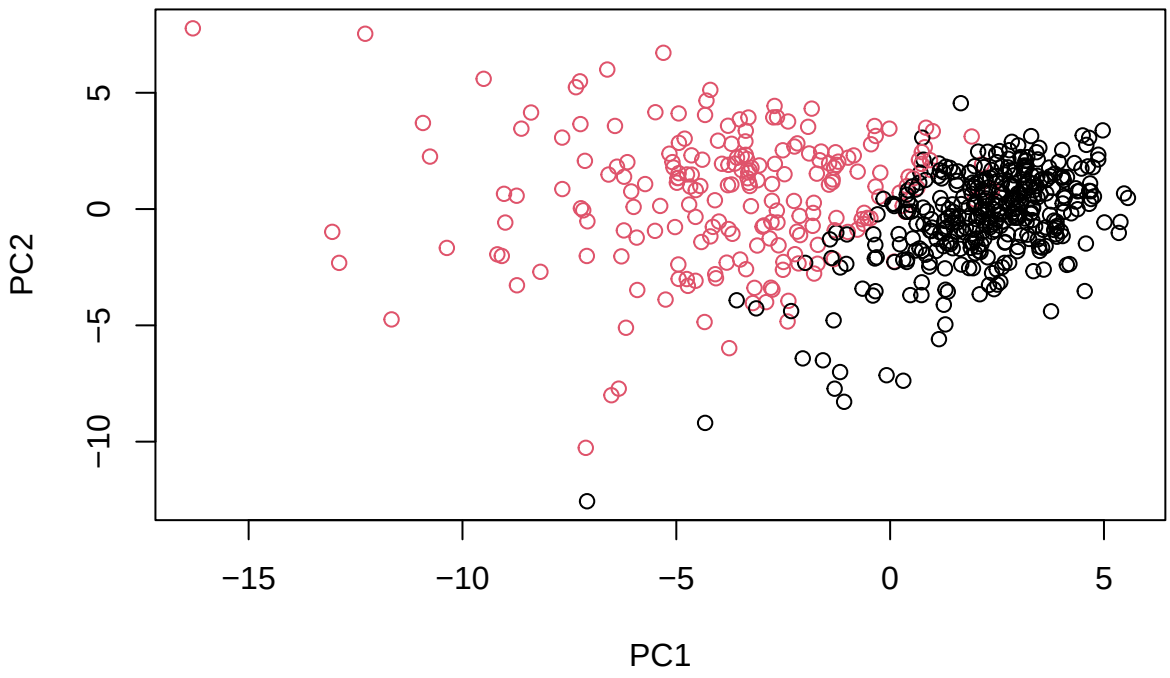
```
table(grps, diagnosis)
```

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

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




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



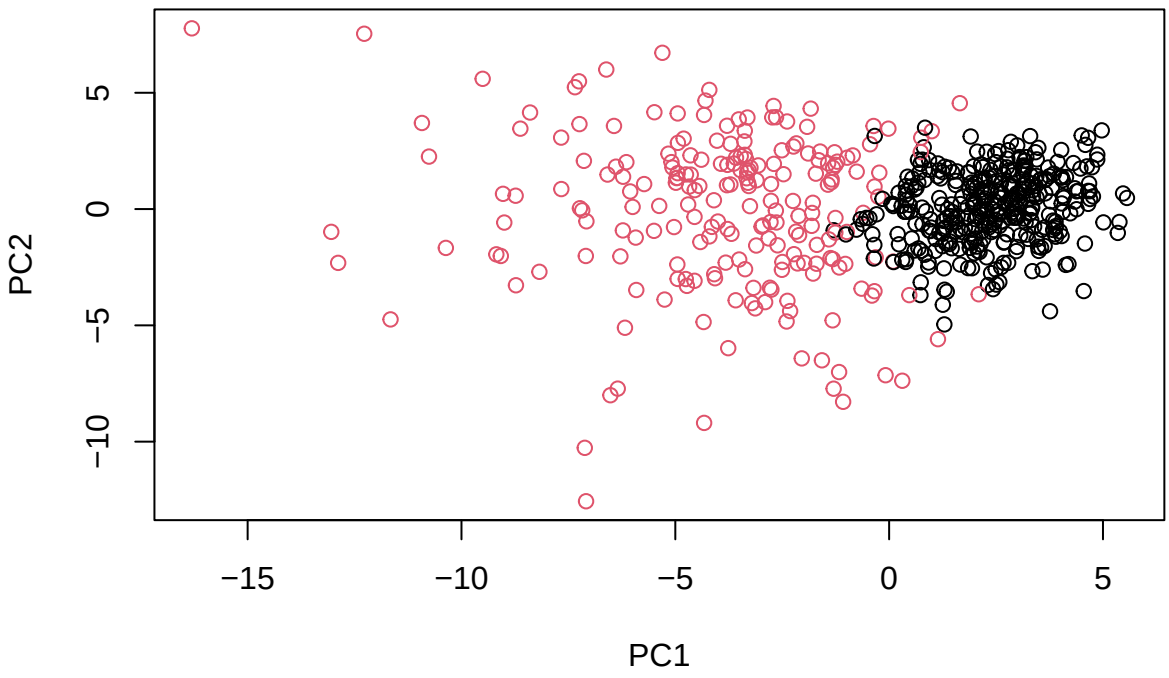

```
g <- as.factor(grps)
levels(g)
```

```
## [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")

## Cut the hierarchical cluster model into 2 clusters
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
```

The model with 4 clusters separates the diagnoses really well because it's clear that each cluster contains a majority of only one kind of diagnosis. Cluster 1 primarily contains malignant diagnoses while cluster 2 contains mainly benign diagnoses.

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.

Question 16 can be skipped since section 4 (K-means clustering) was optional.

6. Sensitivity/Specificity

Sensitivity

```
# from wisc.hclust into 4 clusters
165/(165+40)
```

```
## [1] 0.804878
```

```
# from Q15
188/(188+24)
```

```
## [1] 0.8867925
```

Specificity

```
# from wisc.hclust into 4 clusters
343/(343+12)
```

```
## [1] 0.9661972
```

```
# from Q15
329/(329+28)
```

```
## [1] 0.9215686
```

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

Cutting wisc.pr.hclust into 2 clusters gave the best sensitivity, which is 0.88679245. Cutting wisc.hclust into 4 clusters gave the best specificity, which is 0.96619718.

7. Prediction

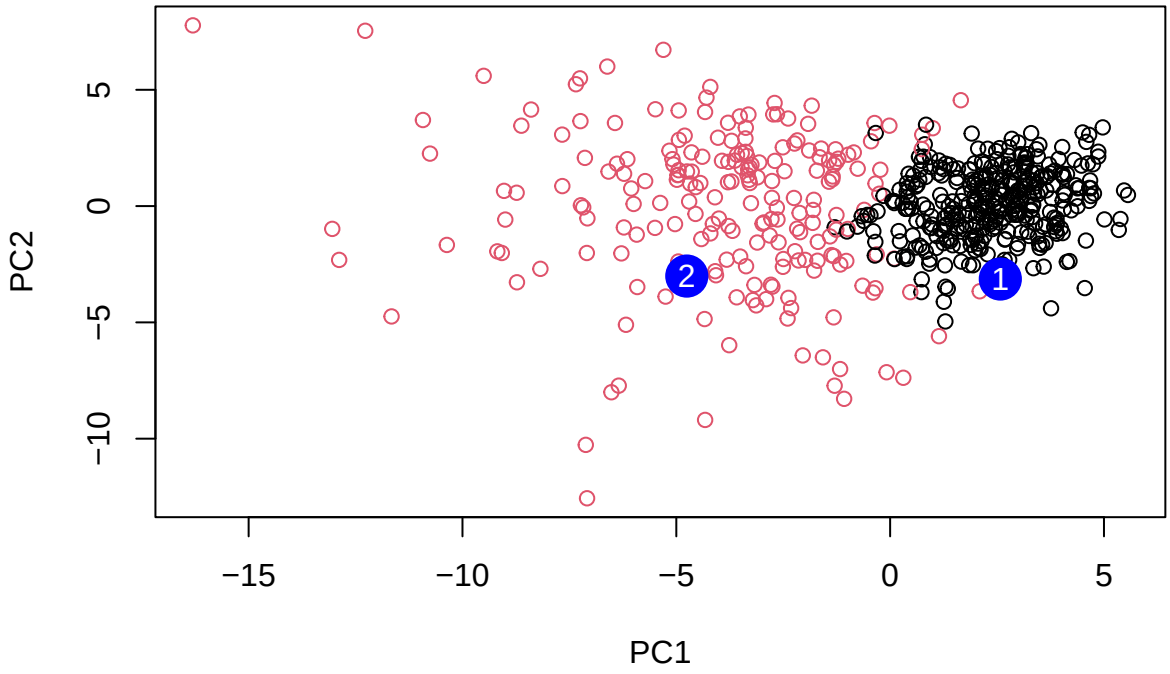
Here we read some new data and use our PCA model to examine whether they most closely resemble M or B patients from our original dataset.

```
#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,]  2.576616 -3.135913  1.3990492 -0.7631950  2.781648 -0.8150185 -0.3959098
## [2,] -4.754928 -3.009033 -0.1660946 -0.6052952 -1.140698 -1.2189945  0.8193031
##            PC8       PC9       PC10      PC11      PC12      PC13     PC14
## [1,] -0.2307350 0.1029569 -0.9272861 0.3411457  0.375921 0.1610764 1.187882
## [2,] -0.3307423 0.5281896 -0.4855301 0.7173233 -1.185917 0.5893856 0.303029
```

```
##           PC15       PC16        PC17        PC18        PC19       PC20
## [1,] 0.3216974 -0.1743616 -0.07875393 -0.11207028 -0.08802955 -0.2495216
## [2,] 0.1299153  0.1448061 -0.40509706  0.06565549  0.25591230 -0.4289500
##            PC21       PC22       PC23       PC24        PC25         PC26
## [1,]  0.1228233 0.09358453 0.08347651  0.1223396  0.02124121  0.078884581
## [2,] -0.1224776 0.01732146 0.06316631 -0.2338618 -0.20755948 -0.009833238
##              PC27        PC28         PC29         PC30
## [1,]  0.220199544 -0.02946023 -0.015620933  0.005269029
## [2,] -0.001134152  0.09638361  0.002795349 -0.019015820
```

```
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")
```



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

Patient 2