Class 08

Darby Patterson

Unsupervised Learning Analysis of Human Breast Cancer Cells

We first need to download our data from the WisconsinCancer.csv datafile.

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


# 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]
diagnosis <- wisc.df$diagnosis
diagnosis <- as.factor(diagnosis)

diagnosis <- factor(diagnosis, levels = c("B", "M"))
#diagnosis
#diagnosis <- as.factor(diagnosis)
#unique(diagnosis)
```

- **Q1**. How many observations are in this dataset?

```
dim(wisc.data)
```

```
[1] 569  30
```

- There are 569 observations in the dataset.
- **Q2**. How many of the observations have a malignant diagnosis?

```
table(wisc.df$diagnosis == "M")
```

```
FALSE  TRUE
  357   212
```

```
#sum(wisc.data$diagnosis)

table(wisc.df$diagnosis)
```

```
  B   M
357 212
```

- We have 212 observations with the diagnosis malignant
- **Q3**. How many variables/features in the data are suffixed with `_mean`?

```
grepl("_mean$", colnames(wisc.data))
```

```
 [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE
[13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[25] FALSE FALSE FALSE FALSE FALSE FALSE
```

```
sum(grepl("_mean$", names(wisc.data)))
```

```
[1] 10
```

There are 10 feature in the data suffixed with `_mean`.

```
# 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
```

Performing PCA

```
# Perform PCA on wisc.data by completing the following code
wisc.pr <- prcomp(wisc.data,scale=TRUE)
#summary(wisc.pr)
```

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

```
prop_var_PC1 <- (wisc.pr$sdev[1]^2) / sum(wisc.pr$sdev^2)
prop_var_PC1
```

```
[1] 0.4427203
```

- Around 44.27% of the original variance is captured by the first principal component.
- **Q5**. How many principal components (PCs) are required to describe at least 70% of the original variance in the data?

```
prop_var_cumsum <- cumsum(wisc.pr$sdev^2) / sum(wisc.pr$sdev^2)
num_PCs_70 <- which(prop_var_cumsum > 0.7)[1]
num_PCs_70
```

```
[1] 3
```

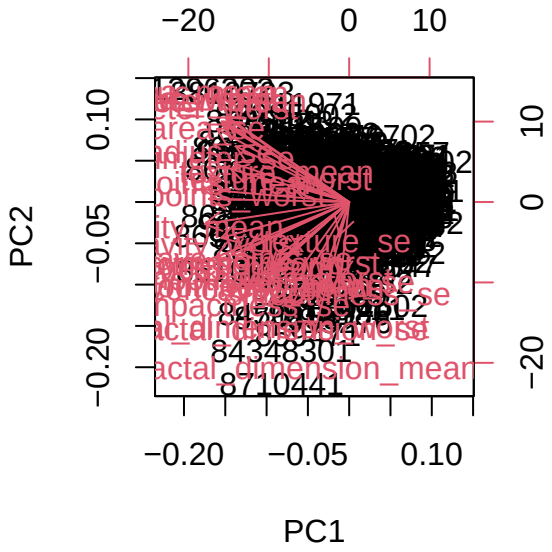
- 3 principal components (PCs) are required to describe at least 70% of the original variance in the data.
- **Q6**. How many principal components (PCs) are required to describe at least 90% of the original variance in the data?

```
prop_var_cumsum <- cumsum(wisc.pr$sdev^2) / sum(wisc.pr$sdev^2)
num_PCs_90 <- which(prop_var_cumsum > 0.9)[1]
num_PCs_90
```

```
[1] 7
```

- 7 principal components (PCs) are required to describe at least 90% of the original variance in the data.

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

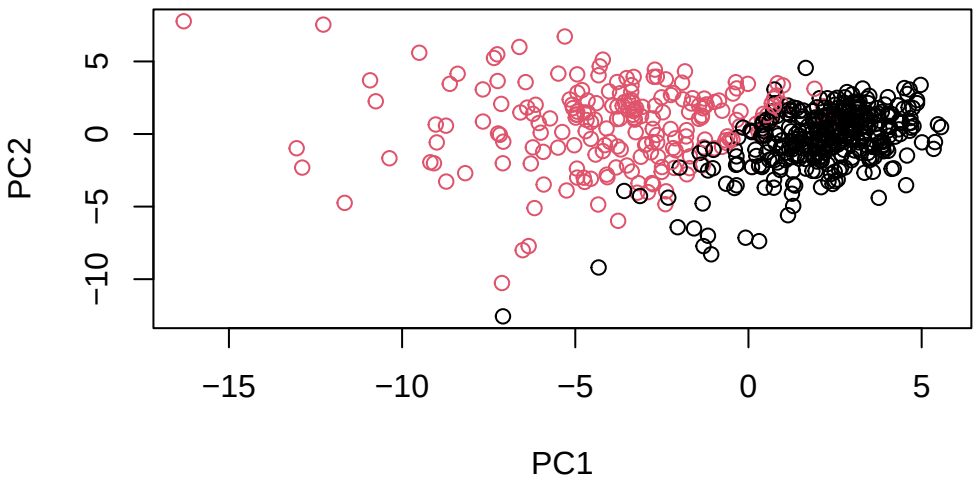


Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

This plot is very hard to understand, we want to analyze it further to get data that we can actually use. If we plot different PC's as x and y on a scatterplot, we can label our malignant and benign patients.

Scatterplot Analysis

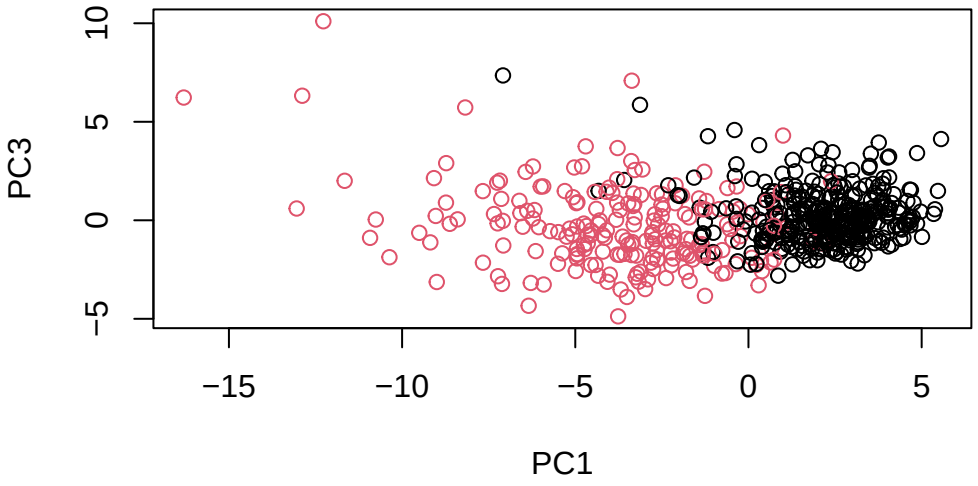
```
# Scatter plot observations by components 1 and 2
plot(wisc.pr$x[,1], wisc.pr$x[,2], col = as.numeric(diagnosis),
     xlab = "PC1", ylab = "PC2")
```



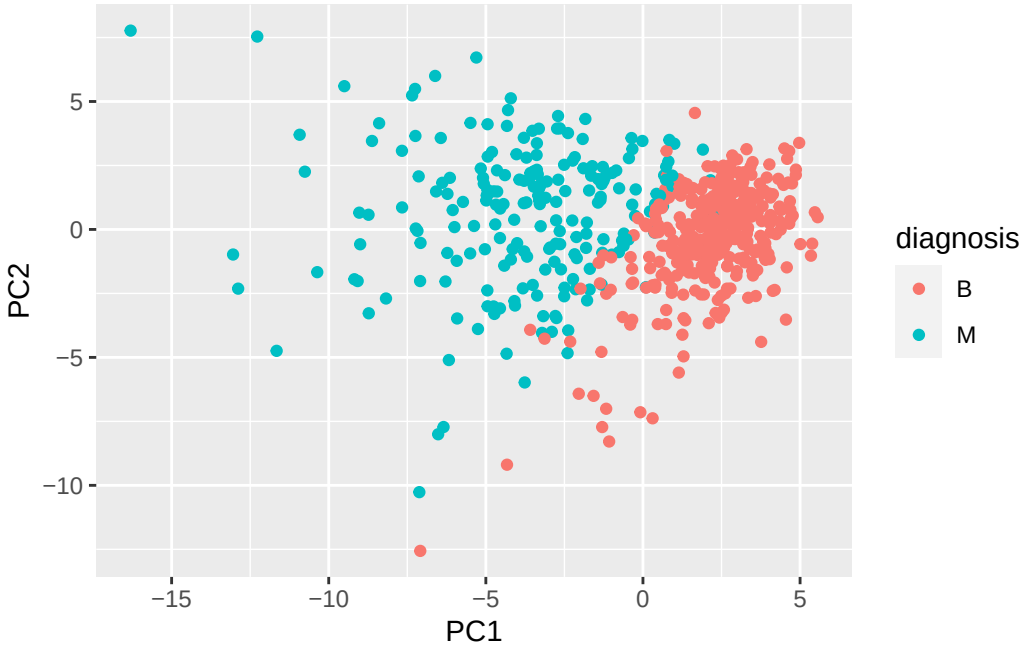
Q8. Generate a similar plot for principal components 1 and 3. What do you notice about these plots?

Principal component 2 explains more variance than principal component 3, so our first plot has cleaner separation than the plot below. Principal component 1 explains our data best.

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



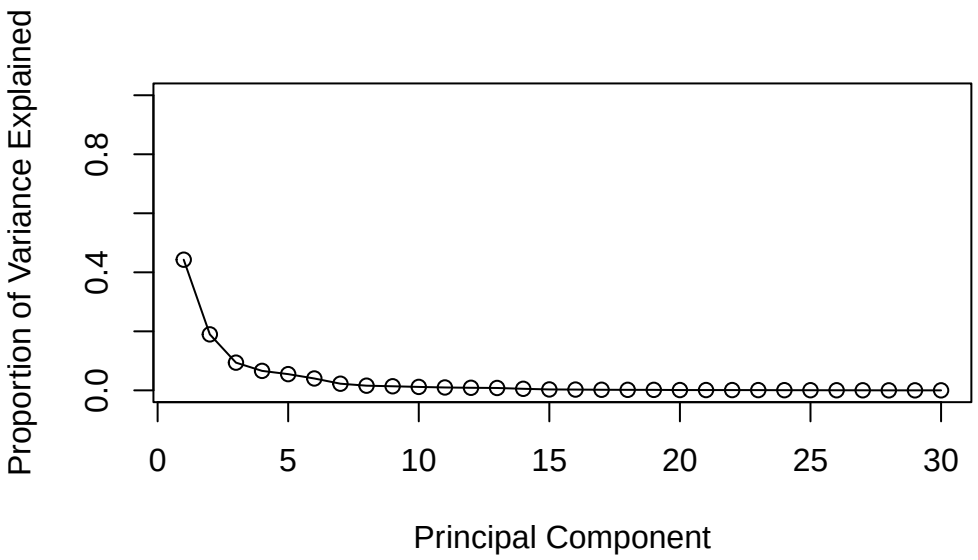

```
# Create a data.frame for ggplot
df <- as.data.frame(wisc.pr$x)
df$diagnosis <- diagnosis

# Load the ggplot2 package
library(ggplot2)

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




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

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

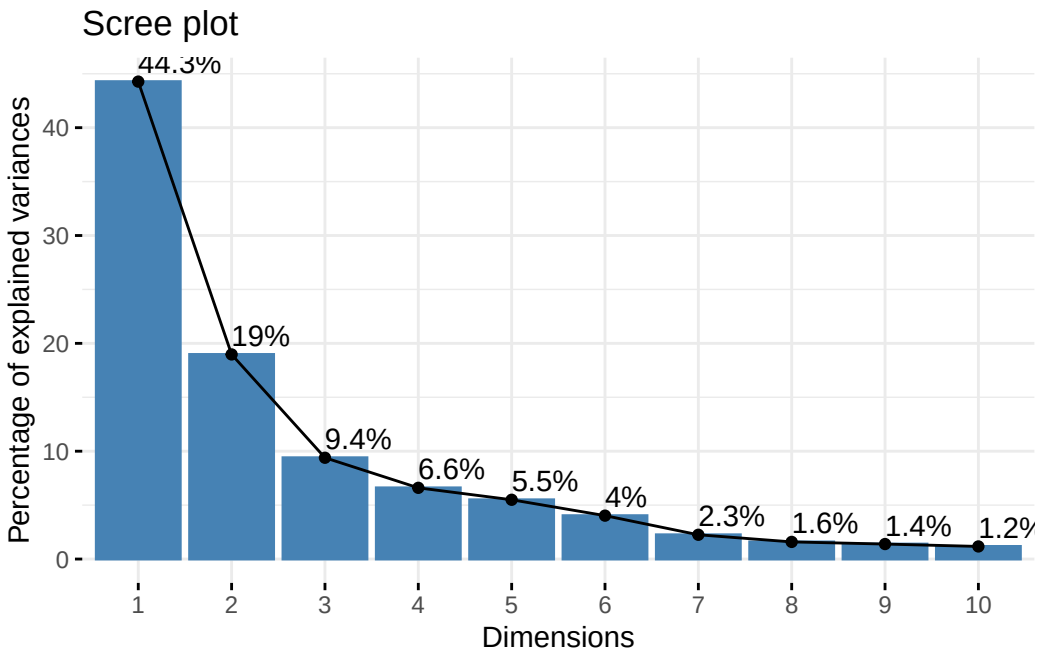
```
# Calculate the variance explained by each principal component
pve <- pr.var / sum(pr.var)

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




```
## 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`?

This tells us how much this original feature contributes to the first PC.

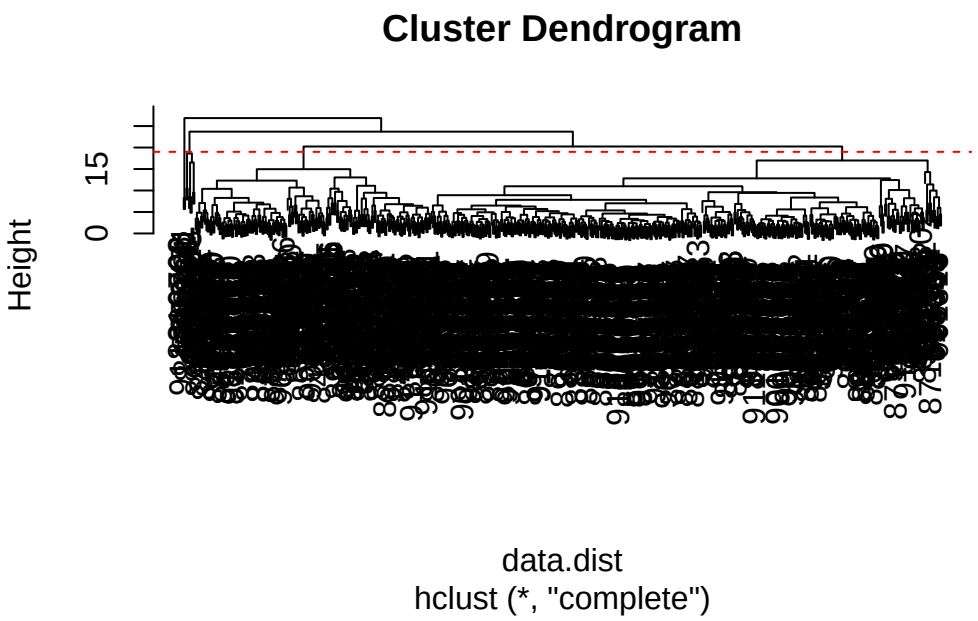
Hierarchical Clustering

```
wisc.pr$rotation["concave.points_mean", 1]
```

```
[1] -0.2608538
```

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

# Calculate pairwise distances between observations using the "dist()" function
data.dist <- dist(data.scaled)
# Create hierarchical clustering model with complete linkage
wisc.hclust <- hclust(data.dist, method = "complete")
```

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



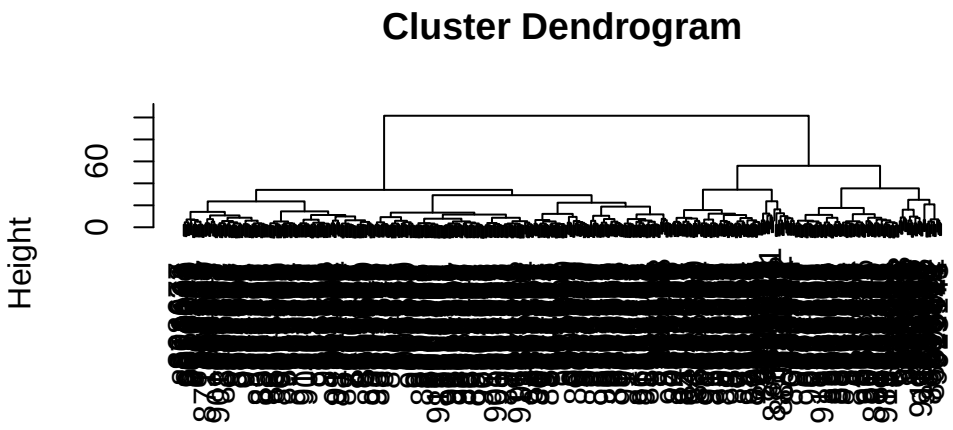
- **Q10.** Using the `plot()` and `abline()` functions, what is the height at which the clustering model has 4 clusters?

  At a height of 19 the clustering model has 4 clusters.

```
# Cut the tree into 4 clusters
wisc.hclust.clusters <- cutree(wisc.hclust, k = 4)

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

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

```
# Create hierarchical clustering model with "ward.D2" linkage method
pve <- pr.var / sum(pr.var)
cumulative_pve <- cumsum(pve)
num_components <- sum(cumulative_pve <= 0.9)


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

plot(wisc.pr.hclust)
```



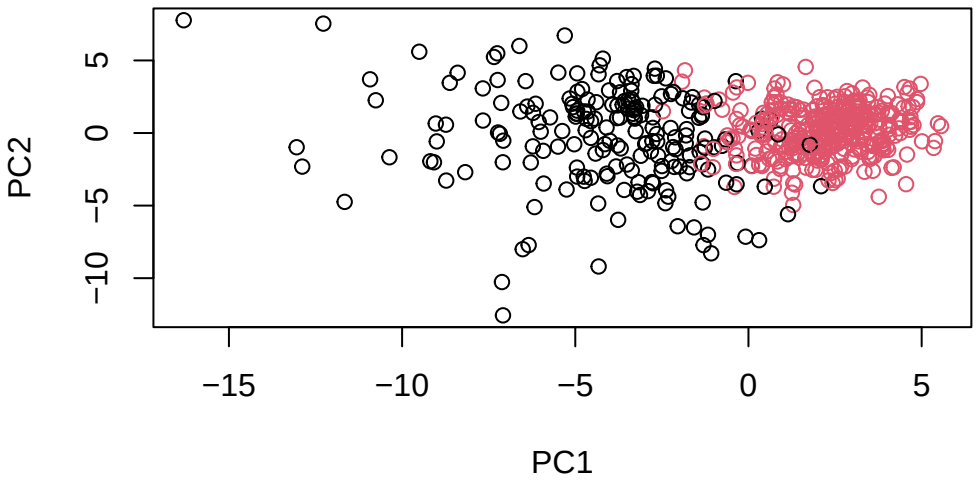

```
grps <- cutree(wisc.pr.hclust, k=2)
table(grps, diagnosis)
```

```
    diagnosis
grps   B   M
   1  25 177
   2 332  35
```

– **Q12.** Which method gives your favorite results for the same `data.dist` dataset? Explain your reasoning.

For our data, the complete method works just fine and we don't need to keep trimming the tree down. The other methods are useful for data that isn't as clear.

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



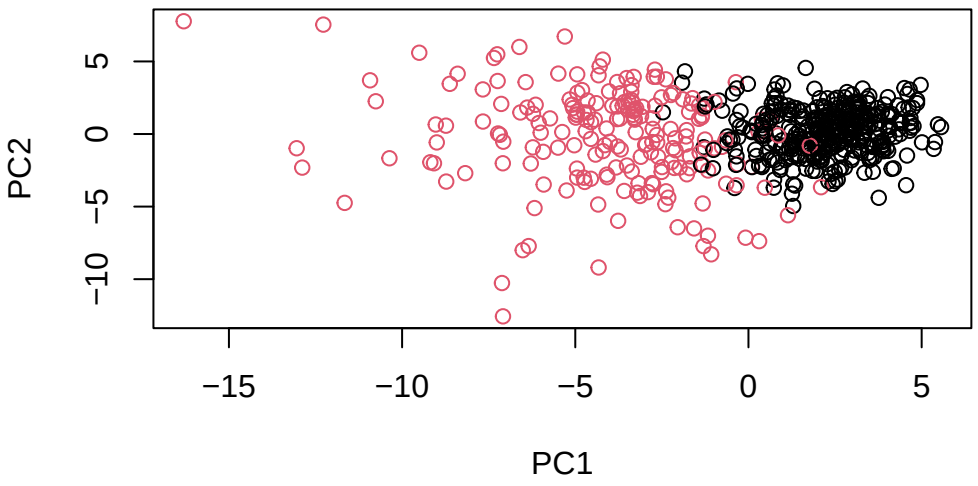

```
#plot(wisc.pr$x[,1:2], col=diagnosis)

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)
```




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

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


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

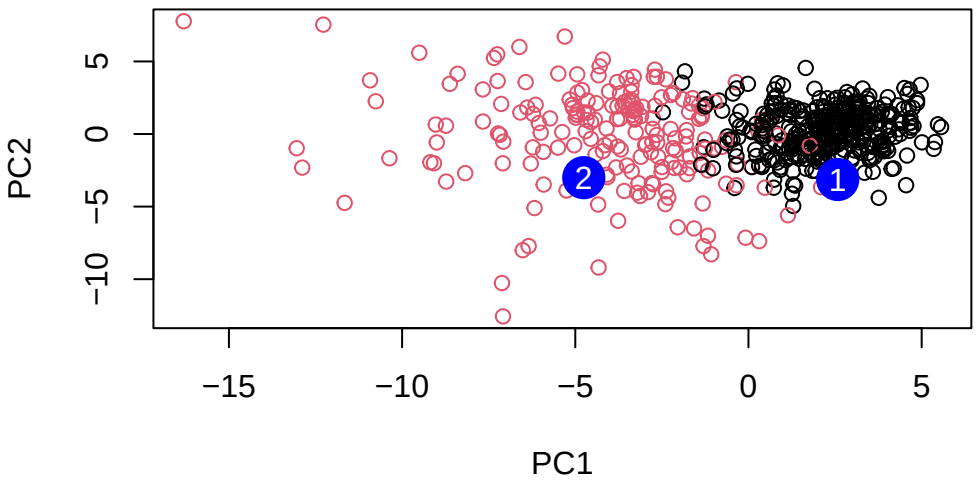
```
                     diagnosis
wisc.pr.hclust.clusters   B   M
                      1  28 188
                      2 329  24
```

  - **Q13**. How well does the newly created model with four clusters separate out the two diagnoses?

It separates them more efficiently, we can see a clear distinction in our malignant and benign groups.

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

```
                    diagnosis
wisc.hclust.clusters   B   M
```
```

```
1  12 165
2   2   5
3 343  40
4   0   2
```

– **Q14**. How well do the 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.

This model does a pretty good job separating the groups but isn't as precise as when we performed a PCA.

Prediction

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

- ```
  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")
  ```
  ```



- **Q16.** Which of these new patients should we prioritize for follow up based on your results?

Patient 2, because they fall in the primarily malignant section of the data.