Class 8: Breast Cancer Analysis

Wade Ingersoll (PID: 69038080)

Table of contents

Background

The goal of today's mini-project is to explore a complete analysis using the unsupervised learning techniques covered in the last class. We will extend what we learned by combining PCA as a preprocessing step to clustering using data that consist of measurements of cell nuclei of human breast massess.

The data itself comes from the Wisconsin Breast Cancer Diagnostic Data Set first reported by K. P. Benne and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets".

Values in this data set describe characteristics of the cell nuclei present in digitized images of a fine needle aspiration (FNA) of a breast mass.

Data import

The data is available as a CSV from class website:

```
wisc.df <- read.csv("WisconsinCancer (2).csv", row.names=1)
```

Make sure we do not include sample id or diagnosis columns in the data that we analyze below.

```
diagnosis <- as.factor(wisc.df$diagnosis)
wisc.data <- wisc.df[, -1]
dim(wisc.data)
```

```
[1] 569  30
```

Exploratory data analysis

Q1. How many observations are in this dataset?

There are 569 observations/samples/patients in the data set. I used nrow(wisc.data) to find that number.

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

There are 212; see code below (two ways to determine)

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

```
[1] 212
```

```
table(wisc.df$diagnosis)
```

```
  B   M
357 212
```

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

There are 10; see code below

```
#colnames(wisc.data)
length( grep("_mean", colnames(wisc.data)) )
```

```
[1] 10
```

Easier to read:

```
n <- colnames(wisc.data)
inds <- grep("_mean", n)
length(inds)
```

```
[1] 10
```

Principal Component Analysis

The main function in Base R for PCA is called `prcomp()`. An optional argument `scale` should nearly always be switched to `scale=TRUE` for this function.

```
wisc.pr <- prcomp(wisc.data, scale=TRUE)
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)?

The answer is 0.4427203, see code below

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

```
 [1] 4.427203e-01 1.897118e-01 9.393163e-02 6.602135e-02 5.495768e-02
 [6] 4.024522e-02 2.250734e-02 1.588724e-02 1.389649e-02 1.168978e-02
[11] 9.797190e-03 8.705379e-03 8.045250e-03 5.233657e-03 3.137832e-03
[16] 2.662093e-03 1.979968e-03 1.753959e-03 1.649253e-03 1.038647e-03
[21] 9.990965e-04 9.146468e-04 8.113613e-04 6.018336e-04 5.160424e-04
[26] 2.725880e-04 2.300155e-04 5.297793e-05 2.496010e-05 4.434827e-06
```

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

A: Three PCs are required (PC1-3). See summary below.

```
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
```

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

A: Seven PCs are required (PC1-7). See summary below.

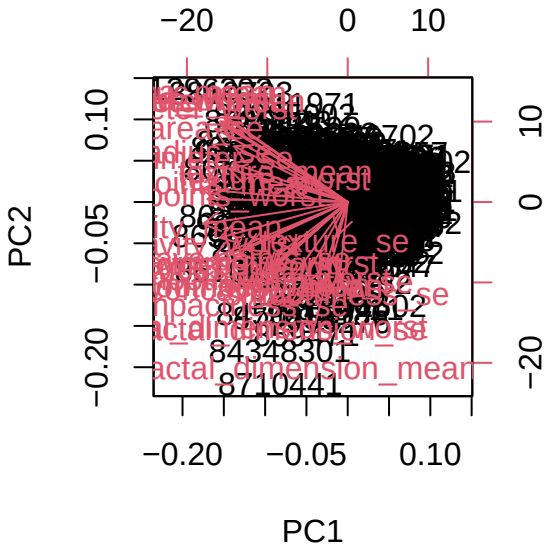
```
summary(wisc.pr)
```

```
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
```

Interpreting PCA results

> Create a biplot of the wisc.pr using the `biplot()` function.

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

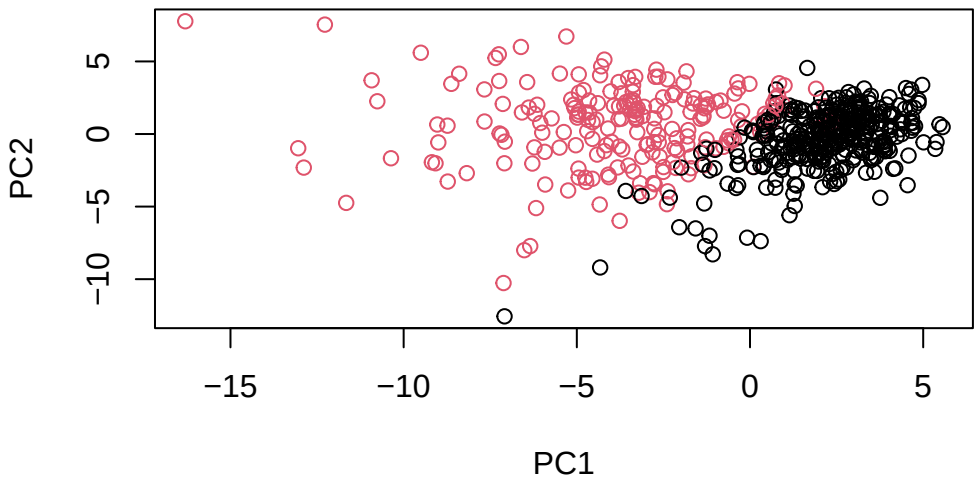


Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

A: What stands out is that the plot is very cluttered with lots of overlap between labels, making it very difficult to interpret the PCA result.

Scatter plot observations by components 1 and 2

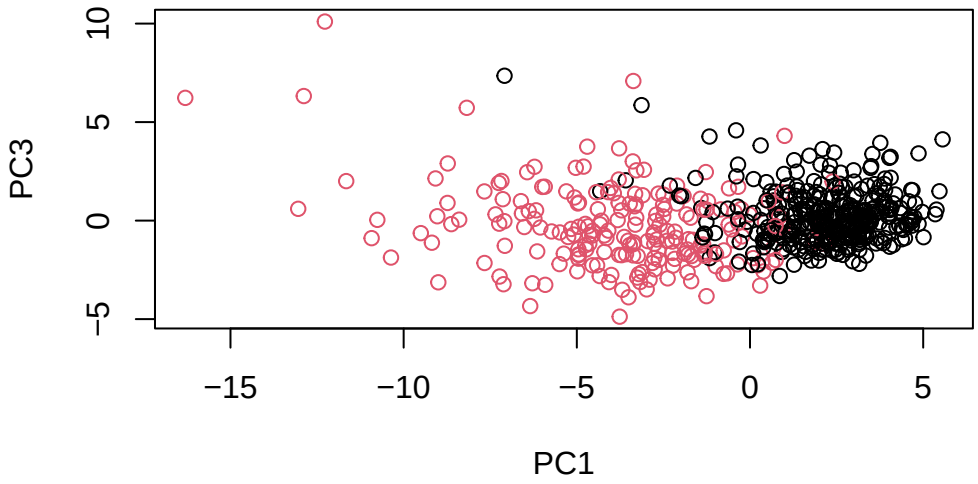
```
plot( wisc.pr$x[,1], wisc.pr$x[,2] , col = diagnosis ,
     xlab = "PC1", ylab = "PC2")
```


PC2
5
0
−5
−10
−15
−10
−5
0
5
PC1

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

A: I notice the first plot (PC1 and 2) has less overlap between clusters (i.e. it has a cleaner cut). See plot for PC1 and 3 below.

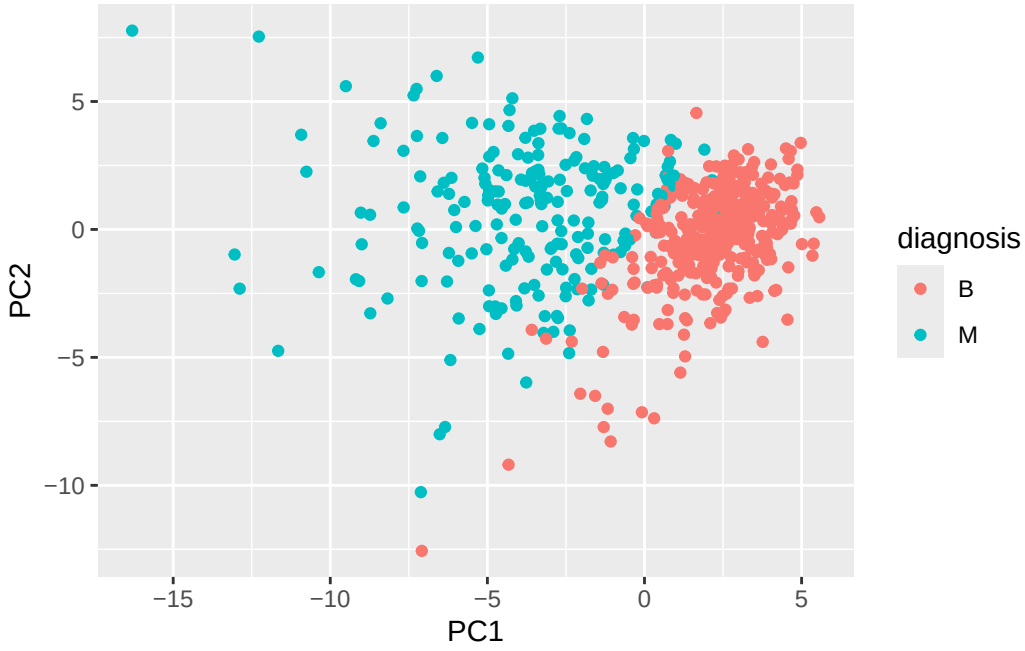
```
# Repeat for components 1 and 3
plot( wisc.pr$x[,1], wisc.pr$x[,3] , col = diagnosis ,
     xlab = "PC1", ylab = "PC3")
```



Let's make our main result figure - the "PC plot" or "score plot", "ordienation plot"...

```
library(ggplot2)

ggplot(wisc.pr$x) +
  aes(PC1, PC2, col=diagnosis) +
  geom_point()
```



Variance Explained

Calculate variance of each component

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

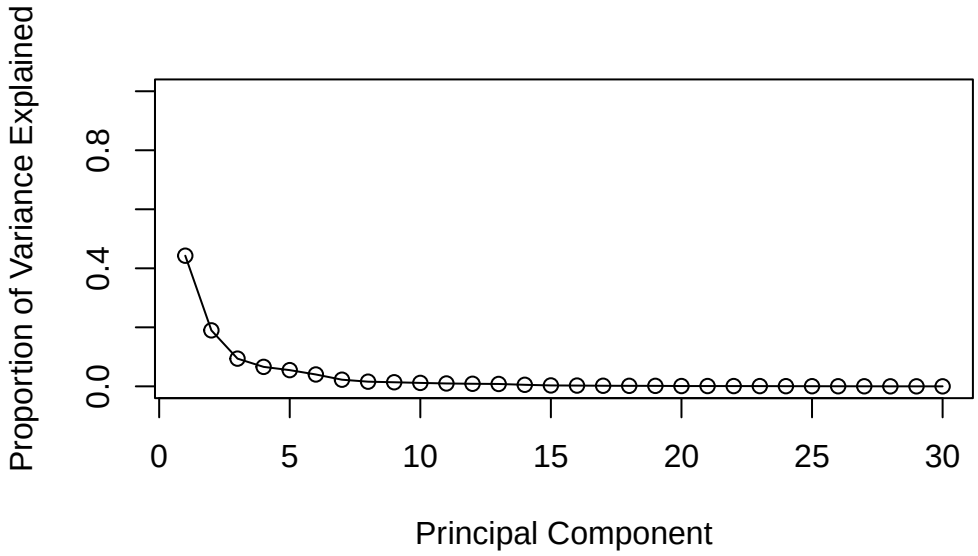
```
[1] 13.281608  5.691355  2.817949  1.980640  1.648731  1.207357
```

Variance explained by each principal component: pve

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

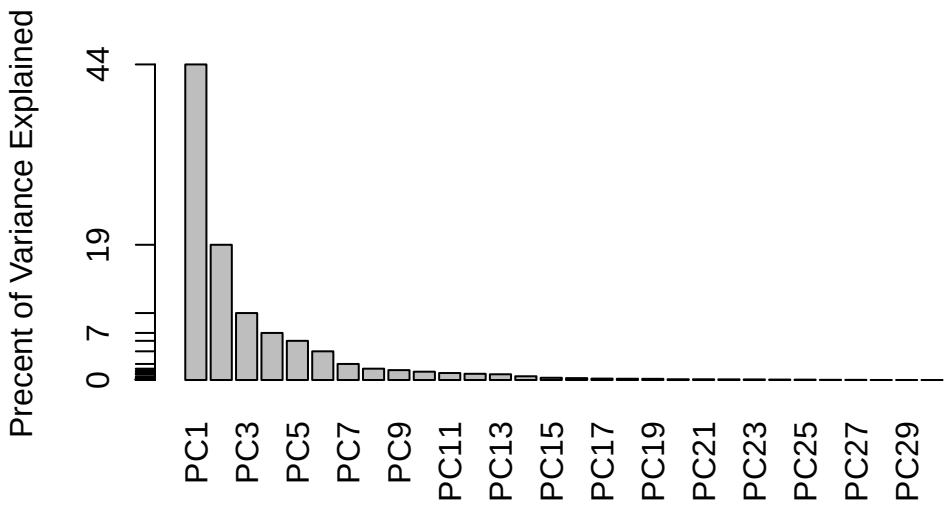
Plot variance explained for each principal component

```
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 )
```



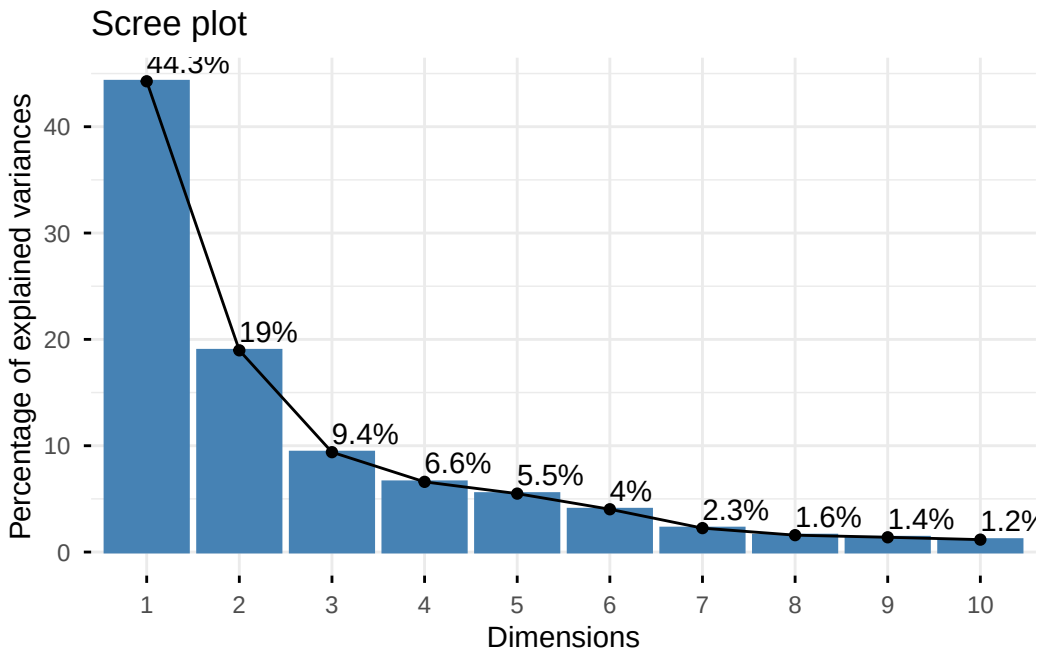
ggplot based graph

```
library(factoextra)
```

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

```
fviz_eig(wisc.pr, addlabels = TRUE)
```

```
Warning in geom_bar(stat = "identity", fill = barfill, color = barcolor, :
Ignoring empty aesthetic: `width`.
```



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

A: Using the code below, the loading vector is -0.26085376

```
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
```

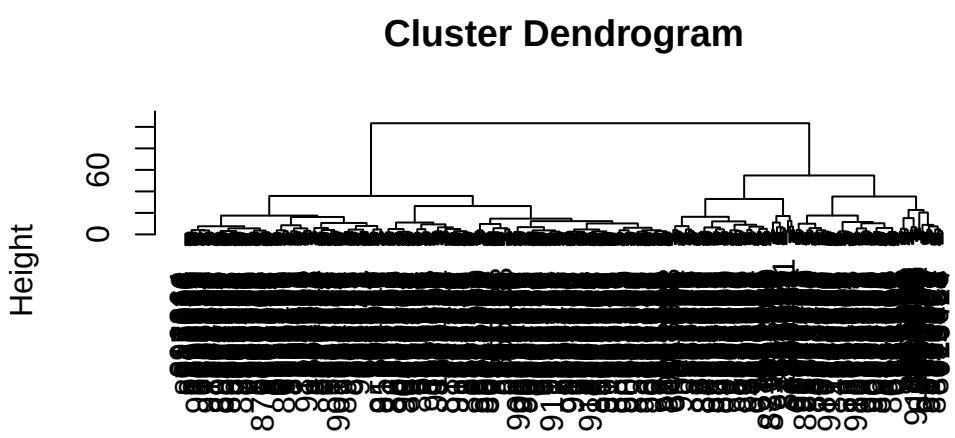
Hierarchical Clustering

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

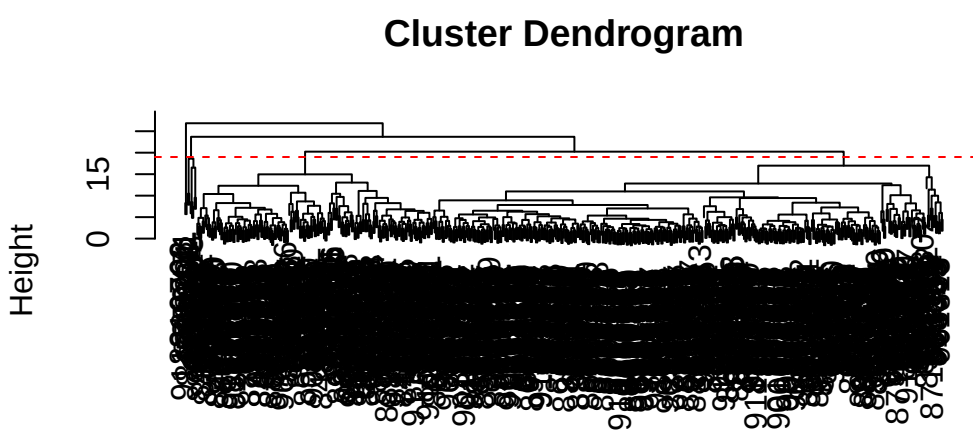
```
            radius_mean            texture_mean          perimeter_mean
                      1                       1                       1
              area_mean         smoothness_mean        compactness_mean
                      1                       1                       1
         concavity_mean     concave.points_mean           symmetry_mean
                      1                       1                       1
 fractal_dimension_mean              radius_se             texture_se
                      1                       1                       1
          perimeter_se                area_se           smoothness_se
                      1                       1                       1
         compactness_se           concavity_se       concave.points_se
                      1                       1                       1
           symmetry_se    fractal_dimension_se            radius_worst
                      1                       1                       1
         texture_worst         perimeter_worst              area_worst
                      1                       1                       1
     smoothness_worst       compactness_worst         concavity_worst
                      1                       1                       1
concave.points_worst          symmetry_worst fractal_dimension_worst
                      1                       1                       1
```

Combining PCA and clustering

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




```
data.scaled <- scale(wisc.data)
data.dist <- dist(data.scaled)
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?

A: According to the plot above, **19** is the height at which the clustering model has 4 clusters.

Selecting Number of Clusters

Get my cluster membership vector

```
grps <- cutree(wisc.pr.hclust, h=70)
table(grps)
```

```
grps
  1   2
203 366
```

```
table(diagnosis)
```

```
diagnosis
  B   M
357 212
```

Senstivity

Make a "cross-table"

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

```
    diagnosis
grps   B   M
   1  24 179
   2 333  33
```

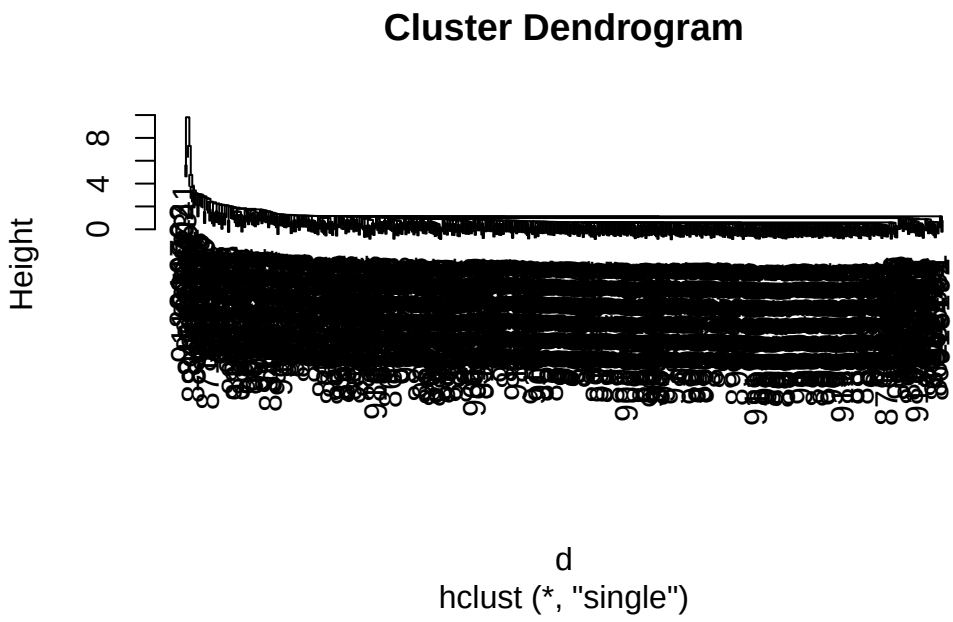
TP: 179 FP: 24

Sensitivity: TP/(TP+FN) = 0.8817734

Using Different Methods ("single")

> Q12. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning.
>
> A: My favorite method is "ward.D2" because it generates a plot that allows for use of abline() to find where the clustering model has 4 clusters.

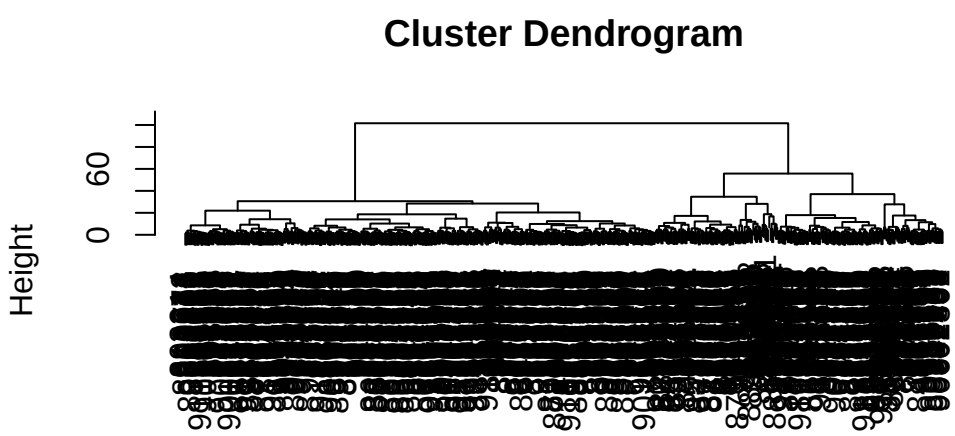
```
d <- dist( wisc.pr$x[,1:3] )
wisc.pr.hclust <- hclust(d, method="single")
plot(wisc.pr.hclust)
```



Combining Methods

Clustering on PCA results

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



Determine if these two clusters are malignant and benign

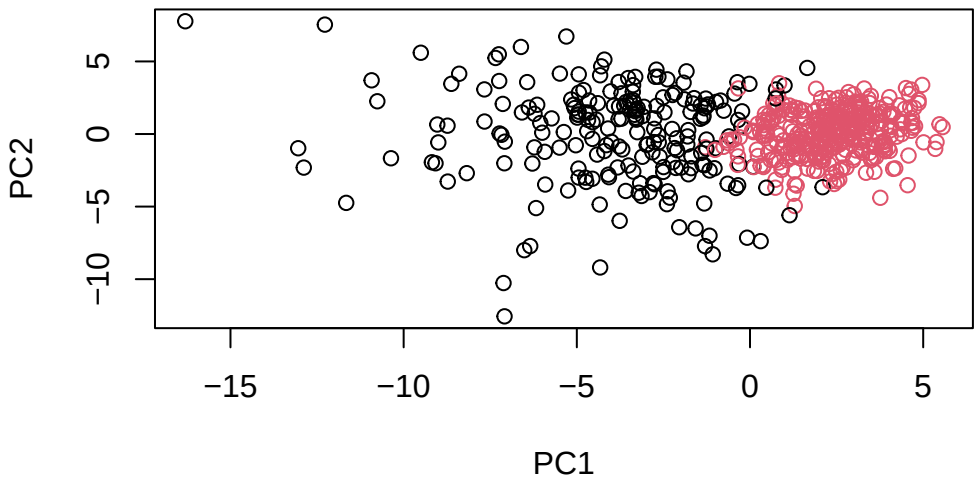
```
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
grps
  1   2
216 353
```

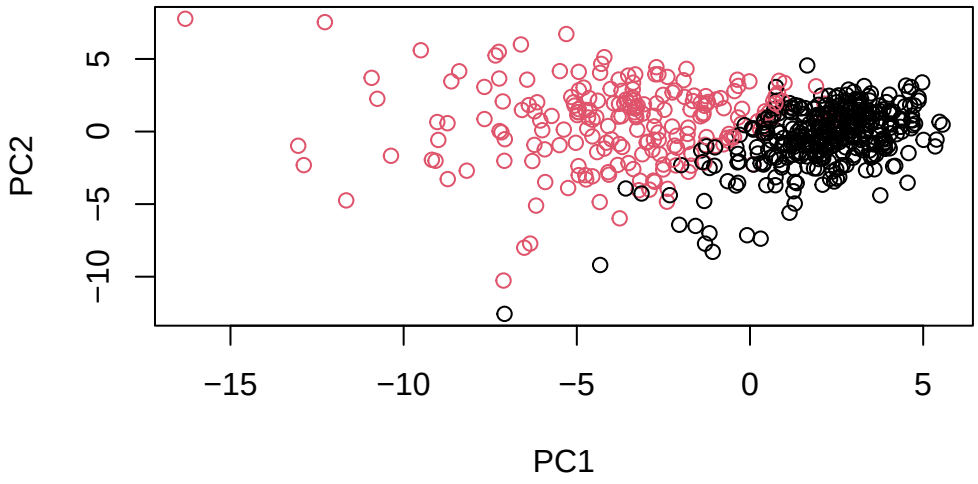
```
table(grps, diagnosis)
```

```
    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)
```



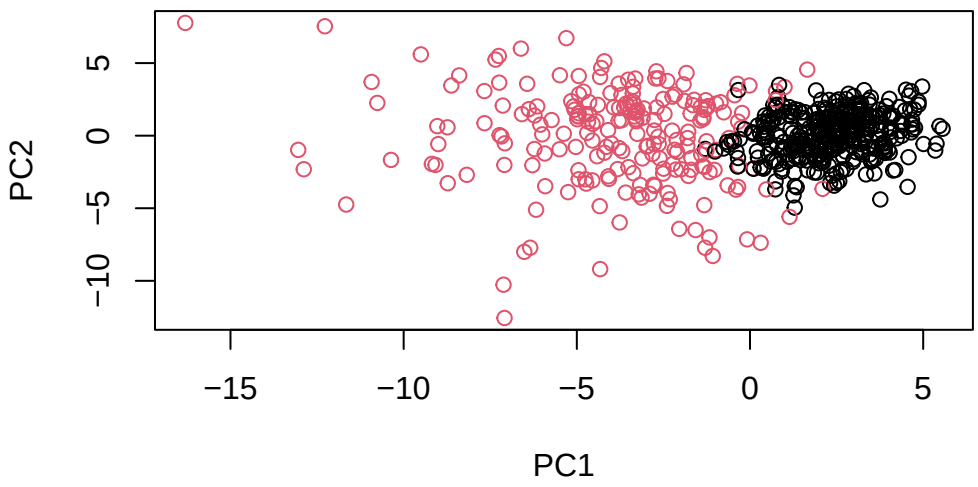
```
g <- as.factor(grps)
levels(g)
```

```
[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 this hierarchical clustering model into 2 clusters and assign the results to wisc.pr.hclust.clusters.

```
wisc.pr.hclust.clusters <- cutree(wisc.pr.hclust, k=4)
```

Using table(), compare the results from your new hierarchical clustering model with the actual diagnoses.

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

```
                    diagnosis
wisc.pr.hclust.clusters   B   M
                      1   0  45
                      2   2  77
                      3  26  66
                      4 329  24
```

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

A: It does so much better.

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.

A: They are much better.