mini-project

Joshua Mac

2025-02-03

Exploring Data Analysis

```
# Save your input data file into your Project directory

url<-"https://bioboot.github.io/bimm143_S20/class-material/WisconsinCancer.csv"
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)
#wisd.df dang that's a lot of data
```

Let's exclude the diagnosis column, which we will not be using.

```
wisc.data <- wisc.df[,-1]
```

We will have it separate for now.

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

Q1. How many observations are in this dataset?

```
# Rows will be observations and columns will be variables, so:
nrow(wisc.data)
```

```
[1] 569
```

A1. There are 59 observations in the dataset.

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

```
sum(diagnosis=="M")
```

```
[1] 212
```

A2. There are 212 observations with a malignant diagnosis.

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

```
length(grep("_mean$", names(wisc.data)))
```

```
[1] 10
```

A3. There are 10 variables that are suffixed with _mean.

PCA First,

```
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
```

```
wisc.pr <- prcomp(wisc.data)
summary(wisc.pr)
```

```
Importance of components:
                          PC1      PC2      PC3     PC4     PC5     PC6    PC7
Standard deviation     666.170 85.49912 26.52987 7.39248 6.31585 1.73337 1.347
Proportion of Variance   0.982  0.01618  0.00156 0.00012 0.00009 0.00001 0.000
Cumulative Proportion    0.982  0.99822  0.99978 0.99990 0.99999 0.99999 1.000
                           PC8    PC9   PC10   PC11    PC12    PC13    PC14
Standard deviation     0.6095 0.3944 0.2899 0.1778 0.08659 0.05623 0.04649
Proportion of Variance 0.0000 0.0000 0.0000 0.0000 0.00000 0.00000 0.00000
Cumulative Proportion  1.0000 1.0000 1.0000 1.0000 1.00000 1.00000 1.00000
                          PC15   PC16    PC17    PC18    PC19    PC20     PC21
Standard deviation     0.03642 0.0253 0.01936 0.01534 0.01359 0.01281 0.008838
Proportion of Variance 0.00000 0.0000 0.00000 0.00000 0.00000 0.00000 0.000000
Cumulative Proportion  1.00000 1.0000 1.00000 1.00000 1.00000 1.00000 1.000000
                          PC22     PC23     PC24     PC25     PC26     PC27
Standard deviation     0.00759 0.005909 0.005329 0.004018 0.003534 0.001918
```

```
Proportion of Variance 0.00000 0.000000 0.000000 0.000000 0.000000 0.000000
Cumulative Proportion  1.00000 1.000000 1.000000 1.000000 1.000000 1.000000
                           PC28     PC29      PC30
Standard deviation     0.001688 0.001416 0.0008379
Proportion of Variance 0.000000 0.000000 0.0000000
Cumulative Proportion  1.000000 1.000000 1.0000000
```

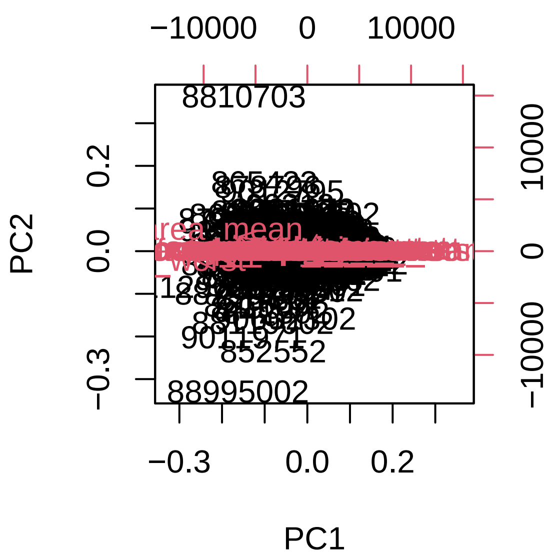
> Q4. From your results, what proportion of the original variance is captured by the first principal components (PC1)?
>
> A4. 0.982 or 98.2%! Essentially narrowing to one dimension from 30/31.
>
> Q5. How many principal components (PCs) are required to describe at least 70% of the original variance in the data?
>
> A5. Just one! PC1
>
> Q6. How many principal components (PCs) are required to describe at least 90% of the original variance in the data?
>
> A6. Again, just one! Still PC1 alone describes more than 90% of variance.

Biplot time!

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

```
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
```

```
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
Warning in arrows(0, 0, y[, 1L] * 0.8, y[, 2L] * 0.8, col = col[2L], length =
arrow.len): zero-length arrow is of indeterminate angle and so skipped
```

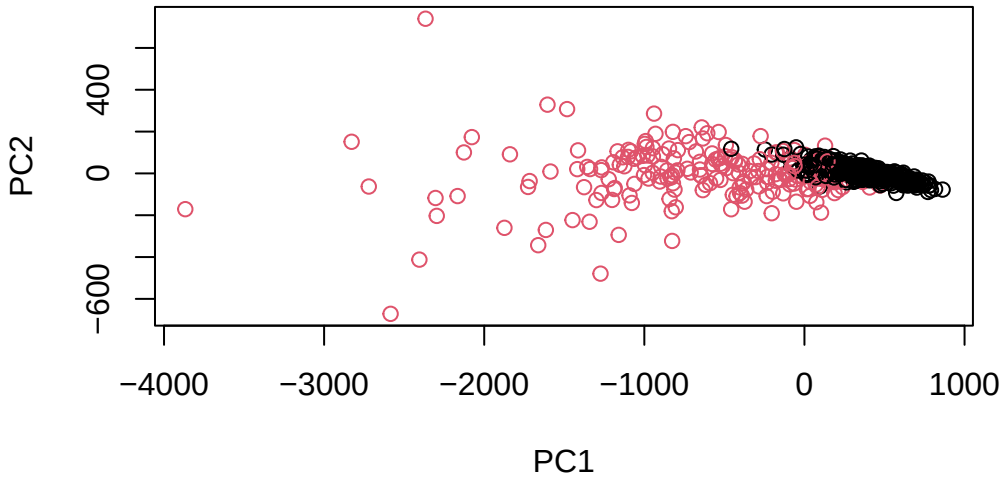


Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

A7. Pretty much everything, it's a mess plotting every single observation with its #. It's very difficult to understand like this, just yikes.

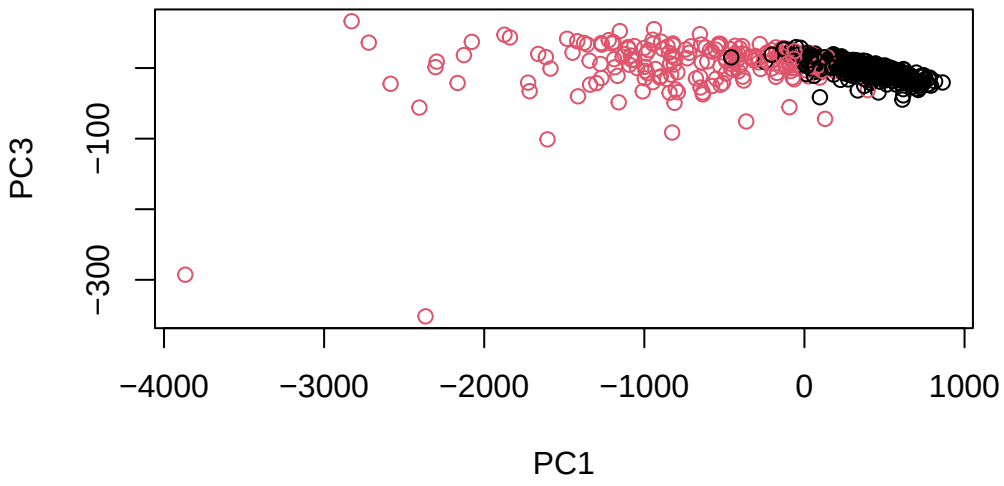
Let's use this:

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



Now PC1 vs PC3 >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 = as.factor(diagnosis),
     xlab = "PC1", ylab = "PC3")
```



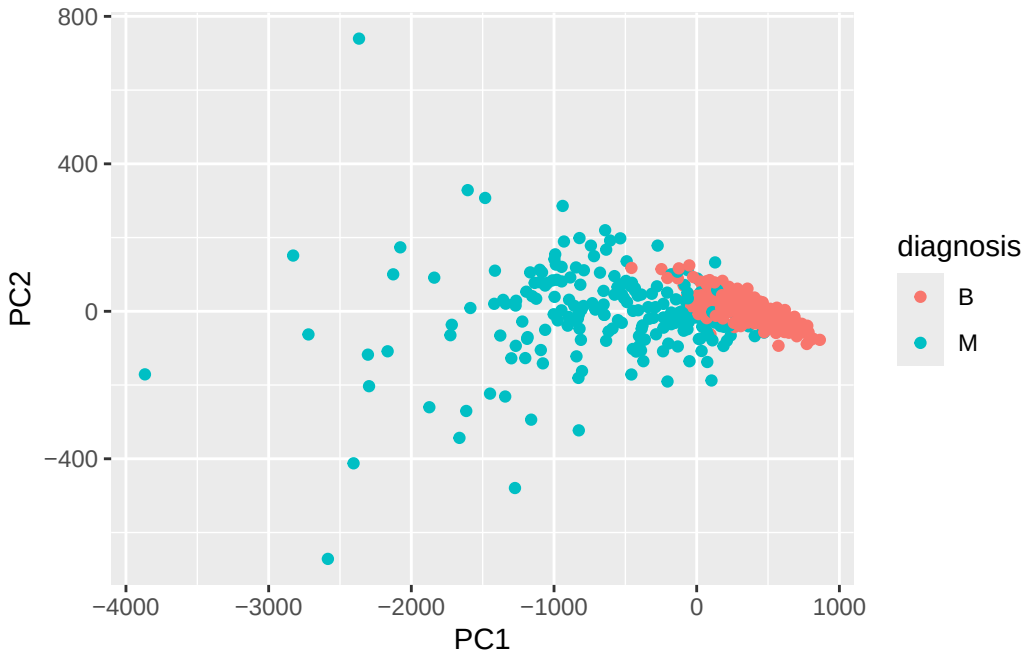
A8. The plot for PC1 vs PC2 is easier to visualize with a center, as PC2 accounts for more variance than PC3. The plots keep their shape and their general pattern though.

Let's move to ggplot, because this won't work alone.

```
# 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()
```



Variance Explained

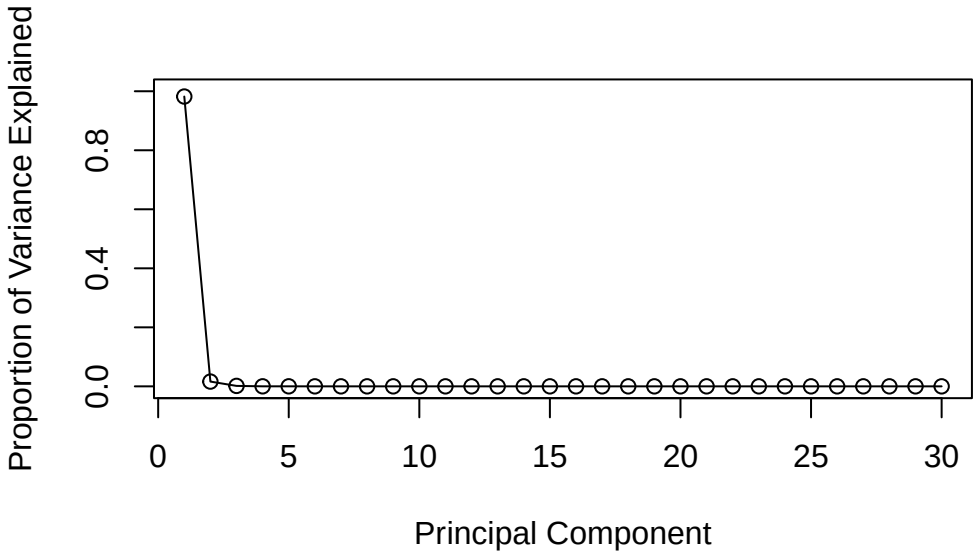
```
# Calculate variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

```
[1] 4.437826e+05 7.310100e+03 7.038337e+02 5.464874e+01 3.989002e+01
[6] 3.004588e+00
```

Now to loading variables!

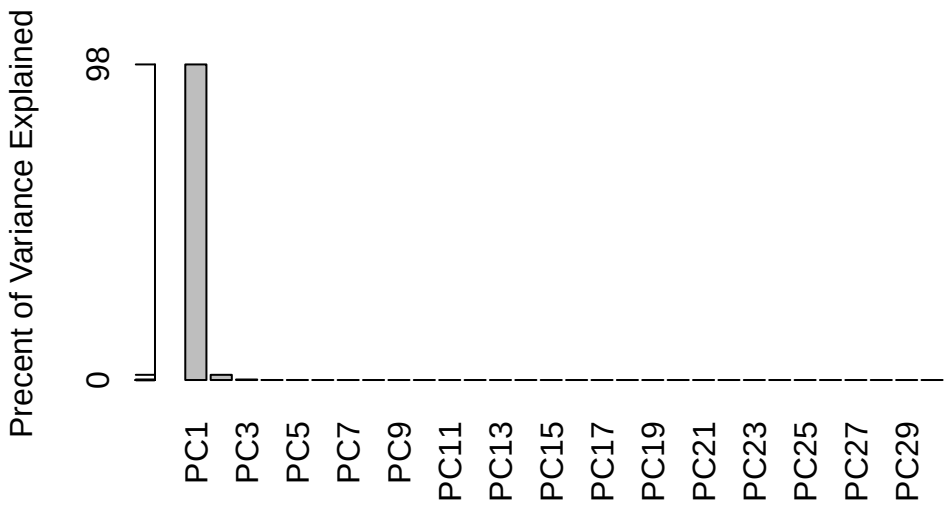
```
pve <- pr.var / sum(pr.var)

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



An alternative:

```
# 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 )
```

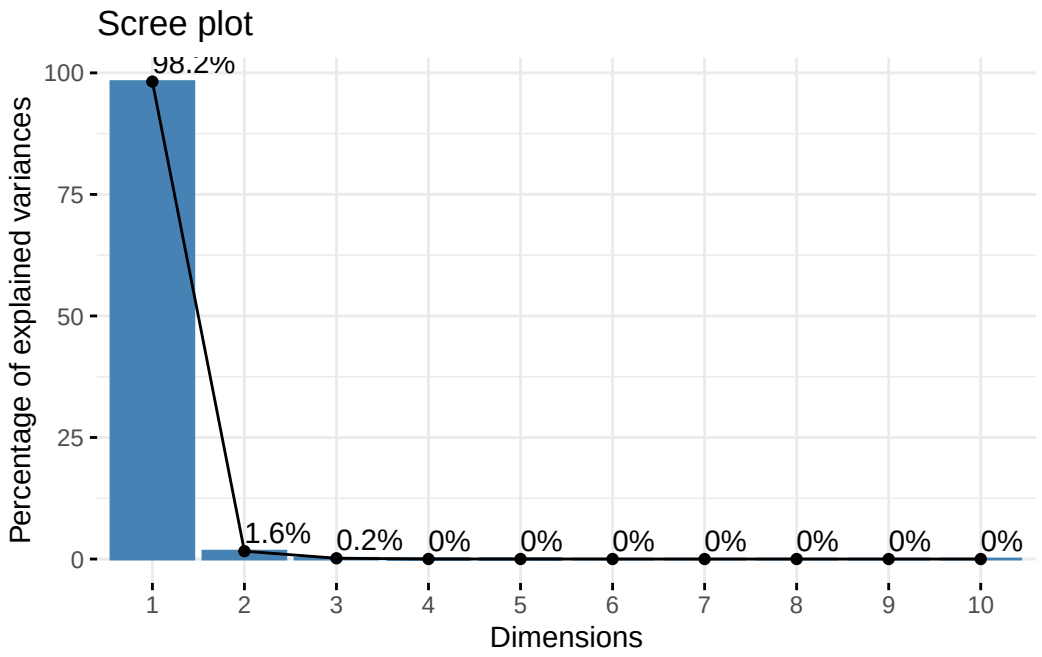


Exploring factoextra ## Using install.packages("factoextra")

```
library(factoextra)
```

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

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



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

```
[1] -4.778078e-05
```

A9. The loading value for the first principal component of the feature concave.points_mean is -4.778e-05.

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

A10. Still just PC1, so one!

Hierarchical Clustering

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

Euclidian distances:

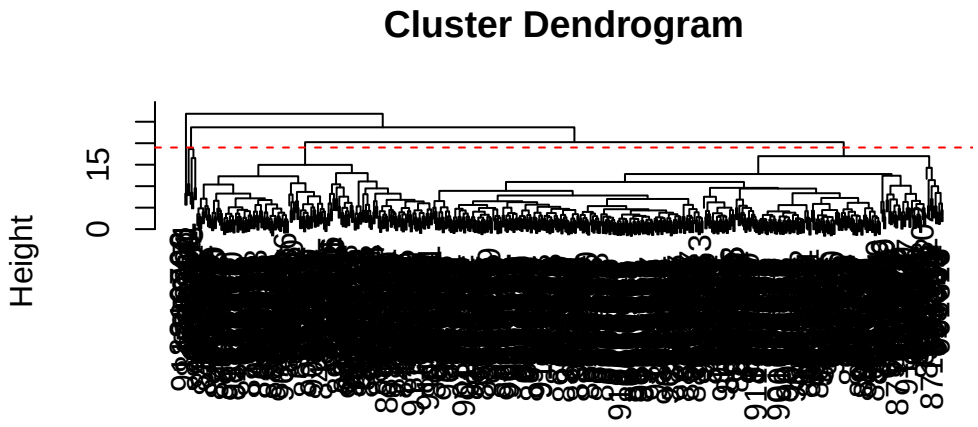
```
data.dist <- dist(data.scaled)
```

Now, a complete linkage model of hierarchical clustering.

```
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(h=19, col="red", lty=2)
```



A11. At approximately h=19, the model has 4 clusters

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

```
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.clusters <- cutree(wisc.hclust, h=14)
table(wisc.hclust.clusters, diagnosis)
```

```
                   diagnosis
wisc.hclust.clusters   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
```

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

A13. I prefer the "average" method, as average as a statistical analysis tool provides a lot of information in the context of other information (especially sdev)

Combining methods

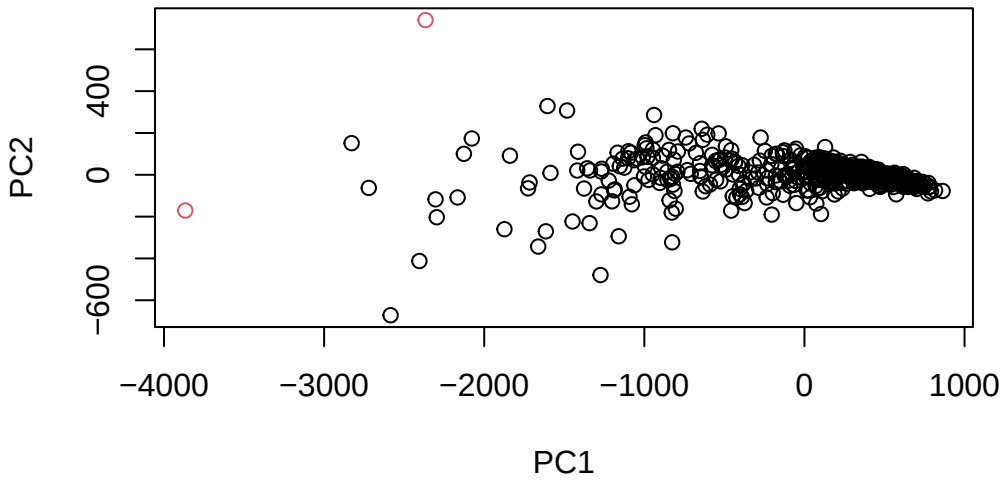
```
grps <- cutree(wisc.hclust, k=2)
table(grps)
```

```
grps
  1   2
567   2
```

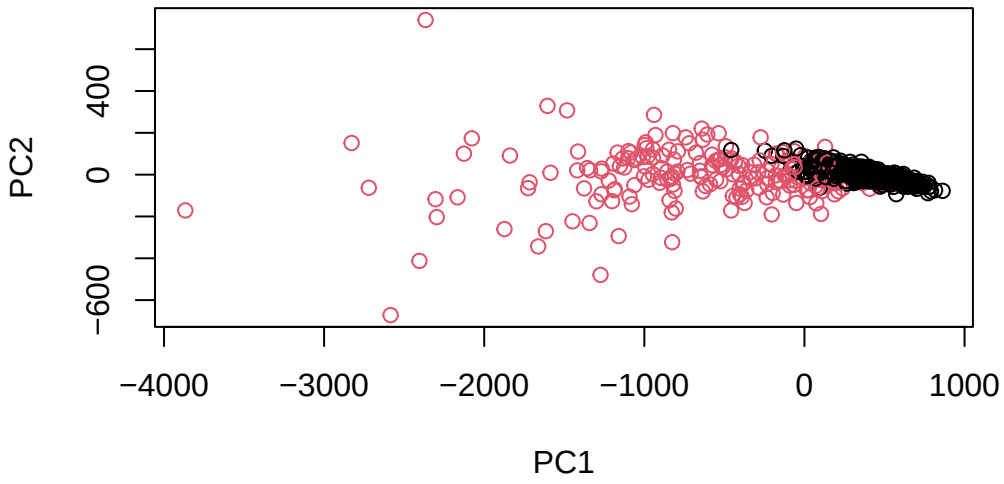
```
table(grps, diagnosis)
```

```
    diagnosis
grps   B   M
   1 357 210
   2   0   2
```

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




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



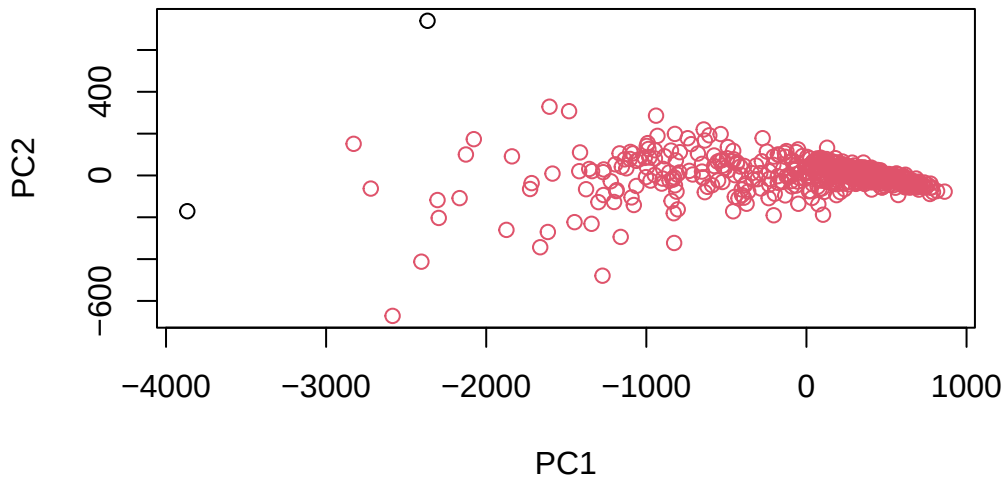
```
g <- as.factor(grps)
levels(g)
```

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

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

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

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

```
wisc.pr.hclust.clusters
  1   2
 86 483
```

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   0  86
                      2 357 126
```

A15. Not so well... There were 212 actual M and here there were 86 in the first cluster and 126 in the second, which adds up! But, cluster 2 still has a big mix between B and M 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.

```
## Note, I didn't do km clusters
table(wisc.hclust.clusters, diagnosis)
```

```
                    diagnosis
wisc.hclust.clusters   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
```

A16. The hierarchial model does a lot better separating, as I see many 0s and good separation in cluster 1, 2, and 4 rather than a big mix of both diagnoses.

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

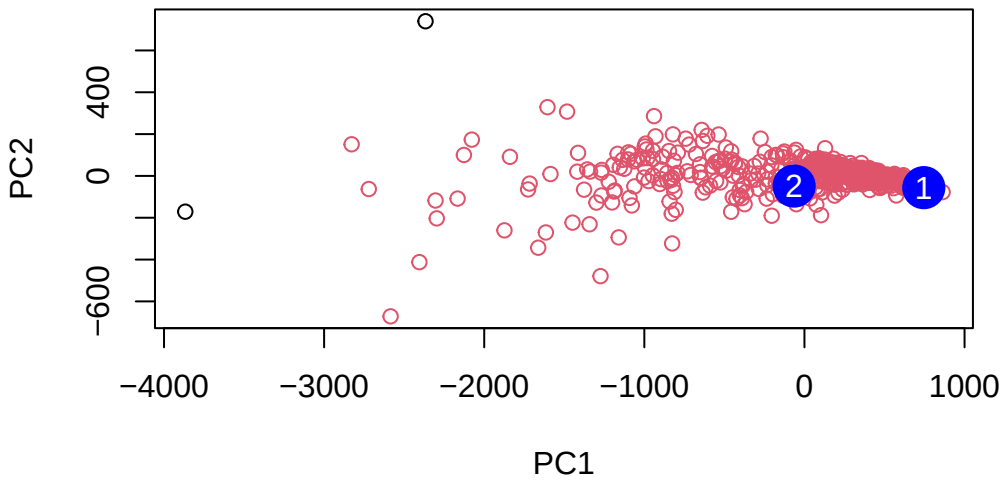
A17. The PCA was best for specificity, while the hierarchial clustering with method=complete was good for sensitivity.

Prediction

```
#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,] 745.60081 -56.16454 -21.15609 -3.330663  9.355518  2.317462 -1.147268
[2,] -64.40839 -48.46996 -15.93413 12.089591 -4.636008 -1.045210 -0.295228
           PC8         PC9        PC10       PC11        PC12         PC13
[1,] -0.7644759  0.11704582  0.06401851 0.1191717 -0.05611973 -0.040020096
[2,] -0.7454142 -0.09167106 -0.76173550 0.3206674  0.02602751  0.005023528
           PC14         PC15        PC16        PC17        PC18         PC19
[1,]  0.01354667 -0.018755904 -0.01050870 -0.01183961 0.020946097  0.030567858
[2,] -0.11943490  0.008958015  0.03391077 -0.02468455 0.008002482 -0.006896744
            PC20         PC21        PC22         PC23         PC24
[1,] -0.007960122 -0.003773165 0.018561168 0.0001875602 -0.005463212
[2,]  0.007001178 -0.022182056 0.008725155 0.0075849336  0.004619616
            PC25        PC26         PC27         PC28         PC29
[1,] -0.005992320 0.005357732 4.550233e-05  0.003252776 0.0012510265
[2,]  0.002804663 0.003229335 1.977351e-03 -0.002261832 0.0009130702
             PC30
[1,] -0.0009794321
[2,] -0.0009078383
```

```
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?

A18. Both patient 1 and 2 are at risk, but patient 1 because it is in an extreme position in the context of the rest of the points than 2 is.