mini-project

Joshua Martin (PID: A18545389)

Table of contents

1 Exploratory Data Analysis

1.1 Background

The goal of this mini-project is for you to explore a complete analysis using the unsupervised learning techniques covered in class. You'll extend what you've learned by combining PCA as a preprocessing step to clustering using data that consist of measurements of cell nuclei of human breast masses. This expands on our RNA-Seq analysis from last day.

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.

2 Data import

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

2.1 Examine Data

```
head(wisc.df)
```

2.2 New data frame removing first row (diagnosis column)

```
wisc.data <- wisc.df[,-1]
View(wisc.data)
head(wisc.df)
```

```
         diagnosis radius_mean texture_mean perimeter_mean area_mean
842302           M       17.99        10.38         122.80    1001.0
842517           M       20.57        17.77         132.90    1326.0
84300903         M       19.69        21.25         130.00    1203.0
84348301         M       11.42        20.38          77.58     386.1
84358402         M       20.29        14.34         135.10    1297.0
843786           M       12.45        15.70          82.57     477.1
         smoothness_mean compactness_mean concavity_mean concave.points_mean
842302           0.11840          0.27760         0.3001             0.14710
842517           0.08474          0.07864         0.0869             0.07017
84300903         0.10960          0.15990         0.1974             0.12790
84348301         0.14250          0.28390         0.2414             0.10520
84358402         0.10030          0.13280         0.1980             0.10430
843786           0.12780          0.17000         0.1578             0.08089
         symmetry_mean fractal_dimension_mean radius_se texture_se perimeter_se
842302          0.2419                0.07871    1.0950     0.9053        8.589
842517          0.1812                0.05667    0.5435     0.7339        3.398
84300903        0.2069                0.05999    0.7456     0.7869        4.585
84348301        0.2597                0.09744    0.4956     1.1560        3.445
84358402        0.1809                0.05883    0.7572     0.7813        5.438
843786          0.2087                0.07613    0.3345     0.8902        2.217
         area_se smoothness_se compactness_se concavity_se concave.points_se
842302    153.40      0.006399        0.04904      0.05373           0.01587
842517     74.08      0.005225        0.01308      0.01860           0.01340
84300903   94.03      0.006150        0.04006      0.03832           0.02058
84348301   27.23      0.009110        0.07458      0.05661           0.01867
84358402   94.44      0.011490        0.02461      0.05688           0.01885
843786     27.19      0.007510        0.03345      0.03672           0.01137
         symmetry_se fractal_dimension_se radius_worst texture_worst
842302       0.03003             0.006193        25.38         17.33
842517       0.01389             0.003532        24.99         23.41
```

```
84300903     0.02250             0.004571        23.57         25.53
84348301     0.05963             0.009208        14.91         26.50
84358402     0.01756             0.005115        22.54         16.67
843786       0.02165             0.005082        15.47         23.75
         perimeter_worst area_worst smoothness_worst compactness_worst
842302            184.60     2019.0           0.1622            0.6656
842517            158.80     1956.0           0.1238            0.1866
84300903          152.50     1709.0           0.1444            0.4245
84348301           98.87      567.7           0.2098            0.8663
84358402          152.20     1575.0           0.1374            0.2050
843786            103.40      741.6           0.1791            0.5249
         concavity_worst concave.points_worst symmetry_worst
842302            0.7119               0.2654         0.4601
842517            0.2416               0.1860         0.2750
84300903          0.4504               0.2430         0.3613
84348301          0.6869               0.2575         0.6638
84358402          0.4000               0.1625         0.2364
843786            0.5355               0.1741         0.3985
         fractal_dimension_worst
842302                   0.11890
842517                   0.08902
84300903                 0.08758
84348301                 0.17300
84358402                 0.07678
843786                   0.12440
```

```
diagnosis <- as.factor(wisc.df$diagnosis)
View(diagnosis)
```

2.3 Confirm Structures

```
str(wisc.data)
```

```
'data.frame':   569 obs. of  30 variables:
 $ radius_mean            : num  18 20.6 19.7 11.4 20.3 ...
 $ texture_mean           : num  10.4 17.8 21.2 20.4 14.3 ...
 $ perimeter_mean         : num  122.8 132.9 130 77.6 135.1 ...
 $ area_mean              : num  1001 1326 1203 386 1297 ...
 $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
 $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
 $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
 $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
 $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
 $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
 $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
 $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
 $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
 $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
 $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
 $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
 $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
 $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
 $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
 $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
 $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
 $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
 $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
 $ area_worst             : num  2019 1956 1709 568 1575 ...
 $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
 $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
 $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
 $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
 $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
 $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...
```

```r
table(diagnosis)
```

```
diagnosis
  B   M 
357 212 
```

2.4 Questions:

Q1. How many observations are in this dataset?

```r
dim(wisc.df)
```

```
[1] 569  31
```

```r
nrow(wisc.df)
```

```
[1] 569
```

There are 569 observations/patients in the dataset.

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

```r
table(diagnosis)
```

```
diagnosis
  B   M 
357 212 
```

There are 212 malignant (M) and 357 benign (B) cases.

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

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

```
[1] 10
```

There are 10 variables ending in _mean.

3 Principal Component Analysis

The `prcomp()` function to do PCA has a `scale=FALSE` default. In general we always want to set this to TRUE so our analysis is not dominated by columns/variables in our dataset that have high standard deviation and mean when compared to others just because the units of measurement are on different scales.

4 Check column means and standard deviations

```
colnames(wisc.data)
```

```
 [1] "radius_mean"             "texture_mean"           
 [3] "perimeter_mean"          "area_mean"              
 [5] "smoothness_mean"         "compactness_mean"       
 [7] "concavity_mean"          "concave.points_mean"    
 [9] "symmetry_mean"           "fractal_dimension_mean" 
[11] "radius_se"               "texture_se"             
[13] "perimeter_se"            "area_se"                
[15] "smoothness_se"           "compactness_se"         
[17] "concavity_se"            "concave.points_se"      
[19] "symmetry_se"             "fractal_dimension_se"   
[21] "radius_worst"            "texture_worst"          
[23] "perimeter_worst"         "area_worst"             
[25] "smoothness_worst"        "compactness_worst"      
[27] "concavity_worst"         "concave.points_worst"   
[29] "symmetry_worst"          "fractal_dimension_worst"
```

```
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, scale = TRUE)
```

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

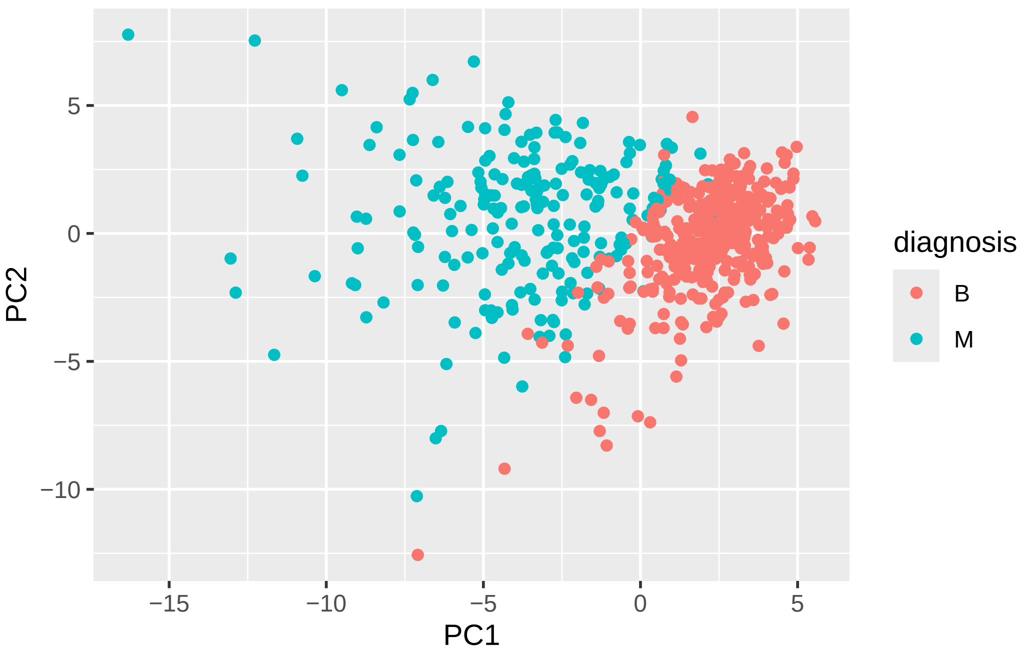
```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172
Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251
Cumulative Proportion  0.4427 0.6324 0.72636 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
```

The main PC result figure is called a “score plot” or “PC plot” or “ordination plot”…

```
library(ggplot2)
wisc.pr$x
```

```
library(ggplot2)

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



4.1 Questions

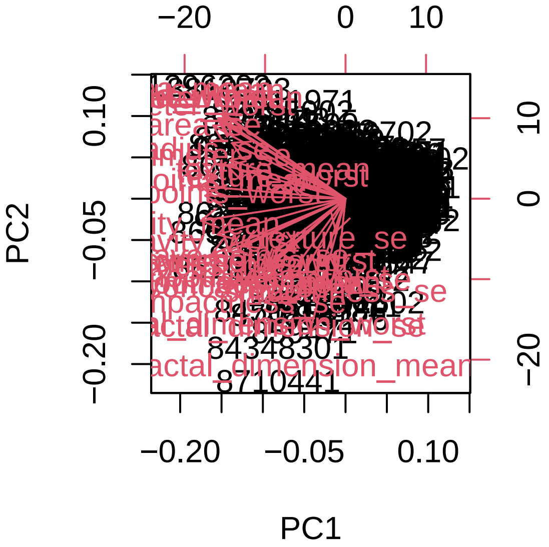
Q4. From your results, what proportion of the original variance is captured by the first principal components (PC1)? PC1 captures approximately 44.3% of the total variance in the dataset.

Q5. How many principal components (PCs) are required to describe at least 70% of the original variance in the data? At least 3 principal components (PC1-PC3) are needed to explain at least 70% of the variance.

Q6. How many principal components (PCs) are required to describe at least 90% of the original variance in the data? At least 7 principal components (PC1-PC7) are needed to explain at least 90% of the variance.

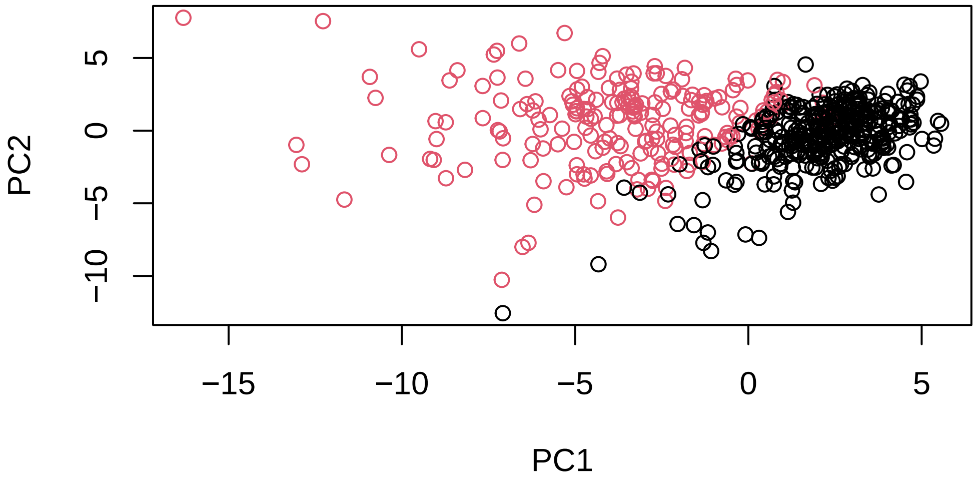
4.2 Create Biplot

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



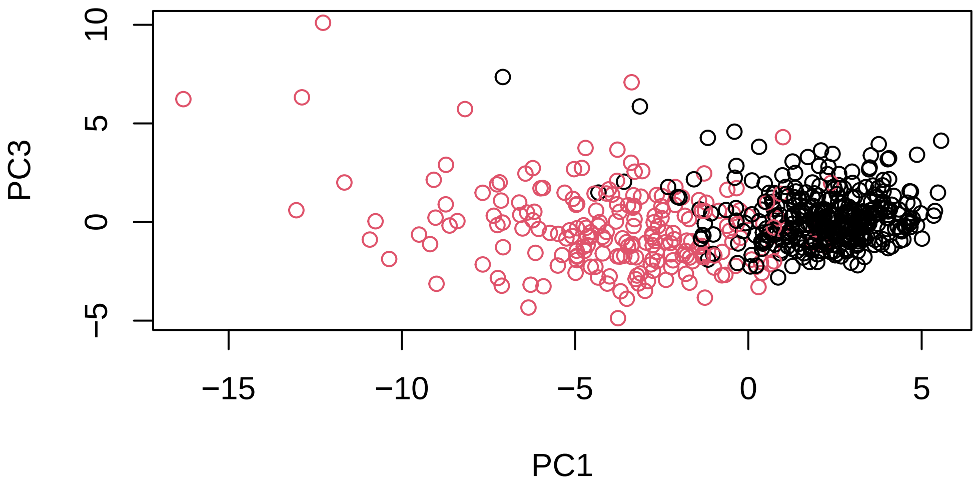
Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why? This plot is messy and difficult to analyze.

```r
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

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

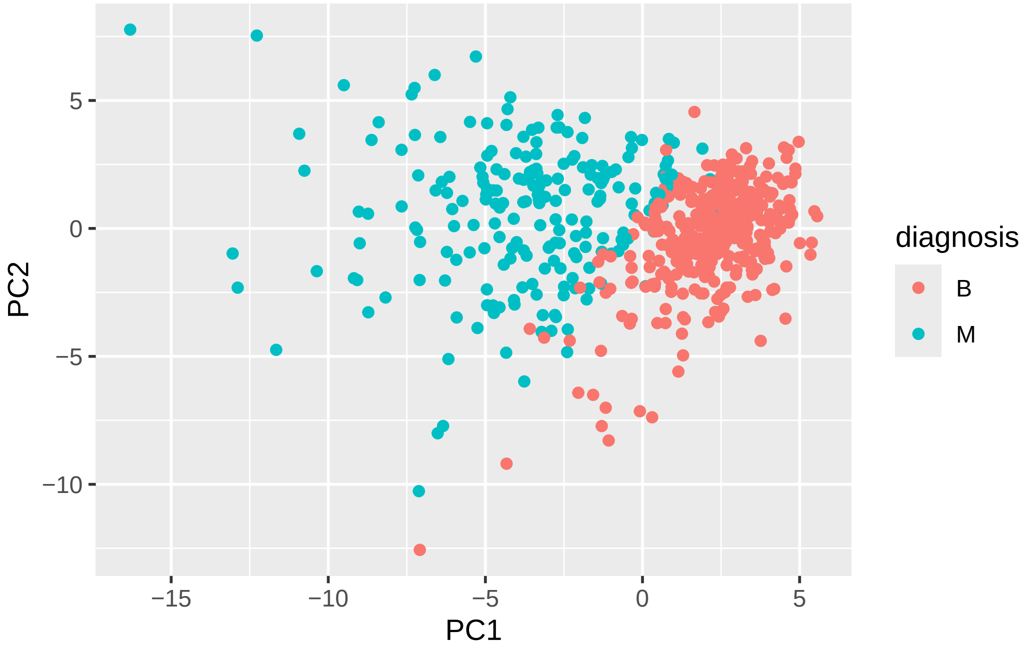


In each, there is strong clustering/less separation within the PC2 and PC3 groups, and strong separation along the PC1.

```
# 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()
```

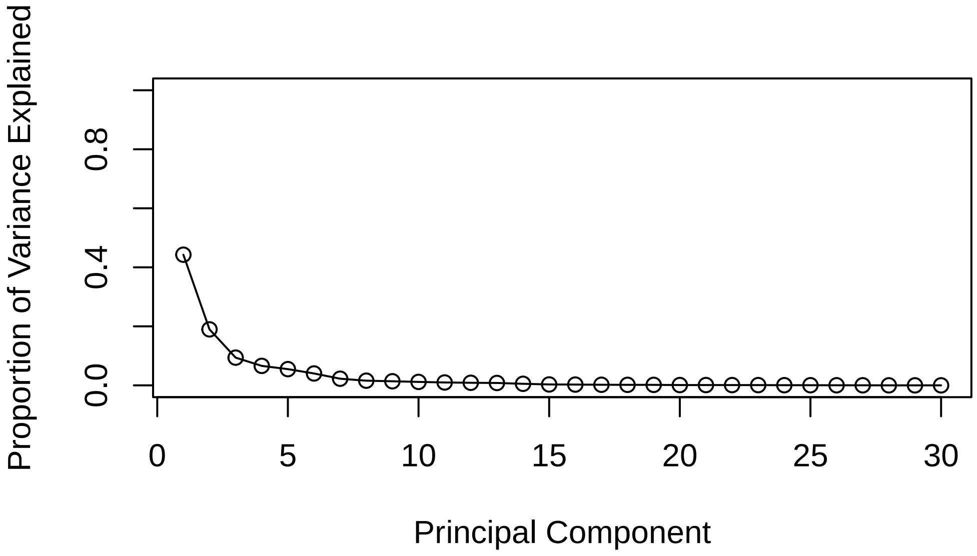


```
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

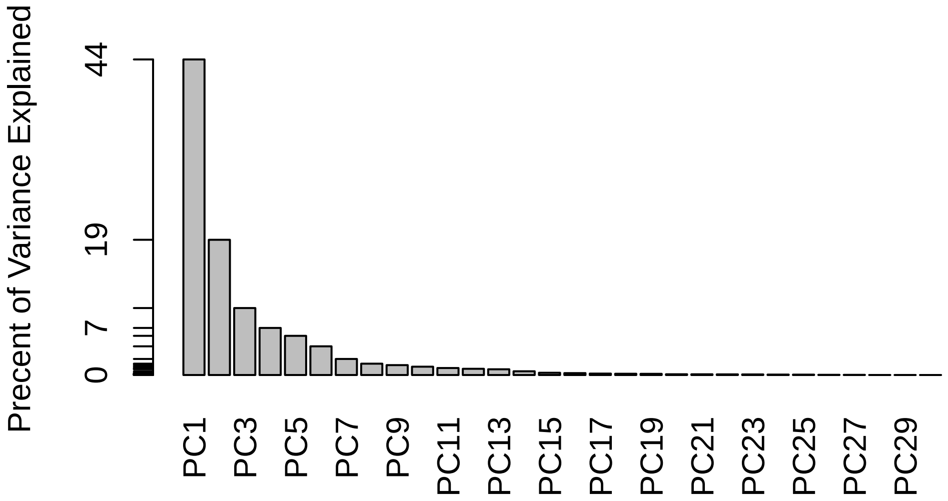
```
[1] 13.281608  5.691355  2.817949  1.980640  1.648731  1.207357
```

```
# 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 )
```

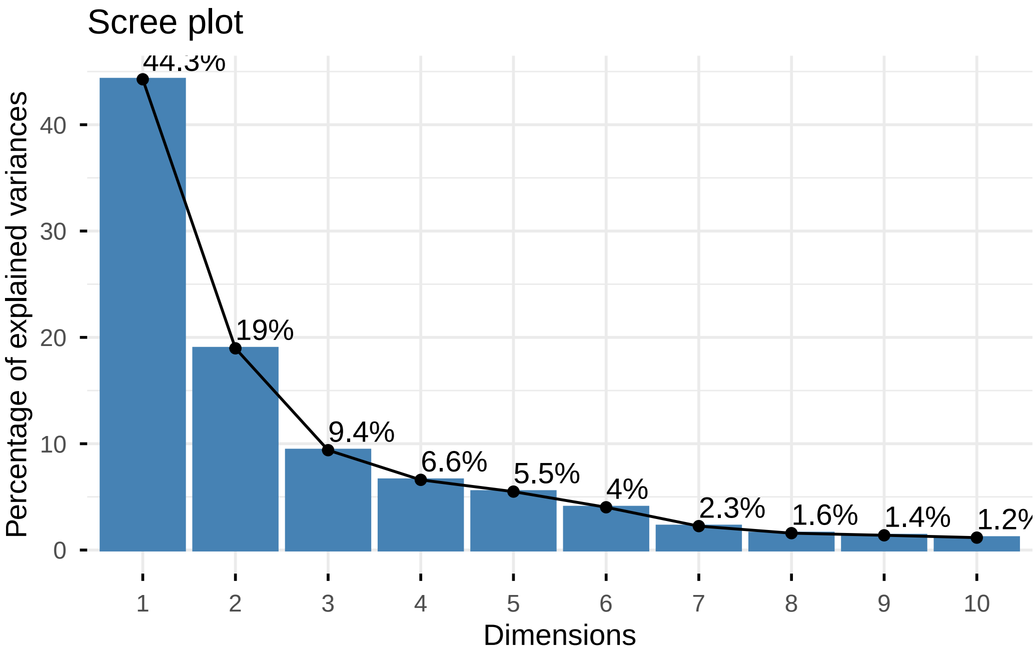


```r
## ggplot based graph
#install.packages("factoextra")
library(factoextra)
```

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

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

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



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 loading of the concave.points_mean on PC1 is approximately `wisc.pr$rotation["concave.points_mean", 1]`. So, higher values of concave.points_mean correspond to lower PC1 scores. PC1 separates malignant and benign cases, helping to distinguish them.

Q10. What is the minimum number of principal components required to explain 80% of the variance of the data? To explain at least 80% of the total variance, 4 principal components (PC1-PC$) is needed.

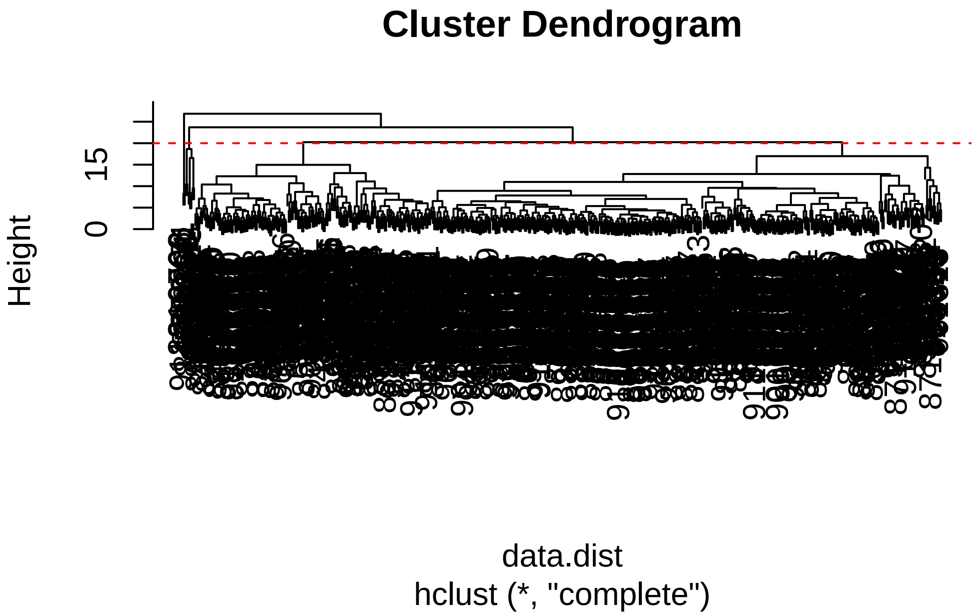
5 Hierarchical clustering

```
# 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")
```

5.1 Results of Hierarchical Clustering

```
plot(wisc.hclust)
abline(h = 20, col = "red", lty = 2)
```



```
table(cutree(wisc.hclust,k=4))
```

```
  1   2   3   4 
177   7 383   2 
```

This looks terrible.

Q11. Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters? The height at which 4 clusters occur is 20.

5.2 Selecting number of clusters

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

# Compare cluster assignments 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
```

```r
for (k in 2:10) {
  cat("\nNumber of clusters:", k, "\n")
  print(table(cutree(wisc.hclust, k = k), diagnosis))
}
```

```

Number of clusters: 2
   diagnosis
      B   M
  1 357 210
  2   0   2

Number of clusters: 3
   diagnosis
      B   M
  1 355 205
  2   2   5
  3   0   2

Number of clusters: 4
   diagnosis
      B   M
  1  12 165
  2   2   5
  3 343  40
  4   0   2

Number of clusters: 5
   diagnosis
      B   M
  1  12 165
  2   0   5
  3 343  40
  4   2   0
  5   0   2

Number of clusters: 6
   diagnosis
      B   M
  1  12 165
  2   0   5
  3 331  39
  4   2   0
  5  12   1
  6   0   2

Number of clusters: 7
   diagnosis
      B   M
  1  12 165
  2   0   3
  3 331  39
  4   2   0
  5  12   1
  6   0   2
  7   0   2

Number of clusters: 8
   diagnosis
      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
```

```
Number of clusters: 9
   diagnosis
      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
```

```
Number of clusters: 10
    diagnosis
       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
```

Q12. Can you find a better cluster vs diagnoses match by cutting into a different number of clusters between 2 and 10? Cutting the dendrogram into 4 clusters gives the best match to the true diagnoses. One cluster is mostly malignant, the other is mostly benign. Fewer clusters mix the two groups, and does not improve separation.

Q13. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning. The ward.D2 method gives my preferred result. It creates clearer, interpretable groupings for this dataset.

6 K-means Clustering

```
# Create a k-means model with 2 clusters, scaled data, and 20 random starts
wisc.km <- kmeans(scale(wisc.data), centers = 2, nstart = 20)

# Compare k-means cluster membership to actual diagnoses
table(wisc.km$cluster, diagnosis)
```

```
   diagnosis
      B   M
  1 343  37
  2  14 175
```

Q14. How well does k-means separate the two diagnoses? How does it compare to your hclust results? K-means clustering separates the two diagnoses very well. Cluster 1 is mostly malignant and cluster 2 is mostly benign. K-means clustering is better at distinguishing clusters compared to hierarchical clustering.

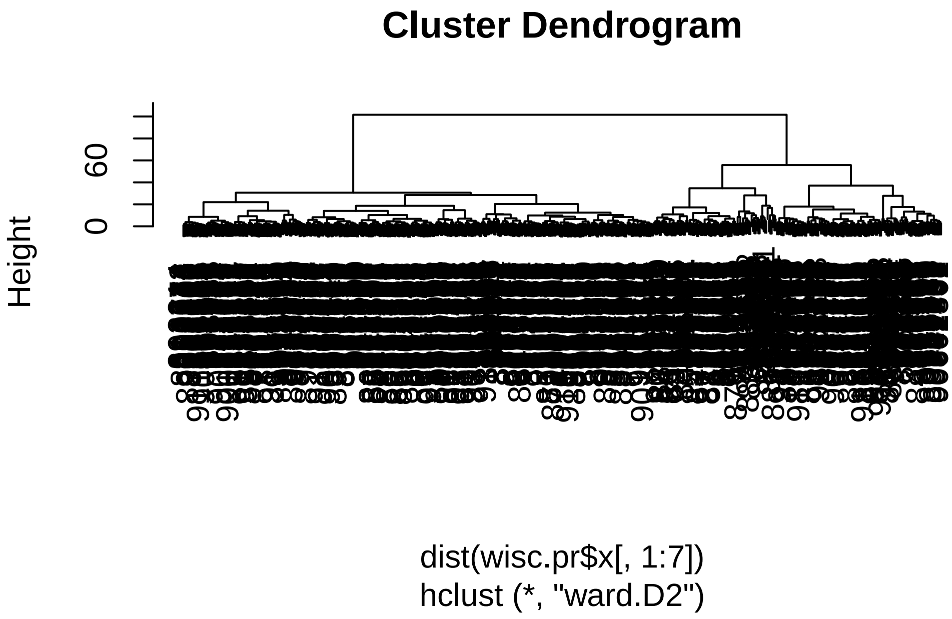
```
# Compare k-means clusters to hierarchical clustering clusters
table(wisc.hclust.clusters, wisc.km$cluster)
```

```
wisc.hclust.clusters   1   2
                   1  17 160
                   2   0   7
                   3 363  20
                   4   0   2
```

7 Combining Methods

```
# Use first 7 PCs ( 90% variance)
wisc.pr.hclust <- hclust(dist(wisc.pr$x[, 1:7]), method = "ward.D2")

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



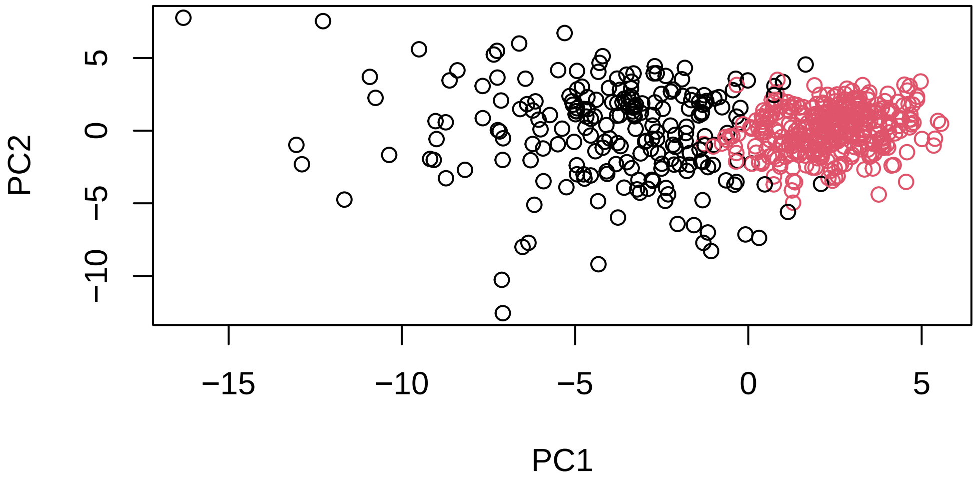
```
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
grps
  1   2 
216 353 
```

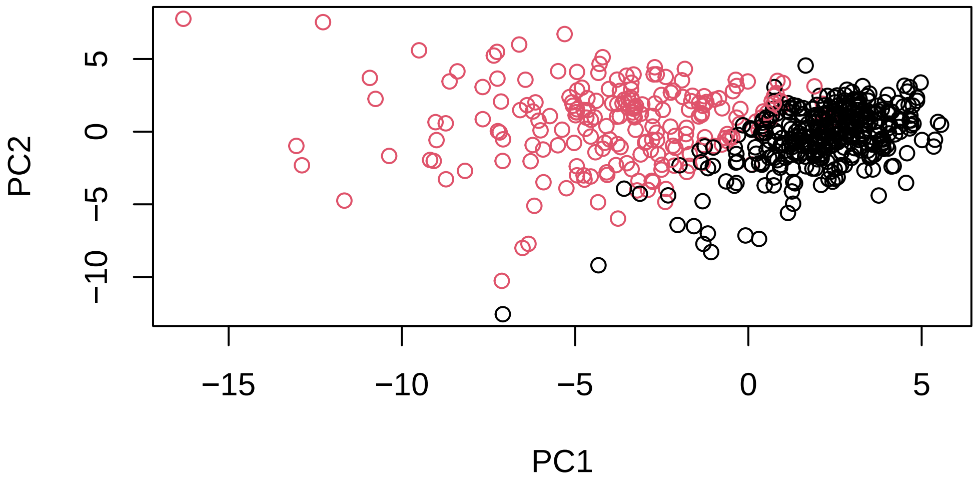
```
table(grps, diagnosis)
```

```
    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)
```



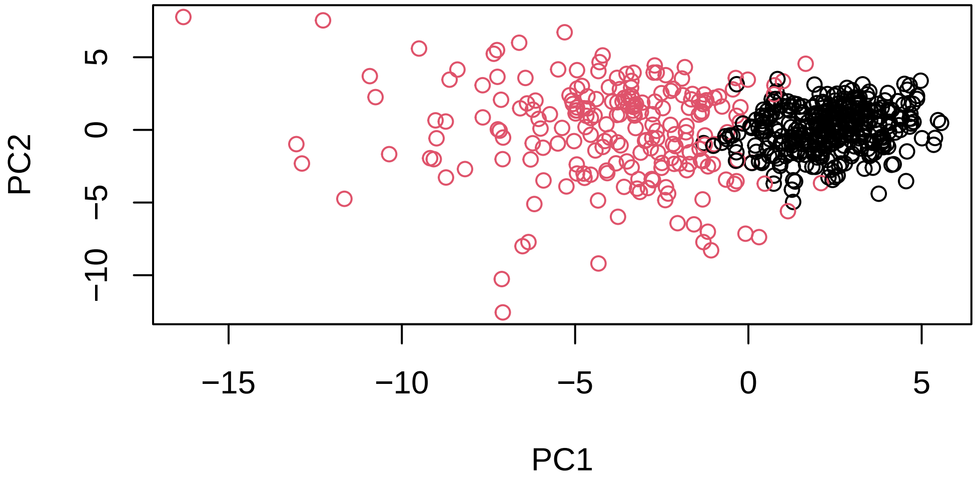
```r
g <- as.factor(grps)
levels(g)
```

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

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

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

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



```r
library(rgl)
plot3d(wisc.pr$x[,1:3], xlab="PC 1", ylab="PC 2", zlab="PC 3", cex=1.5, size=1, type="s", col
```

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