Class 9 Mini Project- Breast Cancer Cell Analysis

Vera Sophia Beliaev

2/13/2022

1. Exploratory data analysis

Preparing the data

```
#Saving the input data file into my Project Directory
fna.data <-  "WisconsinCancer.csv"
#Input the data and store it under wisc.df
wisc.df <- read.csv(fna.data, row.names=1)
#Inspect data
View(wisc.df)
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
```

The wisc.df$diagnosis gives us the actual “answer” to whether a sample is benign or malignant, so we will exclude it from our data analysis

```
#Create new data frame that omits diagnosis column
wisc.data <- wisc.df[,-1]
```

```
#Create vector w/ diagnoses
diagnosis <- factor(wisc.df$diagnosis)
diagnosis
```

```
##   [1] M M 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 M M
##  [38] B M 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 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 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 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 B M B
## [482] B B B B B B M B M B B M 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? There are 590 observations in wisc.data

```
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 ...
```

Q2. How many of the observations have a malignant diagnosis? 212 of the observations have a malignant diagnosis.

```
table(diagnosis)
```

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

Q3. How many variables/features in the data are suffixed with _mean? 10 variables in the data are suffixed with _mean.

```
#Not quite sure how to reason this out but google helped
meanvect <- names(wisc.data)[grep("_mean", names(wisc.data))]
length(meanvect)
```

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

2. Principal COmponent Analysis

```
#Check column means and st dev
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, use t() for transpose of the data, use scaling
wisc.pr <- prcomp(wisc.data, scale=TRUE)
```

```
#Summary of the PCA 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)?

The proportion of the original variance captured by the first principal component (PC1) is 44.27%

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

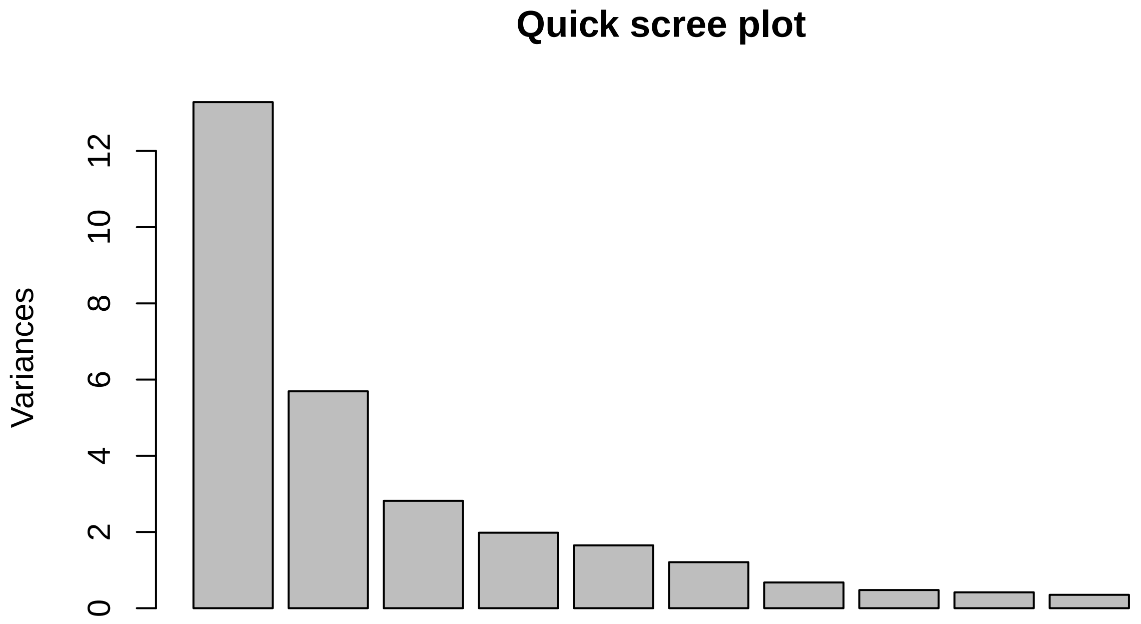
At least 3 PCs are required to describes at least 70% of the original variance in the data.

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

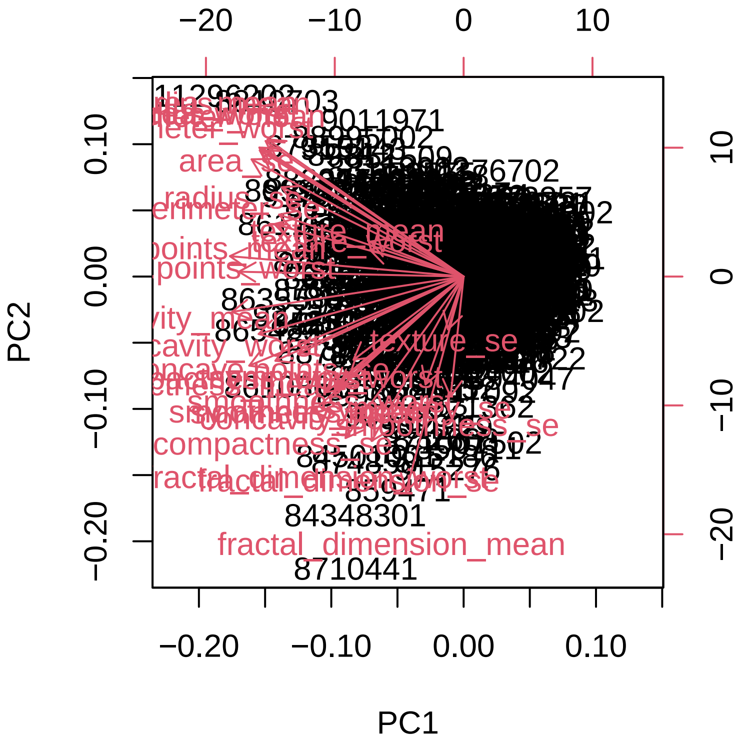
At least 7 PCs are required to describe at least 90% of the original variance in the data.

Interpreting PCA Results

```
#Scree plot
plot(wisc.pr, main= "Quick scree plot")
```



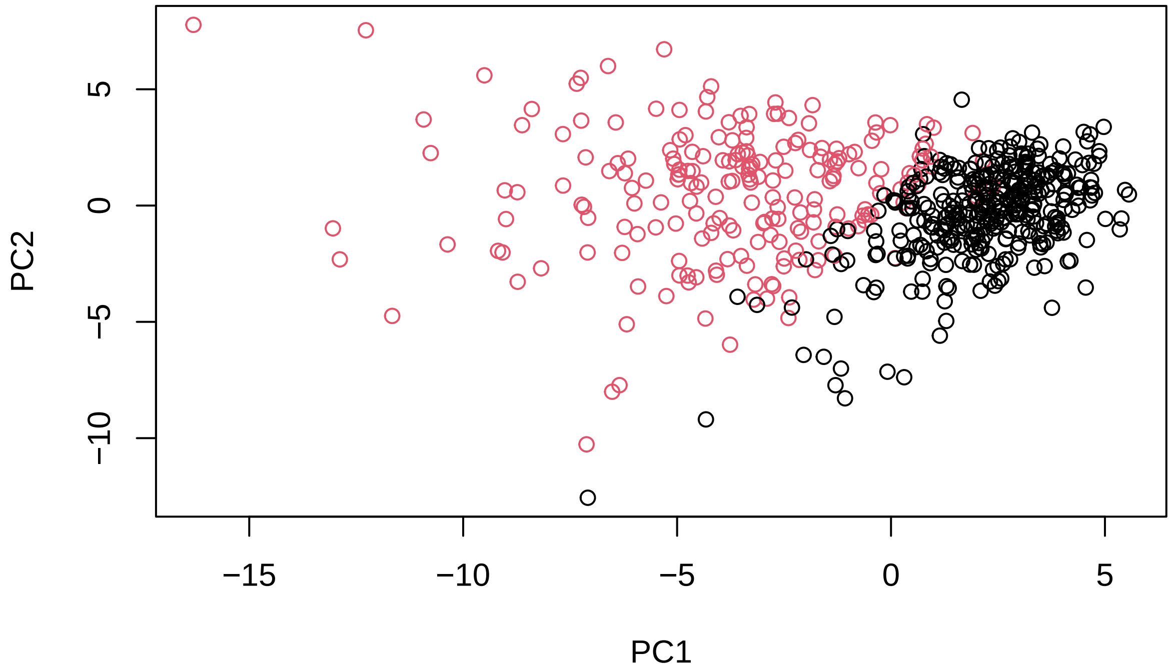
```
#Biplot
biplot(wisc.pr)
```



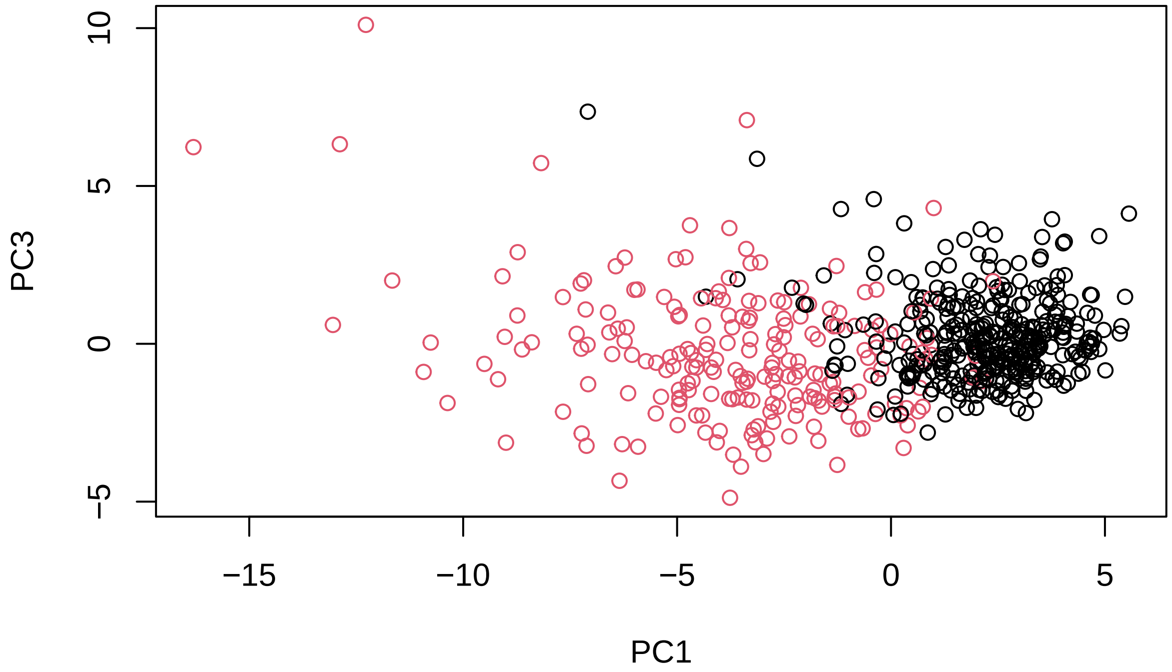
Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

The biplot is basically impossible to interpret. It's unclear and you can't make any conclusions with assurance by looking at it.

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



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



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

The plots are very similar however plotting PC1 vs PC3 shifts the plotted points downwards visually. Overall, there does not seem to be too much difference between the two plots.

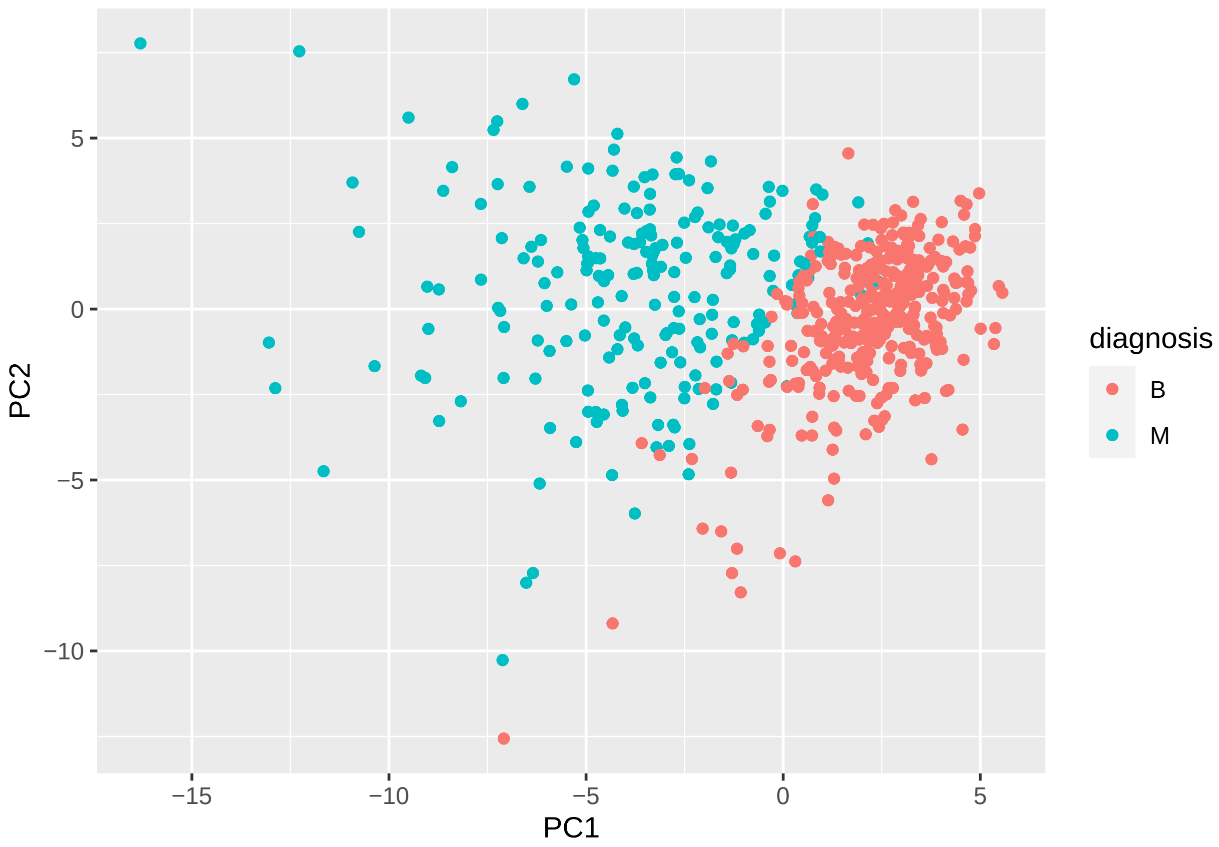
Plot with ggplot2

```
library(ggplot2)
```

ggplot uses data frames for input, and diagnosis vect must be converted to column

```
df <- as.data.frame(wisc.pr$x)
df$diagnosis <- diagnosis
```

```
#Scatter plot colored by dx
ggplot(df) + aes(PC1, PC2, col=diagnosis) + geom_point()
```



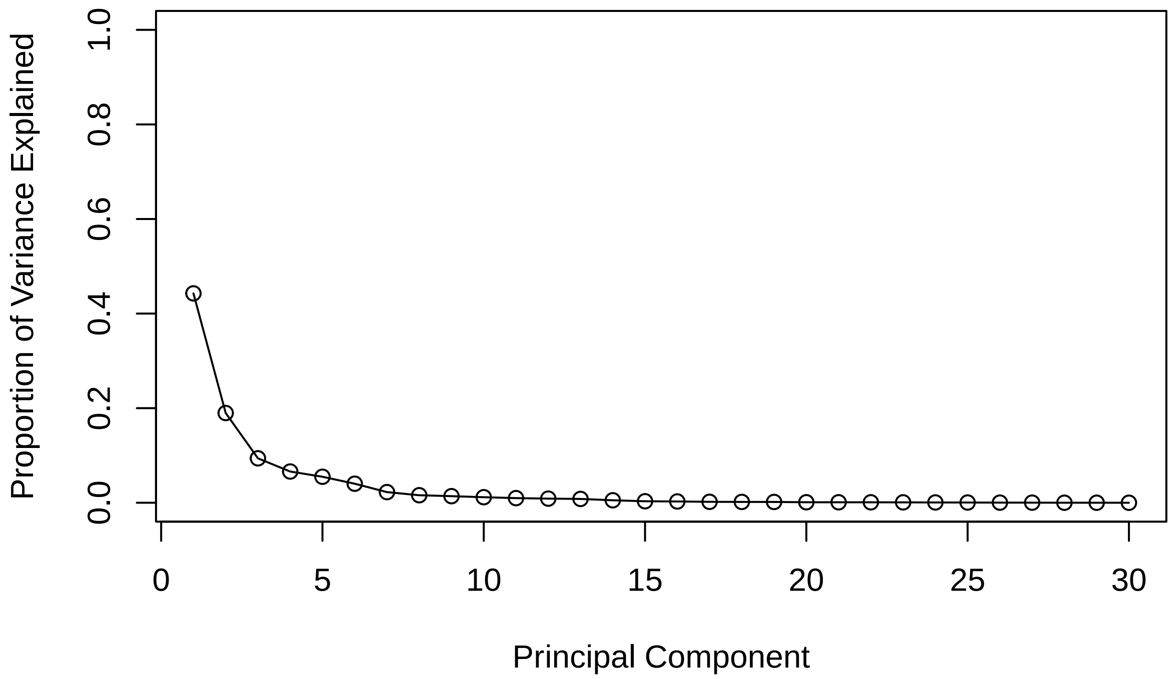
Variance explained

```
#The variance of e/ component
pr.var <-  wisc.pr$sdev^2
head(pr.var)
```

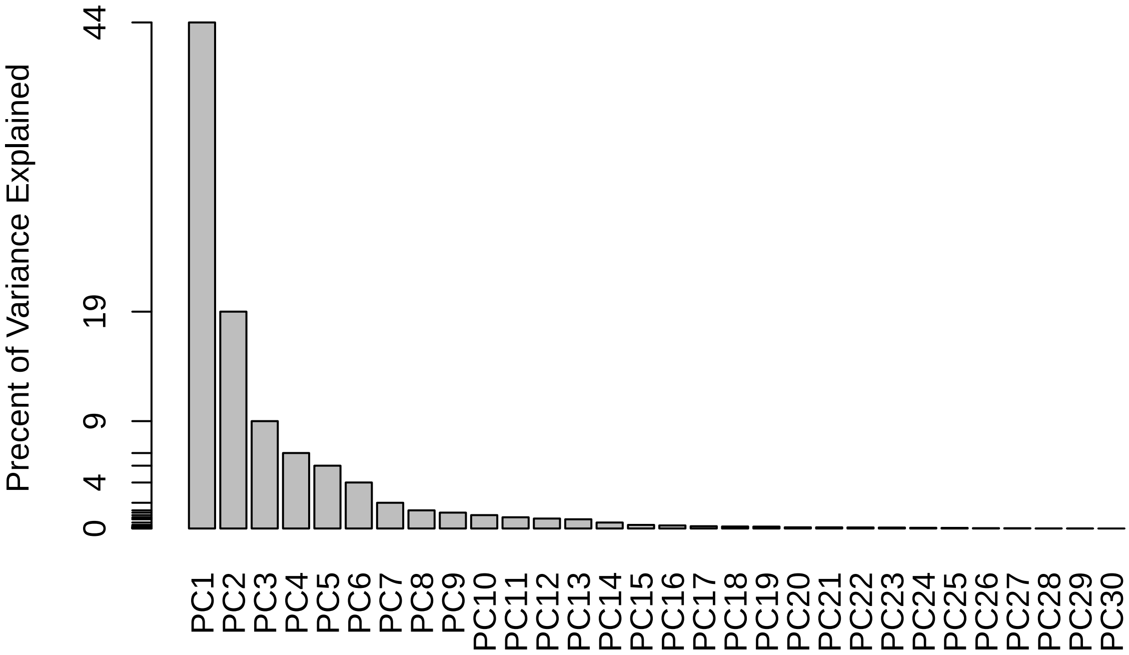
```
## [1] 13.281608  5.691355  2.817949  1.980640  1.648731  1.207357
```

```
#Variance explained by e/ PC
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 )
```



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?

It 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
```

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

At least 5 principal components are required to explain 80% of the variance of the data.

Using alt programs for PCA (ade4 and factoextra)

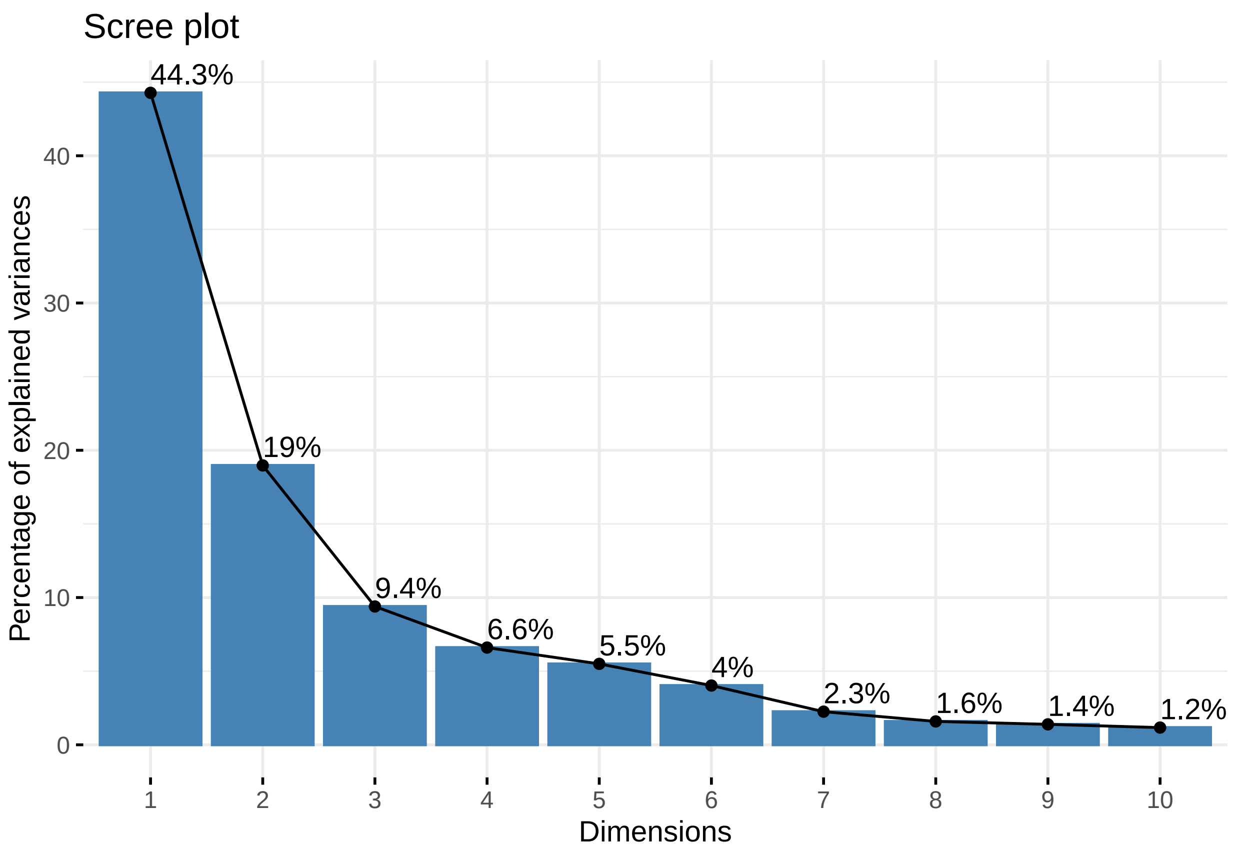
```
#Scree plot
library(ade4)
dudipca <- dudi.pca(df = wisc.data, center = TRUE, scale = TRUE, scannf = FALSE, nf = 30)
dudipca
```

```
## Duality diagramm
## class: pca dudi
## $call: dudi.pca(df = wisc.data, center = TRUE, scale = TRUE, scannf = FALSE,
##     nf = 30)
##
## $nf: 30 axis-components saved
## $rank: 30
## eigen values: 13.28 5.691 2.818 1.981 1.649 ...
##   vector length mode    content
## 1 $cw    30     numeric column weights
## 2 $lw    569    numeric row weights
## 3 $eig   30     numeric eigen values
##
##   data.frame nrow ncol content
## 1 $tab       569  30   modified array
## 2 $li        569  30   row coordinates
## 3 $l1        569  30   row normed scores
## 4 $co        30   30   column coordinates
## 5 $c1        30   30   column normed scores
## other elements: cent norm
```

```
library(factoextra)
```

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

```
#Scree plot
fviz_eig(dudipca, addlabels=TRUE)
```

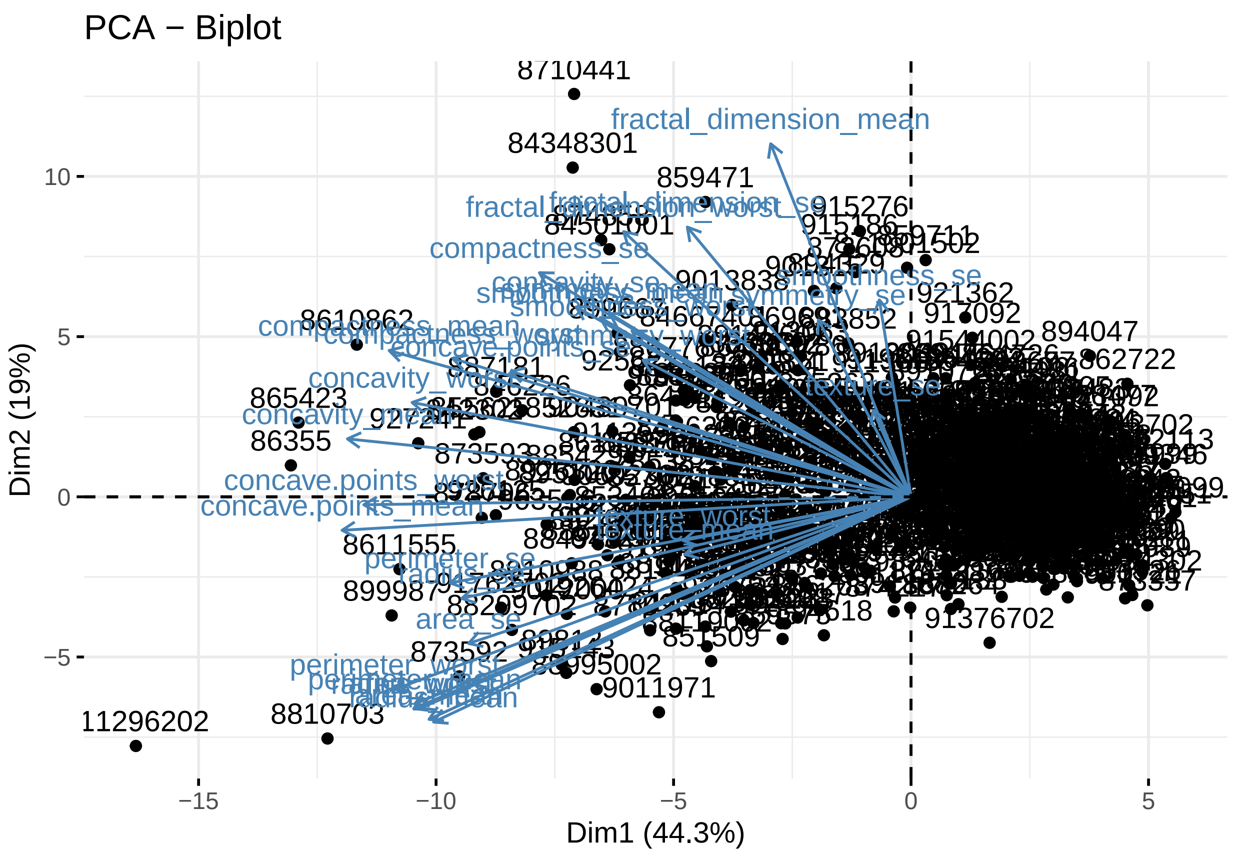


The above Scree plot shows PC1 containing 44.3% of variance, PC2 containing 19%, PC3 containing 9.4% and so on

```
get_eig(dudipca)
```

```
##          eigenvalue variance.percent cumulative.variance.percent
## Dim.1  1.328161e+01     4.427203e+01                    44.27203
## Dim.2  5.691355e+00     1.897118e+01                    63.24321
## Dim.3  2.817949e+00     9.393163e+00                    72.63637
## Dim.4  1.980640e+00     6.602135e+00                    79.23851
## Dim.5  1.648731e+00     5.495768e+00                    84.73427
## Dim.6  1.207357e+00     4.024522e+00                    88.75880
## Dim.7  6.752201e-01     2.250734e+00                    91.00953
## Dim.8  4.766171e-01     1.588724e+00                    92.59825
## Dim.9  4.168948e-01     1.389649e+00                    93.98790
## Dim.10 3.506935e-01     1.168978e+00                    95.15688
## Dim.11 2.939157e-01     9.797190e-01                    96.13660
## Dim.12 2.611614e-01     8.705379e-01                    97.00714
## Dim.13 2.413575e-01     8.045250e-01                    97.81166
## Dim.14 1.570097e-01     5.233657e-01                    98.33503
## Dim.15 9.413497e-02     3.137832e-01                    98.64881
## Dim.16 7.986280e-02     2.662093e-01                    98.91502
## Dim.17 5.939904e-02     1.979968e-01                    99.11302
## Dim.18 5.261878e-02     1.753959e-01                    99.28841
## Dim.19 4.947759e-02     1.649253e-01                    99.45334
## Dim.20 3.115940e-02     1.038647e-01                    99.55720
## Dim.21 2.997289e-02     9.990965e-02                    99.65711
## Dim.22 2.743940e-02     9.146468e-02                    99.74858
## Dim.23 2.434084e-02     8.113613e-02                    99.82971
## Dim.24 1.805501e-02     6.018336e-02                    99.88990
## Dim.25 1.548127e-02     5.160424e-02                    99.94150
## Dim.26 8.177640e-03     2.725880e-02                    99.96876
## Dim.27 6.900464e-03     2.300155e-02                    99.99176
## Dim.28 1.589338e-03     5.297793e-03                    99.99706
## Dim.29 7.488031e-04     2.496010e-03                    99.99956
## Dim.30 1.330448e-04     4.434827e-04                   100.00000
```

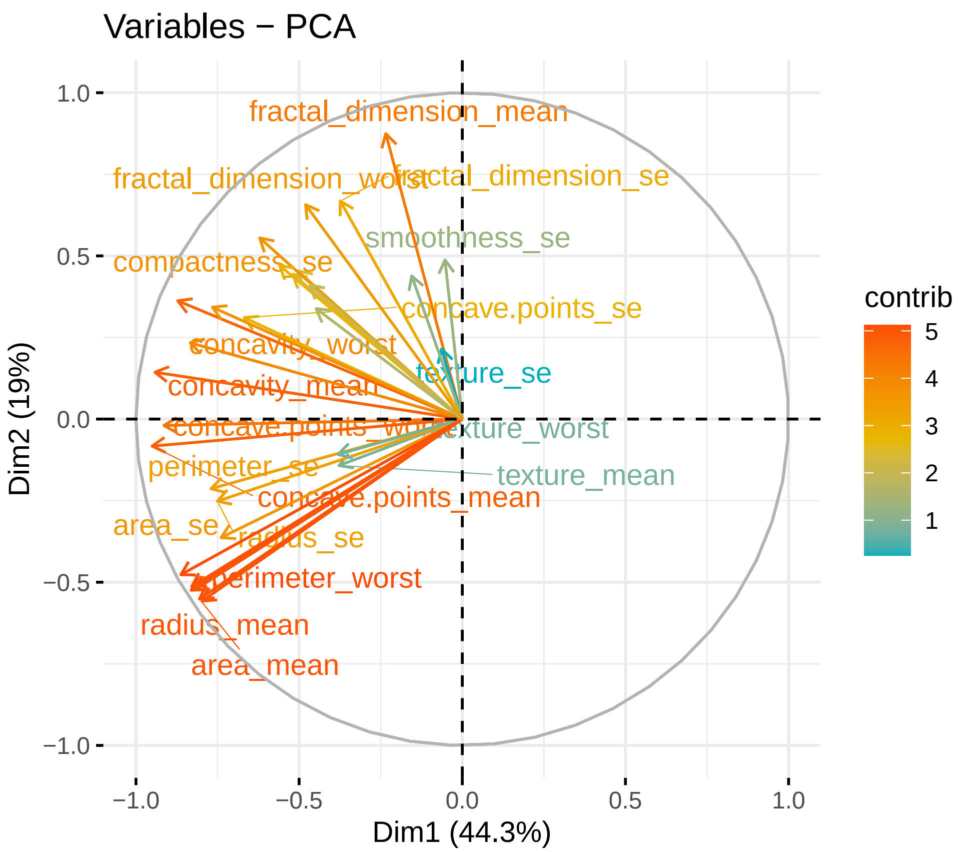
```
fviz_pca(dudipca)
```



The above biplot is extremely complicated

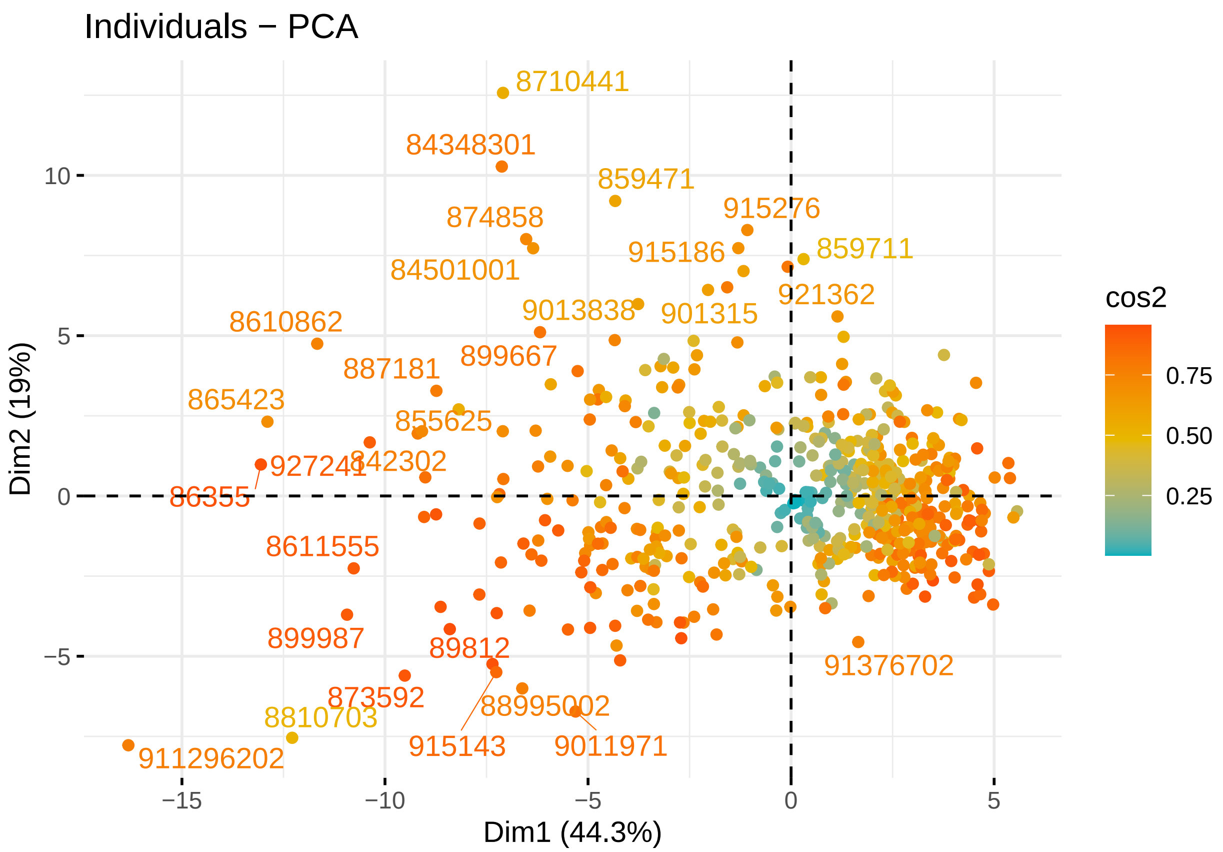
```
fviz_pca_var(dudipca, col.var="contrib",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE # Avoid text overlapping
             )
```

```
## Warning: ggrepel: 11 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

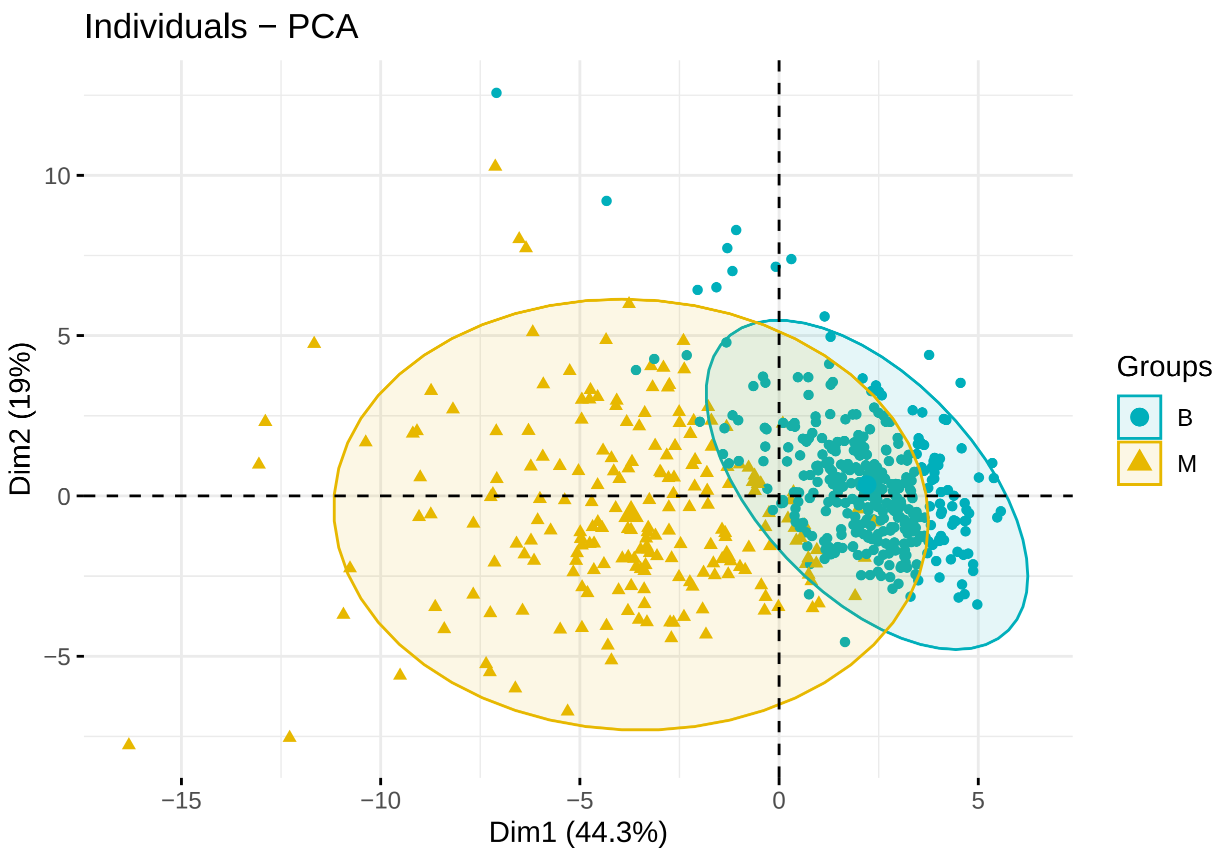


```r
fviz_pca_ind(dudipca, col.ind = "cos2",
             gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
             repel = TRUE # Avoid text overlapping (slow if many points)
             )
```

```
## Warning: ggrepel: 540 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```
fviz_pca_ind(dudipca,
             label = "none", # hide individual labels
             habillage = wisc.df$diagnosis, # color by groups
             palette = c("#00AFBB", "#E7B800"),
             addEllipses = TRUE # Concentration ellipses
             )
```



Hierarchical Clustering

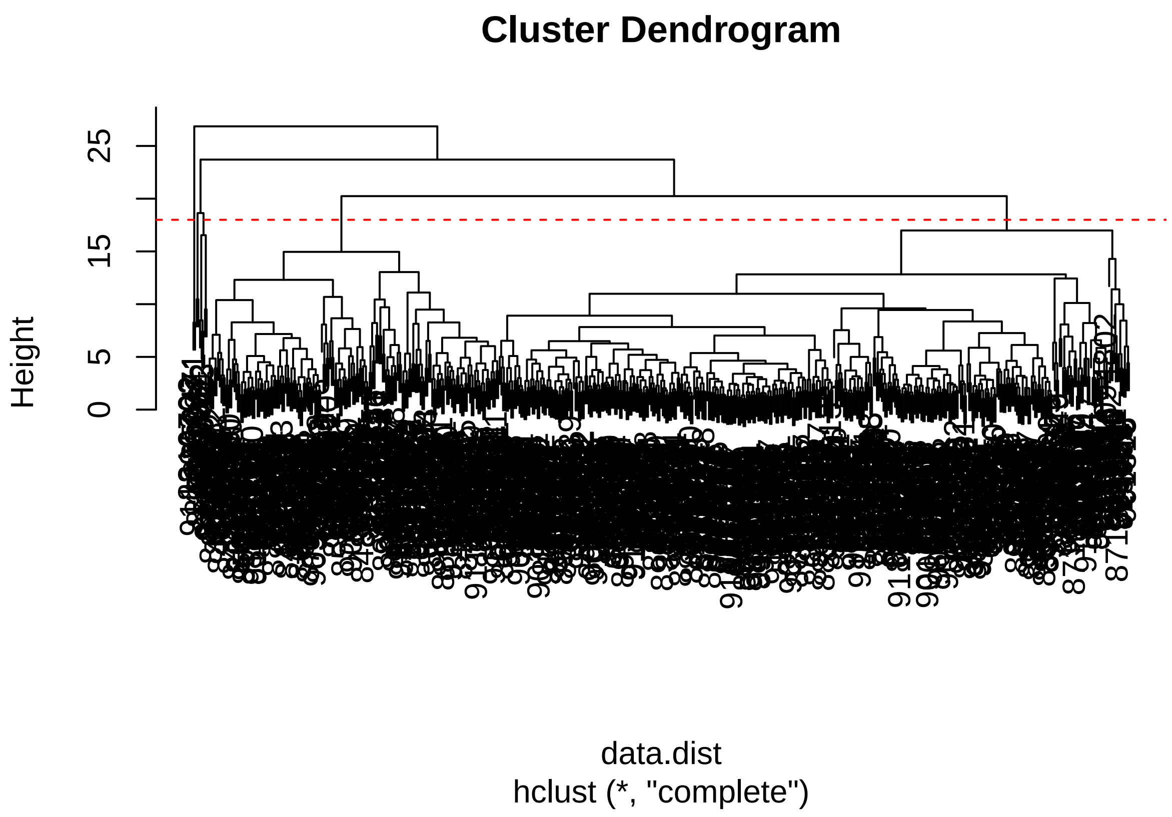
```
#Scale the data
data.scaled <- scale(wisc.data)
```

```
#Calc Euclidean dists b/w pairs of observations
data.dist <-  dist(data.scaled)
```

```
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? The height at which the clustering model has 4 clusters is approximately 18.

```
plot(wisc.hclust)
abline(h=18, col= "red", lty=2)
```



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

```
#COmpare our clusters 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
```

THis showed that cluster 1 has mostly malignant cells, and cluster 3 has mostly benign cells.

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

```
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 205
##                    2   2   5
##                    3   0   2
```

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

```
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 357 210
##                    2   0   2
```

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

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

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

```
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1  12 165
##                    2   0   5
##                    3 331  39
##                    4   2   0
##                    5  12   1
##                    6   0   2
```

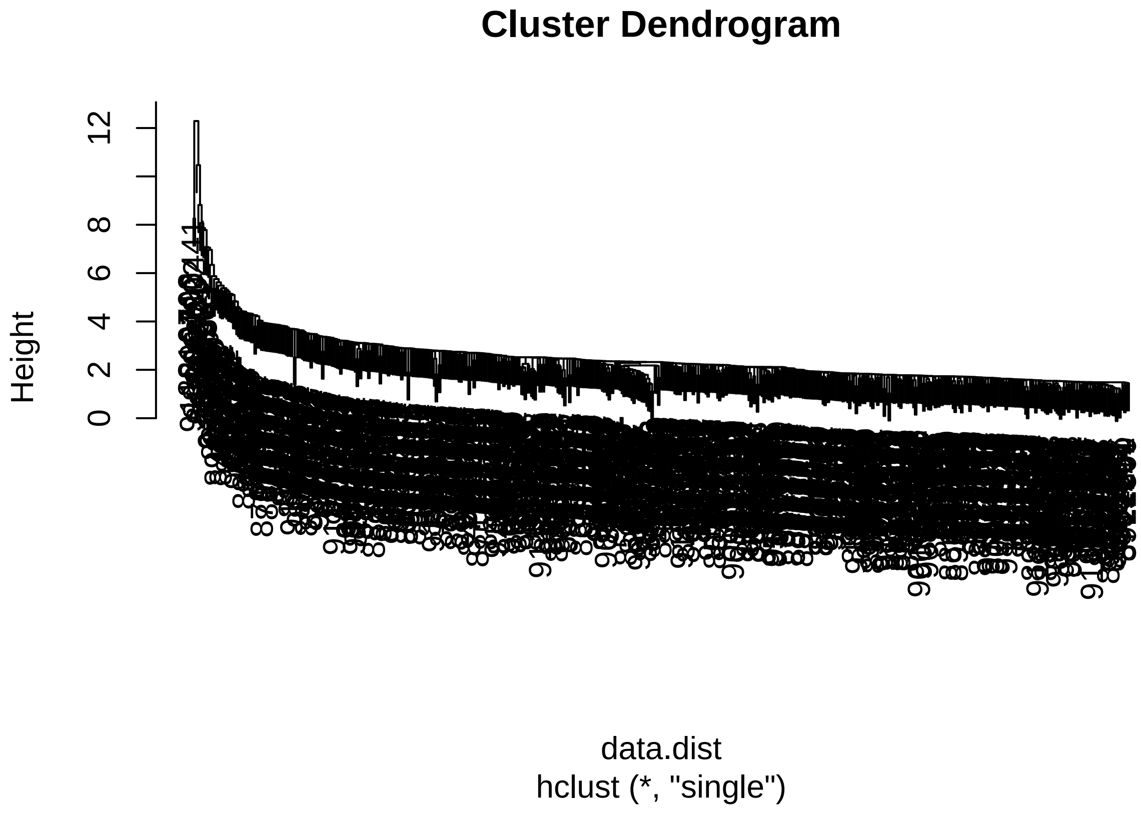
```
wisc.hclust.clusters <- cutree(wisc.hclust, k=10)
table(wisc.hclust.clusters, diagnosis)
```

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

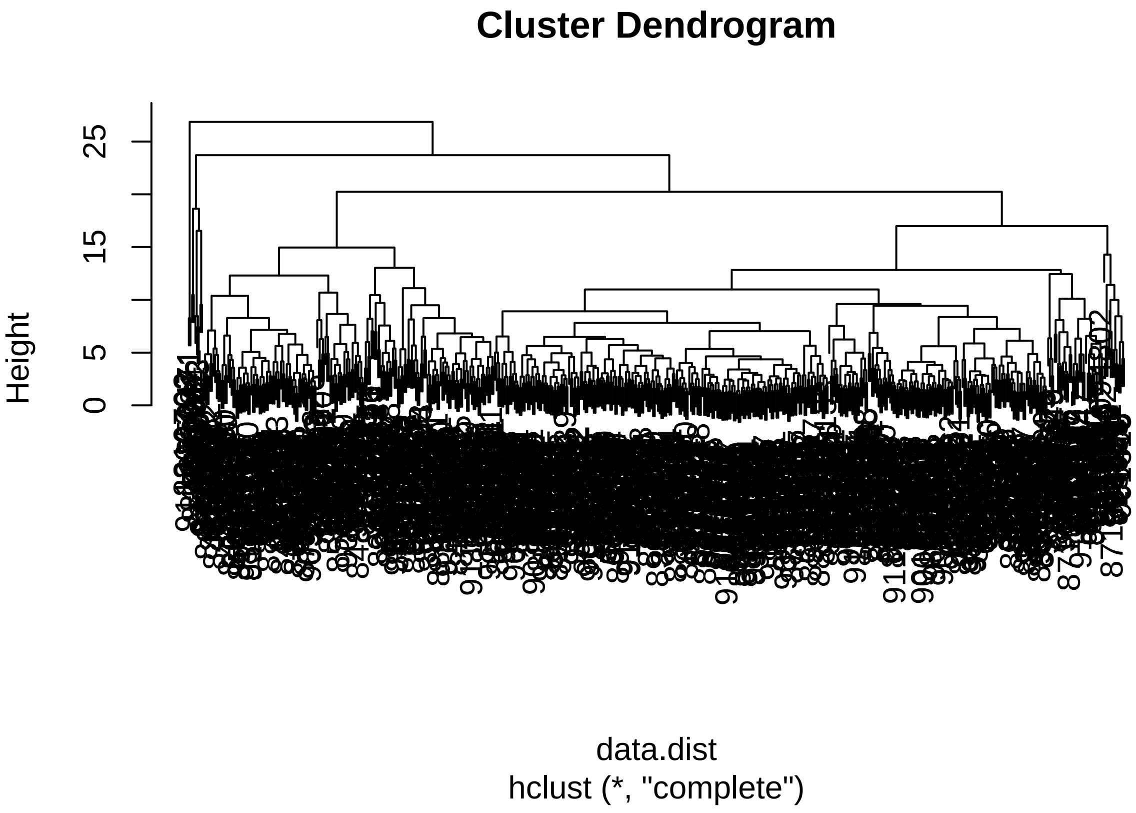
> Q12. Can you find a better cluster vs diagnoses match by cutting into a different number of clusters between 2 and 10?

No, 4 clusters seems to be the best match.

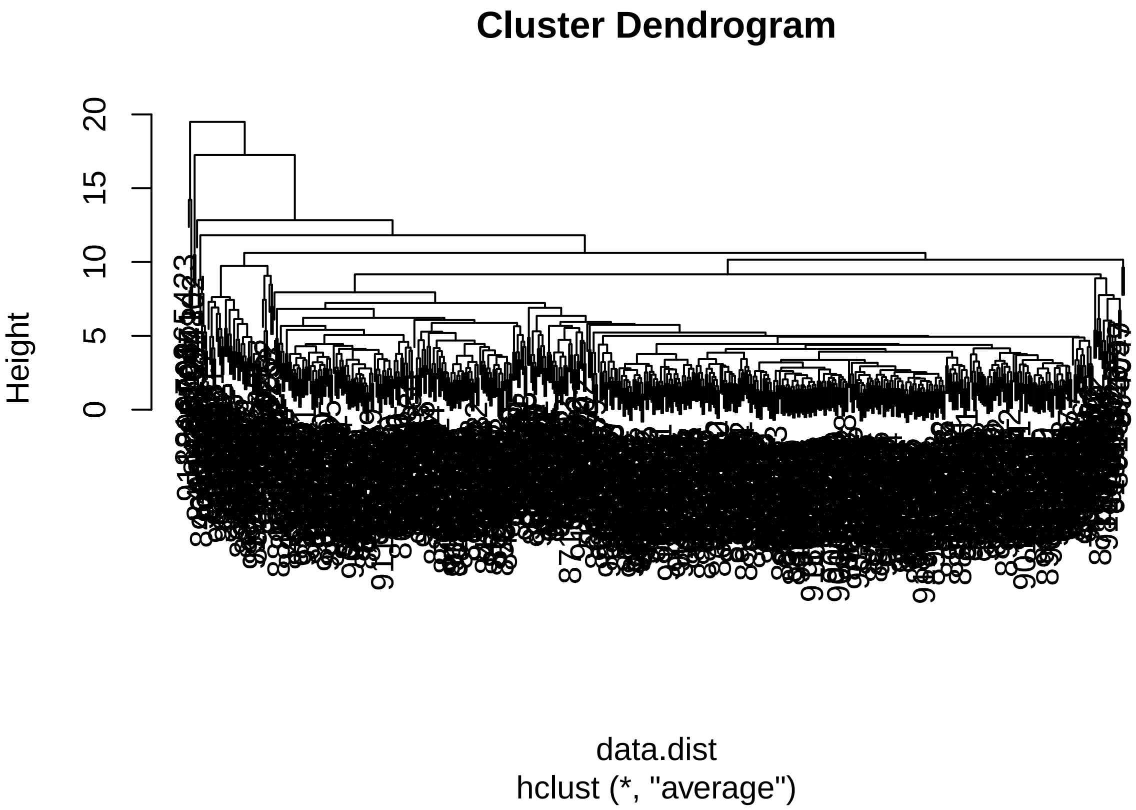
```
#Create plots using, single, complete, average, and ward.D2
plot(hclust(data.dist, method = "single"))
```



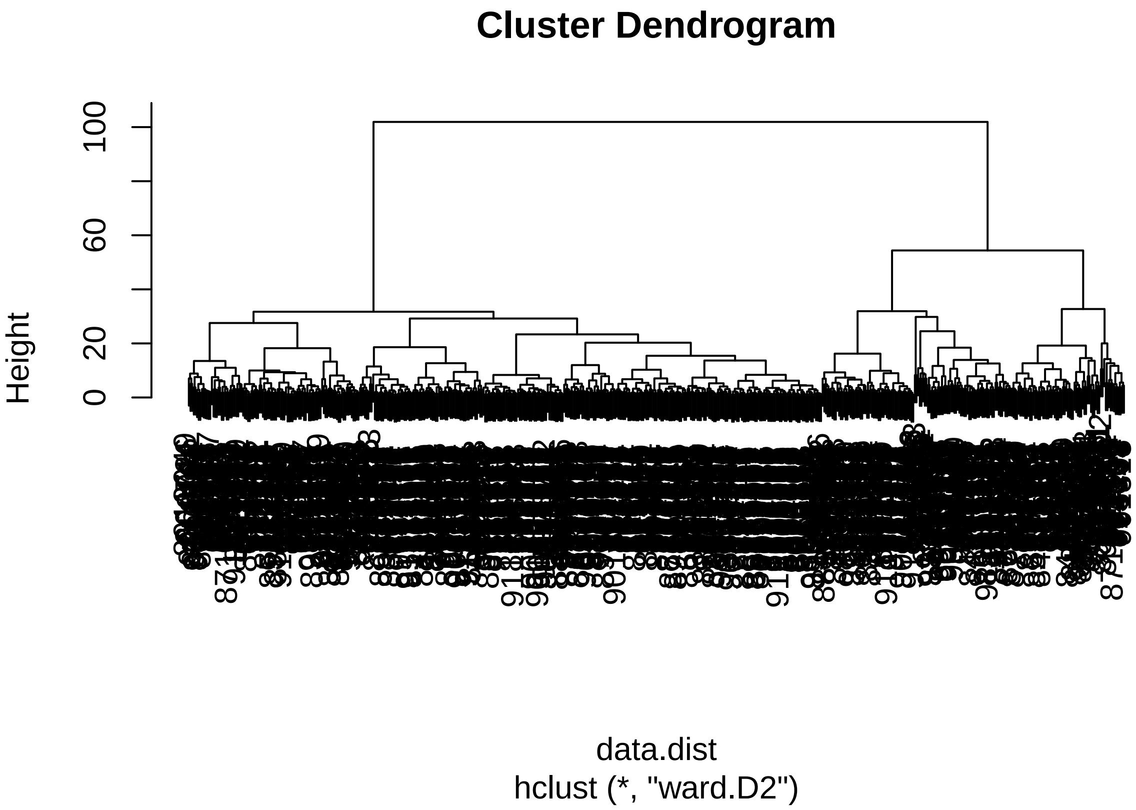
```
plot(hclust(data.dist, method = "complete"))
```



```
plot(hclust(data.dist, method = "average"))
```



```r
plot(hclust(data.dist, method = "ward.D2"))
```



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

Using the ward.D2 method seems to give the "cleanest" looking dendogram. The runner up is "complete."

K Means Clustering

```
wisc.km <-  kmeans(wisc.data, centers=2, nstart = 20)
```

```
table(wisc.km$cluster, diagnosis)
```

```
##    diagnosis
##       B   M
##   1   1 130
##   2 356  82
```

Q14. How well does k-means separate the two diagnoses? How does it compare to your hclust results?

The two diagnoses seem to have an obvious/significant split. It is overall similar to the hclust results when k=4.

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

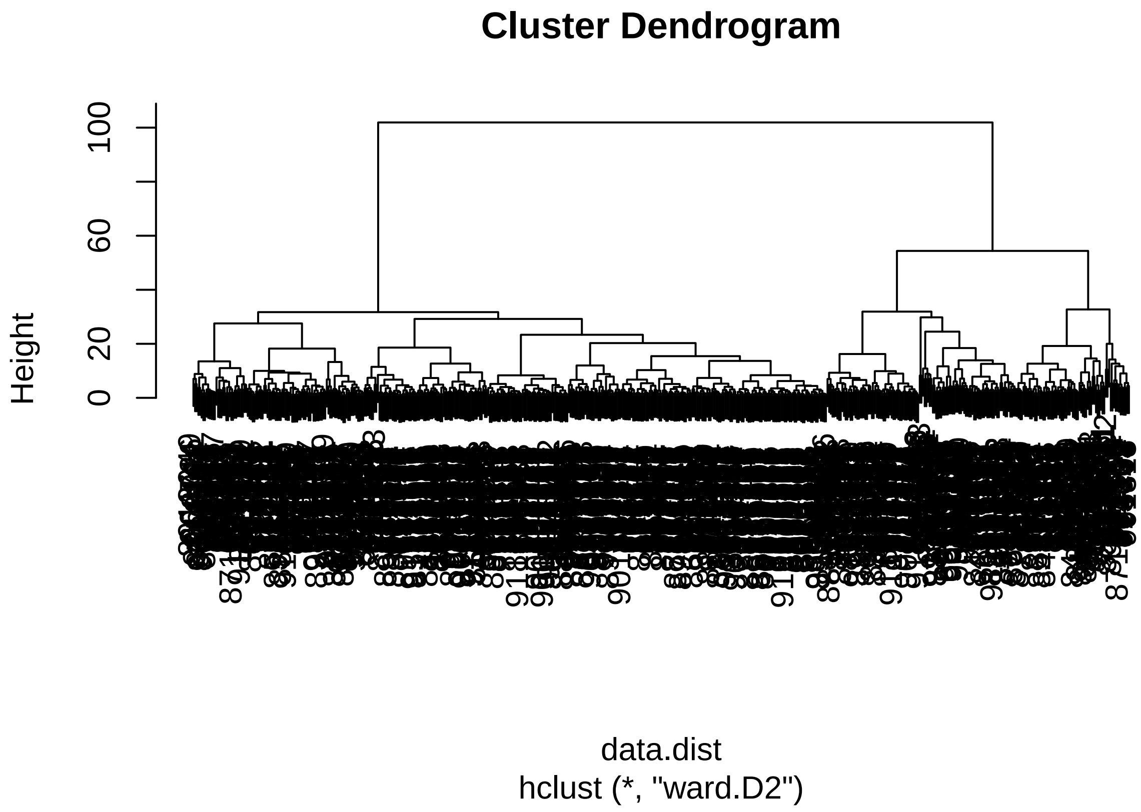
```
##
##       1   2   3   4
##   1 109   2  18   2
##   2  68   5 365   0
```

Combining Methods

Clustering on PCA Results

```
wisc.pr.hclust <- (hclust(data.dist, method = "ward.D2"))
```

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



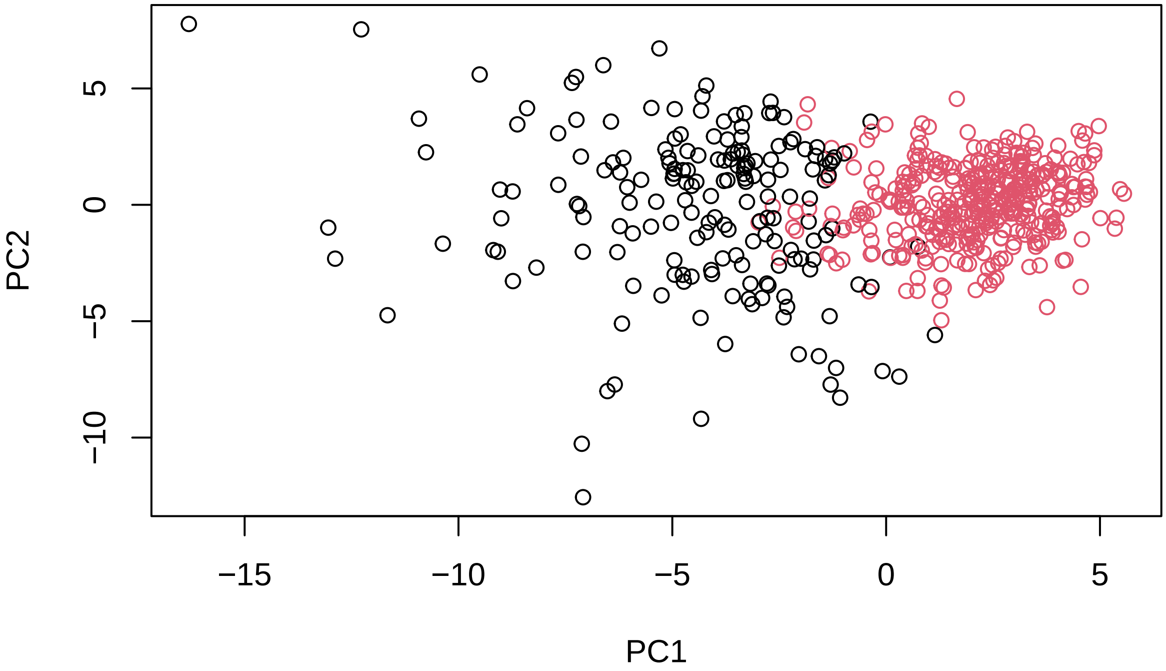
```
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
## grps
##   1   2
## 184 385
```

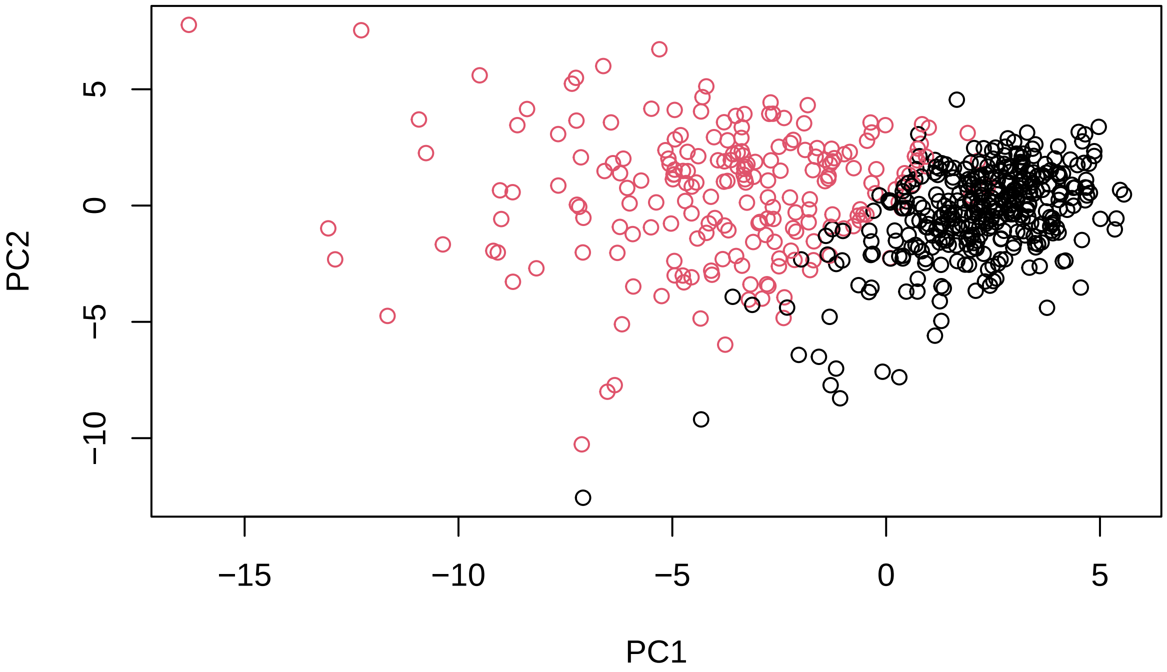
```
table(grps, diagnosis)
```

```
##     diagnosis
## grps   B   M
##    1  20 164
##    2 337  48
```

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



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



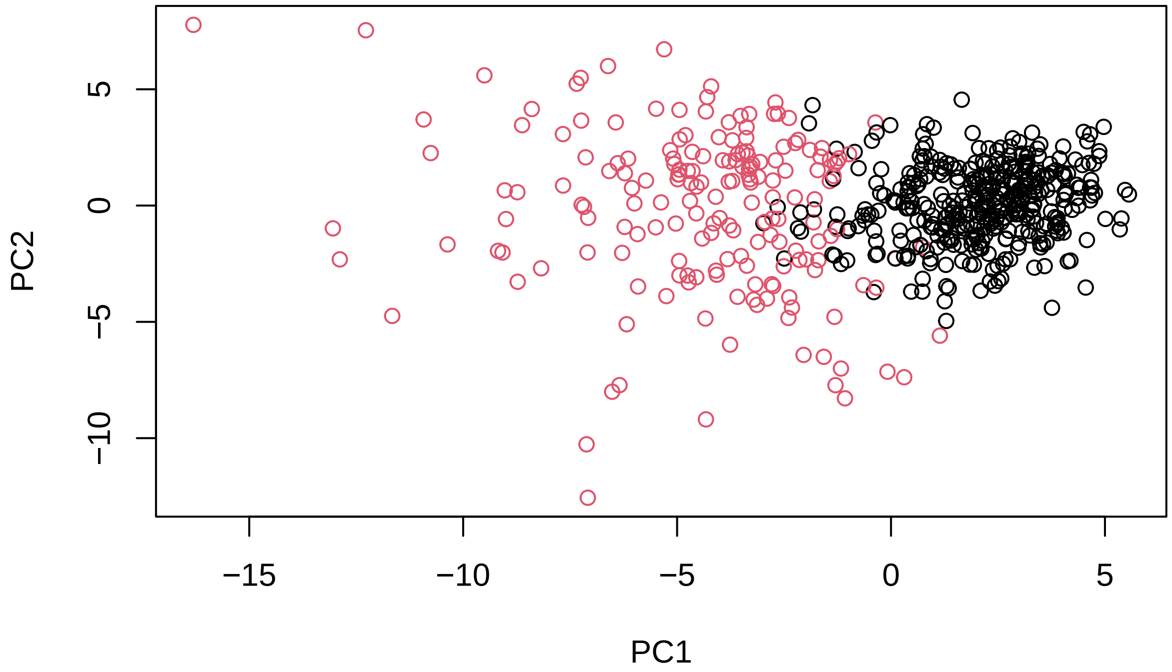
```
g <- as.factor(grps)
levels(g)
```

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

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

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

```
#plot w/ reordered factor
plot(wisc.pr$x[,1:2], col=g)
```



```
#clustering w/ 1st 7 PCs
wisc.pr.hclust.clusters <- cutree(wisc.pr.hclust, k=2)
```

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

```
##                        diagnosis
## wisc.pr.hclust.clusters   B   M
##                       1  20 164
##                       2 337  48
```

> Q15. How well does the newly created model with two clusters separate out the two diagnoses?

The two diagnoses seem to be separated significantly by using the new model with two clusters.

> 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.

The k-means and clustering models are roughly good at separating the diagnoses but they give ballpark results.

```
table(diagnosis)
```

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

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

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

```
table(wisc.km$cluster, diagnosis)
```

```
##    diagnosis
##       B   M
##   1   1 130
##   2 356  82
```

Sensitivity/Specificity

Sensitivity refers to a test's ability to correctly detect ill patients who do have the condition. In our example here the sensitivity is the total number of samples in the cluster identified as predominantly malignant (cancerous) divided by the total number of known malignant samples. In other words: TP/(TP+FN).

Specificity relates to a test's ability to correctly reject healthy patients without a condition. In our example specificity is the proportion of benign (not cancerous) samples in the cluster identified as predominantly benign that are known to be benign. In other words: TN/(TN+FN).

```
#hclust

#sensitivity
165/212
```

```
## [1] 0.7783019
```

```
#Specificity
343/357
```

```
## [1] 0.9607843
```

```
#k-means

#Sensitivity
130/212
```

```
## [1] 0.6132075
```

```
#Specificity
356/357
```

```
## [1] 0.9971989
```

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

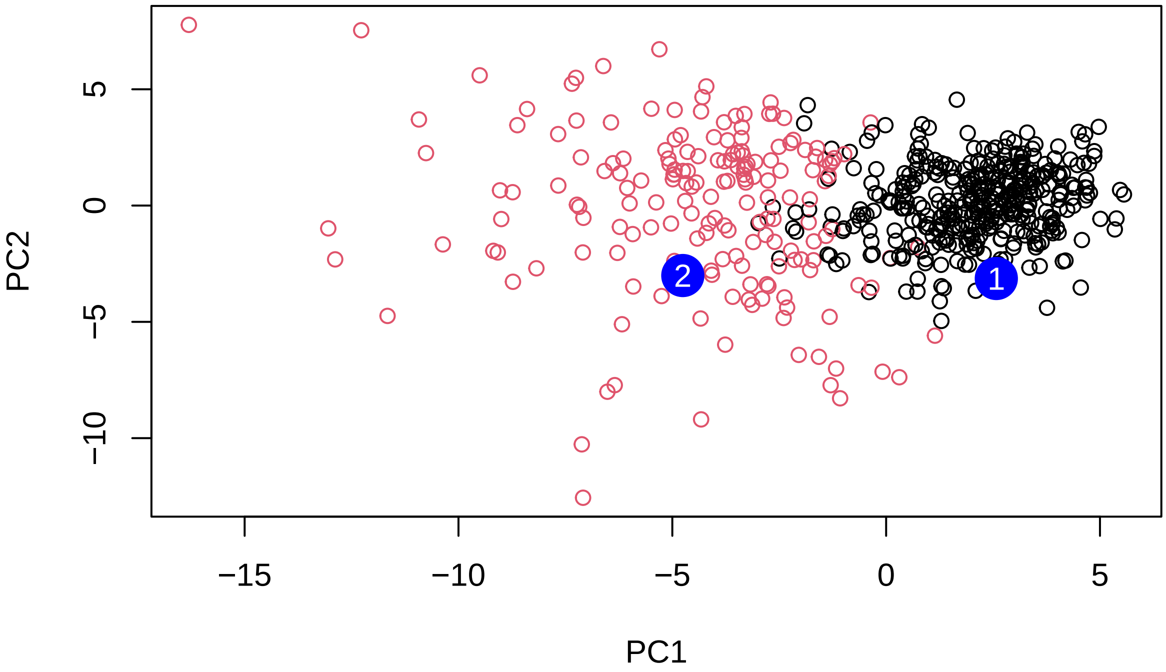
K-means gives the best specificity while hclustering gives the best sensitivity.

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,]  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 (located in the red/malignant cluster) should be prioritized for follow-up.