miniprojmark

2023-05-08

```r
# save csv file to project folder
fna.data <- "WisconsinCancer.csv"
#assign to a variable
wisc.df <- read.csv(fna.data, row.names = 1)
```

```r
# remove expert diagnosis from data aka first column
wisc.data <- wisc.df[,-1]
#save first column for later
diagnosis <- factor(wisc.df[,1])
```

```r
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 there? 569 observations

Q2. How many observations are malignant? 212 observations

```r
grep("M", diagnosis)
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  39  40
##  [37]  41  42  43  44  45  46  48  54  55  57  58  63  65  66  71  73  74  76
##  [55]  78  79  83  84  86  87  88  92  95  96 100 101 106 109 118 119 120 122
##  [73] 123 127 128 130 132 133 135 136 139 142 147 157 162 163 165 168 169 172
##  [91] 173 178 181 182 183 185 187 191 194 195 197 198 199 200 202 203 204 206
## [109] 208 211 213 214 215 216 219 220 224 230 231 234 237 238 240 245 251 253
## [127] 254 255 256 257 258 259 260 261 262 263 264 265 266 273 275 278 281 283
## [145] 284 298 301 303 318 322 324 329 330 331 336 338 340 344 352 353 354 366
## [163] 367 369 370 371 373 374 380 386 390 393 394 401 409 415 418 431 433 434
## [181] 436 442 445 447 450 452 461 462 469 480 488 490 493 499 500 502 504 510
## [199] 513 515 517 518 522 534 536 537 563 564 565 566 567 568
```

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

```
#check column means and standard dev for scaling purposes
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
```

```r
#perform pca on wisc.data
wisc.pr <- prcomp(wisc.data, scale=TRUE)
apply
```

```
## function (X, MARGIN, FUN, ..., simplify = TRUE)
## {
##     FUN <- match.fun(FUN)
##     simplify <- isTRUE(simplify)
##     dl <- length(dim(X))
##     if (!dl)
##         stop("dim(X) must have a positive length")
##     if (is.object(X))
##         X <- if (dl == 2L)
##             as.matrix(X)
##         else as.array(X)
##     d <- dim(X)
##     dn <- dimnames(X)
##     ds <- seq_len(dl)
##     if (is.character(MARGIN)) {
##         if (is.null(dnn <- names(dn)))
##             stop("'X' must have named dimnames")
##         MARGIN <- match(MARGIN, dnn)
##         if (anyNA(MARGIN))
##             stop("not all elements of 'MARGIN' are names of dimensions")
##     }
##     d.call <- d[-MARGIN]
##     d.ans <- d[MARGIN]
##     if (anyNA(d.call) || anyNA(d.ans))
##         stop("'MARGIN' does not match dim(X)")
##     s.call <- ds[-MARGIN]
##     s.ans <- ds[MARGIN]
##     dn.call <- dn[-MARGIN]
##     dn.ans <- dn[MARGIN]
##     d2 <- prod(d.ans)
##     if (d2 == 0L) {
##         newX <- array(vector(typeof(X), 1L), dim = c(prod(d.call),
##             1L))
##         ans <- forceAndCall(1, FUN, if (length(d.call) < 2L) newX[,
##             1] else array(newX[, 1L], d.call, dn.call), ...)
##         return(if (is.null(ans)) ans else if (length(d.ans) <
##             2L) ans[1L][-1L] else array(ans, d.ans, dn.ans))
##     }
##     newX <- aperm(X, c(s.call, s.ans))
##     dim(newX) <- c(prod(d.call), d2)
##     ans <- vector("list", d2)
##     if (length(d.call) < 2L) {
##         if (length(dn.call))
##             dimnames(newX) <- c(dn.call, list(NULL))
##         for (i in 1L:d2) {
##             tmp <- forceAndCall(1, FUN, newX[, i], ...)
##             if (!is.null(tmp))
##                 ans[[i]] <- tmp
##         }
##     }
##     else for (i in 1L:d2) {
##         tmp <- forceAndCall(1, FUN, array(newX[, i], d.call,
##             dn.call), ...)
##         if (!is.null(tmp))
##             ans[[i]] <- tmp
##     }
##     ans.list <- !simplify || is.recursive(ans[[1L]])
##     l.ans <- length(ans[[1L]])
##     ans.names <- names(ans[[1L]])
##     if (!ans.list)
##         ans.list <- any(lengths(ans) != l.ans)
##     if (!ans.list && length(ans.names)) {
##         all.same <- vapply(ans, function(x) identical(names(x),
##             ans.names), NA)
##         if (!all(all.same))
##             ans.names <- NULL
##     }
##     len.a <- if (ans.list)
##         d2
##     else length(ans <- unlist(ans, recursive = FALSE))
##     if (length(MARGIN) == 1L && len.a == d2) {
##         names(ans) <- if (length(dn.ans[[1L]]))
##             dn.ans[[1L]]
##         ans
##     }
##     else if (len.a == d2)
##         array(ans, d.ans, dn.ans)
##     else if (len.a && len.a%%d2 == 0L) {
##         if (is.null(dn.ans))
##             dn.ans <- vector(mode = "list", length(d.ans))
##         dn1 <- list(ans.names)
##         if (length(dn.call) && !is.null(n1 <- names(dn <- dn.call[1])) &&
##             nzchar(n1) && length(ans.names) == length(dn[[1]]))
##             names(dn1) <- n1
##         dn.ans <- c(dn1, dn.ans)
##         array(ans, c(len.a%/%d2, d.ans), if (!is.null(names(dn.ans)) ||
##             !all(vapply(dn.ans, is.null, NA)))
##             dn.ans)
##     }
##     else ans
## }
## <bytecode: 0x0000014b50a72dd8>
## <environment: namespace:base>
```

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

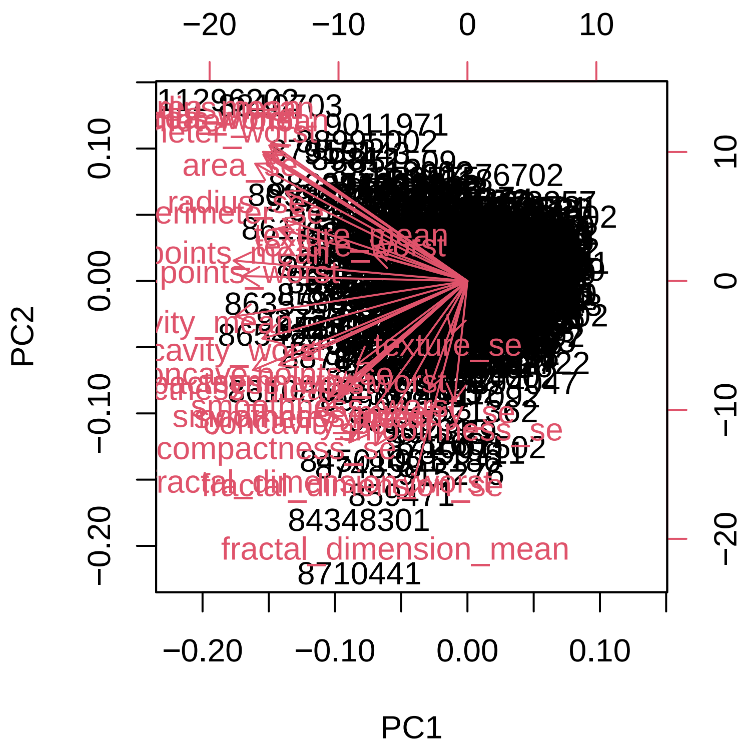
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172
## Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251
## Cumulative Proportion  0.4427 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. What proportion of the original variance is captured by PC1? 0.4427 or 44.27%

Q5. How many PCs required to describe atleast 70% of the original variance? 3 PCs are required.

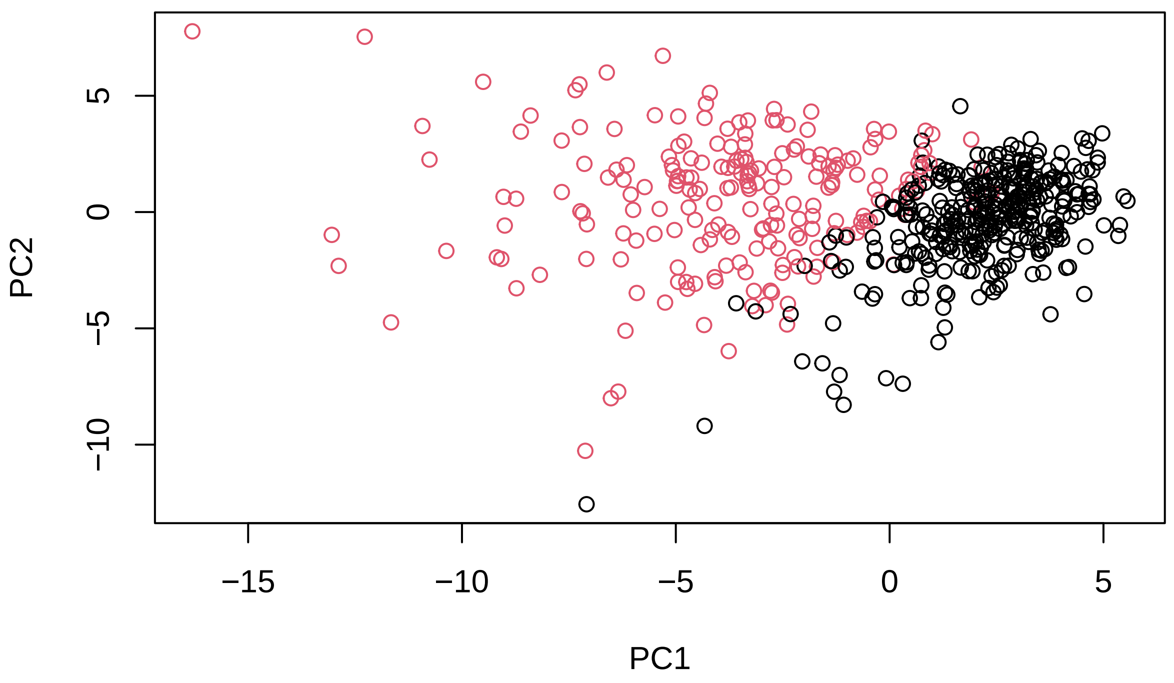
Q6. How many PCs are required to describe atleast 90% of the original variance? 7 PCs are required.

```
#visualizing the PCA model
biplot(wisc.pr)
```

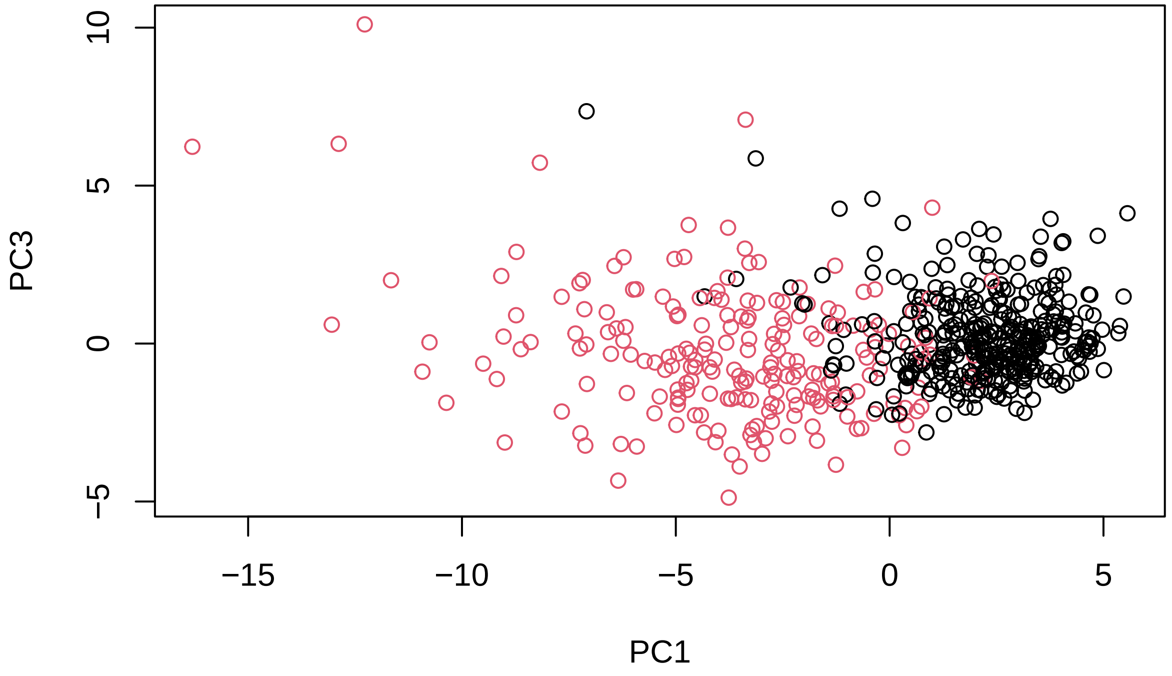


Q7. What stands out of this plot? I can't make much of this plot. It is very chaotic and full of overlapping data.

```
#scatter plot observations by component 1 and 2
plot(wisc.pr$x[,1:2], col = diagnosis ,
     xlab = "PC1", ylab = "PC2")
```



```
#scatter plot observations by component 1 and 3
plot(wisc.pr$x[,1-3], col = diagnosis ,
     xlab = "PC1", ylab = "PC3")
```

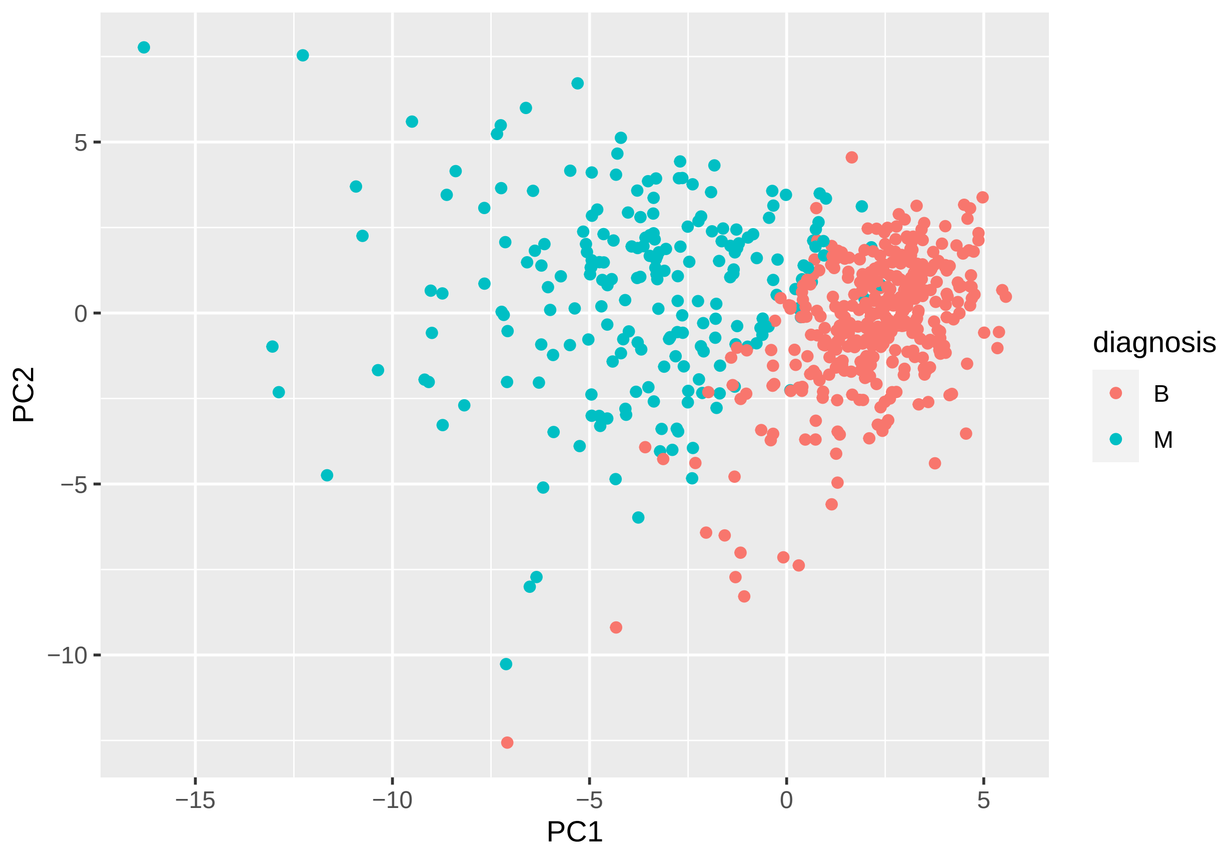


Q8. What do you notice about the plot for PC1/3? There seems to be a slightly clearler distinction between red and black as well as closer grouping.

```
# 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()
```

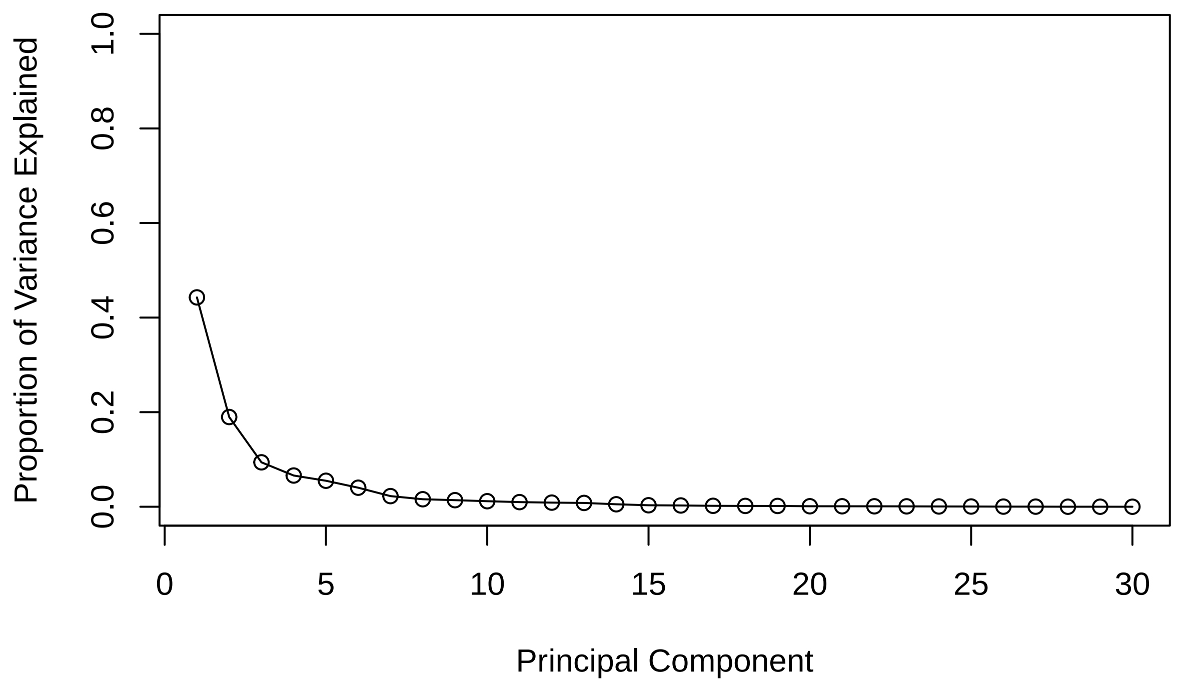


```
#calculate variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

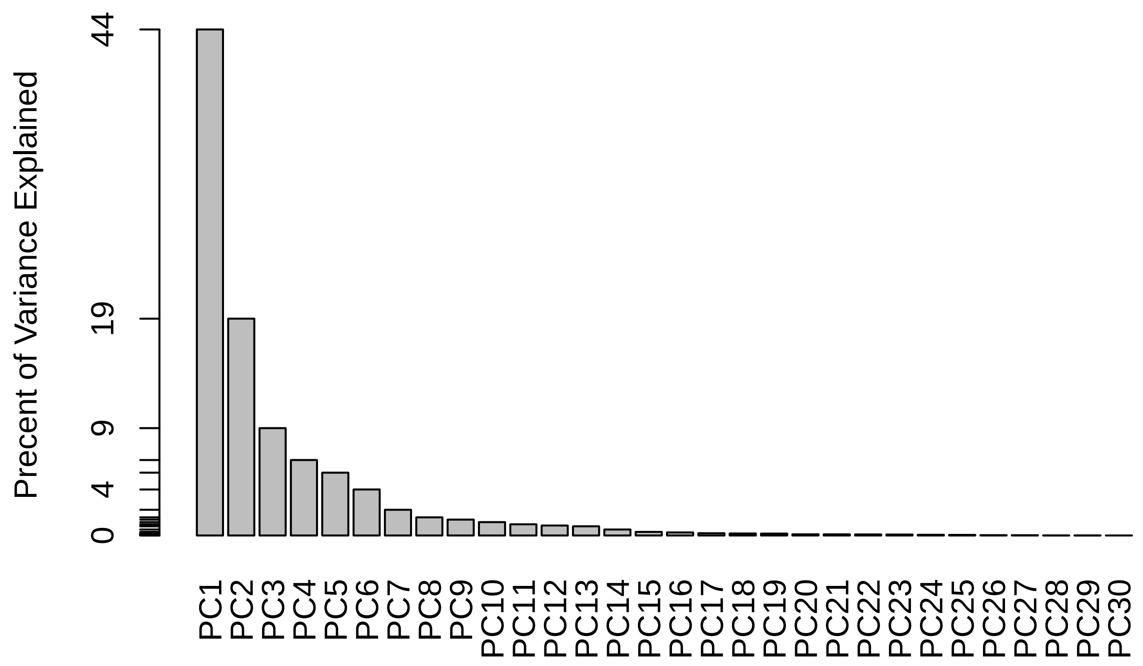
```
## [1] 13.281608  5.691355  2.817949  1.980640  1.648731  1.207357
```

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

# plot variance
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 )
```

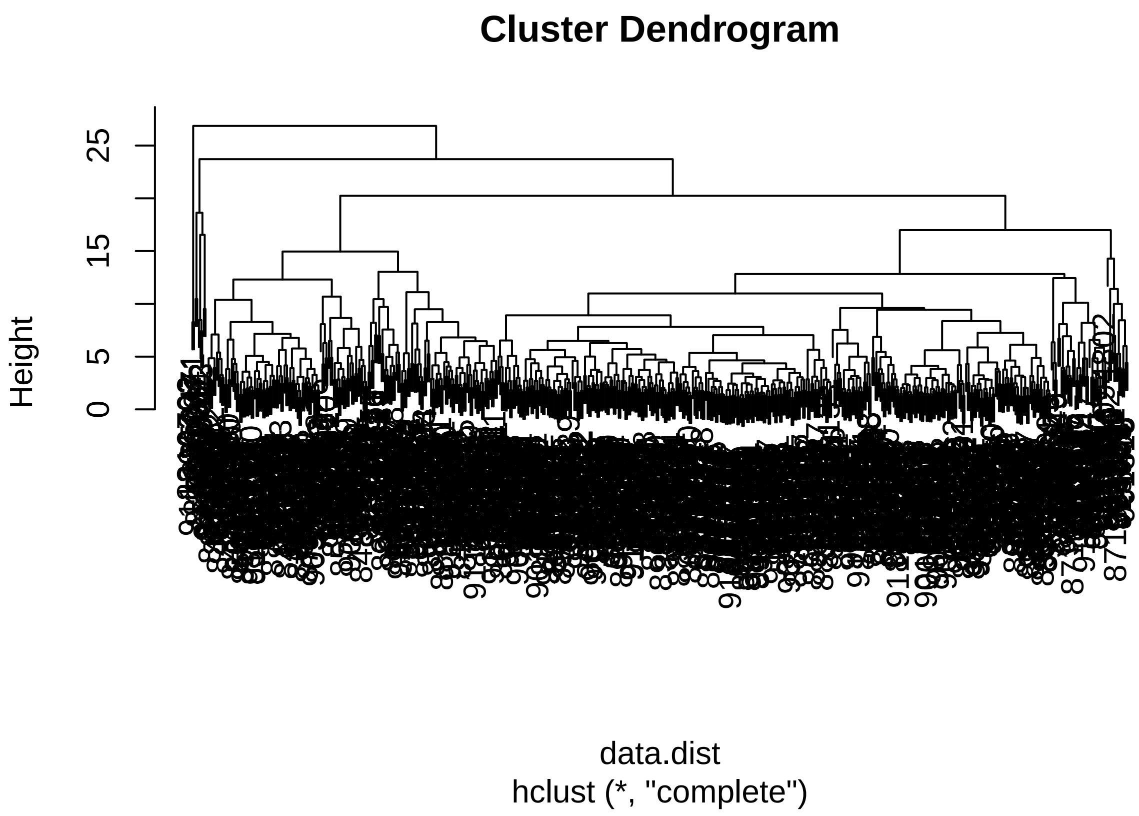


Q9. What is the component of the loading vector for concave.points_mean for PC1? -0.26085376

Q10. Minimum number of PCs required to explain 80% of variance? 5 PCs

```
#scale wisc.data
data.scaled <- scale(wisc.data)
#calculate euclidean distance between all pairs of observations in scaled data
data.dist <- dist(data.scaled)
#create a heirarchal clustering model using complete linkage
wisc.hclust <- hclust(data.dist, method = "complete")
```

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



Q11. At what height is there 4 clusters? 20 (a horizontal line crosses 4 time at height 20)

```r
#cutting so only 4 clusters
wisc.hclust.clusters <- cutree(wisc.hclust, k = 4)

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?

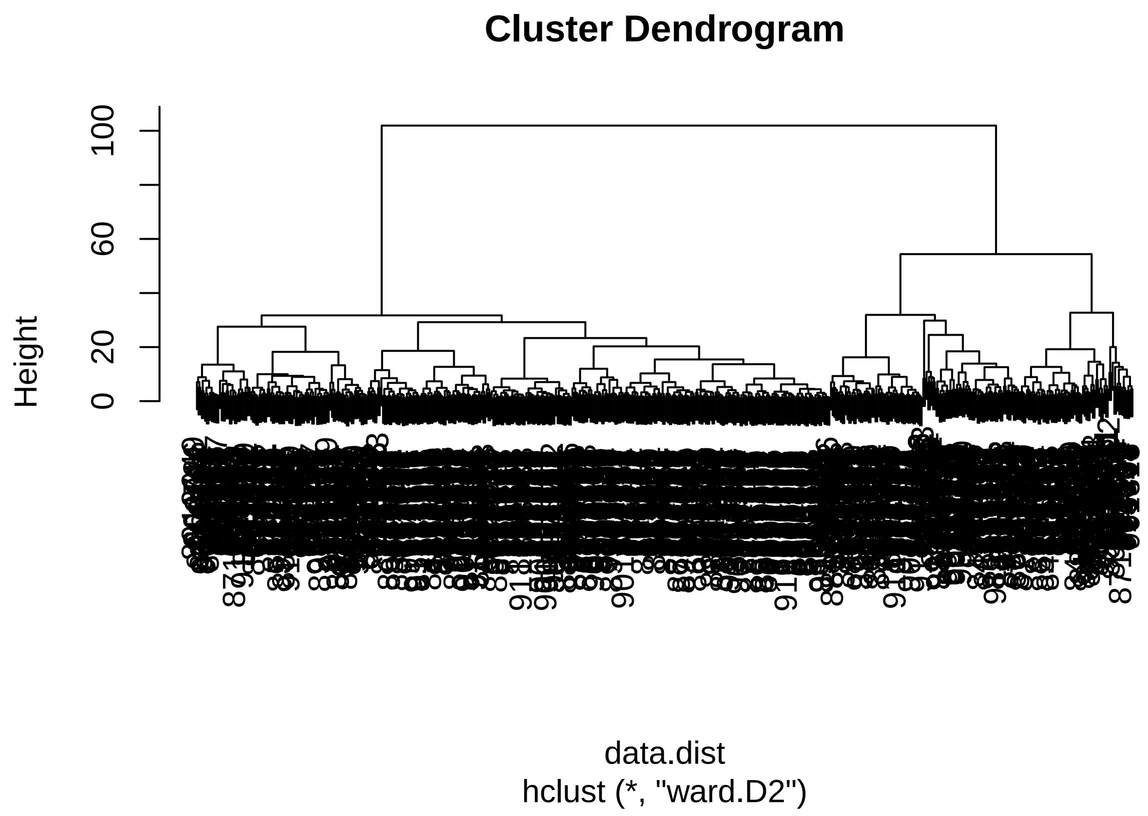
```r
#cutting so only 4 clusters
wisc.hclust.clusterz <- cutree(wisc.hclust, k = 2)

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

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

Q12. Can you find a better cluster v diagnosis match by changing the # of clusters? Observing 2 clusters seems like a good match to observe. Less groups give us a clearer image of benign vs malignant.

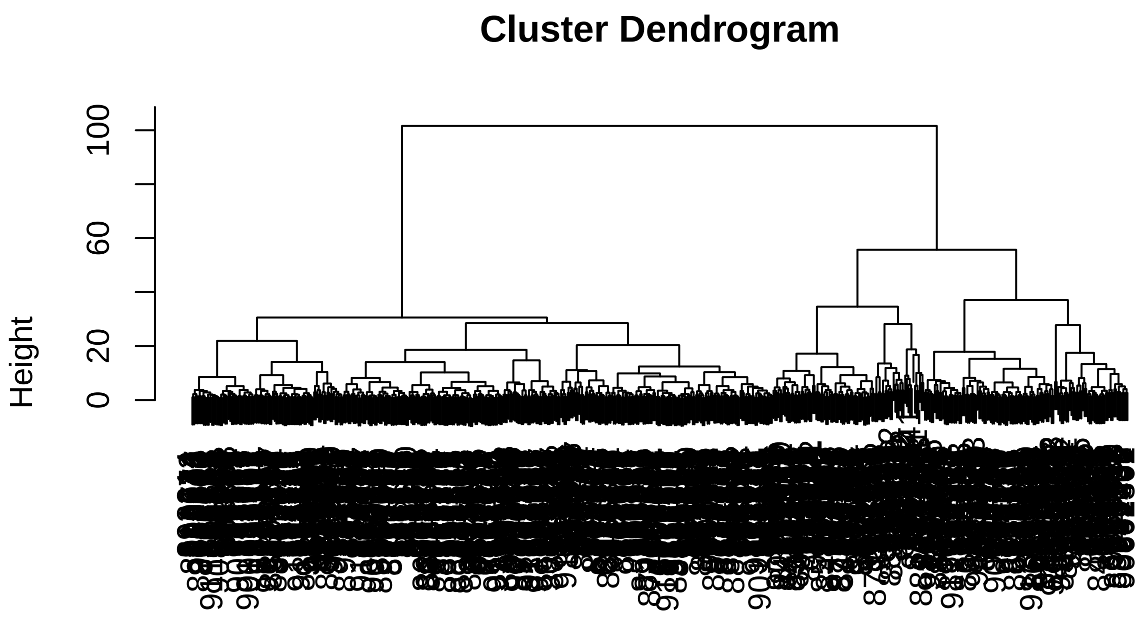
```r
wisc.hclustzz <- hclust(data.dist, method = "ward.D2")
plot(wisc.hclustzz)
```



Q13. Favorite method? I think I like ward.D2 as well. Minimizing the variance creates such a clean and aesthetically pleasing picture to observe and understand.

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

plot(wisc.pr.hclust)
```



dist(wisc.pr$x[, 1:7])
hclust (*, "ward.D2")

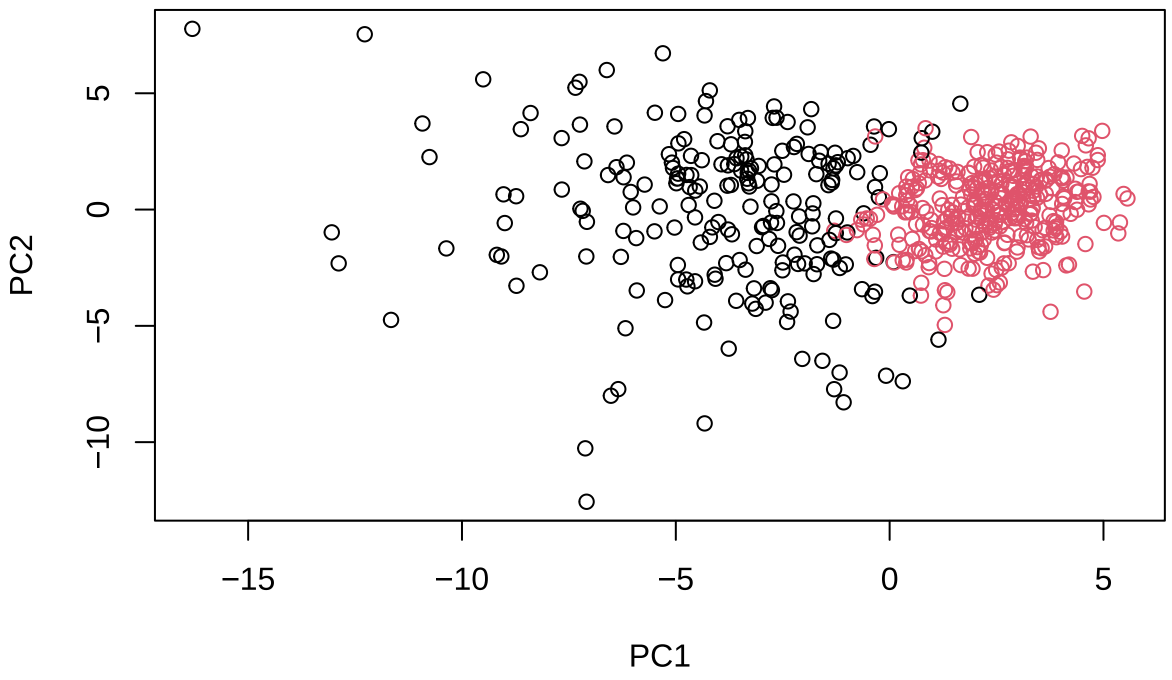
```
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
## grps
##   1   2
## 216 353
```

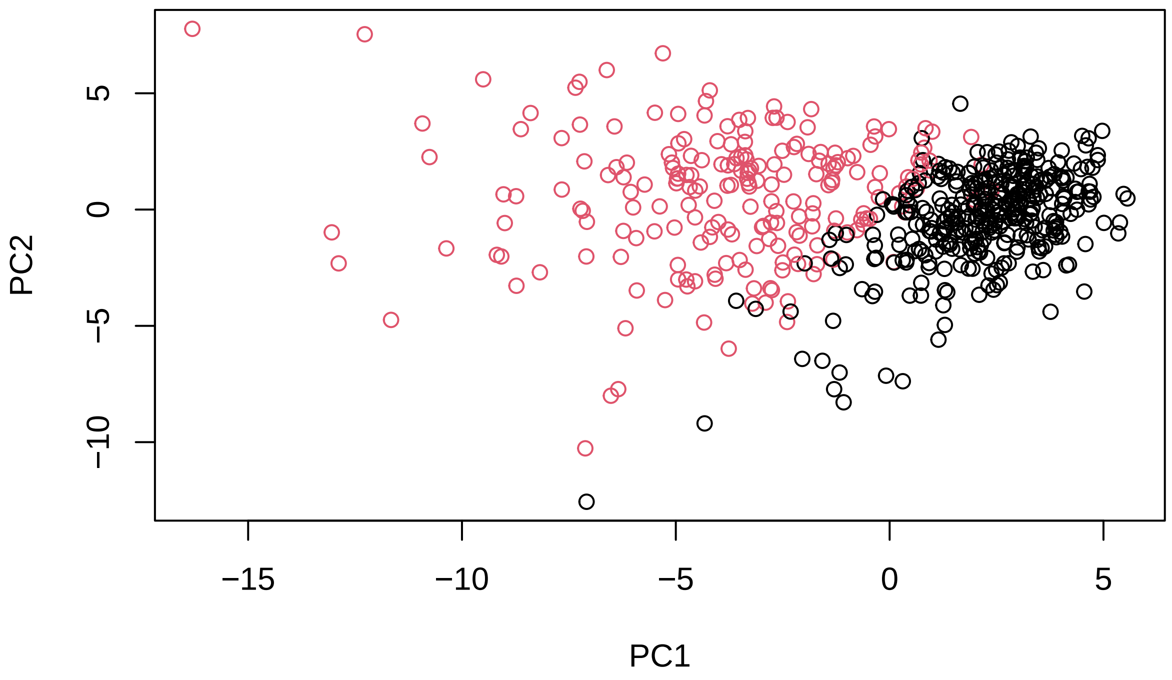
```
table(grps, diagnosis)
```

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

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

```r
#compare diagnosis
table(wisc.pr.hclust.clusters, diagnosis)
```

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

Q15. How well does the newly created model with four clusters seperate out the two diagnosis? I think it works really well. We are able to get a clearer picture of the diagnosis data and they are grouped together efficiently.

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

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

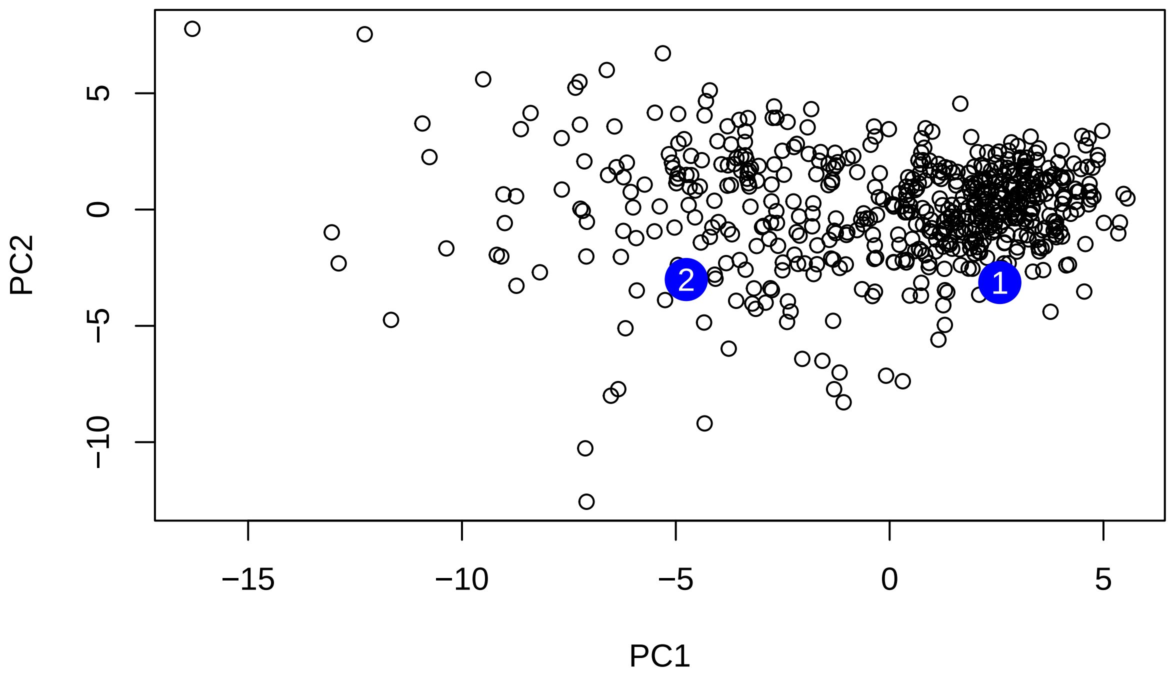
Q16. How do the other models do in terms of serperating diagnosis? H-cluster model does not work as well in seperating diagnoses. The model we just made is more clear, has less groups to interpret. It is good to note that the models had similar outputs in terms of data.

Q17. Which analysis procedures resulted in the best specificity? Best sensitivity? I think the ward.D2 procedure had best specificity and sensitivity.

```
#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])
points(npc[,1], npc[,2], col="blue", pch=16, cex=3)
text(npc[,1], npc[,2], c(1,2), col="white")
```



Q18. Which of the new patients should be prioritized for follow up based on results? Patient 2 needs to be followed up on for their results.