test2

Cuiling He

2023-02-03

In today's class we will ecplore a complete analysis using the unsupervised leanring techiques covered in the last class. We will combine PCA as a preprocessing step to clustering using data that consist of measurements of cell nuclei of human breast masses.

Import the dataset

Before we can begin our analysis we first have to download and import our data correctly into our R session.

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

Q How many smaples are in this dataset?

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

```
## [1] 569
```

Q. How many variables (columns)

```
ncol(wisc.df)
```

```
## [1] 31
```

Q How many M and B samples are there?

```
wisc.df$diagnosis ==  "M"
```

```
##   [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE
##  [25]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [37]  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
##  [49] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE  TRUE FALSE FALSE
##  [61] FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE
##  [73]  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE
```

```
##  [85] FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE
##  [97] FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [109]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE
## [121] FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE
## [133]  TRUE FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE
## [145] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157]  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE FALSE FALSE  TRUE
## [169]  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [181]  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE
## [193] FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE
## [205] FALSE  TRUE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
## [217] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [229] FALSE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE
## [241] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
## [253]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [265]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE
## [277] FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [301]  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [313] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE
## [325] FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE
## [337] FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [349] FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [361] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE FALSE
## [373]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE
## [385] FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE FALSE
## [397] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [409]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE
## [421] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
## [433]  TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [445]  TRUE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE
## [457] FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [469]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
## [481] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE
## [493]  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE
## [505] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE FALSE
## [517]  TRUE  TRUE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [529] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE FALSE
## [541] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [553] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
## [565]  TRUE  TRUE  TRUE  TRUE FALSE
```

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

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

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

```
length(grep("_mean", colnames(wisc.df)))
```

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

Q What features are “mean” values (or have “mean” in their name)?

```
grep("_mean", colnames(wisc.df), value = TRUE)
```

```
##  [1] "radius_mean"            "texture_mean"           "perimeter_mean"
##  [4] "area_mean"              "smoothness_mean"        "compactness_mean"
##  [7] "concavity_mean"         "concave.points_mean"    "symmetry_mean"
## [10] "fractal_dimension_mean"
```

I need to remove the first diagnosis column form my data before doing any analysis. I will store it for later as a factor.

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

2. Principal Component Analysis

the main pca function in base r is called `prcomp()`

Before doing anything like PCA, It is important to check if the data need to be scaled before performing PCA. Recall two common reasons for scaling data include:

-The input variables use different units of measurement. -The input variables have significantly different variances.

Looks like we need to scale by seting `scale = TRUE` in our `promp()` function call.

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

```
##             radius_mean            texture_mean          perimeter_mean
##                    3.52                    4.30                   24.30
##               area_mean         smoothness_mean        compactness_mean
##                  351.91                    0.01                    0.05
##          concavity_mean     concave.points_mean           symmetry_mean
##                    0.08                    0.04                    0.03
##  fractal_dimension_mean               radius_se              texture_se
##                    0.01                    0.28                    0.55
##            perimeter_se                 area_se           smoothness_se
##                    2.02                   45.49                    0.00
##          compactness_se            concavity_se       concave.points_se
##                    0.02                    0.03                    0.01
##             symmetry_se    fractal_dimension_se             radius_worst
##                    0.01                    0.00                    4.83
##           texture_worst         perimeter_worst               area_worst
##                    6.15                   33.60                  569.36
##        smoothness_worst       compactness_worst         concavity_worst
##                    0.02                    0.16                    0.21
##    concave.points_worst          symmetry_worst fractal_dimension_worst
##                    0.07                    0.06                    0.02
```

Time for PCA

```
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
```

```
attributes(wisc.pr)
```

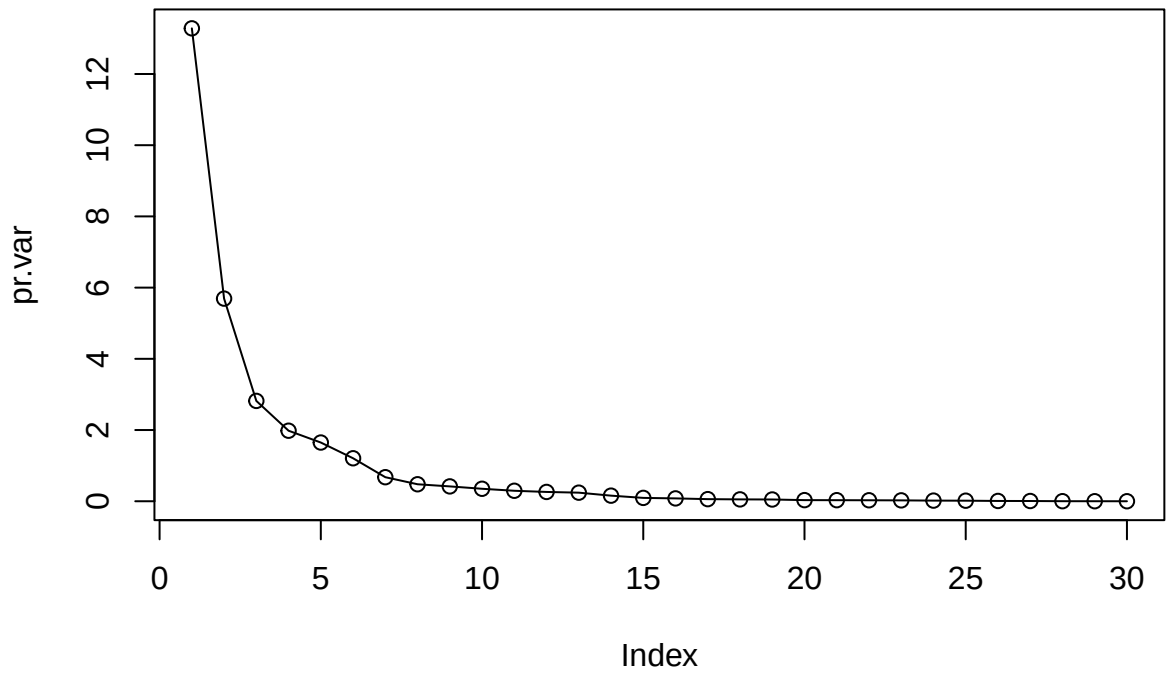
```
## $names
## [1] "sdev"     "rotation" "center"   "scale"    "x"
##
## $class
## [1] "prcomp"
```

scree plot

```
pr.var <- wisc.pr$sdev^2
#proportion of variance
pr.var/sum(pr.var)
```

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

```
plot(pr.var, typ = "o")
```



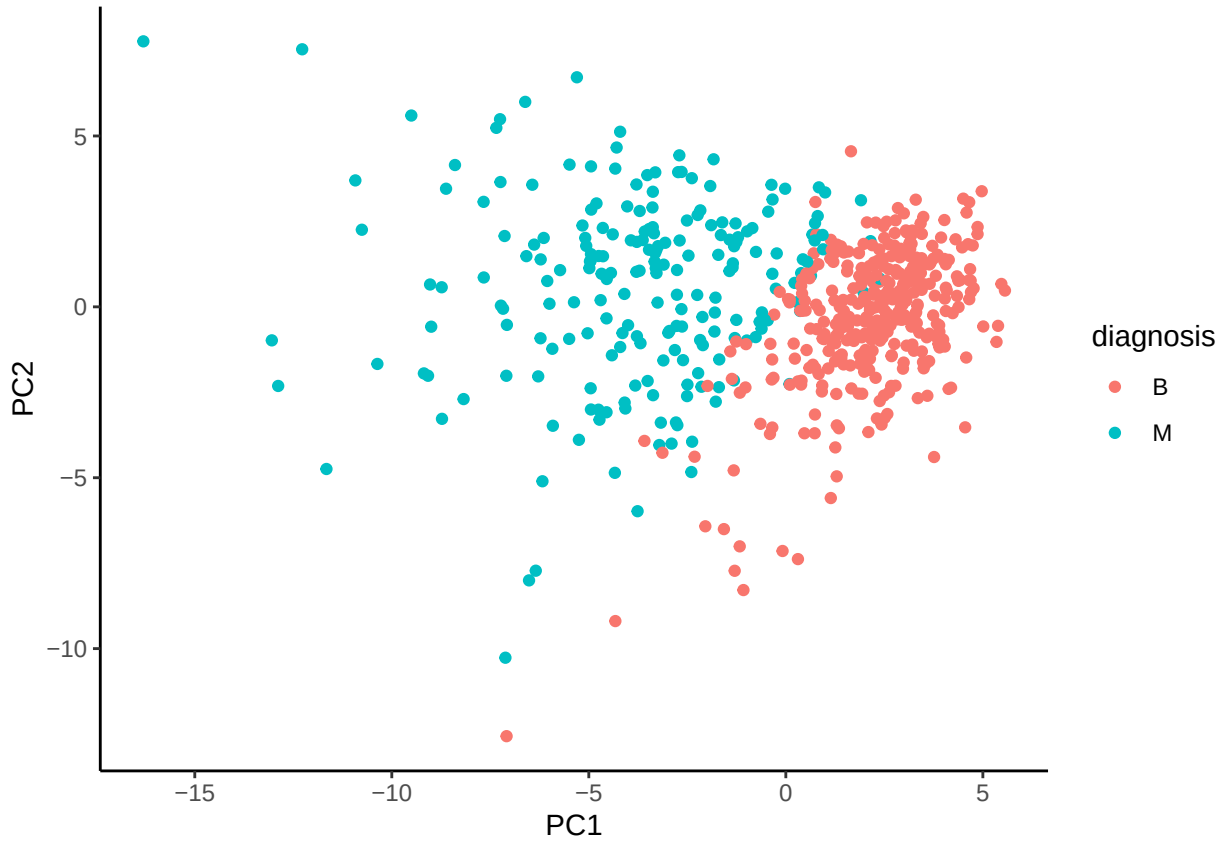
lets make our manin results figure from our PCA our score plot (a.k.a "PC plot"), "PC1 vs PC2 plot", etc.)

```
install.packages("ggplot2")
```

```
## Installing ggplot2 [3.4.0] ...
##  OK [linked cache]
```

```
library(ggplot2)
```

```
pc <- as.data.frame(wisc.pr$x)
ggplot(pc) +
  aes(PC1, PC2, col = diagnosis) +
  geom_point() +
  theme_classic()
```



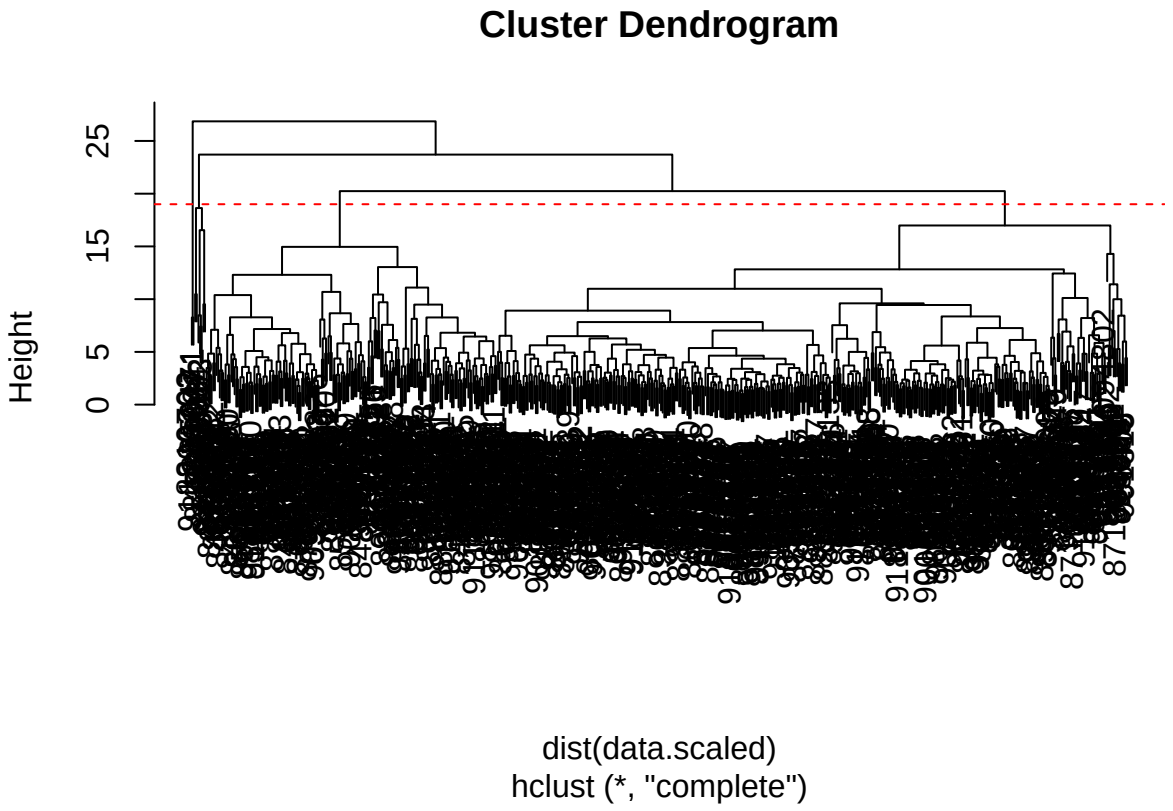
3. Hierarchical clustering

Now have a look at the tree

```
data.scaled <- scale(wisc.data)
head(apply(data.scaled, 2, sd))
```

```
##      radius_mean     texture_mean   perimeter_mean        area_mean
##                1                1                1                1
##  smoothness_mean compactness_mean
##                1                1
```

```
wisc.hclust <- hclust(dist(data.scaled))
plot(wisc.hclust)
abline(h=19, col = "red", lty = 2)
```



to get a cluster membership vector i will use the `cutree()` function and "cut into 4 or so grps or clusters.

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

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

I can also use the `table()` to cross tabulate

```
table(diagnosis)
```

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

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

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

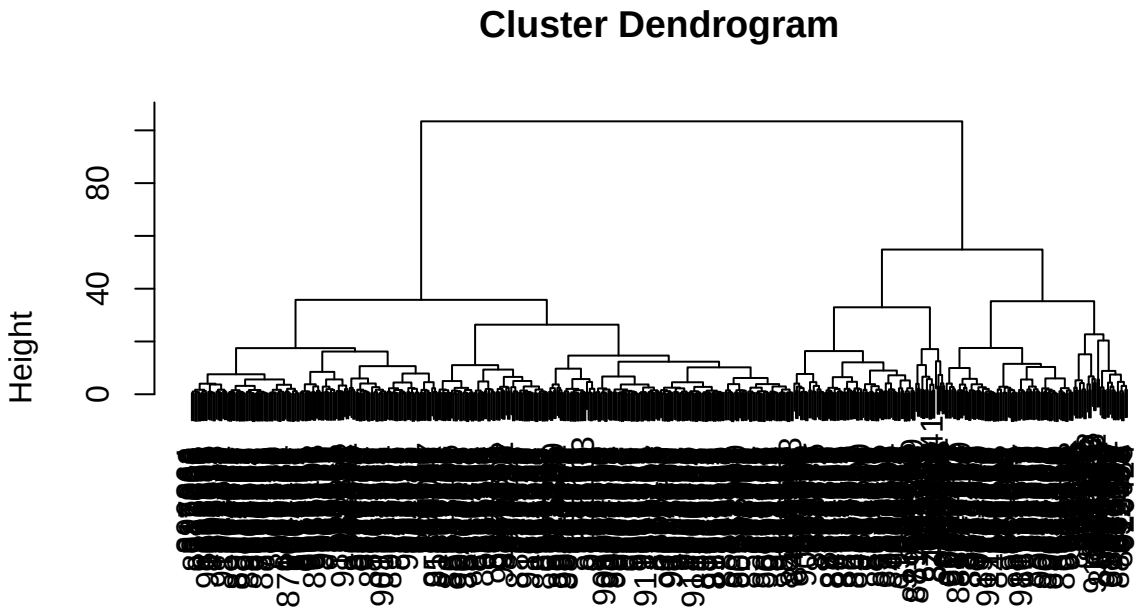
Clustering on PCA results

I can cluster in PC-space and use as many or as few PCs as I want. To start with I will use 3 PCs, that is I will cluster along PC1, PC2, and PC3.

```
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
```

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



This looks much nicer than our previous clustering result. Let's find the two major clusters with the `cutree()` function.

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

```
## grps
##   1   2
## 203 366
```

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

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

we could calculate accuracy - the proportion of smaples we got correct if we take cluster 1 to represent all M and cluster 2 to represent all B.

```
(179+333)/nrow(wisc.data)
```

```
## [1] 0.8998243
```

```
#url <- "new_samples.csv"
url <- "https://tinyurl.com/new-samples-CSV"
read.csv(url)
```

```
##   radius_mean texture_mean perimeter_mean area_mean smoothness_mean
## 1       8.598        20.98          54.66     221.8          0.1243
## 2      14.250        22.15          96.42     645.7          0.1049
##   compactness_mean concavity_mean concave.points_mean symmetry_mean
## 1          0.08963         0.0300            0.009259        0.1828
## 2          0.20080         0.2135            0.086530        0.1949
##   fractal_dimension_mean radius_se texture_se perimeter_se area_se
## 1                0.06757    0.3582      2.067        2.493   18.39
## 2                0.07292    0.7036      1.268        5.373   60.78
##   smoothness_se compactness_se concavity_se concave.points_se symmetry_se
## 1      0.011930        0.03162      0.03000          0.009259     0.03357
## 2      0.009407        0.07056      0.06899          0.018480     0.01700
##   fractal_dimension_se radius_worst texture_worst perimeter_worst area_worst
## 1             0.003048        9.565         27.04           62.06      273.9
## 2             0.006113       17.670         29.51          119.10      959.5
##   smoothness_worst compactness_worst concavity_worst concave.points_worst
## 1           0.1639            0.1698         0.09001               0.02778
## 2           0.1640            0.6247         0.69220               0.17850
##   symmetry_worst fractal_dimension_worst
## 1         0.2972                 0.07712
## 2         0.2844                 0.11320
```

```
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
```

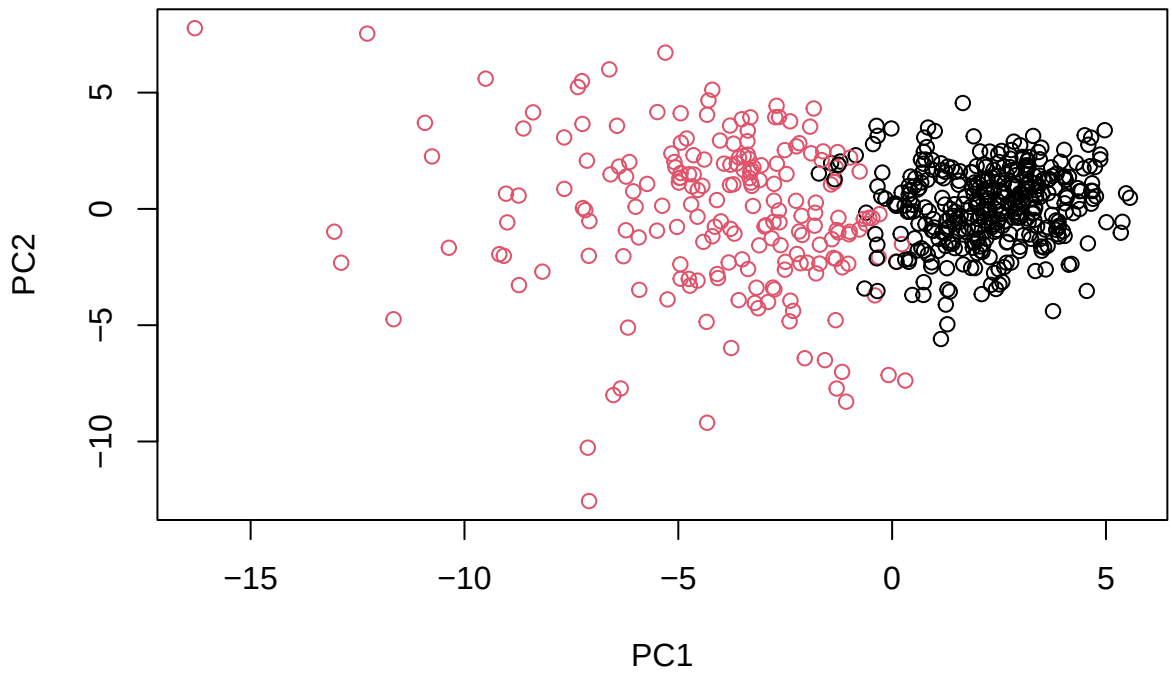
```
g <- as.factor(grps)
levels(g)
```

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

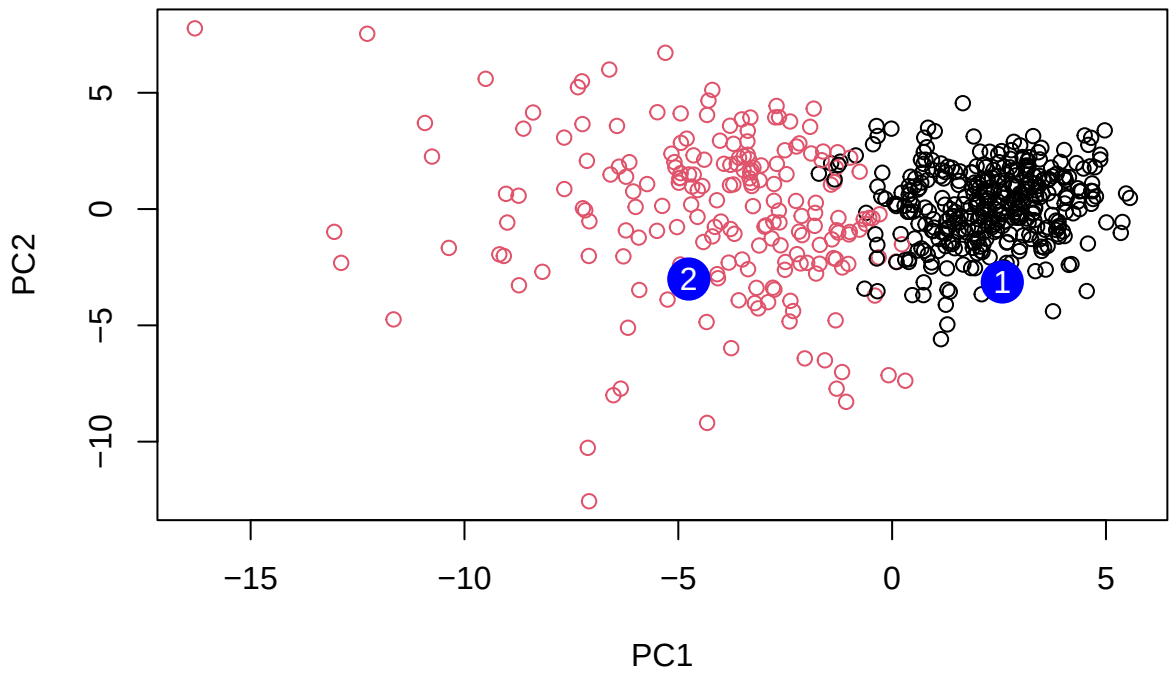
```
g <- relevel(g,2)
levels(g)
```

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

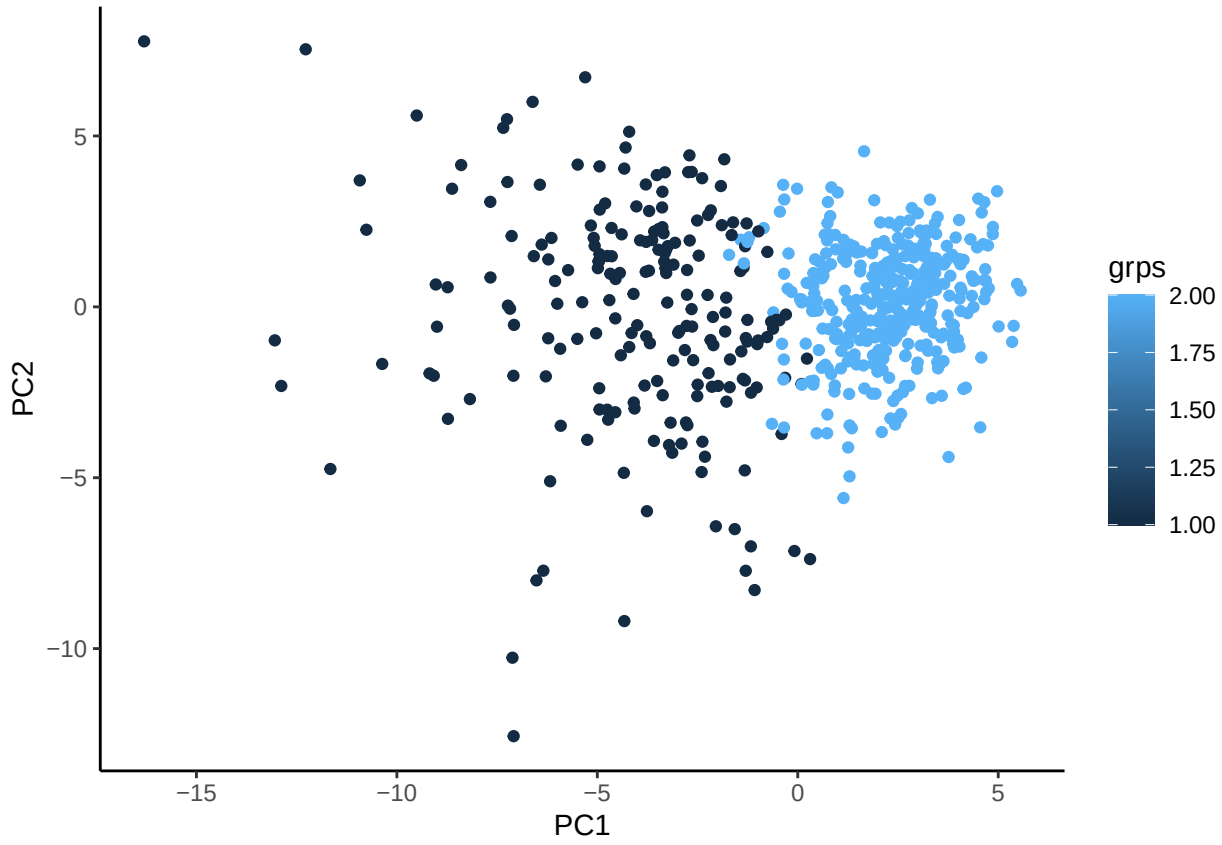
```
plot(wisc.pr$x[,1:2], col=g)
```




```
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")
```




```
pc <- as.data.frame(wisc.pr$x)
ggplot(pc) +
  aes(PC1, PC2, col = grps) +
  geom_point() +
  theme_classic()
```


PC2
5
0
−5
−10
−15
−10
−5
0
5
PC1
grps
2.00
1.75
1.50
1.25
1.00