class08_mini_project

A59013200 (Mina Wu)

Unsupervised LEarning Analysis of Human Breast Cancer Cells.

Our data

```
# Save your input data file into your Project directory
fna.data <- "WisconsinCancer.csv"

# Complete the following code to input the data and store as wisc.df
wisc.df <- read.csv(fna.data, row.names=1)
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
```

Q1. How many observations/samples/patients/rows?

There are 569 individuals in this dtaset.

Q2. What is in the '$diagnosis‘ column? How many of each type?

```
sum(wisc.df$diagnosis == "M")
```

```
[1] 212
```

```
  sum(wisc.df$diagnosis =="B")
```

```
[1] 357
```

Use table.

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

```
  B   M
357 212
```

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

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

```
[1] 10
```

Q. How many variables/dimensions we have?

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

```
[1] 31
```

Save the diagnosis for reference later

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

```
[1] M M M M M M
Levels: B M
```

and remove or exclude this (diagnosis) column from any of our analysis.

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

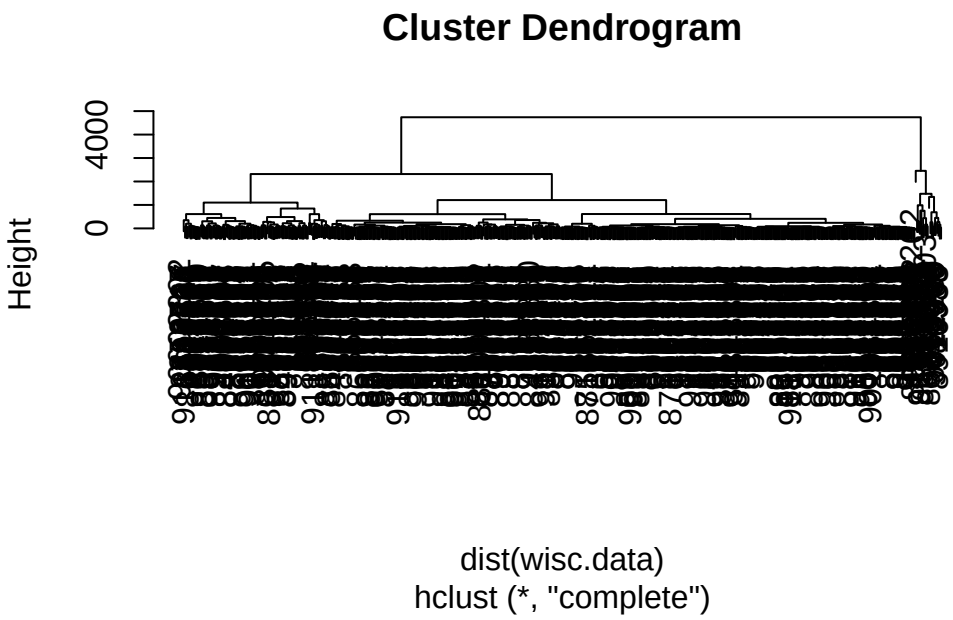
```
         radius_mean texture_mean perimeter_mean area_mean smoothness_mean
842302         17.99        10.38         122.80    1001.0         0.11840
842517         20.57        17.77         132.90    1326.0         0.08474
84300903       19.69        21.25         130.00    1203.0         0.10960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

Let's try clustering this data:

Hierarchical Clustering with `hclust()`

```
wisc.hc <- hclust(dist(wisc.data))
plot(wisc.hc)
```



Pricipal Component Analysis

Let's try PCA on this data. Before doing any analysis like weshould check if our input data needs to be called first?

side-note: example with mtcars

```
head(mtcars)
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
apply(mtcars,2,mean)
```

```
      mpg        cyl       disp         hp       drat         wt       qsec
20.090625   6.187500 230.721875 146.687500   3.596563   3.217250  17.848750
       vs         am       gear       carb
 0.437500   0.406250   3.687500   2.812500
```

```
apply(mtcars,2,sd)
```

```
      mpg        cyl       disp         hp       drat         wt
 6.0269481   1.7859216 123.9386938  68.5628685   0.5346787   0.9784574
     qsec         vs         am       gear       carb
 1.7869432   0.5040161   0.4989909   0.7378041   1.6152000
```

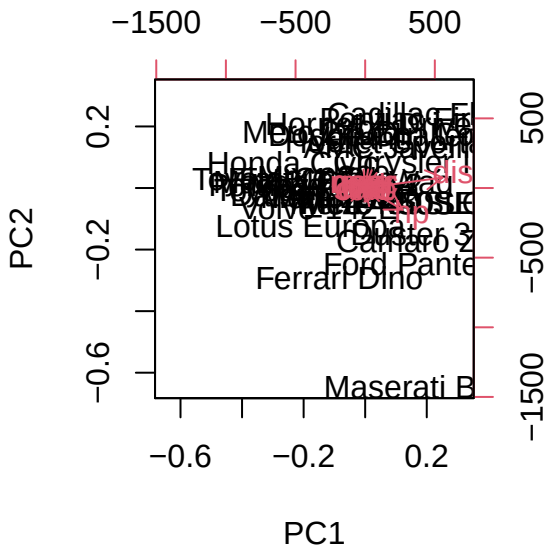
```
#the variance is high, so scaling it helps to bring the values together with less spread s
```

Let's try a PCA on this car dataset.

```
pc<- prcomp(mtcars)
summary(pc)
```

```
Importance of components:
                          PC1      PC2     PC3     PC4     PC5     PC6    PC7
Standard deviation     136.533 38.14808 3.07102 1.30665 0.90649 0.66354 0.3086
Proportion of Variance   0.927  0.07237 0.00047 0.00008 0.00004 0.00002 0.0000
Cumulative Proportion    0.927  0.99937 0.99984 0.99992 0.99996 0.99998 1.0000
                          PC8    PC9   PC10   PC11
Standard deviation     0.286 0.2507 0.2107 0.1984
Proportion of Variance 0.000 0.0000 0.0000 0.0000
Cumulative Proportion  1.000 1.0000 1.0000 1.0000
```

```
biplot(pc)
```

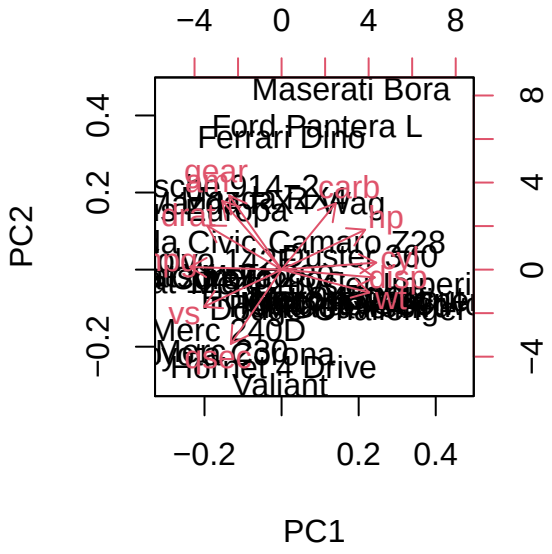


Scale the mtcars.

```
pc.scale<- prcomp(mtcars, scale= TRUE)
summary(pc.scale)
```

```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     2.5707 1.6280 0.79196 0.51923 0.47271 0.46000 0.3678
Proportion of Variance 0.6008 0.2409 0.05702 0.02451 0.02031 0.01924 0.0123
Cumulative Proportion  0.6008 0.8417 0.89873 0.92324 0.94356 0.96279 0.9751
                           PC8    PC9    PC10   PC11
Standard deviation     0.35057 0.2776 0.22811 0.1485
Proportion of Variance 0.01117 0.0070 0.00473 0.0020
Cumulative Proportion  0.98626 0.9933 0.99800 1.0000
```

```
biplot(pc.scale)
```



We want to scale,because some data have different spreads, so we want to scale to reduce that. We can check the plot before scaling and after scaling to check if we need to do that. Scale helps to reduce the SD.

Back to our cancer data set.

Do we need to scale this dataset? Yes, we do because the spread is very different.

```
# Check column means and standard deviations
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
```

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

```
    radius_mean     texture_mean   perimeter_mean        area_mean
     3.52404883       4.30103577      24.29898104     351.91412918
smoothness_mean compactness_mean
     0.01406413       0.05281276
```

```
# Perform PCA on wisc.data by completing the following code
wisc.pr <- prcomp( wisc.data, scale= TRUE)
#wisc.pr
```

How well do the PCS capture the variance in the original data?

```
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)?

PC1 captures 44.27%

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

PC3 >Q6. How many principal components (PCs) are required to describe at least 90% of the original variance in the data?

PC7

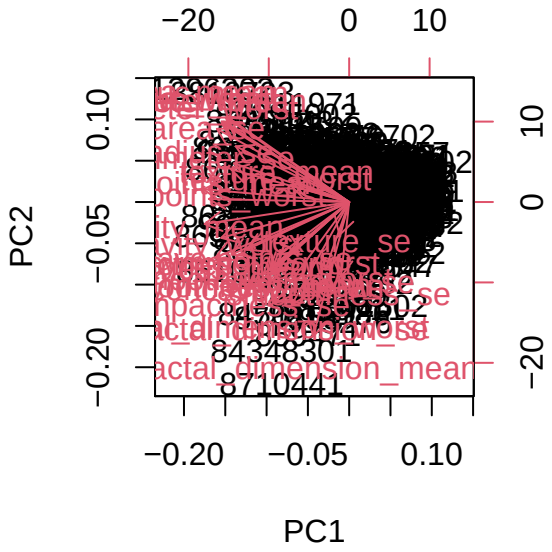
our main PC score plot(aka PC plot, PC1 vs PC2, coordination plot)

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

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

$class
[1] "prcomp"
```

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




```
#plot is hard to read
```

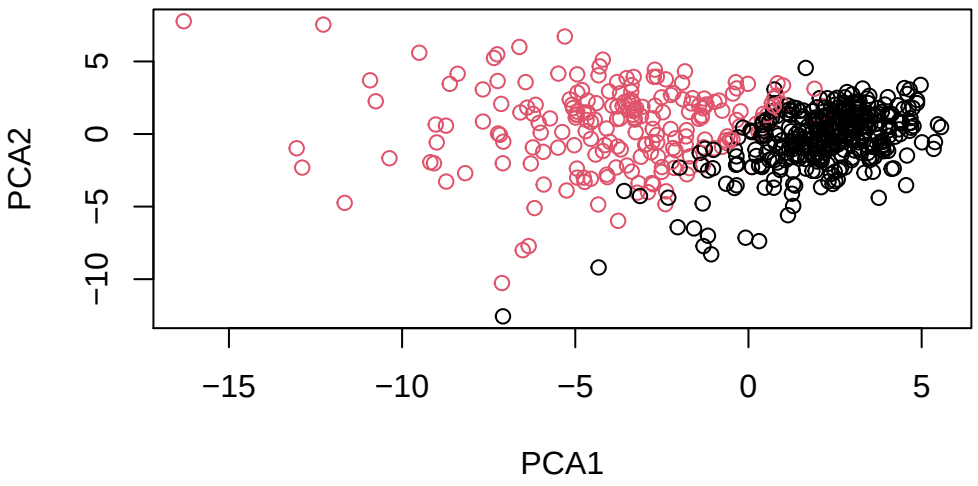
> Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

The plot generated by the biplot is hard to read as the cells are too compact together.

we need to build our own plot here:

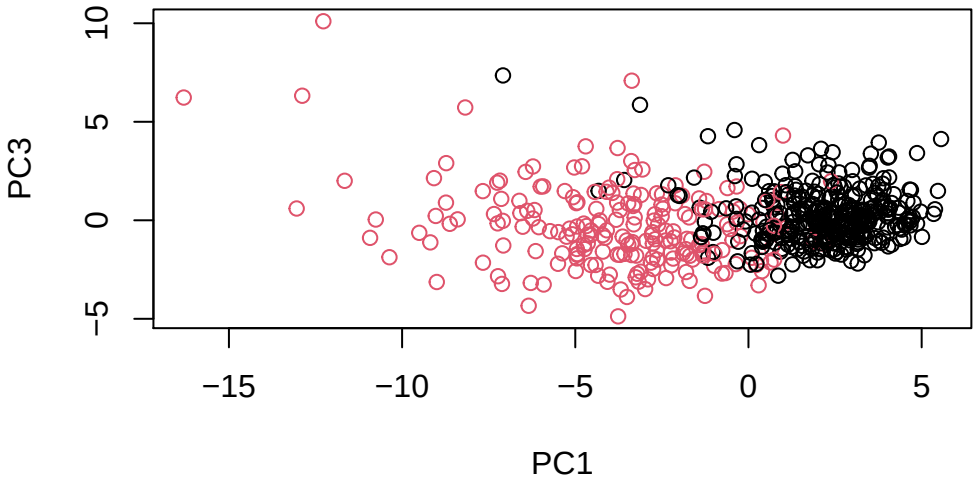
```
#wisc.pr$x
```

```
plot(wisc.pr$x[,1], wisc.pr$x[,2], col = diagnosis,xlab= "PCA1", ylab= "PCA2")
```



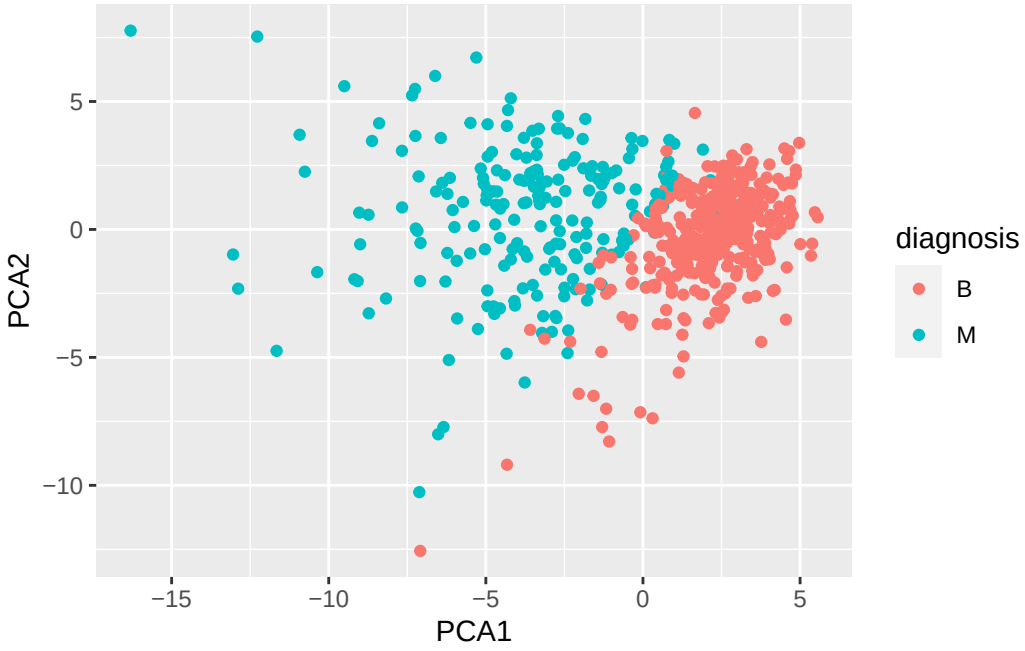
Q8. Q8. Generate a similar plot for principal components 1 and 3. What do you notice about these plots? The variance of the cells lie toward PCA1 (capture by PCA1)

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



Make a nice ggplot version

```
library(ggplot2)
pc <- as.data.frame(wisc.pr$x)
ggplot(pc, aes(wisc.pr$x[,1], wisc.pr$x[,2], col = diagnosis))+
  geom_point()+
  xlab( "PCA1")+
  ylab( "PCA2")
```




```
v<-summary(wisc.pr)
v$importance[2,]
```

```
    PC1     PC2     PC3     PC4     PC5     PC6     PC7     PC8     PC9    PC10
0.44272 0.18971 0.09393 0.06602 0.05496 0.04025 0.02251 0.01589 0.01390 0.01169
   PC11    PC12    PC13    PC14    PC15    PC16    PC17    PC18    PC19    PC20
0.00980 0.00871 0.00805 0.00523 0.00314 0.00266 0.00198 0.00175 0.00165 0.00104
   PC21    PC22    PC23    PC24    PC25    PC26    PC27    PC28    PC29    PC30
0.00100 0.00091 0.00081 0.00060 0.00052 0.00027 0.00023 0.00005 0.00002 0.00000
```

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

Calculate the variance of each principal component by squaring the sdev component of wisc.pr (i.e. wisc.pr$sdev^2). Save the result as an object called pr.var.

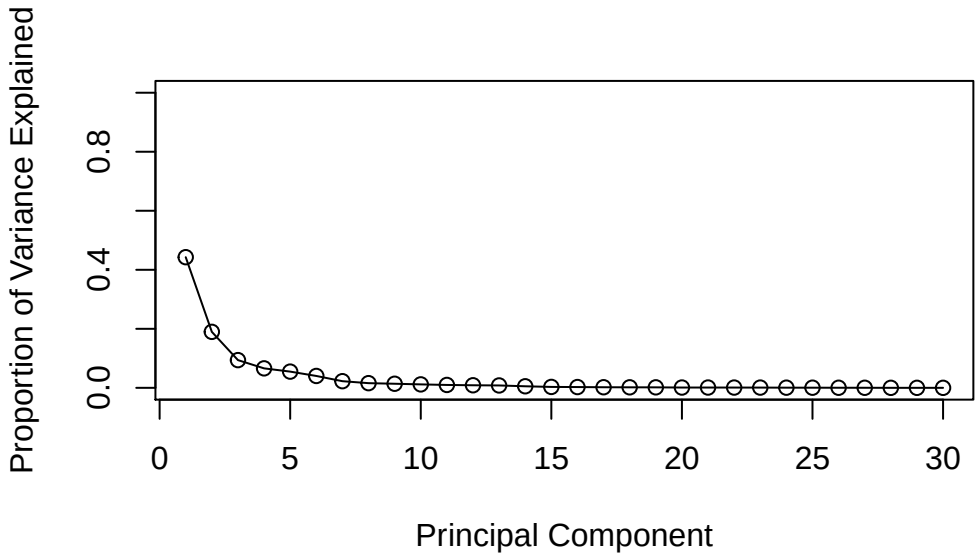
```
# Calculate variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

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

Calculate the variance explained by each principal component by dividing by the total variance explained of all principal components. Assign this to a variable called pve and create a plot of variance explained for each principal component.

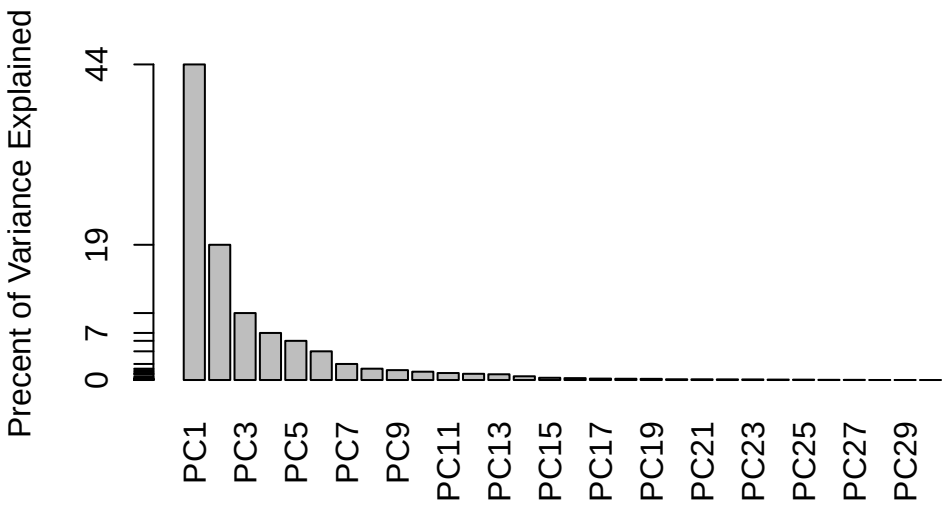
```
# Variance explained by each principal component: pve
pve <- pr.var / sum(pr.var)

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



The elbow is around 0.1.

```
# 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 )
```



Communicating PCA results: > Q9. For the first principal component, what is the component of the loading vector (i.e. wisc.pr$rotation[,1]) for the feature concave.points_mean? This tells us how much this original feature contributes to the first PC.

-0.26085376

```
head(wisc.pr$rotation)
```

```
                       PC1         PC2          PC3         PC4         PC5
radius_mean      -0.2189024  0.23385713 -0.008531243  0.04140896 -0.03778635
texture_mean     -0.1037246  0.05970609  0.064549903 -0.60305000  0.04946885
perimeter_mean   -0.2275373  0.21518136 -0.009314220  0.04198310 -0.03737466
area_mean        -0.2209950  0.23107671  0.028699526  0.05343380 -0.01033125
smoothness_mean  -0.1425897 -0.18611302 -0.104291904  0.15938277  0.36508853
compactness_mean -0.2392854 -0.15189161 -0.074091571  0.03179458 -0.01170397
                        PC6         PC7          PC8          PC9         PC10
radius_mean      0.018740790 -0.12408834  0.007452296 -0.223109764  0.09548644
texture_mean    -0.032178837  0.01139954 -0.130674825  0.112699390  0.24093407
perimeter_mean   0.017308445 -0.11447706  0.018687258 -0.223739213  0.08638562
area_mean       -0.001887748 -0.05165343 -0.034673604 -0.195586014  0.07495649
smoothness_mean -0.286374497 -0.14066899  0.288974575  0.006424722 -0.06929268
compactness_mean -0.014130949  0.03091850  0.151396350 -0.167841425  0.01293620
```

```
                       PC11        PC12       PC13         PC14        PC15
radius_mean      -0.04147149  0.05106746 0.01196721  0.059506135 -0.05111877
texture_mean      0.30224340  0.25489642 0.20346133 -0.021560100 -0.10792242
perimeter_mean   -0.01678264  0.03892611 0.04410950  0.048513812 -0.03990294
area_mean        -0.11016964  0.06543751 0.06737574  0.010830829  0.01396691
smoothness_mean   0.13702184  0.31672721 0.04557360  0.445064860 -0.11814336
compactness_mean  0.30800963 -0.10401704 0.22928130  0.008101057  0.23089996
                       PC16        PC17         PC18        PC19        PC20
radius_mean      -0.1505839  0.20292425  0.146712338  0.22538466 -0.04969866
texture_mean     -0.1578420 -0.03870612 -0.041102985  0.02978864 -0.24413499
perimeter_mean   -0.1144540  0.19482131  0.158317455  0.23959528 -0.01766501
area_mean        -0.1324480  0.25570576  0.266168105 -0.02732219 -0.09014376
smoothness_mean  -0.2046132  0.16792991 -0.352226802 -0.16456584  0.01710096
compactness_mean  0.1701784 -0.02030771  0.007794138  0.28422236  0.48868633
                       PC21        PC22           PC23        PC24         PC25
radius_mean      -0.06857001 -0.07292890 -0.0985526942 -0.18257944 -0.01922650
texture_mean      0.44836947 -0.09480063 -0.0005549975  0.09878679  0.08474593
perimeter_mean   -0.06976904 -0.07516048 -0.0402447050 -0.11664888  0.02701541
area_mean        -0.01844328 -0.09756578  0.0077772734  0.06984834 -0.21004078
smoothness_mean  -0.11949175 -0.06382295 -0.0206657211  0.06869742  0.02895489
compactness_mean  0.19262140  0.09807756  0.0523603957 -0.10413552  0.39662323
                       PC26        PC27           PC28         PC29
radius_mean      -0.12947640 -0.13152667  2.111940e-01  0.211460455
texture_mean     -0.02455666 -0.01735731 -6.581146e-05 -0.010533934
perimeter_mean   -0.12525595 -0.11541542  8.433827e-02  0.383826098
area_mean         0.36272740  0.46661248 -2.725083e-01 -0.422794920
smoothness_mean  -0.03700369  0.06968992  1.479269e-03 -0.003434667
compactness_mean  0.26280847  0.09774871 -5.462767e-03 -0.041016774
                       PC30
radius_mean       0.702414091
texture_mean      0.000273661
perimeter_mean   -0.689896968
area_mean        -0.032947348
smoothness_mean  -0.004847458
compactness_mean  0.044674186
```

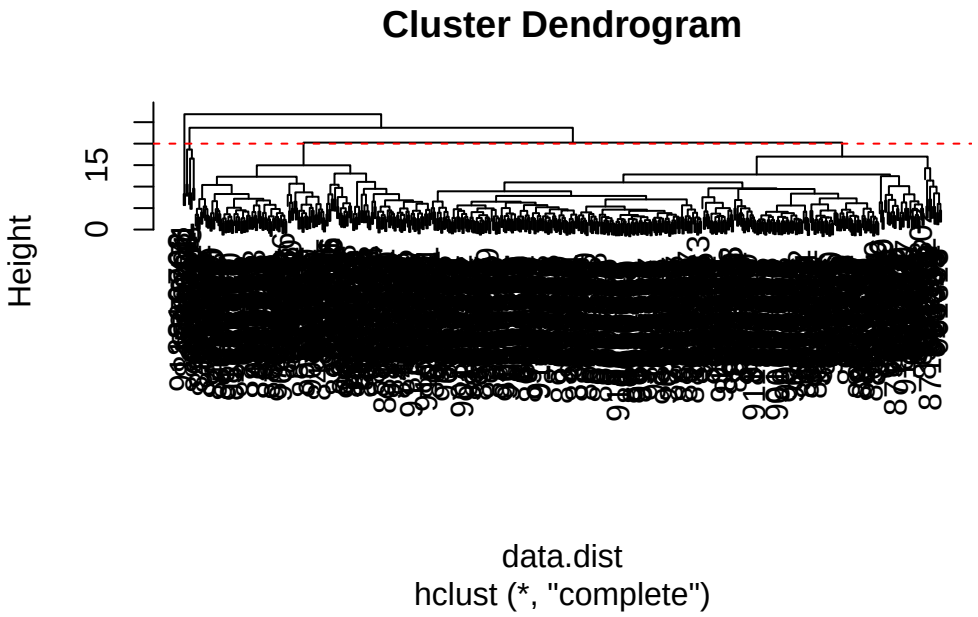
Calculate the (Euclidean) distances between all pairs of observations in the new scaled dataset and assign the result to data.dist.

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

Create a hierarchical clustering model using complete linkage. Manually specify the method argument to hclust() and assign the results to wisc.hclust.

```
wisc.hclust <- hclust(data.dist, method ="complete")
```

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



Q10. Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters?

H=20

Use cutree() to cut the tree so that it has 4 clusters. Assign the output to the variable wisc.hclust.clusters.

```
wisc.hclust.clusters <- cutree(wisc.hclust, h=20)
```

We can use the table() function to compare the cluster membership to the 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
```

Using different methods

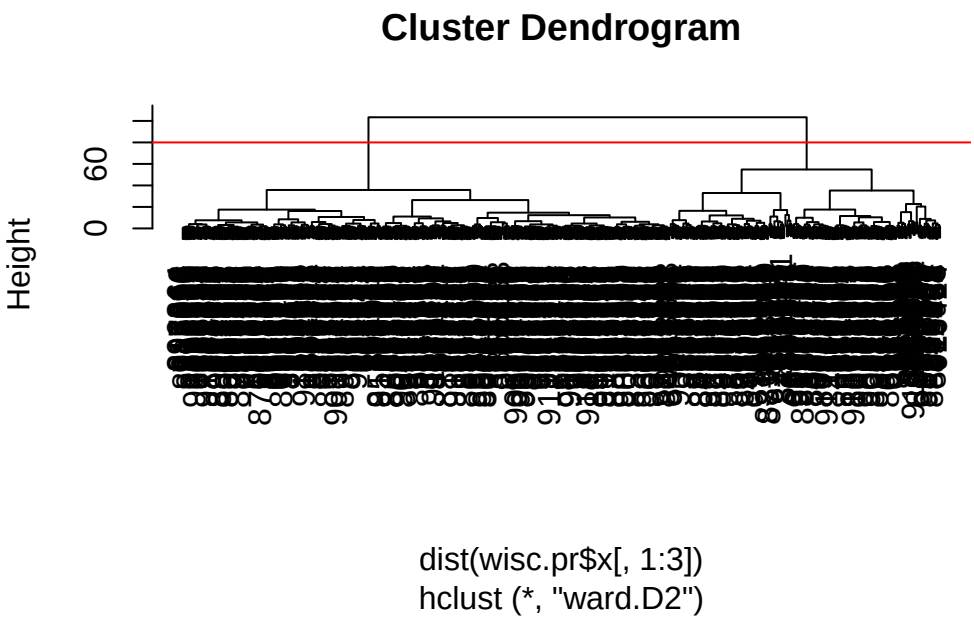
> Q12. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning.

The `ward.D2` gives me the better results, as I can distinct the number of clusters clearly.

4. Combining methods

Here we will use the results of PCA as the input to a clustering analysis. We stare with 3 PCs.

```
wisc.pr.hclust<- hclust(dist(wisc.pr$x[,1:3]), method ="ward.D2")
plot(wisc.pr.hclust)
abline(h=80, col="red")
```



Q13. How well does the newly created model with four clusters separate out the two diagnoses?

By grps 1 and 2.

```
grps <- cutree(wisc.pr.hclust, h=80)
table(grps)
```

```
grps
  1   2
203 366
```

Q14. How well do the 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.

By grps 1 and 2 and diagnosis by B and M.

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

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

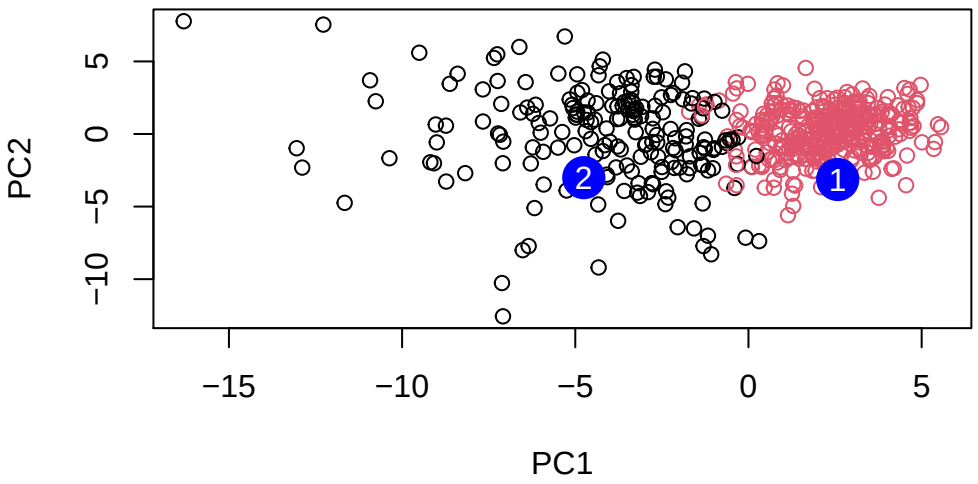
Q14. How well do the hierarchical clustering models you created in previous sections (i.e. before PCA) do in terms of separating the diagnoses? Again, use the table() function to compare the output of each model (wisc.km$cluster and wisc.hclust.clusters) with the vector containing the actual diagnoses.

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
```

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



Q16. Which of these new patients should we prioritize for follow up based on your

results?

Patient 2.