# class08:Machine Learning Mini Project

**Quarto**

Quarto enables you to weave together content and executable code into a finished document. To learn more about Quarto see https://quarto.org.

**Running Code**

When you click the **Render** button a document will be generated that includes both content and the output of embedded code. You can embed code like this:

```
1 + 1
```

```
[1] 2
```

You can add options to executable code like this

```
[1] 4
```

The `echo: false` option disables the printing of code (only output is displayed).

## Breast Cancer Project

Today we are going to explore some data from the University of Wisconsin Cancer Center on Breast biopsy data.

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

|          | 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 patient samples are in this dataset? There are 569 patient samples.

```
nrow(wisc.data)
```

[1] 569

There are 569 parients in this dataset

Q. How many cancer (M) and non cancer (B) samples are there?

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

```
  B   M
357 212
```

Save the diagnosis for later use as a reference to compare how well we do with PCA etc.

```
diagnosis <- as.factor(wisc.data$diagnosis)
#diagnosis
```

Now exclude the diagnosis column from the data:

```
wisc<-wisc.data[, -1]
```

Q. How many "dimenstions", "variables", "columns" are there is this dataset?

```r
ncol(wisc)
```

[1] 30

# Principal Component Analysis (PCA)

To perform PCA in R, we can use the `prcomp()` function. It takes as input a numeric dataset and optional `scale=FALSE/TRUE` argument.

We generally always want to set `scale=TRUE` but let's make sure by checking if the meand and standard deviation values are different across these 30 columns.

```r
round(colMeans(wisc))
```

|                     radius_mean |             texture_mean |            perimeter_mean |
|--------------------------------:|-------------------------:|--------------------------:|
|                              14 |                       19 |                        92 |
|                       area_mean |          smoothness_mean |          compactness_mean |
|                             655 |                        0 |                         0 |
|                  concavity_mean |      concave.points_mean |             symmetry_mean |
|                               0 |                        0 |                         0 |
|          fractal_dimension_mean |                radius_se |                 texture_se |
|                               0 |                        0 |                         1 |
|                    perimeter_se |                  area_se |              smoothness_se |
|                               3 |                       40 |                         0 |
|                  compactness_se |             concavity_se |          concave.points_se |
|                               0 |                        0 |                         0 |
|                     symmetry_se |     fractal_dimension_se |               radius_worst |
|                               0 |                        0 |                        16 |
|                   texture_worst |           perimeter_worst |                area_worst |
|                              26 |                      107 |                       881 |
|                smoothness_worst |         compactness_worst |           concavity_worst |
|                               0 |                        0 |                         0 |
|            concave.points_worst | symmetry_worst fractal_dimension_worst |               |
|                               0 |                        0 |                         0 |

```r
pca<-prcomp(wisc, scale=TRUE)
summary(pca)
```

```
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
```

```r
attributes(pca)
```

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

$class
[1] "prcomp"
```

```r
plot(pca$x[,1],pca$x[,2], col=diagnosis)
```
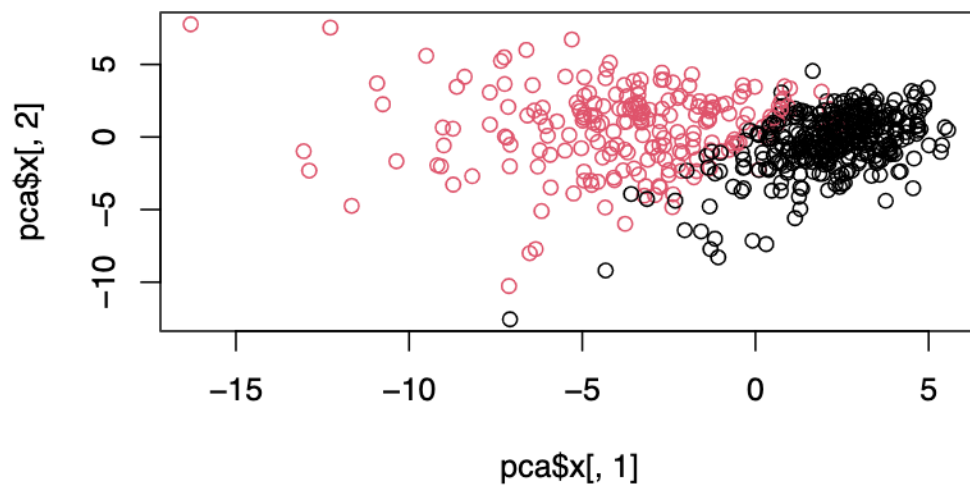
```r
library(ggplot2)

x<-as.data.frame(pca$x)

ggplot(x)+
  aes(PC1, PC2, col=diagnosis)+
  geom_point()
```

Q. How much variance is captured in the top 3 PCs.

They capture 76% of the total variance.

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.

```
pca$rotation["concave.points_mean",1]
```

```
[1] -0.2608538
```

```
attributes(pca)
```

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

$class
[1] "prcomp"
```

# Combine PCA results with clustering

We can use our new PCA variables (i.e. the scores along the PCs contained in t `pca$x`) as input for other methods such as clustering.

```
#Hclust needs a distance matrix as input
d<-dist(pca$x[,1:3])
hc<-hclust(d, method="ward.D2")
plot(hc)
```

**Cluster Dendrogram**



d
hclust (*, "ward.D2")
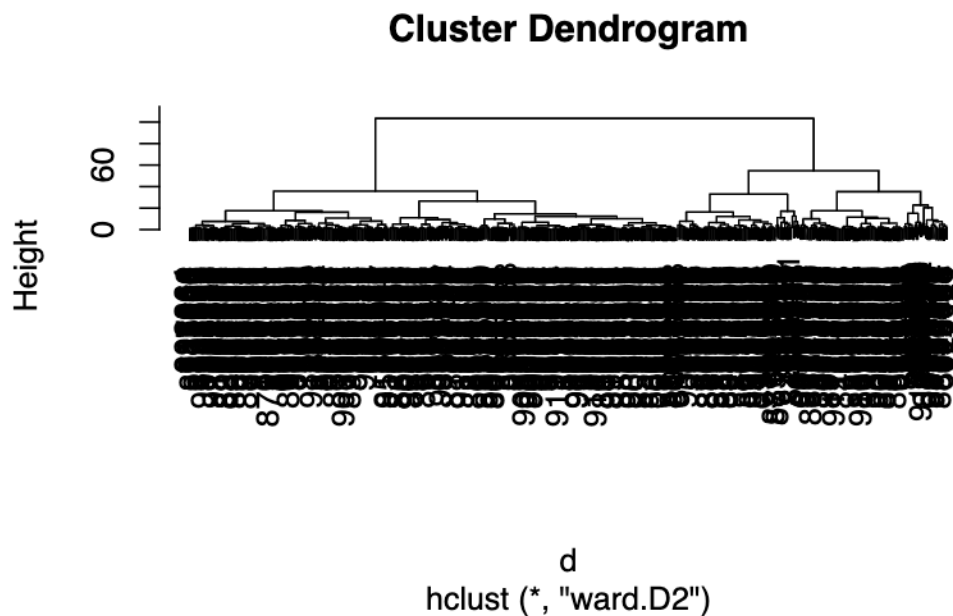
To get our cluster membership vector we can use the `cutree()` function and specify a height (`h`) or number of groups (`k`).

```
grps<-cutree(hc,h=80)
table(grps)
```

```
grps
  1   2
203 366
```

I want to find out how many diagnosis "M" and "B" are in each group?

```
table(diagnosis)
```

```
diagnosis
  B   M
357 212
```

```
table(diagnosis, grps)
```
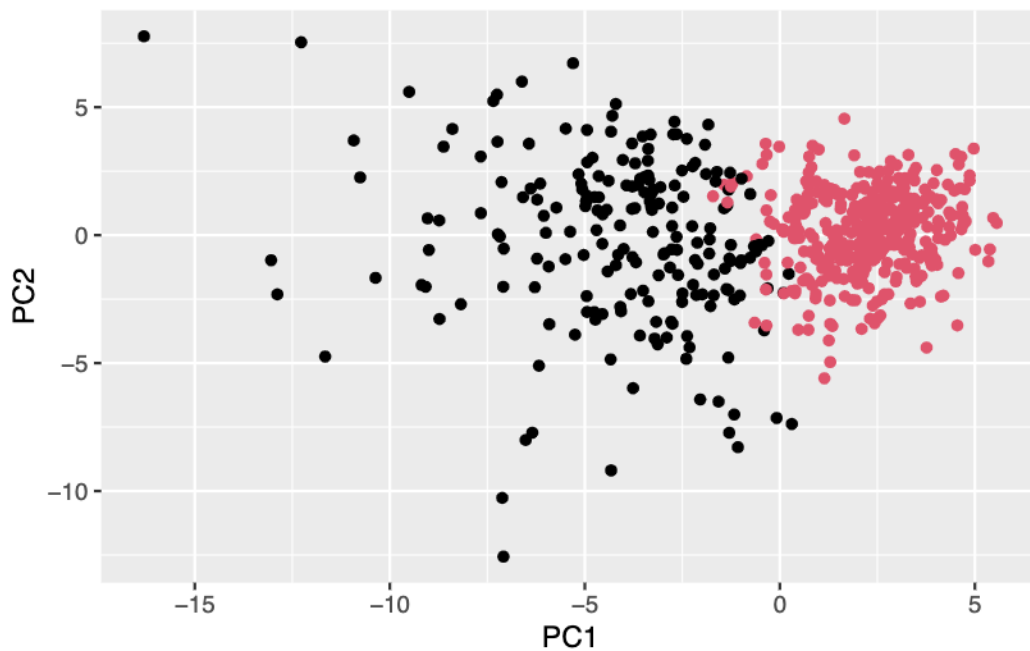
```
         grps
diagnosis   1    2
        B  24  333
        M 179   33
```

We can also plot our results using our clustering vector `grps`

```
plot(pca$x[,1],pca$x[,2], col=grps)
```
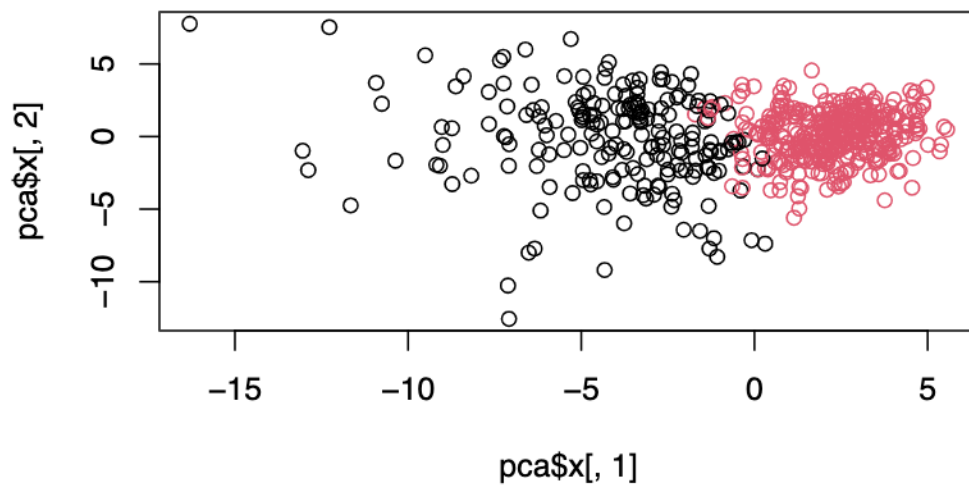
```
ggplot(x)+
  aes(PC1, PC2) +
  geom_point(col=grps)
```



Q15. What is the specificity and sensitivity of our current results?

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

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

Specificity= TP/(TP+FN)

```
179/(179+33)
```

```
[1] 0.8443396
```

Sensitivity=TN/(TN+FN)

```r
333/(333+33)
```

```
[1] 0.9098361
```