Class 08: Breast Cancer Mini Project

Neha (PID: A17567541)

Before we ge stuck into project work, we will have a quick look at applying PCA to some example RNAseq data (tail end of lab 7)

```
url2 <- "https://tinyurl.com/expression-CSV"
rna.data <- read.csv(url2, row.names=1)
head(rna.data)
```

```
      wt1 wt2  wt3  wt4 wt5 ko1 ko2 ko3 ko4 ko5
gene1  439 458  408  429 420  90  88  86  90  93
gene2  219 200  204  210 187 427 423 434 433 426
gene3 1006 989 1030 1017 973 252 237 238 226 210
gene4  783 792  829  856 760 849 856 835 885 894
gene5  181 249  204  244 225 277 305 272 270 279
gene6  460 502  491  491 493 612 594 577 618 638
```

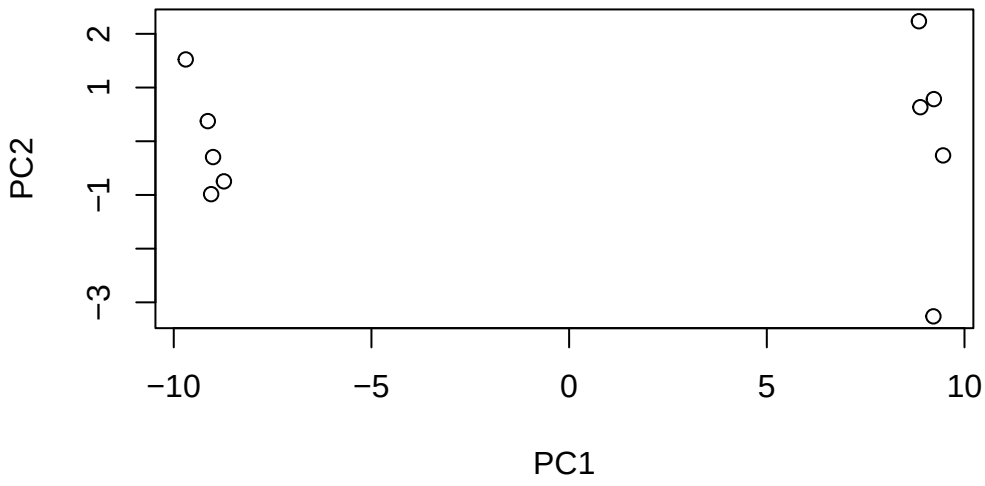
Q. How many genes are in this dataset?

```
nrow(rna.data)
```

```
[1] 100
```

Run PCA

```
## Again we have to take the transpose of our data
pca <- prcomp(t(rna.data), scale=TRUE)

## Simple un polished plot of pc1 and pc2
plot(pca$x[,1], pca$x[,2], xlab="PC1", ylab="PC2")
```




```
summary(pca)
```

```
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     9.6237 1.5198 1.05787 1.05203 0.88062 0.82545 0.80111
Proportion of Variance 0.9262 0.0231 0.01119 0.01107 0.00775 0.00681 0.00642
Cumulative Proportion  0.9262 0.9493 0.96045 0.97152 0.97928 0.98609 0.99251
                           PC8     PC9      PC10
Standard deviation     0.62065 0.60342 3.457e-15
Proportion of Variance 0.00385 0.00364 0.000e+00
Cumulative Proportion  0.99636 1.00000 1.000e+00
```

```
#We have 5 wt and 5 ko samples
pca$x
```

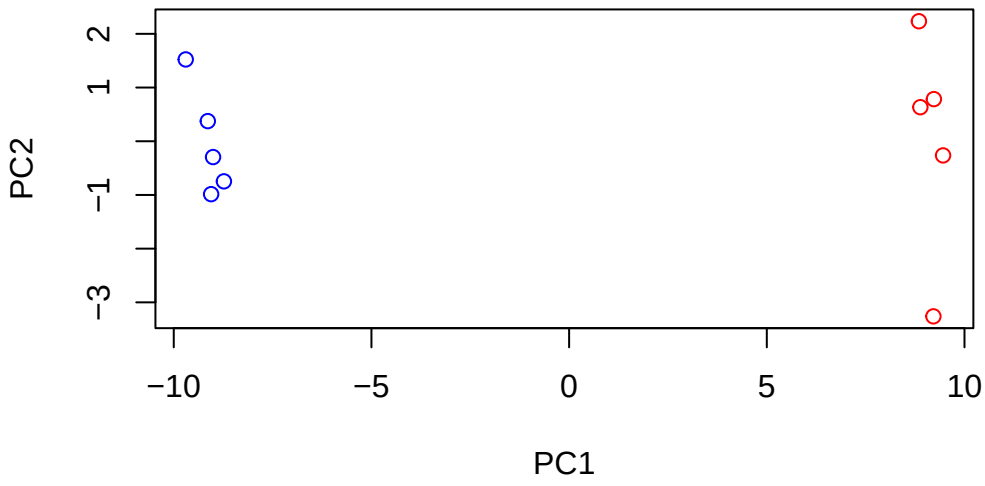
```
          PC1        PC2        PC3        PC4        PC5        PC6
wt1 -9.697374  1.5233313 -0.2753567  0.7322391 -0.6749398  1.1823860
wt2 -9.138950  0.3748504  1.0867958 -1.9461655  0.7571209 -0.4369228
wt3 -9.054263 -0.9855163  0.4152966  1.4166028  0.5835918  0.6937236
wt4 -8.731483 -0.7468371  0.5875748  0.2268129 -1.5404775 -1.2723618
```

```
wt5 -9.006312 -0.2945307 -1.8498101 -0.4303812  0.8666124 -0.2496025
ko1  8.846999  2.2345475 -0.1462750 -1.1544333 -0.6947862  0.7128021
ko2  9.213885 -3.2607503  0.2287292 -0.7658122 -0.4922849  0.9170241
ko3  9.458412 -0.2636283 -1.5778183  0.2433549  0.3654124 -0.5837724
ko4  8.883412  0.6339701  1.5205064  0.7760158  1.2158376 -0.1446094
ko5  9.225673  0.7845635  0.0103574  0.9017667 -0.3860869 -0.8186668
            PC7         PC8         PC9         PC10
wt1 -0.24446614  1.03519396  0.07010231 3.073930e-15
wt2 -0.03275370  0.26622249  0.72780448 1.963707e-15
wt3 -0.03578383 -1.05851494  0.52979799 2.893519e-15
wt4 -0.52795595 -0.20995085 -0.50325679 2.872702e-15
wt5  0.83227047 -0.05891489 -0.81258430 1.693090e-15
ko1 -0.07864392 -0.94652648 -0.24613776 4.052314e-15
ko2  0.30945771  0.33231138 -0.08786782 3.268219e-15
ko3 -1.43723425  0.14495188  0.56617746 2.636780e-15
ko4 -0.35073859  0.30381920 -0.87353886 3.615164e-15
ko5  1.56584821  0.19140827  0.62950330 3.379241e-15
```

```
mycols<-c(rep("blue",5), rep("red",5))
mycols
```

```
 [1] "blue" "blue" "blue" "blue" "blue" "red"  "red"  "red"  "red"  "red"
```

```
plot(pca$x[,1], pca$x[,2], xlab="PC1", ylab="PC2", col=mycols)
```



I could examine which genes contribute most to the first PC

```
head(sort(abs(pca$rotation[,1]), decreasing=T))
```

```
  gene100     gene66     gene45     gene68     gene98     gene60
0.1038708 0.1038455 0.1038402 0.1038395 0.1038372 0.1038055
```

#Analysis of FNA data

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

Note that the first column here `wisc.df$diagnosis` is a pathologist provided expert diagnosis. We will not be using this for our unsupervised analysis as it is essentially the "answer" to the question which cell samples are malignant or benign.

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

I want to make sure I remove that column from the dataset

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

Q1. How many observations are in this dataset?

```
dim(wisc.data)
```

```
[1] 569  30
```

569 rows

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

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

```
  B   M
357 212
```

212 patients have a malignant diagnosis

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

```
names<-colnames(wisc.data)
length(grep("_mean",names))
```

```
[1] 10
```

```
#length tells you how many, grep gives you the position of the variables
```

Principal Component Analysis

Here, we will use `prcomp()` on the `wisc.data` object- the one without the diagnosis column.

First, we have to decide whether to use `scale=TRUE` argument when we run `prcomp()`

We can look at the means and sd of each column. If they are all similar, we are all good. If not, we should use `scale=TRUE`

```
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
```

These are very different, so we should set `scale=TRUE`

```
wisc.pr<-prcomp(wisc.data,scale=TRUE)
#summary(wisc.pr)
```

> Q4. From your results, what proportion of the original variance is captured by the first principal components (PC1)?

44.27%

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

3 PCs capture 72.6 of the original variance

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

7 PCs

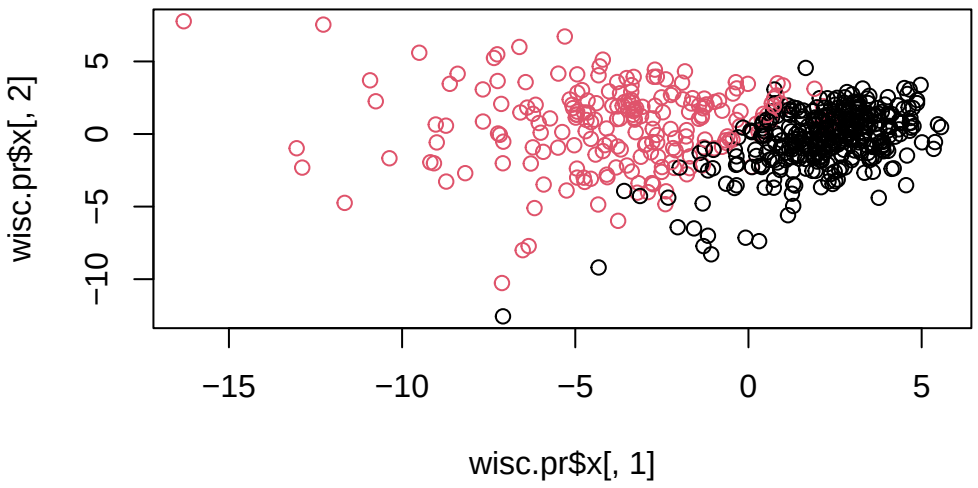
```
#biplot(wisc.pr)
```

we need to make our own plot.

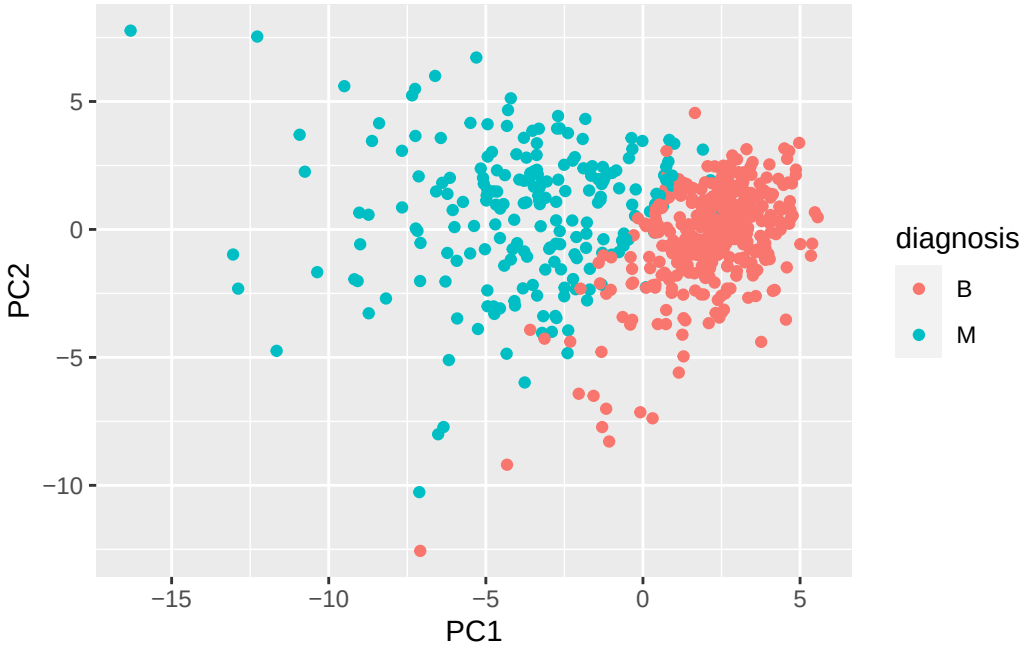
```
attributes(wisc.pr)
```

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

$class
[1] "prcomp"
```

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




```
library(ggplot2)
pc<-as.data.frame(wisc.pr$x)

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



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?

```
wisc.pr$rotation["concave.points_mean",1]
```

```
[1] -0.2608538
```

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

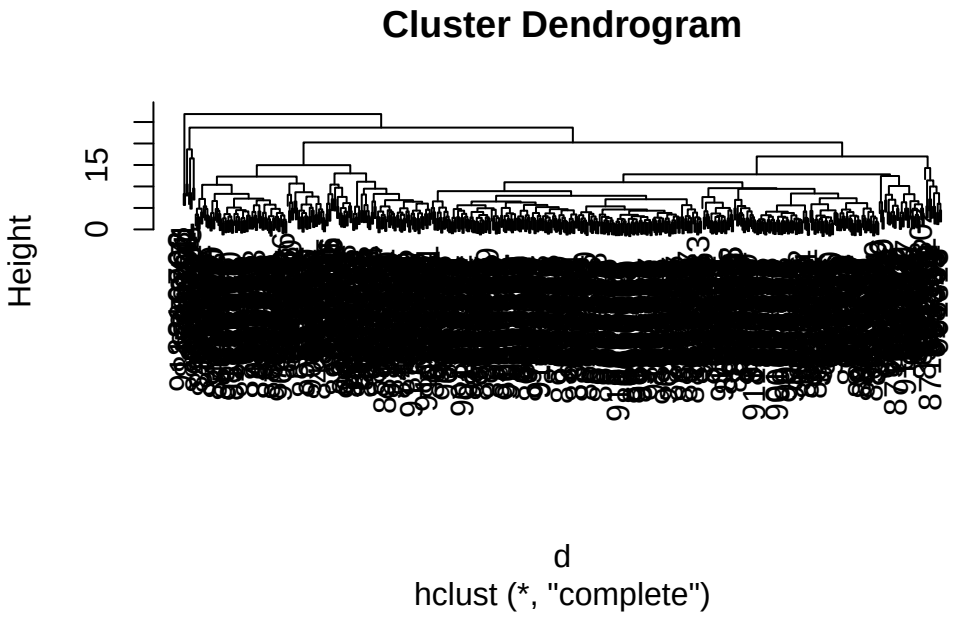
```
tbl<-summary(wisc.pr)
which(tbl$importance[3,]>0.8)[1]
```

```
PC5
  5
```

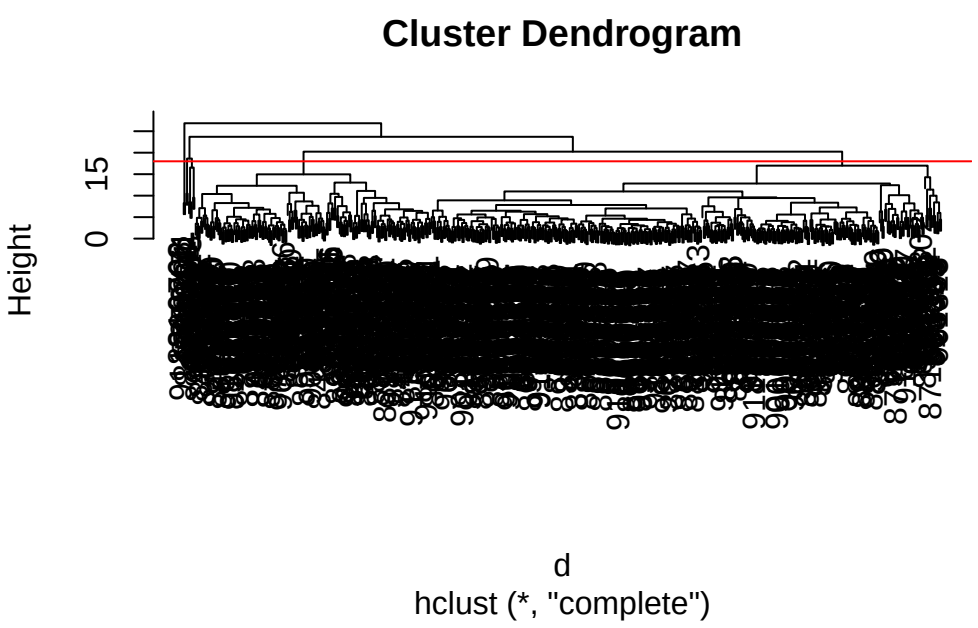
Hierarchical clustering

The main function for hierarchical clustering is called `hclust()`. It takes a distance matrix as input.

```
d<- dist(scale(wisc.data))
wisc.hclust <- hclust(d)
plot(wisc.hclust)
```




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




```
grps<-cutree(wisc.hclust, h=18)
table(grps)
```

```
grps
  1   2   3   4   5
177   5 383   2   2
```

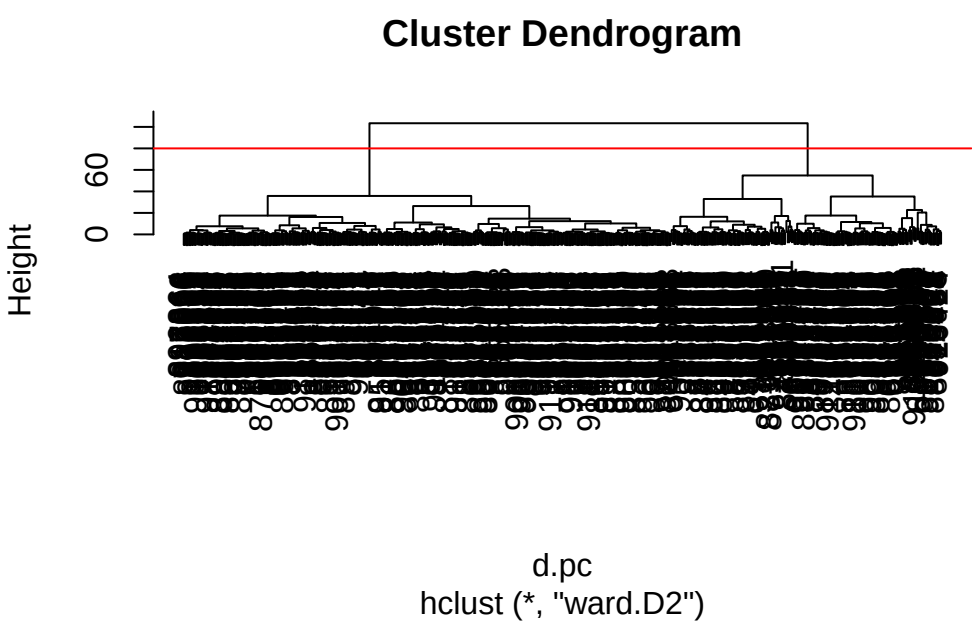
Come back here to see how cluster groups correspond to M or B

Combining methods

Here we will perform clustering on our PCA results rather than the original data.

In other words we will cluster using `wisc.pr$x`- our new better variables or PCs. We can choose as many or as few PCs as we like. It is your call.

```
#Did we turn the PC into an hclust plot?
d.pc<-dist(wisc.pr$x[,1:3])
wisc.pr.hclust<-hclust(d.pc, method="ward.D2")
plot(wisc.pr.hclust)
abline(h=80, col="red")
```




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

```
grps
  1   2
203 366
```

We can use `table()` function to make a cross-table as well as just a count table.

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

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

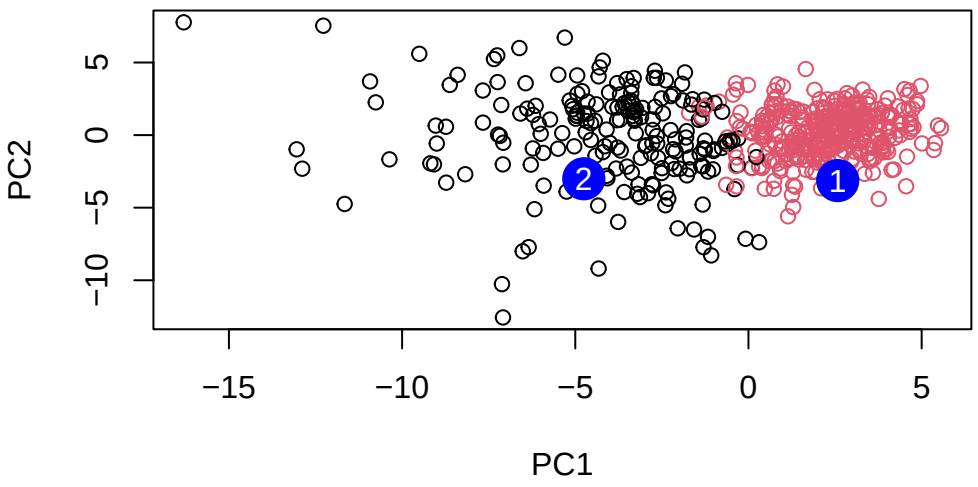
Cluster 1 mostly captures cancer (M) and cluster 2 mostly captures healthy (B) individuals. There are some false positives and negatives, but it is a good indication of the real expert results. We can play with the PC and hclust to make it better.

Prediction

```
url <- "https://tinyurl.com/new-samples-CSV"
new <- read.csv(url)
npc <- predict(wisc.pr, newdata=new)
```

And plot this up

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

We should look at patient 2 for follow up.