Unsupervised Learning Mini-Project

Mirte Ciz Marieke Kuijpers

11/02/2022

Set Up

The first step is loading the data.

```
# Save the input data file into your Project directory then read in the data
wisc.df <- read.csv("WisconsinCancer.csv", row.names=1)

# Check the data is as expected
str(wisc.df)
```

```
## 'data.frame':    569 obs. of  31 variables:
##  $ diagnosis              : chr  "M" "M" "M" "M" ...
##  $ radius_mean            : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean           : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean         : num  122.8 132.9 130 77.6 135.1 ...
##  $ area_mean              : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
##  $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911 0.01149
...
##  $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511
...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst             : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
```

```
##  $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
##  $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...
```

Note that the id's of each observation are changed to row names to help eliminate bias. Given that the first column provides an expert's diagnosis, this column is basically the answers the code is meant to find. Therefore, to ensure the model we create does not use the 'answers', this column should be removed for now. Later these 'answers' can be used to test/check the model.

```
# Use -1 to remove the first column
wisc.data <- wisc.df[,-1]

# Create diagnosis vector to check work with later
diagnosis <- wisc.df[,1]

# Check the data is as expected
str(wisc.data)
```

```
## 'data.frame':    569 obs. of  30 variables:
##  $ radius_mean            : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean           : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean         : num  122.8 132.9 130 77.6 135.1 ...
##  $ area_mean              : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
##  $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911 0.01149
...
##  $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511
...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst             : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
##  $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
```

```
##  $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...

str(diagnosis)

##  chr [1:569] "M" "M" "M" "M" "M" "M" "M" "M" "M" "M" "M" "M" "M" "M" "M" "M"
...
```

Now that the data is loaded, analysis can begin.

Exploratory Data Analysis

Question 1

To find the number of observations in the original data the `dim()` or `str()` functions could be used.

```
# Find the number of observations in the original data
dim(wisc.df)

## [1] 569  31
```

The dim() function reveals that the original data has 596 observations (i.e. patients) with 31 variables noted for each. Note that the `str()` function also provides this information, but gives further information, necessary for this question, and thus was not used in this case. Another possibility would have been to use the `nrow()` function.

Question 2

To find the number of observations (i.e. patients) with a malignant diagnosis, we can `sum()` across booleans because TRUE = 1 and FALSE = 0 in R.

```
# Make a vector giving TRUE when the diagnosis is malignant and FALSE
otherwise
malignant <- wisc.df$diagnosis == "M" # Note that in the data malignant is
denoted by an M

# Check vector is as expected
malignant

##   [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
```

```
# sum() to find the number of malignant diagnoses
sum(malignant)
```

```
## [1] 212
```

This shows that 212 cases (i.e. patients) were diagnosed as malignant. Note that the above code would work just as well if diagnosis vector created earlier was used in the creation of the malignant vector. A more efficient way to have done this would have been to use `table()`.

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

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

Question 3

To find how many variables end in "_mean", one can use the `grep()` function, which acts a bit like a search, on the column names.

```
# Obtain the column names in a separate vector for ease
col.names <- colnames(wisc.df)

# Use grep() to find the _mean ending variables
m <- grep("*_mean$", col.names)

# Number of variables with this ending
length(m)
```

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

This code reveals there are 10 variables ending in "_mean".

Question 4

Given that there are many variables, principle component analysis seems appropriate here. We have to first check if the different variables have similar means and standard deviations, such that they equally contribute to the following analysis

```
# 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
```

```
# and standard deviations
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
```

Given these are not all similar, it is appropriate to scale the data, such that the PCA function (`prcomp()` for base R) will give equal weight to each variable.

```
# Perform PCA on wisc.data
wisc.pr <- prcomp(wisc.data, scale = TRUE)

# Look at a summary of the results
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
```

This PCA finds that 44.3% (to 3 s.f.) of the original variance is captured by the first principal coordinate (PC1).

Question 5

To find how many principle components are required to capture 70% of the original variance we can use a `while` loop with a `sum()` function. The principle components are already ordered by the amount of variance they explain; if this were not the case it would be important to first order them in descending order.

```
# Include summary data in r object
summ <- summary(wisc.pr)

# Double check this worked
summ$importance

##                             PC1      PC2      PC3      PC4      PC5
PC6
## Standard deviation     3.644394 2.385656 1.678675 1.407352 1.284029
1.098798
## Proportion of Variance 0.442720 0.189710 0.093930 0.066020 0.054960
0.040250
## Cumulative Proportion  0.442720 0.632430 0.726360 0.792390 0.847340
0.887590
##                               PC7       PC8       PC9      PC10      PC11
## Standard deviation     0.8217178 0.6903746 0.6456739 0.5921938 0.5421399
## Proportion of Variance 0.0225100 0.0158900 0.0139000 0.0116900 0.0098000
## Cumulative Proportion  0.9101000 0.9259800 0.9398800 0.9515700 0.9613700
##                              PC12      PC13      PC14      PC15      PC16
## Standard deviation     0.5110395 0.4912815 0.3962445 0.3068142 0.2826001
## Proportion of Variance 0.0087100 0.0080500 0.0052300 0.0031400 0.0026600
```

```
## Cumulative Proportion  0.9700700 0.9781200 0.9833500 0.9864900 0.9891500
##                            PC17      PC18      PC19      PC20      PC21
## Standard deviation     0.2437192 0.2293878 0.2224356 0.1765203 0.1731268
## Proportion of Variance 0.0019800 0.0017500 0.0016500 0.0010400 0.0010000
## Cumulative Proportion  0.9911300 0.9928800 0.9945300 0.9955700 0.9965700
##                            PC22      PC23      PC24      PC25      PC26
## Standard deviation     0.1656484 0.1560155 0.1343689 0.1244238 0.0904303
## Proportion of Variance 0.0009100 0.0008100 0.0006000 0.0005200 0.0002700
## Cumulative Proportion  0.9974900 0.9983000 0.9989000 0.9994200 0.9996900
##                             PC27      PC28       PC29       PC30
## Standard deviation     0.08306903 0.0398665 0.02736427 0.01153451
## Proportion of Variance 0.00023000 0.0000500 0.00002000 0.00000000
## Cumulative Proportion  0.99992000 0.9999700 1.00000000 1.00000000

summ$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

# Make loop to answer question

## Create a counter and sum variable
count <- 0
x <- 0

while(x < 0.7){
  count <- count + 1
  x <- x + summ$importance[2, count]
}

print(paste("The first", count, "principle components account for", x*100, "%
of the original variance."))

## [1] "The first 3 principle components account for 72.636 % of the original
variance."
```

This analysis reveals that the first three principle components account for at least 70% of the original variance.

Question 6

A similar method can be used to find how many components are required to capture at least 90% of the original variance.

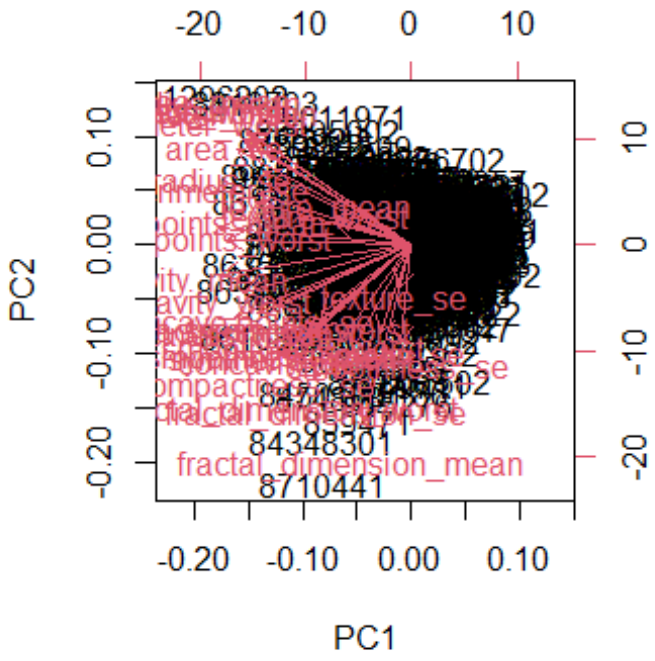
```
# Make loop to answer question

## Create a counter and sum variable
count2 <- 0
x2 <- 0

while(x2 < 0.9){
  count2 <- count2 + 1
  x2 <- x2 + summ$importance[2, count2]
}

print(paste("The first", count2, "principle components account for", x2*100,
"% of the original variance."))

## [1] "The first 7 principle components account for 91.01 % of the original
variance."
```

This reveals that the first 7 principle components are required to account for at least 90% of the original variance.

Question 7

Often plotting the data is the best way to understand it. One plotting technique sometimes used for PCA output is to use the `biplot()` function.

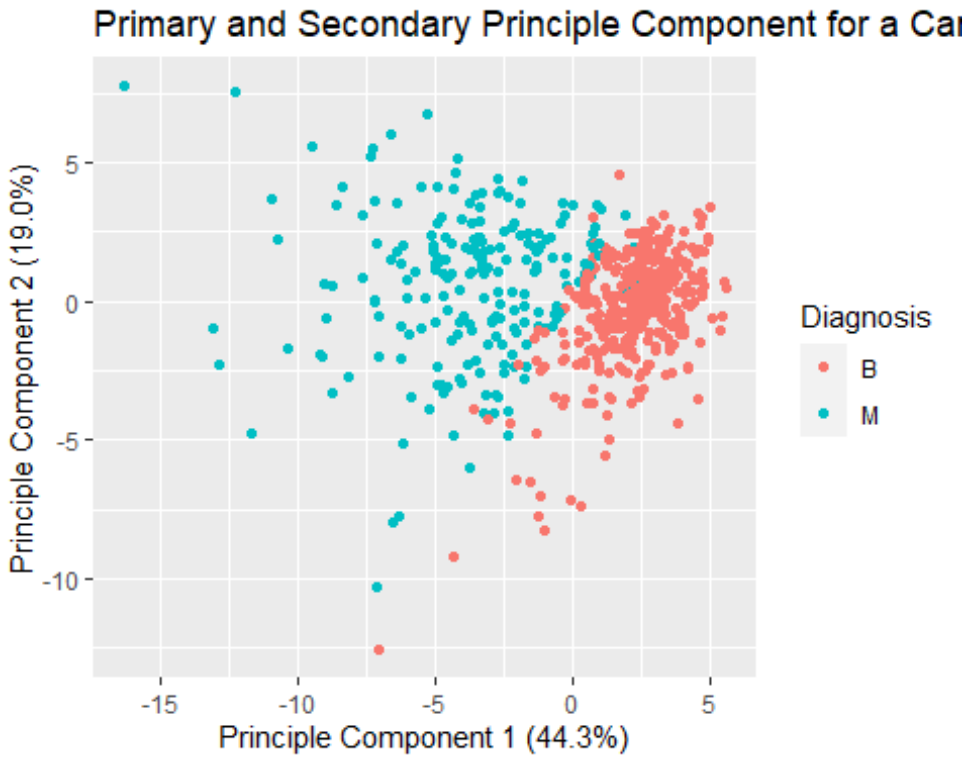
```
biplot(wisc.pr)
```



This shows the direction the different variables cause observations to tend towards (the top and right axes indicate the units of the lines which are vectors that represent the loadings of each variable on PC1 and PC2), and labels each observation with its ID (row name). For a smaller data set with short observation row names, this would be useful. However, in this case, the plot is extremely busy and the plot is very difficult to interpret. Nevertheless, the lines (vectors) are interesting to consider, as they give an idea of the magnitude and direction of impact of each variable, and which of them are correlated etc.

A better PCA plot is shown below, while it doesn't show the loadings vectors, it is far cleaner and easier to interpret.

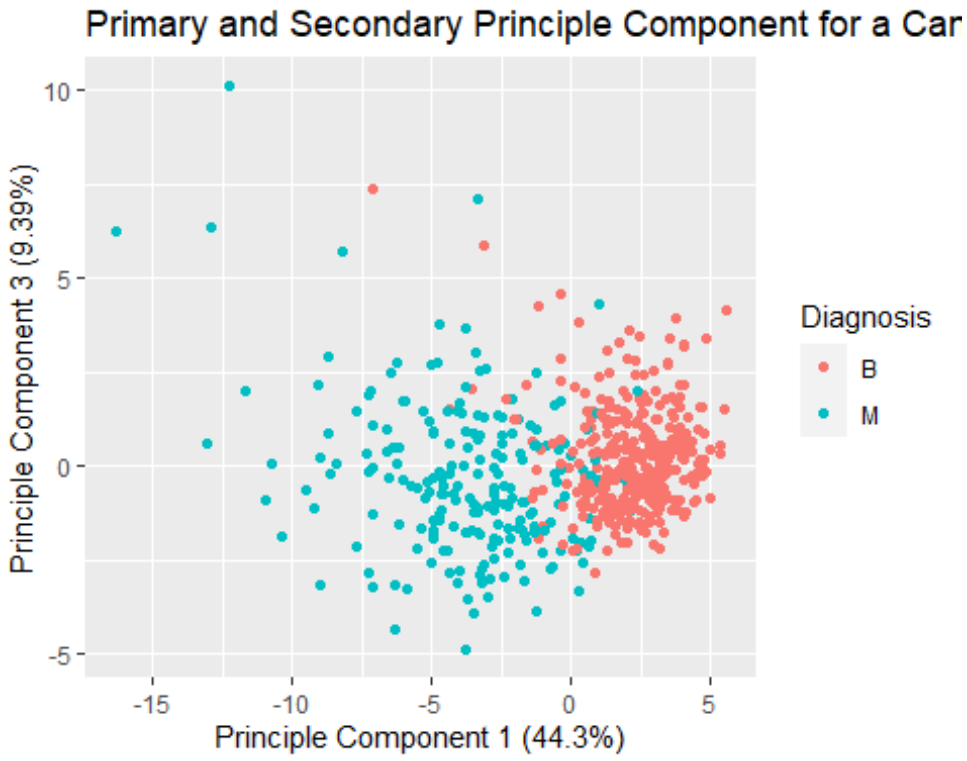
```
# Create a data.frame for ggplot
df <- as.data.frame(wisc.pr$x)
df$diagnosis <- diagnosis


# Make sure ggplot2 package is loaded
library(ggplot2)


# Plot data
ggplot(df, aes(PC1, PC2, col=diagnosis)) +
  geom_point() +
  labs(title = "Primary and Secondary Principle Component for a Cancer
Dataset", x = "Principle Component 1 (44.3%)", y = "Principle Component 2
(19.0%)", col = "Diagnosis")
```



Question 8

We can also make a plot for other principle components.

```
# N.B. Make sure the previous code chunk has been run, so that the necessary
libraries are loaded and objects created

# Plot data
ggplot(df, aes(PC1, PC3, col=diagnosis)) +
  geom_point() +
  labs(title = "Primary and Secondary Principle Component for a Cancer
Dataset", x = "Principle Component 1 (44.3%)", y = "Principle Component 3
(9.39%)", col = "Diagnosis")
```



This plot is not as informative as the previous one, but a relatively clear separation between the two groups is still visible.

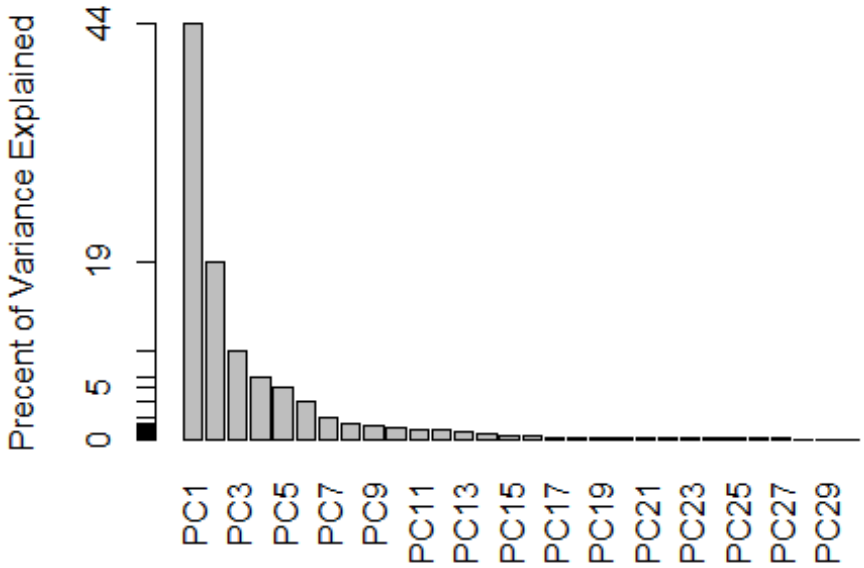
Variance Explained

To better evaluate the PCA, it is a good idea to investigate the amount of variance each principle component captures. This can be caculated by using the square of the standard deviations provided by `prcomp()`.

```
# 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

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

This data can then be plotted for easier interpretation.

```
# Scree plot with a data driven y-axis for easier interpretation
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 )
```




```
# TRY GET THIS INTO A GGPLOT FORMAT!!!!
# Work with this code already given
## ggplot based graph
#install.packages("factoextra")
#library(factoextra)
#fviz_eig(wisc.pr, addlabels = TRUE)
```

Question 9

It is also possible to see how important each variable is to each principle component. These loading values are saved in the `prcomp()` output in the rotations output.

```
# Print the loading values for the first principle component
wisc.pr$rotation[,1]
```

```
##             radius_mean            texture_mean          perimeter_mean
##             -0.21890244             -0.10372458             -0.22753729
##               area_mean         smoothness_mean        compactness_mean
##             -0.22099499             -0.14258969             -0.23928535
##          concavity_mean     concave.points_mean           symmetry_mean
##             -0.25840048             -0.26085376             -0.13816696
##  fractal_dimension_mean               radius_se              texture_se
##             -0.06436335             -0.20597878             -0.01742803
##            perimeter_se                 area_se           smoothness_se
##             -0.21132592             -0.20286964             -0.01453145
##          compactness_se            concavity_se       concave.points_se
```

```
##             -0.17039345             -0.15358979             -0.18341740
##               symmetry_se    fractal_dimension_se            radius_worst
##             -0.04249842             -0.10256832             -0.22799663
##           texture_worst         perimeter_worst              area_worst
##             -0.10446933             -0.23663968             -0.22487053
##         smoothness_worst       compactness_worst         concavity_worst
##             -0.12795256             -0.21009588             -0.22876753
##     concave.points_worst          symmetry_worst fractal_dimension_worst
##             -0.25088597             -0.12290456             -0.13178394

# Find concave.points_mean specifically
wisc.pr$rotation["concave.points_mean",1]

## [1] -0.2608538
```

These values indicate the effect of each variable in that PC, so concave.points_mean leads to 0.261 (to 3 s.f.) magnitude and negative direction changes along PC1.

Question 10

This can be answered as before in question 5.

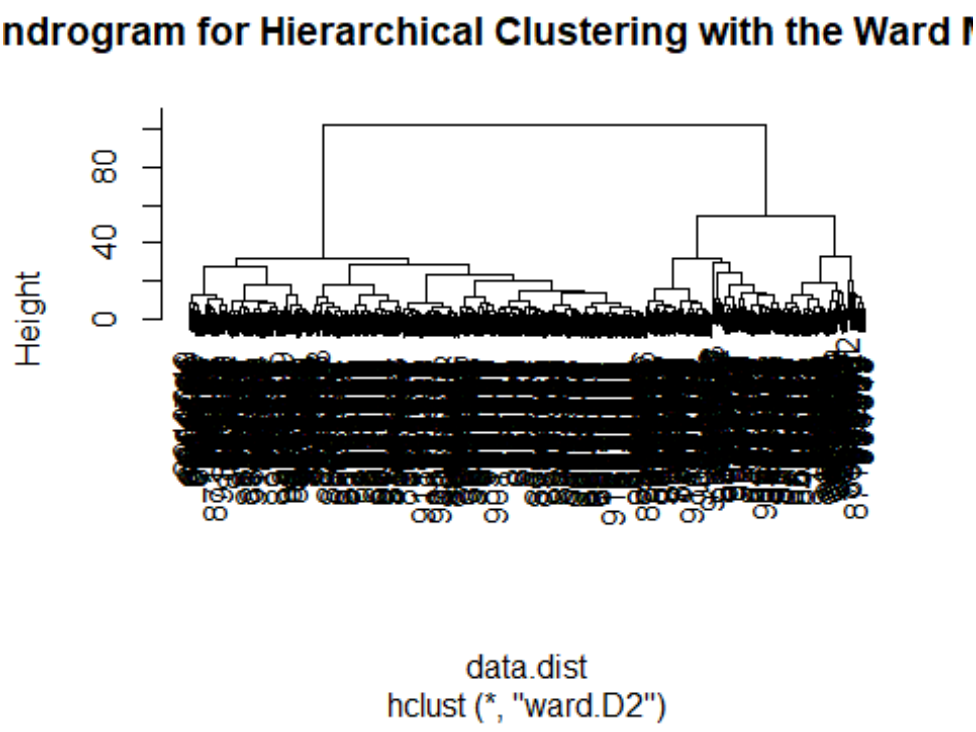
```
# Note that the Q5 code chunk must have been initialised for this chunk to
run correctly

#create counter objects
count3 <- 0
x3 <- 0

while(x3 < 0.8){
  count3 <- count3 + 1
  x3 <- x3 + summ$importance[2, count3]
}

print(paste("The first", count3, "principle components account for", x3*100,
"% of the original variance."))

## [1] "The first 5 principle components account for 84.734 % of the original
variance."
```

This reveals that the first five principle components are required to account for at least 80% of the original variance.

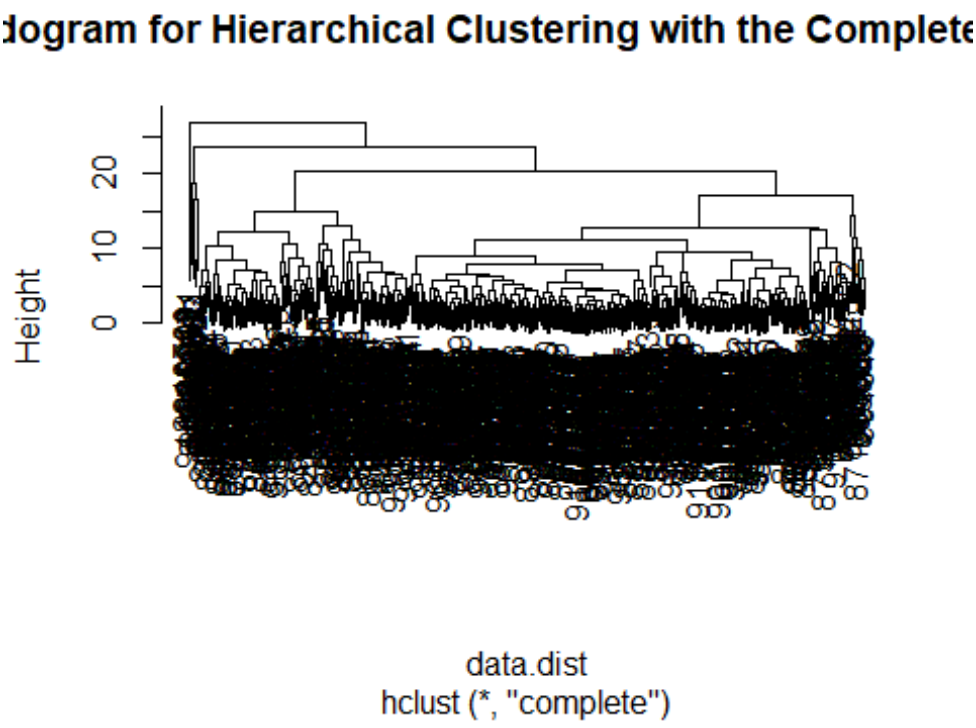
Question 11

Instead of PCA we could also use hierarchical clustering. As we will want to compare the results of different methods it is important to scale the data first again.

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

# hclust() requires distances as an input, so calculate these
data.dist <- dist(data.scaled)

# Perform hierarchical clustering
wisc.hclust.ward <- hclust(data.dist, method="ward.D2") #the method you use can have a big effect on the results, I choose to use the one recommended later in this tutorial

# Cluster with a different hclust method
wisc.hclust.comp <- hclust(data.dist, method="complete")

# Plot the results
plot(wisc.hclust.ward, main = "Dendrogram for Hierarchical Clustering with the Ward Method")
```

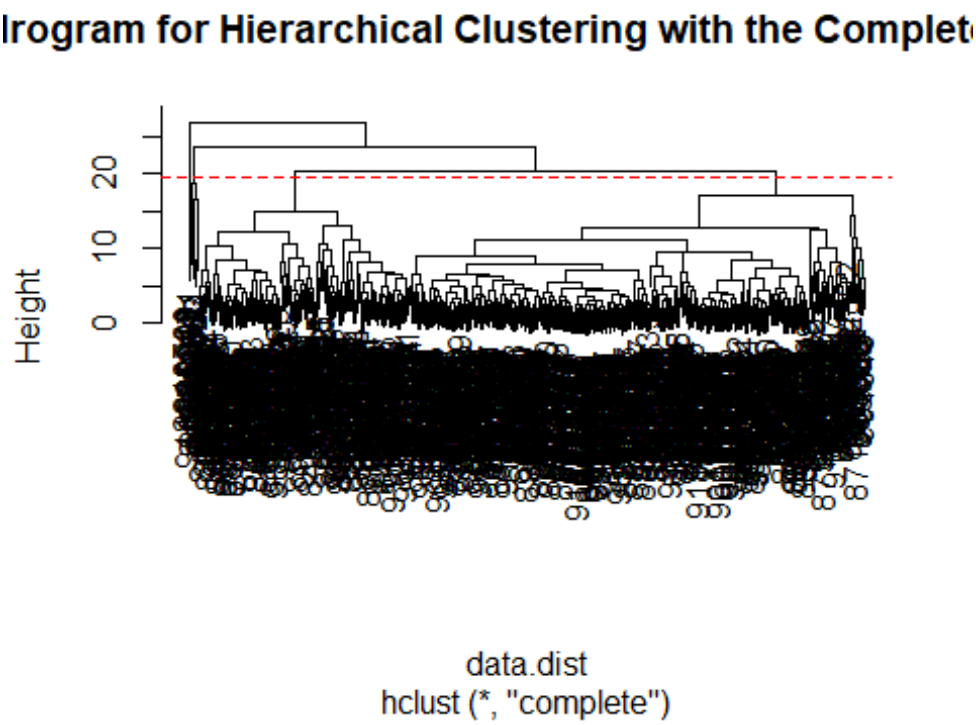



```r
plot(wisc.hclust.comp, main = "Dendogram for Hierarchical Clustering with the Complete Method")
```



To divide the data into a specific number of clusters (in this case 4) one can add a horizontal line to the plots as follows:

```
# Plot the dendrogram with a horizontal cut line
plot(wisc.hclust.comp, main = "Dendrogram for Hierarchical Clustering with
the Complete Method")
abline(h = 19.5, col="red", lty=2)
```



In a previous version of R the dndextend package had a function for finding the height at which to cut given that you wanted k clusters. However, this package is no longer available for the current version of R, so the height of the horizontal line was found by estimation and trial and error.

Question 12

We can now test how successful the hierarchical clustering was. I will uste the `wisc.hclust.comp` as it was the 'complete' method that was used in the tutorial for this section. First the data needs to be split into clusters, then `table()` can be used to compare the results.

```
# Separate into clusters
wisc.hclust.clusters <- cutree(wisc.hclust.comp, k = 4)

# Compare results to 'true answers' with table()
table(wisc.hclust.clusters, diagnosis)
```

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

Clearly, cluster 3 corresponds relatively well to benign and cluster 1 to malignant cells. We can also check if other numbers of clusters better describe the data.

```
for(i in 2:10){
  # Separate into clusters
  wisc.hclust.cluster <- cutree(wisc.hclust.comp, k = i)

# Compare results to 'true answers' with table()
  print(paste("Comparison of true diagnosis and heirarchical clustering
with", i, "clusters."))
  print(table(wisc.hclust.cluster, diagnosis))

}
```

```
## [1] "Comparison of true diagnosis and heirarchical clustering with 2
clusters."
##                     diagnosis
## wisc.hclust.cluster   B   M
##                   1 357 210
##                   2   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 3
clusters."
##                     diagnosis
## wisc.hclust.cluster   B   M
##                   1 355 205
##                   2   2   5
##                   3   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 4
clusters."
##                     diagnosis
## wisc.hclust.cluster   B   M
##                   1  12 165
##                   2   2   5
##                   3 343  40
##                   4   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 5
clusters."
##                     diagnosis
## wisc.hclust.cluster   B   M
##                   1  12 165
##                   2   0   5
##                   3 343  40
##                   4   2   0
##                   5   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 6
clusters."
##                     diagnosis
## wisc.hclust.cluster   B   M
##                   1  12 165
##                   2   0   5
```

```
##                   3 331  39
##                   4   2   0
##                   5  12   1
##                   6   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 7
clusters."
##                    diagnosis
## wisc.hclust.cluster   B   M
##                   1  12 165
##                   2   0   3
##                   3 331  39
##                   4   2   0
##                   5  12   1
##                   6   0   2
##                   7   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 8
clusters."
##                    diagnosis
## wisc.hclust.cluster   B   M
##                   1  12  86
##                   2   0  79
##                   3   0   3
##                   4 331  39
##                   5   2   0
##                   6  12   1
##                   7   0   2
##                   8   0   2
## [1] "Comparison of true diagnosis and heirarchical clustering with 9
clusters."
##                    diagnosis
## wisc.hclust.cluster   B   M
##                   1  12  86
##                   2   0  79
##                   3   0   3
##                   4 331  39
##                   5   2   0
##                   6  12   0
##                   7   0   2
##                   8   0   2
##                   9   0   1
## [1] "Comparison of true diagnosis and heirarchical clustering with 10
clusters."
##                    diagnosis
## wisc.hclust.cluster   B   M
##                  1   12  86
##                  2    0  59
##                  3    0   3
##                  4  331  39
##                  5    0  20
##                  6    2   0
```

```
##                   7   12   0
##                   8    0   2
##                   9    0   2
##                   10   0   1
```

Surprisingly, two clusters is a bad approximation, it requires at least four clusters to observe the expected separation into benign and malignant.

Question 13

Given the results from question 13 it might be worth investigating other methods of clustering.

```
for(i in c("ward.D2", "single", "average", "complete")){
  # Perform hierarchical clustering
  wisc.hclust.i <- hclust(data.dist, method= i)

  # Separate into clusters
  wisc.hclust.clusters <- cutree(wisc.hclust.i, k = 4)

  # Compare results to 'true answers' with table()
  print(paste("Table comparing true diagnoisis to clustering method", i,
"diagnosis."))
  print(table(wisc.hclust.clusters, diagnosis))
}
```

```
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1   0 115
##                    2   6  48
##                    3 337  48
##                    4  14   1
## [1] "Table comparing true diagnoisis to clustering method single
diagnosis."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1 356 209
##                    2   1   0
##                    3   0   2
##                    4   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 209
##                    2   2   0
##                    3   0   1
```

```
##                     4   0   2
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1  12 165
##                     2   2   5
##                     3 343  40
##                     4   0   2
```

This shows that when k = 4 the ward.D2 and complete methods are both reasonable (though not perfect) methods, while the single and average are not. However, it is possible that at other values of k they might be.

```
# Put the first loop in a second loop
for(j in 2:10){

  print(paste("Considering", j, "clusters."))

  for(i in c("ward.D2", "single", "average", "complete")){
    # Perform hierarchical clustering
    wisc.hclust.i <- hclust(data.dist, method= i)

    # Separate into clusters
    wisc.hclust.clusters <- cutree(wisc.hclust.i, k = j)

    # Compare results to 'true answers' with table()
    print(paste("Table comparing true diagnoisis to clustering method", i,
"diagnosis when split into", j, "clusters."))
    print(table(wisc.hclust.clusters, diagnosis))
  }

  print("/n")
}
```

```
## [1] "Considering 2 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 2 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1  20 164
##                     2 337  48
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 2 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1 357 210
##                     2   0   2
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 2 clusters."
```

```
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 357 209
##                    2   0   3
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 2 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 357 210
##                    2   0   2
## [1] "/n"
## [1] "Considering 3 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 3 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1   0 115
##                    2  20  49
##                    3 337  48
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 3 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 356 210
##                    2   1   0
##                    3   0   2
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 3 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 209
##                    2   2   0
##                    3   0   3
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 3 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 205
##                    2   2   5
##                    3   0   2
## [1] "/n"
## [1] "Considering 4 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 4 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1   0 115
##                    2   6  48
##                    3 337  48
##                    4  14   1
```

```
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 4 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 356 209
##                    2   1   0
##                    3   0   2
##                    4   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 4 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 209
##                    2   2   0
##                    3   0   1
##                    4   0   2
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 4 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1  12 165
##                    2   2   5
##                    3 343  40
##                    4   0   2
## [1] "/n"
## [1] "Considering 5 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 5 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  59
##                    2   0  56
##                    3   6  48
##                    4 337  48
##                    5  14   1
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 5 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 356 209
##                    2   1   0
##                    3   0   1
##                    4   0   1
##                    5   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 5 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 208
##                    2   2   0
```

```
##                    3   0   1
##                    4   0   2
##                    5   0   1
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 5 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1  12 165
##                    2   0   5
##                    3 343  40
##                    4   2   0
##                    5   0   2
## [1] "/n"
## [1] "Considering 6 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 6 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  59
##                    2   0  56
##                    3   6  48
##                    4 235  46
##                    5 102   2
##                    6  14   1
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 6 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 356 208
##                    2   0   1
##                    3   1   0
##                    4   0   1
##                    5   0   1
##                    6   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 6 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 202
##                    2   0   6
##                    3   2   0
##                    4   0   1
##                    5   0   2
##                    6   0   1
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 6 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1  12 165
##                    2   0   5
```

```
##                     3 331  39
##                     4   2   0
##                     5  12   1
##                     6   0   2
## [1] "/n"
## [1] "Considering 7 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 7 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1   0  57
##                     2   0  56
##                     3   6  48
##                     4 235  46
##                     5 102   2
##                     6  14   1
##                     7   0   2
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 7 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1 356 207
##                     2   0   1
##                     3   0   1
##                     4   1   0
##                     5   0   1
##                     6   0   1
##                     7   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 7 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1   0  40
##                     2 355 162
##                     3   0   6
##                     4   2   0
##                     5   0   1
##                     6   0   2
##                     7   0   1
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 7 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                     1  12 165
##                     2   0   3
##                     3 331  39
##                     4   2   0
##                     5  12   1
##                     6   0   2
##                     7   0   2
```

```
## [1] "/n"
## [1] "Considering 8 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 8 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  57
##                    2   0  56
##                    3   6  48
##                    4  34  41
##                    5 201   5
##                    6 102   2
##                    7  14   1
##                    8   0   2
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 8 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 207
##                    2   0   1
##                    3   1   0
##                    4   0   1
##                    5   1   0
##                    6   0   1
##                    7   0   1
##                    8   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 8 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  40
##                    2 355 162
##                    3   0   6
##                    4   1   0
##                    5   0   1
##                    6   1   0
##                    7   0   2
##                    8   0   1
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 8 clusters."
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1  12  86
##                    2   0  79
##                    3   0   3
##                    4 331  39
##                    5   2   0
##                    6  12   1
##                    7   0   2
##                    8   0   2
```

```
## [1] "/n"
## [1] "Considering 9 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 9 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  57
##                    2   0  56
##                    3   6  48
##                    4  34  41
##                    5 201   5
##                    6  69   2
##                    7  33   0
##                    8  14   1
##                    9   0   2
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 9 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 206
##                    2   0   1
##                    3   1   0
##                    4   0   1
##                    5   0   1
##                    6   1   0
##                    7   0   1
##                    8   0   1
##                    9   0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 9 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  40
##                    2 353 162
##                    3   0   6
##                    4   1   0
##                    5   0   1
##                    6   1   0
##                    7   2   0
##                    8   0   2
##                    9   0   1
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 9 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1  12  86
##                    2   0  79
##                    3   0   3
##                    4 331  39
##                    5   2   0
```

```
##                    6  12   0
##                    7   0   2
##                    8   0   2
##                    9   0   1
## [1] "/n"
## [1] "Considering 10 clusters."
## [1] "Table comparing true diagnoisis to clustering method ward.D2
diagnosis when split into 10 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  51
##                    2   0  56
##                    3   6  48
##                    4  34  41
##                    5   0   6
##                    6 201   5
##                    7  69   2
##                    8  33   0
##                    9  14   1
##                    10  0   2
## [1] "Table comparing true diagnoisis to clustering method single diagnosis
when split into 10 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 205
##                    2   0   1
##                    3   1   0
##                    4   0   1
##                    5   0   1
##                    6   1   0
##                    7   0   1
##                    8   0   1
##                    9   0   1
##                    10  0   1
## [1] "Table comparing true diagnoisis to clustering method average
diagnosis when split into 10 clusters."
##                      diagnosis
## wisc.hclust.clusters   B   M
##                    1   0  33
##                    2 353 162
##                    3   0   6
##                    4   0   7
##                    5   1   0
##                    6   0   1
##                    7   1   0
##                    8   2   0
##                    9   0   2
##                    10  0   1
## [1] "Table comparing true diagnoisis to clustering method complete
diagnosis when split into 10 clusters."
```

```
##                     diagnosis
## wisc.hclust.clusters   B   M
##                    1  12  86
##                    2   0  59
##                    3   0   3
##                    4 331  39
##                    5   0  20
##                    6   2   0
##                    7  12   0
##                    8   0   2
##                    9   0   2
##                    10  0   1
## [1] "/n"
```

While this provides all the information, it is not particularly easy to interpret. It would be necessary to write some code to find the best result for each method separately and then output only that best result to compare with the others.

Question 14

We can also test kmeans clustering. As before the data should be scaled, additionally, as we know there are two groups, we should tell kmeans to separate the data into two clusters. Finally, it seems wise to use several repetitions, so as to find the best model, thus nstart will be set to 20.

```
# Find the best kmeans clusters
wisc.km <- kmeans(scale(wisc.data), centers= 2, nstart= 20)

# Compare the results to the true diagnosis
table(wisc.km$cluster, diagnosis)
```

```
##    diagnosis
##       B   M
##   1 343  37
##   2  14 175
```

This is pretty reasonable although there are 14 false negatives and 37 false positives. In comparison, hierarchical clustering, which required 4 clusters to reasonably approximate the data, had 14 false negatives and 47 false positives for the 'complete' method and considerably more for the 'ward.D2' method.

It is also possible to directly compare the two methods.

```
# Directly compare the kmeans and hclust complete method
table(wisc.km$cluster, wisc.hclust.clusters)
```

```
##    wisc.hclust.clusters
##       1   2   3   4   5   6   7   8   9  10
```

```
##   1  17   0   0 358   0   0   5   0   0   0
##   2  81  59   3  12  20   2   7   2   2   1
```

Question 15

Clustering on PCA results. PCA often used to identify outliers. Is generally a very useful first exploratory analysis that can then be followed up by other analyses. Text below from lab handout summarizes the situation well:

"Recall from earlier sections that the PCA model required significantly fewer features to describe 70%, 80% and 95% of the variability of the data. In addition to normalizing data and potentially avoiding over-fitting, PCA also uncorrelates the variables, sometimes improving the performance of other modeling techniques."

Thus, let us try hierarchical clustering on the PCA results, using only enough principle components to capture 90% of variance.

```
# Isolate only the first x principle components necessary to capture 90% of
variance (x=7 in this case), also, hclust requires differences, so use dist
on this subset of the data
v90 <- dist(wisc.pr$x[,1:7])

# Hierarchical clustering with PCA data
wisc.pr.hclust <- hclust(v90, method="ward.D2")
```

We can then analyse the results of this clustering.

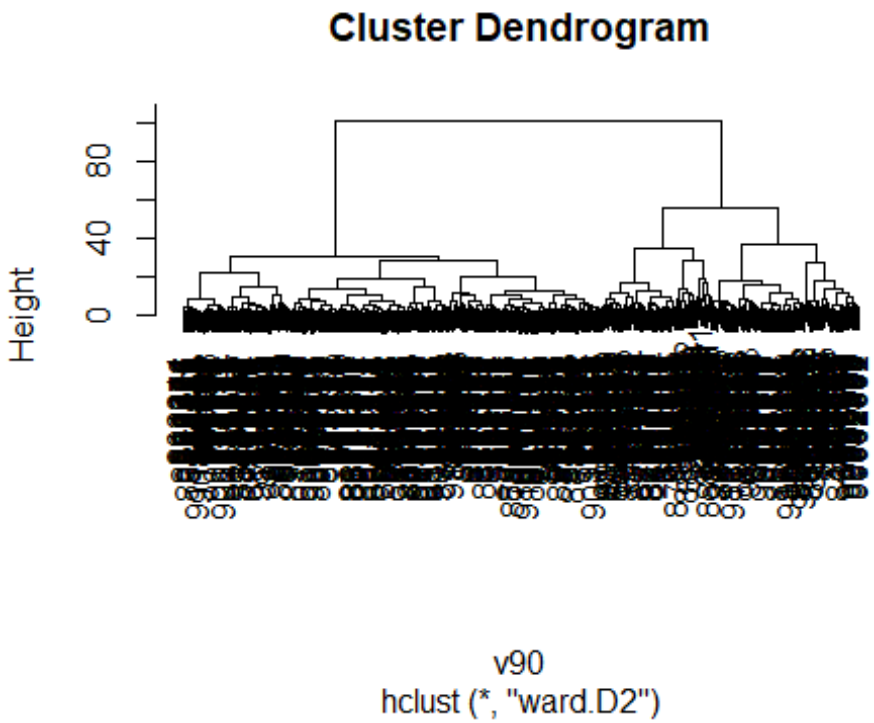
```
# Cluster into two groups
grps <- cutree(wisc.pr.hclust, k=2)

# Results
table(grps)

## grps
##   1   2
## 216 353

# Compare to 'true answers'
table(diagnosis)

## diagnosis
##   B   M
## 357 212

table(grps, diagnosis)

##     diagnosis
## grps   B   M
##    1  28 188
##    2 329  24
```

The last table above has true positives and negatives, false positives and false negatives as follows

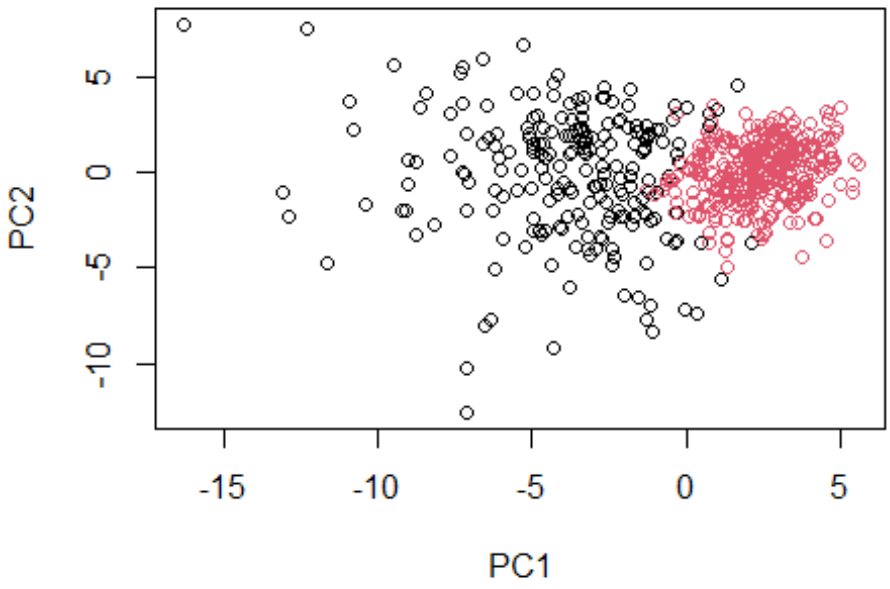
```
False Negative | True Positive
--------------------------------
True Negative  | False Positive
```

While using the PCA is therefore not perfect, it is a pretty good start. Visualizing the results is also helpful.

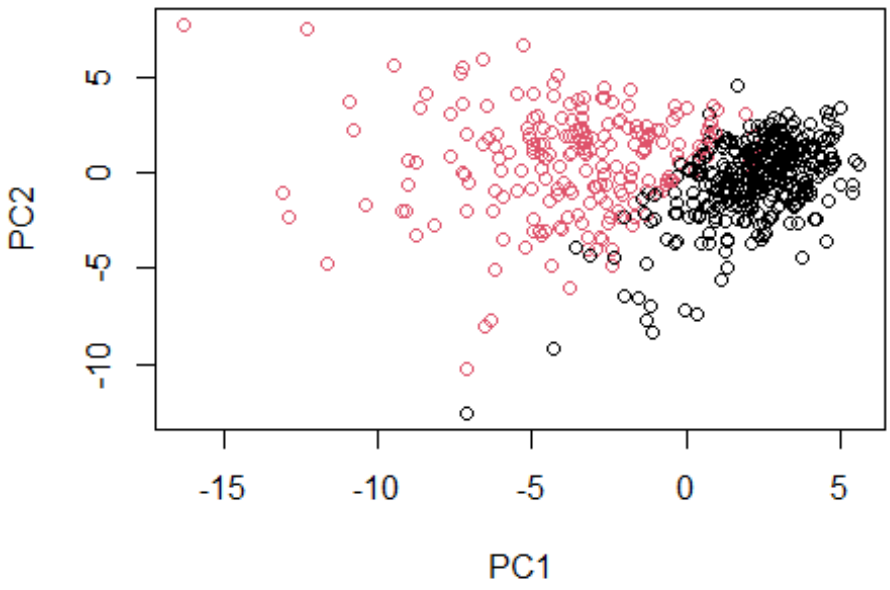
```
# Plot results as dendogram
plot(wisc.pr.hclust)
```




```
# Plot PC with hclust grouping used to define colours
plot(wisc.pr$x[,1:2], col=grps)
```

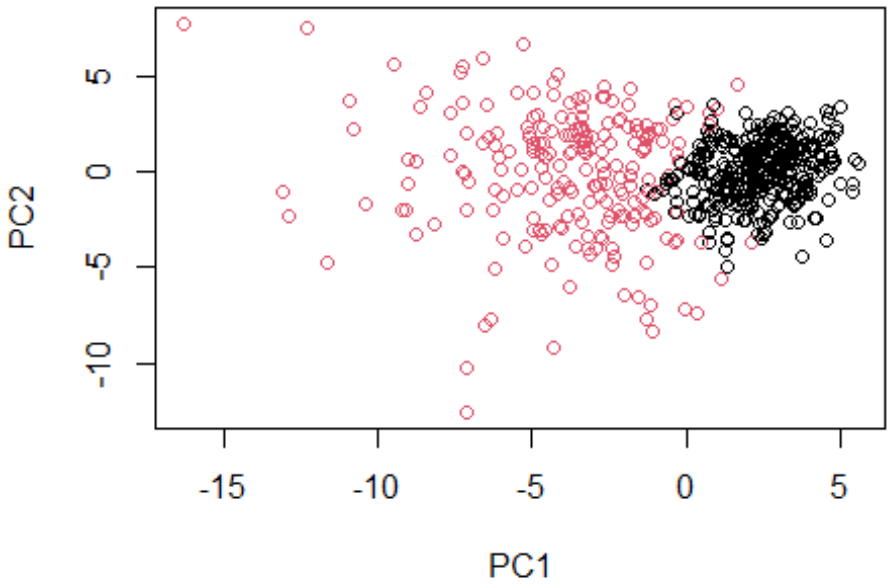



```
# Compare to colouring by diagnosis
plot(wisc.pr$x[,1:2], col=as.factor(diagnosis))
```



The dendogram is not as useful for interpretation. The following two plots appear almost identical in their colour groupings, although one is from the clustering and the other from the 'true answers', which shows that the clustering using PCA results has led to apparently satisfactory results. The one other difference between the plots is the reversal of the colouring, due to how the groups are ordered in the two objects. This can be fixed by re-ordering as below:

```
# Current ordering of grps
g <- as.factor(grps)
levels(g)

## [1] "1" "2"

# Re-order object
g <- relevel(g,2)
levels(g)

## [1] "2" "1"

# Plot using our re-ordered factor
plot(wisc.pr$x[,1:2], col=g)
```



Q16

In a similar vein to in the previous section, we can use the `table()` function to see how other results compare.

```
# Compare results of k means clustering to 'true answers'
table(wisc.km$cluster, diagnosis)
```

```
##    diagnosis
##       B   M
##   1 343  37
##   2  14 175
```

```
# Compare results of hclustering to 'true answers'
table(wisc.hclust.clusters, diagnosis)
```

```
##                     diagnosis
## wisc.hclust.clusters   B   M
##                   1   12  86
##                   2    0  59
##                   3    0   3
##                   4  331  39
##                   5    0  20
##                   6    2   0
##                   7   12   0
##                   8    0   2
##                   9    0   2
##                   10   0   1
```

These results are actually on par if not a little more accurate than the clustering after PCA (see Q15).

Q17

To determine which method is objectively best we could consider selectivity (True Positive/(True Positive + False Negative)) and sensitivity (True Negative/(True Negative + False Negative)).

```
# Put all the results into r objects
kmeans.tmp <- table(wisc.km$cluster, diagnosis)
hclust.tmp <- table(wisc.hclust.clusters, diagnosis)
pca.hclust.tmp <- table(grps, diagnosis)

# Convert data to format useful for calculations
hclust <- as.matrix(cbind(c(sum(hclust.tmp[-which.max(hclust.tmp[,1]),1]),
hclust.tmp[which.max(hclust.tmp[,1]),1]), c(hclust.tmp[
which.max(hclust.tmp[,2]) , 2], sum(hclust.tmp[-
which.max(hclust.tmp[,2]),2]))))

colnames(hclust) <- c("B", "M")

kmeans <- as.matrix(cbind(c(sum(kmeans.tmp[-which.max(kmeans.tmp[,1]),1]),
kmeans.tmp[which.max(kmeans.tmp[,1]),1]), c(kmeans.tmp[
which.max(kmeans.tmp[,2]) , 2], sum(kmeans.tmp[-
```

```
which.max(kmeans.tmp[,2]),2])))

colnames(kmeans) <- c("B", "M")

pca.hclust <- as.matrix(cbind(c(sum(pca.hclust.tmp[-
which.max(pca.hclust.tmp[,1]),1]),
pca.hclust.tmp[which.max(pca.hclust.tmp[,1]),1]), c(pca.hclust.tmp[
which.max(pca.hclust.tmp[,2]) , 2], sum(pca.hclust.tmp[-
which.max(pca.hclust.tmp[,2]),2]))))

colnames(pca.hclust) <- c("B", "M")

# Calculate selectivity and sensitivity


# For some reason the for loop is only accepting the first item of the
data.frame as i - really weird
# for(i in c(kmeans, hclust, pca.hclust)){
#   print(i)
#   print(kmeans)
#
#   # Formula to calculate selectivity
#   sel <- i[1,2]/(i[1,2] + i[1,1])
#
#   # Formula to calculate sensitivity
#   sens <- i[2,1]/(i[2,1] + i[1,1])
#
#   # Print results
#   print(paste("The", i, "method has a selectivity of", sel, "and a
sensitivity of", sens, "."))
# }

# Instead write selectivity and sensitivity functions
sel <- function(i){i[1,2]/(i[1,2] + i[1,1])}
sens <- function(i){i[2,1]/(i[2,1] + i[1,1])}

# Use this on the matrices
print(paste("The kmeans method has a selectivity of", sel(kmeans), "and a
sensitivity of", sens(kmeans), "."))

## [1] "The kmeans method has a selectivity of 0.925925925925926 and a
sensitivity of 0.96078431372549 ."

print(paste("The hclust method has a selectivity of", sel(hclust), "and a
sensitivity of", sens(hclust), "."))

## [1] "The hclust method has a selectivity of 0.767857142857143 and a
sensitivity of 0.927170868347339 ."
```

```
print(paste("The pca.hclust method has a selectivity of", sel(pca.hclust),
"and a sensitivity of", sens(pca.hclust), "."))
```

```
## [1] "The pca.hclust method has a selectivity of 0.87037037037037 and a
sensitivity of 0.92156862745098 ."
```

From this we can see that kmeans has the best selectivity, with pca.hclust second. Kmeans also has the highest sensitivity.

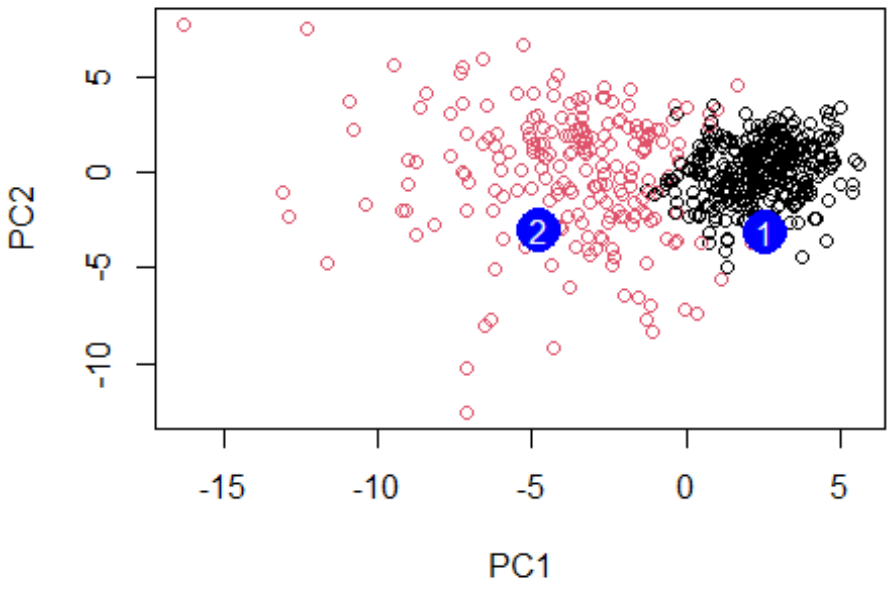
Q18

Lastly, we can use the model for prediction with some new data.

```
# Get new data
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 the new data and two points of interest
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")
```



From this we can clearly see that the two points fall in the two different clusters and that point 2 in the malignant cluster is the patient we should be more concerned for.

```
# Provide session info for reproducibility
sessionInfo()

## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United Kingdom.1252
## [2] LC_CTYPE=English_United Kingdom.1252
## [3] LC_MONETARY=English_United Kingdom.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United Kingdom.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] ggplot2_3.3.5
##
## loaded via a namespace (and not attached):
##  [1] pillar_1.7.0     compiler_4.1.2   highr_0.9        tools_4.1.2
```

```
##  [5] digest_0.6.29    evaluate_0.14    lifecycle_1.0.1  tibble_3.1.6
##  [9] gtable_0.3.0     pkgconfig_2.0.3  rlang_1.0.0      cli_3.1.1
## [13] yaml_2.2.2       xfun_0.29        fastmap_1.1.0    withr_2.4.3
## [17] stringr_1.4.0    dplyr_1.0.7      knitr_1.37       generics_0.1.2
## [21] vctrs_0.3.8      grid_4.1.2       tidyselect_1.1.1 glue_1.6.1
## [25] R6_2.5.1         fansi_1.0.2      rmarkdown_2.11   purrr_0.3.4
## [29] farver_2.1.0     magrittr_2.0.2   scales_1.1.1     ellipsis_0.3.2
## [33] htmltools_0.5.2  colorspace_2.0-2 labeling_0.4.2   utf8_1.2.2
## [37] stringi_1.7.6    munsell_0.5.0    crayon_1.4.2
```