Class 8: PCA Mini Project

Carly Chang (A16843962)

PCA: prcomp(x) Clustering: kmeans(x) hclust(dist(x))

Today we will do a complete analysis of some breast cancer biopsy data but first let's revisit the main PCA function in R `prcomp()` and see what `scale=TRUE/FALSE` does.

```
mtcars
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
Chrysler Imperial	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
Fiat 128	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
Toyota Corolla	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1
Toyota Corona	21.5	4	120.1	97	3.70	2.465	20.01	1	0	3	1
Dodge Challenger	15.5	8	318.0	150	2.76	3.520	16.87	0	0	3	2
AMC Javelin	15.2	8	304.0	150	3.15	3.435	17.30	0	0	3	2
Camaro Z28	13.3	8	350.0	245	3.73	3.840	15.41	0	0	3	4

```
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

Find the mean value per column of this dataset?

```
apply(mtcars, 2, mean) #apply function mean on the columns (denoted by 2) of mtcars matrix
```

```
      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
```

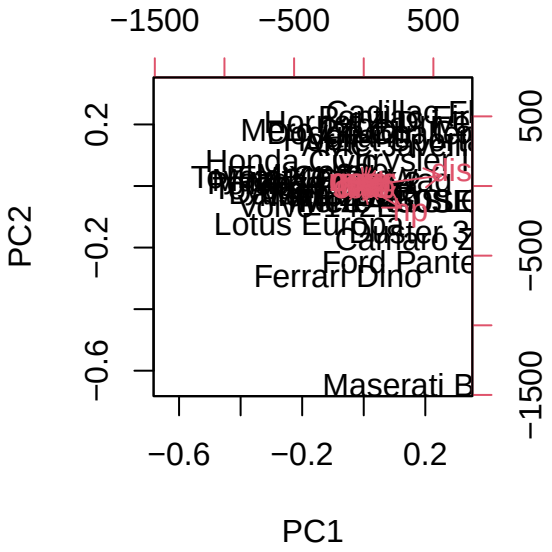
```
apply(mtcars, 2, sd)
```

```
       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
```

It is clear the "disp" and "hp" have the highest mean values and the highest standard deviation here. They will likely dominate any analysis I do on this dataset. Let's see:

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

```
biplot(pc.noscale)
```



The biplot only shows 2 red arrow - disp and hp are the only ones that contribute to the PCA.

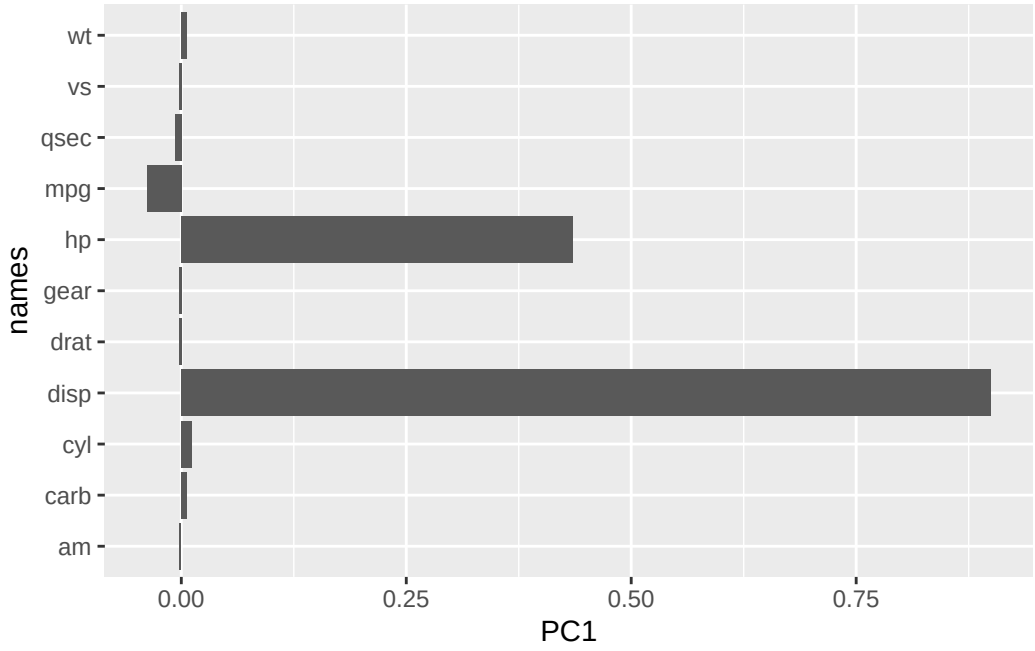
```
pc.noscale$rotation[,1] #rotation gives the PCA for each column (variable). We look at PC1.
```

```
         mpg          cyl         disp           hp         drat           wt
-0.038118199  0.012035150  0.899568146  0.434784387 -0.002660077  0.006239405
        qsec           vs           am         gear         carb
-0.006671270 -0.002729474 -0.001962644 -0.002604768  0.005766010
```

disp and hp has large absolute values, meaning they contribute the most to the PCA

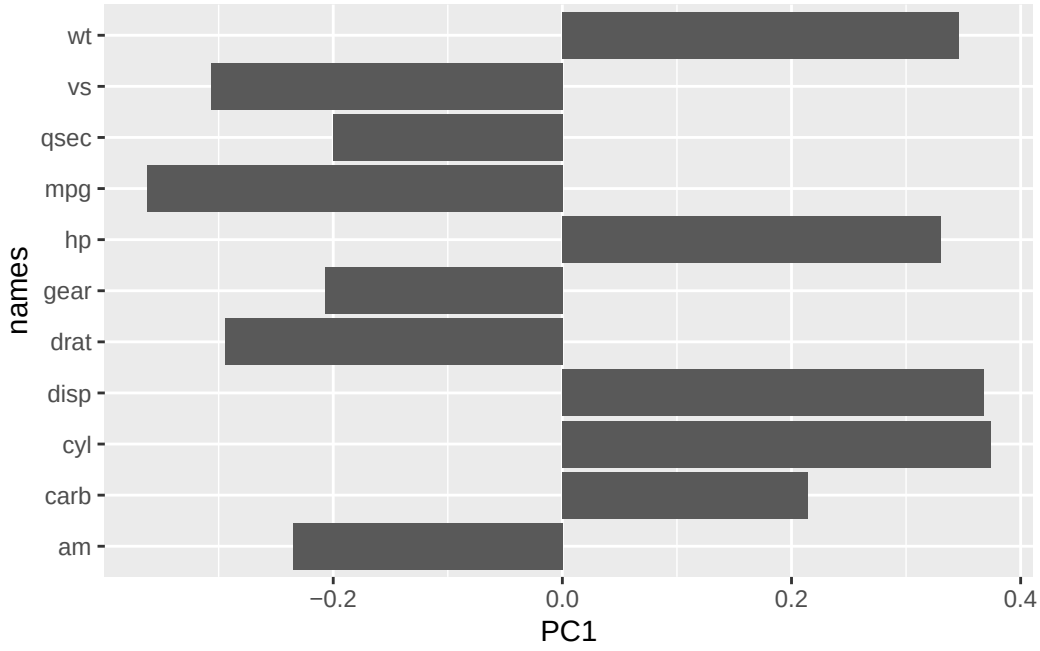
Plot the loadings:

```
library(ggplot2)

r1 <- as.data.frame(pc.noscale$rotation)
r1$names <- rownames(pc.noscale$rotation)

ggplot(r1) +
  aes(PC1, names) +
  geom_col()
```

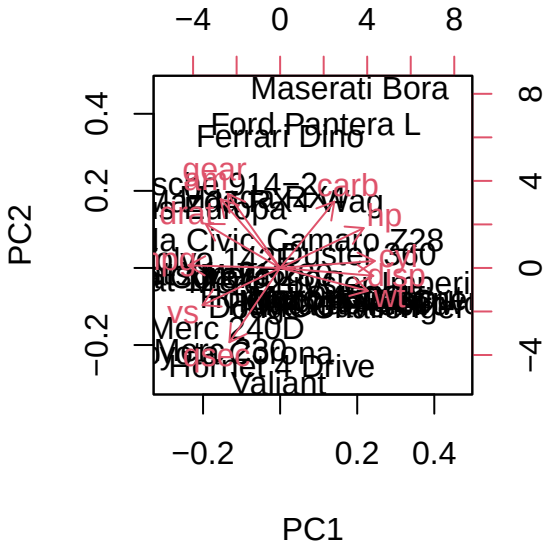


Scaling gives a better distribution of loadings because it makes all the variances set to 1:

```
r2 <- as.data.frame(pc.scale$rotation)
r2$names <- rownames(pc.scale$rotation)

ggplot(r2) +
  aes(PC1, names) +
  geom_col()
```




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



The biplot of the scaled PCA shows more red arrows - all of the variables contribute.

Take-home: Generally we always to to set `scale=TRUE` when we do this typ eof analysis to avoid our analysis being dominated by individual variables with the largest variance just due ot their unit of measurment.

FNA breast cancer data

Load the data into R. Download WisconsinCancer.csv into Class08 folder.

```
wisc.df <- read.csv("WisconsinCancer.csv", row.names=1) #sets the first column as row names,
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 are in this dataset?

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

```
[1] 569
```

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

There are two ways to do this.

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

```
[1] 212
```

The `table()` function is super useful here:

```
table(wisc.df$diagnosis) #gives summary of number of each diagnosis type
```

```
  B   M
357 212
```

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

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

```
[1] 31
```

```
colnames(wisc.df) #give column names
```

```
 [1] "diagnosis"               "radius_mean"
 [3] "texture_mean"            "perimeter_mean"
 [5] "area_mean"               "smoothness_mean"
 [7] "compactness_mean"        "concavity_mean"
 [9] "concave.points_mean"     "symmetry_mean"
[11] "fractal_dimension_mean"  "radius_se"
[13] "texture_se"              "perimeter_se"
[15] "area_se"                 "smoothness_se"
[17] "compactness_se"          "concavity_se"
[19] "concave.points_se"       "symmetry_se"
[21] "fractal_dimension_se"    "radius_worst"
[23] "texture_worst"           "perimeter_worst"
[25] "area_worst"              "smoothness_worst"
[27] "compactness_worst"       "concavity_worst"
[29] "concave.points_worst"    "symmetry_worst"
[31] "fractal_dimension_worst"
```

A useful function for this is `grep()`, which finds patterns in a vector.

```
grep("_mean", colnames(wisc.df)) #finds the element # in column names of wisc.df that has "_m
```

```
 [1]  2  3  4  5  6  7  8  9 10 11
```

```
length(grep("_mean", colnames(wisc.df))) #gives length, aka how many
```

```
[1] 10
```

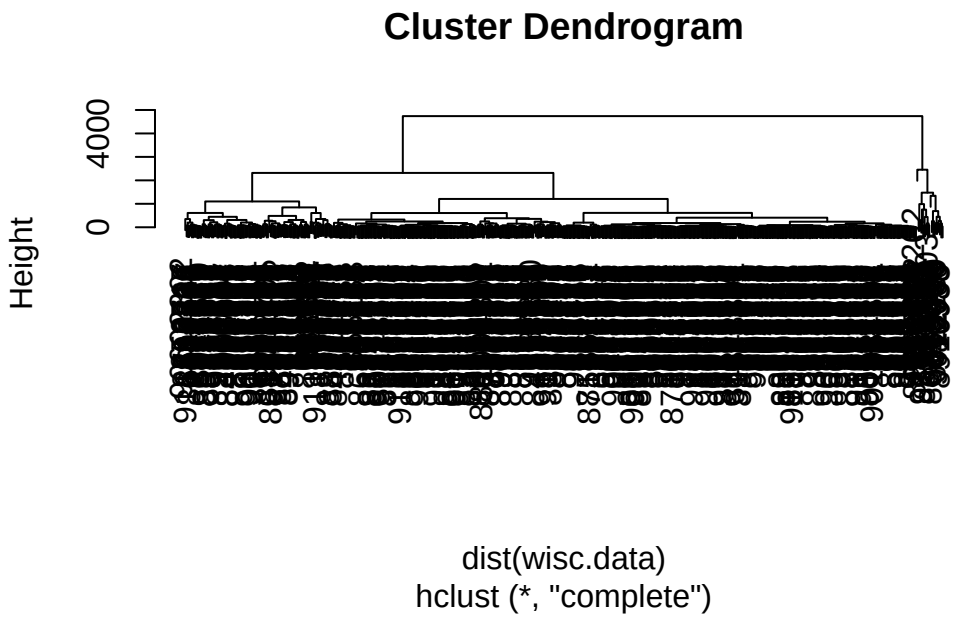
Before we go any further, we need to exclude the diagnosis column from any future analysis - this tells us whether a sample to cancer or non-cancer. We will store the diagnosis column in `diagnosis` as a factor - a list of variables that can be of multiple types

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

```
wisc.data <- wisc.df[,-1] #everything but the first column
```

Let's see if we can cluster the `wisc.data` to find some structure in the dataset.

```
hc <- hclust(dist(wisc.data)) #Must plot dist in hclust for dendrograms
plot(hc)
```



Principal Component Analysis (PCA)

```
wisc.pr <- prcomp(wisc.data, scale=T )
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)?

0.4427

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

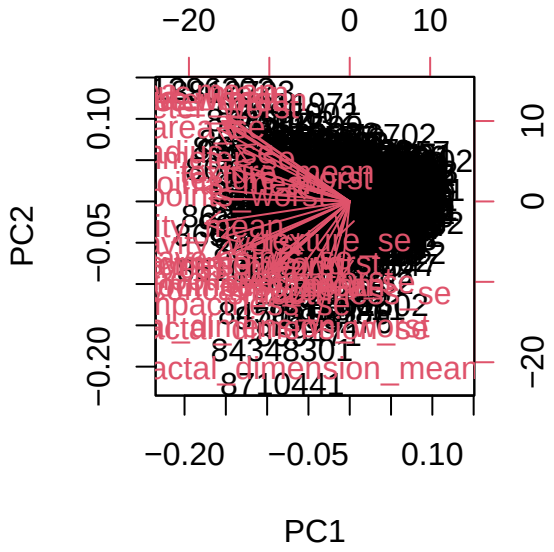
3

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

7

Interpreting PCA results

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



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

This biplot sucks! It is not easy to understand because there are too many data points. Biplots are meant to plot smaller data sets, which is not what we have. We need to build our own PCA score plot of PC1 vs PC2

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

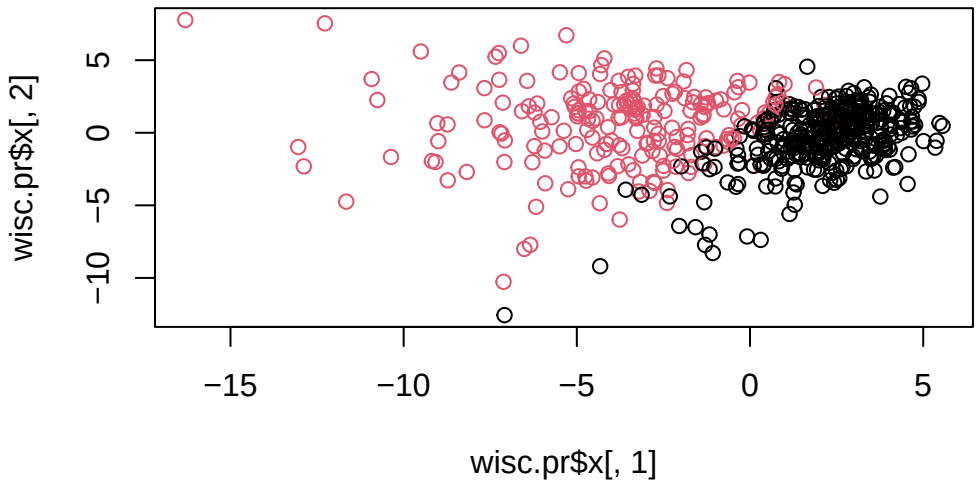
```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

```
head(wisc.pr$x) #gives wisc.pr PCAs for each patient ID (row)
```

```
              PC1        PC2        PC3       PC4        PC5         PC6
842302   -9.184755  -1.946870 -1.1221788 3.6305364  1.1940595  1.41018364
842517   -2.385703   3.764859 -0.5288274 1.1172808 -0.6212284  0.02863116
84300903 -5.728855   1.074229 -0.5512625 0.9112808  0.1769302  0.54097615
84348301 -7.116691 -10.266556 -3.2299475 0.1524129  2.9582754  3.05073750
84358402 -3.931842   1.946359  1.3885450 2.9380542 -0.5462667 -1.22541641
843786   -2.378155  -3.946456 -2.9322967 0.9402096  1.0551135 -0.45064213
                PC7         PC8         PC9        PC10        PC11        PC12
842302    2.15747152  0.39805698 -0.15698023 -0.8766305 -0.2627243 -0.8582593
842517    0.01334635 -0.24077660 -0.71127897  1.1060218 -0.8124048  0.1577838
84300903 -0.66757908 -0.09728813  0.02404449  0.4538760  0.6050715  0.1242777
84348301  1.42865363 -1.05863376 -1.40420412 -1.1159933  1.1505012  1.0104267
84358402 -0.93538950 -0.63581661 -0.26357355  0.3773724 -0.6507870 -0.1104183
843786    0.49001396  0.16529843 -0.13335576 -0.5299649 -0.1096698  0.0813699
               PC13         PC14         PC15        PC16        PC17
842302    0.10329677 -0.690196797  0.601264078  0.74446075 -0.26523740
842517   -0.94269981 -0.652900844 -0.008966977 -0.64823831 -0.01719707
84300903 -0.41026561  0.016665095 -0.482994760  0.32482472  0.19075064
84348301 -0.93245070 -0.486988399  0.168699395  0.05132509  0.48220960
84358402  0.38760691 -0.538706543 -0.310046684 -0.15247165  0.13302526
843786   -0.02625135  0.003133944 -0.178447576 -0.01270566  0.19671335
               PC18       PC19        PC20         PC21        PC22
842302   -0.54907956  0.1336499  0.34526111  0.096430045 -0.06878939
842517    0.31801756 -0.2473470 -0.11403274 -0.077259494  0.09449530
84300903 -0.08789759 -0.3922812 -0.20435242  0.310793246  0.06025601
84348301 -0.03584323 -0.0267241 -0.46432511  0.433811661  0.20308706
84358402 -0.01869779  0.4610302  0.06543782 -0.116442469  0.01763433
843786   -0.29727706 -0.1297265 -0.07117453 -0.002400178  0.10108043
               PC23         PC24         PC25         PC26        PC27
842302    0.08444429  0.175102213  0.150887294 -0.201326305 -0.25236294
842517   -0.21752666 -0.011280193  0.170360355 -0.041092627  0.18111081
84300903 -0.07422581 -0.102671419 -0.171007656  0.004731249  0.04952586
84348301 -0.12399554 -0.153294780 -0.077427574 -0.274982822  0.18330078
84358402  0.13933105  0.005327110 -0.003059371  0.039219780  0.03213957
843786    0.03344819 -0.002837749 -0.122282765 -0.030272333 -0.08438081
                PC28         PC29          PC30
842302   -0.0338846387  0.045607590  0.0471277407
842517    0.0325955021 -0.005682424  0.0018662342
84300903  0.0469844833  0.003143131 -0.0007498749
84348301  0.0424469831 -0.069233868  0.0199198881
84358402 -0.0347556386  0.005033481 -0.0211951203
843786    0.0007296587 -0.019703996 -0.0034564331
```

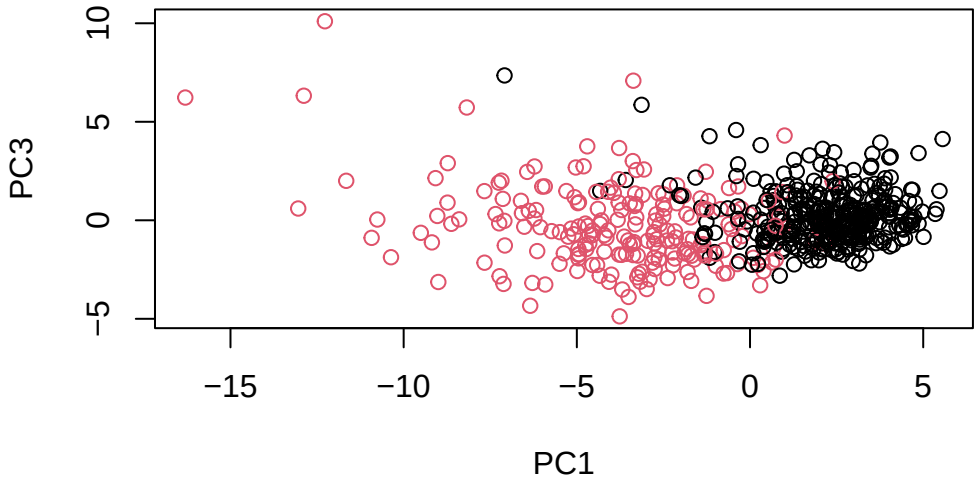
Plot PC1 vs PC2, which are the first two columns

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



Q8. Generate a similar plot for principal components 1 and 3. What do you notice about these plots?

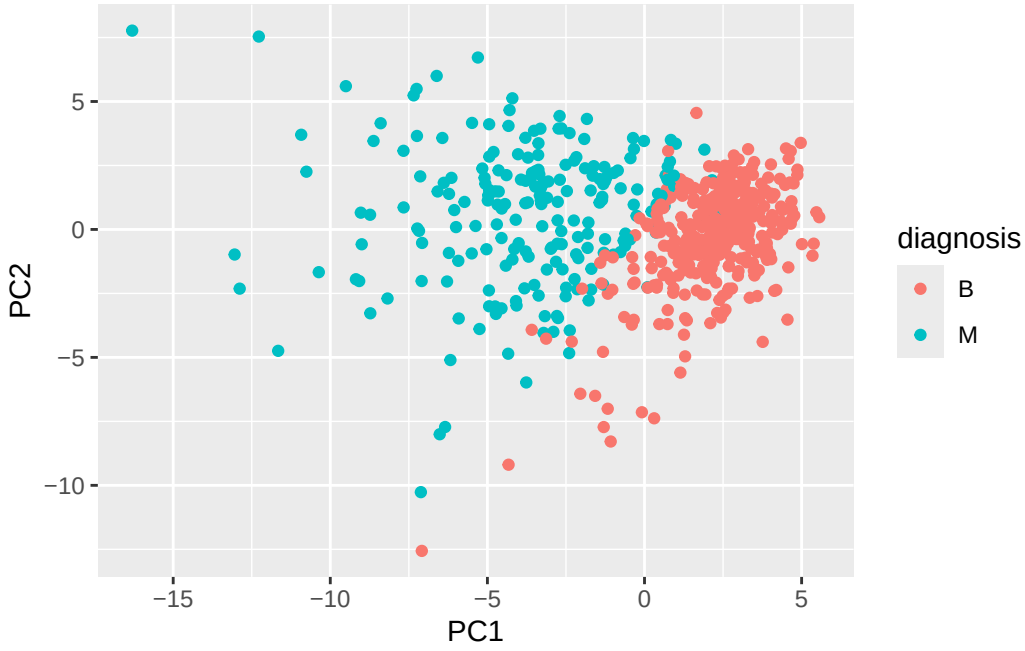
```
plot(wisc.pr$x[, 1], wisc.pr$x[,3], col=diagnosis,
     xlab = "PC1", ylab = "PC3")
```



Both PC1 vs PC2 and PC1 vs PC3 plots look relatively similar. This means that PC3 does not capture a large amount of variance in the data. This is confirmed by the summary, where PC1 captures almost 50% of the variance, PC2 19%, and PC3 only 9%.

Make a gpplot version of this score plot for PC1 vs PC2:

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

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



PCA clusters each group into one point to make it easier to read. The closer points will merge until there are distinct groups of data points that are significantly different from one another (reducing dimensionality). PCA rotates the axis so that there is the least variability in the points (ie. little variability on y-axis). These rotated axes are the PCs. Some points will have more influence on PC than others. These tend to be the points farther from the center of the axis. PC1 score = sum of (read counts*PC1). We can plot this for each PC and the PCs closer to each other are more similar.

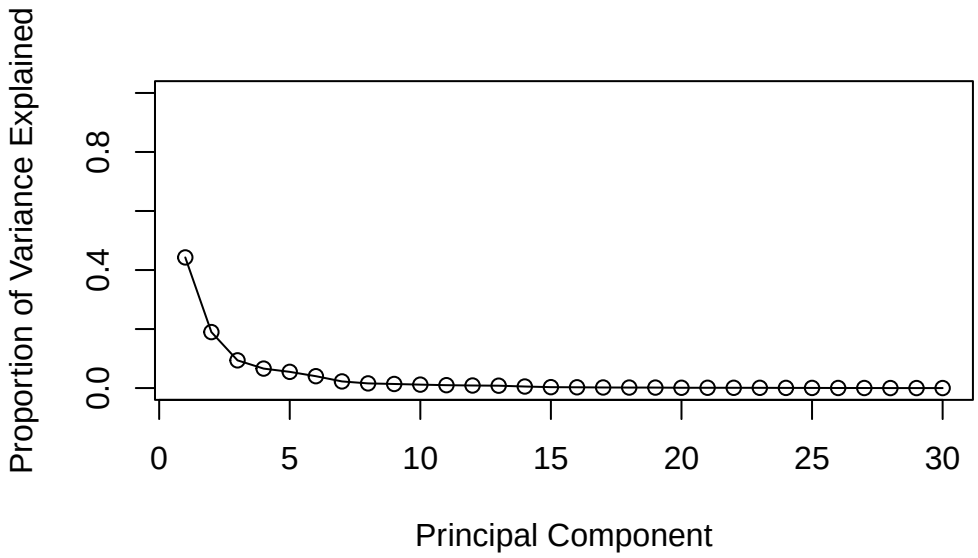
Variance Explained

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

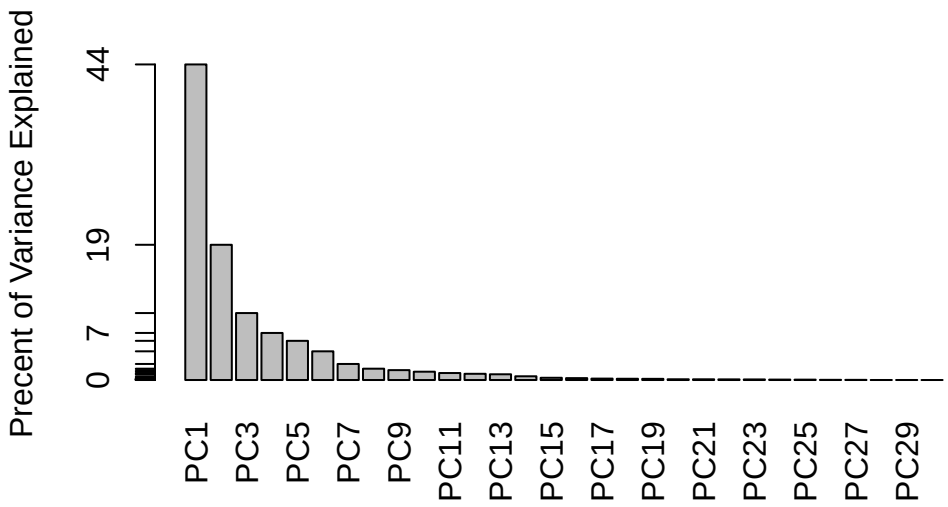
```
[1] 13.281608  5.691355  2.817949  1.980640  1.648731  1.207357
```

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

# Plot variance explained for each principal component
plot(pve, xlab = "Principal Component",
```

```
     ylab = "Proportion of Variance Explained",
     ylim = c(0, 1), type = "o")
```



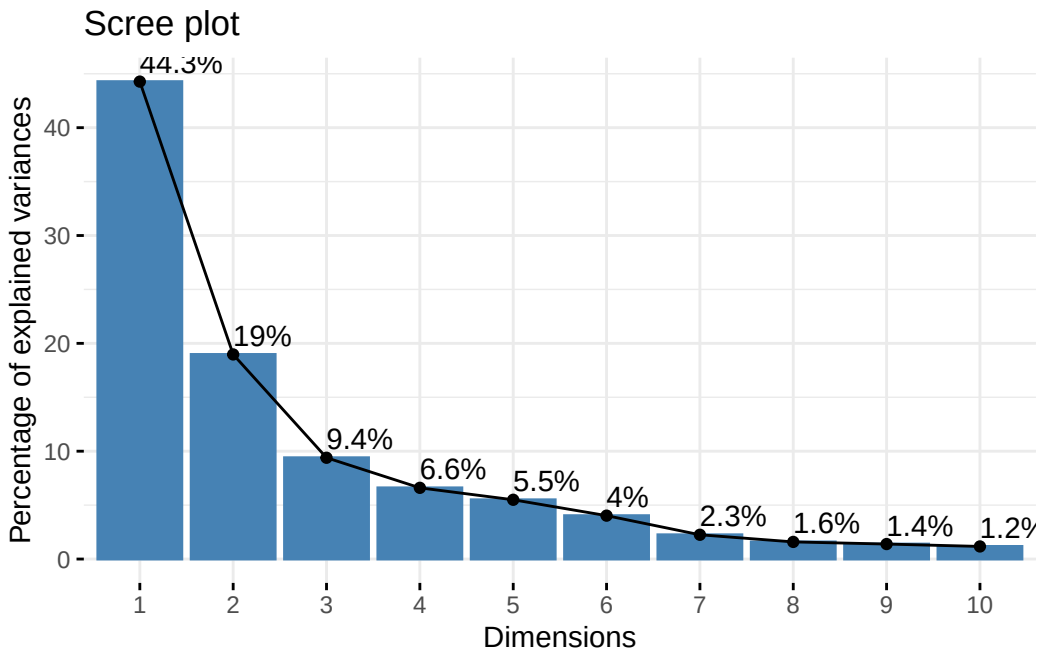

```
# Alternative scree plot of the same data, note data driven y-axis
barplot(pve, ylab = "Precent of Variance Explained",
     names.arg=paste0("PC",1:length(pve)), las=2, axes = FALSE)
axis(2, at=pve, labels=round(pve,2)*100 )
```




```
## ggplot based graph
library(factoextra)
```

```
Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
fviz_eig(wisc.pr, addlabels = TRUE)
```



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?

```
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?

5

Hierarchical clustering

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

Calculate the (Euclidean) distances between all pairs of observations in the new scaled dataset:

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

```
[1] 10.309426  6.771675 10.463467  8.663413  8.402233  9.843286
```

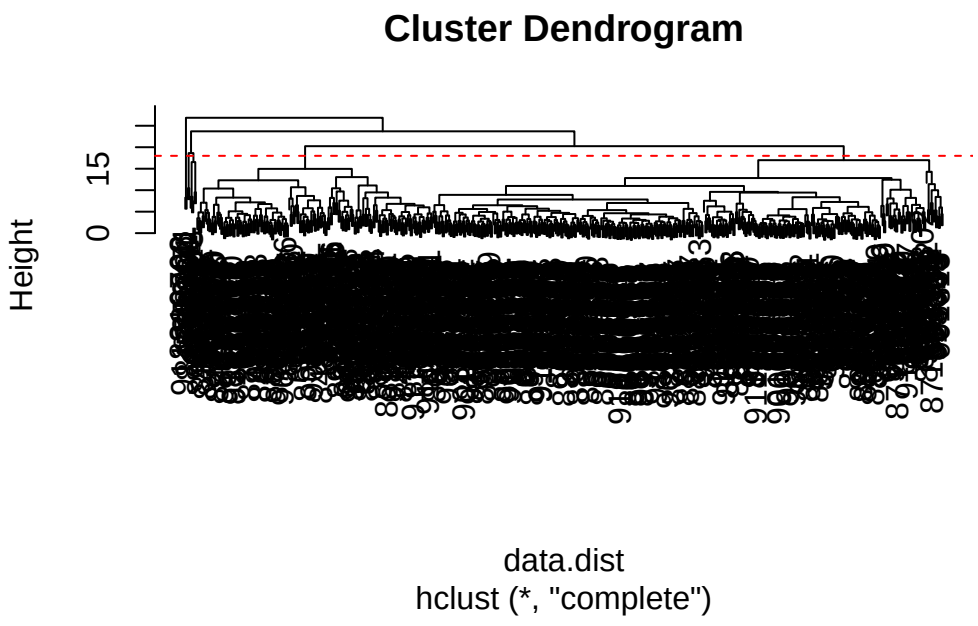
Create a hierarchical clustering model using complete linkage. Manually specify the method argument to hclust():

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

Results of hierarchical clustering

Q11. Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters?

```
plot(wisc.hclust)
abline(h=18, col="red", lty=2)
```



Height at ~18 cuts the tree into 4 clusters

Selecting number of clusters

Use cutree() to cut the tree so that it has 4 clusters:

```
wisc.hclust.clusters <- cutree(wisc.hclust, k=4)
```

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
```

> Q12. Can you find a better cluster vs diagnoses match by cutting into a different number of clusters between 2 and 10?

```
wisc.hclust.clusters2 <- cutree(wisc.hclust, k=6)
table(wisc.hclust.clusters2, diagnosis)
```

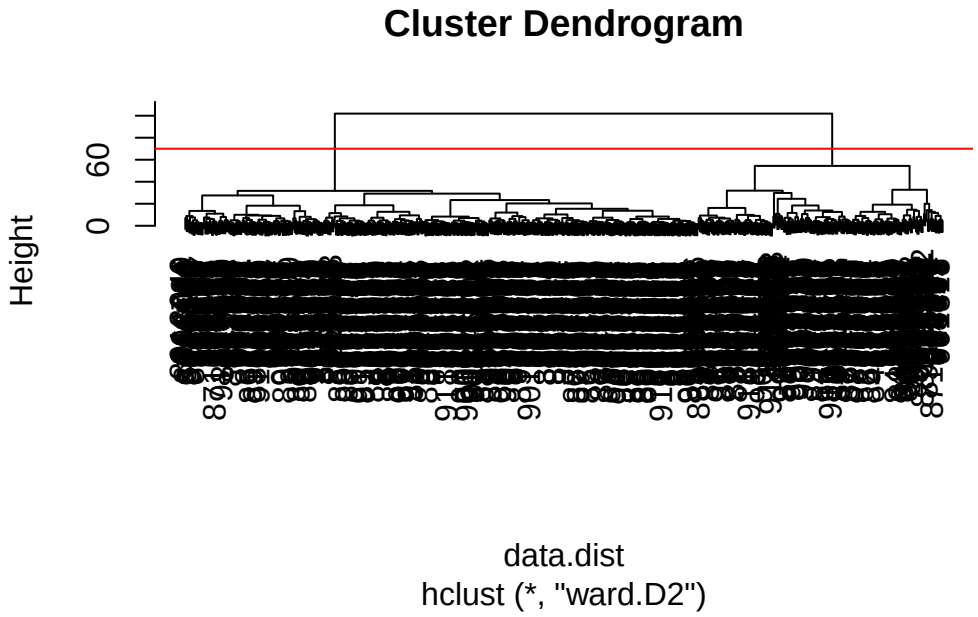
```
                     diagnosis
wisc.hclust.clusters2   B   M
                    1  12 165
                    2   0   5
                    3 331  39
                    4   2   0
                    5  12   1
                    6   0   2
```

No, no matter the k, there are false positives or negatives - there are no 2 distinct groups for diagnosis (benign and malignant). For example, PC1 for k=6 shows that 12 people are benign and 165 people are malignant, so the 12 benign people are likely false negatives.

Using different methods

> Q13. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning.

```
wisc.hclust.d2 <- hclust(data.dist, "ward.D2")
plot(wisc.hclust.d2)
abline(h=70, col="red")
```



`ward.D2` is my favorite because it branches into 2 distinct groups with relatively even distribution.

K-means clustering

2 clusters, corresponding to the actual number of diagnosis. Also, remember to scale the data (with the scale() function and repeat the algorithm 20 times

```
wisc.km <- kmeans(wisc.data, centers=2, nstart=20)
table(wisc.km$cluster, diagnosis)
```

```
   diagnosis
      B   M
  1 356  82
  2   1 130
```

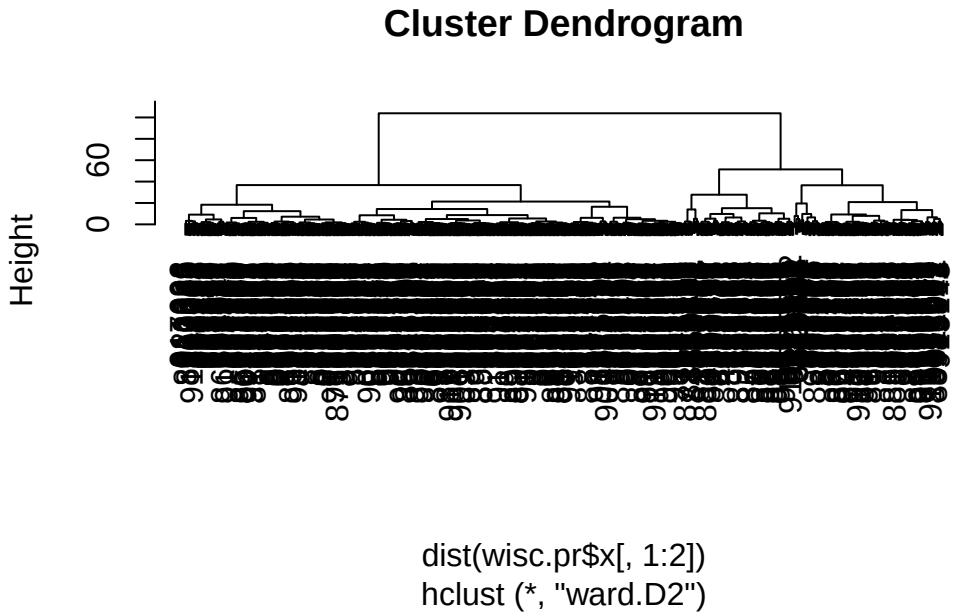
Q14. How well does k-means separate the two diagnoses? How does it compare to your hclust results?

kmeans does better than hclust because it has less false negatives (1) and false positives (82).

Combining methods

Clustering on PCA results

```
hc <- hclust(dist(wisc.pr$x[,1:2]), method="ward.D2") #only shows PC1 and PC2

plot(hc)
```



Cluster membership vector:

```
grps <- cutree(hc, h=70)
table(grps)
```

```
grps
  1   2
195 374
```

```r
table(diagnosis)
```

```
diagnosis
  B   M
357 212
```

Cross-table to see how my clustering groups correspond to the expert diagnosis vector of M and B values

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

```
    diagnosis
grps   B   M
   1  18 177
   2 339  35
```

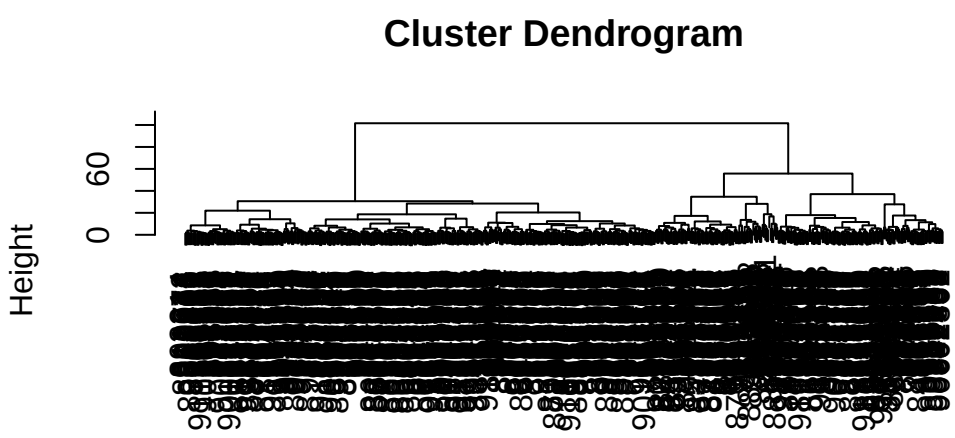
In group 1, majority are malignant, while in group 2, majority are benign.

Positive = cancer (M) Negative = non-cancer (B)

True = cluster/grp 1 False = cluster/grp 2

True positive 177 True negative 339 False positive 35 False negative 18

```r
# Use the distance along the first 7 PCs for clustering (minimum PCs to describe at least 90%
wisc.pr.hclust <- hclust(dist(wisc.pr$x[,1:7]), method="ward.D2")
plot(wisc.pr.hclust)
```



There are two main branches leading to two main clusters - maybe these are malignant and benign. Let's find out!

```
grps <- cutree(wisc.pr.hclust, k=2) # Cut this hierarchical clustering model into 2 clusters
table(grps)
```

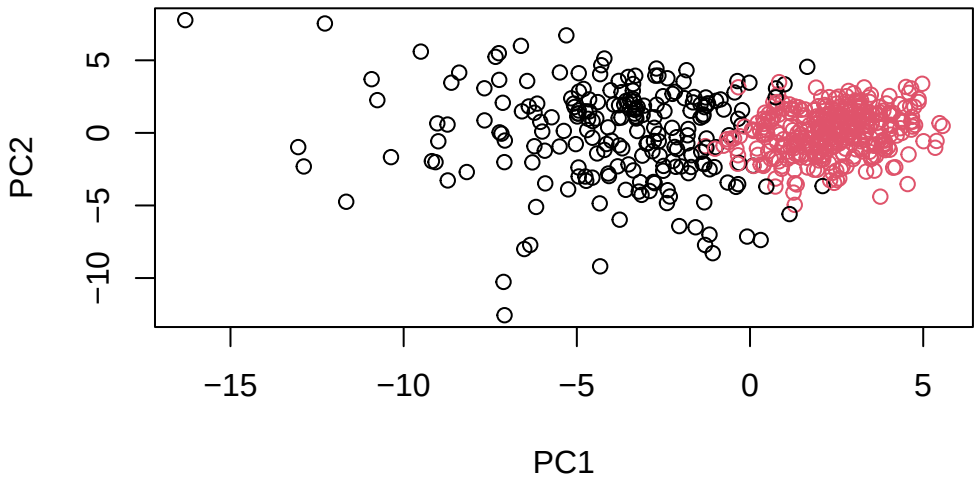
```
grps
  1   2
216 353
```

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

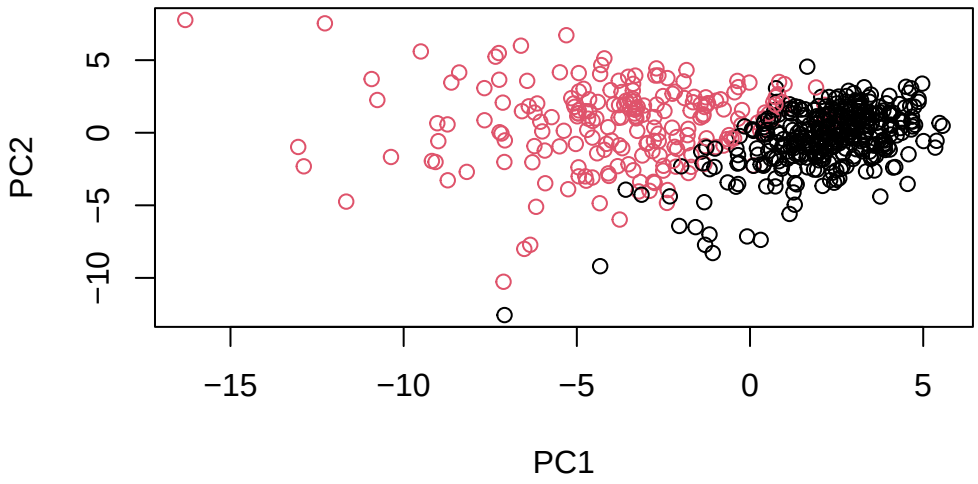
```
    diagnosis
grps   B   M
   1  28 188
   2 329  24
```

grp 1 (grps table) is the sum of the malignant cases in the diagnosis table, while grp 2 is the sum of the benign cases in the diagnosis table.

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




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



The plots for color by grps and diagnosis are the same, except that the colors are swapped. To fix this, we will turn our grps into a factor and reorder the levels so that cluster 2 (B) comes first (black) and cluster 1 (M) gets the second color (red).

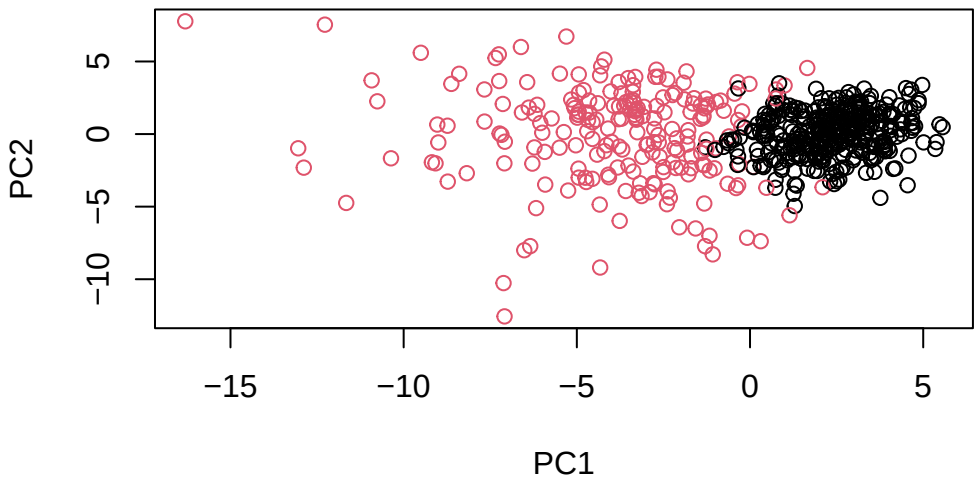
```
g <- as.factor(grps)
levels(g)
```

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

```
g <- relevel(g,2) #reordered starting at 2 [M,B]
levels(g)
```

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

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



Q15. How well does the newly created model with four clusters separate out the two diagnoses?

```
# Compare to actual diagnoses
wisc.pr.hclust.clusters <- cutree(wisc.pr.hclust, k=4)
table(wisc.pr.hclust.clusters, diagnosis)
```

```
                        diagnosis
wisc.pr.hclust.clusters   B   M
                      1   0  45
                      2   2  77
                      3  26  66
                      4 329  24
```

The clusters separates most of the benign and malignant cases correctly. Cluster 1 shows mostly malignant cases, with some misclassified benign cases. Cluster 2 shows mostly benign cases, with some misclassified malignant cases. There are 28 false negatives and 24 false positives.

> Q16. How well do the k-means and 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.

```
table(wisc.km$cluster, diagnosis)
```

```
   diagnosis
      B   M
  1 356  82
  2   1 130
```

```
table(wisc.hclust.clusters, diagnosis)
```

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

Both have false positives/negatives, but k-means does a better job of separating the 2 diagnoses because it has a more distinct separation of the 2 groups and only 1 false negatives and 82 false positives. On the other hand, hclust shows 12 false negatives in PC1 and 40 false positives in PC3. hclust with k=4 also shows groups with mixed characteristics (ie. PC2).

Sensitivity/Specificity

Sensitivity: test's ability to correctly detect ill patient (true positive)

Specificity: test's ability to reject healthy patients (true negative)

> Q17. Which of your analysis procedures resulted in a clustering model with the best specificity? How about sensitivity?

```
130/(130 + 82) #Sensitivity of kmeans
```

```
[1] 0.6132075
```

```
(165+5+2)/(165+5+40+2) #Sensitivity of hclust
```

```
[1] 0.8113208
```

```
356/(1+356) #Specificity of kmeans
```

```
[1] 0.9971989
```

```
343/(12+2+343+0) #Specificity of hclust
```

```
[1] 0.9607843
```

kmeans has better specificity and sensitivity compared to hclust (higher proportion of TP and TN).

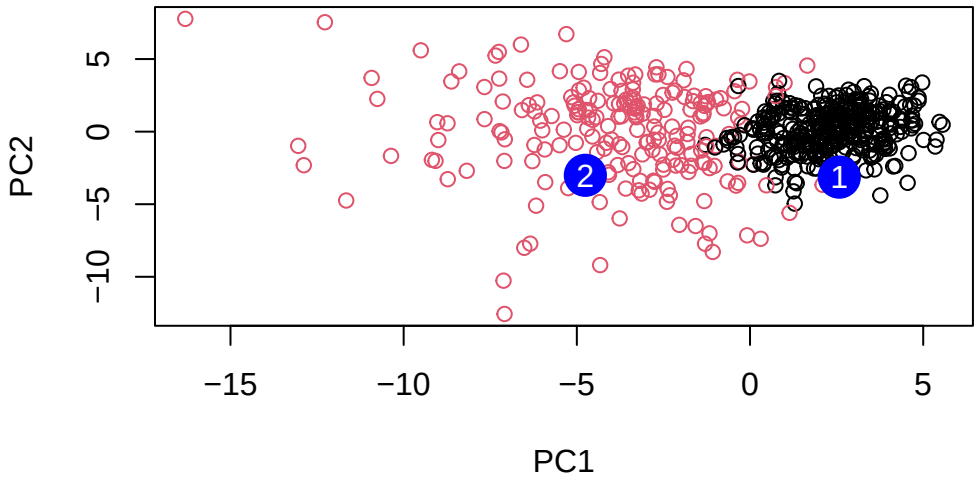
Prediction

We can use our PCA results (wisc.pr) to make predictions on new unseen data. We will use predict() which will take our PCA model from before and new cancer cell data and project that data onto our PCA space.

```
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
plot(wisc.pr$x[,1:2], col=g) #plot PC1 vs PC2 scores
points(npc[,1], npc[,2], col="blue", pch=16, cex=3) #added 2 new points: PC1 and PC2 from new
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 prioritize patient 2 because they are centered/near the malignant cases (red), while patient 1 is within benign cluster.