Class 8 Mini-Project Lab

Zainab Fatima (PID: A16880407)

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.

Practicing on Cars Data First

```
head(mtcars)
```

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

Find the mean value per column of this dataset?

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

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

Standard deviation in each column?

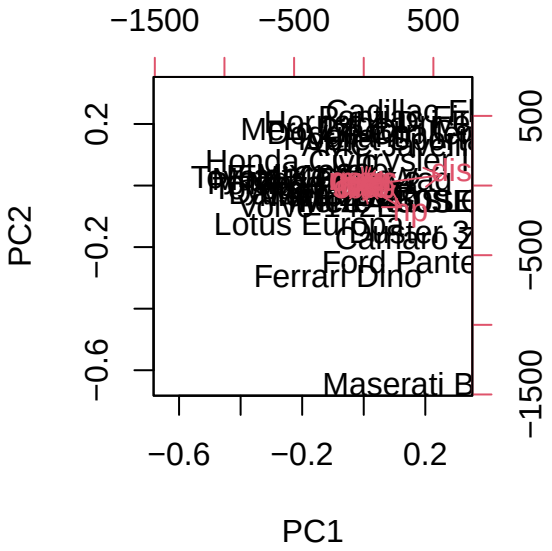
```
apply(mtcars, 2, sd)
```

```
       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 that "displacement" and "horsepower" have the highest mean values and the highest standard deviation. They will likely dominate any analysis I do on this dataset. Let's see

```
pc.noscale <- prcomp(mtcars, scale = FALSE)
pc.scale  <- prcomp(mtcars, scale = TRUE)
#will have different results
```

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



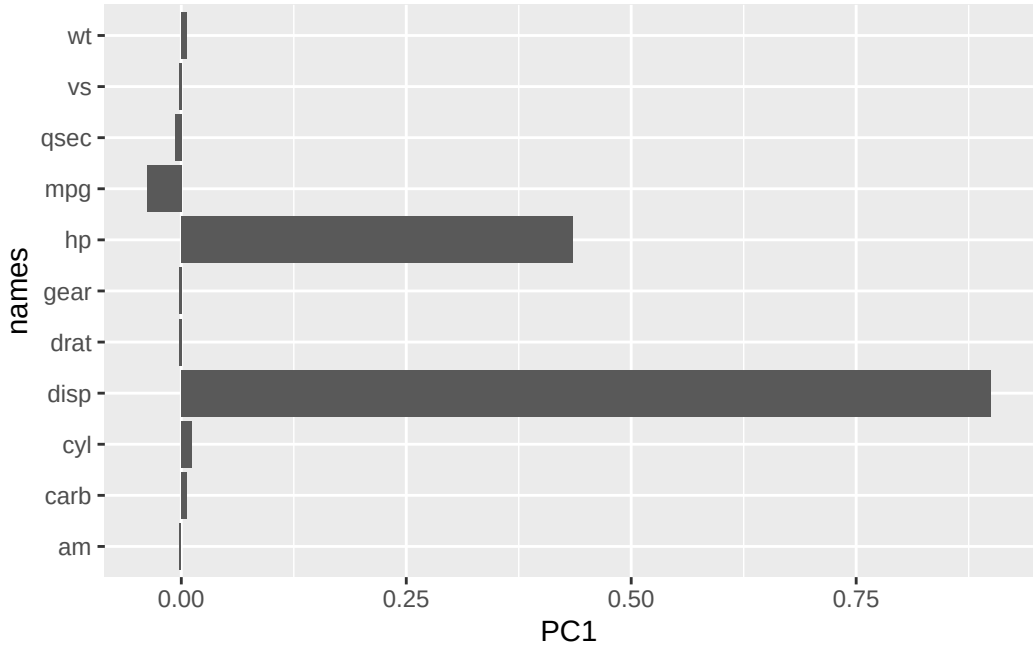

```
pc.noscale$rotation[,1]
```

```
         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
```

```
# shows that hp and disp dominate this dataset since they have highest magnitude
```

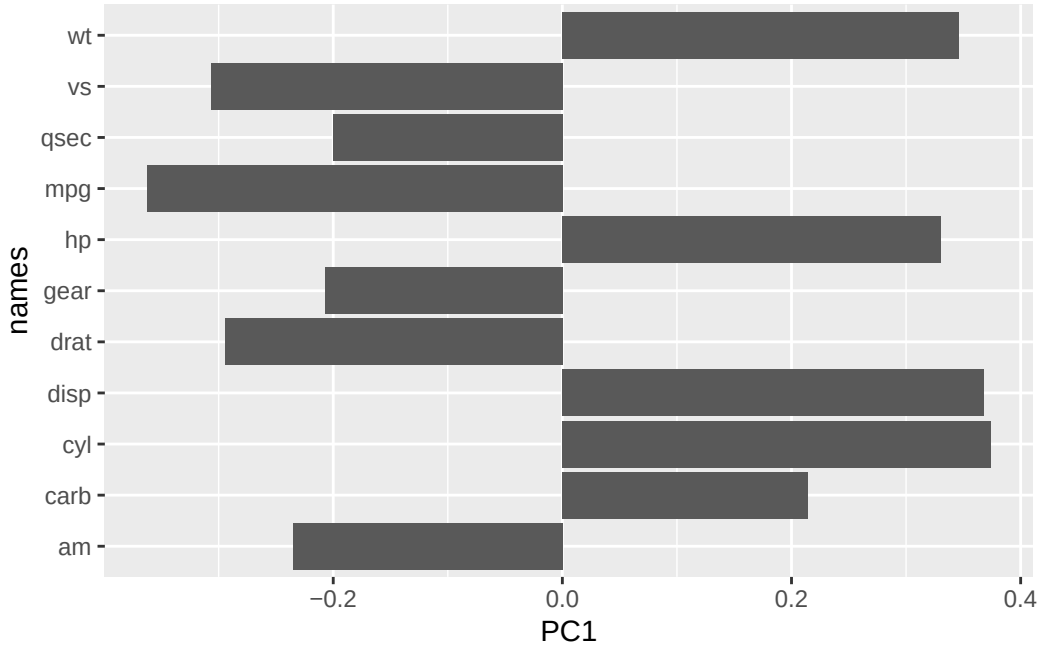
Plot the loadings (how the original varaibles contribute to the PCs)

```
library(ggplot2)

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

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



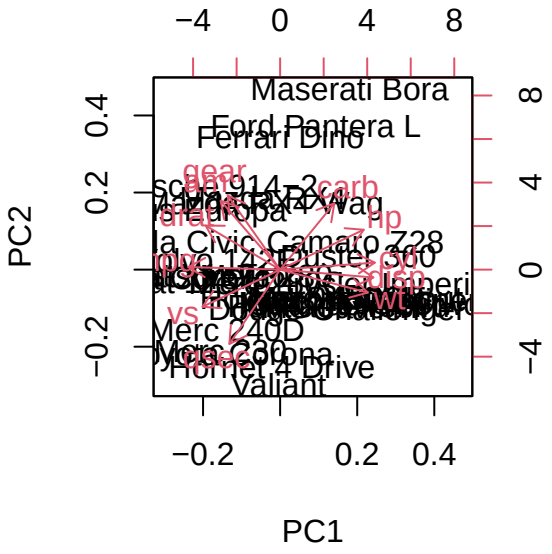

```
#the disp and hp bars are dominating
```

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

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




```
#no dominating columns, more columns are equally spread
#analysis takes information about all columns
#if i measured with different units, scaling the data prevents one variable from dominating t
```

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




```
#much better plot, all variables have red arrows so they all contribute
#more accurate groups of cars
```

> **Take home**: Generally we always want to set `scale = TRUE` when we do this type of analysis to avoid our analysis being dominated by individual variables with the largest variance just due to their unit of measurement.

Cancer Biopsies

FNA Breast Cancer Data

Load the data into R.

```
wisc.df <- read.csv("WisconsinCancer (2).csv", row.names = 1)
#row.names = 1 removes patient ID column title
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 observations have a malignant diagnosis?

```
# == says if M = true, N = false which can be summed
# true = 1, False = 0
sum(wisc.df$diagnosis == "M")
```

```
[1] 212
```

The `table()` function can also be used to give number of B and M.

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

```
  B   M
357 212
```

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

There are the 10 features in the data with the "_mean" suffix. The code is listed below.

```
ncol(wisc.df) #gives number of columns
```

```
[1] 31
```

```
colnames(wisc.df) #gives 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()`

```
#grep("_mean", colnames(wisc.df))
#output is the # of the vector that contains the "_mean" (1 to 11)
length(grep("_mean", colnames(wisc.df)))
```

```
[1] 10
```

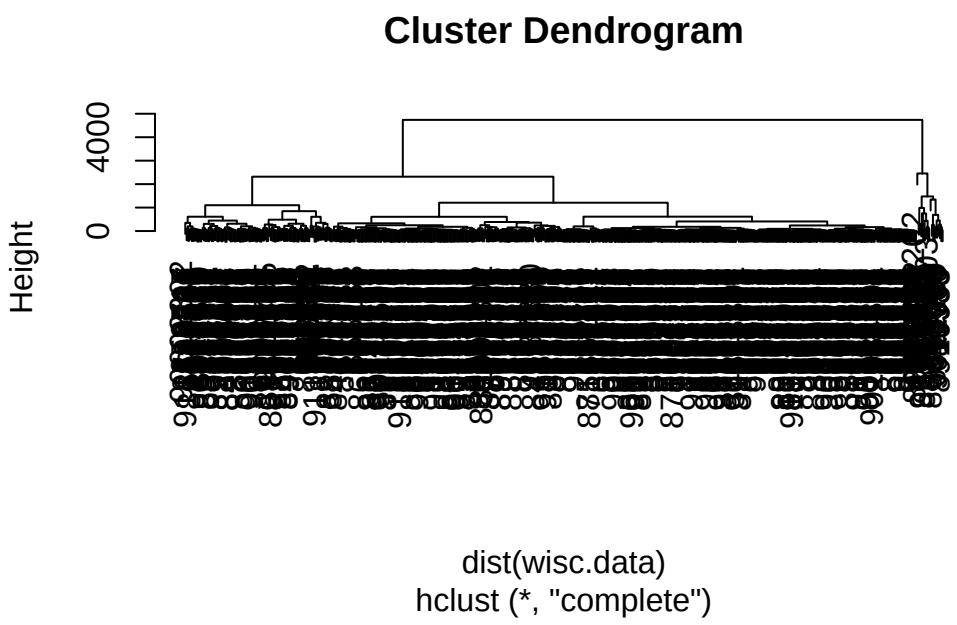
Before we go any further we need to exclude the diagnosis column from any future analysis - this tells us whether a sample matches to cancer or non-cancer.

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

```
wisc.data <- wisc.df[,-1]
#gives everything other than 1st column (diagnosis)
```

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

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

The first principal component captures 44% of the original variance.

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

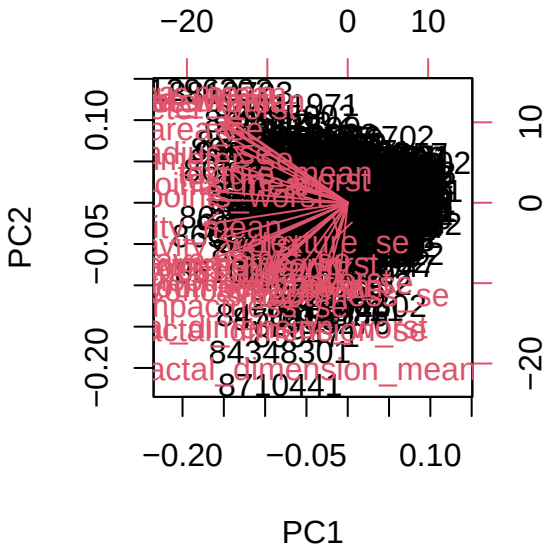
3 principal components are required to capture 73% of data.

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

7 principal components capture 91% of data.

Building a Bi-plot

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



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

The aspect of the plot that stands out to me is the variables are messy and on top of each other. This bipolot is difficult to understand and not useful. This is because bi-plots work best on smaller datasets. 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)
```

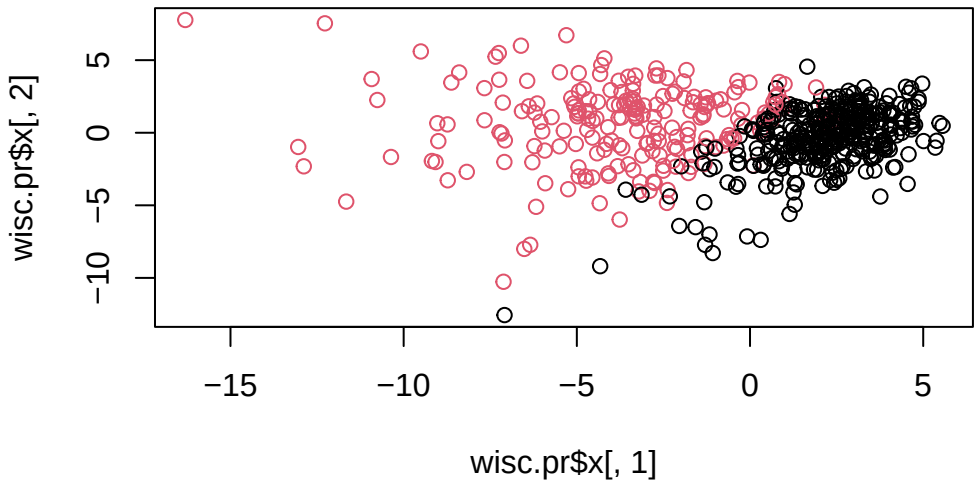
```
              PC1        PC2        PC3       PC4        PC5         PC6
842302   -9.184755  -1.946870 -1.1221788 3.6305364  1.1940595  1.41018364
842517   -2.385703   3.764859 -0.5288274 1.1172808 -0.6212284  0.02863116
84300903 -5.728855   1.074229 -0.5512625 0.9112808  0.1769302  0.54097615
84348301 -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
```

Scatter plot observations by compoents 1 and 2

Plot of PC1 vs PC2 the first two columns

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

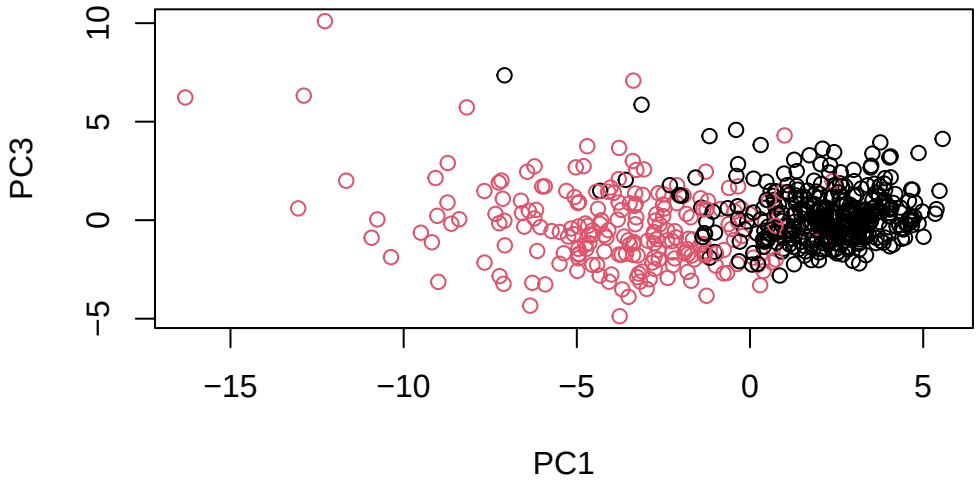


```
#red = cancer, black = benign
```

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

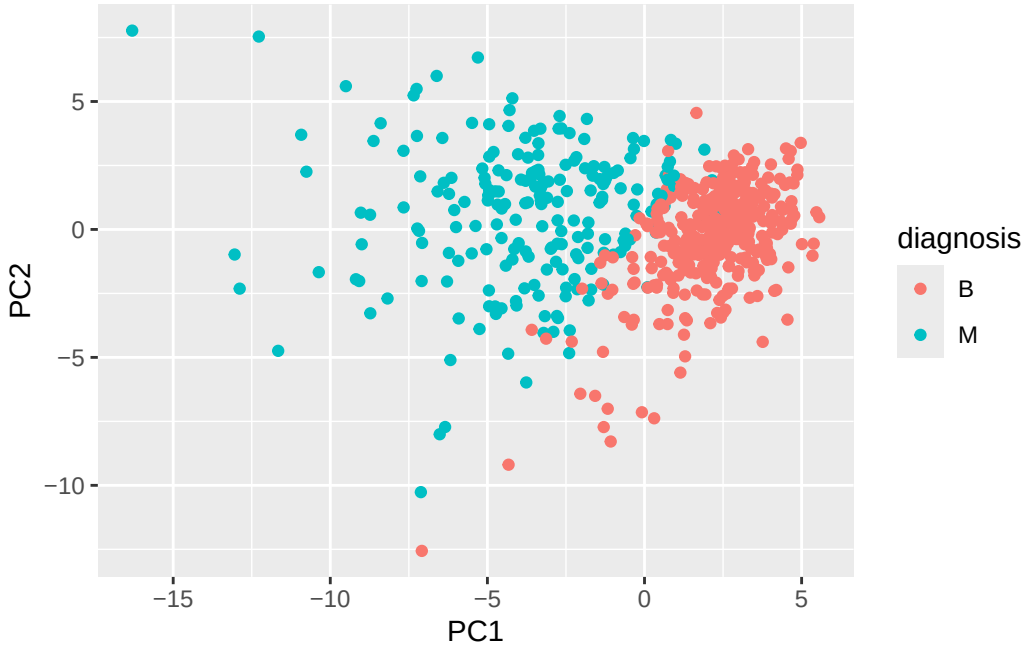
PC 3 captures less variance than PC 2. The plot PC 1 vs PC3 has two main clusters based on diagnosis but there is a lot over overlap and mixing between the data points.

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



Make a ggplot version of the score plot for component 1 and 2

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

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



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 from each principal component
pve <- pr.var / sum(pr.var)
head(pve)  # Display the first few values
```

```
[1] 0.44272026 0.18971182 0.09393163 0.06602135 0.05495768 0.04024522
```

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 (Lab Sheet Version). What is the minimum number of principal components required to explain 80% of the variance of the data?

4 principal components are needed to cover 80% of the data, based on the table below.

```
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
```

Section 3: Hierarchical Clustering

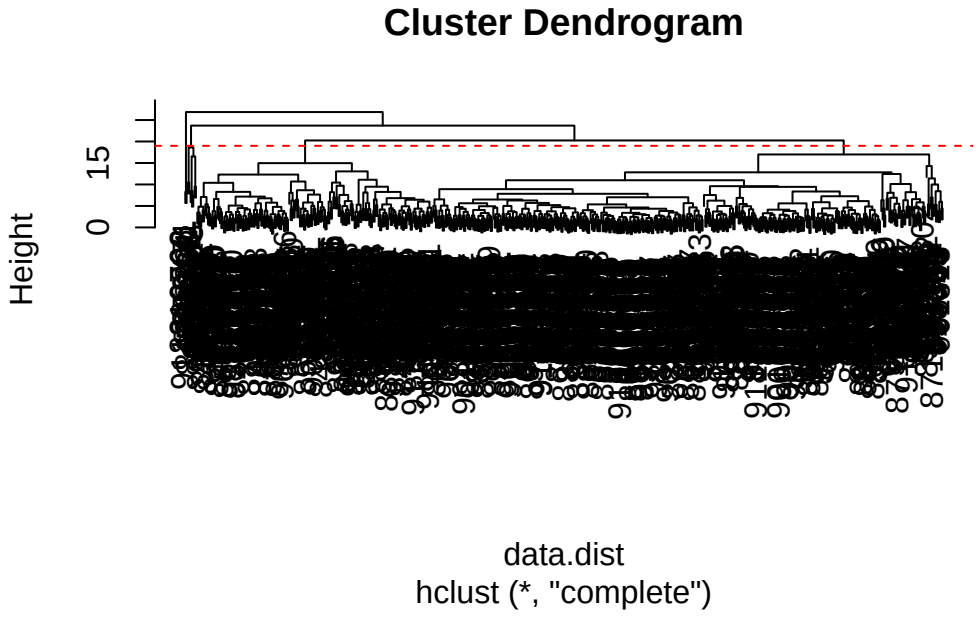
> Q10. Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters?

Based on the plot below, the height of 19 would yeild 4 clusters.

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

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

```
plot(wisc.hclust)
abline(h = 19, col= "red", lty =2)
```



There are 4 clusters at a height of 19.

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

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

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

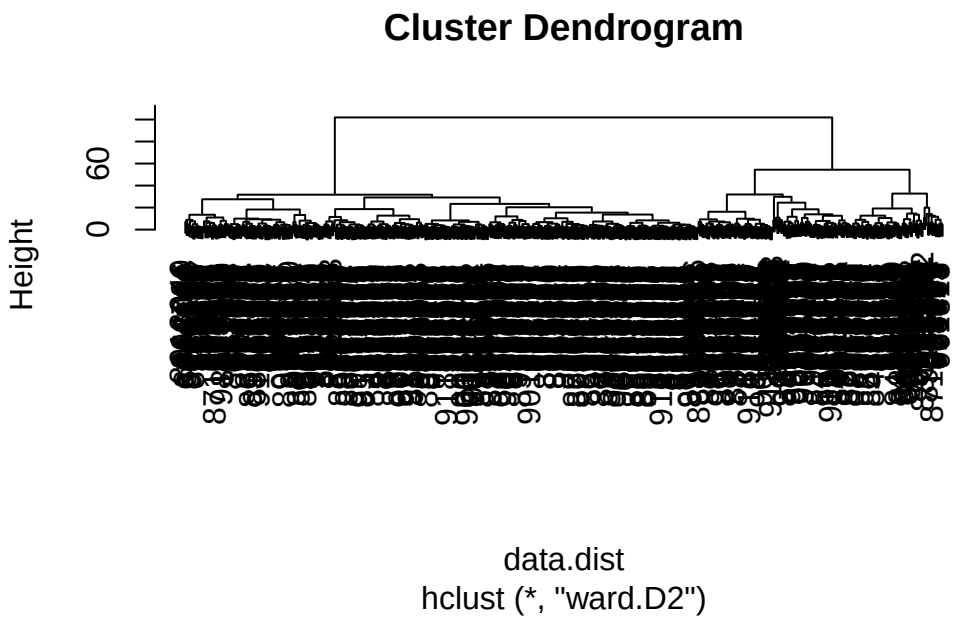
The model with 4 clusters separates the two diagnoses well with cluster 1 mostly malignant cells and cluster 3 is mostly benign cells, but there are some exceptions within cluster 1 and 3 were this is not true. Cluster 4 is mostly malignant and cluster 2 is also mostly malignant.

Using different methods

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

I prefer the Ward.D2 method for hierarchical clustering because it focuses on minimizing within-cluster variance, which means that the points in each group are more similar and the groups are more defined. This can be seen in the below plot where the 2 main clusters have more branches within them compared to the previous dendrogram where the the clusters had more variation within them.

```
wisc.hclust.2 <- hclust(data.dist, method = "ward.D2")

plot(wisc.hclust.2)
```



#Section 4: K-means clustering

```
wisc.km <- kmeans(data.scaled, centers= 2, nstart= 20)
```

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

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

```r
table(wisc.hclust.clusters, wisc.km$cluster)
```

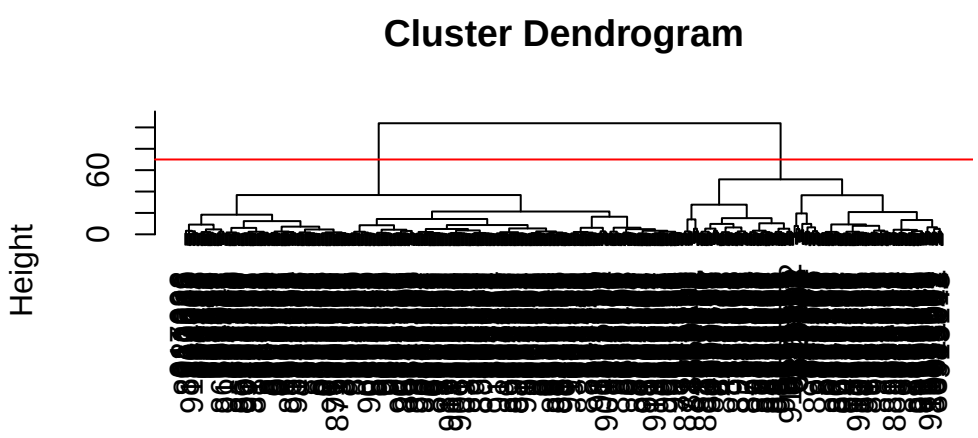
```
wisc.hclust.clusters   1   2
                   1  17 160
                   2   0   7
                   3 363  20
                   4   0   2
```

Based on the second table, clusters 1, 2, and 4 from the hierarchical clustering model can be interpreted as the cluster 1 equivalent from the k-means algorithm, and cluster 3 from the hierarchical clustering model can be interpreted as the cluster 2 equivalent from the k-means algorithm.

The k-means algorithm provided a clean divide between the benign and malignant tumors. The hierarchical clustering into 4 groups created 3 groups that are similar, but may reflect smaller divisions within the data.

Section 5: Combining Methods

```r
wisc.pr.hclust <- hclust(dist(wisc.pr$x[, 1:2]), method="ward.D2")
plot(wisc.pr.hclust)
abline(h=70, col ="red")
```



Cluster membership vector

In class, we used 2 clusters instead of 4.

```
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
grps
  1   2
195 374
```

```
table(diagnosis)
```

```
diagnosis
  B   M
357 212
```

> Q13. How well does the newly created model with 4 clusters seperate out the 2 diagnoses?

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

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

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

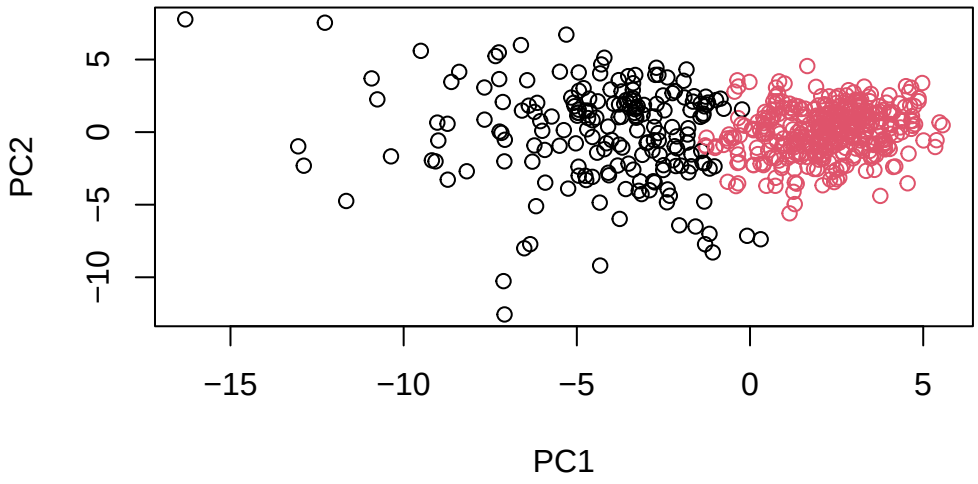
Majority of grp 1 is malignant, majority of grp 2 is benign. The clustering worked well since the diagnosis in each group is majority one diagnosis and the diagnoses in the minority of each reflect the false positives and false negatives.

positive => cancer M negative => non-cancer B

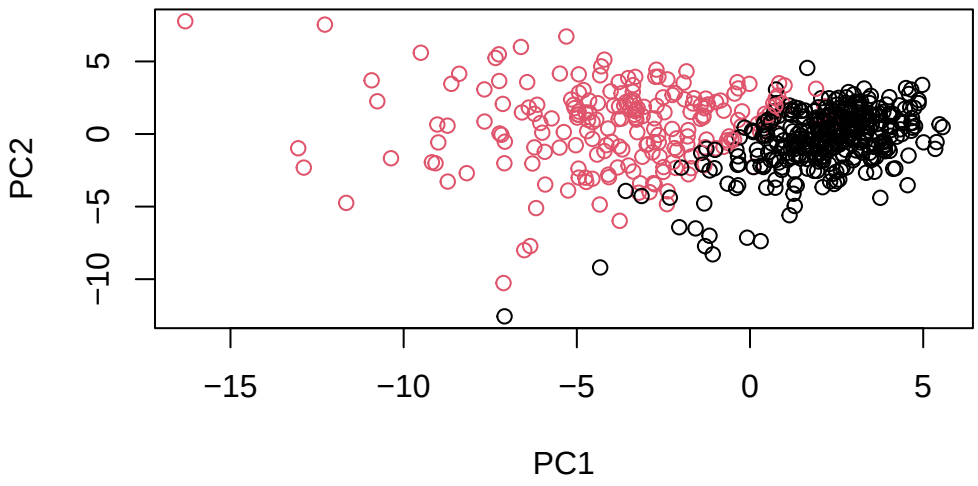
True positive = 177 False positive = 18 True negative = 339 false negative = 35

- One way to get great sensitivity: everyone is M
- One way to great specificity = accurately divide into M and B

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




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



Q14. How well do the hierarchical clustering models you created in previous sections (ie before PCA) do in terms of seperating the diagnoses? Again, use the table() function to compare the output of each model with the vector containing the actual diagnoses.

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

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

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

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

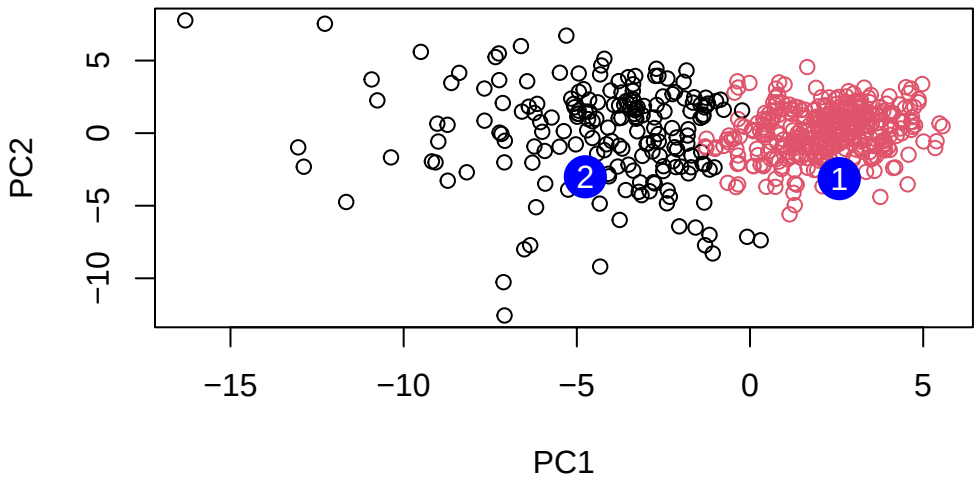
Both models have two main clusters (mostly malignant or benign) that have false positives or false negatives. K-means has a more clear divide between malignant and benign due to the smaller amount of clusters. Hierarchical clustering with PCA performed more closely to k-means in terms of separation. Hierarchical clustering before PCA created smaller clusters that didn't have distinct differences in disease diagnosis from the main two clusters.

Prediction

We can use our PCA results (wisc.pr) to amke predictions on new unseen data.

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

```
plot(wisc.pr$x[,1:2], col=grps)
points(npc[,1], npc[,2], col="blue", pch=16, cex=3)
text(npc[,1], npc[,2], c(1,2), col="white")
```



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

Patient #2 since it's near the malignant cluster.