Breast Cancer Mini Project

Andy Hsu

Analysis of a Cancer Dataset

In this project, we will analyze a dataset of breast cancer biopsies from fine needle aspiration (FNA).

Getting Started

We will start this project by downloading the dataset, setting patient IDs to the row names.

```r
wisc.df <- read.csv("wisconsincancer.csv",row.names=1)
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
```

We will also separate the diagnosis, as it will not be used until later.

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

Running commands to gain some basic information on the dataset, we find that there are 569 observations, 212 malignant diagnoses, and 10 variables suffixed with "_mean". Importantly, we can use the `grep()` function to match the `colnames()` to the pattern of `"_mean"`, then enter the function `length()` to count the total.

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

```
[1] 569  30
```

```
table(diagnosis)
```

```
diagnosis
  B   M 
357 212 
```

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

```
[1] 10
```

Performing PCA

Before we can begin with PCA, we need to check that PCA can be applied to the dataset.

```
colMeans(wisc.data)
```

```
            radius_mean            texture_mean          perimeter_mean 
           1.412729e+01            1.928965e+01            9.196903e+01 
              area_mean         smoothness_mean        compactness_mean 
           6.548891e+02            9.636028e-02            1.043410e-01 
         concavity_mean     concave.points_mean           symmetry_mean 
           8.879932e-02            4.891915e-02            1.811619e-01 
 fractal_dimension_mean               radius_se              texture_se 
           6.279761e-02            4.051721e-01            1.216853e+00 
           perimeter_se                 area_se           smoothness_se 
           2.866059e+00            4.033708e+01            7.040979e-03 
         compactness_se           concavity_se       concave.points_se
           2.547814e-02           3.189372e-02            1.179614e-02
            symmetry_se    fractal_dimension_se            radius_worst
           2.054230e-02           3.794904e-03            1.626919e+01
          texture_worst        perimeter_worst              area_worst
           2.567722e+01           1.072612e+02            8.805831e+02
       smoothness_worst      compactness_worst         concavity_worst
           1.323686e-01           2.542650e-01            2.721885e-01
   concave.points_worst         symmetry_worst fractal_dimension_worst
           1.146062e-01           2.900756e-01            8.394582e-02
```

```
apply(wisc.data,2,sd)
```

```
            radius_mean           texture_mean          perimeter_mean
           3.524049e+00           4.301036e+00            2.429898e+01
              area_mean        smoothness_mean        compactness_mean
           3.519141e+02           1.406413e-02            5.281276e-02
         concavity_mean    concave.points_mean           symmetry_mean
           7.971981e-02           3.880284e-02            2.741428e-02
 fractal_dimension_mean              radius_se              texture_se
           7.060363e-03           2.773127e-01            5.516484e-01
           perimeter_se                area_se           smoothness_se
           2.021855e+00           4.549101e+01            3.002518e-03
         compactness_se           concavity_se       concave.points_se
           1.790818e-02           3.018606e-02            6.170285e-03
            symmetry_se    fractal_dimension_se            radius_worst
           8.266372e-03           2.646071e-03            4.833242e+00
          texture_worst        perimeter_worst              area_worst
           6.146258e+00           3.360254e+01            5.693570e+02
       smoothness_worst      compactness_worst         concavity_worst
           2.283243e-02           1.573365e-01            2.086243e-01
   concave.points_worst         symmetry_worst fractal_dimension_worst
           6.573234e-02           6.186747e-02            1.806127e-02
```

Now, we can run a PCA using the `prcomp()` function.

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

```
Importance of components:
```

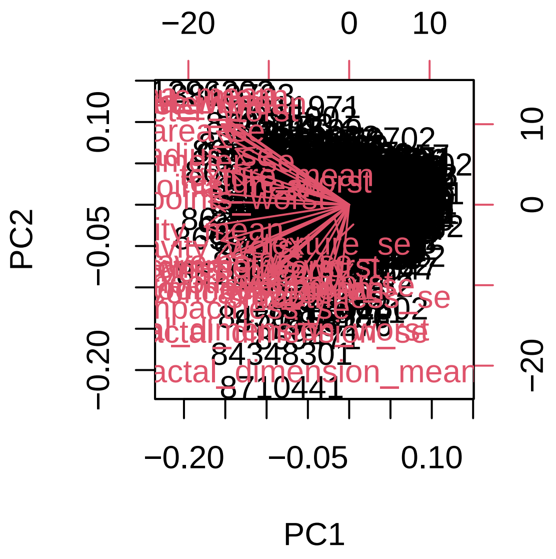
```
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172
Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251
Cumulative Proportion  0.4427 0.6324 0.72636 0.79239 0.84734 0.88759 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
```

Reading off the summary table, we can see that PC1 accounts for 44.3% of the original variance. Additionally, we need the first 3 PCs to reach 70% variance and 7 PCs to reach 90% variance.

Interpreting PCA

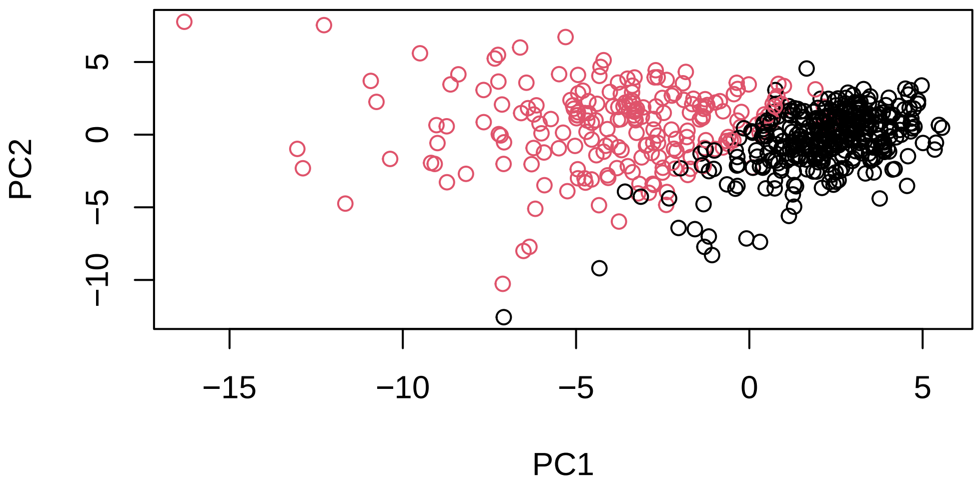
To take a look at our dataset, we can try to plot a biplot.

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



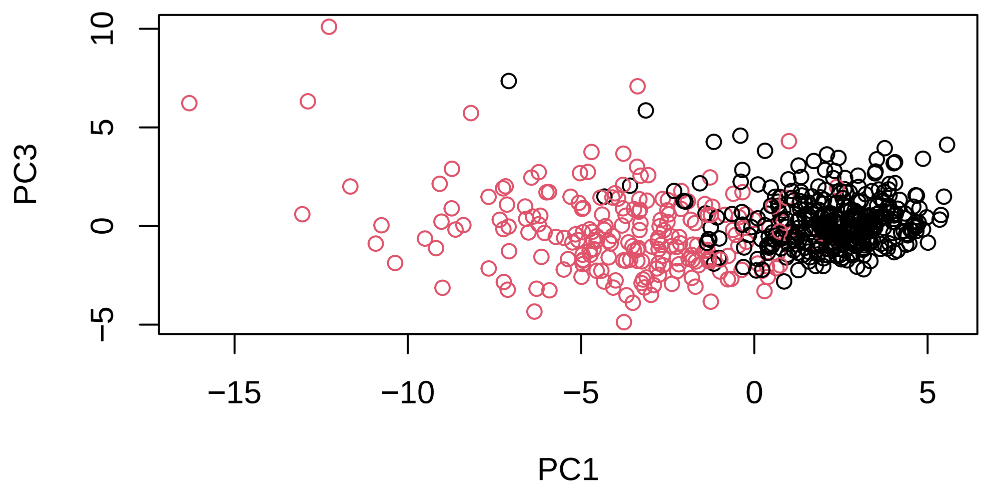
This plot is impossible to read, though. The overlapping text makes it too hard to tell where any datapoint is. Perhaps this plot would be better off with points labeled as dots rather than text.

```r
plot(wisc.pr$x, col = (diagnosis=="M")+1, xlab = "PC1", ylab = "PC2")
```



To view this same plot with PC1 against PC3, we can modify the code. Notice that the distribution of the first plot shows a clearer separation between the 2 groups.

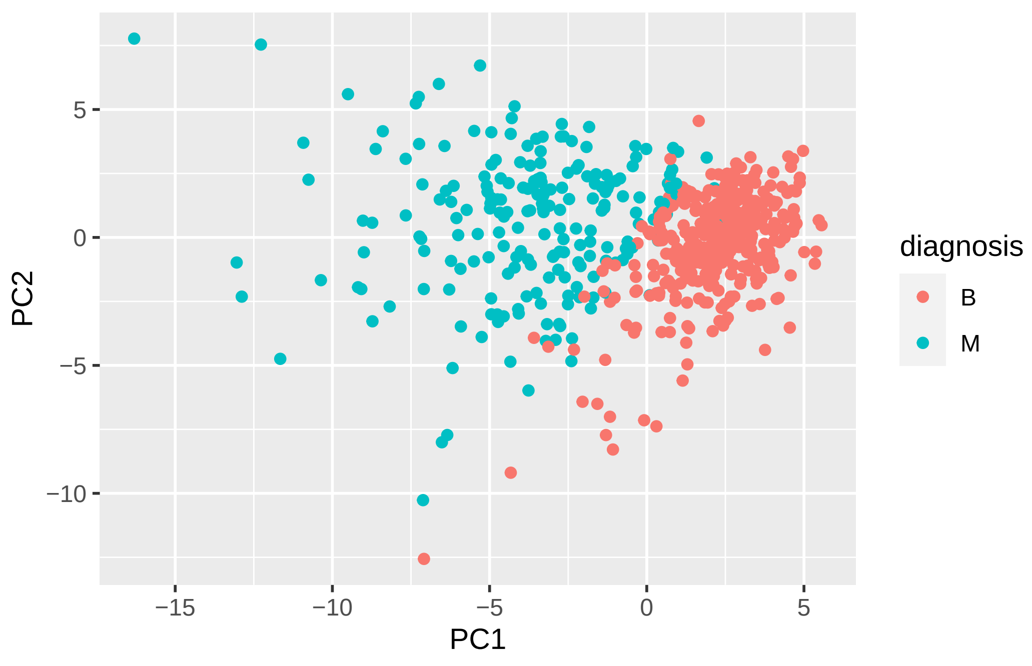
```
plot(wisc.pr$x[,c(1,3)], col = (diagnosis=="M")+1, xlab = "PC1", ylab = "PC3")
```



It's now time to plot our results using ggplot2. First, we need to convert our data to a dataframe, including the diagnosis, then we can plot.

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

library(ggplot2)
ggplot(df, aes(PC1,PC2,col=diagnosis)) +
  geom_point()
```



Plotting Variance on a Scree Plot

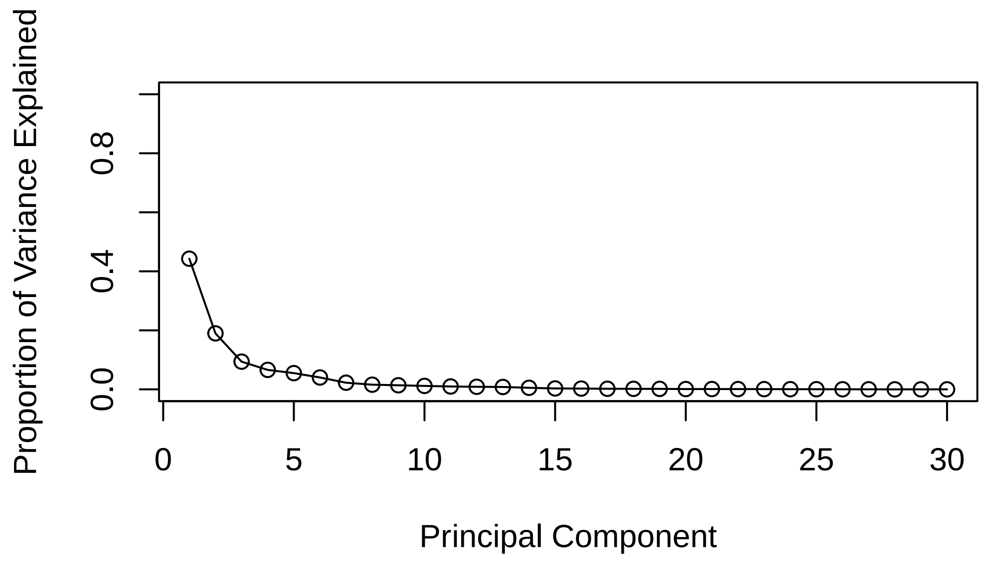
We can also visualize our variance against the number of PCs via a scree plot. First, we calculate the variance of each PC.

```r
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

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

Next, we can calculate the proportion of the total variance explained by each PC, then plot this for each PC.

```r
pve <- pr.var/sum(pr.var)
plot(pve, xlab = "Principal Component",
     ylab = "Proportion of Variance Explained",
     ylim = c(0, 1), type = "o")
```



To find specific data, we can call the rotation portion of our results, finding that the PC1 component for the variable "concave.points_mean" is -0.26. Going back to the summary of results, we can also see that a minimum of 5 PCs brings us to 80% variance.

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

```
[1] -0.2608538
```

```
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
```

```r
sum(pve[1:5])
```

```
[1] 0.8473427
```

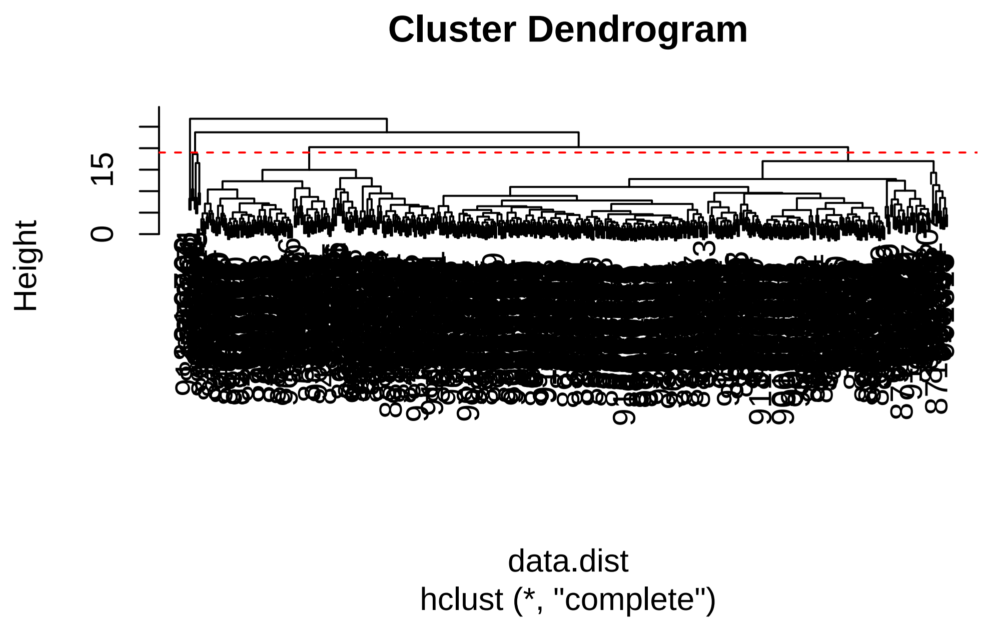
Heirarchical Clustering

Now, let's perform some h-clusting on the data to see how that algorithm handles this data. First, we prepare the dataset by scaling it and finding distances. Then, we can find the `hclust()`.

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

Plotting the results, we find that the height where there are 4 clusters is at 19.

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



Judging from the dendrogram, it may be worthwhile to cut the tree into 4 clusters. We can then use the `table()` function to compare against the diagnoses.

```
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
```

Repeating this for different cluster counts from 2 to 10 shows that 4 is probably the best cluster vs diagnosis match in terms of ratios. Below is an example of one of these repeats.

```
wisc.hclust.clusters8 <- cutree(wisc.hclust,k=8)
table(wisc.hclust.clusters8,diagnosis)
```

```
                     diagnosis
wisc.hclust.clusters8   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
```

We can also test this using the different h-clust methods available to see if any will fit better than the default `"complete"` method. Below is the code for the exploration of these alternative methods.

```r
wisc.hclust.s <- hclust(data.dist,method="single")
wisc.hclust.s.clusters <- cutree(wisc.hclust.s,k=4)
table(wisc.hclust.s.clusters,diagnosis)
```

```
                      diagnosis
wisc.hclust.s.clusters   B   M
                     1 356 209
                     2   1   0
                     3   0   2
                     4   0   1
```

```r
wisc.hclust.a <- hclust(data.dist,method="average")
wisc.hclust.a.clusters <- cutree(wisc.hclust.a,k=5)
table(wisc.hclust.a.clusters,diagnosis)
```

```
                      diagnosis
wisc.hclust.a.clusters   B   M
                     1 355 208
                     2   2   0
                     3   0   1
                     4   0   2
                     5   0   1
```

```r
wisc.hclust.w <- hclust(data.dist,method="ward.D2")
wisc.hclust.w.clusters <- cutree(wisc.hclust.w,k=2)
table(wisc.hclust.w.clusters,diagnosis)
```

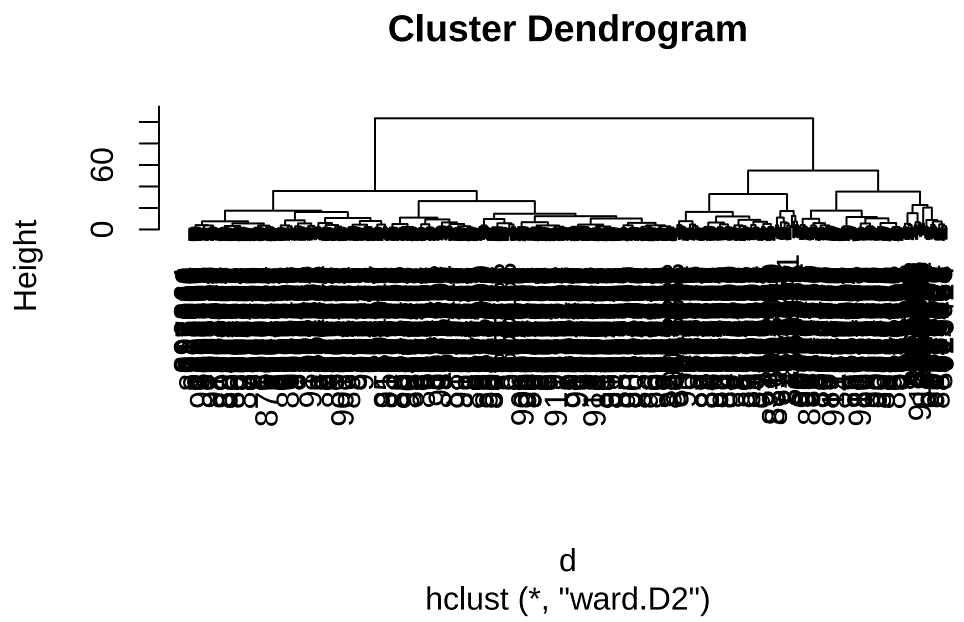
```
                      diagnosis
wisc.hclust.w.clusters   B   M
                     1  20 164
                     2 337  48
```

Of these new methods, ward.D2 seems to be the best, but still is not as representative of the diagnoses as the complete method, when judging by the ratios of “inaccurate” vs “accurate” data points.

Combining Methods

With both PCA and h-clusting, we can combine these methods to obtain a potentially better result of grouping. To do this, we can take the results from our PCA and perform h-clusting on it.

```r
d <- dist(wisc.pr$x[,1:3])
wisc.pr.hclust <- hclust(d,method="ward.D2")
plot(wisc.pr.hclust)
```



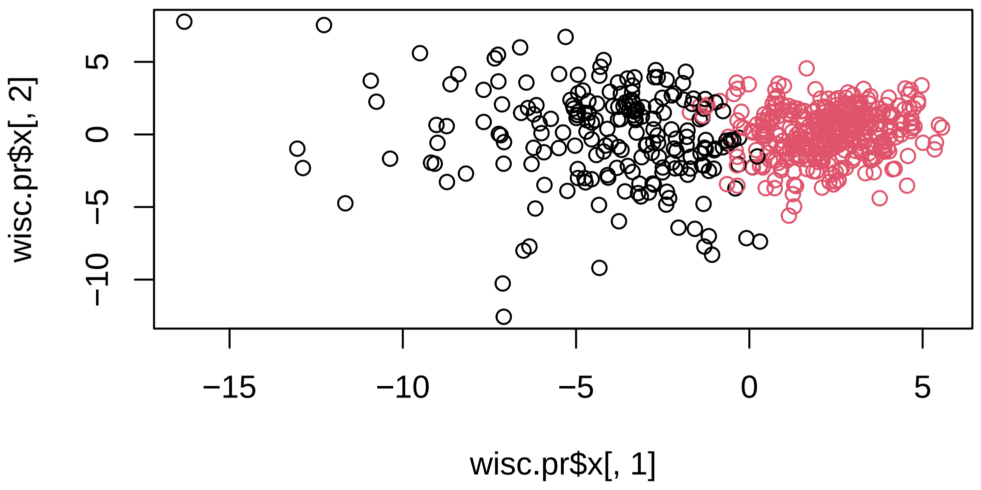
With 2 groups clearly delineated, we can cut the tree into 2.

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

```
 842302   842517 84300903 84348301 84358402   843786 
      1        1        1        1        1        1 
```

Finally, we can plot our PC1 vs PC2, colored by the groups found in our previous step.

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



To check against our expert diagnoses, we call the `table()` function.

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

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

Examining our results, we find that this combination of methods obtained the most “accurate” groups according to the ratios. Comparing to the other methods of heirarchical clustering and PCA, a combination of the two produced the best results.