lab8, Sydney Ackermann PID A69036053

```r
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
```

Let's look at the average mean value of every column.:

```r
apply(mtcars, 2, mean) # 2 for col, 1 for rows
```

```
       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 
```

```r
# says average mpg is 20
```

Now lets look at spread in each of these columns. each car (row) is a dimension of the data set.

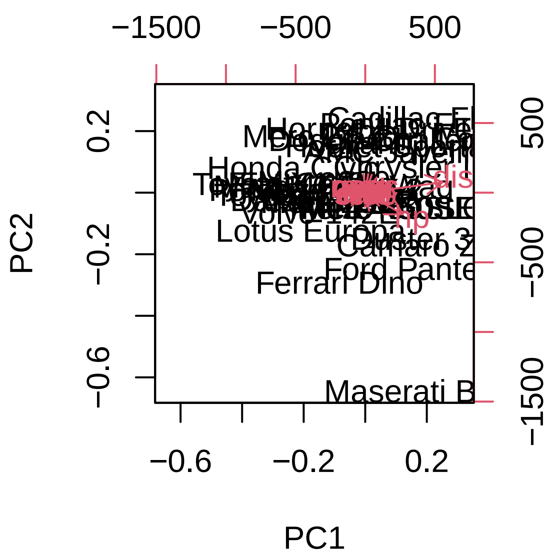
Let's look at the spread via `sd()`.

```r
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 
```

Lets do a pca on it

```r
pca <- prcomp(mtcars)
biplot(pca)
```



Here the problem is that the columns are measured in different units. Lets try scaling the data. what is it?

```r
mtscale <- scale(mtcars)
head(mtscale) # now they are in the same units
```

```
                         mpg        cyl        disp         hp       drat
Mazda RX4          0.1508848 -0.1049878 -0.57061982 -0.5350928  0.5675137
Mazda RX4 Wag      0.1508848 -0.1049878 -0.57061982 -0.5350928  0.5675137
Datsun 710         0.4495434 -1.2248578 -0.99018209 -0.7830405  0.4739996
Hornet 4 Drive     0.2172534 -0.1049878  0.22009369 -0.5350928 -0.9661175
Hornet Sportabout -0.2307345  1.0148821  1.04308123  0.4129422 -0.8351978
```

```
Valiant            -0.3302874 -0.1049878 -0.04616698 -0.6080186 -1.5646078
                           wt       qsec         vs          am        gear
Mazda RX4          -0.610399567 -0.7771651 -0.8680278  1.1899014  0.4235542
Mazda RX4 Wag      -0.349785269 -0.4637808 -0.8680278  1.1899014  0.4235542
Datsun 710         -0.917004624  0.4260068  1.1160357  1.1899014  0.4235542
Hornet 4 Drive     -0.002299538  0.8904872  1.1160357 -0.8141431 -0.9318192
Hornet Sportabout   0.227654255 -0.4637808 -0.8680278 -0.8141431 -0.9318192
Valiant             0.248094592  1.3269868  1.1160357 -0.8141431 -0.9318192
                         carb
Mazda RX4           0.7352031
Mazda RX4 Wag       0.7352031
Datsun 710         -1.1221521
Hornet 4 Drive     -1.1221521
Hornet Sportabout  -0.5030337
Valiant            -1.1221521
```

What is the mean of each dimension/column in mtscale?

```r
round(apply(mtscale, 2, mean), 3) # round to 3 sig figs
```

```
 mpg  cyl disp   hp drat   wt qsec   vs   am gear carb 
   0    0    0    0    0    0    0    0    0    0    0 
```

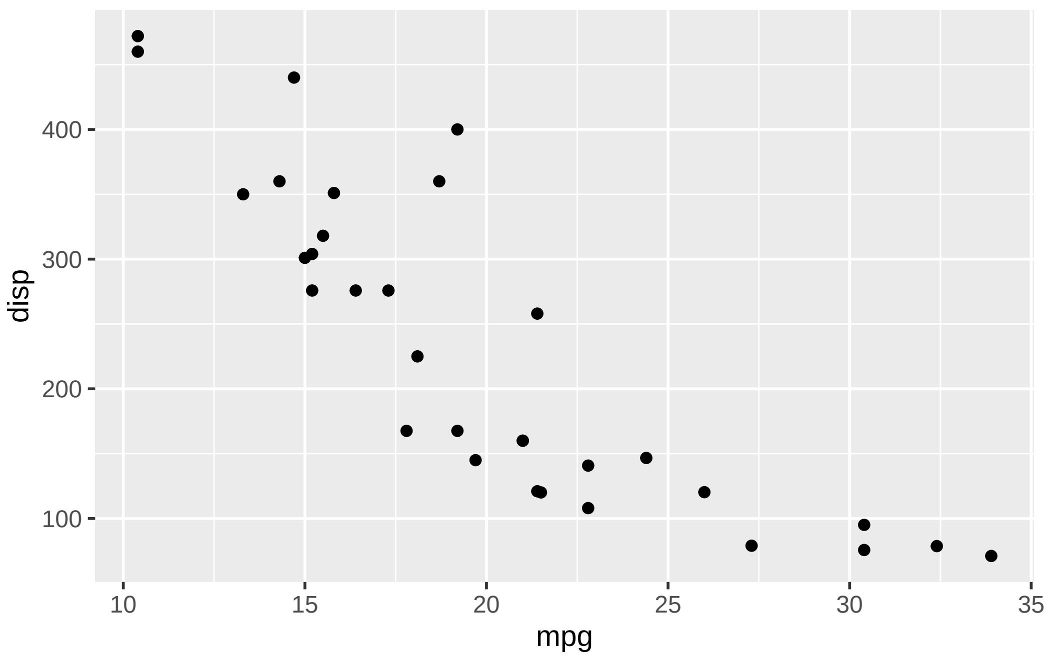
```r
#scaling finds the mean center - find the mean of all the data and subtract it from zero
```

```r
round(apply(mtscale, 2, sd), 3)
```

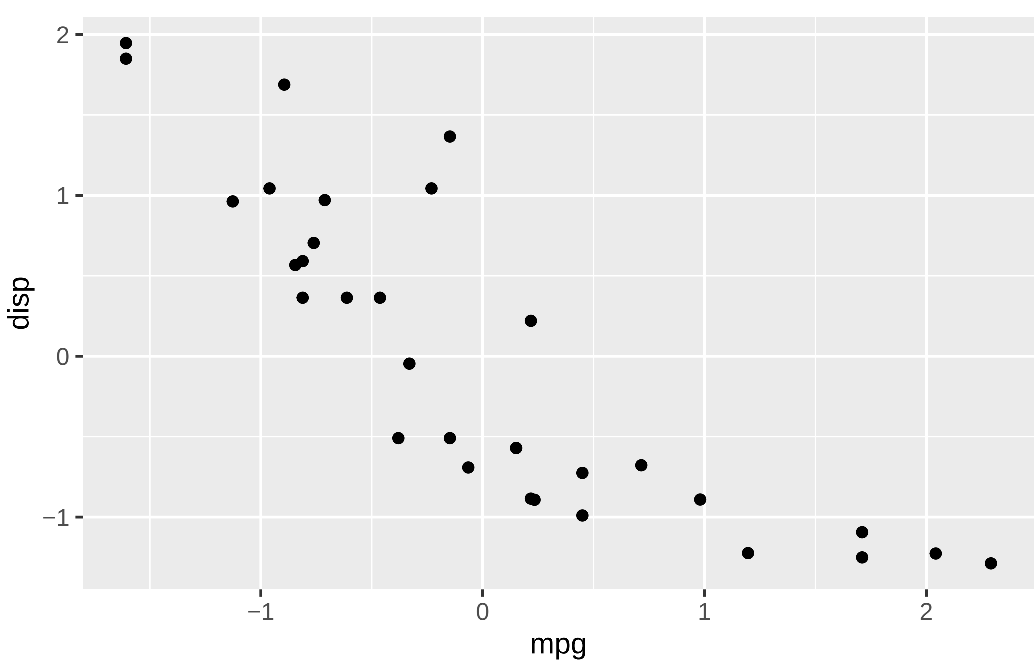
```
 mpg  cyl disp   hp drat   wt qsec   vs   am gear carb 
   1    1    1    1    1    1    1    1    1    1    1 
```

Let's plot `mpg` vs `disp` for both mtcars and the scaled version of it (mtscale).

```r
library(ggplot2)

ggplot(mtcars) +
  aes(mpg, disp) +
  geom_point()
```

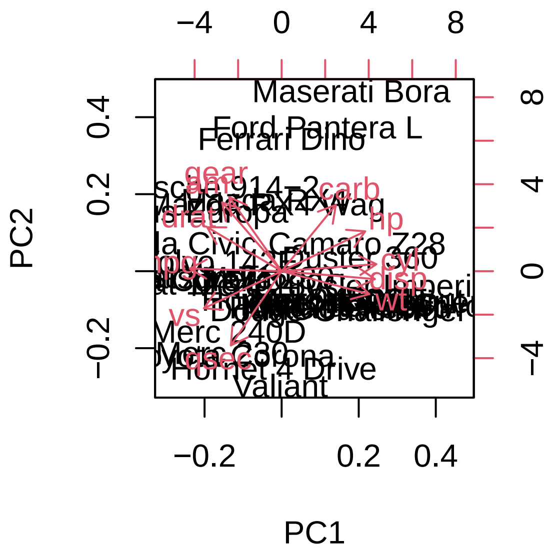



```r
ggplot(mtscale) +
  aes(mpg, disp) +
  geom_point()
```



The only difference is that it is centerd at zero. doesnt change the relationships between the data - it just scales it.

```r
pca2 <- prcomp(mtscale)
biplot(pca2)
```



More fair representation of all the cars because its not being dominated by different units.

##Breast Cancer FNA data # were going to do PCA/clustering on this data

```r
# first step is to download the csv file and save it in the same directory as your script
fna.data <- "WisconsinCancer.csv"
wisc.df <- read.csv(fna.data, row.names=1) # what does row.names=1 mean? You set the row nam
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
```

Remove the first column from the data set

Q1. How many rows/patients/subjust.

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

```
[1] 569
```

How many malignants are there? "M"

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

```

  B   M 
357 212 
```

Get rid of diagnosis column

```
wisc.data <- wisc.df[,-1] # gets rid of first column
diagnosis <-as.factor(wisc.df$diagnosis)  # benign or malignant, save as a factor
head(wisc.data)
```

```
         radius_mean texture_mean perimeter_mean area_mean smoothness_mean
842302         17.99        10.38         122.80    1001.0         0.11840
842517         20.57        17.77         132.90    1326.0         0.08474
84300903       19.69        21.25         130.00    1203.0         0.10960
84348301       11.42        20.38          77.58     386.1         0.14250
84358402       20.29        14.34         135.10    1297.0         0.10030
843786         12.45        15.70          82.57     477.1         0.12780
         compactness_mean concavity_mean concave.points_mean symmetry_mean
842302            0.27760         0.3001             0.14710        0.2419
842517            0.07864         0.0869             0.07017        0.1812
84300903          0.15990         0.1974             0.12790        0.2069
84348301          0.28390         0.2414             0.10520        0.2597
```

```
84358402          0.13280        0.1980             0.10430        0.1809
843786            0.17000        0.1578             0.08089        0.2087
         fractal_dimension_mean radius_se texture_se perimeter_se area_se
842302                  0.07871    1.0950     0.9053        8.589  153.40
842517                  0.05667    0.5435     0.7339        3.398   74.08
84300903                0.05999    0.7456     0.7869        4.585   94.03
84348301                0.09744    0.4956     1.1560        3.445   27.23
84358402                0.05883    0.7572     0.7813        5.438   94.44
843786                  0.07613    0.3345     0.8902        2.217   27.19
         smoothness_se compactness_se concavity_se concave.points_se
842302        0.006399        0.04904      0.05373           0.01587
842517        0.005225        0.01308      0.01860           0.01340
84300903      0.006150        0.04006      0.03832           0.02058
84348301      0.009110        0.07458      0.05661           0.01867
84358402      0.011490        0.02461      0.05688           0.01885
843786        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
```

```
# now there is no diagnosis column, because we dont want to include that in our analysis

#will compare with it at the end
```

Useful functions: table(), grep() -> finds matching patterns

Q3. How many variables/features (can by called by colnames()) in the data are suffixed with _mean?

```
#colnames(wisc.data)
length(grep("_mean", colnames(wisc.data), value="T"))
```

```
[1] 10
```

2 PCA Principle Component Analysis

We want to scale our data before PCA by setting the `scale=True` argument.

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

How much variance is captured in each Principle component?

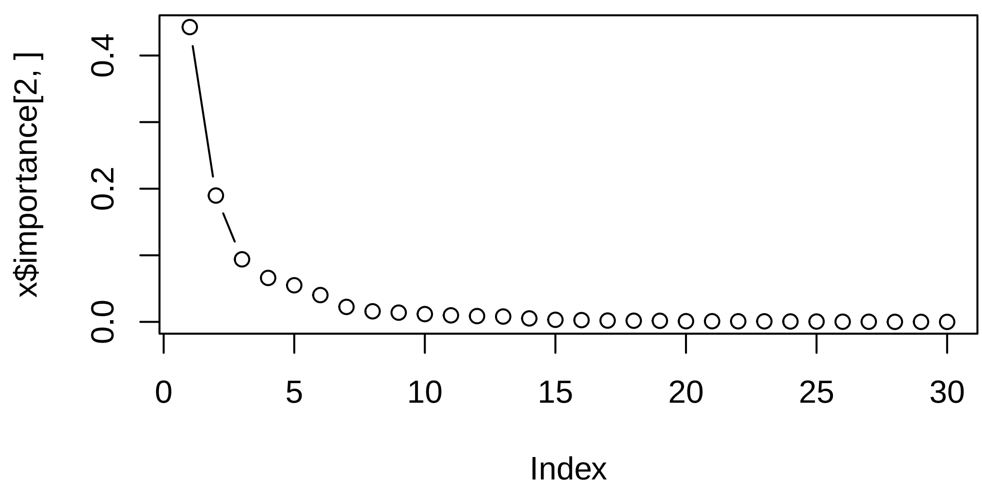
```
x <-summary(wisc.pr)
x$importance
```

```
                            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
```

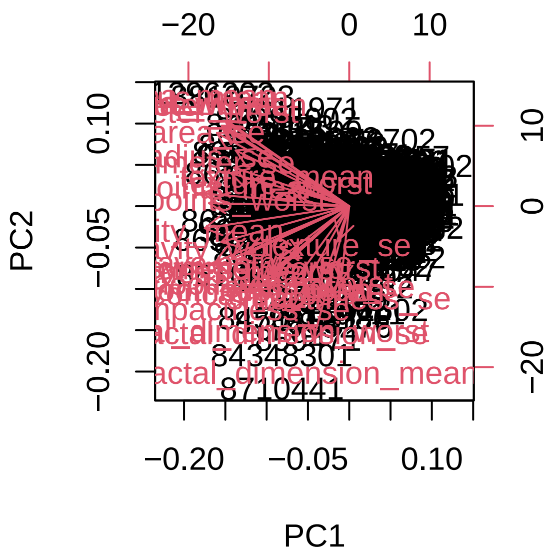
```
# plot variance against PC and look for elbow point

# look here at the summulative proportion numbers
```

```
plot(x$importance[2,], typ="b")
```



Elbow happens around index 3.

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




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

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

$class
[1] "prcomp"
```

x is what we're after

```r
head(wisc.pr$x)
```

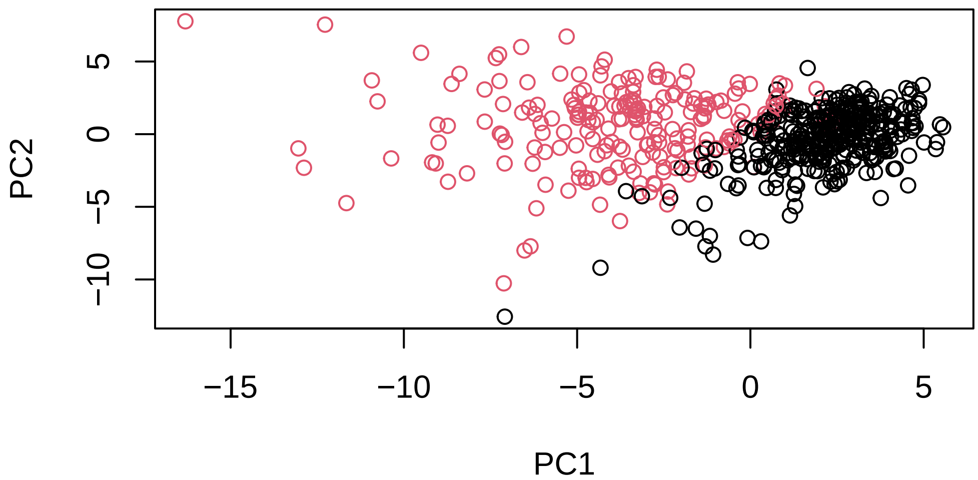
```
               PC1        PC2        PC3       PC4        PC5         PC6
842302   -9.184755  -1.946870 -1.1221788 3.6305364  1.1940595  1.41018364
842517   -2.385703   3.764859 -0.5288274 1.1172808 -0.6212284  0.02863116
84300903 -5.728855   1.074229 -0.5512625 0.9112808  0.1769302  0.54097615
84348301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

These our the coordinates of the patients on the new axis main pca plot could plot whatever biplot of pcx vs pcy that you want

My main PC result figure (cordination plot)

```
plot(wisc.pr$x, col=diagnosis) # will plot pc1 vs pc2, first two columns
```



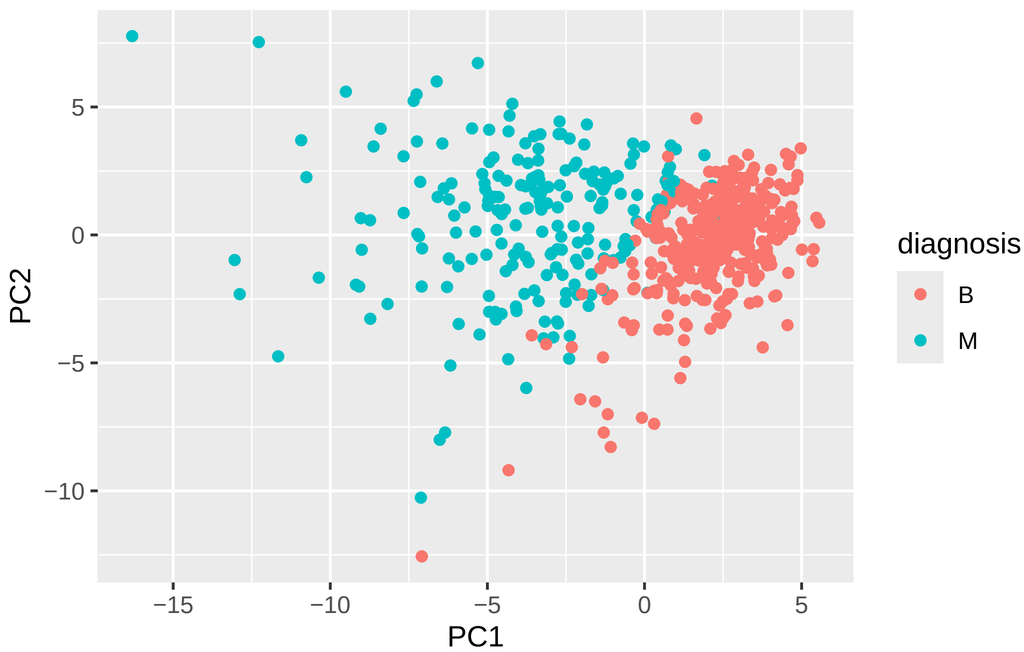

```
# colour red for malignant, bengin for black
```

Still dont understand PC - > what is PC1? made up of many factors the point is to reduce the dimensionality of the data, to figure out which key factors make up PC1 ie explain most of the variation

Each point represents a patient.

Points with little influence are closer to 0.

```
# create a data frame to plot

df <- as.data.frame(wisc.pr$x) # just the x column
df$diagnosis <- diagnosis

library(ggplot2)

# Make a scatter plot and colour by diagnosis

# just want coordinates a diagnosis so thats why we are creating our own new data frame.

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




```r
head(df)
```

```
               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 diagnosis
842302   -0.0338846387  0.045607590  0.0471277407         M
842517    0.0325955021 -0.005682424  0.0018662342         M
84300903  0.0469844833  0.003143131 -0.0007498749         M
84348301  0.0424469831 -0.069233868  0.0199198881         M
84358402 -0.0347556386  0.005033481 -0.0211951203         M
843786    0.0007296587 -0.019703996 -0.0034564331         M
```

Varience explained

```
# calculate the varience of each principle component

pr.var <- wisc.pr$sdev^2 # what is wisc.pr again? a table of the PC's and their standard devi
head(pr.var)
```

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

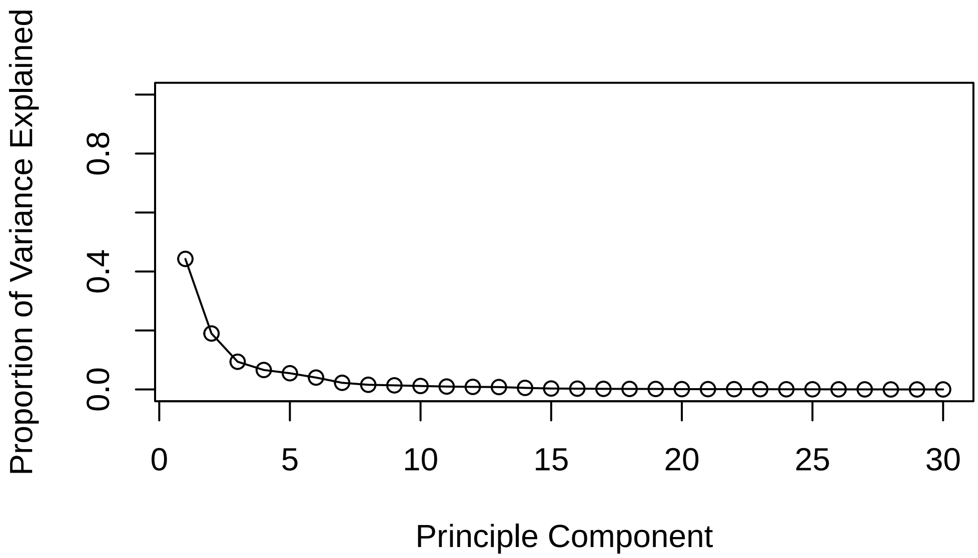
Now we will calculate the varience explained by each PC by dividing by the total variance explained by all PCs

```
pve <- (pr.var )/(sum(pr.var ))
pve
```

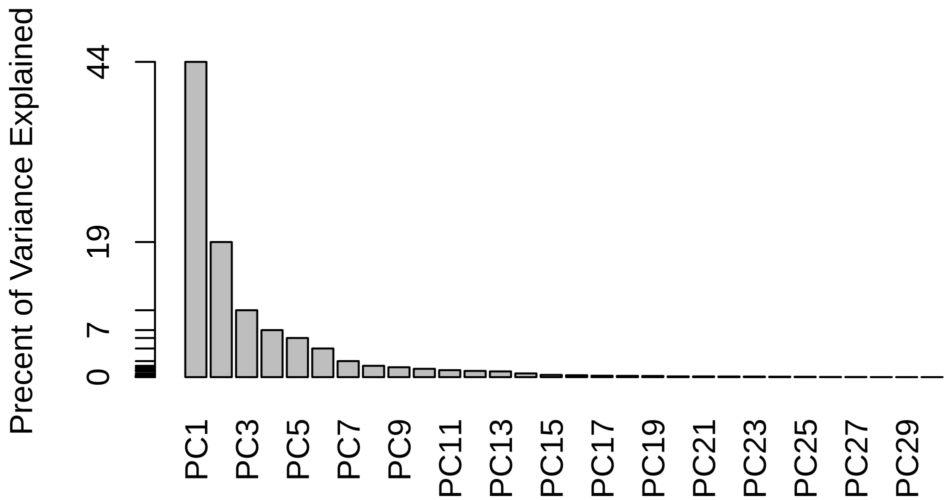
```
 [1] 4.427203e-01 1.897118e-01 9.393163e-02 6.602135e-02 5.495768e-02
```

```
 [6] 4.024522e-02 2.250734e-02 1.588724e-02 1.389649e-02 1.168978e-02
[11] 9.797190e-03 8.705379e-03 8.045250e-03 5.233657e-03 3.137832e-03
[16] 2.662093e-03 1.979968e-03 1.753959e-03 1.649253e-03 1.038647e-03
[21] 9.990965e-04 9.146468e-04 8.113613e-04 6.018336e-04 5.160424e-04
[26] 2.725880e-04 2.300155e-04 5.297793e-05 2.496010e-05 4.434827e-06
```

```r
# now plot the varience explained by each pc: pve
plot(pve, xlab="Principle Component", ylab = "Proportion of Variance Explained", ylim = c(0,1
```




```r
# 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 )
```



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? This tells us how much this original feature contributes to the first PC.

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

```
concave.points_mean
         -0.2608538
```

Hierarchical Clustering

We are going to do hierarchical clustering of the original data This kind of analysis doesnt require us to know how many cluster there should be in advance - unlike kmeans clustering.

First scale the wisc.data data and assign the result to data.scaled

```
head(wisc.data)
```

```
         radius_mean texture_mean perimeter_mean area_mean smoothness_mean
842302         17.99        10.38         122.80    1001.0         0.11840
842517         20.57        17.77         132.90    1326.0         0.08474
84300903       19.69        21.25         130.00    1203.0         0.10960
```

```
84348301       11.42        20.38           77.58     386.1          0.14250
84358402       20.29        14.34          135.10    1297.0          0.10030
843786         12.45        15.70           82.57     477.1          0.12780
         compactness_mean concavity_mean concave.points_mean symmetry_mean
842302            0.27760         0.3001             0.14710        0.2419
842517            0.07864         0.0869             0.07017        0.1812
84300903          0.15990         0.1974             0.12790        0.2069
84348301          0.28390         0.2414             0.10520        0.2597
84358402          0.13280         0.1980             0.10430        0.1809
843786            0.17000         0.1578             0.08089        0.2087
         fractal_dimension_mean radius_se texture_se perimeter_se area_se
842302                  0.07871    1.0950     0.9053        8.589  153.40
842517                  0.05667    0.5435     0.7339        3.398   74.08
84300903                0.05999    0.7456     0.7869        4.585   94.03
84348301                0.09744    0.4956     1.1560        3.445   27.23
84358402                0.05883    0.7572     0.7813        5.438   94.44
843786                  0.07613    0.3345     0.8902        2.217   27.19
         smoothness_se compactness_se concavity_se concave.points_se
842302        0.006399        0.04904      0.05373           0.01587
842517        0.005225        0.01308      0.01860           0.01340
84300903      0.006150        0.04006      0.03832           0.02058
84348301      0.009110        0.07458      0.05661           0.01867
84358402      0.011490        0.02461      0.05688           0.01885
843786        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
```

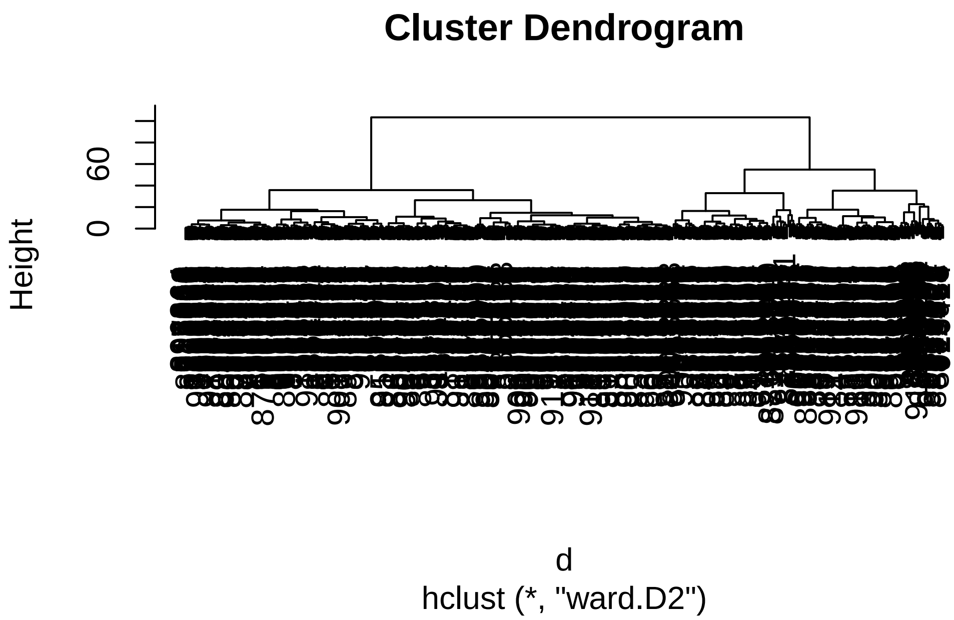
Try to cluster the wisc.data

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

```
  1   2
438 131
```

In other words use my PCA results as a basis of clustering. PCA is giving some signal. now we will cluster based on that signal

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




```
#use these moved variables , pc1, pc2, pc3  as input to cluster rather than the original data
```

Cut this tree to yeild s groups/clusters

```
grps <- cutree(hc, k=2)
table(grps)
```

```
grps
  1   2 
203 366 
```

Compare to my expert M and B diagnosis

```
table(diagnosis)
```

```
diagnosis
  B   M 
357 212 
```

Cross table

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

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

179+33 = 212 and the vast majority are cluster 1 this table shows how the clustering and expert diagnosis correspond.

Getting 179 correct, and 33 not correct - figuring out the false positives ideally want to get all M's into cluster 1 so you are 100% good at catching all M's

Trade-off between sensitivity and specificity.

do up to Q12

3 Hierarchical Clustering

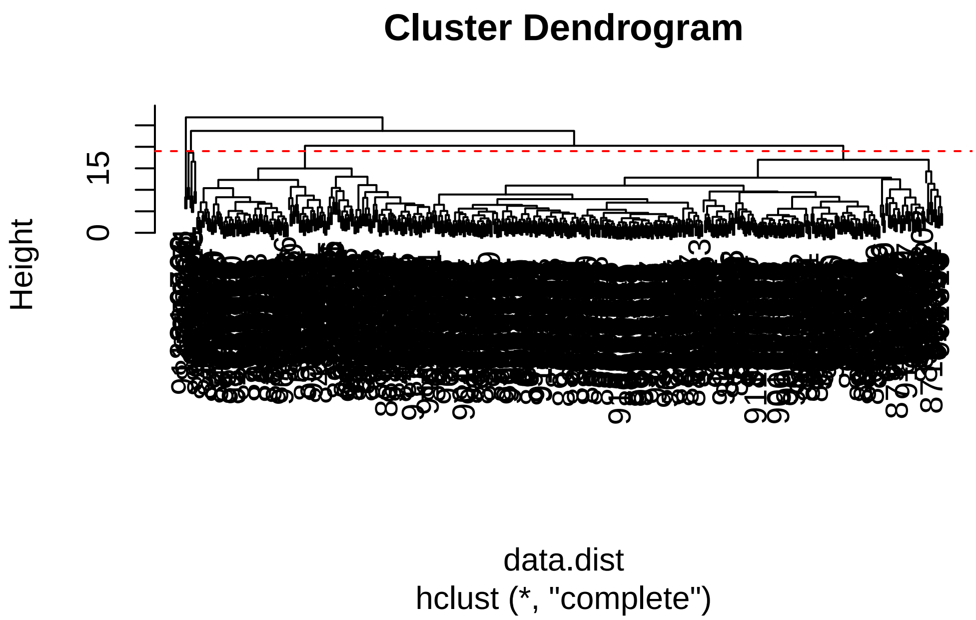
First scale the wisc.data

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

# now calculate the distance between all pairs in the scaled version
data.dist <- dist(data.scaled, method = "euclidean")

# create a hierarchical clustering model

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

Results of hierarchical clustering

> 10 What is the height at which the clustering model has 4 clusters

```
plot(wisc.hclust)
abline(19, 0, col="red", lty=2)
```



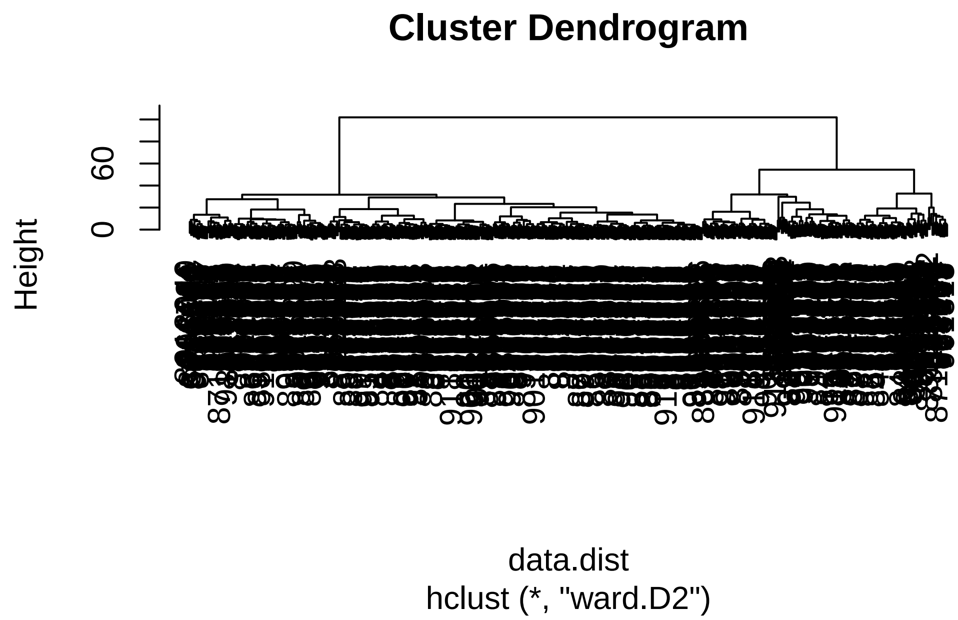
It happens at height 19

Using different methods

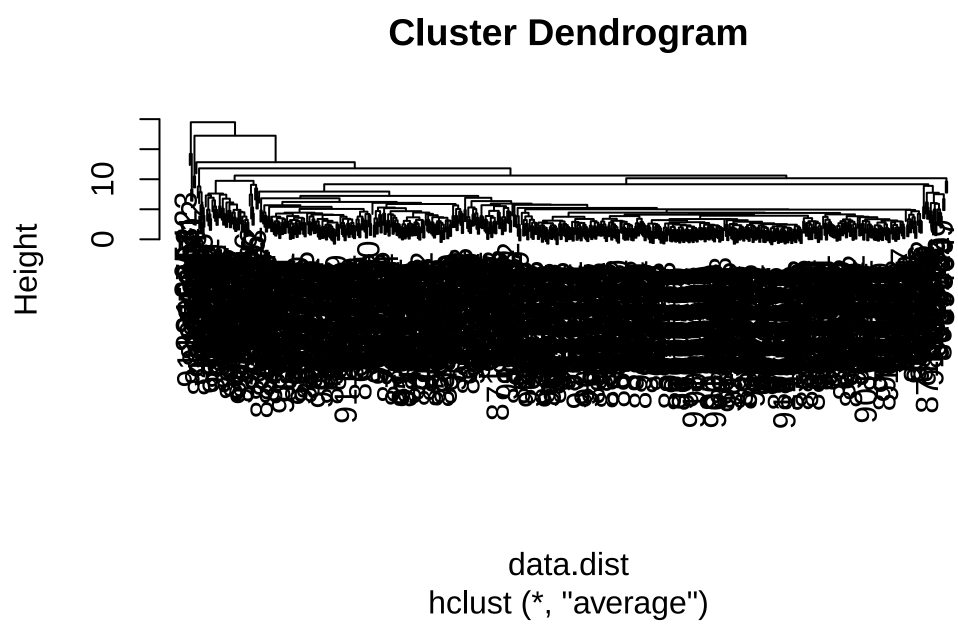
As we discussed in our last class videos there are number of different “methods” we can use to combine points during the hierarchical clustering procedure. These include “single”, “complete”, “average” and (my favorite) “ward.D2” >Q12 Which method gives your favorite results for the same data.dist dataset? Explain your reasoning.

“Complete” and “ward.D2” are my favourites because they produce the trees that are the easiest to read.

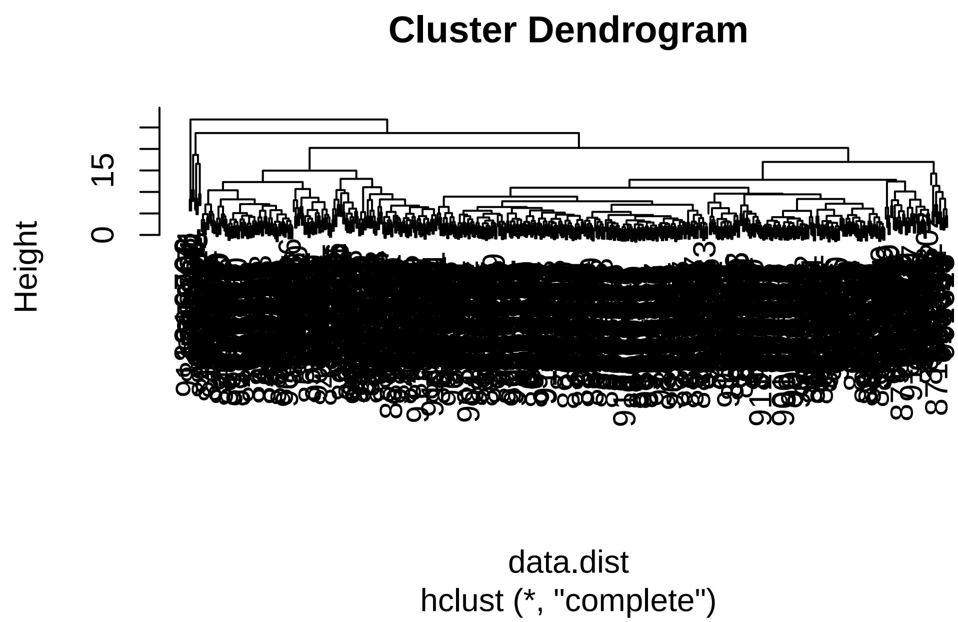
```
wisc.hclust <- hclust(data.dist, method="ward.D2")
plot(wisc.hclust)
```



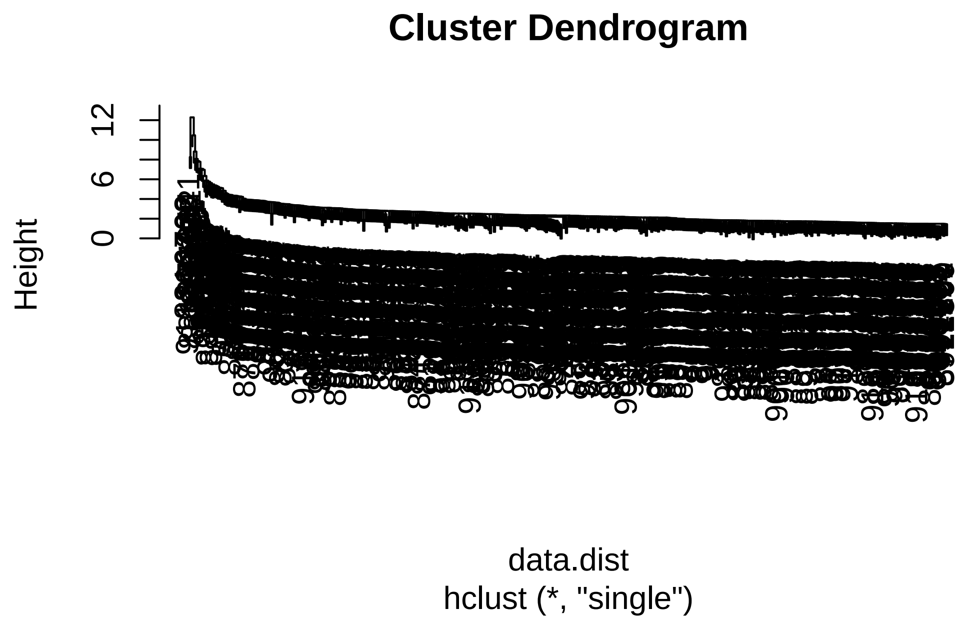

```
wisc.hclust <- hclust(data.dist, method="average")
plot(wisc.hclust)
```



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




```r
wisc.hclust <- hclust(data.dist, method="single")
plot(wisc.hclust)
```



stop at ## 4 combining methods