

class_08_miniproject

Kyle Alvarez

Class 8 Mini-Project

Exploratory data analysis

Download the WinsconsinCancer.csv file from the class website, and import it into our class_08_miniproject folder.

```
#Can now use read.csv to read the data within the csv file.  
#read.csv("WisconsinCancer.csv")
```

Save the input data file into the project directory and inptu the data and store as `wisc.df`

```
fna.data <- "WisconsinCancer.csv"  
wisc.df <- read.csv(fna.data, row.names=1)
```

Create a new data.frame that omits the first column `wisc.df$diagnosis`.

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

Create a separate vector called `diagnosis` that stores the data from the diagnosis column of the original dataset. Will be used as a factor later.

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

Q1. How many observations are in this dataset?

569 observations are in this dataset.

```
nrow(wisc.data)
```

```
[1] 569
```

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

212 malignant diagnoses.

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

```
  B   M  
357 212
```

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

There are 10 variables/features in the data are suffixed with `_mean`.

```
matches <- colnames(wisc.data, prefix = "_mean")  
length(grep("_mean", matches))
```

```
[1] 10
```

Principal Component Analysis

Perform PCA on `wisc.data` Must check if the data must be scaled because - the input variables use different units of measurement - the input variables have significant different variances Can use `colMeans()` and `apply()`

```
# Check column means and standard deviations  
colMeans((wisc.data))
```

radius_mean	texture_mean	perimeter_mean
1.412729e+01	1.928965e+01	9.196903e+01
area_mean	smoothness_mean	compactness_mean
6.548891e+02	9.636028e-02	1.043410e-01
concavity_mean	concave.points_mean	symmetry_mean
8.879932e-02	4.891915e-02	1.811619e-01
fractal_dimension_mean	radius_se	texture_se
6.279761e-02	4.051721e-01	1.216853e+00
perimeter_se	area_se	smoothness_se
2.866059e+00	4.033708e+01	7.040979e-03
compactness_se	concavity_se	concave.points_se

2.547814e-02	3.189372e-02	1.179614e-02
symmetry_se	fractal_dimension_se	radius_worst
2.054230e-02	3.794904e-03	1.626919e+01
texture_worst	perimeter_worst	area_worst
2.567722e+01	1.072612e+02	8.805831e+02
smoothness_worst	compactness_worst	concavity_worst
1.323686e-01	2.542650e-01	2.721885e-01
concave.points_worst	symmetry_worst	fractal_dimension_worst
1.146062e-01	2.900756e-01	8.394582e-02

```
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

Execute PCA with `prcomp()` function on the `wisc.data` and scale if needed

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

Inspect the summary of results with `summary()`

```
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 proportion of the original variance that was capture by PC1 was 44.27%.

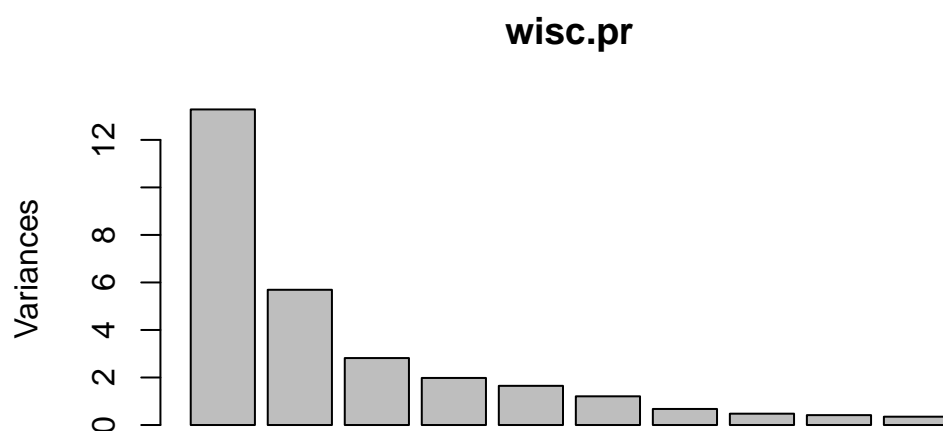
Q5. How many principal components (PCs) are required to describe at least 70% of the original variance in the data?

3 PCs are required to describe at least 70% of the original variance in the data.

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

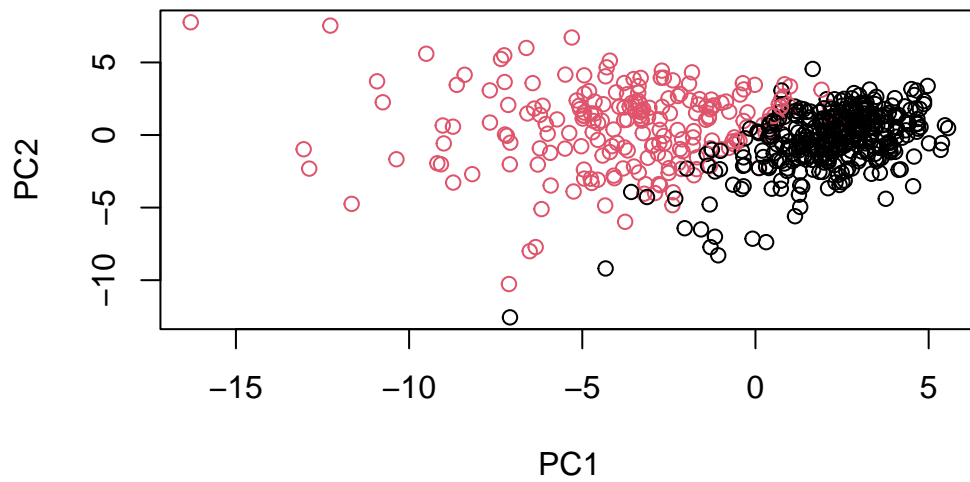
7 PCs are required to describe at least 90% of the original variance in the data.

```
plot(wisc.pr)
```



Let's make a PC plot (a.k.a. "score plot" or "PC1 vs PC2" etc. plot)

```
plot(wisc.pr$x[,1], wisc.pr$x[,2], col = diagnosis, xlab = "PC1", ylab = "PC2")
```

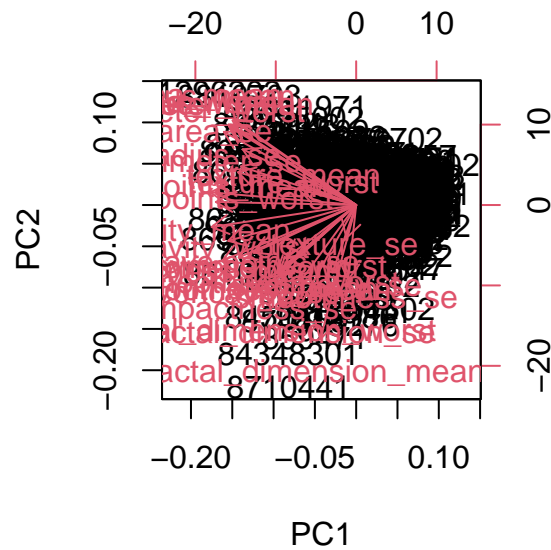


Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

The plot shows the two different diagnoses, malignant and benign. It is difficult to understand because there are so many points, and sections where it is just all clustered.

Create a bi-plot of the `wisc.pr`

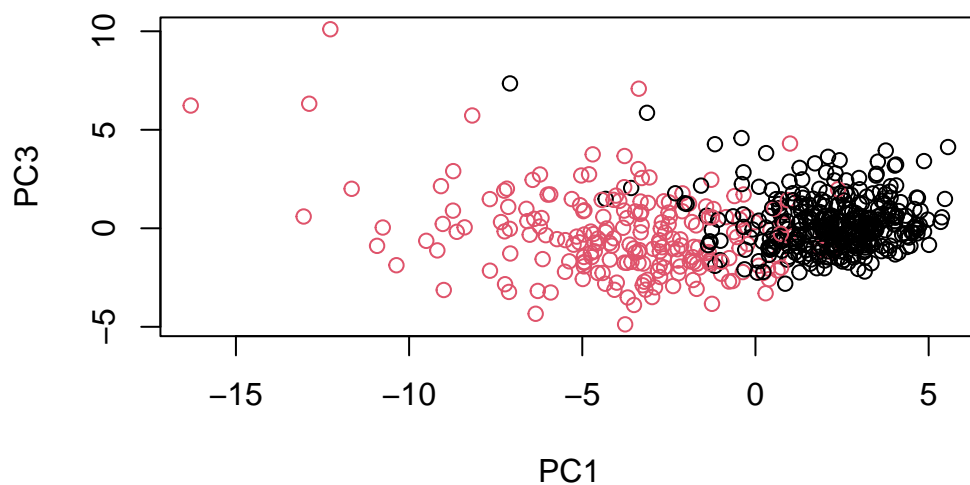
```
biplot(wisc.pr)
```



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

The points on this plot are closer to the PC1 axis and are more closer together compared to the PC1 vs PC2 plot where they were further up from PC1 and further apart. This makes sense because PC2 explains more variance in the data compared to data 3.

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

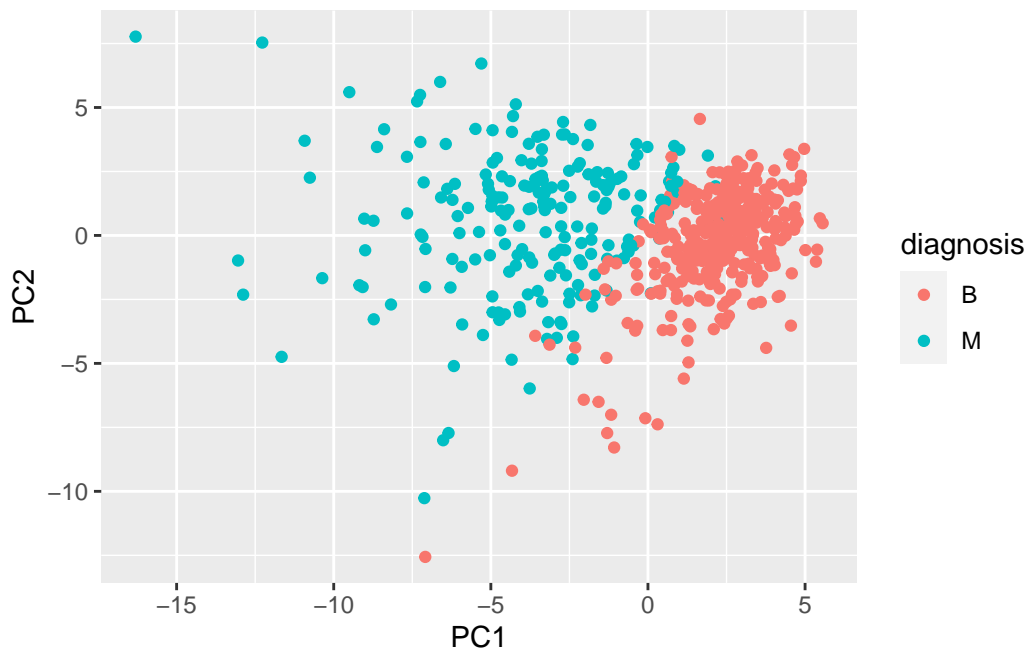


Use ggplot2 to make a more fancy figure of the results.

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

# Load the ggplot2
library(ggplot2)

# Make a scatter plot colored by diagnosis
ggplot(df) +
  aes(PC1, PC2, col = diagnosis) +
  geom_point()
```

Calculate the variance of each principal component by squaring the stdev component of `wisc.pr` (i.e. `wisc.pr$sdev^2`).

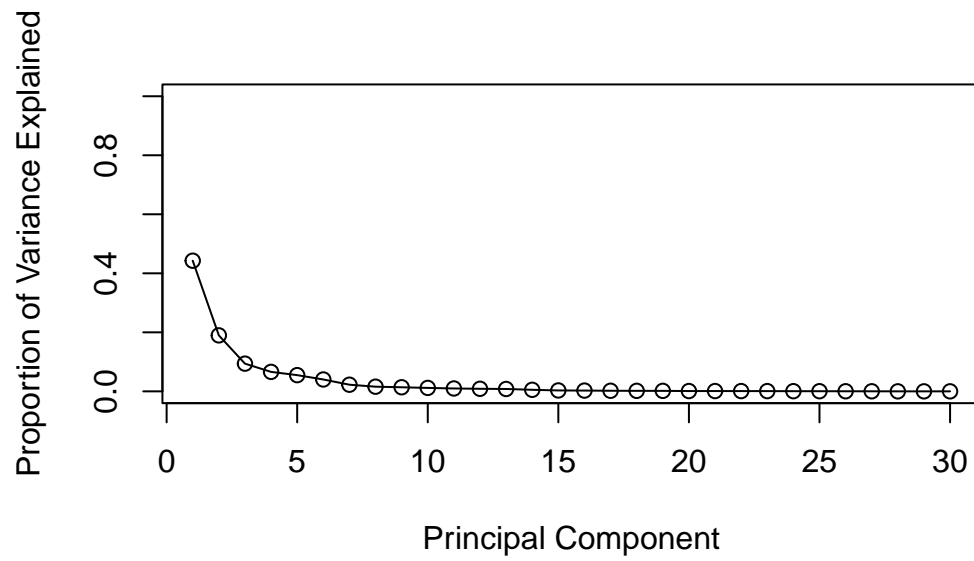
```
# Calculate the variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

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

Calculate the variance explained by each principal component by dividing by the total variance explained of all principal components. We can also plot the variance explained for each principal component after calculating it.

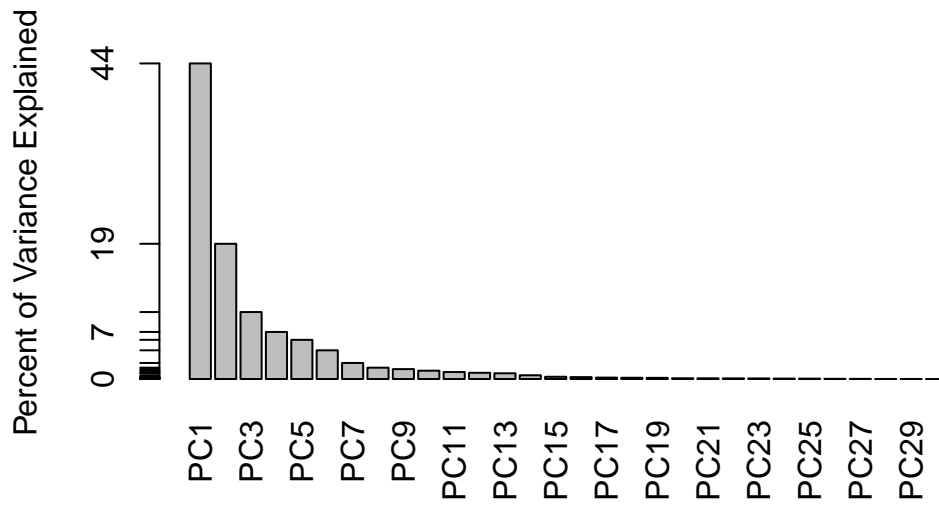
```
# Variance explained by each principal component: pve
pve <- pr.var / sum(pr.var)

# Plot the variance explained for each principal component
plot(pve, xlab = "Principal Component", ylab="Proportion of Variance Explained", ylim = c(0, 1))
```



We can make a alternative scree plot of the same data.

```
barplot(pve, ylab = "Percent of Variance Explained", names.arg=paste0("PC",1:length(pve)),  
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`?

The component is -0.26085376

```
wisc.pr$rotation[,1]
```

radius_mean	texture_mean	perimeter_mean
-0.21890244	-0.10372458	-0.22753729
area_mean	smoothness_mean	compactness_mean
-0.22099499	-0.14258969	-0.23928535
concavity_mean	concave.points_mean	symmetry_mean
-0.25840048	-0.26085376	-0.13816696
fractal_dimension_mean	radius_se	texture_se
-0.06436335	-0.20597878	-0.01742803
perimeter_se	area_se	smoothness_se
-0.21132592	-0.20286964	-0.01453145
compactness_se	concavity_se	concave.points_se
-0.17039345	-0.15358979	-0.18341740
symmetry_se	fractal_dimension_se	radius_worst
-0.04249842	-0.10256832	-0.22799663
texture_worst	perimeter_worst	area_worst
-0.10446933	-0.23663968	-0.22487053

smoothness_worst	compactness_worst	concavity_worst
-0.12795256	-0.21009588	-0.22876753
concave.points_worst	symmetry_worst	fractal_dimension_worst
-0.25088597	-0.12290456	-0.13178394

Q10. What is the minimum number of principal components required to explain 80% of the variance of the data?

4 principal components are required to explain 80% of the variance in the data.

```
y <- summary(wisc.pr)
attributes(y)
```

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

$class
[1] "summary.prcomp"
```

```
sum(y$importance[3,] <= 0.8)
```

```
[1] 4
```

Hierarchical Clustering

Scale the `wisc.data` data and assign the result to `data.scaled`.

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

Calculate the (Euclidean) distances between all pairs of observations in the new scaled dataset and assign the result to `data.dist`.

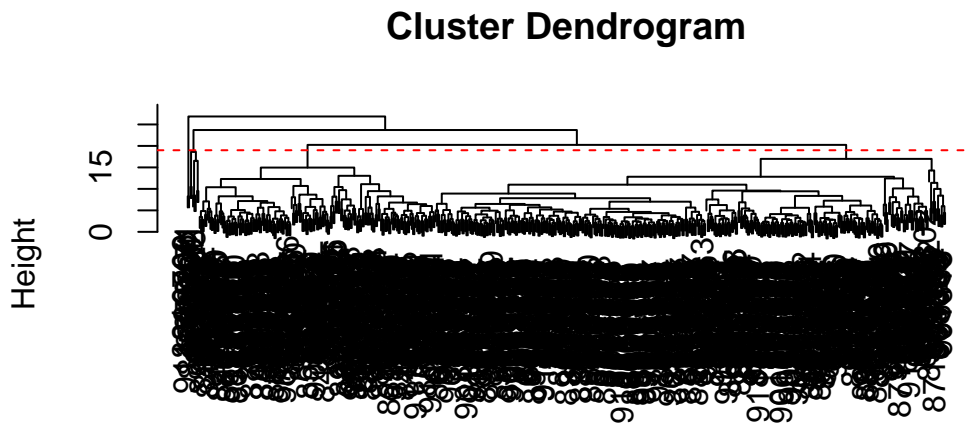
```
data.dist <- dist(data.scaled)
```

Create a hierarchical clustering model using complete linkage. Manually specify the method argument to `hclust()` and assign the results to `wisc.hclust`.

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

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

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



```
data.dist
hclust (*, "complete")
```

Selecting a number of clusters

Use `cutree()` to cut the tree so that it has 4 clusters. Assign the output to the variable `wisc.hclust.clusters`

```
wisc.hclust.clusters <- cutree(wisc.hclust, k = 4)
wisc.hclust.clusters2 <- cutree(wisc.hclust, k = 10)
```

Can use the `table()` function to compare 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

```

```
table(wisc.hclust.clusters2, diagnosis)
```

```

      diagnosis
wisc.hclust.clusters2  B  M
1      12 86
2       0 59
3       0  3
4     331 39
5       0 20
6       2  0
7      12  0
8       0  2
9       0  2
10      0  1

```

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

Yes you can find a better cluster vs diagnoses match by cutting into a different number of cluster between 2 and 10, probably 10 as shows it shows the distribution more towards the other clusters rather than it being all on cluster 1 and 3.

Using different methods

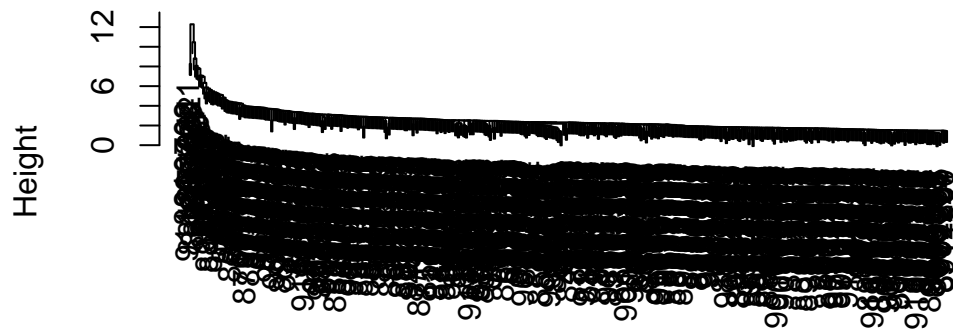
```

wisc.hclust1 <- hclust(data.dist, method = "single")
wisc.hclust2 <- hclust(data.dist, method = "complete")
wisc.hclust3 <- hclust(data.dist, method = "average")
wisc.hclust4 <- hclust(data.dist, method = "ward.D2")

plot(wisc.hclust1)

```

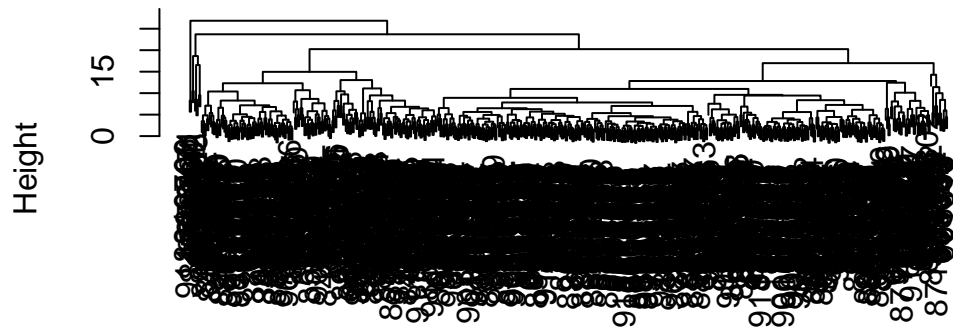
Cluster Dendrogram



```
data.dist  
hclust (*, "single")
```

```
plot(wisc.hclust2)
```

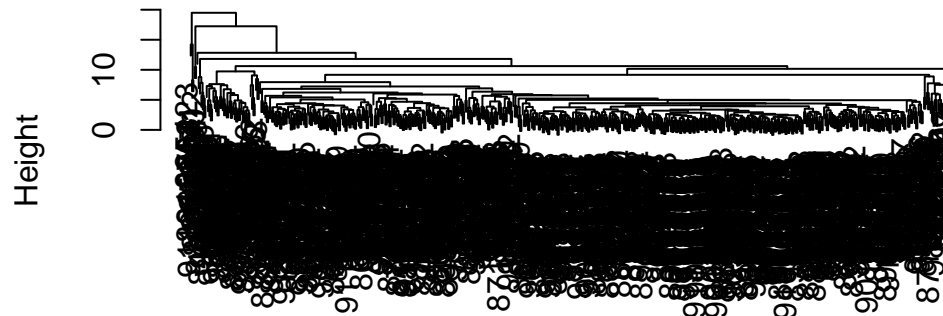
Cluster Dendrogram



```
data.dist  
hclust (*, "complete")
```

```
plot(wisc.hclust3)
```

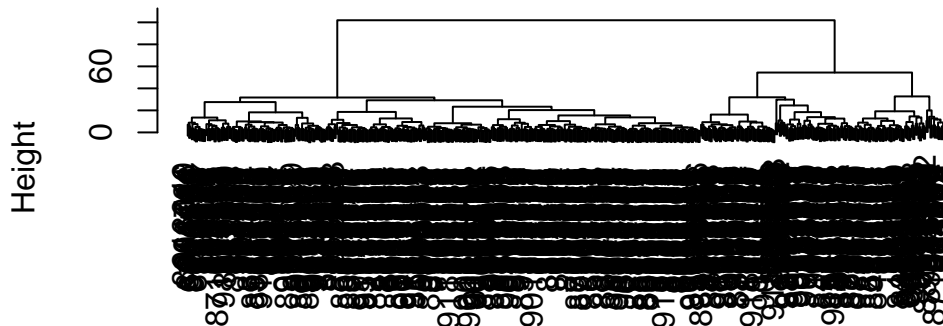
Cluster Dendrogram



```
data.dist  
hclust (*, "average")
```

```
plot(wisc.hclust4)
```


Cluster Dendrogram



```
data.dist
hclust (*, "ward.D2")
```

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

The `ward.d2` method is my favorite result for the same data.dist dataset because you can see the major cluster that continuously divides into the smaller clusters, showing the actual hierarchical connections between the data.

OPTIONAL: K-means clustering

Create k-means model on `wisc.data`, assigning the result to `wisc.km`,

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

K-means clustering with 2 clusters of sizes 189, 380

Cluster means:

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean
1	0.9731199	0.4810905	1.0057496	0.9626801	0.6087185
2	-0.4839991	-0.2392792	-0.5002281	-0.4788067	-0.3027573

	compactness_mean	concavity_mean	concave.points_mean	symmetry_mean
1	1.0197987	1.138428	1.1635583	0.6106013

2	-0.5072157	-0.566218	-0.5787172	-0.3036938	
	fractal_dimension_mean	radius_se	texture_se	perimeter_se	area_se
1	0.2520081	0.8578415	0.04270321	0.8595226	0.8063982
2	-0.1253409	-0.4266633	-0.02123923	-0.4274994	-0.4010770
	smoothness_se	compactness_se	concavity_se	concave.points_se	symmetry_se
1	0.01704563	0.6944395	0.6363352	0.7755561	0.1402588
2	-0.00847796	-0.3453923	-0.3164930	-0.3857371	-0.0697603
	fractal_dimension_se	radius_worst	texture_worst	perimeter_worst	area_worst
1	0.4146673	1.039169	0.5058654	1.0650336	1.0022723
2	-0.2062424	-0.516850	-0.2516015	-0.5297141	-0.4984986
	smoothness_worst	compactness_worst	concavity_worst	concave.points_worst	
1	0.6077580	0.9500013	1.0433804	1.145203	
2	-0.3022796	-0.4725007	-0.5189444	-0.569588	
	symmetry_worst	fractal_dimension_worst			
1	0.5968910	0.6219221			
2	-0.2968747	-0.3093244			

Clustering vector:

842302	842517	84300903	84348301	84358402	843786	844359	84458202
1	1	1	1	1	1	1	1
844981	84501001	845636	84610002	846226	846381	84667401	84799002
1	1	2	1	1	2	1	1
848406	84862001	849014	8510426	8510653	8510824	8511133	851509
2	1	1	2	2	2	1	1
852552	852631	852763	852781	852973	853201	853401	853612
1	1	1	1	1	1	1	1
85382601	854002	854039	854253	854268	854941	855133	855138
1	1	1	1	1	2	2	2
855167	855563	855625	856106	85638502	857010	85713702	85715
2	2	1	2	2	1	2	1
857155	857156	857343	857373	857374	857392	857438	85759902
2	2	2	2	2	1	2	2
857637	857793	857810	858477	858970	858981	858986	859196
1	1	2	2	2	2	1	2
85922302	859283	859464	859465	859471	859487	859575	859711
1	1	2	2	1	2	1	2
859717	859983	8610175	8610404	8610629	8610637	8610862	8610908
1	2	2	1	2	1	1	2
861103	8611161	8611555	8611792	8612080	8612399	86135501	86135502
2	1	1	1	2	1	2	1
861597	861598	861648	861799	861853	862009	862028	86208
2	1	2	2	2	2	1	1
86211	862261	862485	862548	862717	862722	862965	862980

2	2	2	2	2	2	2	2
862989	863030	863031	863270	86355	864018	864033	86408
2	1	2	2	1	2	2	2
86409	864292	864496	864685	864726	864729	864877	865128
1	2	2	2	2	1	1	2
865137	86517	865423	865432	865468	86561	866083	866203
2	1	1	2	2	2	2	1
866458	866674	866714	8670	86730502	867387	867739	868202
1	1	2	1	1	2	1	2
868223	868682	868826	868871	868999	869104	869218	869224
2	2	1	2	2	1	2	2
869254	869476	869691	86973701	86973702	869931	871001501	871001502
2	2	1	2	2	2	2	2
8710441	87106	8711002	8711003	8711202	8711216	871122	871149
1	2	2	2	1	2	2	2
8711561	8711803	871201	8712064	8712289	8712291	87127	8712729
2	1	1	2	1	2	2	1
8712766	8712853	87139402	87163	87164	871641	871642	872113
1	2	2	2	1	2	2	2
872608	87281702	873357	873586	873592	873593	873701	873843
2	1	2	2	1	1	2	2
873885	874158	874217	874373	874662	874839	874858	875093
2	2	2	2	2	2	1	2
875099	875263	87556202	875878	875938	877159	877486	877500
2	1	1	2	1	1	1	1
877501	877989	878796	87880	87930	879523	879804	879830
2	1	1	1	2	2	2	2
8810158	8810436	881046502	8810528	8810703	881094802	8810955	8810987
2	2	1	2	1	1	1	1
8811523	8811779	8811842	88119002	8812816	8812818	8812844	8812877
2	2	1	1	2	2	2	1
8813129	88143502	88147101	88147102	88147202	881861	881972	88199202
2	2	2	2	2	1	1	2
88203002	88206102	882488	88249602	88299702	883263	883270	88330202
2	1	2	2	1	1	2	1
88350402	883539	883852	88411702	884180	884437	884448	884626
2	2	1	2	1	2	2	2
88466802	884689	884948	88518501	885429	8860702	886226	886452
2	2	1	2	1	1	1	2
88649001	886776	887181	88725602	887549	888264	888570	889403
1	1	1	1	1	2	1	2
889719	88995002	8910251	8910499	8910506	8910720	8910721	8910748
1	1	2	2	2	2	2	2

8910988	8910996	8911163	8911164	8911230	8911670	8911800	8911834
1	2	2	2	2	2	2	2
8912049	8912055	89122	8912280	8912284	8912521	8912909	8913
1	2	1	1	2	2	2	2
8913049	89143601	89143602	8915	891670	891703	891716	891923
2	2	1	2	2	2	2	2
891936	892189	892214	892399	892438	892604	89263202	892657
2	2	2	2	1	2	1	2
89296	893061	89344	89346	893526	893548	893783	89382601
2	2	2	2	2	2	2	2
89382602	893988	894047	894089	894090	894326	894329	894335
2	2	2	2	2	1	1	2
894604	894618	894855	895100	89511501	89511502	89524	895299
2	1	2	1	2	2	2	2
8953902	895633	896839	896864	897132	897137	897374	89742801
1	1	1	2	2	2	2	1
897604	897630	897880	89812	89813	898143	89827	898431
2	1	2	1	2	2	2	1
89864002	898677	898678	89869	898690	899147	899187	899667
2	2	2	2	2	2	2	1
899987	9010018	901011	9010258	9010259	901028	9010333	901034301
1	1	2	2	2	2	2	2
901034302	901041	9010598	9010872	9010877	901088	9011494	9011495
2	2	2	2	2	1	1	2
9011971	9012000	9012315	9012568	9012795	901288	9013005	901303
1	1	1	2	1	1	2	2
901315	9013579	9013594	9013838	901549	901836	90250	90251
1	2	2	1	2	2	2	2
902727	90291	902975	902976	903011	90312	90317302	903483
2	2	2	2	2	1	2	2
903507	903516	903554	903811	90401601	90401602	904302	904357
1	1	2	2	2	2	2	2
90439701	904647	904689	9047	904969	904971	905189	905190
1	2	2	2	2	2	2	2
90524101	905501	905502	905520	905539	905557	905680	905686
1	2	2	2	2	2	2	2
905978	90602302	906024	906290	906539	906564	906616	906878
2	1	2	2	2	1	2	2
907145	907367	907409	90745	90769601	90769602	907914	907915
2	2	2	2	2	2	1	2
908194	908445	908469	908489	908916	909220	909231	909410
1	1	2	2	2	2	2	2
909411	909445	90944601	909777	9110127	9110720	9110732	9110944

2	1	2	2	2	2	1	2
911150	911157302	9111596	9111805	9111843	911201	911202	9112085
2	1	2	1	2	2	2	2
9112366	9112367	9112594	9112712	911296201	911296202	9113156	911320501
2	2	2	2	1	1	2	2
911320502	9113239	9113455	9113514	9113538	911366	9113778	9113816
2	1	2	2	1	2	2	2
911384	9113846	911391	911408	911654	911673	911685	911916
2	2	2	2	2	2	2	1
912193	91227	912519	912558	912600	913063	913102	913505
2	2	2	2	2	1	2	1
913512	913535	91376701	91376702	914062	914101	914102	914333
2	2	2	2	1	2	2	2
914366	914580	914769	91485	914862	91504	91505	915143
2	2	1	1	2	1	2	1
915186	915276	91544001	91544002	915452	915460	91550	915664
1	2	2	2	2	1	2	2
915691	915940	91594602	916221	916799	916838	917062	917080
1	2	2	2	1	1	2	2
917092	91762702	91789	917896	917897	91805	91813701	91813702
2	1	2	2	2	2	2	2
918192	918465	91858	91903901	91903902	91930402	919537	919555
2	2	2	2	2	1	2	1
91979701	919812	921092	921362	921385	921386	921644	922296
2	2	2	2	2	2	2	2
922297	922576	922577	922840	923169	923465	923748	923780
2	2	2	2	2	2	2	2
924084	924342	924632	924934	924964	925236	925277	925291
2	2	2	2	2	2	2	2
925292	925311	925622	926125	926424	926682	926954	927241
2	2	1	1	1	1	1	1
92751							
2							

Within cluster sum of squares by cluster:

[1] 6325.137 5249.946

(between_SS / total_SS = 32.1 %)

Available components:

[1] "cluster"	"centers"	"totss"	"withinss"	"tot.withinss"
[6] "betweenss"	"size"	"iter"	"ifault"	

Use the `table()` function to compare the cluster membership of the k-means model (`wisc.km$cluster`) to the actual diagnoses contained in the `diagnosis` vector

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

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

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

The k-means separates the two diagnoses similarly compared to hclust. As the results of the two are nearly identical with differences in the malignant cluster 1.

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

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

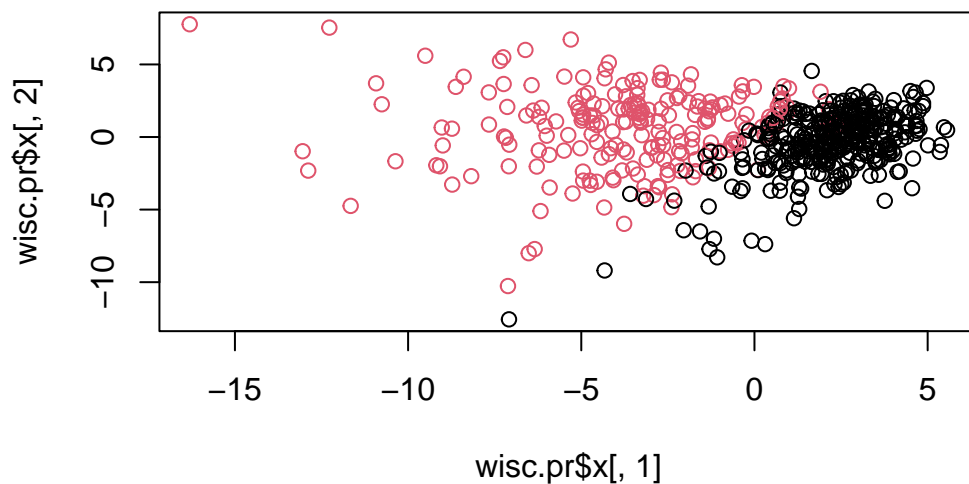
```
table(wisc.km$cluster, wisc.hclust.clusters)
```

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

Combining Methods

I want to cluster in “PC space”

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



```
summary(wisc.pr)
```

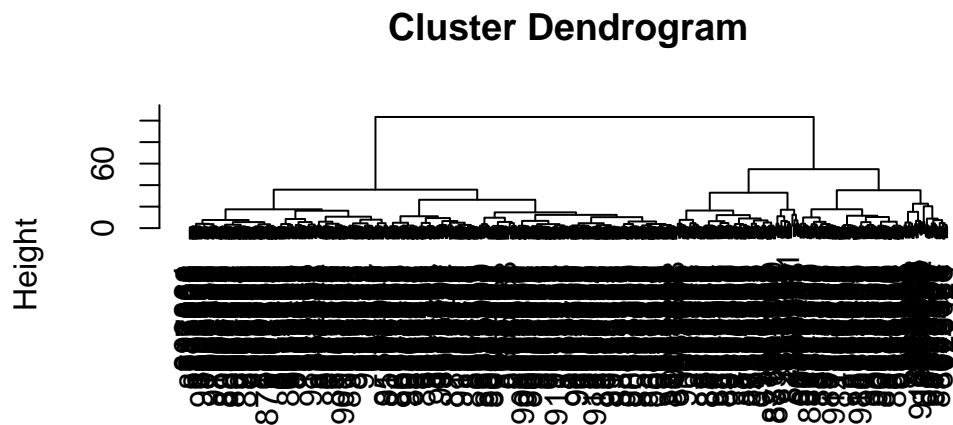
Importance of components:

	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

The `hcluster()` function wants a distance matrix as input..

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



Find the cluster membership vector with `cutree()` function.

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

```
grps
 1  2
203 366
```

,

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



```

      grps
diagnosis  1    2
      B  24 333
      M 179  33

```

Use the distance along the first 7 PCs for clustering i.e. `wisc.pr$x[, 1:7]`

```

d2 <- dist(wisc.pr$x[,1:7])
wisc.pr.hclust <- hclust(d2, method="ward.D2")
wisc.pr.hclust.clusters <- cutree(wisc.pr.hclust, k=2)

```

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

The newly created with four clusters separates out the two diagnoses the same as the the cluster with the distance for the first 3 PCs.

```

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

```

```

      diagnosis
wisc.pr.hclust.clusters  B    M
      1   28 188
      2  329  24

```

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.

The k-means and hierarchical clustering models created in previous section do a decent job of separating the diagnoses by malignant and benign. As we can see between the two, the k-means has two clusters whilst the hcluster has 4 clusters which will cause a discrepancy in values when comparing the two as there are more clusters in which some patients must be closer to a certain cluster.

```

table(wisc.km$cluster, diagnosis)

```

```

diagnosis
  B    M
1  14 175
2 343  37

```

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

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

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

The kmeans clustering model had the better sensitivity as it is able to categorize the benign and malignant patients and correctly detects the ill patients in which have the condition of tumor cells being malignant. Additionally, the hcluster had better specificity as it can correctly reject the patients without a condition as it includes clusters in proportion of benign and is able to correctly reject healthy patients.