

Class 08: Breast Cancer Analysis Project

Selma Cifric (PID A69042976)

Table of contents

Background	1
Data import	1
Principal Component Analysis	5
Hierarchical clustering	13
Alternative methods	15
Combining methods	17
Specificity vs Sensitivity	22

Background

The goal of this mini-project is for you to explore a complete analysis using the unsupervised learning techniques covered in class. You'll extend what you've learned by combining PCA as a preprocessing step to clustering using data that consist of measurements of cell nuclei of human breast masses.

The data itself comes from the Wisconsin Breast Cancer Diagnostic Data Set first reported by K. P. Benne and O. L. Mangasarian: "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets".

Values in this data set describe characteristics of the cell nuclei present in digitized images of a fine needle aspiration (FNA) of a breast mass.

Data import

First, I downloaded the WisconsinCancer.csv file into my class08 folder where this project's directory is saved. Now, I can call it and R will know where to take the data from.

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

We can now examine our data. By using `head()` function, we can preview a first few rows.

```
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			

Note that the first column here `wisc.df$diagnosis` is a pathologist provided expert diagnosis. We will not be using this for our unsupervised analysis as it is essentially the “answer” to the question which cell samples are malignant or benign.

To make sure we don’t accidentally include this in our analysis, lets create a new data.frame that omits this first column.

```
wisc.data <- wisc.df[,-1]
head(wisc.data)
```

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
```

Create diagnosis vector:

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

Q1. How many observations are in this dataset?

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

```
[1] 569
```

There are 569 observations.

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

```
sum(diagnosis == "M")
```

```
[1] 212
```

There are 212 malignant observations.

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

```
length(grep("_mean$", names(wisc.df)))
```

```
[1] 10
```

There are 10 variables in this data set that are suffixed with __mean.

Principal Component Analysis

Before we conduct PCA analysis, we should first check our data so far.

```
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

Then, we can execute PCA with `prcomp()`, ensuring we enable `scale=TRUE` argument. It is important that we scale our data because PCA relies on variance based calculations where bigger outputs could skew the smaller ones.

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

And then, we can inspect the summary of our data.

```
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)?

PC1 proportion of variance is 0.4427 or 44.27%.

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

```
# Perform PCA (make sure diagnosis column is excluded)
wisc.pr <- prcomp(wisc.df[, -1], center = TRUE, scale. = TRUE)

# Calculate variance of each principal component
pr.var <- wisc.pr$sdev^2

# Calculate proportion of variance explained
pve <- pr.var / sum(pr.var)

# Calculate cumulative proportion of variance explained
cum_pve <- cumsum(pve)

# Find how many PCs explain at least 70% of the total variance
which(cum_pve >= 0.7)[1]
```

```
[1] 3
```

We need at least 3 PCs.

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

```
# Perform PCA (make sure diagnosis column is excluded)
wisc.pr <- prcomp(wisc.df[, -1], center = TRUE, scale. = TRUE)

# Calculate variance of each principal component
pr.var <- wisc.pr$sdev^2

# Calculate proportion of variance explained
pve <- pr.var / sum(pr.var)

# Calculate cumulative proportion of variance explained
cum_pve <- cumsum(pve)

# Find how many PCs explain at least 90% of the total variance
which(cum_pve >= 0.9)[1]
```

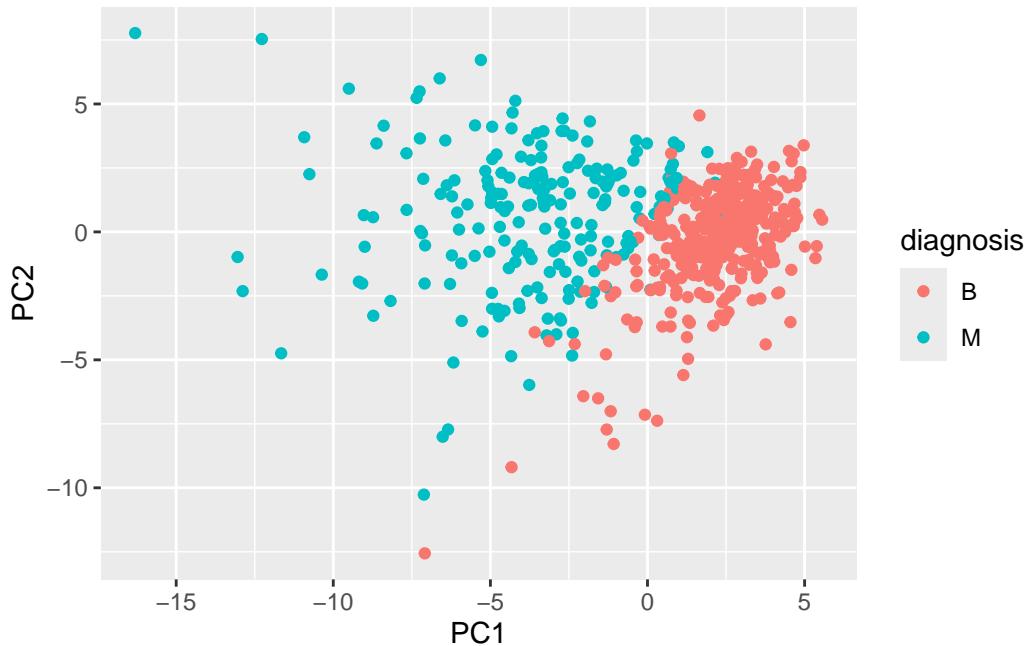
```
[1] 7
```

We would need at least 7 PCs.

Let's make our main result figure - the "PC plot" or "score plot", using `ggplot()`.

```
library(ggplot2)

ggplot(wisc.pr$x) +
  aes(PC1, PC2, col = diagnosis) +
  geom_point()
```

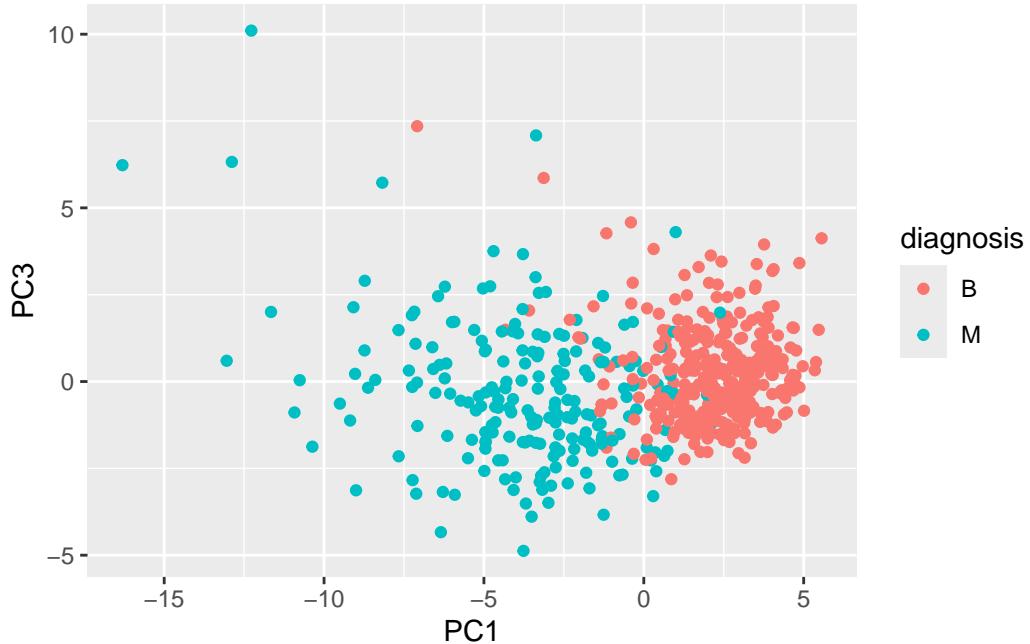


Q7. What stands out to you about this plot? Is it easy or difficult to understand?
Why?

It looks like benign and malignant diagnoses separate in different clusters. It is a good choice of plotting because separating the diagnosis by color clearly shows us the differences.

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

```
ggplot(wisc.pr$x) +
  aes(PC1, PC3, col = diagnosis) +
  geom_point()
```



We see a less defined clustering of benign and malignant patients, indicated by blue and red dots overlapping more.

To calculate variance for each component, we can use the same code as in chunk 14 without limiting the pve to any number.

```
# Perform PCA (make sure diagnosis column is excluded)
wisc.pr <- prcomp(wisc.df[, -1], center = TRUE, scale. = TRUE)

# Calculate variance of each principal component
pr.var <- wisc.pr$sdev^2

# Calculate proportion of variance explained
pve <- pr.var / sum(pr.var)

# Calculate cumulative proportion of variance explained
cum_pve <- cumsum(pve)

# To see all variances
cum_pve
```

```
[1] 0.4427203 0.6324321 0.7263637 0.7923851 0.8473427 0.8875880 0.9100953
[8] 0.9259825 0.9398790 0.9515688 0.9613660 0.9700714 0.9781166 0.9833503
```

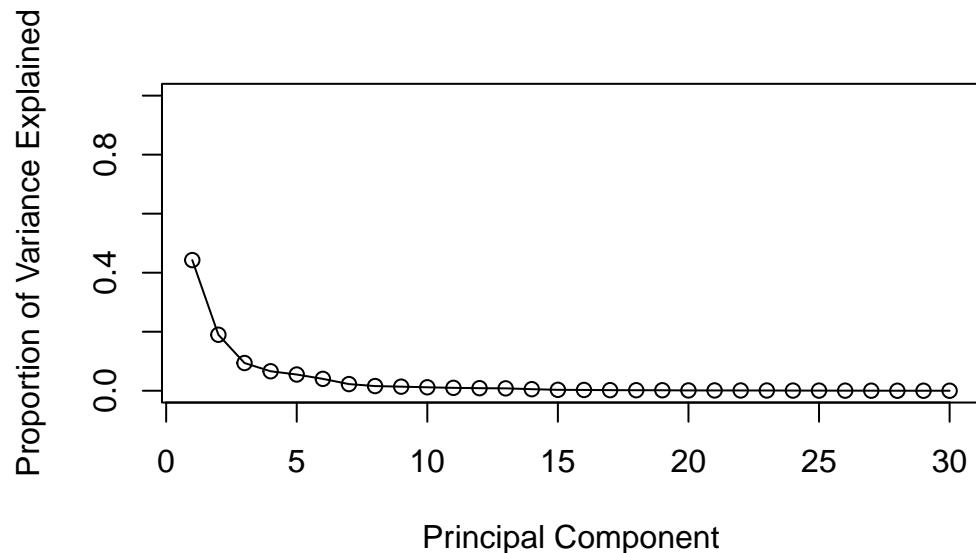
```
[15] 0.9864881 0.9891502 0.9911302 0.9928841 0.9945334 0.9955720 0.9965711  
[22] 0.9974858 0.9982971 0.9988990 0.9994150 0.9996876 0.9999176 0.9999706  
[29] 0.9999956 1.0000000
```

```
# To see the first values  
head(pve)
```

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

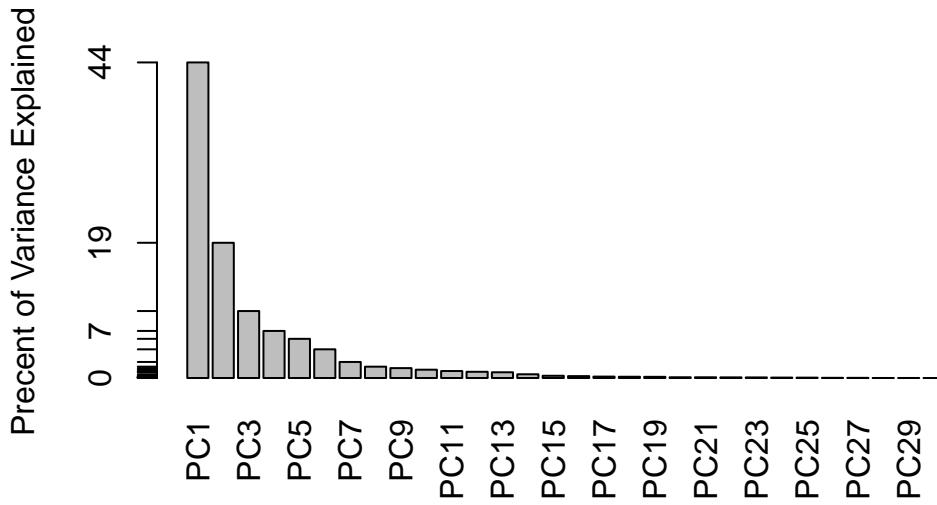
Then, we can plot the variances.

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



To plot this in a histogram graph:

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



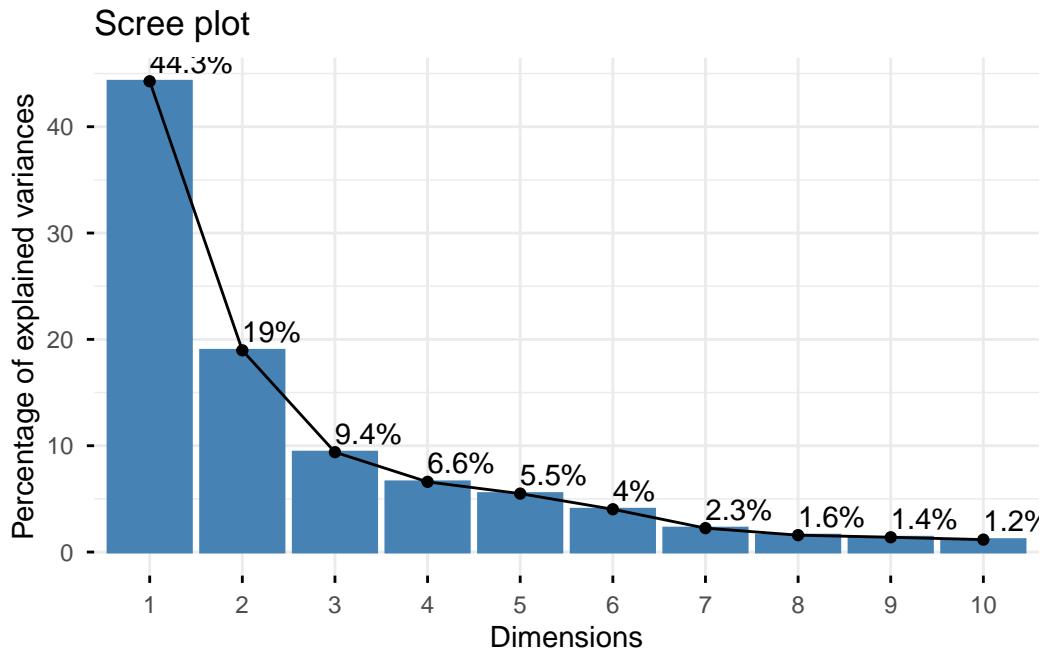
Exploring factoextra package.

```
#ggplot based graph
#install.packages("factoextra") in console, not script.
library(factoextra)
```

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

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

Warning in geom_bar(stat = "identity", fill = barfill, color = barcolor, :
Ignoring empty aesthetic: `width`.



Hierarchical clustering

First, we scale the wisc.data data using the `scale()` function.

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

We can compute Euclidian distance between all the pairs in `data.scaled`.

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

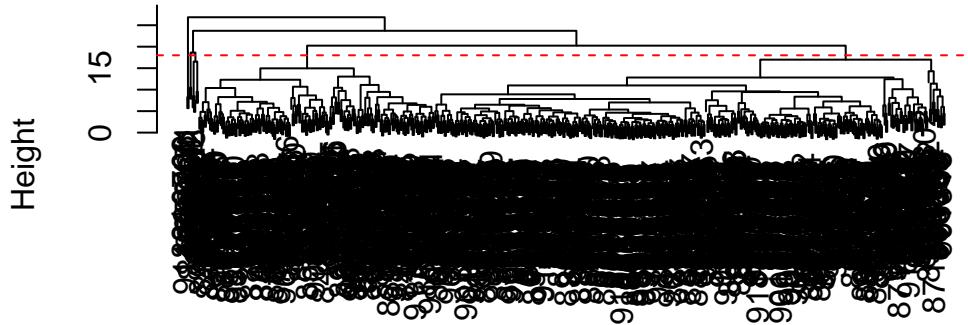
We then perform hierarchical clustering using the complete linkage method, which defines the distance between clusters as the distance between their farthest points.

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

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

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

Cluster Dendrogram



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

I chose height to be 18, as indicated by the red dashed line, because it produces exactly 4 clusters. I identified the cluster by the horizontal lines.

Next, we assign cluster memberships using the `cutree()` function.

```
wisc.hclust.clusters <- cutree(wisc.hclust, h=19)  
  
table(wisc.hclust.clusters, diagnosis)
```

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

From this table, we can see that cluster 1 captures most patients assigned a malignant diagnosis whereas cluster 3 captures most benign patients. Given that we have access to diagnosis in this data set, our clustering method seems to encompass most biologically relevant points.

Alternative methods

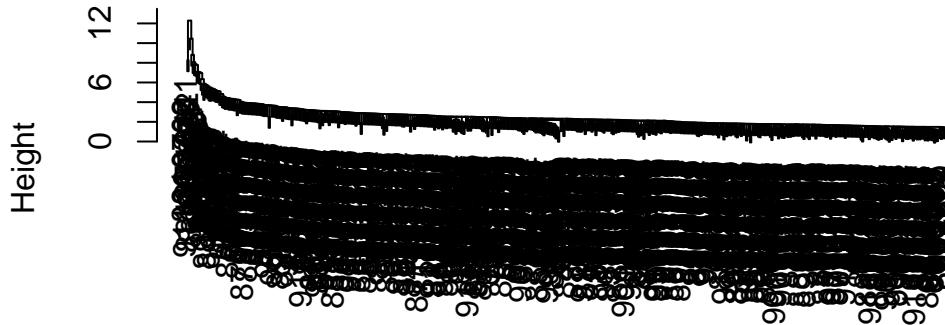
Now, we can try other methods that can help us cluster differently before deciding which may be the best fit. These include “single”, “complete”, “average” and (Barry’s favorite) “ward.D2”.

```
methods <- c("single", "complete", "average", "ward.D2")

for (m in methods) {
  cat("\n\n### Linkage method:", m, "\n")
  plot(hclust(data.dist, method = m),
    main = paste("Hierarchical Clustering using", m, "linkage"),
    xlab = "", sub = "")
}
```

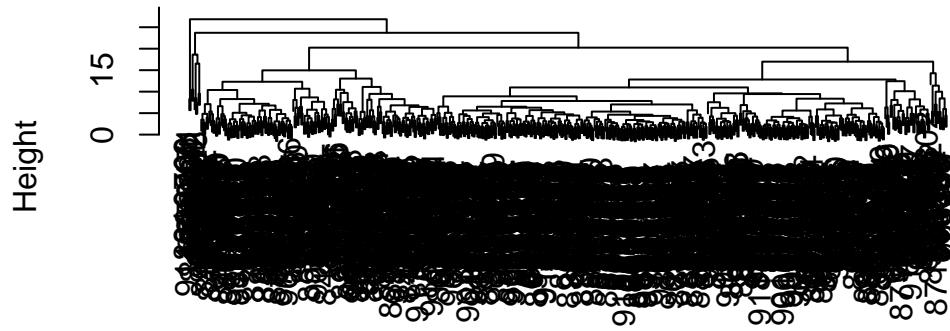
```
### Linkage method: single
```

Hierarchical Clustering using single linkage



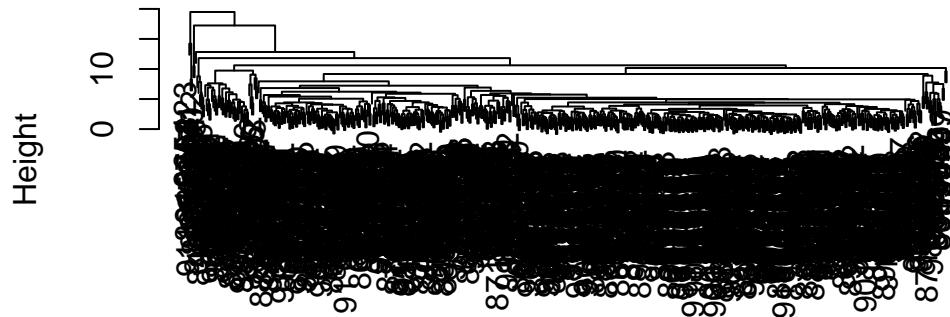
```
### Linkage method: complete
```

Hierarchical Clustering using complete linkage



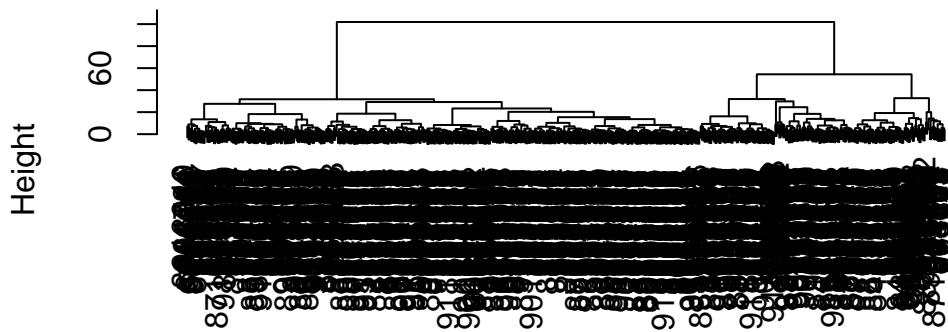
```
### Linkage method: average
```

Hierarchical Clustering using average linkage



```
### Linkage method: ward.D2
```

Hierarchical Clustering using ward.D2 linkage



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

I like ward.D2 and complete linkage methods the best because they yield the cleanest possible dendograms. Single linkage method is too crowded in general and average linkage method contains clusters of varying size which can be confusing to read. Ward.D2 linkage method is very clean and compact and my personal favorite too, with complete linkage method as my second favorite.

Combining methods

Using the minimum number of principal components required to describe at least 90% of the variability in the data, create a hierarchical clustering model with the linkage method="ward.D2". We use Ward's criterion here because it is based on multidimensional variance like principal components analysis. Assign the results to wisc.pr.hclust.

```

# 1. Perform PCA on your scaled data
wisc.pca <- prcomp(data.scaled, center = TRUE, scale. = TRUE)

# 2. Determine how many PCs are needed to explain at least 90% of the variance
explained_var <- summary(wisc.pca)$importance[3, ] # cumulative proportion of variance
num_pcs <- min(which(explained_var >= 0.9)) # minimum PCs for >=90% variance

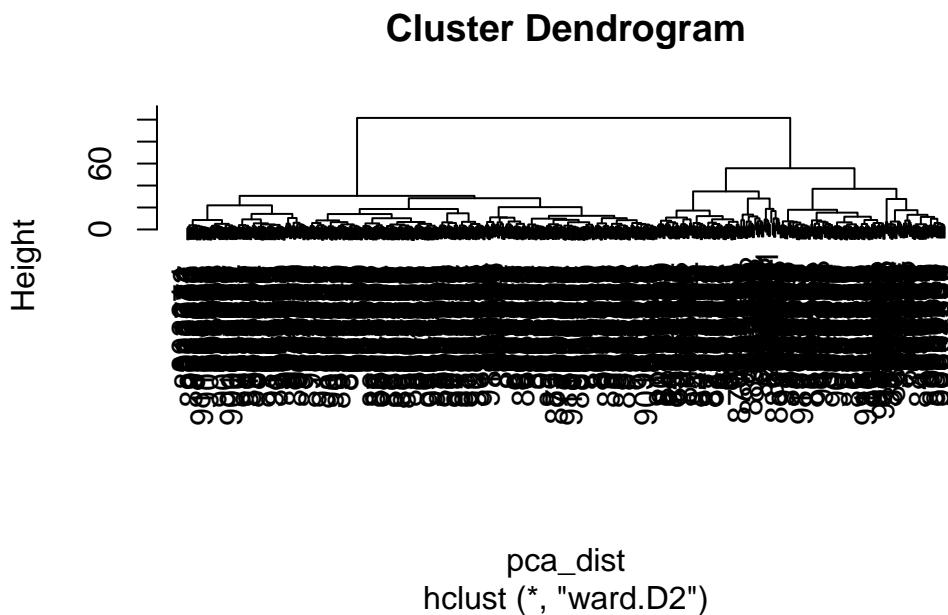
# 3. Extract the scores for those principal components
pca_scores <- wisc.pca$x[, 1:num_pcs]

# 4. Compute distance matrix
pca_dist <- dist(pca_scores)

# 5. Perform hierarchical clustering using Ward's method
wisc.pr.hclust <- hclust(pca_dist, method = "ward.D2")

# Plot wisc.pr.hclust
plot(wisc.pr.hclust)

```



We see two big clusters from this dendrogram. To check if these clusters correspond to malignant and benign diagnoses, we can use `cutree()` function again to examine them separately from the rest of the clusters.

```
grps <- cutree(wisc.pr.hclust, h=70)
table(grps)
```

```
grps
 1   2
216 353
```

Now that we have the two clusters or groups isolated, we can compare them in a table with already known diagnoses.

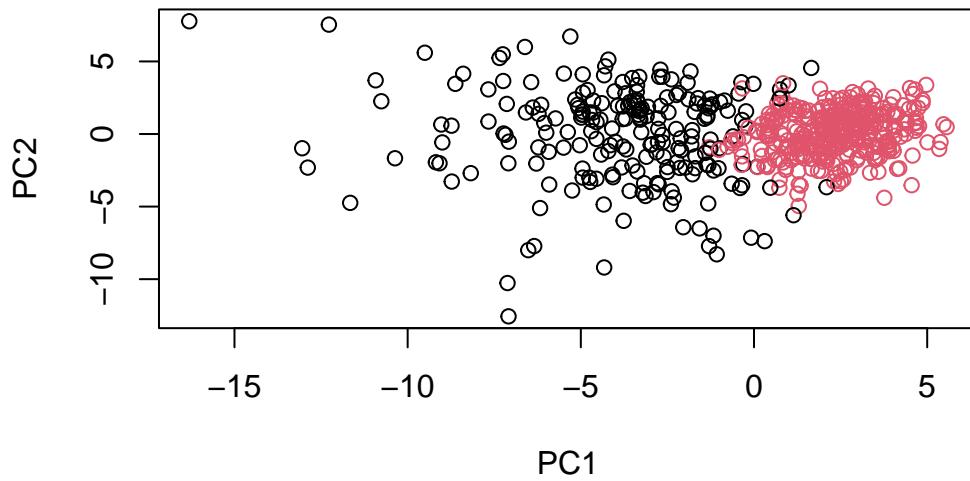
```
table(grps, diagnosis)
```

grps	B	M
1	28	188
2	329	24

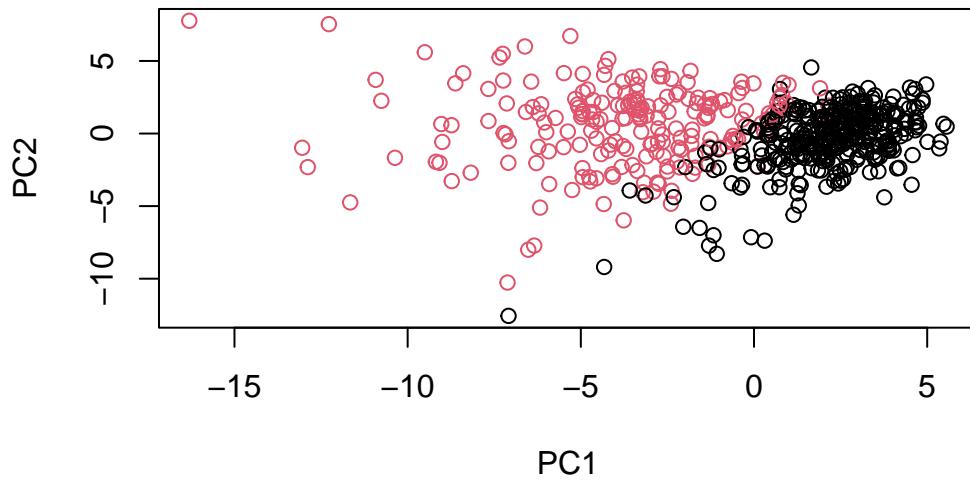
Just from this table, we can see that most of our patient data points cluster well with distinct diagnoses using ward.D2 linkage method (90% variance). Most of the cluster 1 data is malignant patients and most of the cluster 2 encompasses benign patients.

We can plot grps and diagnosis separately to compare them visually.

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



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



I've skipped the optional adjustment of color ordering for group 1 vs 2.

```
## Use the distance along the first 7 PCs for clustering i.e. wisc.pr$x[, 1:7]. Answer from
wisc.pr.hclust <- hclust(dist(wisc.pr$x[, 1:7]), method="ward.D2")
```

Cut this hierarchical clustering model into 2 clusters and assign the results to `wisc.pr.hclust.clusters`.

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

Using `table()`, compare the results from your new hierarchical clustering model with the actual diagnoses.

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

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

Q13. How well does the newly created model with four clusters separate out the two diagnoses?

My previous step contained a model with 2 clusters, not 4. The diagnoses separate pretty well.

Q14. How well do the hierarchical clustering models you created in previous sections (i.e. before PCA) do in terms of separating the diagnoses? Again, use the `table()` function to compare the output of each model (`wisc.km$cluster` and `wisc.hclust.clusters`) with the vector containing the actual diagnoses.

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

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

I did not do k-mean steps in today's class so the only pre-PCA hierarchical clustering model I have is `wisc.hclust.clusters()`. Against diagnosis, doing hierarchical clustering on raw data give me an output with mixed diagnoses which is not a desirable output. Moreover, smaller clusters 2 and 4 further contribute to the noise of this data. Therefore, combining PCA with `hclust` is a better way of deducing this information in a clearer way.

Specificity vs Sensitivity

Sensitivity refers to a test's ability to correctly detect ill patients who do have the condition. In our example here the sensitivity is the total number of samples in the cluster identified as predominantly malignant (cancerous) divided by the total number of known malignant samples. In other words: $TP/(TP+FN)$.

Specificity relates to a test's ability to correctly reject healthy patients without a condition. In our example specificity is the proportion of benign (not cancerous) samples in the cluster identified as predominantly benign that are known to be benign. In other words: $TN/(TN+FN)$.

Q15. OPTIONAL: Which of your analysis procedures resulted in a clustering model with the best specificity? How about sensitivity?

Clustering model before PCA:

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

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

Sensitivity Cluster 1 (malignant mostly): $TP = 165$ (malignant), $FN = 12+2+0=14$

```
#Sensitivity  
165/(165+14)
```

```
[1] 0.9217877
```

Specificity Cluster 3 (benign mostly): $TN = 343$ (benign), $FN = 14$

```
#Specificity  
343/(343+14)
```

```
[1] 0.9607843
```

CLustering + PCA:

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

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

Sensitivity Cluster 1 (malignant mostly): TP = 188 (malignant), FN = 28

```
#Sensitivity  
188/(188+28)
```

```
[1] 0.8703704
```

Specificity Cluster 2 (benign mostly): TN = 329 (benign), FN = 28

```
#Specificity  
329/(329+28)
```

```
[1] 0.9215686
```

My results suggest that without PCA, clustering alone gave me higher specificity and sensitivity which I'm not confident is what I expected. I expected that combined methods give me better results.