

Class 8: Breast Cancer Mini Project

Barry (PID: 911)

Table of contents

Intro	1
Data import	1
Tidy to remove diagnosis	4
Cluster the dataset	4
Principal Component Analysis (PCA)	6
Scaling	6
Breast Cancer PCA	11
Combine PCA and clustering	16
Prediction with our PCA model	18

Intro

Today we will practice applying our PCA and clustering methods from the last class on some breast cancer FNA data.

Data import

Let's get the data into 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		

Q1. How many samples/patients are in this dataset?

There are 569 samples in this dataset

Q2. How many cancer/non-cancer diagnosis samples are there?

```
sum(wisc.df$diagnosis == "M")
```

```
[1] 212
```

The `table()` function is a super useful utility for counting up the number of observations of each type

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

```

  B   M
357 212

```

Q3. How many columns/dimensions are there

```
ncol(wisc.df)
```

```
[1] 31
```

Q4. How many columns are suffixed with “__mean”?

```
colnames(wisc.df)
```

```

[1] "diagnosis"           "radius_mean"
[3] "texture_mean"        "perimeter_mean"
[5] "area_mean"           "smoothness_mean"
[7] "compactness_mean"    "concavity_mean"
[9] "concave.points_mean" "symmetry_mean"
[11] "fractal_dimension_mean" "radius_se"
[13] "texture_se"          "perimeter_se"
[15] "area_se"             "smoothness_se"
[17] "compactness_se"      "concavity_se"
[19] "concave.points_se"   "symmetry_se"
[21] "fractal_dimension_se" "radius_worst"
[23] "texture_worst"       "perimeter_worst"
[25] "area_worst"          "smoothness_worst"
[27] "compactness_worst"   "concavity_worst"
[29] "concave.points_worst" "symmetry_worst"
[31] "fractal_dimension_worst"

```

The `grep()` function can help us find pattern matches here:

```

x <- grep("_mean", colnames(wisc.df))
length(x)

```

```
[1] 10
```

Tidy to remove diagnosis

Save a vector of this expert diagnosis for later and remove it from the data to undergo clustering, PCA etc...

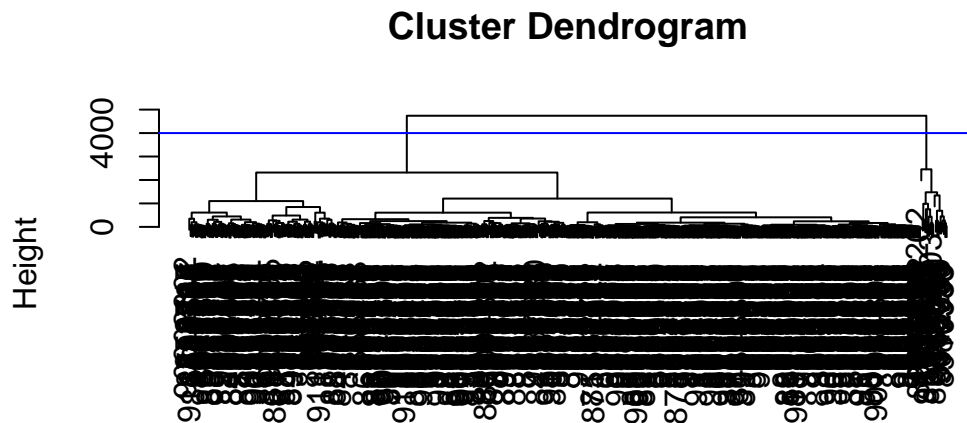
```
diagnosis <- wisc.df$diagnosis
```

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

Cluster the dataset

Let's try a `hclust()`.

```
hc.raw <- hclust(dist(wisc.data))
plot(hc.raw)
abline(h=4000, col="blue")
```



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

To get some clusters out of this I can “cut” the tree at given height:

```
grps <- cutree(hc.raw, h=4000)
table(grps)
```

```
grps
  1   2
549 20
```

To see the correspondence of our cluster `grps` with the expert `diagnosis` I can use `table()`:

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

```
      diagnosis
grps   B    M
  1  357 192
  2    0  20
```

That is not that useful a clustering result...

Principal Component Analysis (PCA)

Scaling

Scaling data before analysis is often critical.

Side-note: The default for `prcomp()` is `sacle=FALSE`.

There is a dataset in R called `mtcars` which has loads of numbers about old cars.

```
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

```
colMeans(mtcars)
```

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			

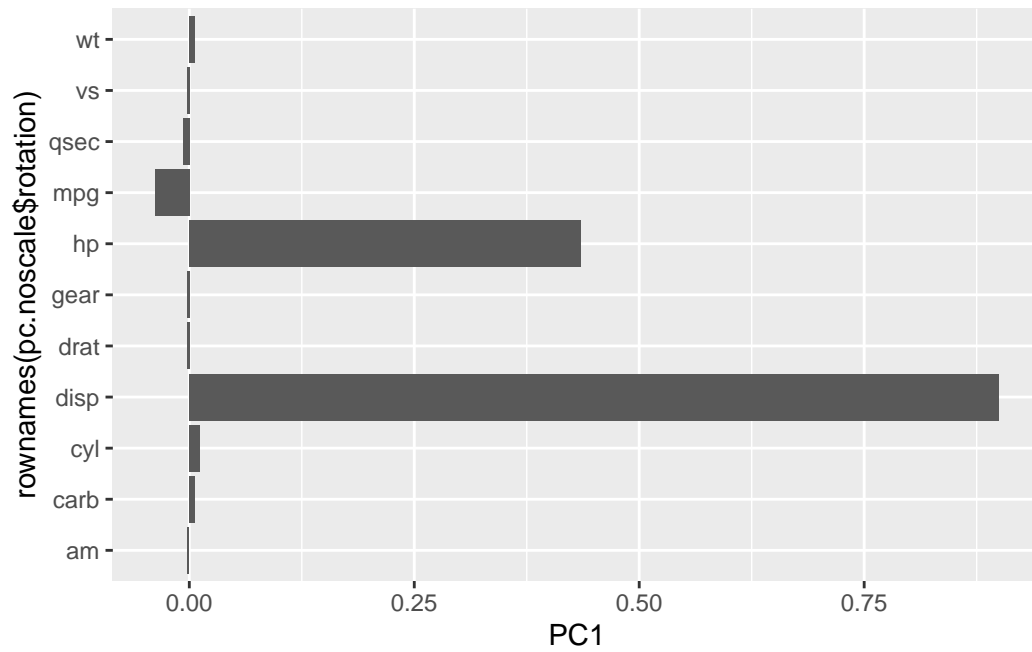
```
apply(mtcars, 2, sd)
```

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	

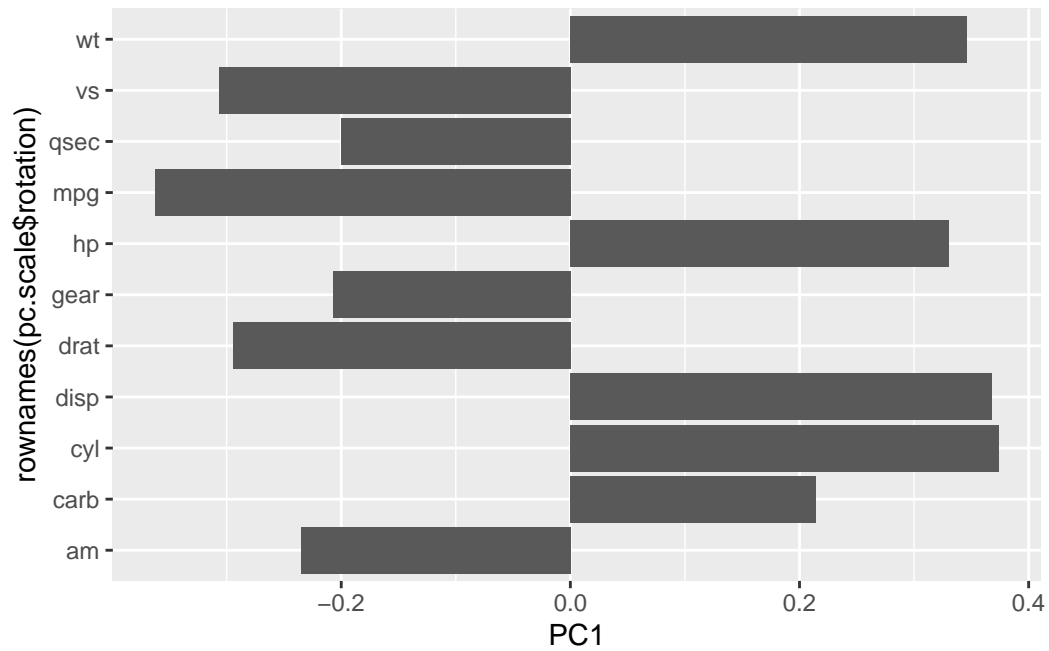
```
pc.noscale <- prcomp(mtcars, scale=FALSE)
pc.scale <- prcomp(mtcars, scale=TRUE)
```

Let's look at the loadings first:

```
library(ggplot2)
ggplot( pc.noscale$rotation ) +
  aes(PC1, rownames(pc.noscale$rotation)) +
  geom_col()
```

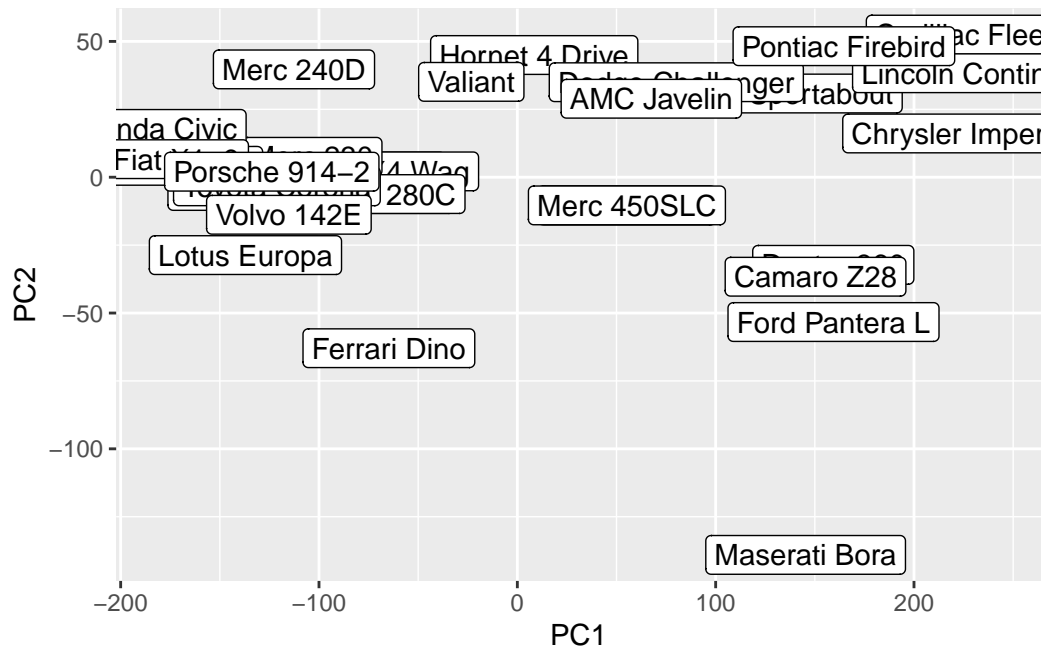


```
ggplot( pc.scale$rotation ) +
  aes(PC1, rownames(pc.scale$rotation)) +
  geom_col()
```

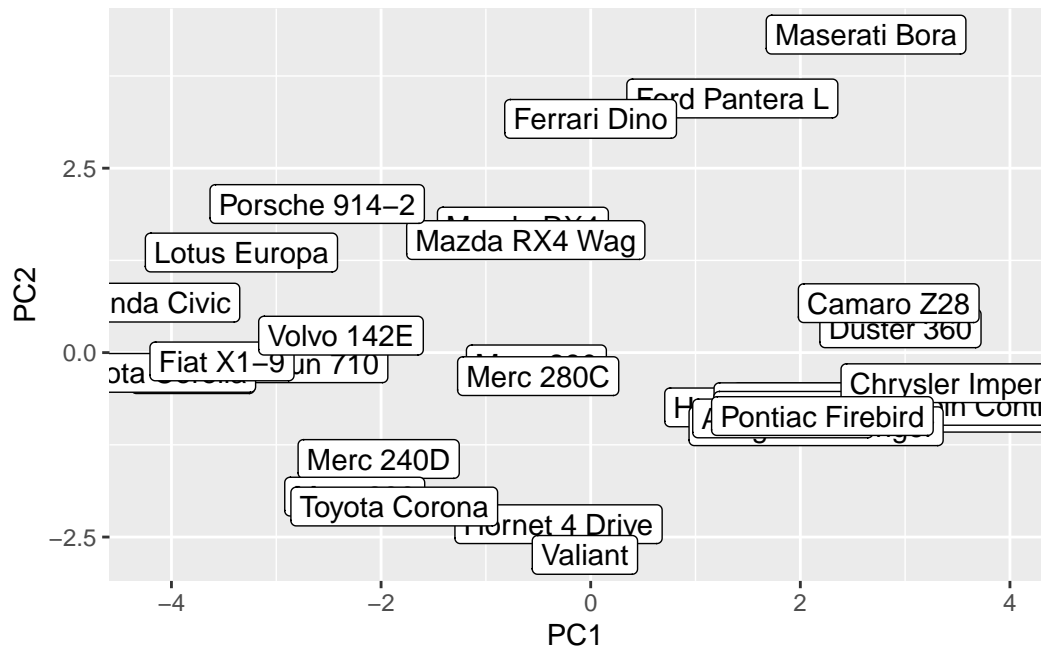


The main PC result figure is often called a “score plot” or “PC plot” or “PC1 vs PC2 plot”

```
ggplot(pc.noscale$x) +
  aes(PC1, PC2, label=rownames(pc.noscale$x)) +
  geom_point() +
  geom_label()
```

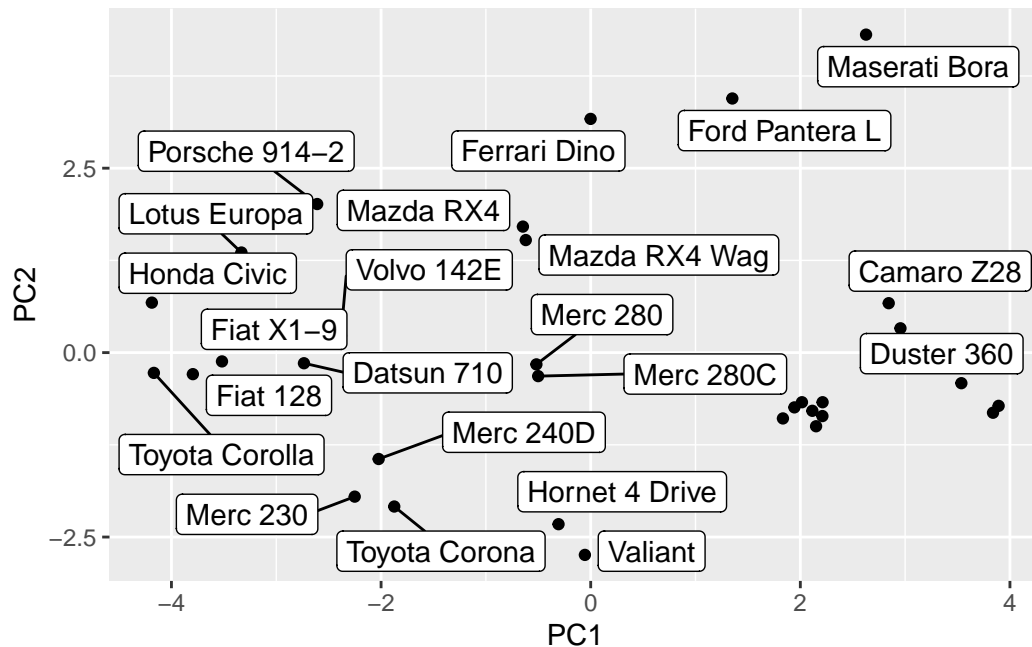



```
ggplot(pc.scale$x) +
  aes(PC1, PC2, label=rownames(pc.scale$x)) +
  geom_point() +
  geom_label()
```



```
library(ggrepel)
ggplot(pc.scale$x) +
  aes(PC1, PC2, label=rownames(pc.scale$x)) +
  geom_point() +
  geom_label_repel()
```

Warning: ggrepel: 10 unlabeled data points (too many overlaps). Consider increasing max.overlaps



```
x <- scale(mtcars)
round( colMeans(x) )
```

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

```
round( apply(x, 2, sd) )
```

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

Key-point: Generally we want to “scale” our data before analysis to avoid being mis-lead due to your data having different measurment units.

Breast Cancer PCA

We will scale our data.

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

See how well we are doing:

```
summary(pca)
```

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

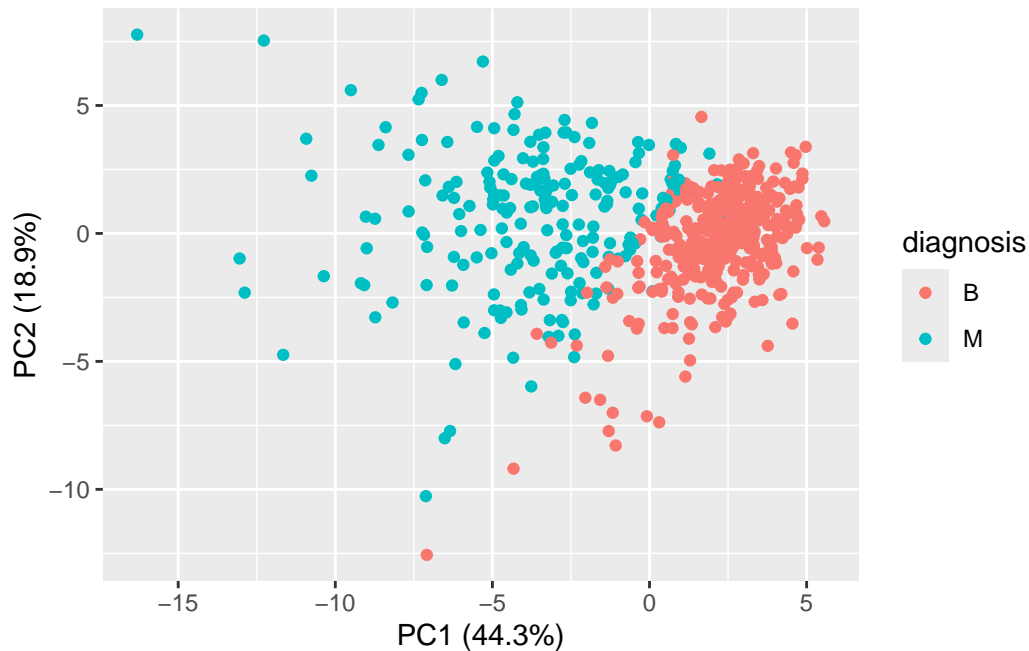
	PC15	PC16	PC17	PC18	PC19	PC20	PC21
Standard deviation	0.30681	0.28260	0.24372	0.22939	0.22244	0.17652	0.1731
Proportion of Variance	0.00314	0.00266	0.00198	0.00175	0.00165	0.00104	0.0010
Cumulative Proportion	0.98649	0.98915	0.99113	0.99288	0.99453	0.99557	0.9966

	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

Our PC plot

```
ggplot(pca$x) +  
  aes(PC1, PC2, col=diagnosis) +  
  geom_point() +  
  xlab("PC1 (44.3%)") +  
  ylab("PC2 (18.9%)")
```



Q. How many PCs capture 80% of the original variance in the dataset?

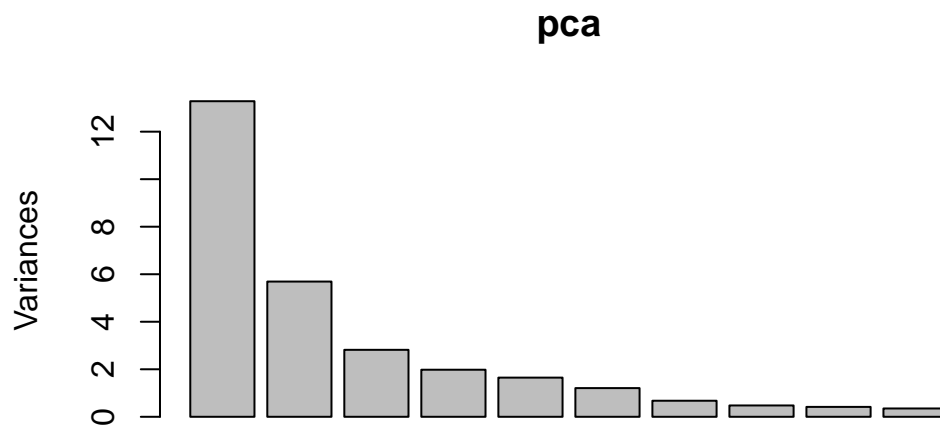
```
summary(pca)
```

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

```
plot(pca)
```



Q. Use ggplot to plot a “scree-plot” of the variance per PC.

```
attributes(pca)
```

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

$class
[1] "prcomp"
```

We can extract the sdev and figure out the variance.

```
v <- pca$sdev^2
sum(v)
```

```
[1] 30
```

The proportion of variance captured in each PC

```
round( v/sum(v), 2)
```

```
[1] 0.44 0.19 0.09 0.07 0.05 0.04 0.02 0.02 0.01 0.01 0.01 0.01 0.01 0.01 0.00
[16] 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
```

Cumulative variance captured

```
cumsum(v/sum(v))
```

```
[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
```

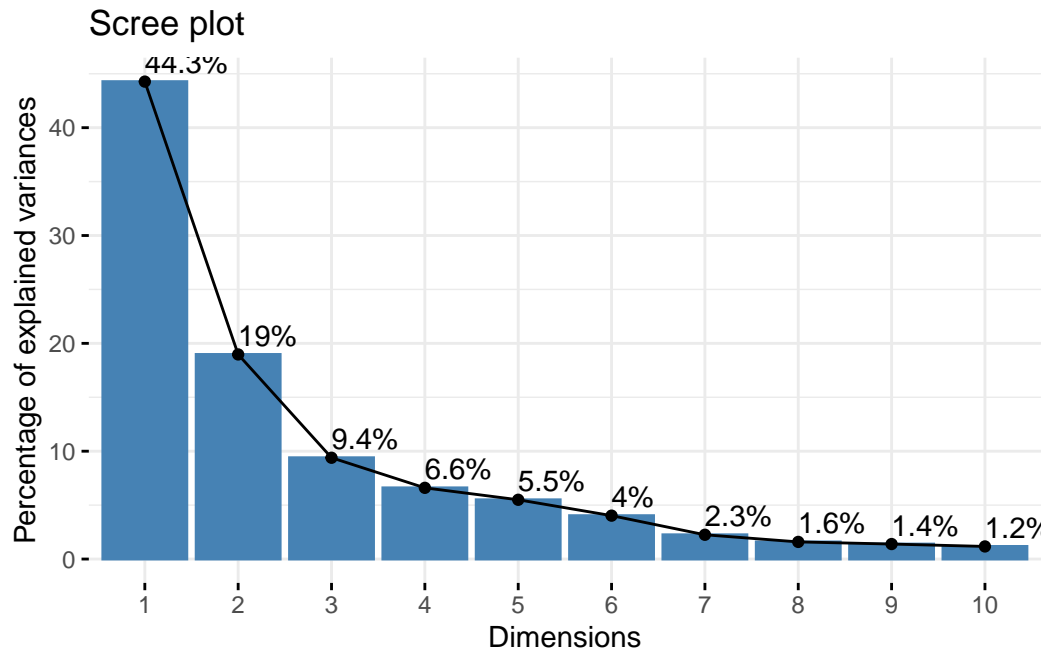
```
which( cumsum(v/sum(v)) > 0.8 )
```

```
[1] 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29
[26] 30
```

```
library(factoextra)
```

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

```
fviz_eig(pca, addlabels = TRUE)
```



Combine PCA and clustering

We saw earlier that clustering the raw data alone did not provide useful results.

We can use our new PC variables (our PCs) as a basis for clustering. Use our `$x` PC scores and cluster in the PC1-2 subspace.

```
hc.pca <- hclust( dist(pca$x[,1:2]), method="ward.D2")  
plot(hc.pca)  
abline(h=70, col="blue")
```


Q. Does your clustering help separate cancer from on-cancer samples (i.e. diagnosis M vs B)?

```
grps
  1  2
195 374
```

	diagnosis	
grps	B	M
1	18	177
2	339	35

```
diagnosis
      B      M
357 212
```

Positive cancer samples “M” Negative non-cancer samples “B”

True our cluster/grp 1 False our cluster/grp 2

Q. How many True Positives (TP) do we have?

Q. How many False Positives (FP) do we have?

Sensitivity $TP/(TP+FN)$. Specificity $TN/(TN+FN)$

Prediction with our PCA model

We can take new data (in this case from UofM) and project it onto our new variables (PCs).

Read the UofM data

```
url <- "https://tinyurl.com/new-samples-CSV"
new <- read.csv(url)
```

Projection

```
npc <- predict(pca, newdata=new)
```

Base R plot

```
plot(pca$x[,1:2], col=grps)

## add the new points
points(npc[,1], npc[,2], col="blue", pch=16, cex=3)
text(npc[,1], npc[,2], c(1,2), col="white")
```

