Lab08

Yushi Li (A15639705)

2/10/2022

1. Exploratory data analysis

```
# Save your input data file into your Project directory
fna.data <- "https://bioboot.github.io/bimm143_S20/class-material/WisconsinCancer.csv"

# Input the data and store as wisc.df
wisc.df <- read.csv(fna.data, 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
```

```
# Remove 1st column:
wisc.data <- wisc.df[,-1]
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
```

```
# Create diagnosis vector for later
diagnosis <- factor(wisc.df$diagnosis)


# Q1. How many observations are in this dataset?

nrow(wisc.data)
```

```
## [1] 569
```

```
# A. There are 569 observations.


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

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

```
## [1] 212
```

```
# A. There are 212 observations with malignant diagnoses.


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

# A. There are 10 variables with _mean.
```

2. Principal Component Analysis

Performing PCA

```
# 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
```

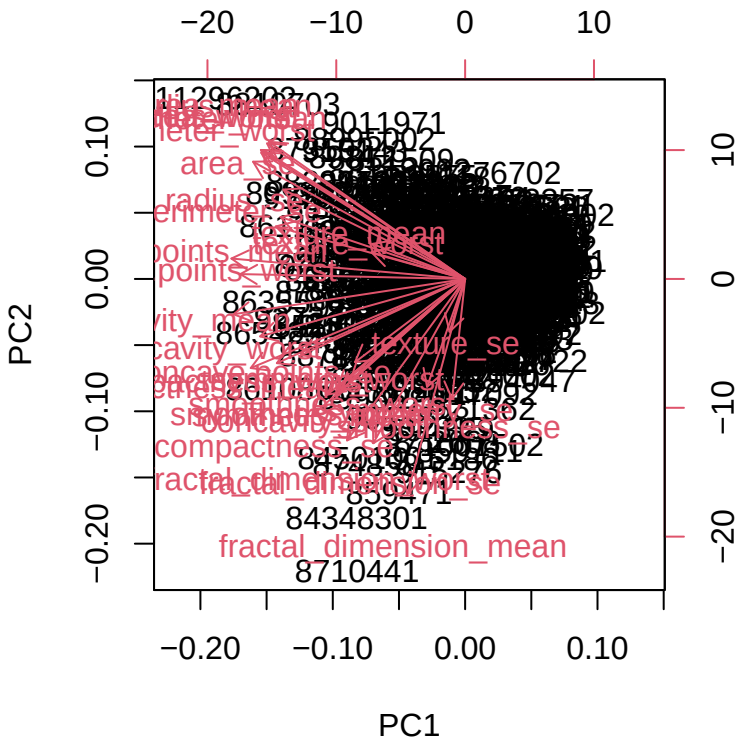
```
# Perform PCA on wisc.data by completing the following code
wisc.pr <- prcomp(wisc.data, scale=TRUE)

# Look at summary of results
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
```

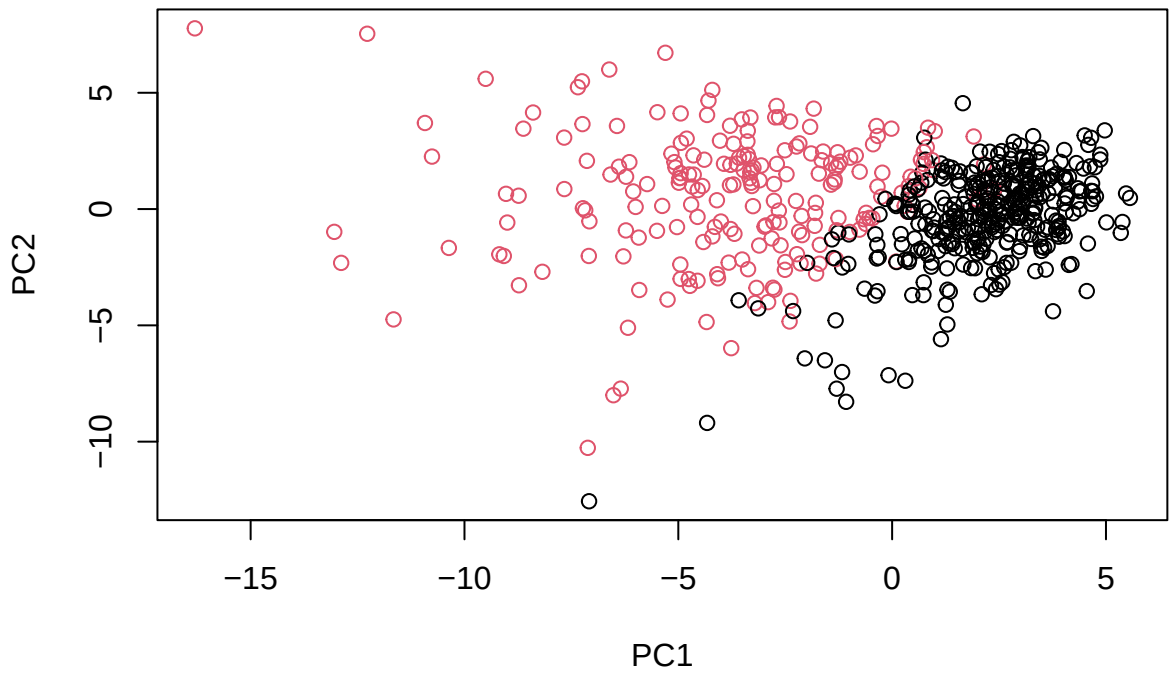
```
# Q4. From your results, what proportion of the original variance is captured
# by the first principal components (PC1)?

# A. 44.27% of original variance is captured by PC1.


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

# A. 3 PCs are needed.


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

# A. 7 PCs are required.
```

Interpreting PCA results

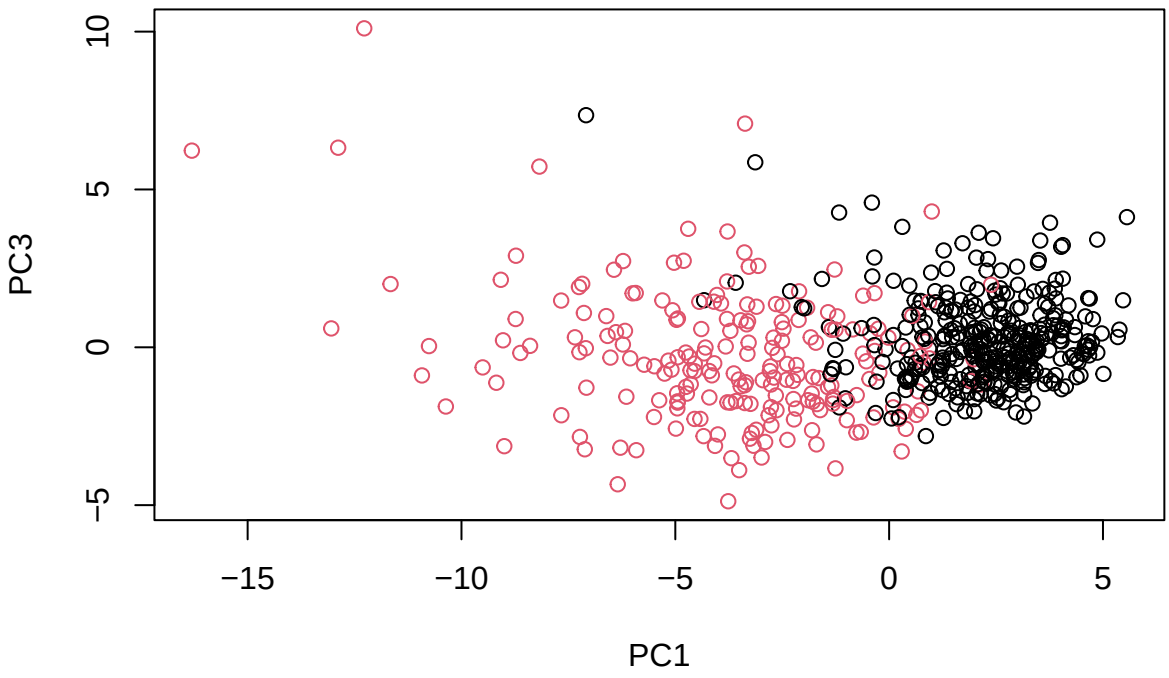
```
# Create a biplot of the wisc.pr using the biplot() function.
biplot(wisc.pr)
```



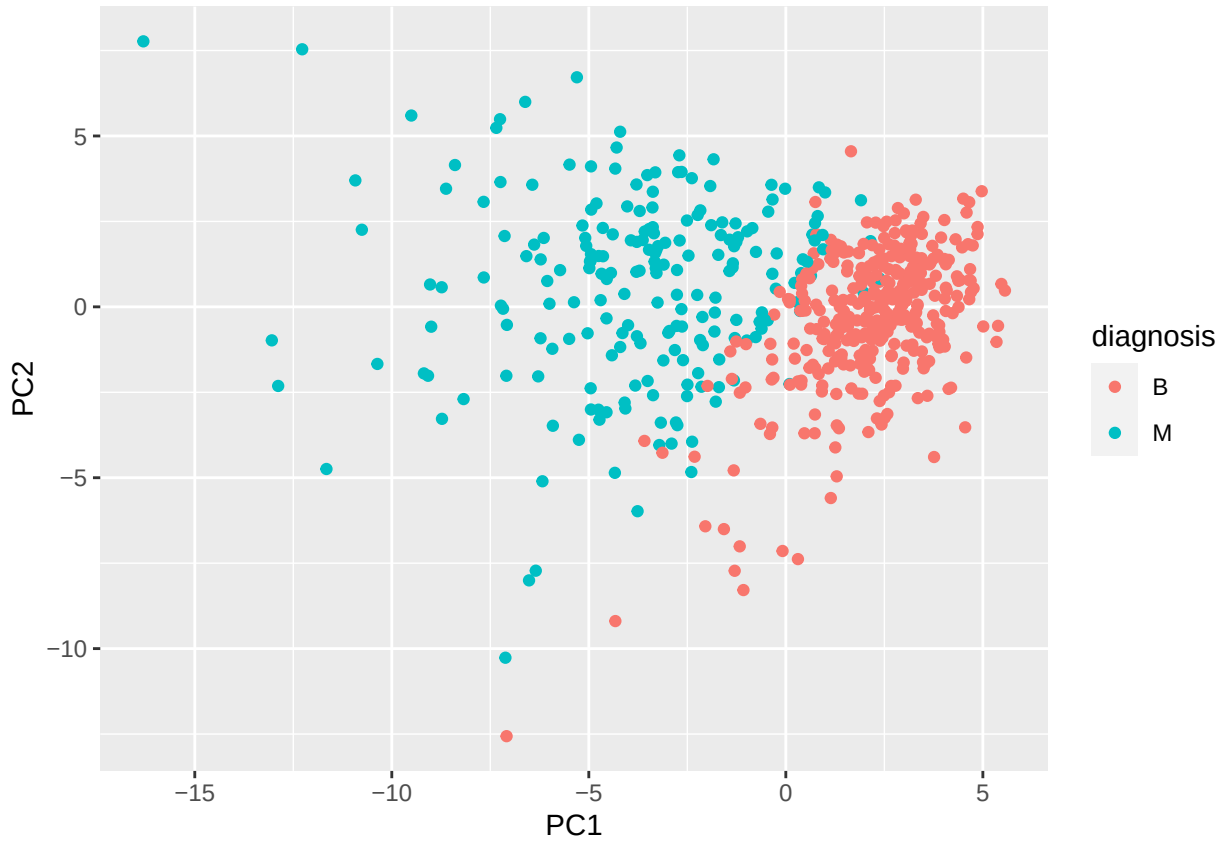

```
# Q7. What stands out to you about this plot?
# Is it easy or difficult to understand? Why?

# A. The first thing that stood out was the large cluster of data within the
# -10 to 10 range on both x- and y- axes. It is difficult to understand the plot
# mainly due to the overlapping texts and arrows.


# Scatter plot observations by components 1 and 2

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




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

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

# A. Most of the separations between malignant and benign diagnoses are along
# the x-axis (PC1). Meaning PC1 contributes most significantly to the different
# distribution patterns of observations in malignant and benign diagnoses.


# Using ggplot2 to better visualize results:
# Create a data.frame for ggplot
df <- as.data.frame(wisc.pr$x)
df$diagnosis <- diagnosis

# Load the ggplot2 package
library(ggplot2)
```




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

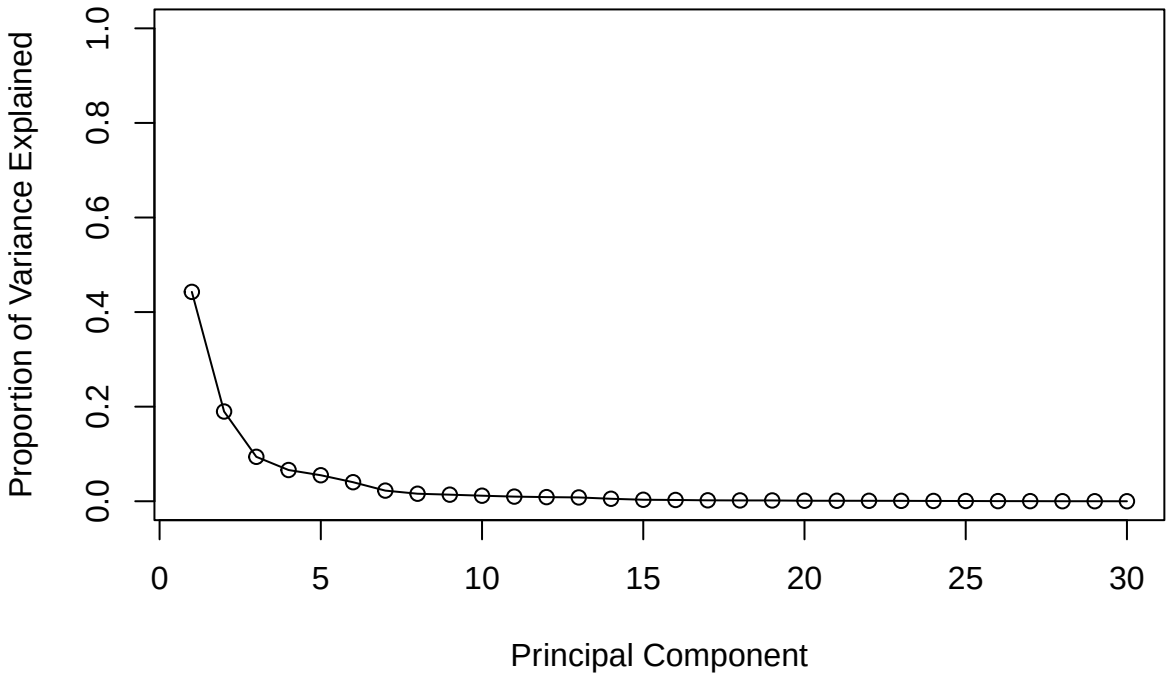


Variance explained

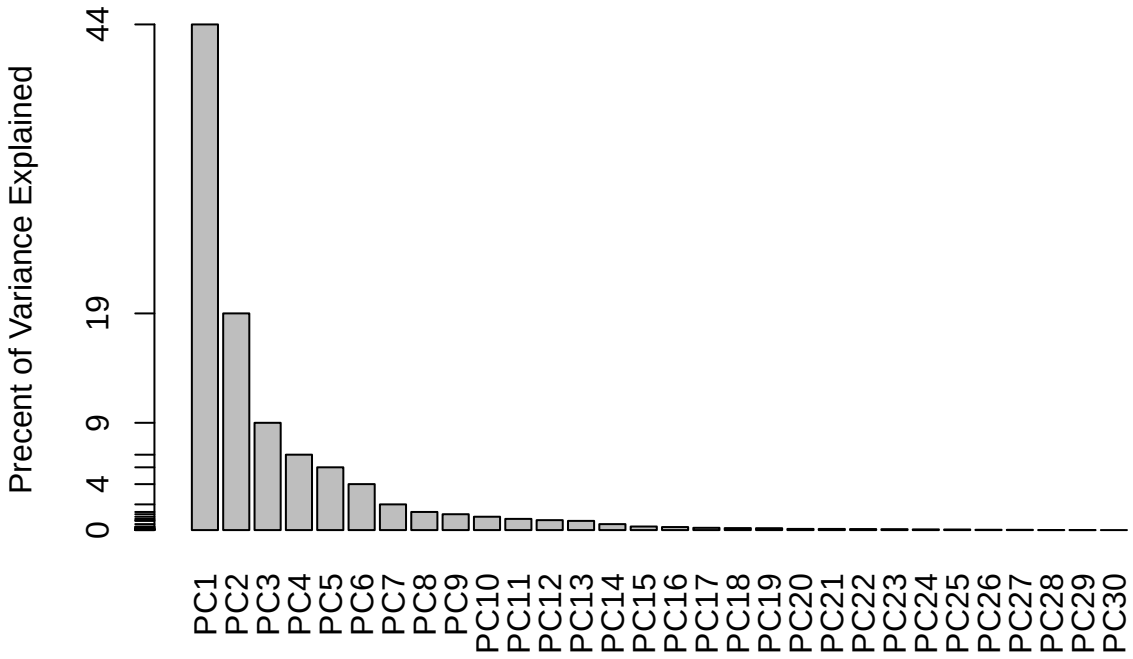
```
# Calculate variance of each component
pr.var <- wisc.pr$sdev^2
head(pr.var)
```

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

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

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



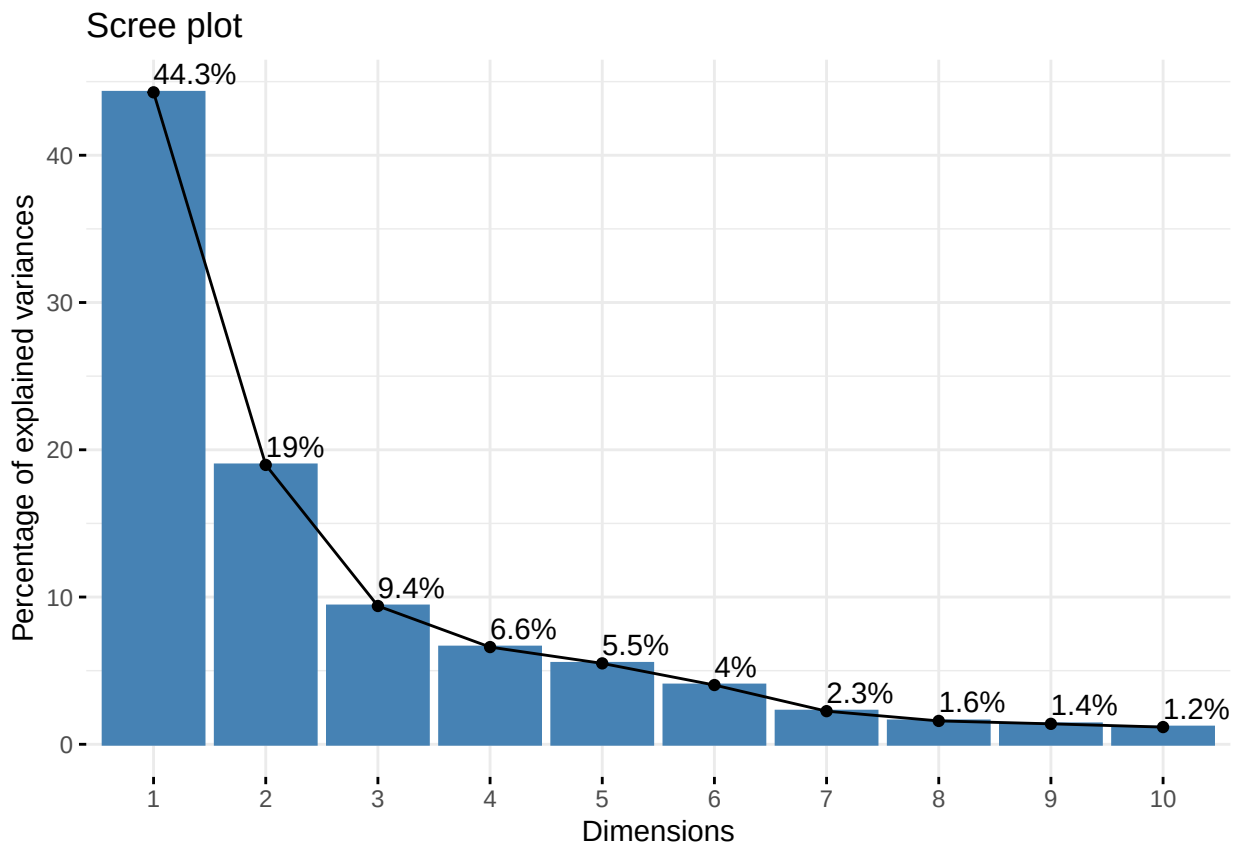

```
# Alternative scree plot of the same data, note data driven y-axis
barplot(pve, ylab = "Precent of Variance Explained",
     names.arg=paste0("PC",1:length(pve)), las=2, axes = FALSE)
axis(2, at=pve, labels=round(pve,2)*100 )
```




```
## (optional) ggplot based graph
#install.packages("factoextra")
library(factoextra)
```

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

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



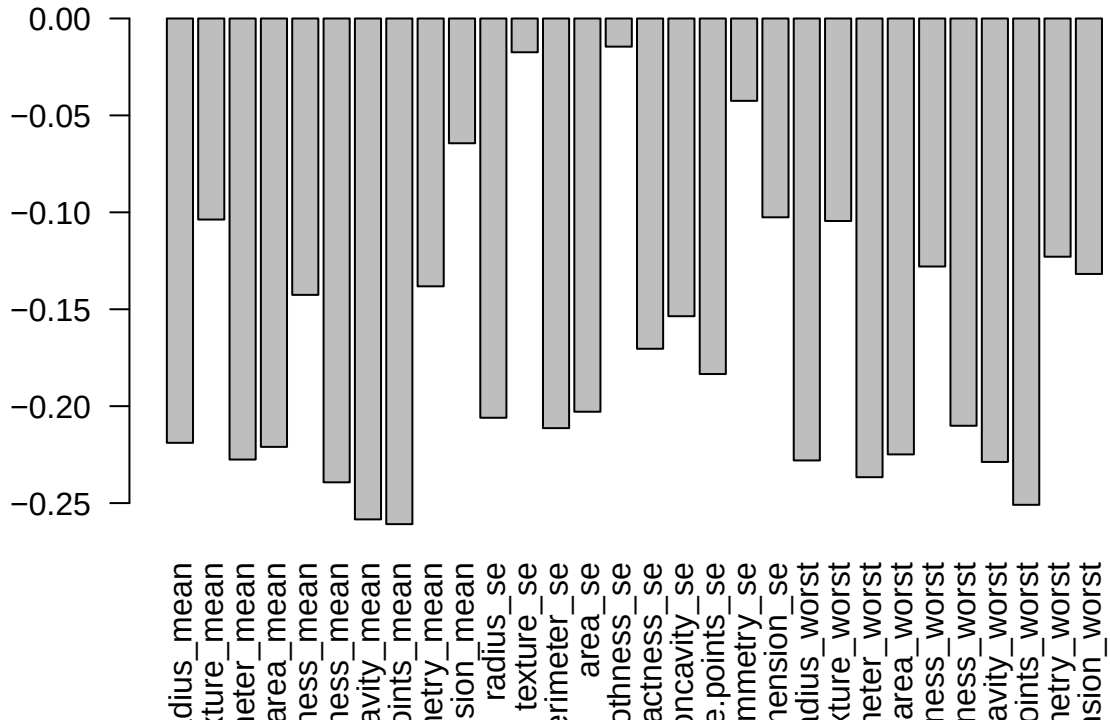
Communicating PCA results

```
# 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?

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
```

```
barplot(wisc.pr$rotation[,1], las=2)
```




```
# A. For PC1, the loading vector for concave.points_mean has a component
# of -0.26085376.


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

# A. A minimum of 5 PCs are required to explain 80% of the variance of the data.
```

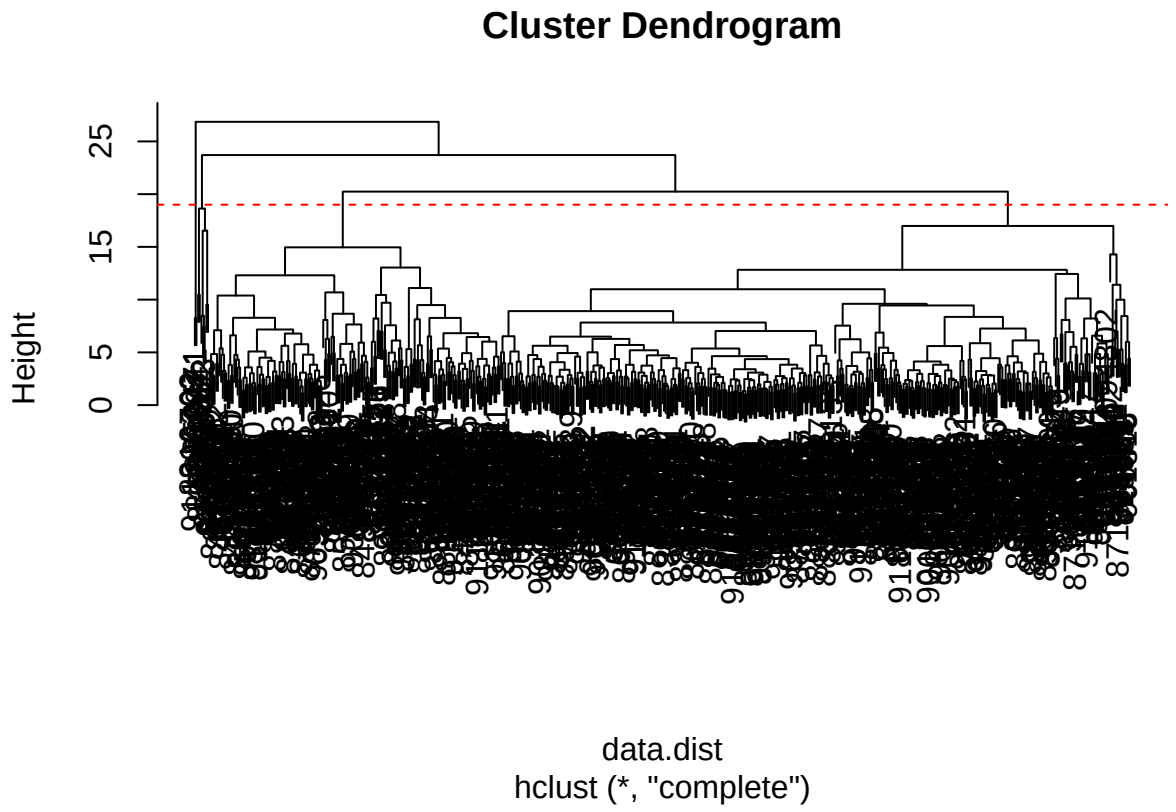
3. Hierarchical clustering

```
# Scale the wisc.data data using the "scale()" function
data.scaled <- scale(wisc.data)

# Calculating the Euclidean distance between pairs:
data.dist <- dist(data.scaled)
```

```
# Create a hierarchical clustering model using complete linkage:
wisc.hclust <- hclust(data.dist, "complete")
```

Results of hierarchical clustering

```
# 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)
```




```
# A. At h = 19, there are 4 clusters in the hierarchical clustering model.
```

Selecting number of clusters

```
# Use cutree() to cut the tree so that it has 4 clusters.
wisc.hclust.clusters <- cutree(wisc.hclust, k = 4)

# Use table() to compare the 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
```

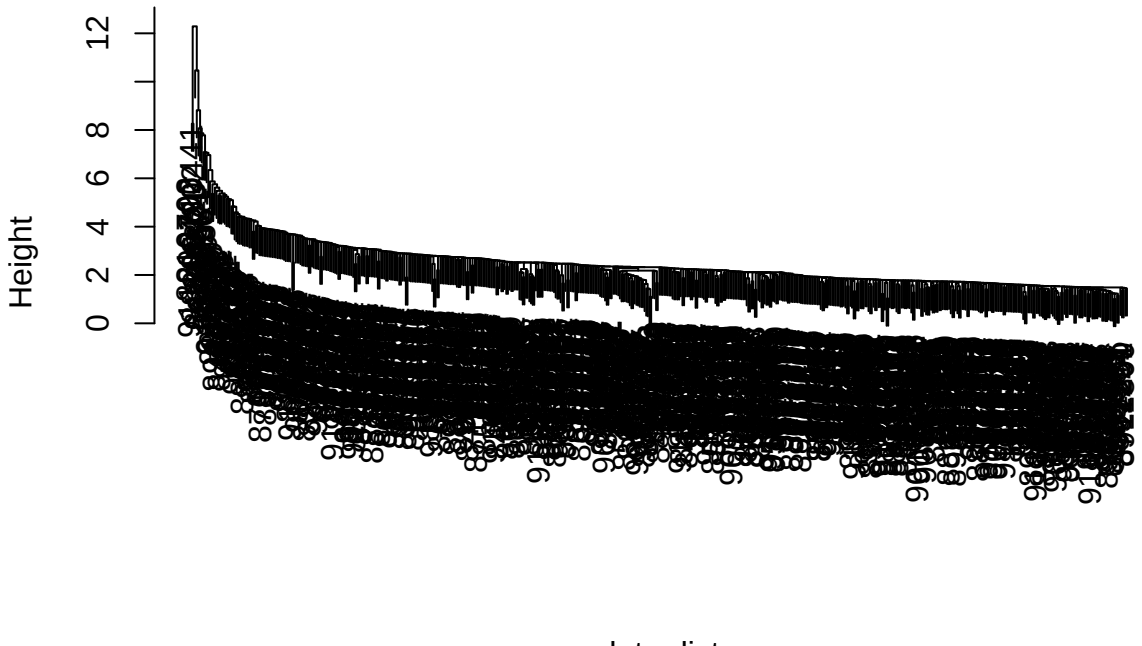
```
# Q12. Can you find a better cluster vs diagnoses match by cutting into a
# different number of clusters between 2 and 10?

wisc.hclust.clusters.5 <- cutree(wisc.hclust, k = 5)
table(wisc.hclust.clusters.5, diagnosis)
```

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

```
# A. Having 5 hierarchical clusters seemed to generate a slightly better match.
# It further sorted cluster 2 from the k = 4 clusters into its own cluster, with
# less overlap between B and M in each cluster.
```

Using different methods

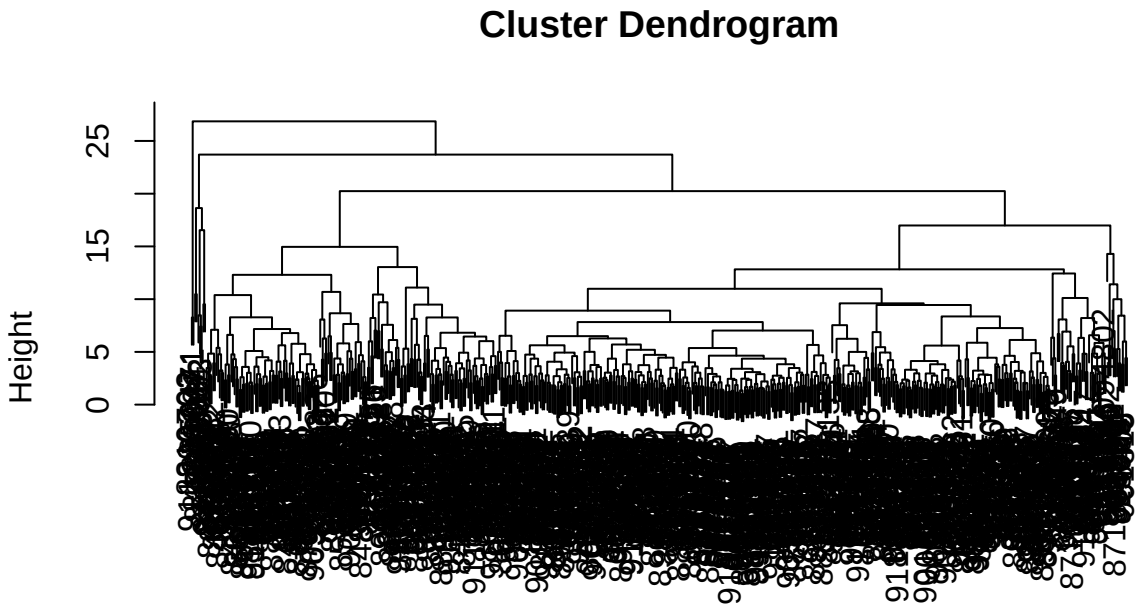
```
# Single (me too :')
wisc.hclust.s <- hclust(data.dist, "single")
plot(wisc.hclust.s)
```




```
wisc.hclust.s.clusters <- cutree(wisc.hclust.s, k = 5)
table(wisc.hclust.s.clusters, diagnosis)
```

```
##                         diagnosis
## wisc.hclust.s.clusters   B   M
##                      1 356 209
##                      2   1   0
##                      3   0   1
##                      4   0   1
##                      5   0   1
```

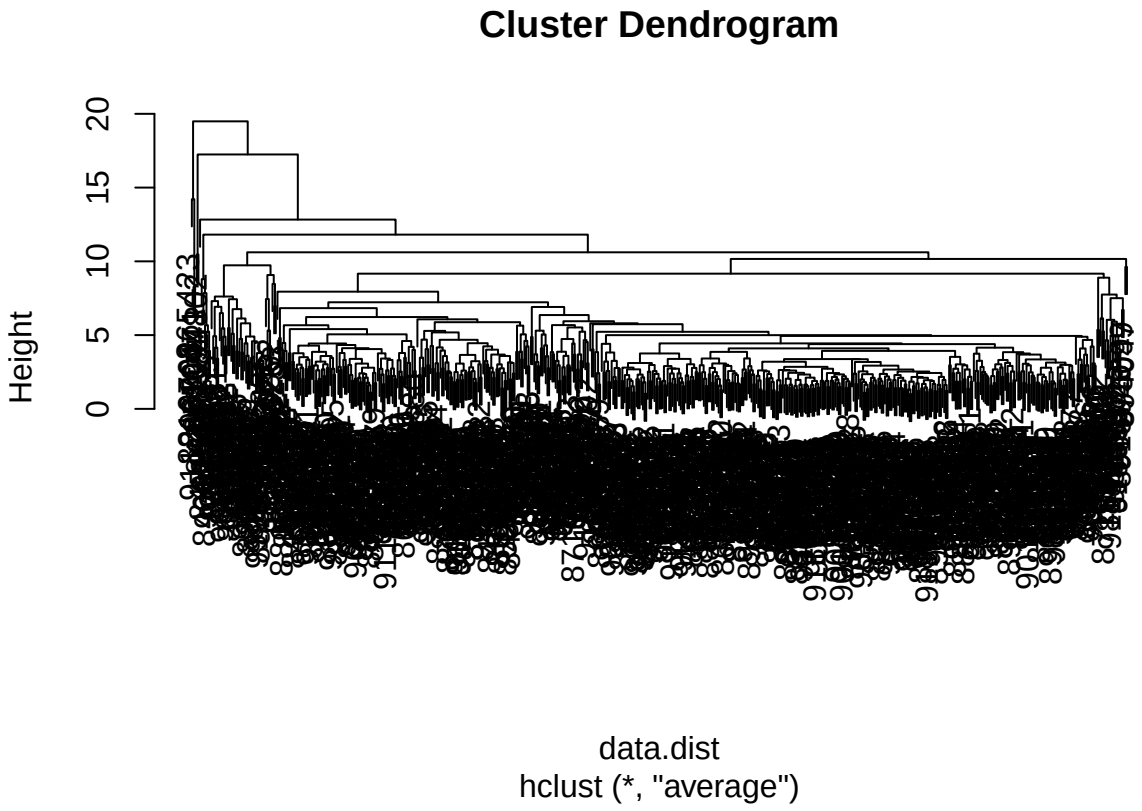
```
# Complete
wisc.hclust.c <- hclust(data.dist, "complete")
plot(wisc.hclust.c)
```




```
wisc.hclust.c.clusters <- cutree(wisc.hclust.c, k = 5)
table(wisc.hclust.c.clusters, diagnosis)
```

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

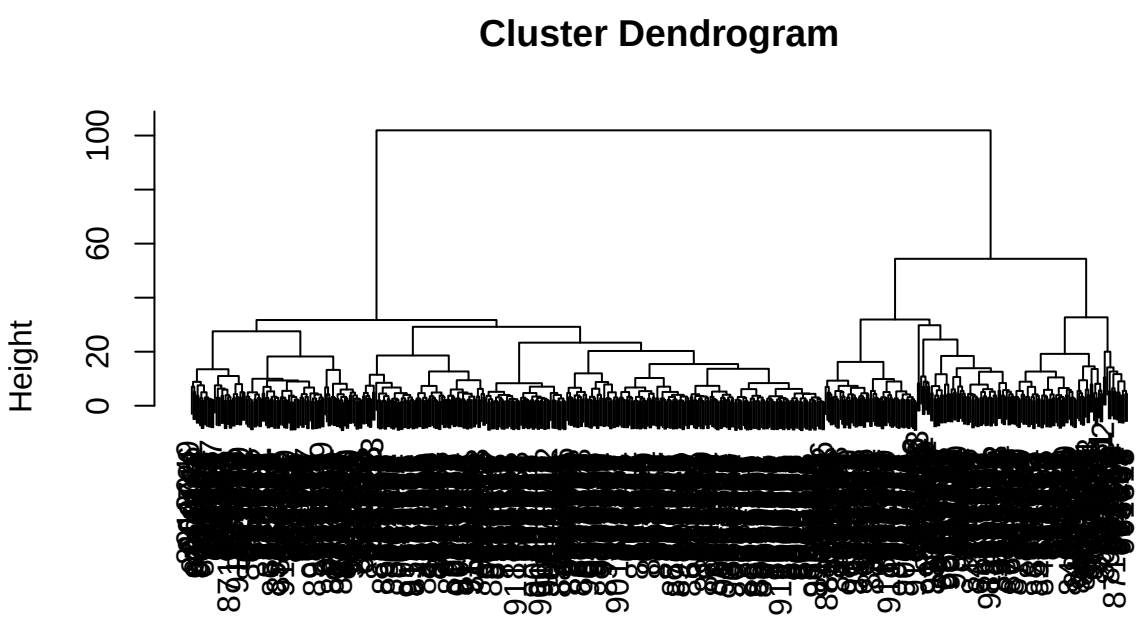
```
# Average
wisc.hclust.a <- hclust(data.dist, "average")
plot(wisc.hclust.a)
```




```
wisc.hclust.a.clusters <- cutree(wisc.hclust.a, k = 5)
table(wisc.hclust.a.clusters, diagnosis)
```

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

```
# ward.D2
wisc.hclust.w <- hclust(data.dist, "ward.D2")
plot(wisc.hclust.w)
```




```
wisc.hclust.w.clusters <- cutree(wisc.hclust.w, k = 5)
table(wisc.hclust.w.clusters, diagnosis)
```

```
##                       diagnosis
## wisc.hclust.w.clusters   B   M
##                      1   0  59
##                      2   0  56
##                      3   6  48
##                      4 337  48
##                      5  14   1
```

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

# A. For k = 5, both "complete" and "ward.D2" generated relatively distinct
# separation between benign and malignant diagnoses, whereas "single" and
# "average" fail to separate the dataset based on diagnoses. I like "ward.D2"
# the most because it generates the most orderly hierarchy of the four methods.
```

4. OPTIONAL: K-means clustering

K-means clustering and comparing results

```
# Create the k-means model
wisc.km <- kmeans(scale(wisc.data), centers= 2, nstart= 20)

# Compare results with table()
table(wisc.km$cluster, diagnosis)
```

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

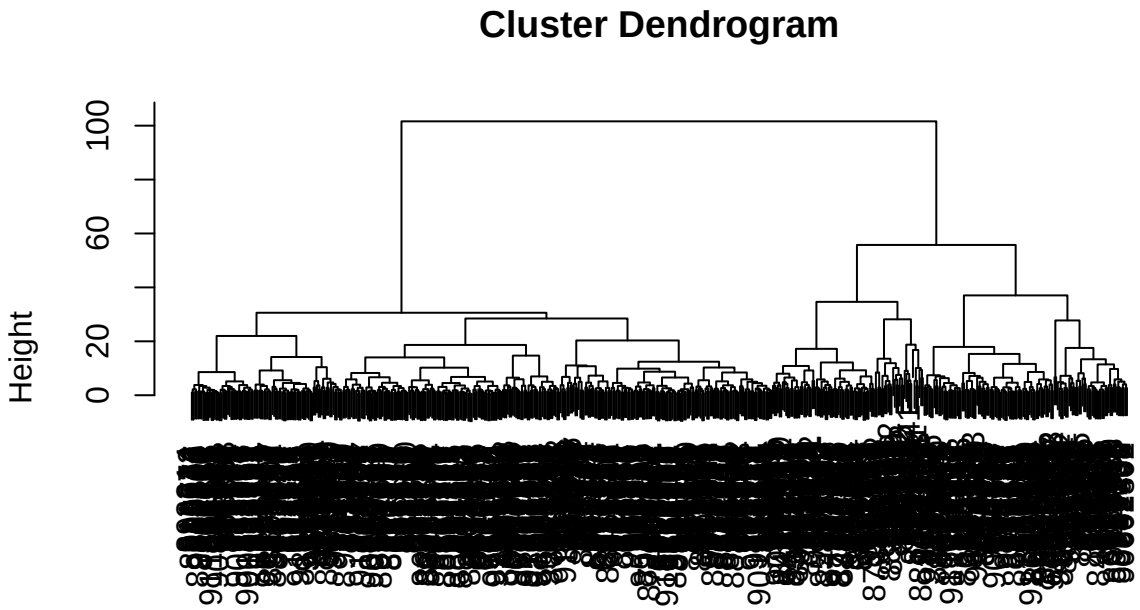
```
# (Optional) Q14. How well does k-means separate the two diagnoses? How does it
# compare to your hclust results?

# A. It separates the two diagnoses relatively well. It is similar to hclust
# in terms of separation. However, it only needs 2 clusters for the separation.


# Compare k-mean results with hclust results
table(wisc.hclust.clusters, wisc.km$cluster)
```

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

5. Combining methods

Clustering on PCA results

```
# Create a hierarchical clustering model with the linkage method="ward.D2"
dist.pr <- dist(wisc.pr$x[,1:7])
wisc.pr.hclust <- hclust(dist.pr, "ward.D2")
plot(wisc.pr.hclust)
```



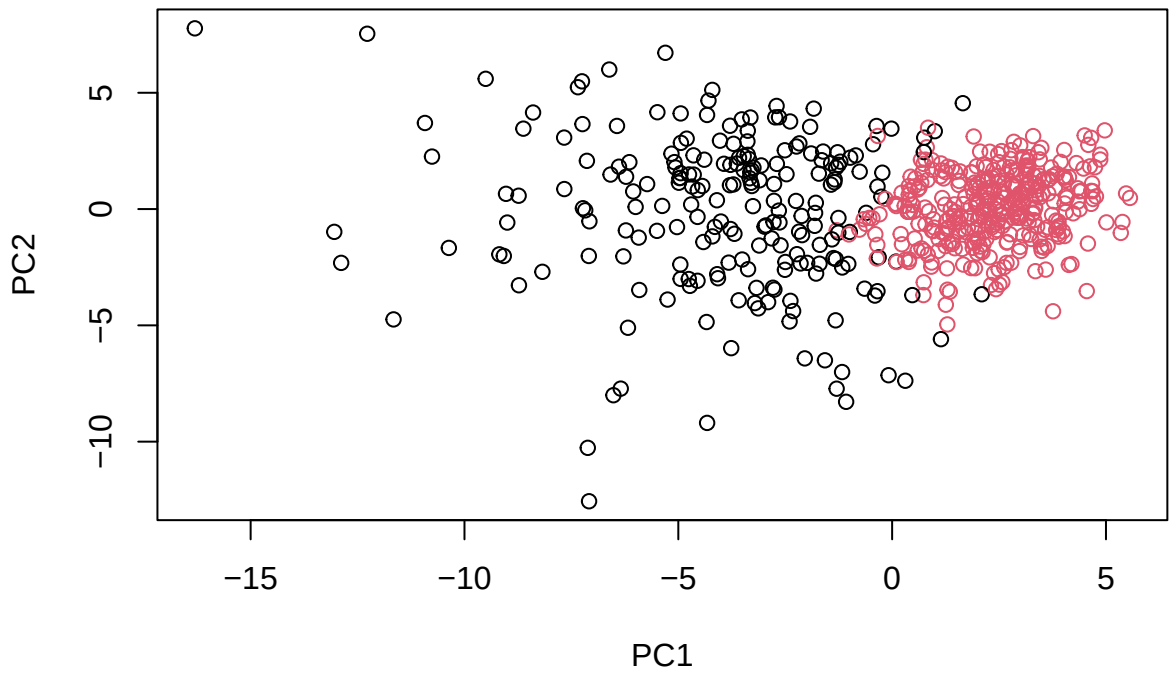

```
# Analyze the content of the two main branches
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
## grps
##   1   2
## 216 353
```

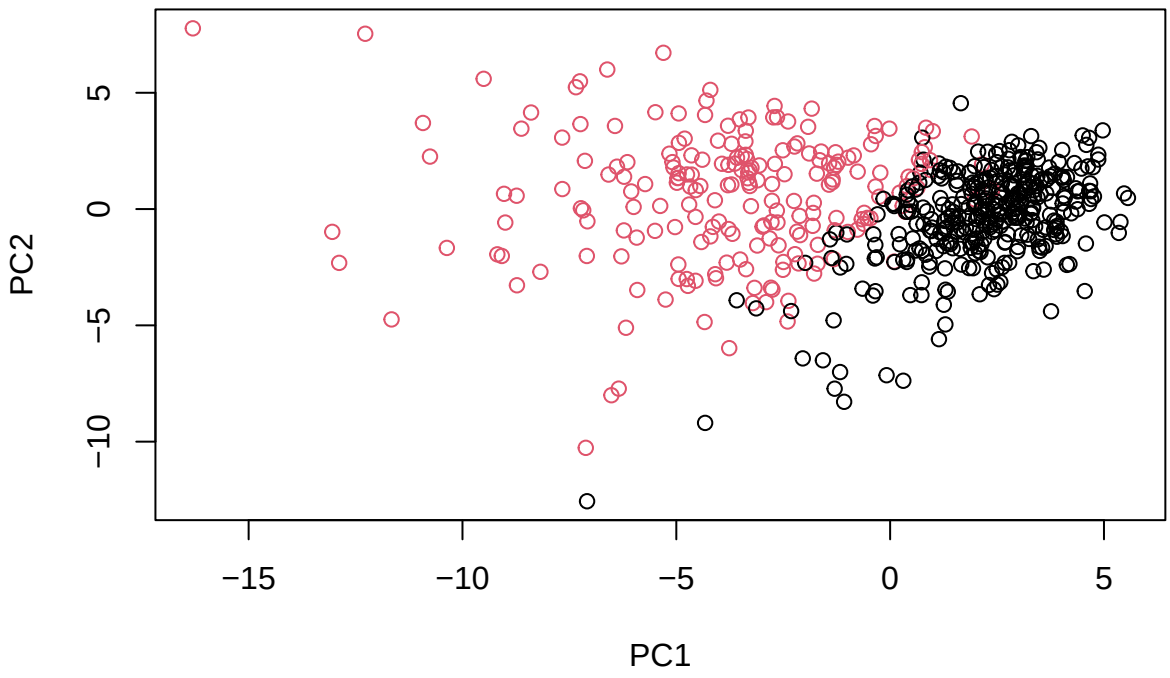
```
table(grps, diagnosis)
```

```
##     diagnosis
## grps   B   M
##    1  28 188
##    2 329  24
```

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




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



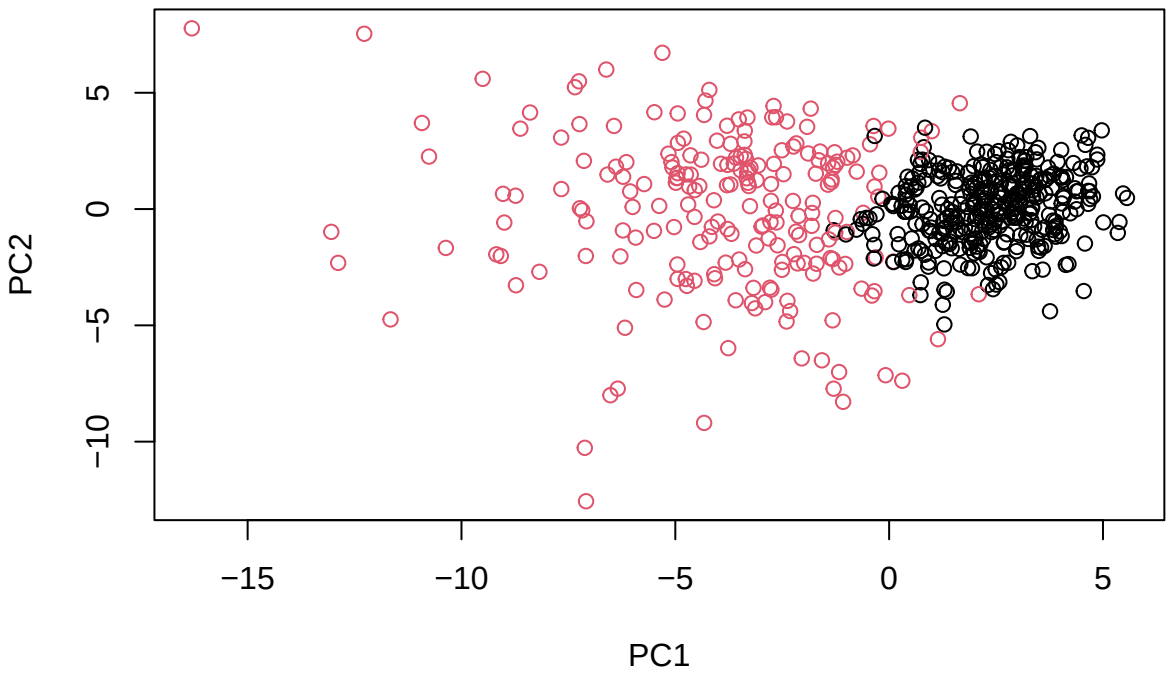

```
# Oops colors are switched. Fixing...
g <- as.factor(grps)
levels(g)
```

```
## [1] "1" "2"
```

```
g <- relevel(g,2)
levels(g)
```

```
## [1] "2" "1"
```

```
# Plot using our re-ordered factor (color fixed)
plot(wisc.pr$x[,1:2], col=g)
```




```
# Time to make a fancy 3-D plot
library(rgl)
plot3d(wisc.pr$x[,1:3], xlab="PC 1", ylab="PC 2", zlab="PC 3", cex=1.5, size=1, type="s", col=grps)


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

table(grps, diagnosis)
```

```
##     diagnosis
## grps   B   M
##    1  28 188
##    2 329  24
```

```
# A. The clusters of this new model show clear separation between benign and
# malignant diagnoses, with group 1 largely corresponding to malignant and group
# 2 to benign.


# 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?

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

```
##    diagnosis
```

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

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

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

```
# A. Both separated the diagnoses relatively well. K-mean produced 2 clusters,
# which are closer to the binary category of benign vs. malignant than the 4
# clusters generated by hierarchical clustering are.
```

6. Sensitivity/Specificity

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

# For sensitivity:
sen.combined <- 188/(188+24)
sen.k <- 175/(175+37)
sen.h <- (165+5+2)/(165+5+40+2)
order(c(sen.combined, sen.k, sen.h), decreasing = TRUE)
```

```
## [1] 1 2 3
```

```
# For specificity:
spe.combined <- 329/(24+329)
spe.k <- 343/(343+37)
spe.h <- 343/(40+343)
order(c(spe.combined, spe.k, spe.h), decreasing = TRUE)
```

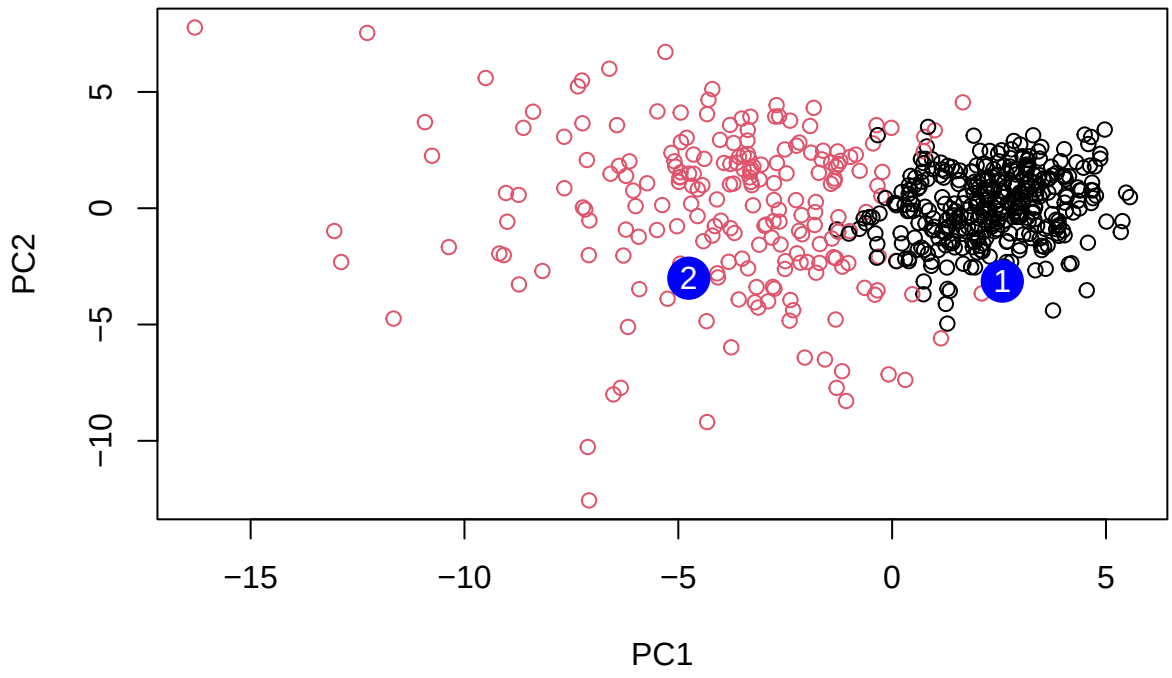
```
## [1] 1 2 3
```

```
# A. The combined clustering model has both the best sensitivity and the best
# specificity.
```

7. Prediction

```
#url <- "new_samples.csv" Import and predict new data based on existing PCA
url <- "https://tinyurl.com/new-samples-CSV"
new <- read.csv(url)
```

```
npc <- predict(wisc.pr, newdata=new)

# Projecting new data onto PCA space
plot(wisc.pr$x[,1:2], col=g)
points(npc[,1], npc[,2], col="blue", pch=16, cex=3)
text(npc[,1], npc[,2], c(1,2), col="white")
```




```
# Q18. Which of these new patients should we prioritize for follow up based on
# your results?

# A. Patient 2 should be prioritized as they fall into the same cluster as known
# malignant diagnoses.
```

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
```

```
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] rgl_0.108.3      factoextra_1.0.7 ggplot2_3.3.5
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_1.1.1  xfun_0.29         purrr_0.3.4       carData_3.0-5
##  [5] colorspace_2.0-2  vctrs_0.3.8       generics_0.1.2    htmltools_0.5.2
##  [9] yaml_2.2.2        utf8_1.2.2        rlang_1.0.1       pillar_1.7.0
## [13] ggpubr_0.4.0      glue_1.6.1        withr_2.4.3       lifecycle_1.0.1
## [17] stringr_1.4.0     munsell_0.5.0     ggsignif_0.6.3    gtable_0.3.0
## [21] htmlwidgets_1.5.4 evaluate_0.14     labeling_0.4.2    knitr_1.37
## [25] fastmap_1.1.0     fansi_1.0.2       highr_0.9         broom_0.7.12
## [29] Rcpp_1.0.8        scales_1.1.1      backports_1.4.1   jsonlite_1.7.2
## [33] abind_1.4-5       farver_2.1.0      digest_0.6.29     stringi_1.7.6
## [37] rstatix_0.7.0     dplyr_1.0.7       ggrepel_0.9.1     grid_4.1.2
## [41] cli_3.1.1         tools_4.1.2       magrittr_2.0.2    tibble_3.1.6
## [45] crayon_1.4.2      tidyr_1.2.0       car_3.0-12        pkgconfig_2.0.3
## [49] ellipsis_0.3.2    rmarkdown_2.11    rstudioapi_0.13   R6_2.5.1
## [53] compiler_4.1.2
```