STAT 380 – Clustering Part 3 (Lecture 20)

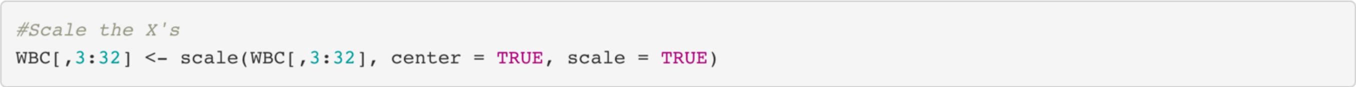
IDEA: In this lecture, we will apply K-means clustering to real data set.

EXAMPLE 1: The L20_WBC.csv contains information on a study related to breast cancer that took place in Wisconsin. More information about the dataset can be found at:

https://www.kaggle.com/uciml/breast-cancer-wisconsin-data?select=data.csv

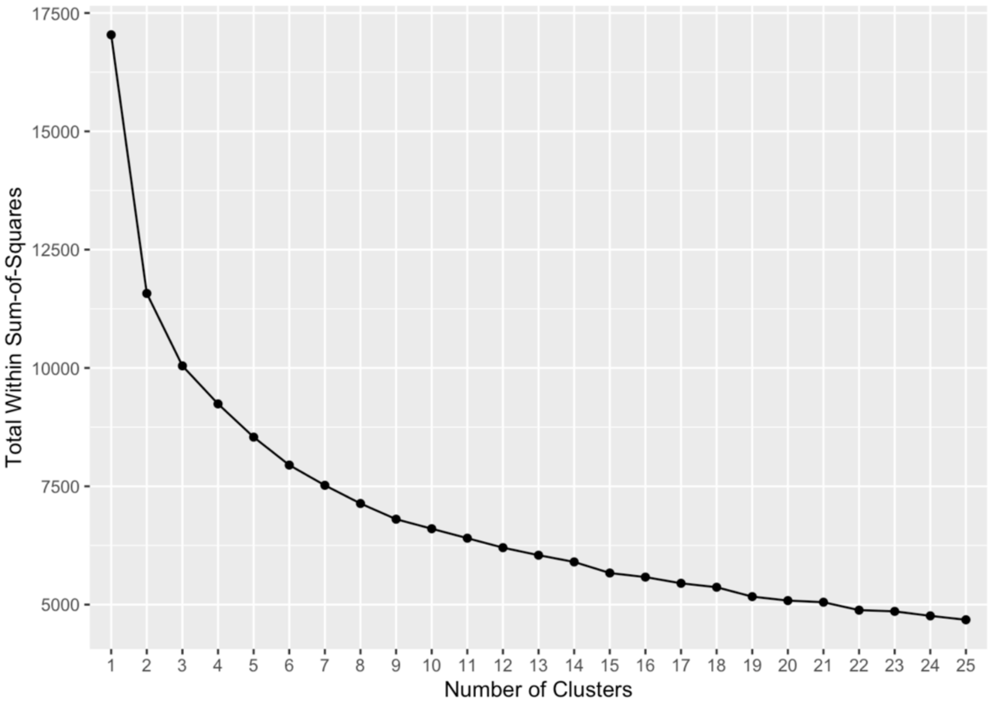
Briefly the data contain 32 columns. The first two columns correspond to the patient's id and diagnosis (M=malignant, B=Benign). The remaining 30 numeric columns are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. Our goal is to use the 30 numeric columns to perform K-means clustering.

a. Before performing K-means, standardize the inputs to have a mean of 0 and a standard deviation of 1.



```
#Scale the X's
WBC[,3:32] <- scale(WBC[,3:32], center = TRUE, scale = TRUE)
```

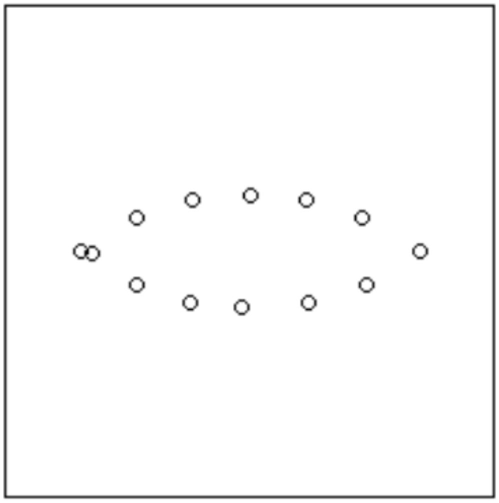
b. Perform K-means for values of K ranging from 1 to 25. For each value of K use 25 random initializations and set the max number of iterations to 20. Use a seed of 123. Create a plot showing the total within sum of squares as a function of K. Which value of K would you choose?

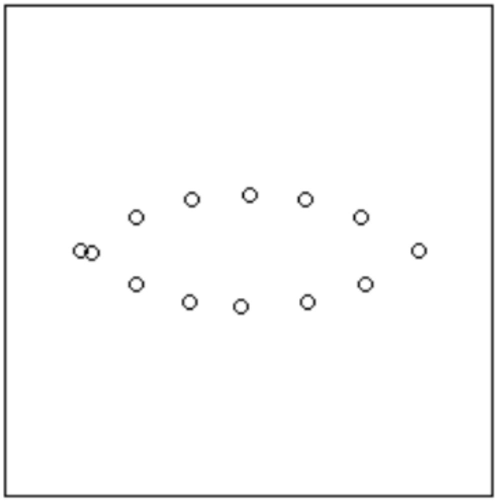


IDEA: It would be nice to visualize the results; however, we clustered in a 30 dimensional space. We learned how to create 2D plots (x and y axis) and sometimes we have added color, shape, etc. to represent a 3rd or 4th input (usually a categorical input). Unfortunately, we no way to display all 30 dimensions.

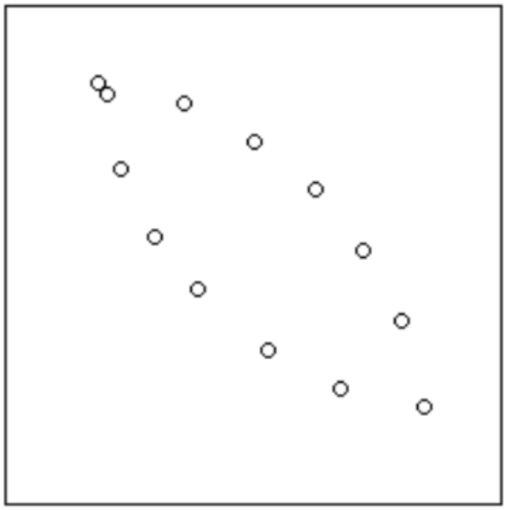
NOTE: One way to handle this problem is to reduce the dimensionality of the data. A common method for doing this is Principal Component Analysis (PCA). PCA seeks to find a set of orthogonal directions that exhibit the greatest amount of variability. These directions are called the principal components.

EXAMPLE 2: Consider the data shown below. Assume the scale for both dimensions is the same in both directions. Find a direction (in other words a straight line) that will exhibit the most spread when the data are projected onto the straight line.





EXAMPLE 3: Consider the data shown below. Assume the scale for both dimensions is the same in both directions. Find a direction (in other words a straight line) that will exhibit the most spread when the data are projected onto the straight line. If you were asked to find a second direction that exhibits the second most variation, what would it be?



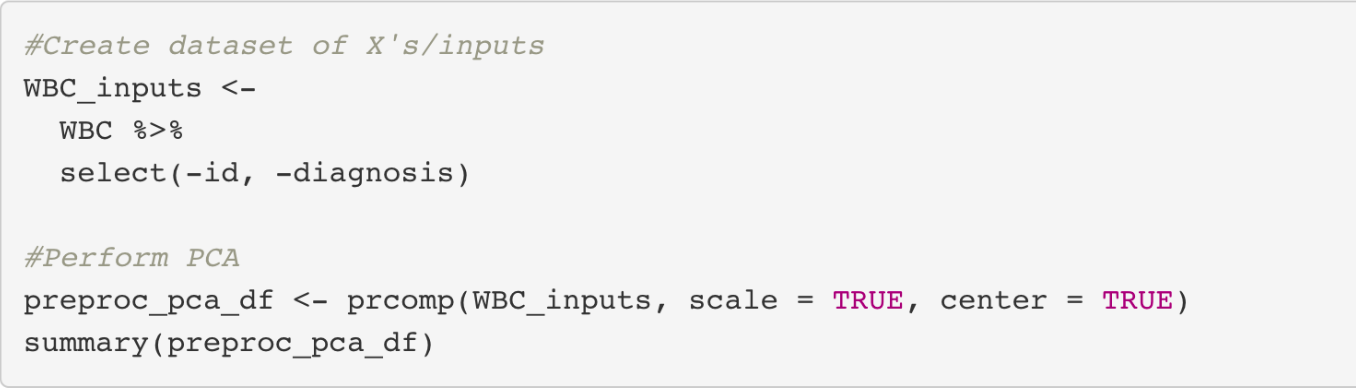
Definition: The principal components are the directions in which there is the most variance. In other words, the principal components are the directions where the data in most spread out.

NOTE: The principal components are NOT limited to the dimensions we have (such as x_1, x_2, and x_3 in the case of 3D data). Instead, the principal components are linear combinations of the dimensions.

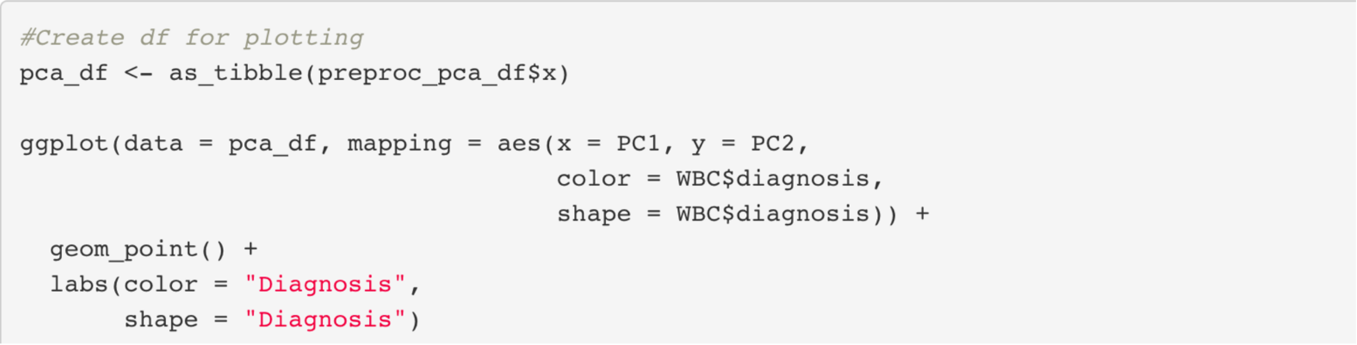
NOTE: IN 2D, when we can easily visualize the data, it is fairly easy to find the direction(s) in which the data exhibit the most variation. How do we do this in 3D? What about 30D when we cannot visualize the data?

IDEA: When we get a set of data points, we can deconstruct the set into eigenvectors and eigenvalues. Eigenvectors and eigenvalues exist in pairs: every eigenvector has a corresponding eigenvalue. An eigenvector is a direction. An eigenvalue is a number, telling you how much variance there is in the data in that direction, In the examples above the eigenvalue is a number telling us how spread out the data are on the line. The eigenvector with the highest eigenvalue is therefore the principal component.

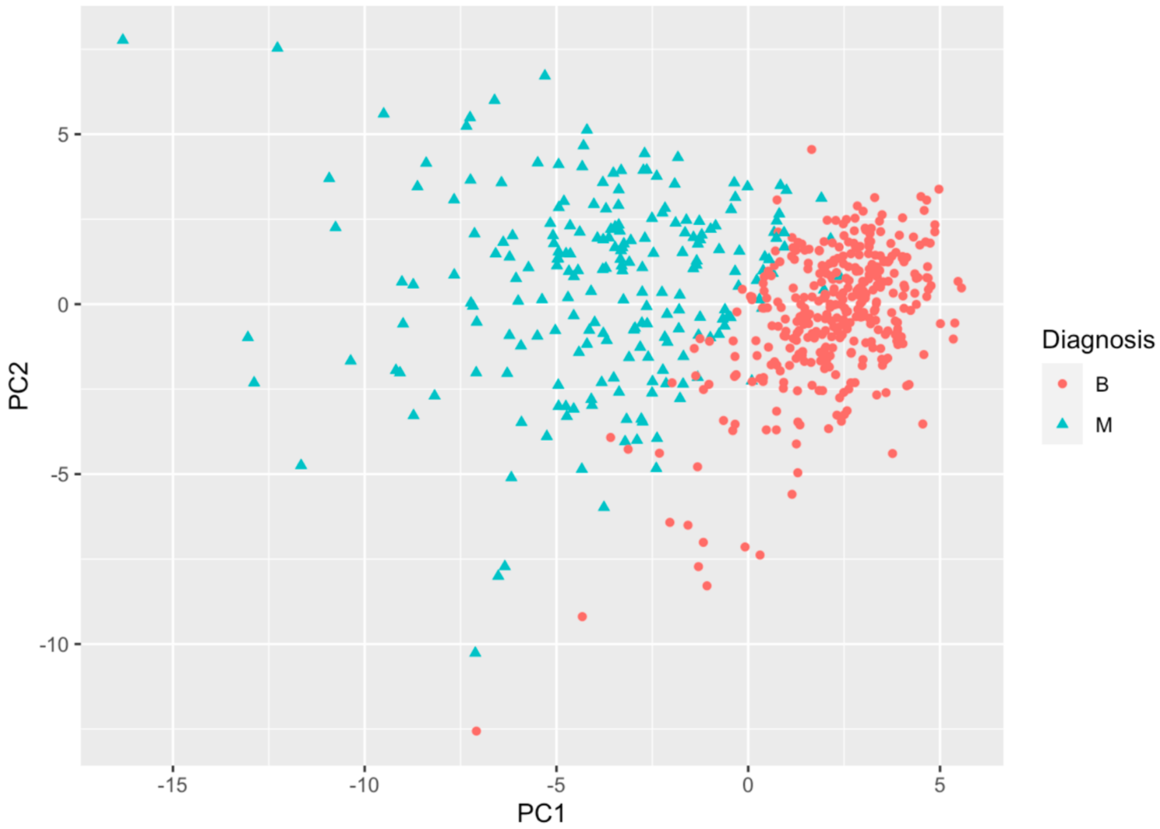
EXAMPLE 4: Perform PCA on the WBC data. Use the prcomp function from the stats (base R) library. Create a plot of the WBC using the first 2 principal components. Add color and shape based on the diagnosis.



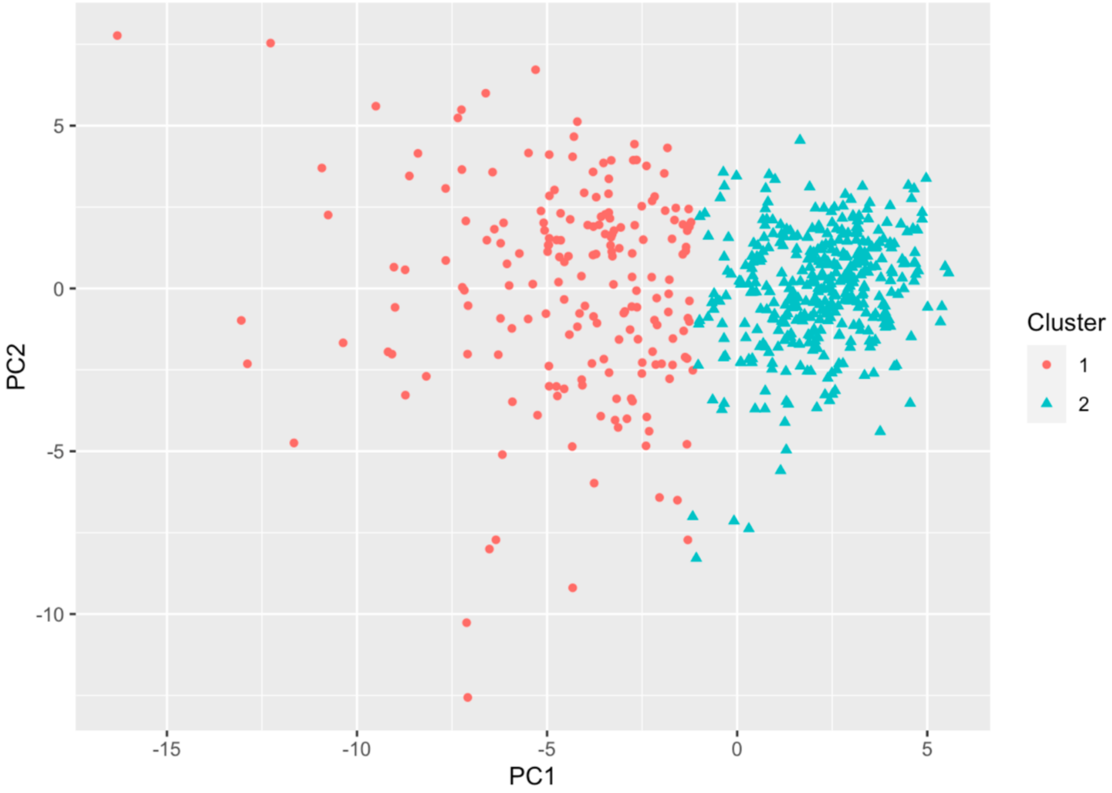
```
#Create dataset of X's/inputs
WBC_inputs <-
  WBC %>%
  select(-id, -diagnosis)

#Perform PCA
preproc_pca_df <- prcomp(WBC_inputs, scale = TRUE, center = TRUE)
summary(preproc_pca_df)
```



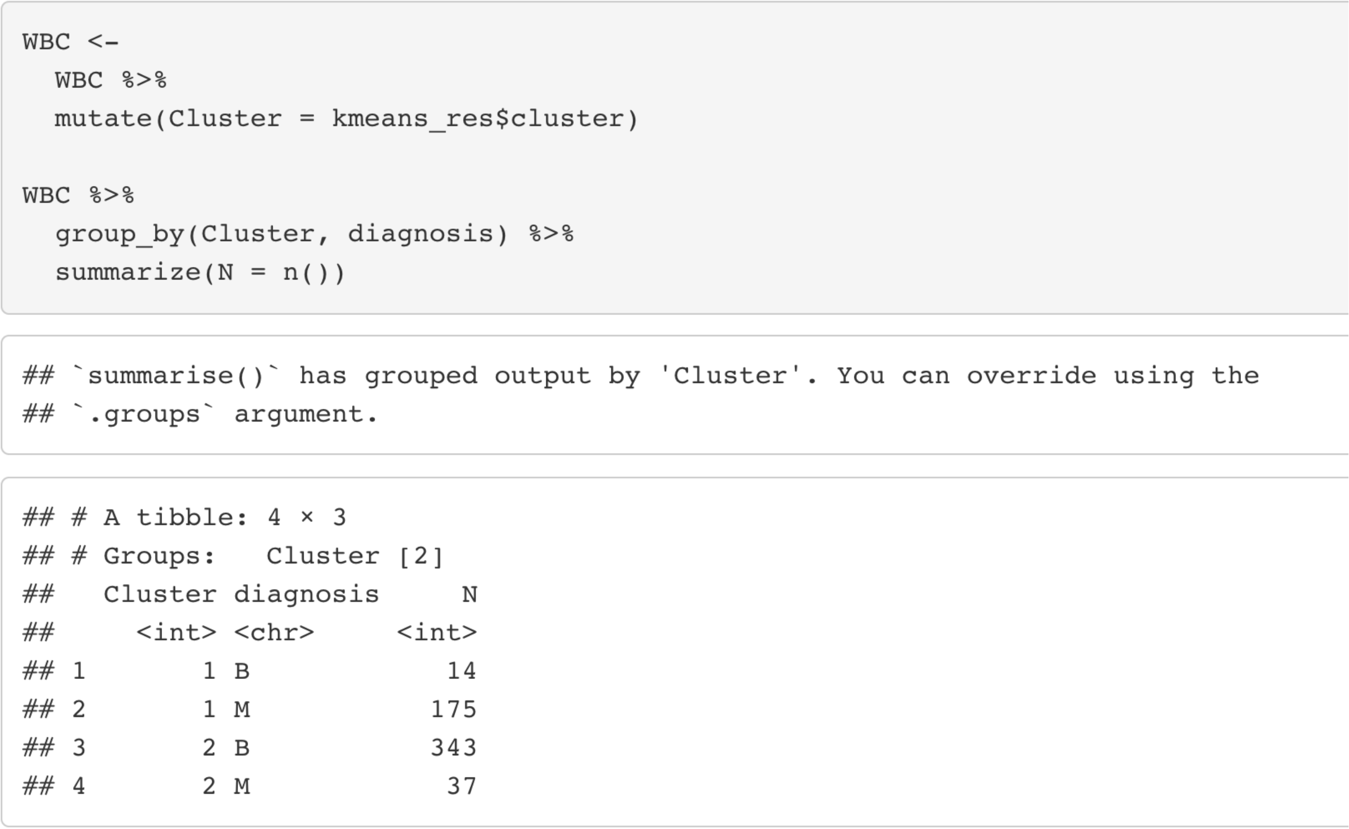
```
#Create df for plotting
pca_df <- as_tibble(preproc_pca_df$x)

ggplot(data = pca_df, mapping = aes(x = PC1, y = PC2,
                                    color = WBC$diagnosis,
                                    shape = WBC$diagnosis)) +
  geom_point() +
  labs(color = "Diagnosis",
       shape = "Diagnosis")
```



EXAMPLE 5: Run K-means clustering on the WBC using K = 2. Use 25 initializations, 20 iterations, and a seed of 123. Recreate the plot from the last example adding shape and color based on the cluster assignments instead of diagnosis. (Code is in STAT380_L20.Rmd.) What do you notice?



EXAMPLE 6: Determine Counts of diagnosis broken down by cluster assignment.



```
WBC <-
  WBC %>%
  mutate(Cluster = kmeans_res$cluster)

WBC %>%
  group_by(Cluster, diagnosis) %>%
  summarize(N = n())
```

```
## `summarise()` has grouped output by 'Cluster'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 4 × 3
## # Groups:   Cluster [2]
##   Cluster diagnosis     N
##     <int> <chr>     <int>
## 1       1 B            14
## 2       1 M           175
## 3       2 B           343
## 4       2 M            37
```