Chase Moreno

@2078503

MATH 4323

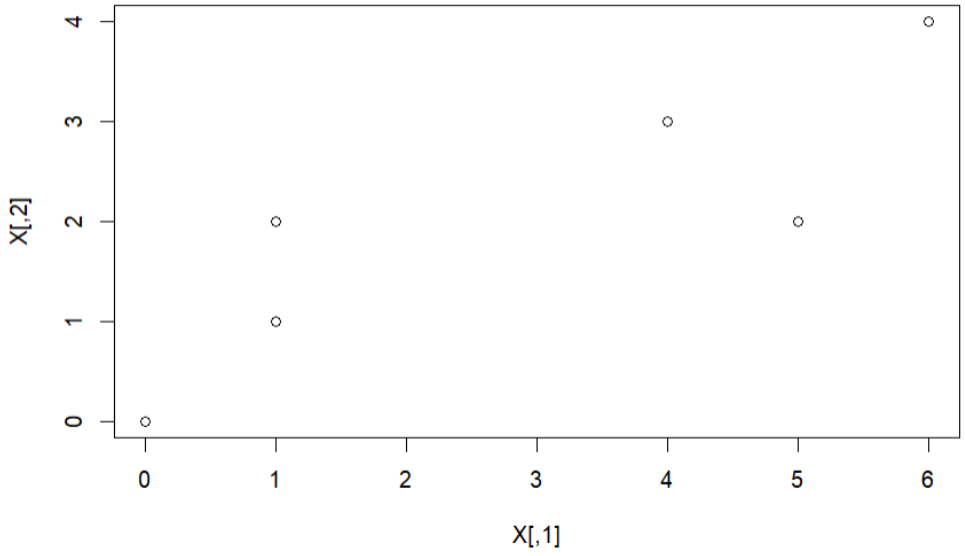
Homework 5

1. In this problem, you will perform K-means clustering manually, with K = 2, on a small example with n = 6 observations and p = 2 features. The observations are as follows.

Obs.	X_1	X_2
1	6	4
2	5	2
3	4	3
4	1	2
5	1	1
6	0	0

(a) Plot the observations.

```
> X <- cbind(c(6, 5, 4, 1, 1, 0), c(4, 2, 3, 2, 1, 0))
> plot(X)
```



(b) Randomly assign a cluster label to each observation as follows

RNGkind(sample.kind = "default")

set.seed(2)

labels <- sample(2, nrow(x), replace = T)

Report the cluster labels for each observation.

```
> RNGkind(sample.kind = "default")
> set.seed(2)
> labels <- sample(2, nrow(X), replace = T)
> labels
[1] 1 1 2 2 2 2

> Xlabeled <- cbind(X, labels)
> Xlabeled
          labels
[1,] 6 4      1
[2,] 5 2      1
[3,] 4 3      2
[4,] 1 2      2
[5,] 1 1      2
[6,] 0 0      2
```

(c) Compute the centroids for each cluster.

```
> croids <- aggregate(Xlabeled, by = list(Xlabeled[,3]), FUN = me
an)
> croids
  Group.1  V1  V2 labels
1       1 5.5 3.0      1
2       2 1.5 1.5      2
>
> croid1 <- as.matrix(centroids[1, 2:3])
> croid2 <- as.matrix(centroids[2, 2:3])
> croid1
   V1 V2
1 5.5  3
> croid2
   V1  V2
2 1.5 1.5
```

(d) Assign each observation to the centroid to which it is closest, in terms of Euclidean distance. Report the cluster labels for each observation.

```
> dist <- as.matrix(dist(rbind(X, croid1, croid2)))
> dist <- dist[-c(nrow(dist), nrow(dist) - 1), c(nrow(dist) - 1,
nrow(dist))]
> rownames(dist) <- 1:nrow(X)
> dist
          1         2
1 1.118034 5.1478151
2 1.118034 3.5355339
3 1.500000 2.9154759
4 4.609772 0.7071068
5 4.924429 0.7071068
6 6.264982 2.1213203
```

Observations 1, 2, and 3 are closer to centroid 1 and observation 4, 5, and 6 are closer to centroid 2.

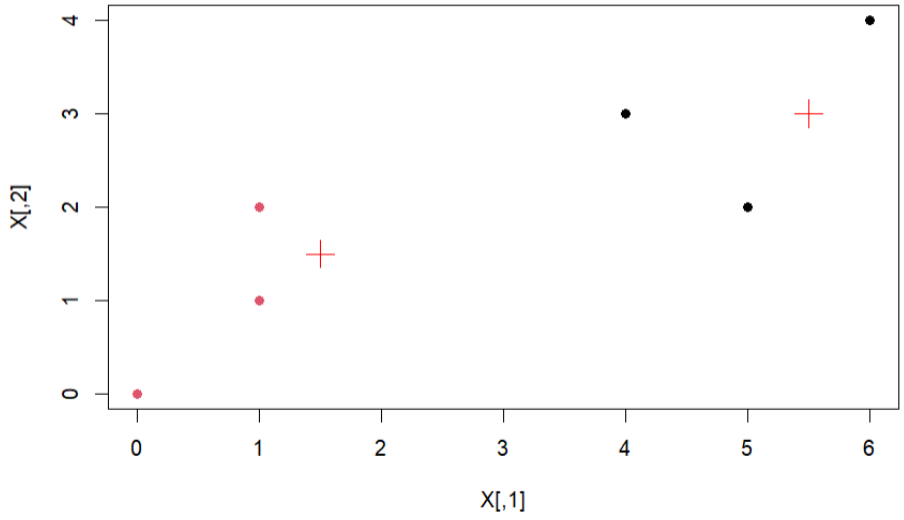
(e) Update the cluster assignments via K-Means algorithm until they stop changing. How many iterations did it take? Hint: it won't take long.

```
> library(factoextra)
> initial_croids <- as.matrix(croids[,2:3])
> km <- kmeans(X, centers = initial_croids)
> km$cluster
[1] 1 1 1 2 2 2
> km$iter
[1] 1
```

It took 1 iteration.

(f) Update your plot from (a), by coloring the observations according to the cluster labels obtained.

```
> plot(X, col = cluster_labels, pch = 16)
> points(croid1, col = "red", pch = 3, cex = 2)
> points(croid2, col = "red", pch = 3, cex = 2)
```



2. Considering a data set of temperature measurements for a certain place of interest, suppose you focus on an eight-hour period from 1:00 PM to 9:00 PM. The sensor reports the following temperatures:

Time	1:00	2:00	3:00	4:00	5:00	6:00	7:00	8:00	9:00
Temperature	48	54	55	60	62	61	56	55	51

You run a K-means clustering algorithm on 9 measurements, where k = 3, and the distance between items is the difference in temperature values.

(a) What are the resulting clusters? Please specify the clusters as sets of time values.

Cluster 1: the sets of time values are {1:00, 2:00, 3:00}

Cluster 2: the sets of time values are {4:00, 5:00, 6:00}

Cluster 3: the sets of time values are {7:00, 8:00, 9:00}

(b) List the cluster means corresponding to the clusters you listed in part (a).

Cluster 1: (48 + 54 + 55) / 3 = 52.333

Cluster 2: (60 + 62 + 61) / 3 = 61

Cluster 3: (56 + 55 + 61) / 3 = 54

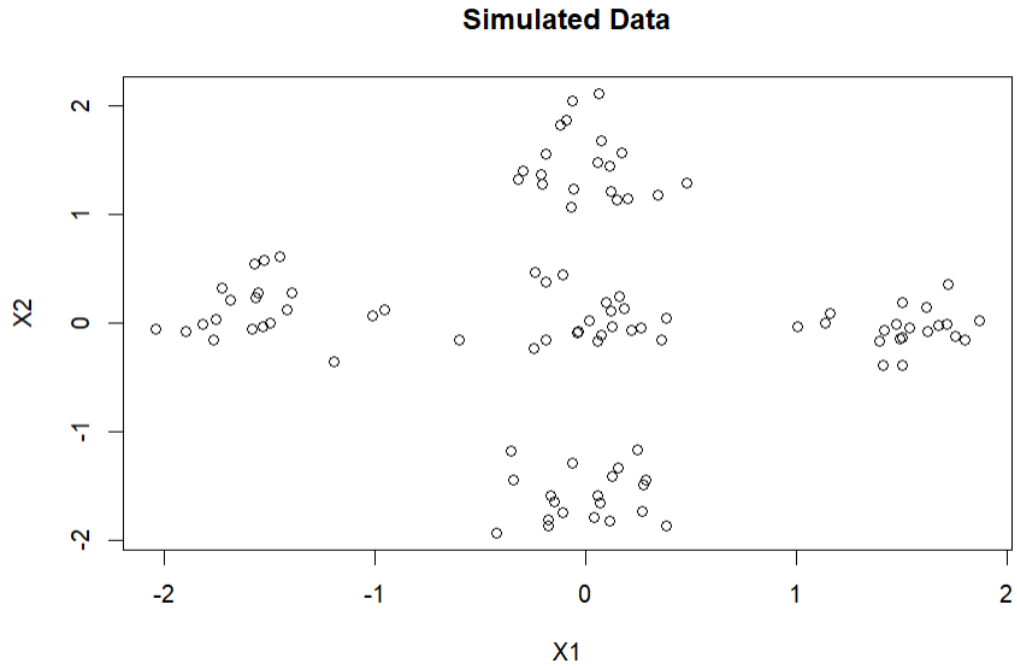
Applied.

3. In this problem, you will generate simulated data, and then perform K-means clustering on the data.

(a) Generate a simulated data set with 20 observations in each of K=5 well-separated clusters, with p = 2 variables describing each observation. Do it in similar fashion to K = 3 case in ”K-means: Selecting K” lecture slides. Plot the resulting data points.

My R code according to do similar way from Ch10 slide #8 but as K = 5.

```
> set.seed(1)
> x <- matrix(rnorm(20*2*5), ncol = 2)
> x[21:40, 1] <- x[21:40, 1] + 6
> x[41:60, 2] <- x[41:60, 2] + 6
> x[61:80, 1] <- x[61:80, 1] - 6
> x[81:100, 2] <- x[81:100, 2] - 6
> x <- scale(x)
> plot(x, xlab = "X1", ylab = "X2", main = "Simulated Data")
```



(b) Run K-means algorithm on your simulated data for K = 4, 5 & 6. Use 50 random starts in each case. Provide the code and report the total within-cluster sum of squares (WSS) for each solution. Which transition caused the bigger drop in total WSS, from K = 4 to K = 5, or from K = 5 to K = 6?

```
> library(factoextra)
```

```
> km.res4 <- eclust(data.frame(x), FUNcluster = "kmeans", k = 4,
nstart = 50)
> km.res4$withinss
[1] 23.481722  2.094286  3.538721  1.525863
> km.res4$tot.withinss
[1] 30.64059
```

The total WSS for k = 4 is 30.64059

```
> km.re5s <- eclust(data.frame(x), FUNcluster = "kmeans", k = 5,
nstart = 50)
> km.res5$withinss
[1] 1.525863 1.896800 2.614760 2.094286 2.600349
> km.res5$tot.withinss
[1] 10.73206
```

The total WSS for k = 5 is 10.73206

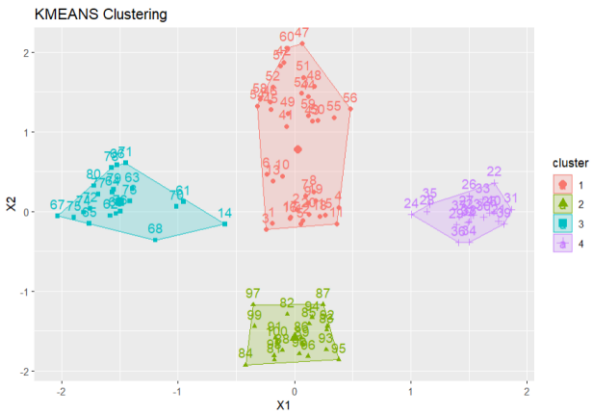
```
> km.res6 <- eclust(data.frame(x), FUNcluster = "kmeans", k = 6,
nstart = 50)
> km.res6$withinss
[1] 1.5258631 1.8968003 0.1526379 1.1281524 2.6003486 2.0942858
> km.res6$tot.withinss
[1] 9.398088
```

The total WSS for k = 6 is 9.398088

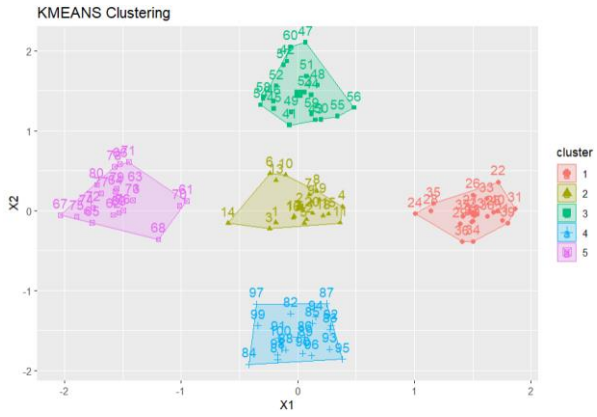
From k = 4 to k = 5 transition caused the bigger drop in total WSS.

(c) Proceed to plot the clustering solutions for K = 4, 5 & 6 (just use the plots automatically generated by eclust() function). Judging by the plots, explain why the respective sizes of WSS drops in part (b) were expected. Use lecture slides (the K = 3 simulation study) for reference.

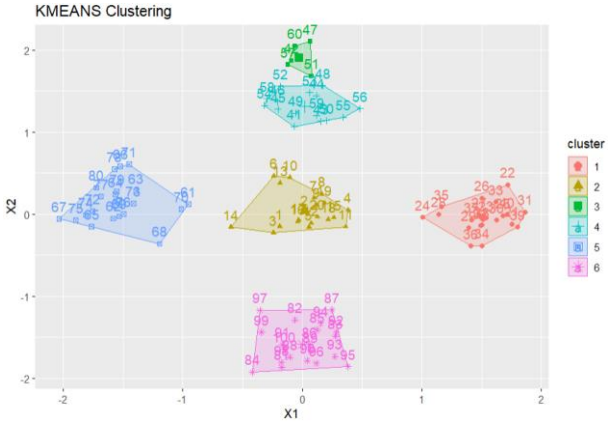
K = 4



K = 5



K = 6

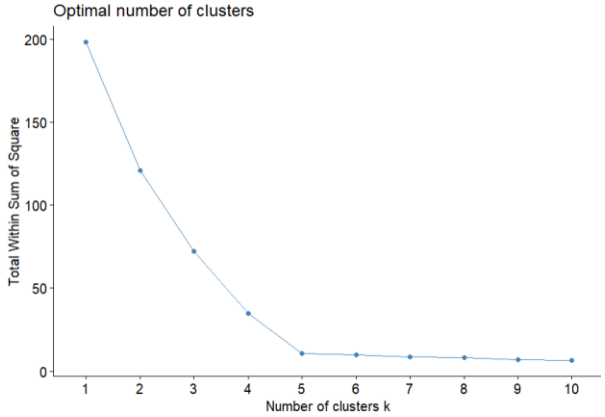


The reason why from K = 4 to K = 5 transition caused the bigger drop in total W SS because of the cluster 1 is large and heterogeneous in K = 4 plot than other K values. So, when it comes to K = 5, the cluster 1 from K = 4 broken down into 2 clusters which we see in K = 5 plot. This is why transition from K = 4 to K = 5 c aused a big drop in total WSS.

(d) Run K-means clustering on your simulated data for K = 1, 2, . . . , 10, record total WSS for each K value. Plot the progression of WSS values. According to ”elbow” method logic, which K value appears as the optimal one?

```
> kclust <- sapply(1:10, function(k) {
+   km.res <- eclust(data.frame(x), FUNcluster = "kmeans", k = k,
nstart = 50)
+   print(paste0("K = ", k, ": ", km.res$tot.withinss))
+ })
[1] "K = 1: 198"
[1] "K = 2: 112.384021670927"
[1] "K = 3: 68.4636658483147"
[1] "K = 4: 30.6405924841574"
[1] "K = 5: 10.7320573979859"
[1] "K = 6: 9.39808806009625"
[1] "K = 7: 8.13046116885735"
[1] "K = 8: 7.30438550390098"
[1] "K = 9: 6.62392171322251"
```

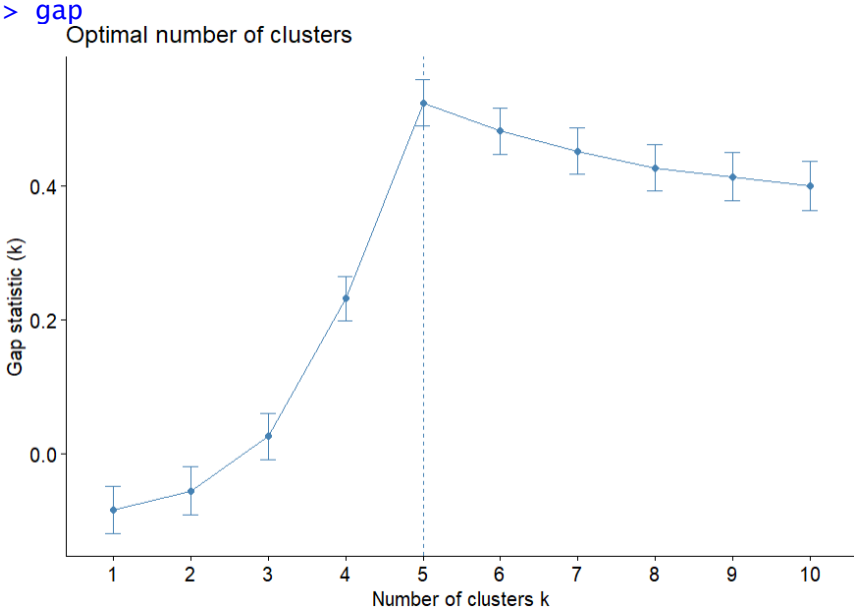
```
[1] "K = 10: 5.89367058710302"
```



K = 5 will be the optimal one.

(e) For a more formal approach (alternative to elbow method from part (d)), run the gap statistic calculations for K = 1, . . . , 10, with 50 replicates each. Which K value is optimal? Does it match with the # of well-separated clusters in our simulated data? Provide the plot of gap statistic values.

```
> gap <- fviz_nbclust(x, FUNcluster = kmeans, nstart = 50, method
= "gap_stat")
> gap
```

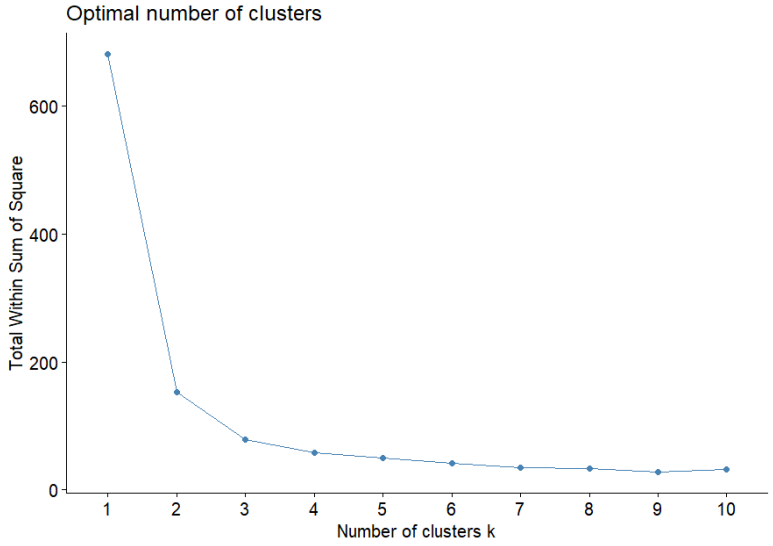


The optimal K value is 5 and yes it does match with the number of well-separated clusters in our simulated data.

4. This problem will work with iris data (available in base R). Make sure to exclude the Species column before applying clustering algorithms.

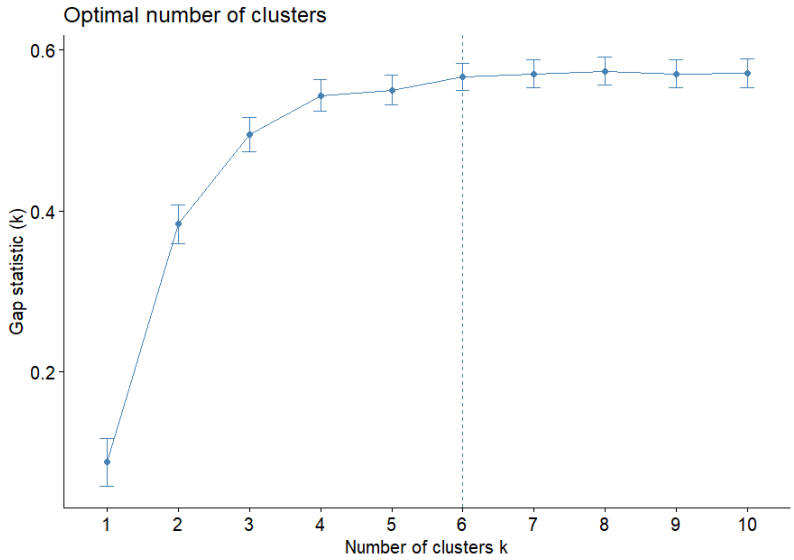
(a) Apply K-Means clustering with K = 1, 2, . . . , 10 via eclust(), for nstarts = 50 each. Provide the code and the plot of total within-cluster sum of squares (WSS) progression for the resulting clustering solutions.

```
> data("iris")
> df_iris <- iris[,c(1:4)]
> kclust <- sapply(1:10, function(k) {
+   km.res <- eclust(df_iris, FUNcluster = "kmeans", k = k, nstar
t = 50)
+   km.res$tot.withinss
+ })
>
> elbow <- fviz_nbclust(df_iris, FUNcluster = kmeans, method = "w
ss")
> elbow
```



(b) Run the gap statistic calculations for K = 1, . . . , 10, with 50 replicates each. Provide the gap statistic plot. Which K value appears to be optimal?

```
> gap <- fviz_nbclust(df_iris, FUNcluster = kmeans, nstart = 50,
method = "gap_stat")
> gap
```



K = 6 appears to be optimal.

(c) Using external cluster validation, and comparing results of your clustering to the actual # of distinct iris species in the data (≡ 3), is the answer in part (b) close to it?

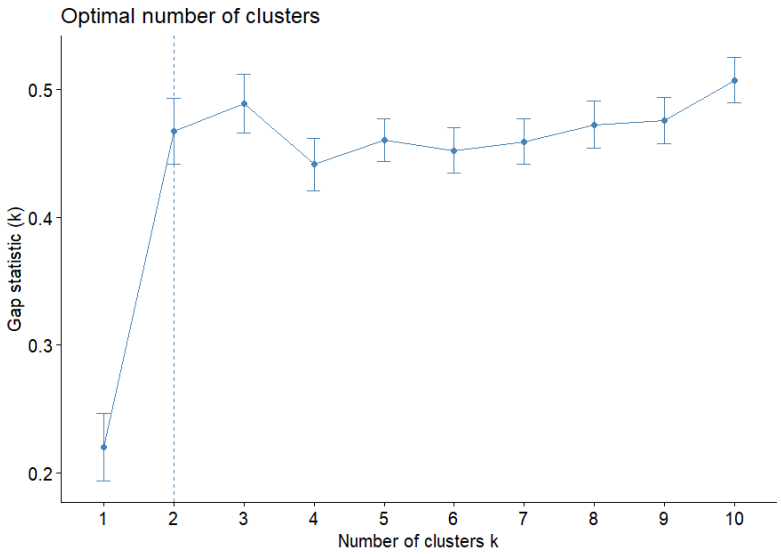
```
> table(iris$Species)

    setosa versicolor  virginica
        50         50         50
```

No, it is not close to my answer in part b because part b got 6 clusters while Species only have 3 clusters.

(d) Proceed to scale iris data, and run the gap statistic calculations for K = 1, . . . , 10, with 50 replicates each. Which K value is optimal?

```
> scaled_df_iris <- scale(df_iris)
> scaled_gap <- fviz_nbclust(scaled_df_iris, FUNcluster = kmeans,
nstart = 50, method = "gap_stat")
> scaled_gap
```



K value of 2 is optimal.

(e) Using external cluster validation, and comparing results of your clustering to the actual # of distinct iris species in the data (≡ 3), is the answer in part (d) close to it?

```
> table(iris$Species)

    setosa versicolor  virginica
        50         50         50
```

Yes, I can say that it is close that I got 2 clusters from part d and is close to 3.

(f) Apply K-means with optimal K selected in part (d). Compare the resulting cluster assignments to the actual species labels. Is there some correspondence? E.g. do any of your clusters contain only elements of a certain species? Or maybe a combination of two species?

```
> km.res <- eclust(scaled_df_iris, FUNcluster = "kmeans", k = 2,
nstart = 50)
K-means clustering with 2 clusters of sizes 50, 100

Cluster means:
```

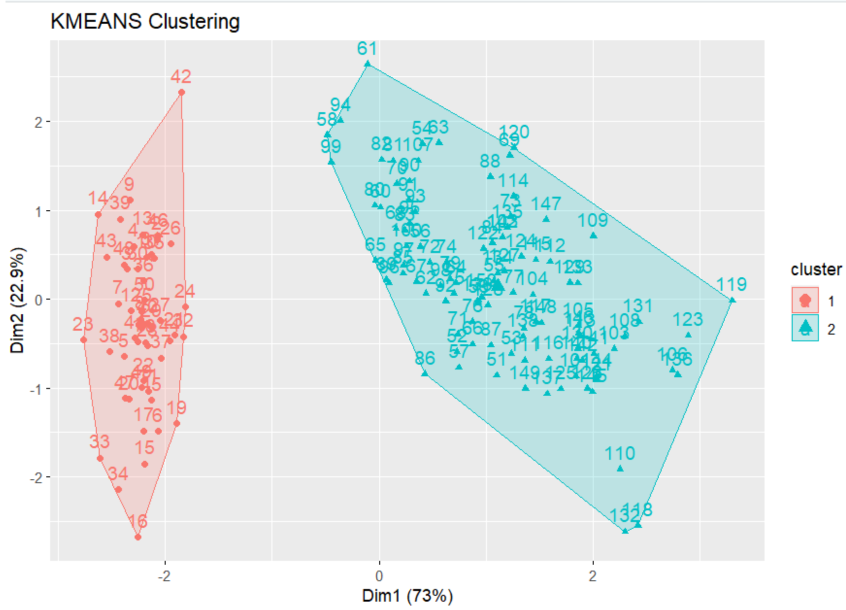
```
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1   -1.0111914   0.8504137    -1.300630  -1.2507035
2    0.5055957  -0.4252069     0.650315   0.6253518

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [45] 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [89] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2
[133] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2

Within cluster sum of squares by cluster:
[1]  47.35062 173.52867
 (between_SS / total_SS =  62.9 %)

Available components:

 [1] "cluster"      "centers"      "totss"        "withinss"
"tot.withinss"
 [6] "betweenss"    "size"         "iter"         "ifault"
"clust_plot"
[11] "silinfo"      "nbclust"      "data"
```




```
> iris$Species
  [1] setosa     setosa     setosa     setosa     setosa     seto
sa     setosa     setosa
  [9] setosa     setosa     setosa     setosa     setosa     seto
sa     setosa     setosa
 [17] setosa     setosa     setosa     setosa     setosa     seto
sa     setosa     setosa
 [25] setosa     setosa     setosa     setosa     setosa     seto
sa     setosa     setosa
 [33] setosa     setosa     setosa     setosa     setosa     seto
sa     setosa     setosa
 [41] setosa     setosa     setosa     setosa     setosa     seto
sa     setosa     setosa
 [49] setosa     setosa     versicolor versicolor versicolor vers
icolor versicolor versicolor
 [57] versicolor versicolor versicolor versicolor versicolor vers
icolor versicolor versicolor
```

```
 [65] versicolor versicolor versicolor versicolor versicolor vers
icolor versicolor versicolor
 [73] versicolor versicolor versicolor versicolor versicolor vers
icolor versicolor versicolor
 [81] versicolor versicolor versicolor versicolor versicolor vers
icolor versicolor versicolor
 [89] versicolor versicolor versicolor versicolor versicolor vers
icolor versicolor versicolor
 [97] versicolor versicolor versicolor versicolor virginica  virg
inica  virginica  virginica
[105] virginica  virginica  virginica  virginica  virginica  virg
inica  virginica  virginica
[113] virginica  virginica  virginica  virginica  virginica  virg
inica  virginica  virginica
[121] virginica  virginica  virginica  virginica  virginica  virg
inica  virginica  virginica
[129] virginica  virginica  virginica  virginica  virginica  virg
inica  virginica  virginica
[137] virginica  virginica  virginica  virginica  virginica  virg
inica  virginica  virginica
[145] virginica  virginica  virginica  virginica  virginica  virg
inica
Levels: setosa versicolor virginica
> iris$cluster <- km.res$cluster
> table(iris$cluster)

  1   2
 50 100
> table(iris$Species)

    setosa versicolor  virginica
        50         50         50
```

Yes, there is some correspondence. My optimal K value has 2 clusters where cluster 1 has 50 elements and cluster 2 has 100 elements. The actual species has 3 clusters where 50 are in each cluster. So, we can see that my optimal K value cluster 2 has 100 elements which could combine both actual species' cluster 2 and 3 (versicolor and virginica). While setosa is same cluster in my optimal K value cluster 1.

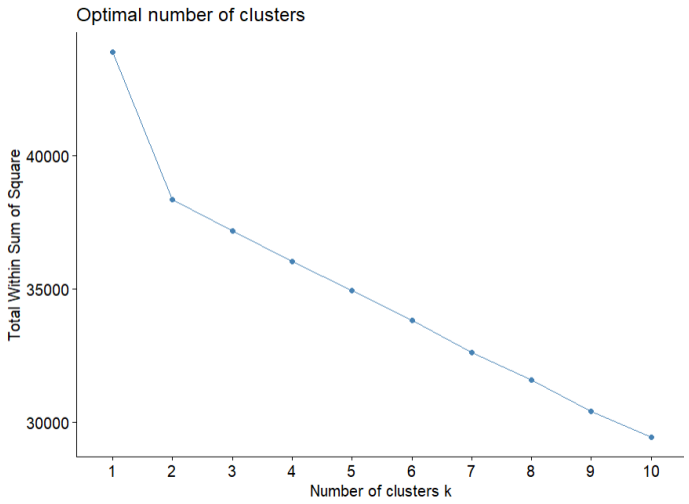
5. On the book website, www.StatLearning.com, there is a gene expression data set (Ch10Ex11.csv) that consists of 40 tissue samples with measurements on 1, 000 genes. The first 20 samples are from healthy patients, while the second 20 are from a diseased group.

(a) Load in the data using read.csv(). You will need to select header = F. Also make sure to transpose the resulting matrix (use t() operation in R), as initially, for some reason, it contains variables as rows, and observations as columns. Should be the other way around.

```
> Ch10Ex11 <- read.csv("C:/Users/chase/Downloads/Ch10Ex11.csv", h
eader=FALSE)
>   View(Ch10Ex11)
> ch10_df <- t(Ch10Ex11)
```

(b) Apply K-Means clustering with K = 1, 2, . . . , 10 via eclust(), for nstarts = 50 each. Provide the code and the plot of total within-cluster sum of squares (WSS) progression for the resulting clustering solutions. Does there appear to be an ”elbow”? If yes, at which value of K?

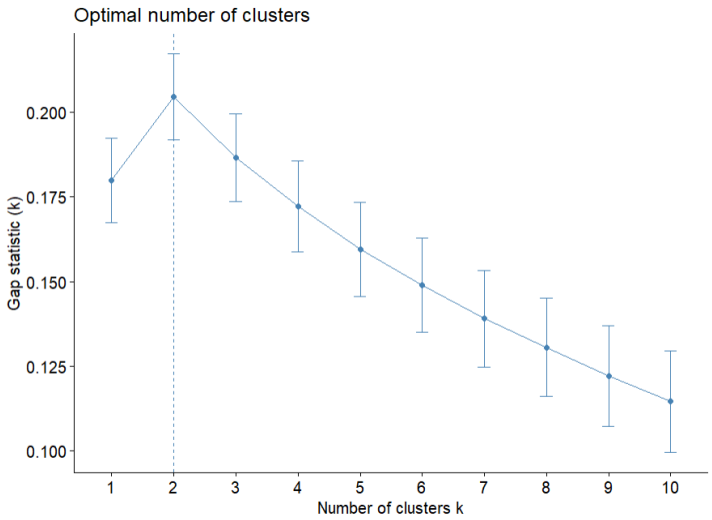
```
> kclust <- sapply(1:10, function(k) {
+   km.res <- eclust(ch10_df, FUNcluster = "kmeans", k = k, nstar
t = 50)
+   km.res$tot.withinss
+ })
>
> elbow <- fviz_nbclust(ch10_df, FUNcluster = kmeans, method = "w
ss")
> elbow
```



Yes, the plot does appear an elbow at K value of 2.

(c) Run the gap statistic calculations for K = 1, . . . , 10, with 20 replicates each (Note: it might take some time). Provide the plot. Which K value is optimal?

```
> gap <- fviz_nbclust(ch10_df, FUNcluster = kmeans, nstart = 50,
method = "gap_stat")
> gap
```



K value of 2 is optimal.

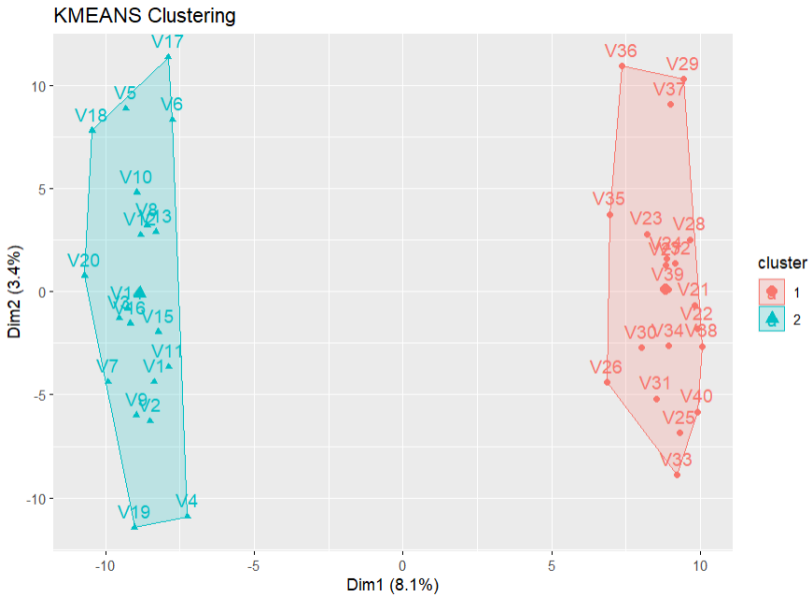
(d) Using external cluster validation, and comparing results of your clustering to the actual # of patient group (which is 2 - healthy and diseased), is the answer in part (c) close to it?

Yes, my answer in part c is close to the actual # of patient group which is 2 and my answer have 2 clusters.

(e) Apply K-means with optimal K selected in part (c). Compare the resulting cluster assignments to the actual patient groups. Is there correspondence? E.g. do any of your clusters contain only healthy (or only diseased) patients?

```
> km.res <- eclust(ch10_df, FUNcluster = "kmeans", k = 2, nstart
= 50)
> ch10_df$cluster <- km.res$cluster
> table(ch10_df$cluster)

 1  2
20 20
```



Yes, there is correspondence. My optimal K value has 2 clusters where it split up by healthy patients and diseased patients. The actual patient groups have 20 healthy patients in 1 cluster and the other 20 in cluster 2 where patients are diseased. Therefore, my plot and my table do show that I do have 20 healthy patients in cluster 1 and 20 diseased patients in cluster 2.