Exercise 16

Abhigyan Misra

October 27th 2020

Clustering

Problem Statement : These assignments are here to provide you with an introduction to the “Data Science” use for these tools. This is your future. It may seem confusing and weird right now but it hopefully seems far less so than earlier in the semester. Attempt these homework assignments. You will not be graded on your answer but on your approach. This should be a, “Where am I on learning this stuff” check. If you can’t get it done, please explain why.

Remember to submit this assignment in an R Markdown report.

Labeled data is not always available. For these types of datasets, you can use unsupervised algorithms to extract structure. The k-means clustering algorithm and the k nearest neighbor algorithm both use the Euclidean distance between points to group data points. The difference is the k-means clustering algorithm does not use labeled data.

In this problem, you will use the k-means clustering algorithm to look for patterns in an unlabeled dataset. The dataset for this problem is found at data/clustering-data.csv.

Solutions

Reading the Data

```
## Set the working directory to the root of your DSC 520 directory
setwd("C:/git-bellevue/dsc520-fork")

## Load the ‘caTools‘ library
library(caTools)
```

```
## Warning: package ’caTools’ was built under R version 4.0.3
```

```
## Load the ‘data/clustering-data.csv‘ to
clustering_df <- read.csv("data/clustering-data.csv")
head(clustering_df)
```

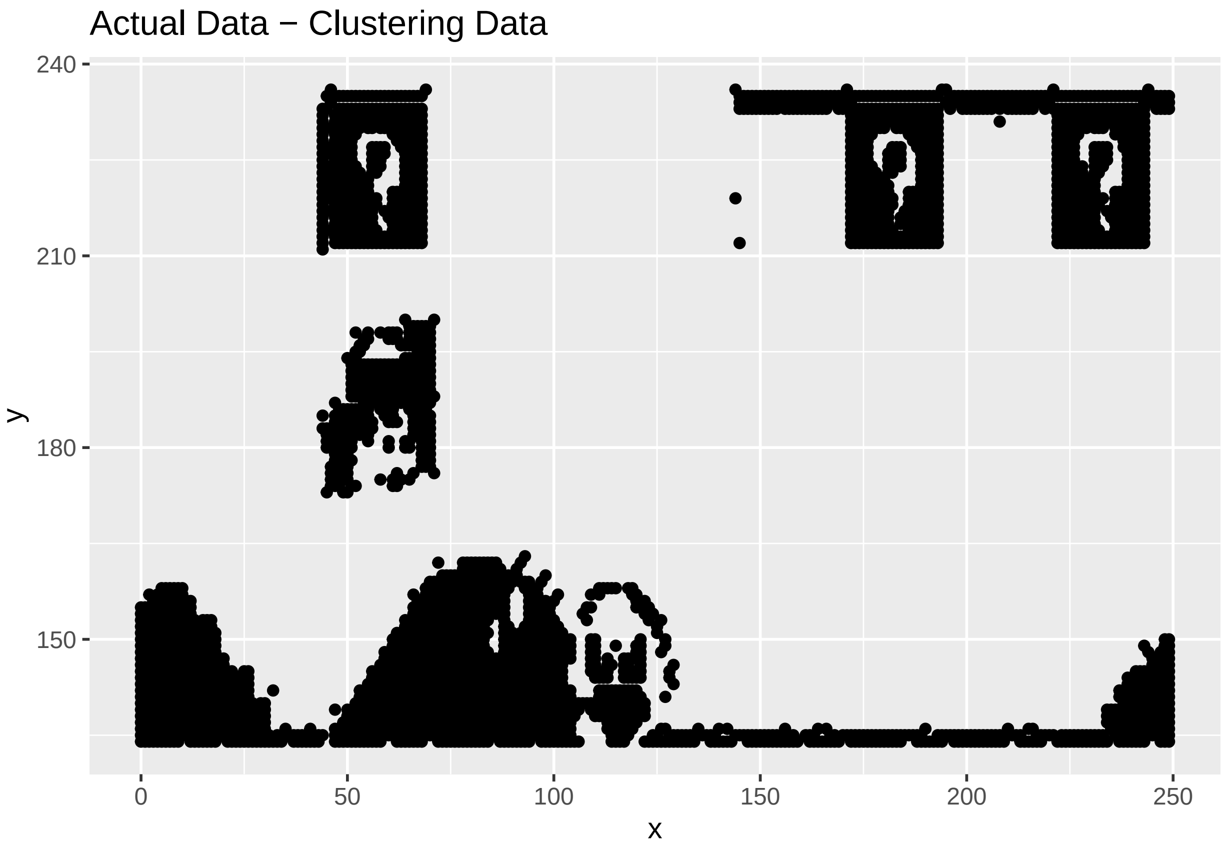
```
##     x   y
## 1  46 236
## 2  69 236
## 3 144 236
## 4 171 236
## 5 194 236
## 6 195 236
```

```
summary(clustering_df)
```

```
##        x               y        
##  Min.   :  0.0   Min.   :134.0  
##  1st Qu.: 56.0   1st Qu.:141.0  
##  Median : 82.0   Median :154.0  
##  Mean   :109.6   Mean   :175.7  
##  3rd Qu.:180.0   3rd Qu.:218.0  
##  Max.   :249.0   Max.   :236.0  
```

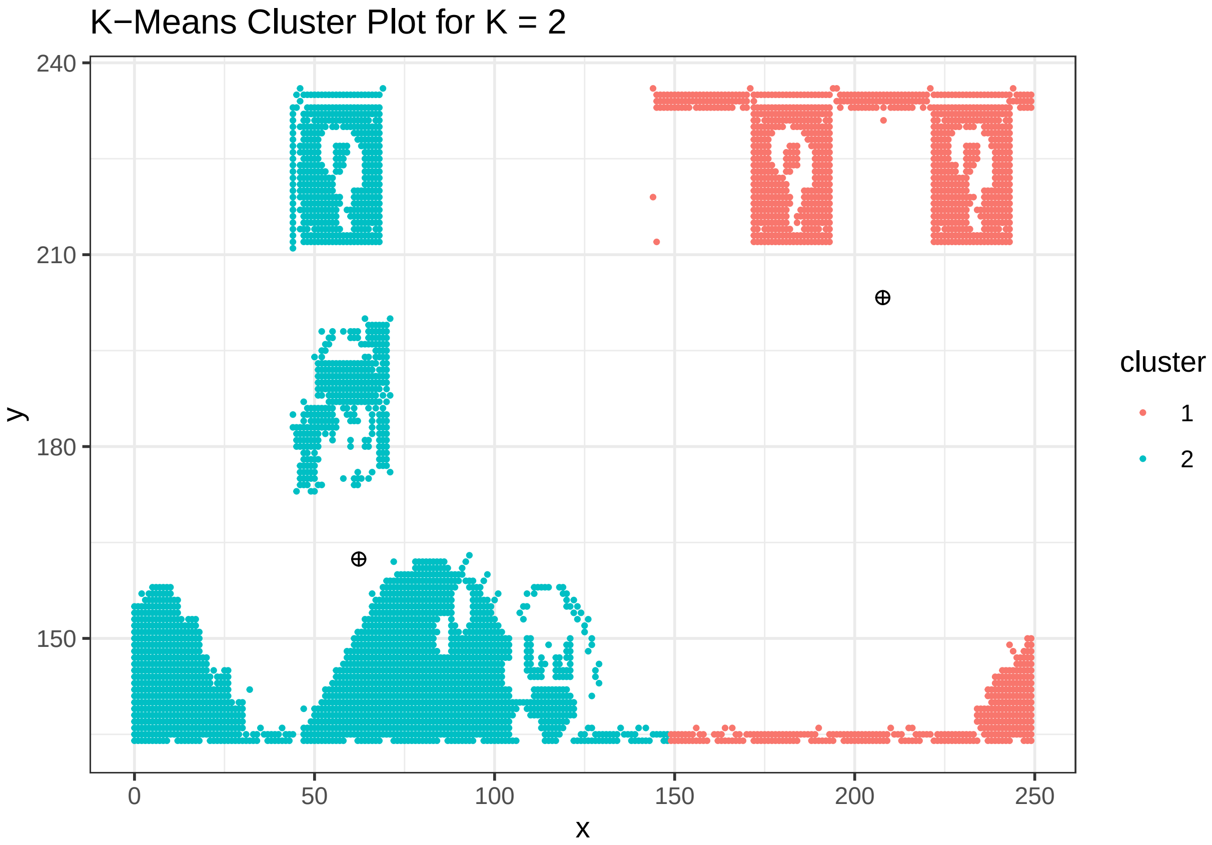
a. Plot the dataset using a scatter plot.

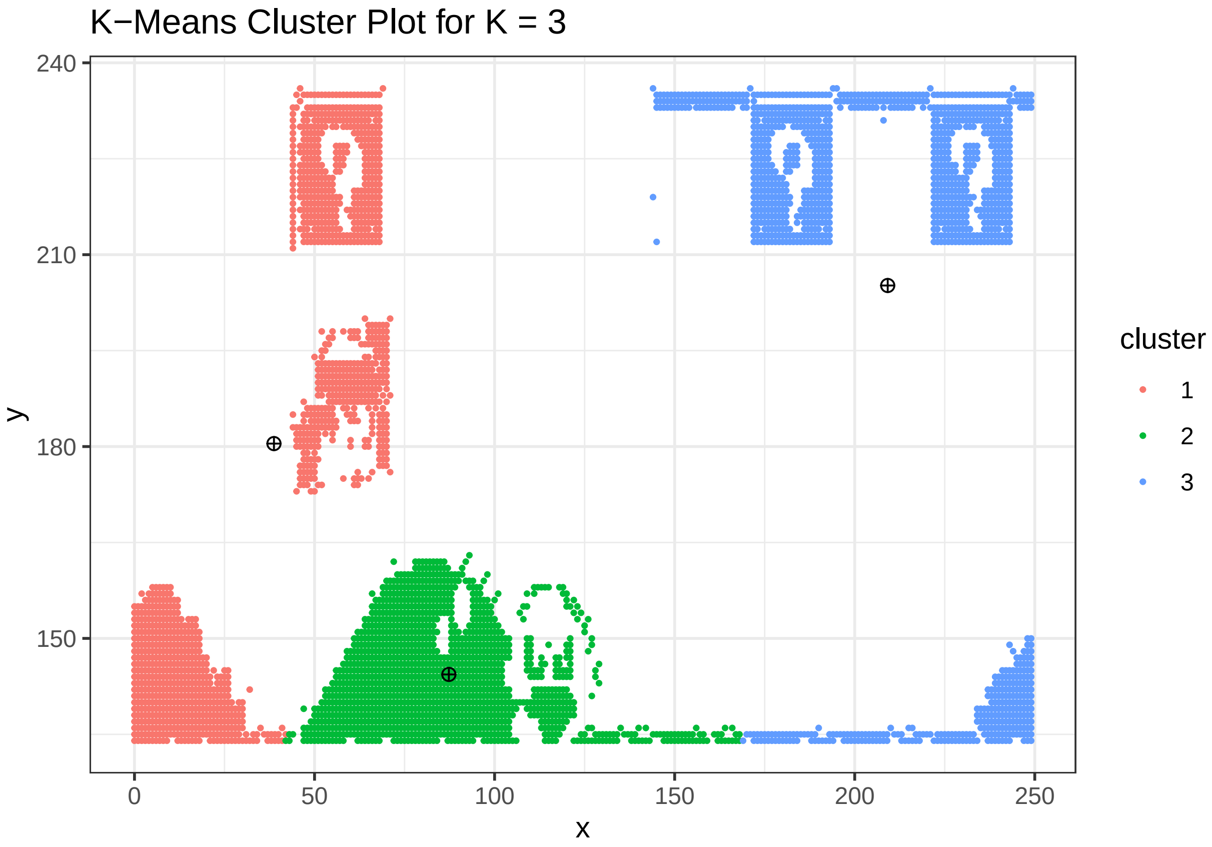
```
library(ggplot2)
ggplot(data = clustering_df, aes(y = y, x = x)) +
  geom_point() + ggtitle("Actual Data - Clustering Data")
```

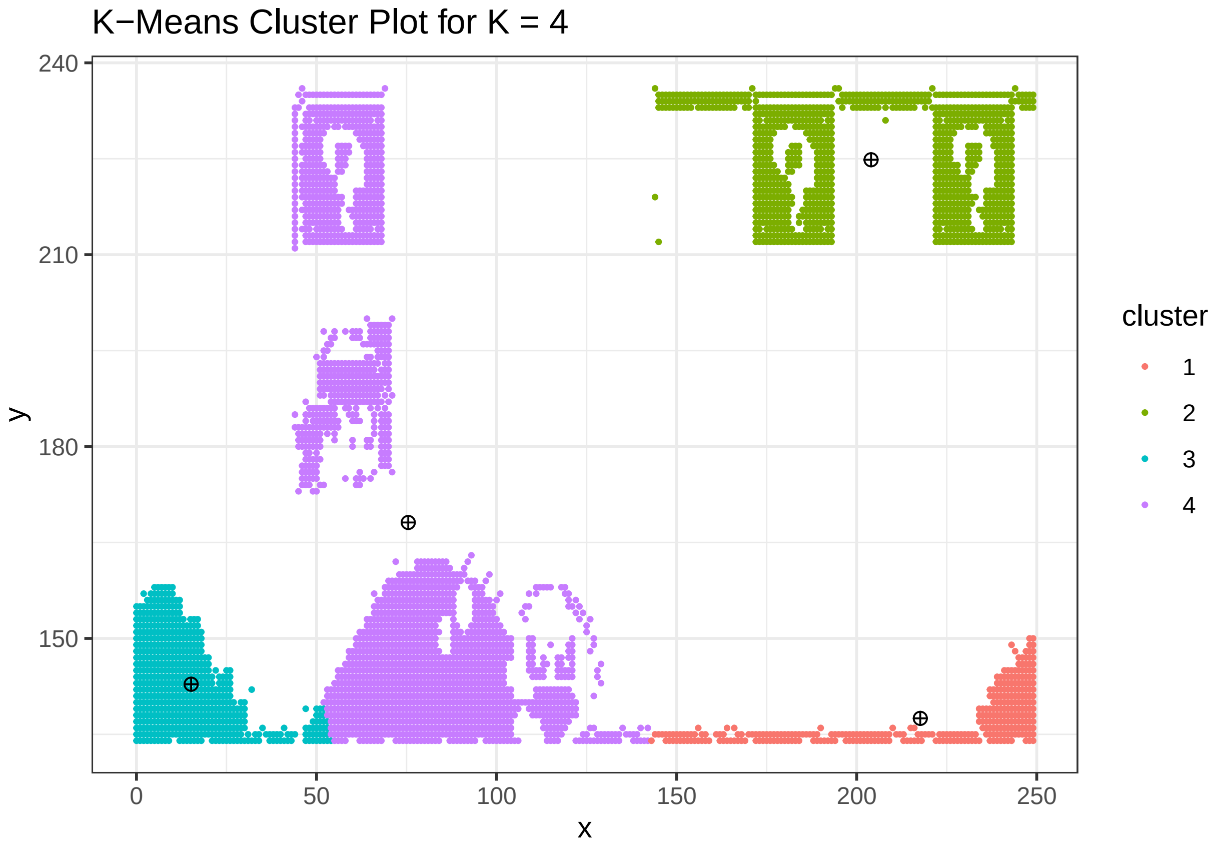


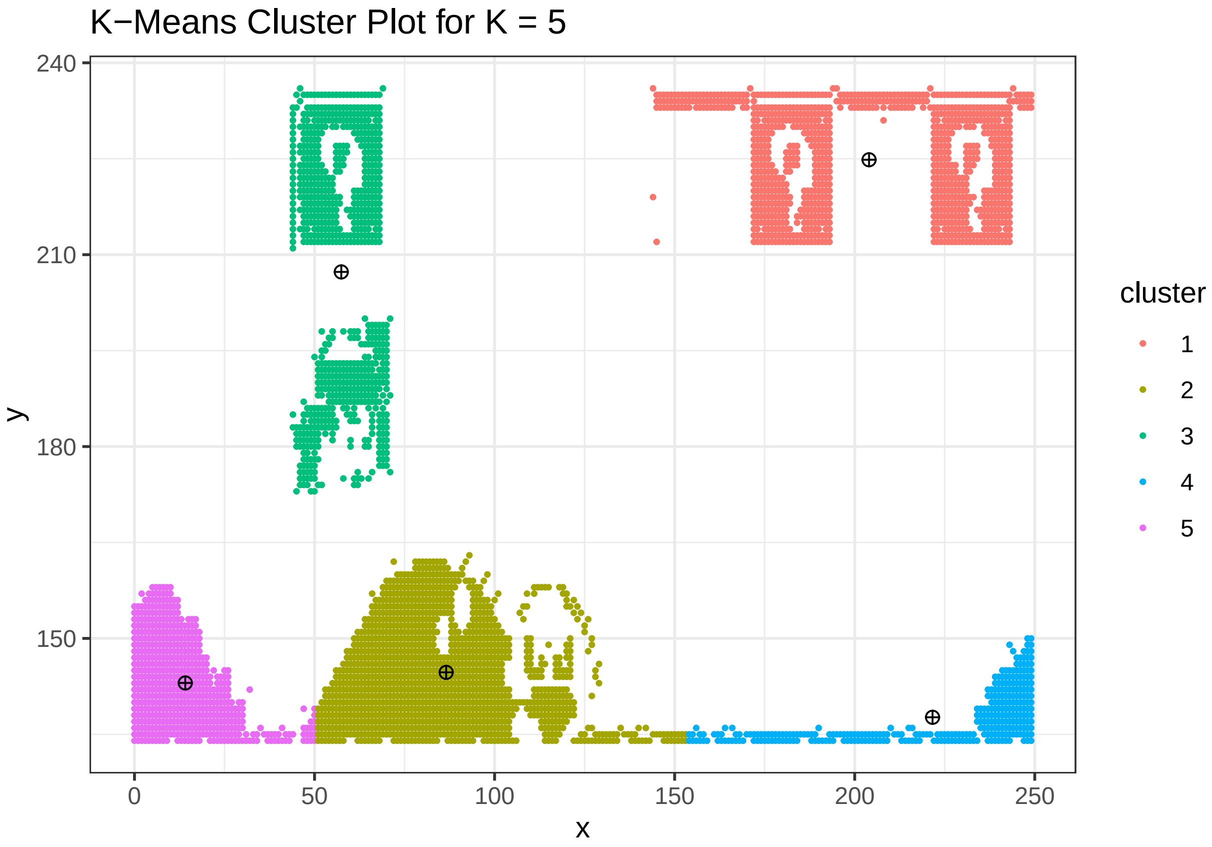
b. Fit the dataset using the k-means algorithm from k=2 to k=12. Create a scatter plot of the resultant clusters for each value of k.

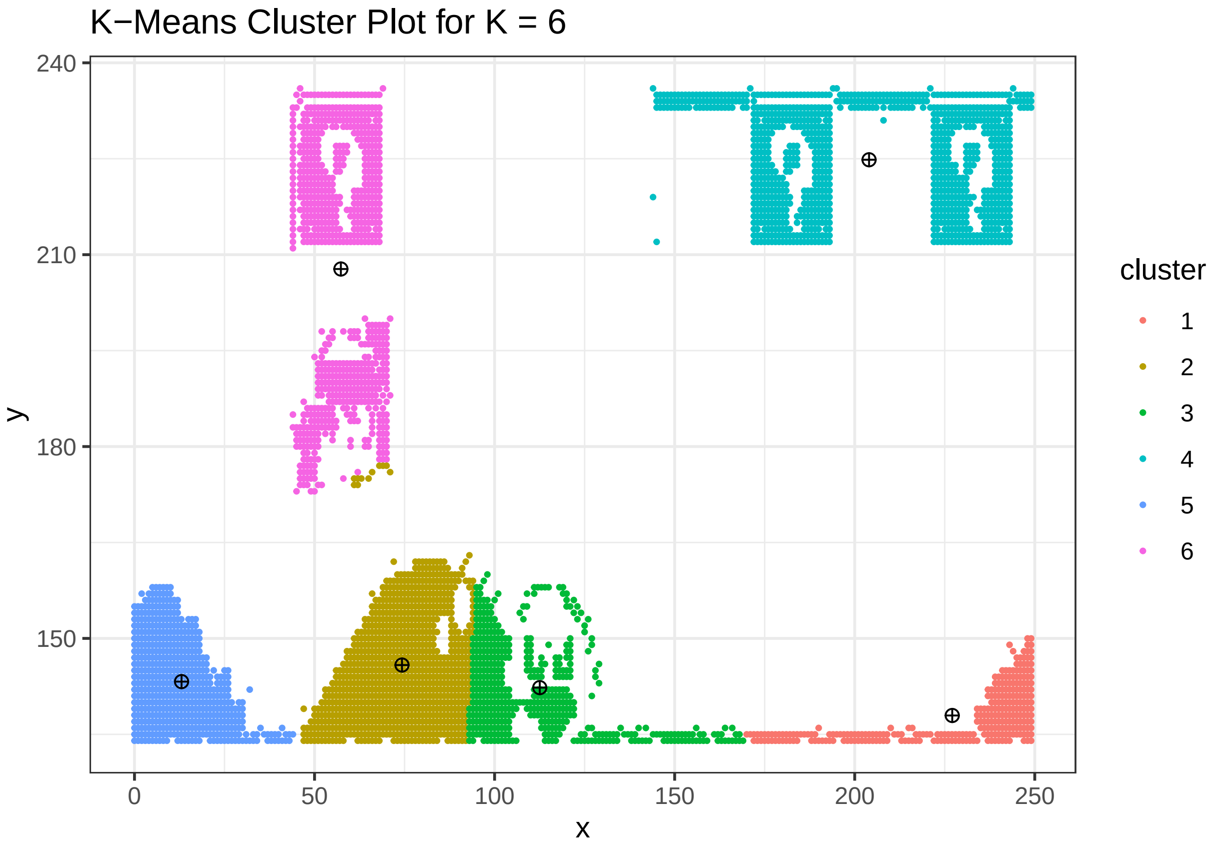
```r
set.seed(100)

k_values <- c()
tot.withinss_values <- c()
errors <- c()

for(i in 2:12){
# Read Once and Mapping the same object Multiple Times
df <- clustering_df
df.cluster <- kmeans(df, i)


df$cluster <- as.factor(df.cluster$cluster)
p <- ggplot(data = df,
            aes(x = x,
                y = y,
                color = cluster)) +
geom_point(size = 0.5) +
geom_point(data = as.data.frame(df.cluster$centers),
           color = "black",
           shape = 10,
           size = 2) +
ggtitle(paste("K-Means Cluster Plot for K = ", i, sep ="")) +
theme_bw()
print(p)

# For Later Analysis & Plot
k_values<- c(k_values, i)
tot.withinss_values <- c(tot.withinss_values, df.cluster$tot.withinss)

x.dist <- df.cluster$centers[df$cluster] - df$x
y.dist <- df.cluster$centers[as.numeric(df$cluster) + i] - df$y
tot.dist <- sqrt((x.dist ** 2) + (y.dist ** 2))
errors <- c(errors, mean(tot.dist))
}
```

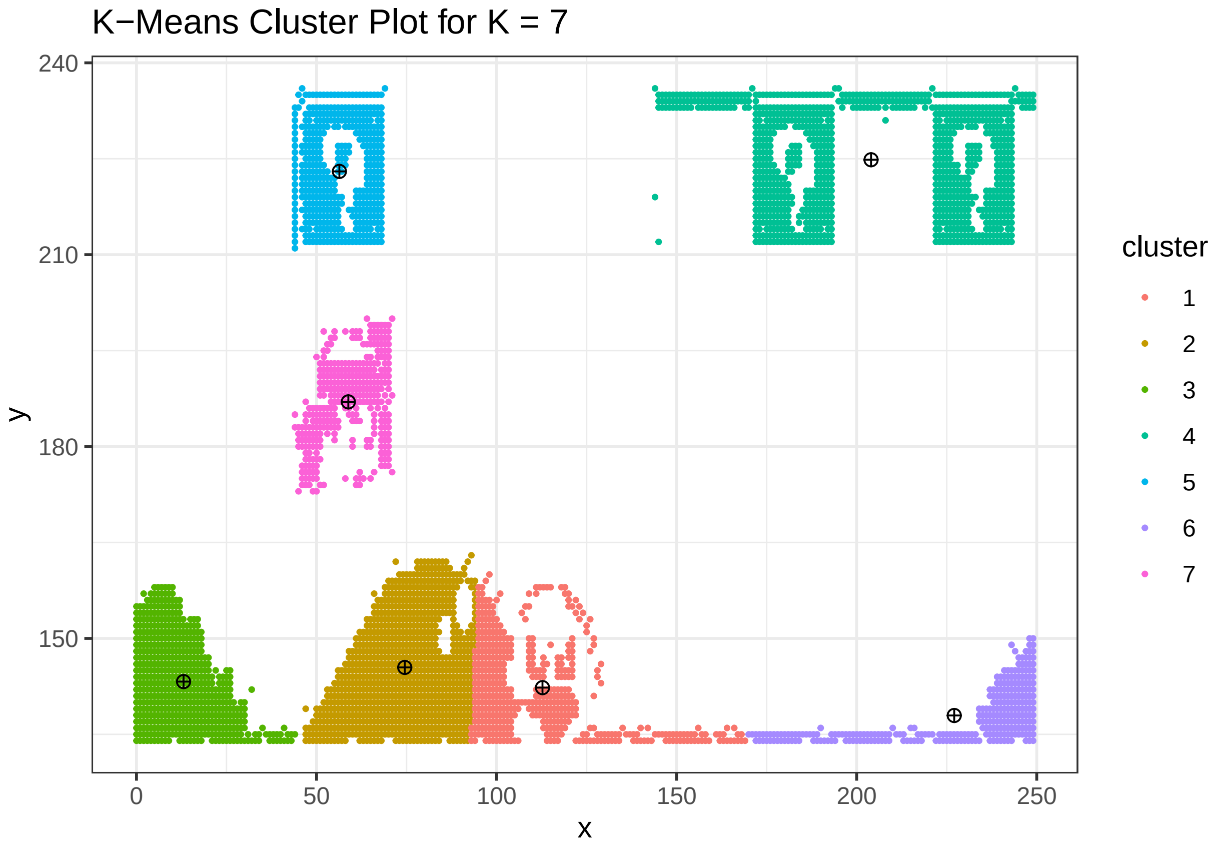

K−Means Cluster Plot for K = 2
y
240
210
180
150
x
0
50
100
150
200
250
cluster
1
2

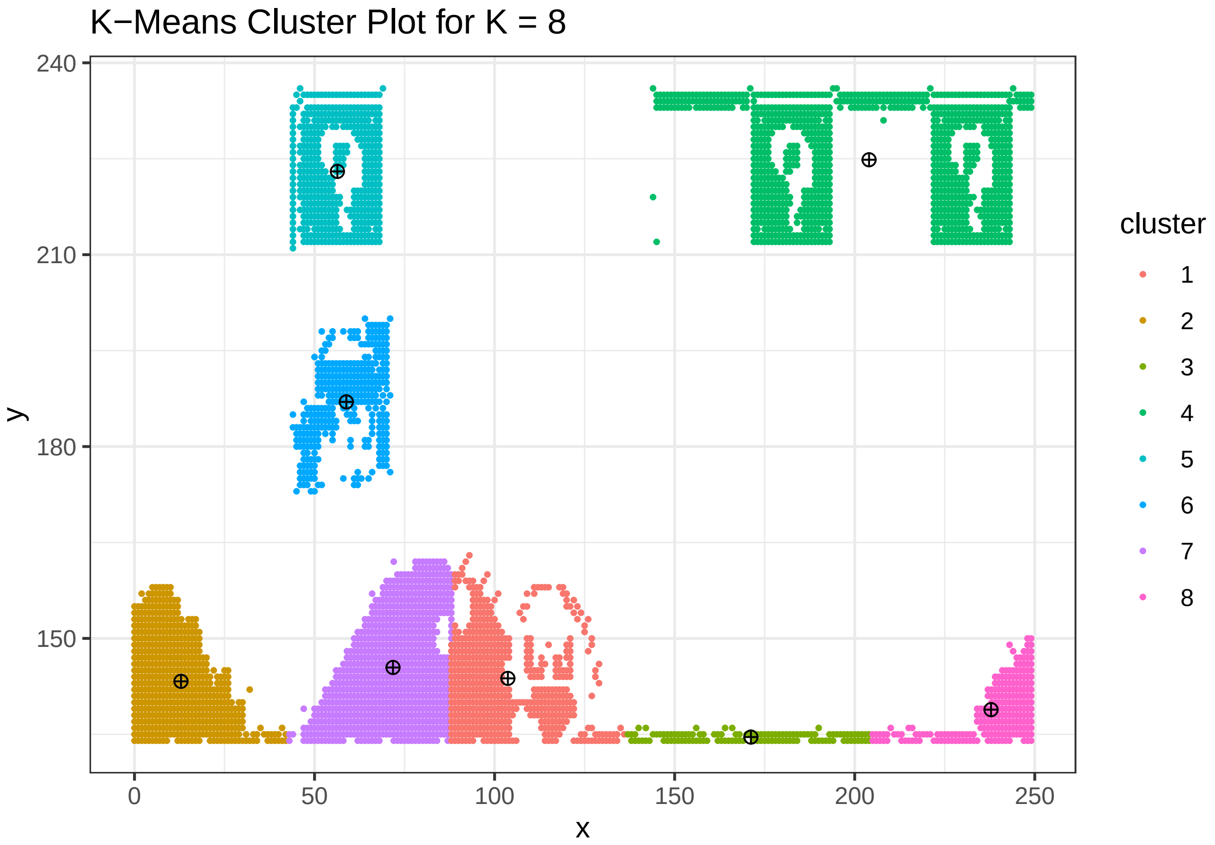

K−Means Cluster Plot for K = 3
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3

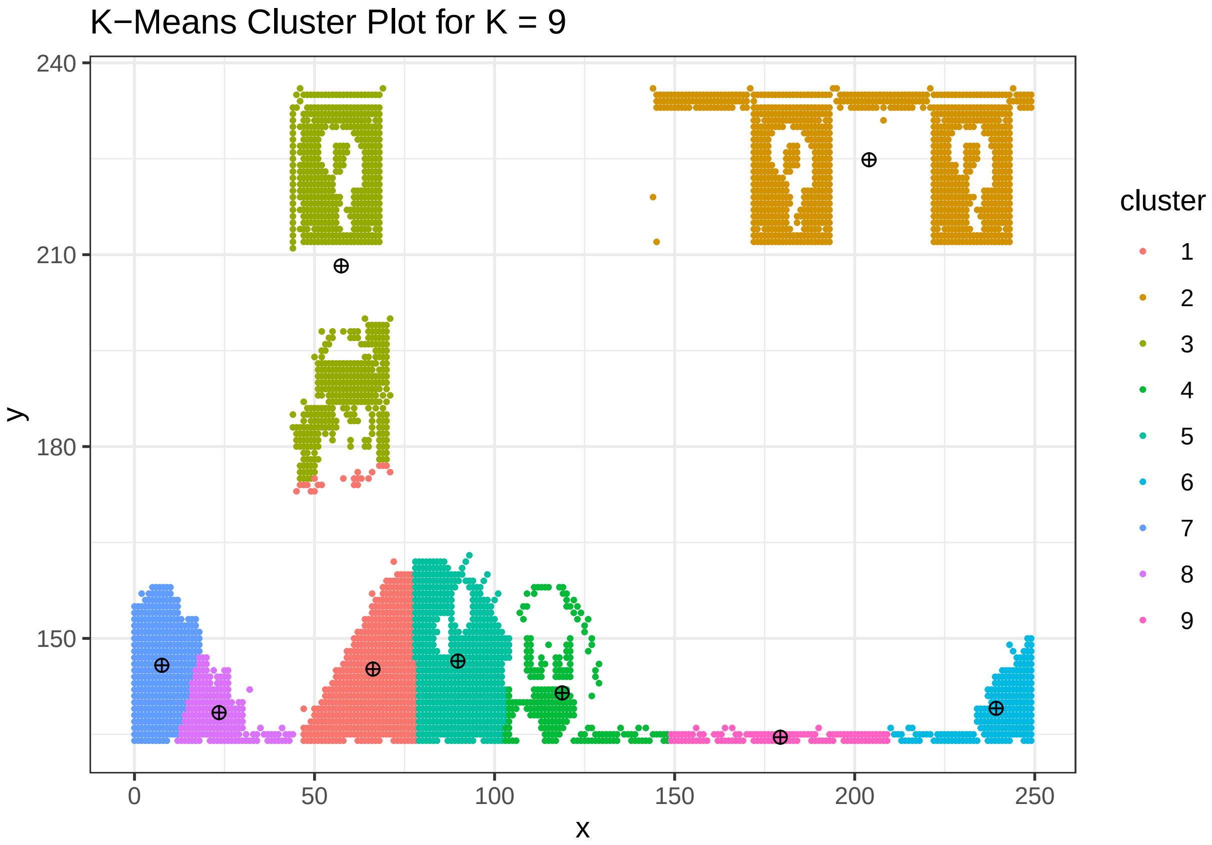

K−Means Cluster Plot for K = 4
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4

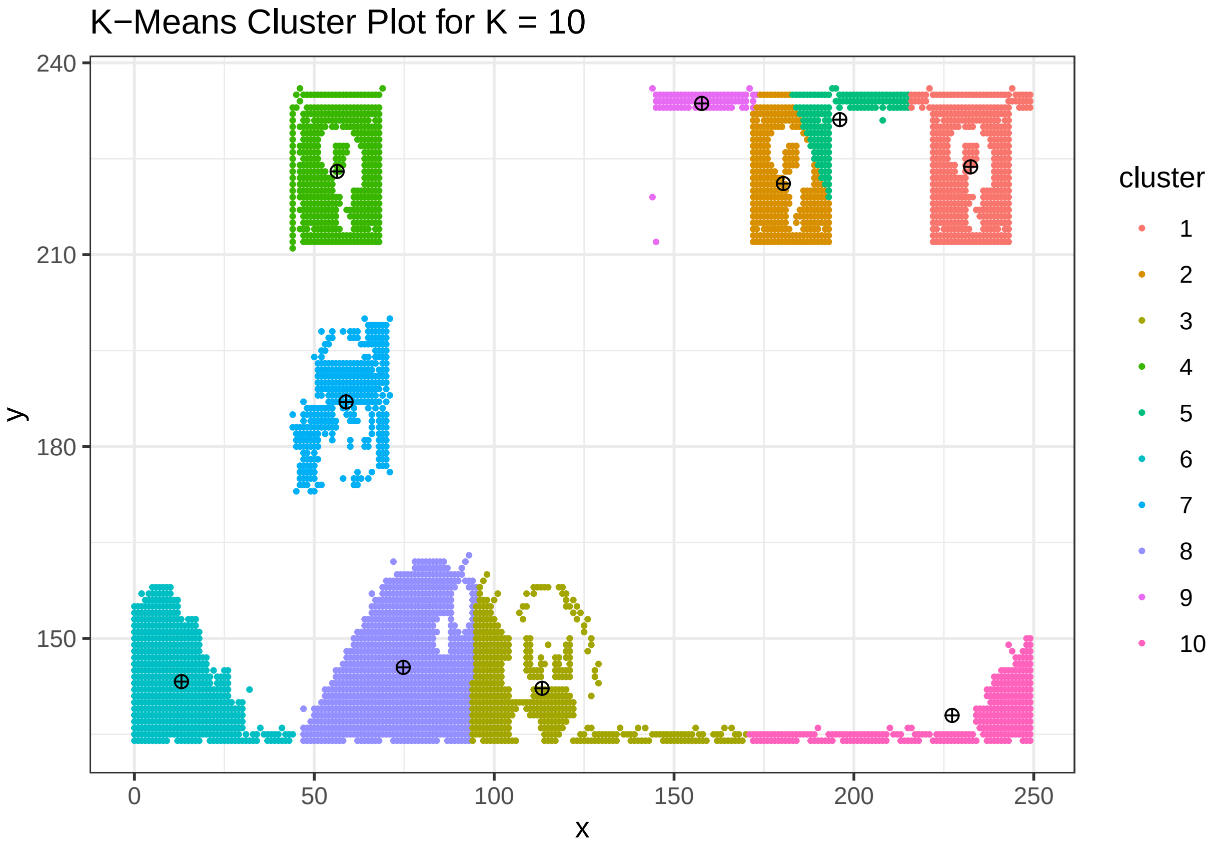

K−Means Cluster Plot for K = 5
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4
5

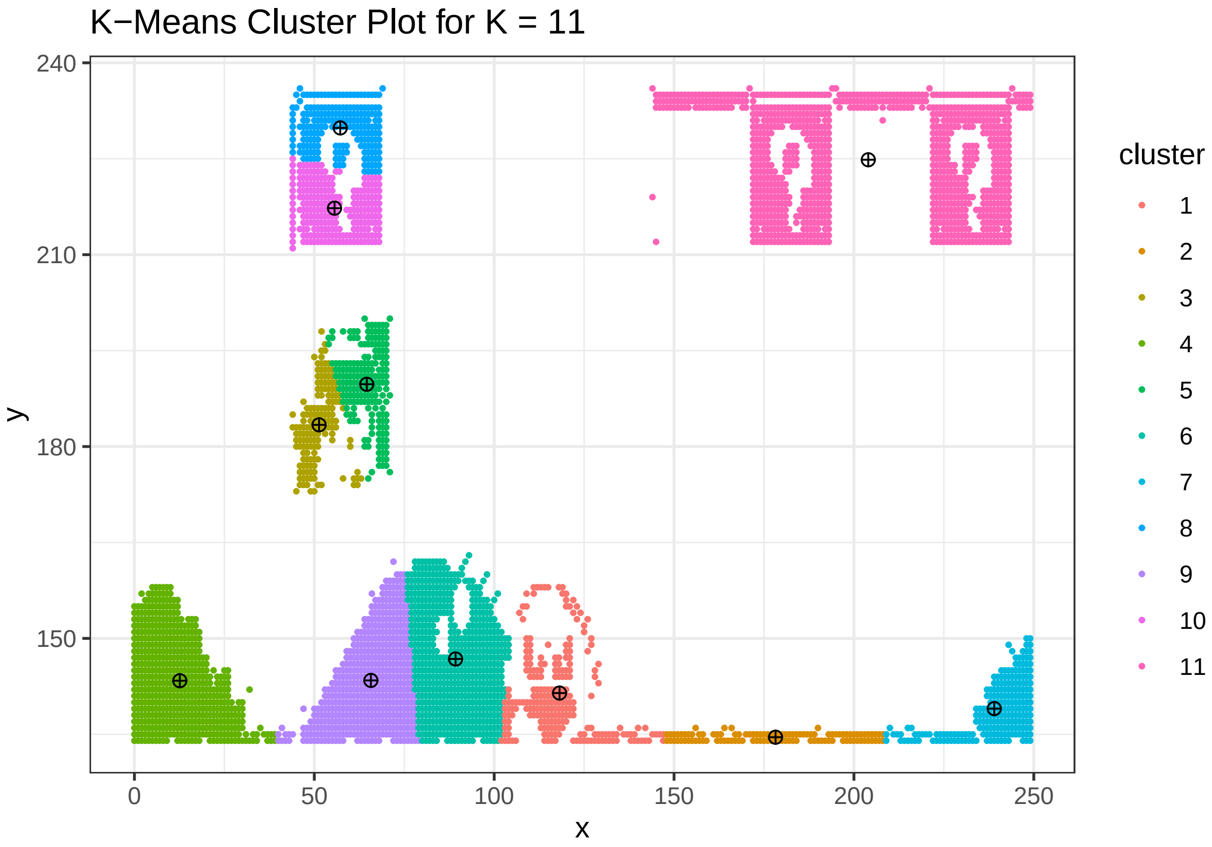

K−Means Cluster Plot for K = 6
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4
5
6

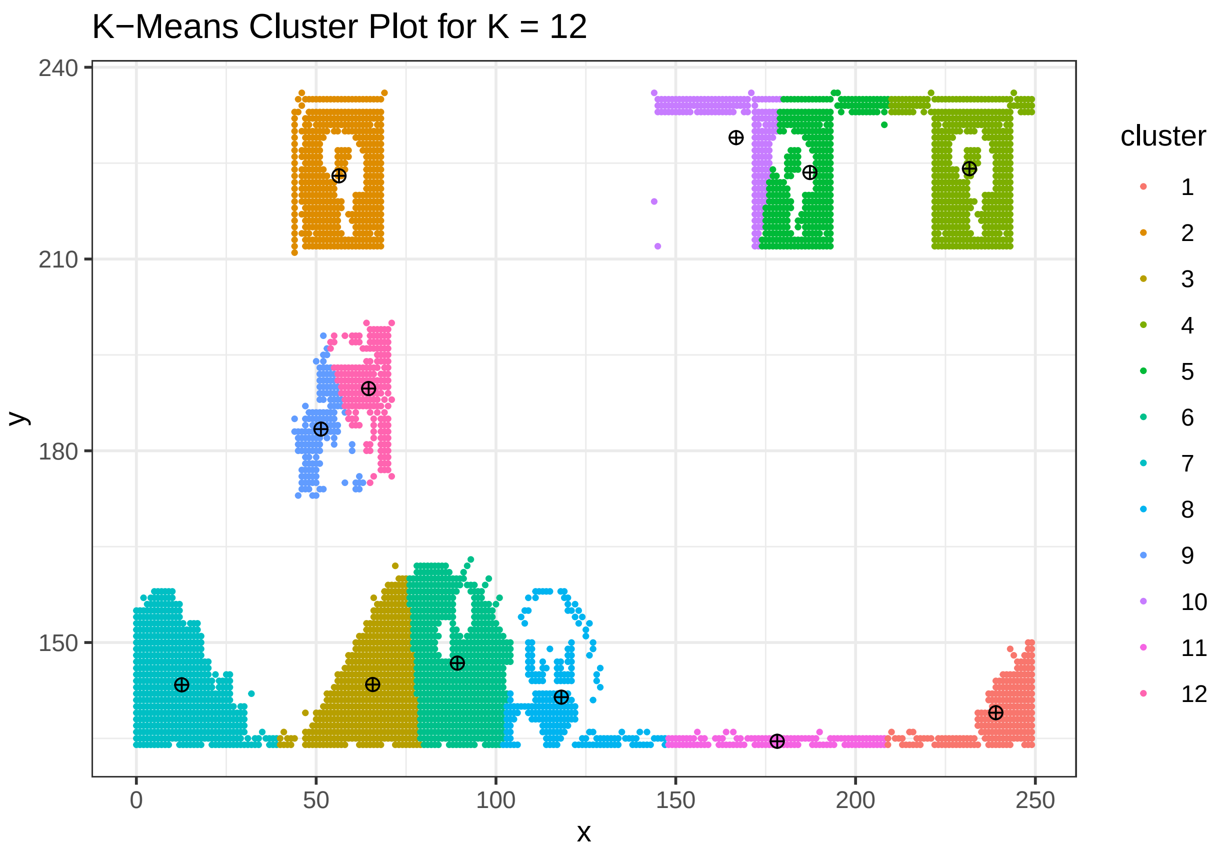

K−Means Cluster Plot for K = 7
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4
5
6
7


K−Means Cluster Plot for K = 8
240
210
180
150
y
0
50
100
150
200
250
x
cluster
● 1
● 2
● 3
● 4
● 5
● 6
● 7
● 8


K−Means Cluster Plot for K = 9
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4
5
6
7
8
9


K−Means Cluster Plot for K = 10
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4
5
6
7
8
9
10


K−Means Cluster Plot for K = 11
240
210
180
150
y
0
50
100
150
200
250
x
cluster
1
2
3
4
5
6
7
8
9
10
11



```
elbow_df <- data.frame(k_values, tot.withinss_values, errors)
```

c. As k-means is an unsupervised algorithm, you cannot compute the accuracy as there are no correct values to compare the output to. Instead, you will use the average distance from the center of each cluster as a measure of how well the model fits the data. To calculate this metric, simply compute the distance of each data point to the center of the cluster it is assigned to and take the average value of all of those distances.

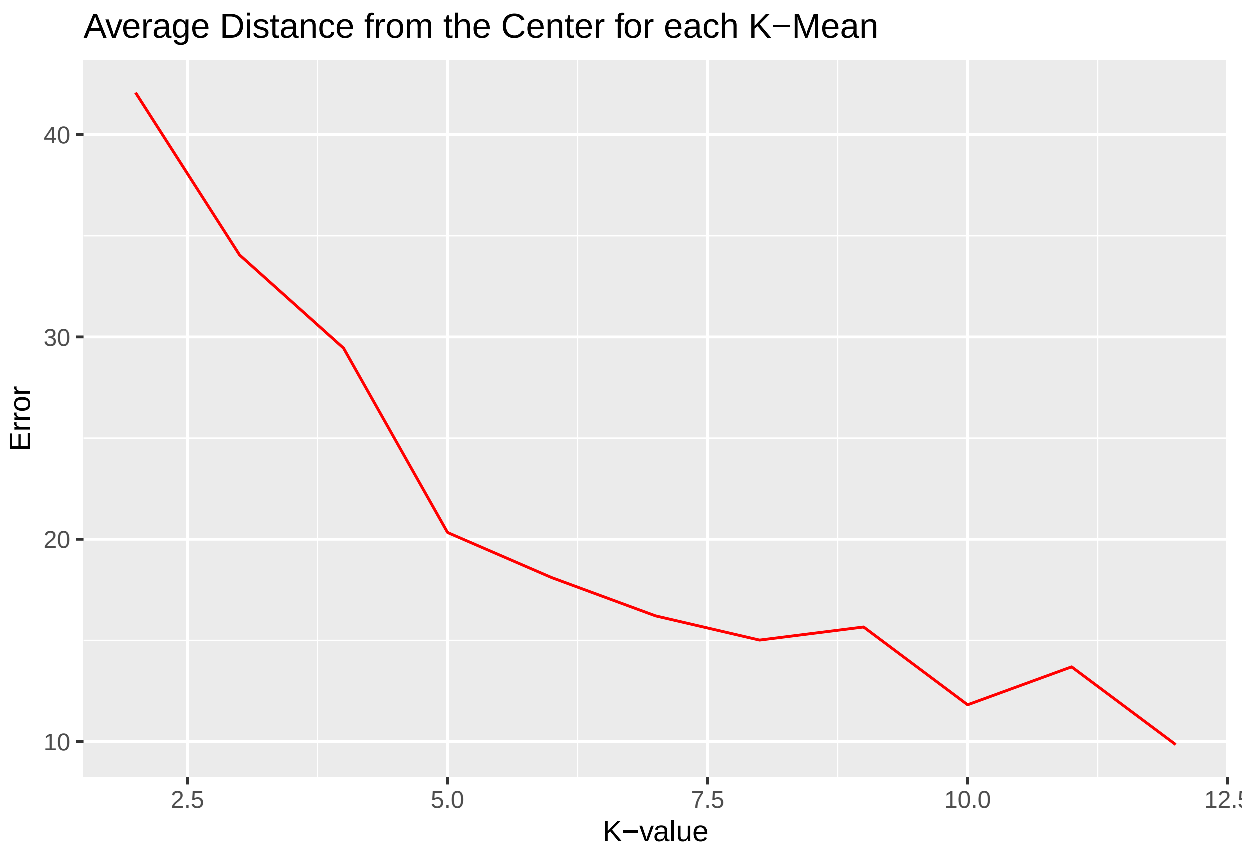
Calculate this average distance from the center of each cluster for each value of k and plot it as a line chart where k is the x-axis and the average distance is the y-axis.

One way of determining the "right" number of clusters is to look at the graph of k versus average distance and finding the "elbow point". Looking at the graph you generated in the previous example, what is the elbow point for this dataset?

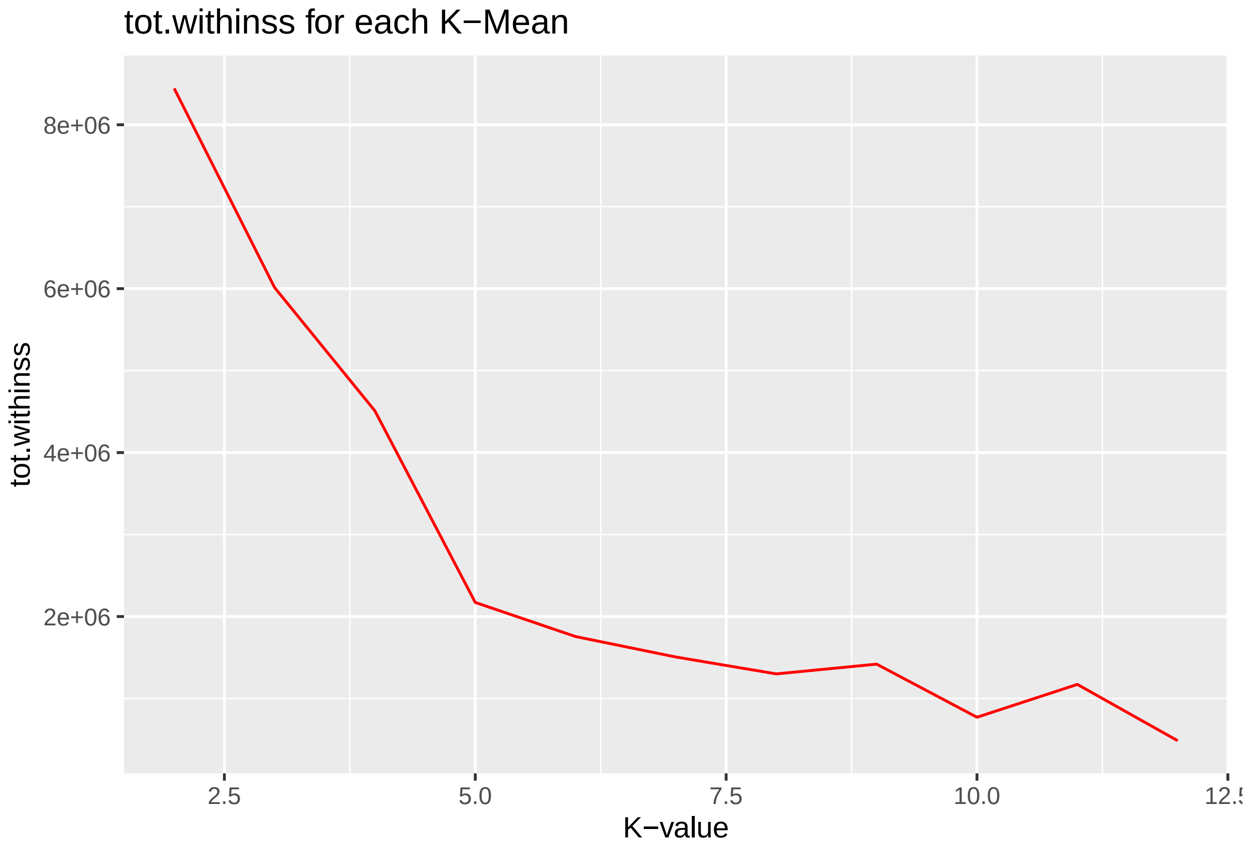
```r
elbow_df
```

```
##    k_values tot.withinss_values   errors
## 1         2           8443681.1 42.07717
## 2         3           6014377.9 34.05374
## 3         4           4509358.0 29.44537
## 4         5           2171612.8 20.33178
## 5         6           1755439.4 18.10784
## 6         7           1505729.6 16.21206
## 7         8           1299901.3 15.01513
## 8         9           1418607.7 15.66172
## 9        10            770764.5 11.81694
## 10       11           1171689.3 13.69285
## 11       12            484239.4  9.85157
```

```r
ggplot(data = elbow_df, aes(x = k_values, y = errors)) +
  geom_line(color = "red") +
  ggtitle("Average Distance from the Center for each K-Mean") +
  xlab("K-value") + ylab("Error") + xlim(2,12)
```



```r
# Also Plotting tot.withinss, as mentioned in the referenced article
ggplot(data = elbow_df, aes(x = k_values, y = tot.withinss_values)) +
  geom_line(color = "red") +
  ggtitle("tot.withinss for each K-Mean") +
  xlab("K-value") + ylab("tot.withinss") + xlim(2,12)
```



I am seeing Elbows at K = 5, 9, 10, 11. If we look at the K-Means Cluster Plots earlier plotted and focus on these values , we may say that its difficult to suggest which one is optimal. The reason being, the plot itself is an image of something.

However regular data might reflect things more clearly and by looking first at the elbow either for errors or totwithinss plots, we may be able to filter out specific K values which will be optimal. And then make the final decision by looking at the K-Mean Cluster plots. This analysis is based on the article referenced from Medium.

References

https://www.datanovia.com/en/lessons/k-means-clustering-in-r-algorith-and-practical-examples/ https://www.datacamp.com/community/tutorials/k-means-clustering-r https://blogs.oracle.com/datascience/introduction-to-k-means-clustering https://medium.com/analytics-vidhya/how-to-determine-the-optimal-k-for-k-means-708505d204eb