11.2 Exercise

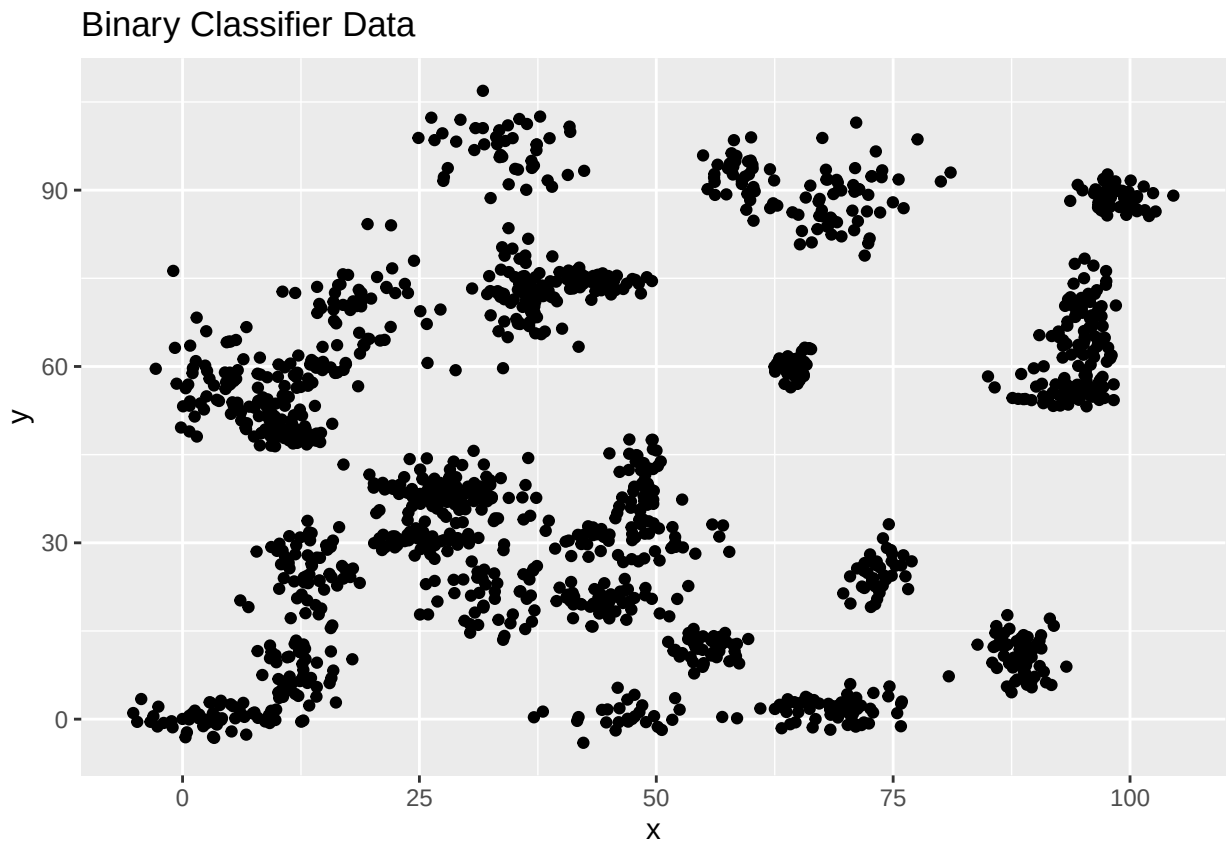
Harlan Wittlieff

11/20/2021

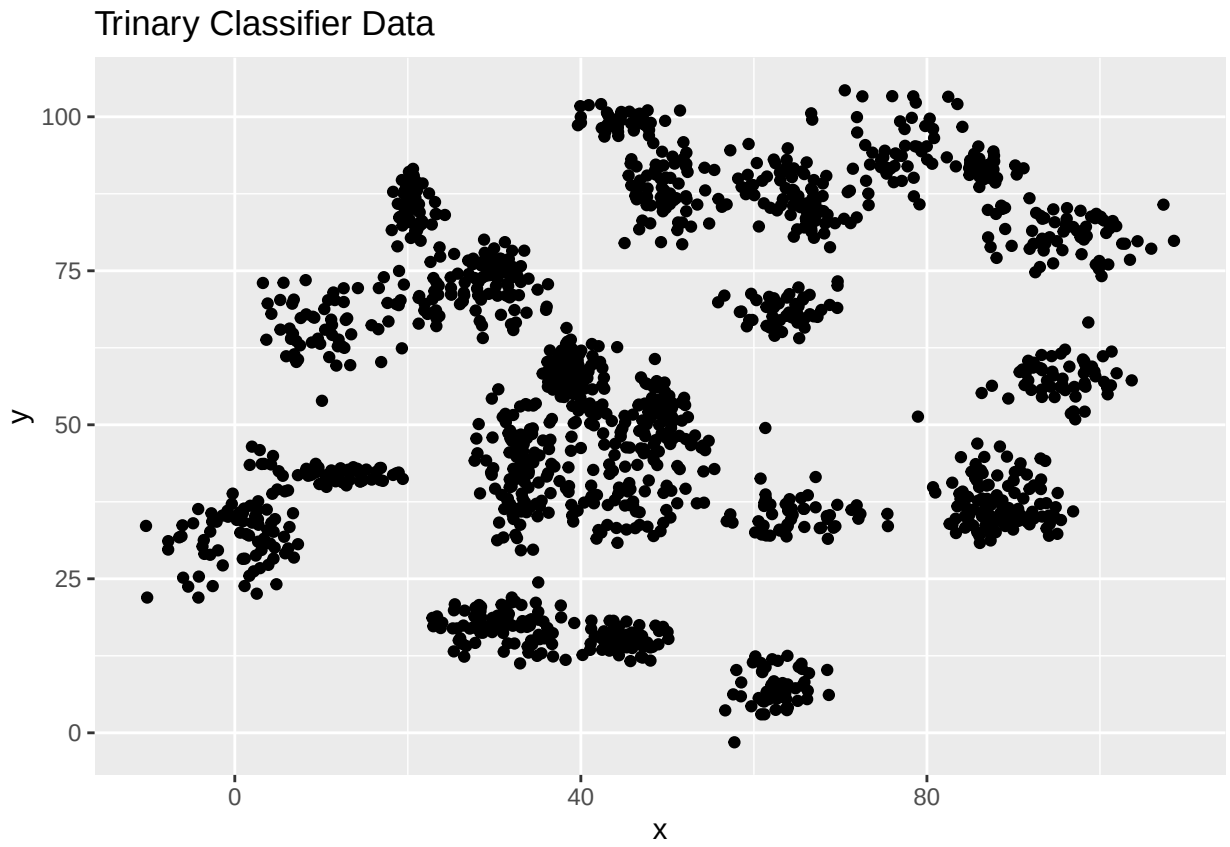
1. Introduction to Machine Learning

i. Plot the data from each dataset using a scatter plot.

```
library('ggplot2')
ggplot(binary_df, aes(x=x, y=y))+ggtitle("Binary Classifier Data")+geom_point()
```




```
ggplot(trinary_df, aes(x=x, y=y))+ggtitle("Trinary Classifier Data") + geom_point()
```



ii. Fit a k nearest neighbors model for each dataset for k=3, k=5, k=10, k=15, k=20, and k=25.

```
# Create training & testing datasets
binary_ran <- sample(1:nrow(binary_df), 0.9 * nrow(binary_df))
binary_train <- binary_df[binary_ran,]
binary_test <- binary_df[-binary_ran,]
binary_target_category <- binary_df[binary_ran,1]
binary_test_category <- binary_df[-binary_ran,1]


trinary_ran <- sample(1:nrow(trinary_df), 0.9*nrow(trinary_df))
trinary_train <- trinary_df[trinary_ran,]
trinary_test <- trinary_df[-trinary_ran,]
trinary_target_category <- trinary_df[trinary_ran,1]
trinary_test_category <- trinary_df[-trinary_ran,1]

# Load the class package
library(class)

# Create the K Nearest Neighbors models
b3 <- knn(binary_train, binary_test, cl=binary_target_category, k=3)
b5 <- knn(binary_train, binary_test, cl=binary_target_category, k=5)
b10 <- knn(binary_train, binary_test, cl=binary_target_category, k=10)
b15 <- knn(binary_train, binary_test, cl=binary_target_category, k=15)
b20 <- knn(binary_train, binary_test, cl=binary_target_category, k=20)
b25 <- knn(binary_train, binary_test, cl=binary_target_category, k=25)
```

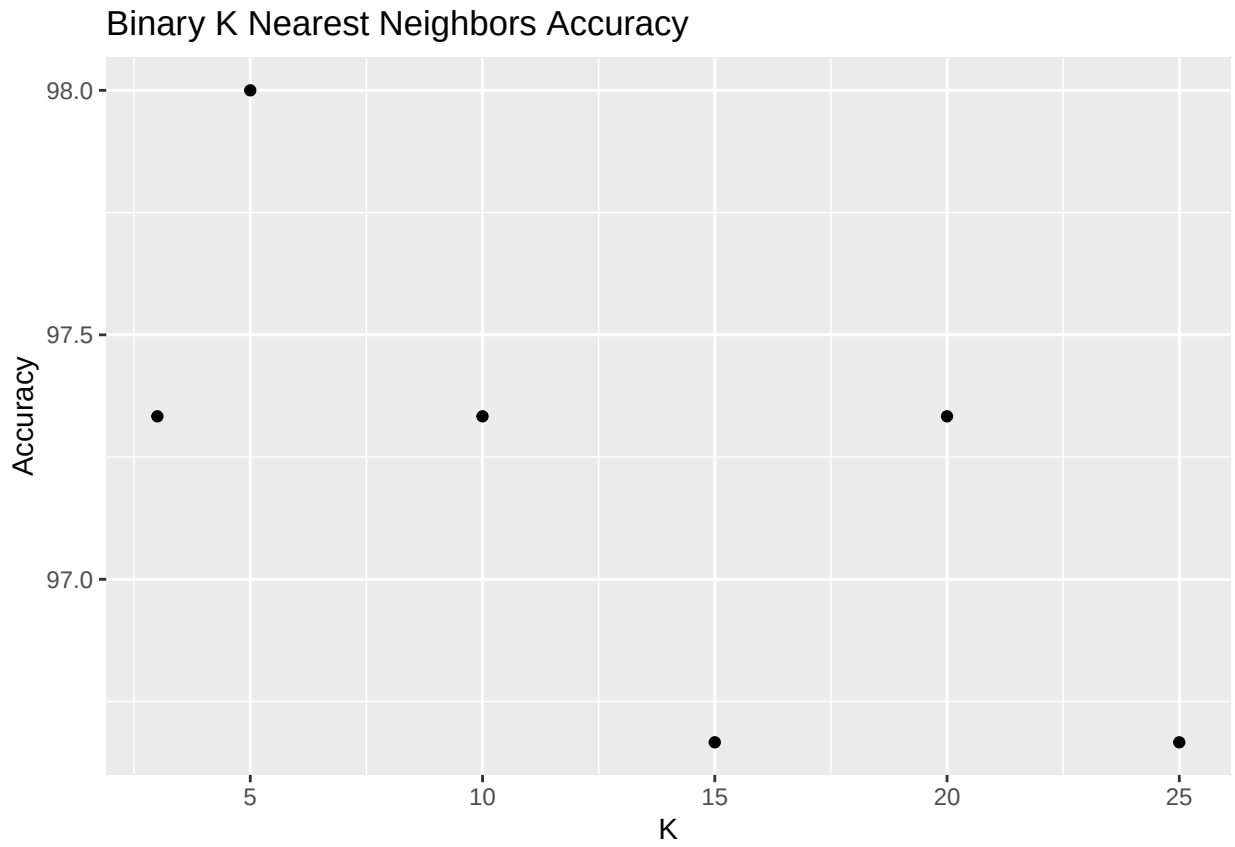
```
t3 <- knn(trinary_train, trinary_test, cl=trinary_target_category, k=3)
t5 <- knn(trinary_train, trinary_test, cl=trinary_target_category, k=5)
t10 <- knn(trinary_train, trinary_test, cl=trinary_target_category, k=10)
t15 <- knn(trinary_train, trinary_test, cl=trinary_target_category, k=15)
t20 <- knn(trinary_train, trinary_test, cl=trinary_target_category, k=20)
t25 <- knn(trinary_train, trinary_test, cl=trinary_target_category, k=25)
```

iii. Compute the accuracy of the resulting models for each value of k.

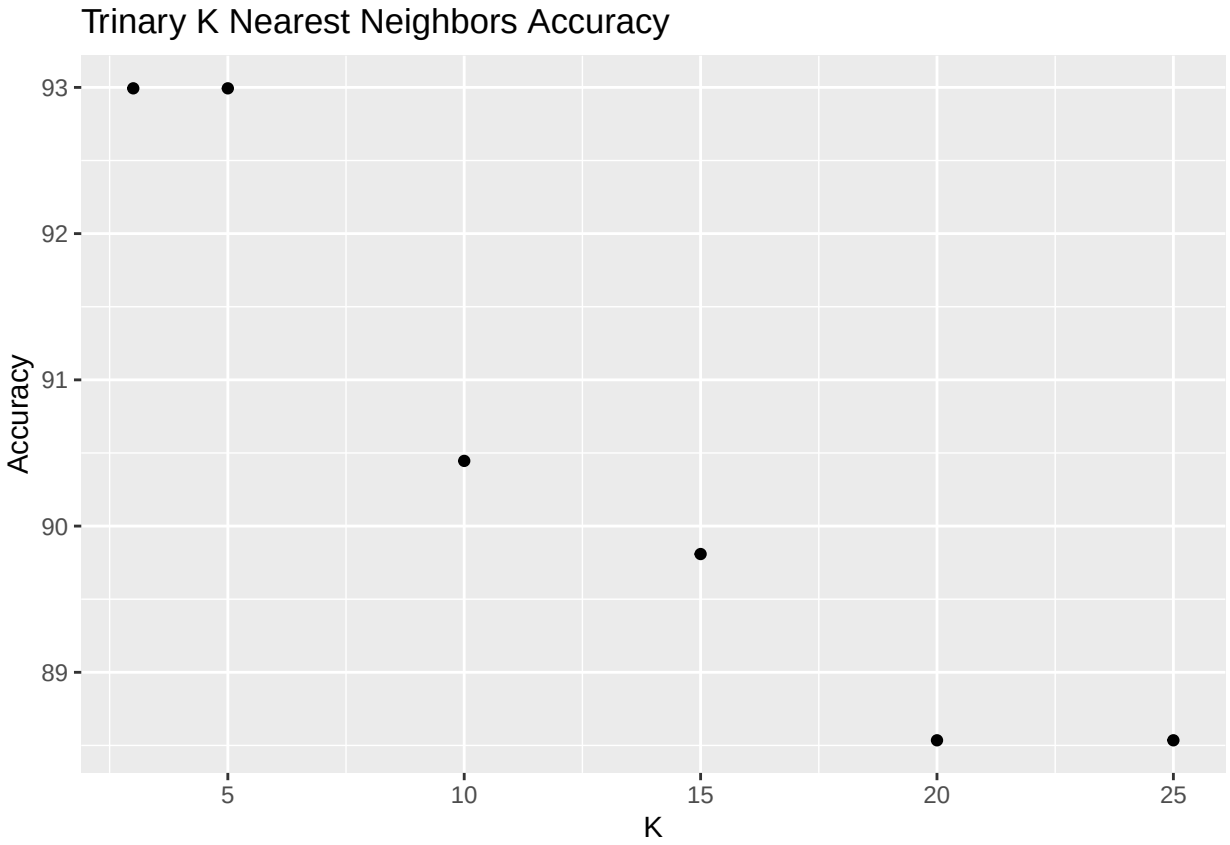
```
# Create confusion matrix
btab3 <- table(b3, binary_test_category)
btab5 <- table(b5, binary_test_category)
btab10 <- table(b10, binary_test_category)
btab15 <- table(b15, binary_test_category)
btab20 <- table(b20, binary_test_category)
btab25 <- table(b25, binary_test_category)

ttab3 <- table(t3, trinary_test_category)
ttab5 <- table(t5, trinary_test_category)
ttab10 <- table(t10, trinary_test_category)
ttab15 <- table(t15, trinary_test_category)
ttab20 <- table(t20, trinary_test_category)
ttab25 <- table(t25, trinary_test_category)

# Compute accuracy of each model
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
ba3 <- accuracy(btab3)
ba5 <- accuracy(btab5)
ba10 <- accuracy(btab10)
ba15 <- accuracy(btab15)
ba20 <- accuracy(btab20)
ba25 <- accuracy(btab25)
ta3 <- accuracy(ttab3)
ta5 <- accuracy(ttab5)
ta10 <- accuracy(ttab10)
ta15 <- accuracy(ttab15)
ta20 <- accuracy(ttab20)
ta25 <- accuracy(ttab25)
```

iv. Plot the results in a graph where the x-axis is the different values of k and the y-axis is the accuracy of the model.

```
ggplot(ba_df, aes(x=K, y=Accuracy))+ggtitle("Binary K Nearest Neighbors Accuracy")+geom_point()
```




```
ggplot(ta_df, aes(x=K, y=Accuracy))+ggtitle("Trinary K Nearest Neighbors Accuracy")+geom_point()
```



v. Looking back at the plots of the data, do you think a linear classifier would work well on these datasets?

I do not think a linear classified would work well on these datasets after reviewing the scatterplots.

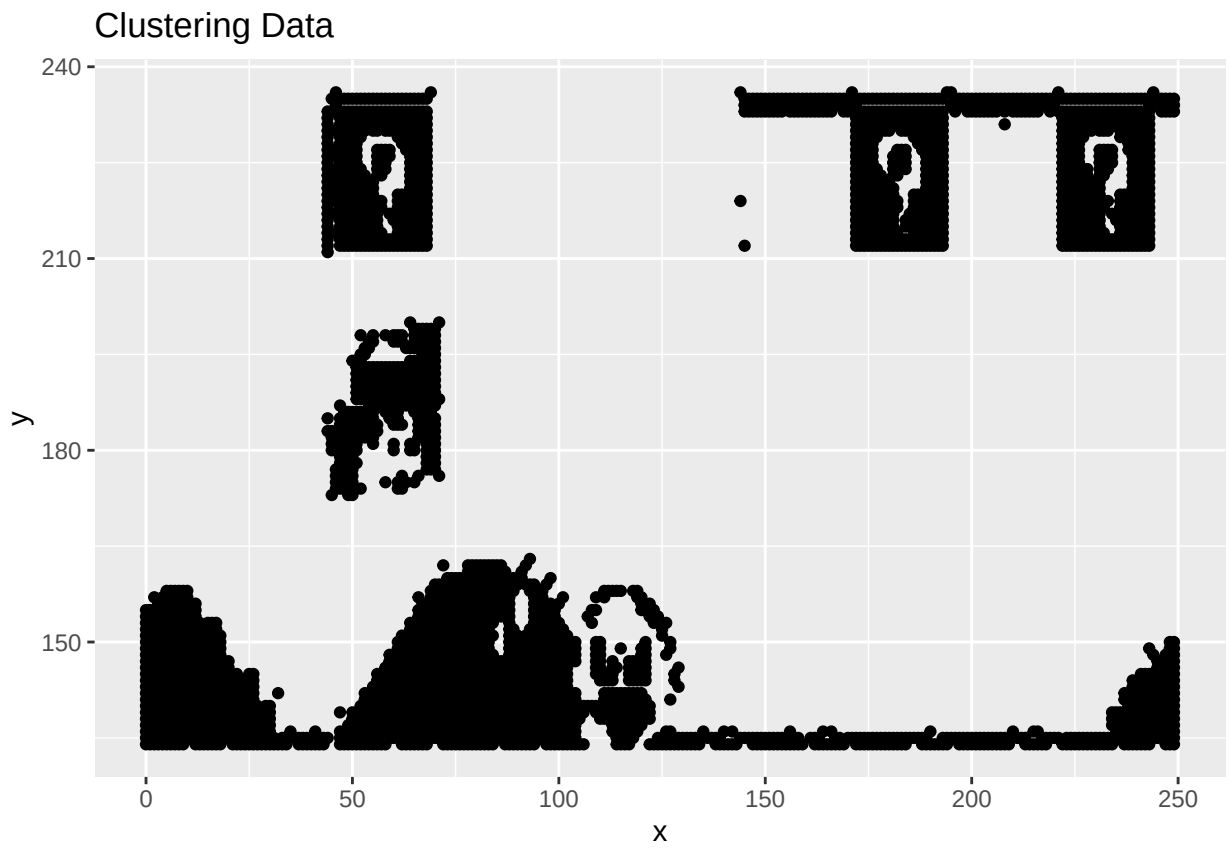
vi. How does the accuracy of your logistic regression classifier from last week compare? Why is the accuracy different between these two methods?

My model in last week's assignment had an accuracy of 58.3%. The K nearest neighbors model fit the data better than a linear model and greatly improves the overall accuracy.

2. Clustering

i. Plot the dataset using a scatter plot.

```
ggplot(clustering_df, aes(x=x, y=y))+ggtitle("Clustering Data")+geom_point()
```



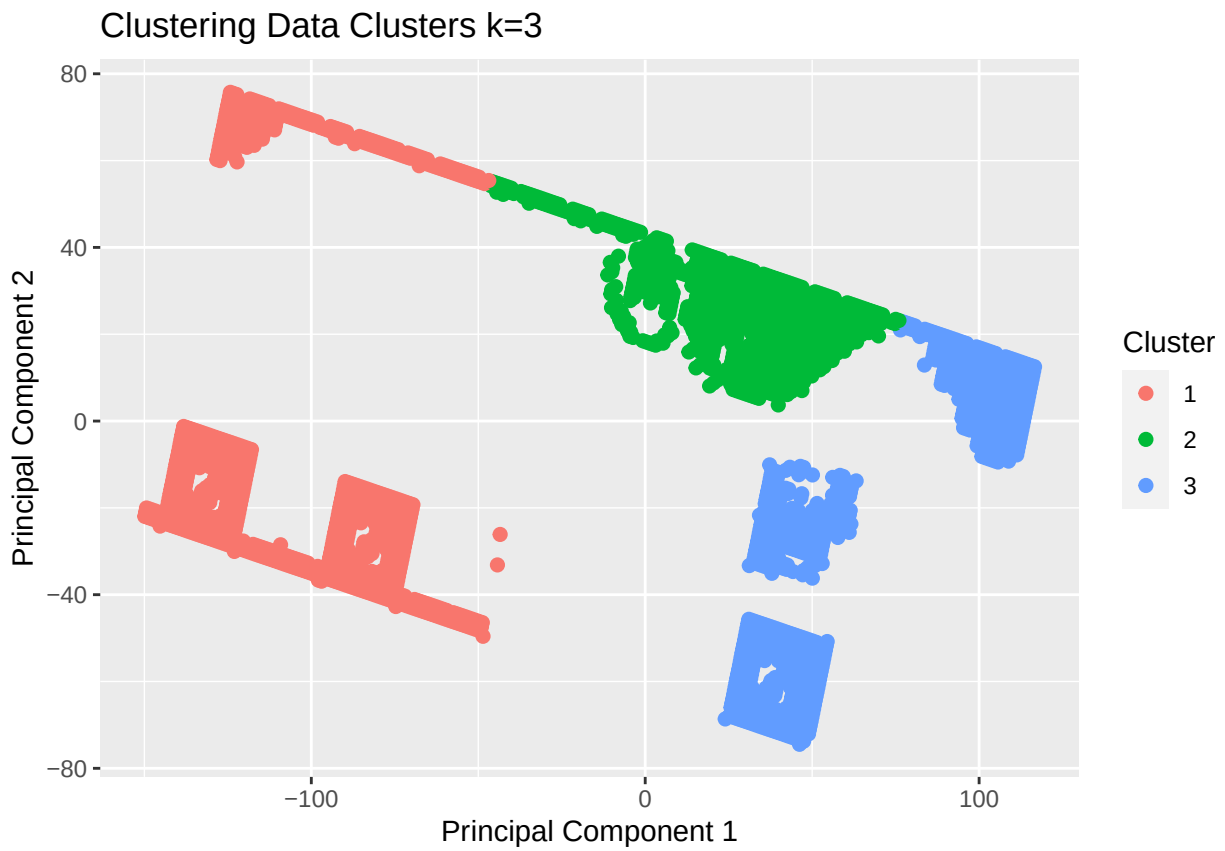
ii. Fit the dataset using the k-means algorithm from k=2 to k=12.

```
set.seed(123)
mariok2 <- kmeans(x=clustering_df, centers=2, nstart = 25)
mariok3 <- kmeans(x=clustering_df, centers=3, nstart = 25)
mariok4 <- kmeans(x=clustering_df, centers=4, nstart = 25)
mariok5 <- kmeans(x=clustering_df, centers=5, nstart = 25)
mariok6 <- kmeans(x=clustering_df, centers=6, nstart = 25)
mariok7 <- kmeans(x=clustering_df, centers=7, nstart = 25)
mariok8 <- kmeans(x=clustering_df, centers=8, nstart = 25)
mariok9 <- kmeans(x=clustering_df, centers=9, nstart = 25)
mariok10 <- kmeans(x=clustering_df, centers=10, nstart = 25)
mariok11 <- kmeans(x=clustering_df, centers=11, nstart = 25)
mariok12 <- kmeans(x=clustering_df, centers=12, nstart = 25)
```

iii. Create a scatter plot of the resultant clusters for each value of k.

```
library(useful)
plot(mariok2, data=clustering_df, title="Clustering Data Clusters k=2")
```


```
plot(mariok3, data=clustering_df, title="Clustering Data Clusters k=3")
```

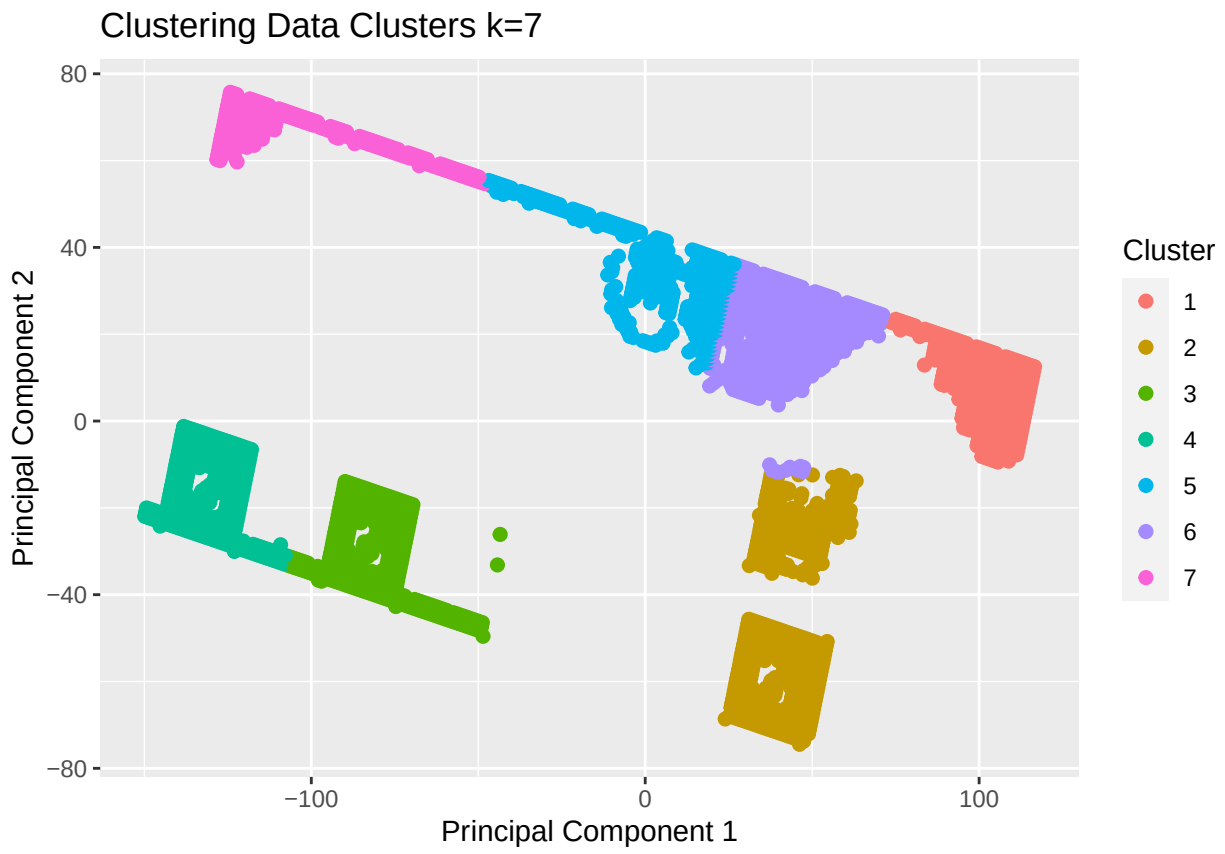



```
plot(mariok4, data=clustering_df, title="Clustering Data Clusters k=4")
```

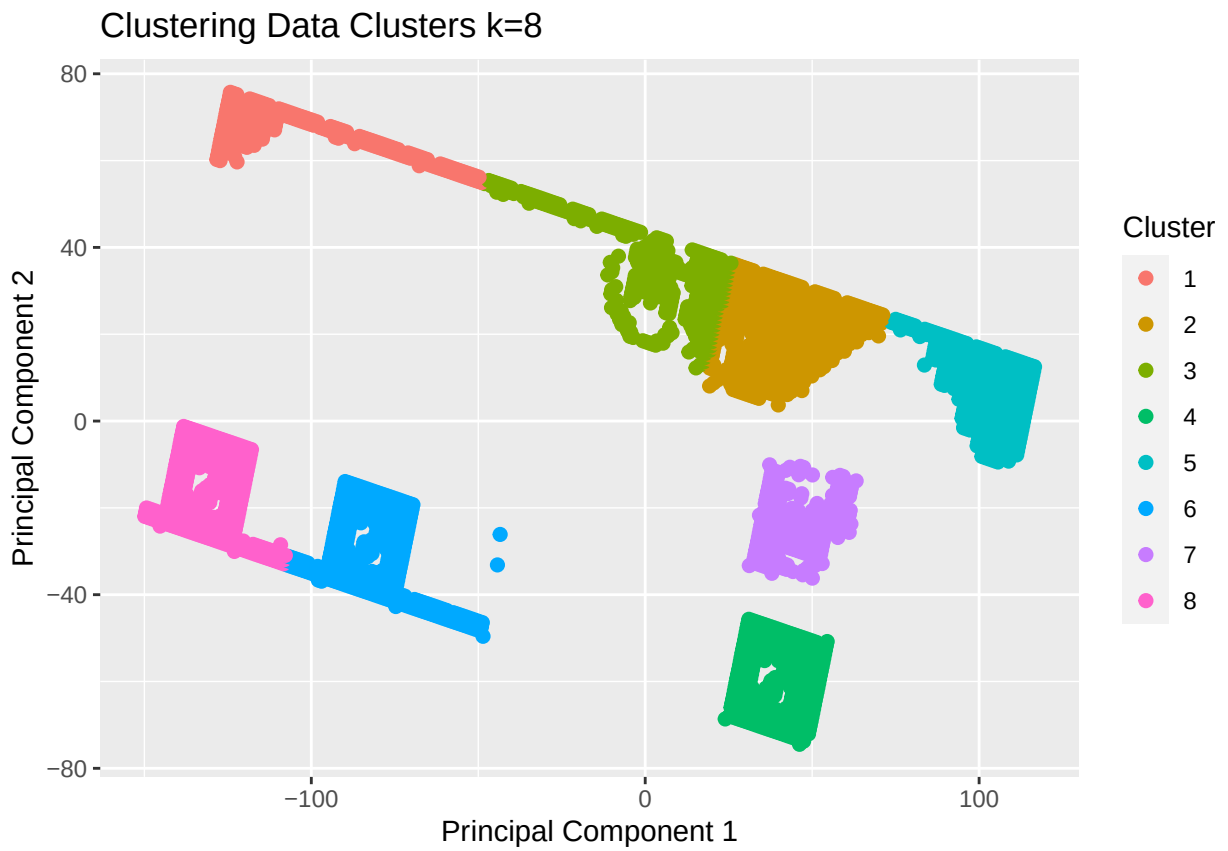

```
plot(mariok5, data=clustering_df, title="Clustering Data Clusters k=5")
```


```
plot(mariok6, data=clustering_df, title="Clustering Data Clusters k=6")
```


```
plot(mariok7, data=clustering_df, title="Clustering Data Clusters k=7")
```




```
plot(mariok8, data=clustering_df, title="Clustering Data Clusters k=8")
```




```
plot(mariok9, data=clustering_df, title="Clustering Data Clusters k=9")
```

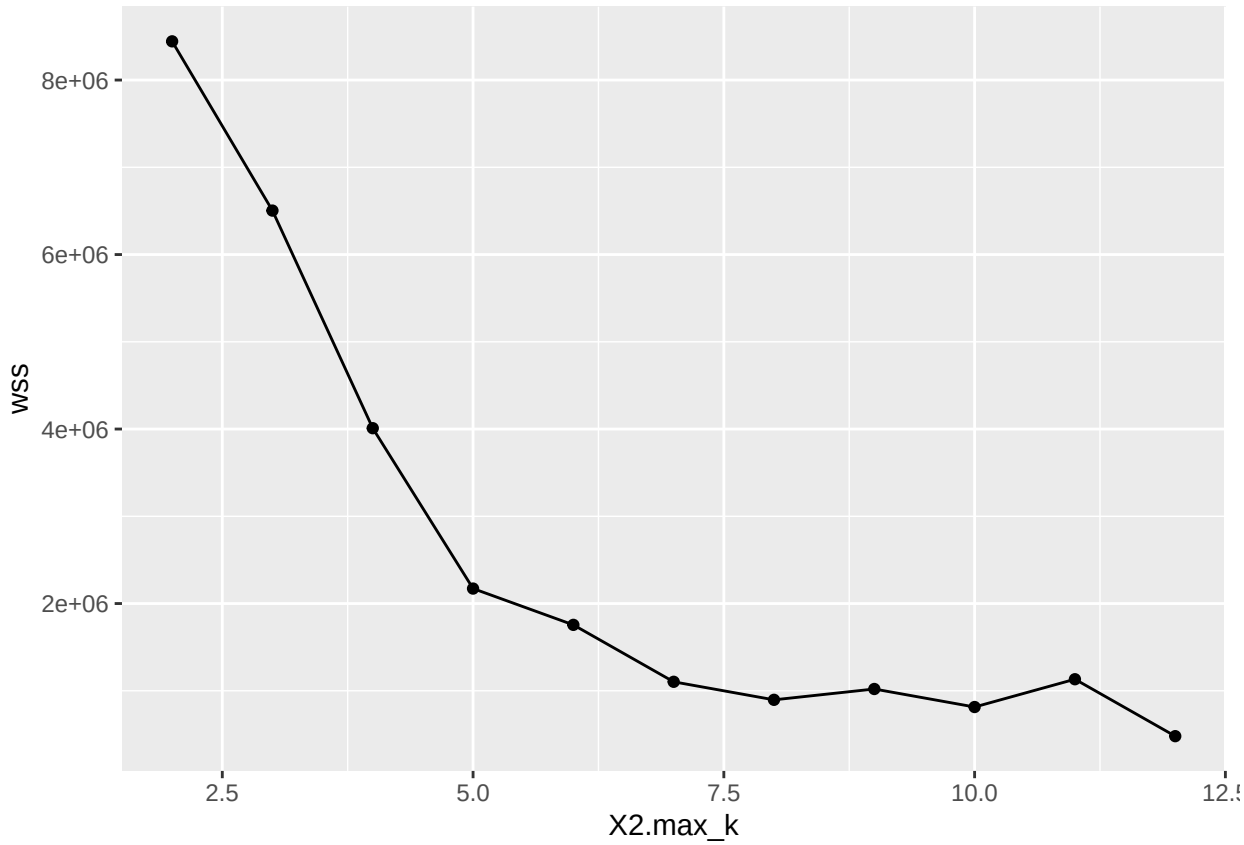

```
plot(mariok10, data=clustering_df, title="Clustering Data Clusters k=10")
```


```
plot(mariok11, data=clustering_df, title="Clustering Data Clusters k=11")
```


```
plot(mariok12, data=clustering_df, title="Clustering Data Clusters k=12")
```

iv. 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.

```
kmean_withinss <- function(k) {
    cluster <- kmeans(clustering_df, k)
    return (cluster$tot.withinss)
}
max_k <- 12
wss <- sapply(2:max_k, kmean_withinss)
elbow <- data.frame(2:max_k, wss)

ggplot(elbow, aes(x = X2.max_k, y = wss)) +
    geom_point() +
    geom_line()
```



v. 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?

The elbow is at 6 clusters.