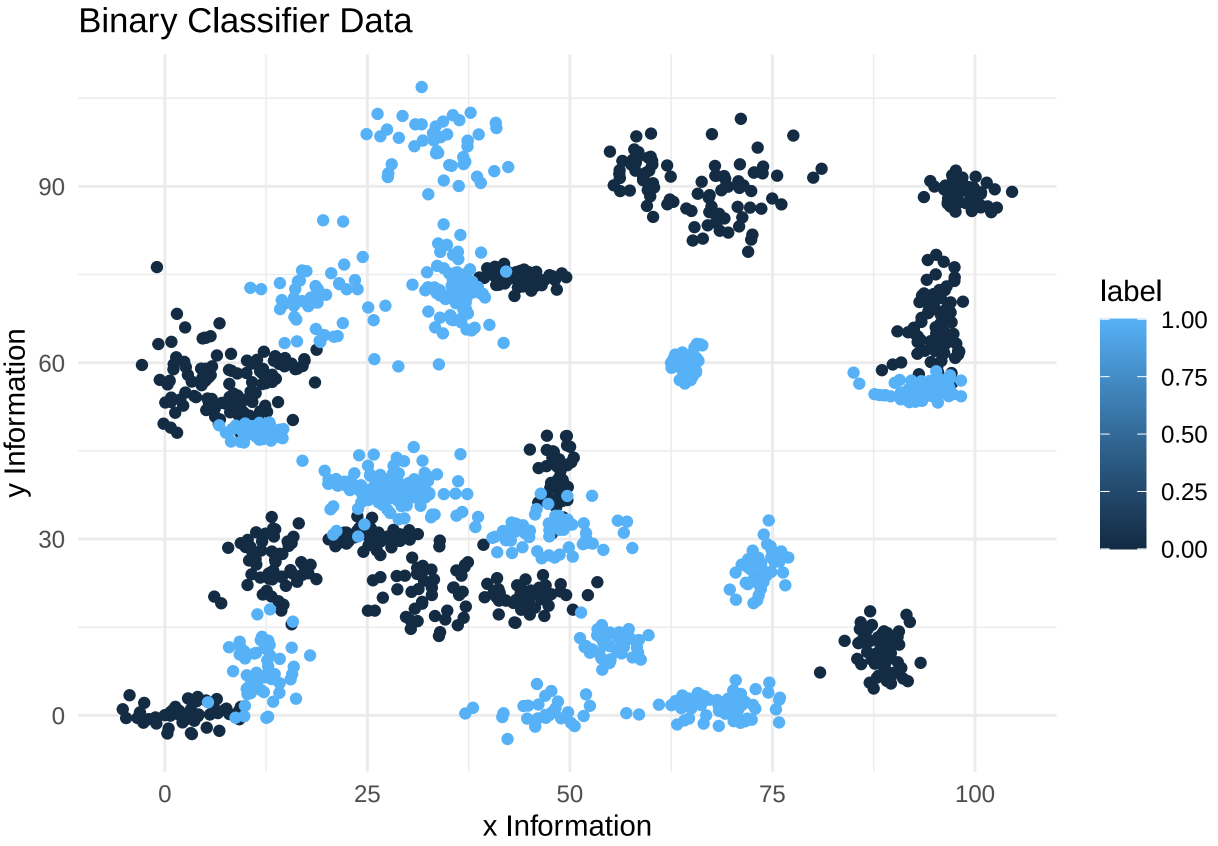
# Assignment 11-12

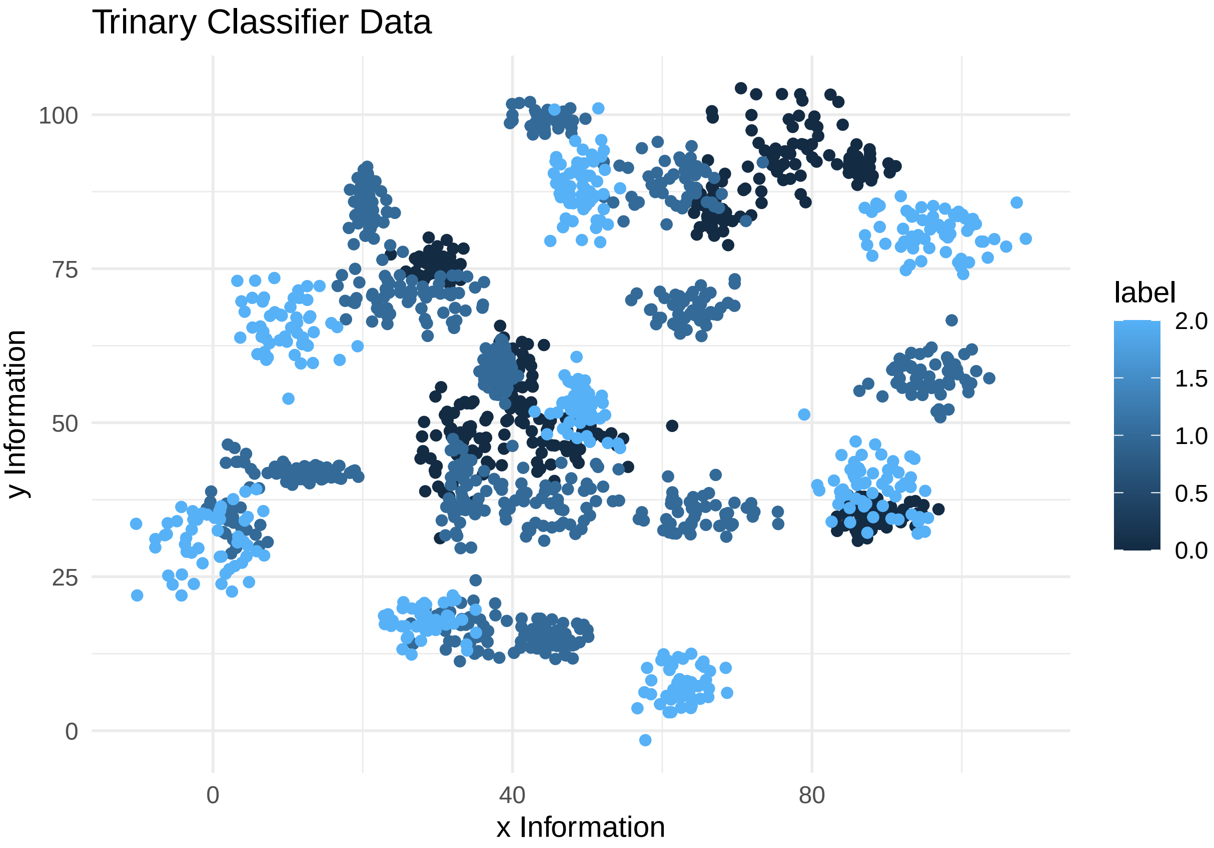
Michelle Helfman

2022-11-19

```r
## Set the working directory to the root of your DSC 520 directory
setwd("/Users/New User/Documents/workspaces/dsc520/data")

## Load data to
bcdata_df <- read.csv("binary-classifier-data.csv")
tcdata_df <- read.csv("trinary-classifier-data.csv")
cldata_df <- read.csv("clustering-data.csv")

## Plot Binary Classifier Data
ggplot(bcdata_df, aes(x=x, y=y, col=label)) + geom_point() +
  ggtitle('Binary Classifier Data') + xlab('x Information') +
  ylab('y Information')
```

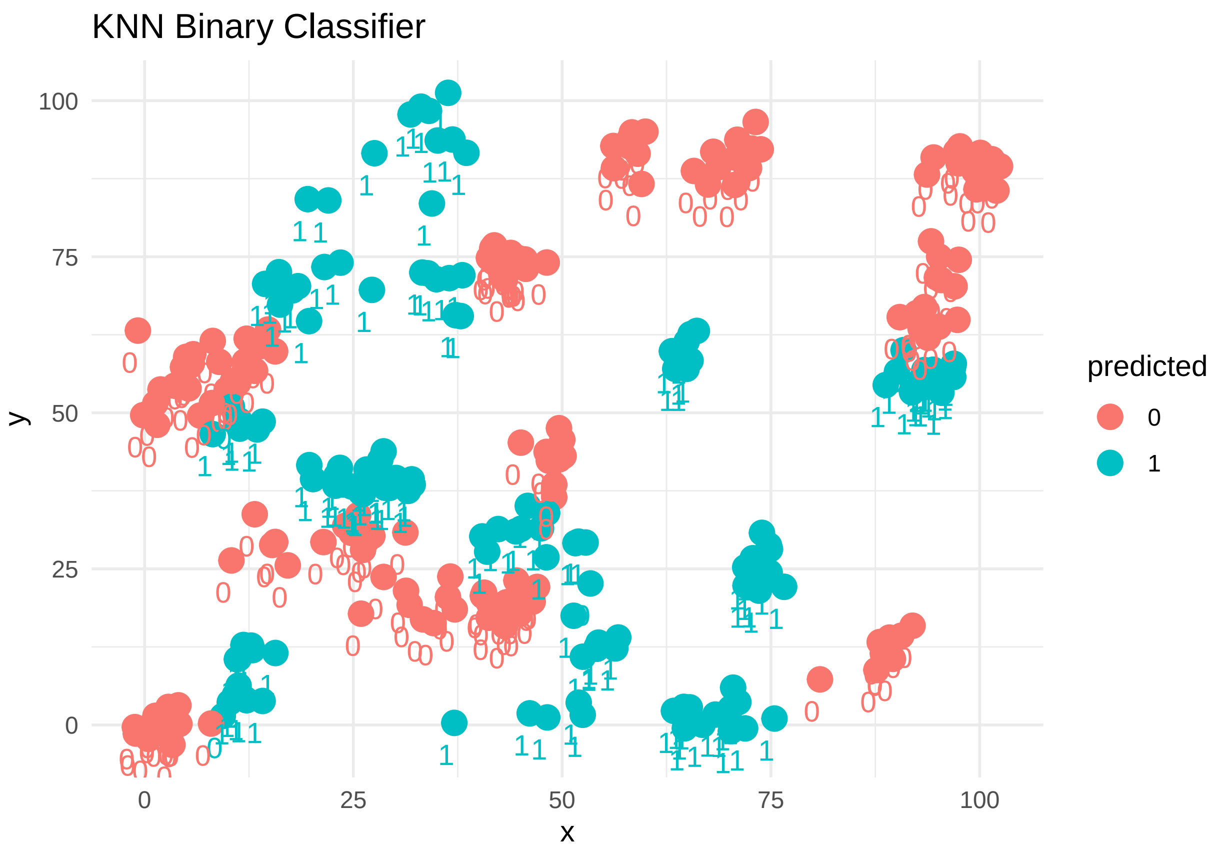


```
## Plot Trinary Classifier Data
ggplot(tcdata_df, aes(x=x, y=y, col=label)) + geom_point() +
  ggtitle('Trinary Classifier Data') + xlab('x Information') +
  ylab('y Information')
```



## Find KNN with Euclidean Distance binary classifier

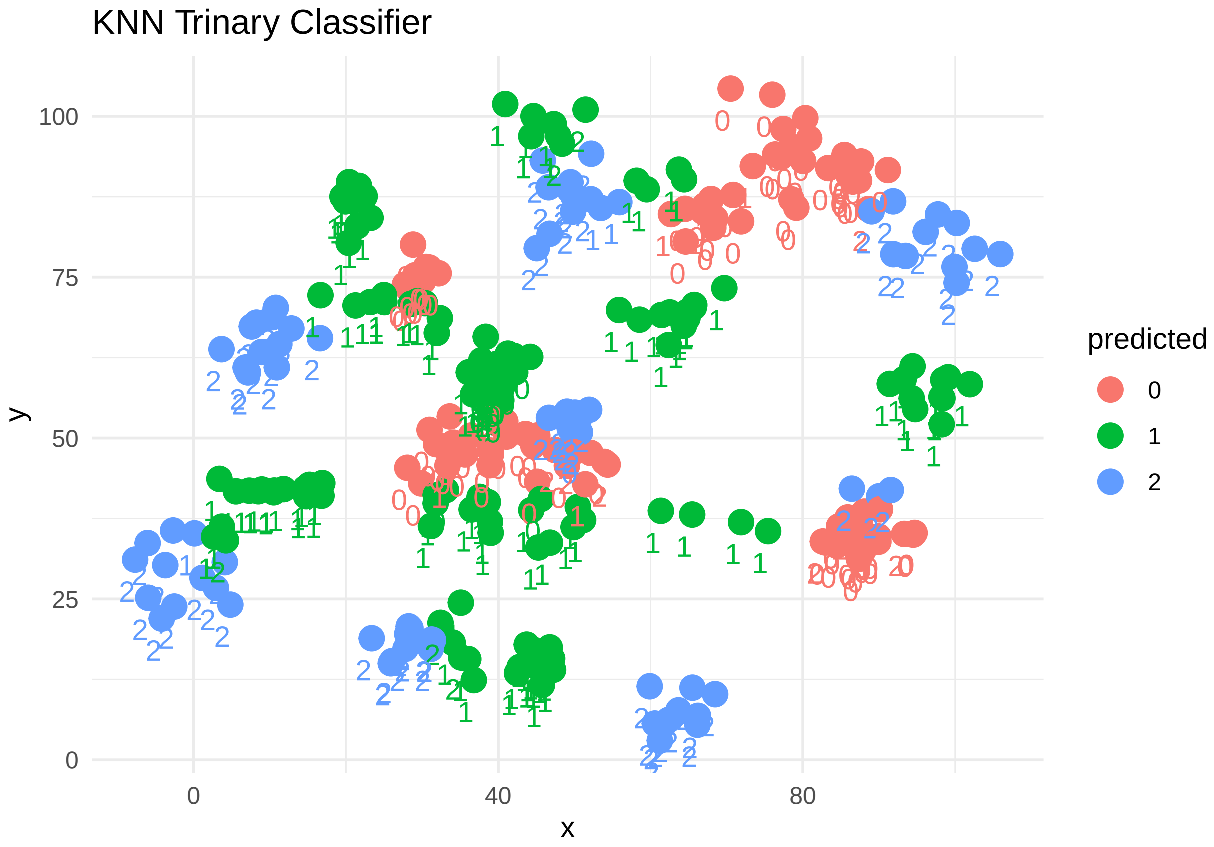
```r
euclidean <- function(a, b) sqrt(sum((a - b)^2))
row_labels <- bcdata_df$label
## Split recs 80/20
set.seed(123)
size <- floor(0.8 * nrow(bcdata_df))

## Get training data
train_ind <- sample(seq_len(nrow(bcdata_df)), size = size)
train_labels <- bcdata_df[train_ind, 1]
test_labels <- row_labels[-train_ind]
data_train <- bcdata_df[train_ind,2:3]
data_test <- bcdata_df[-train_ind,2:3]

# Test Run
predict_rec <- knn(train = data_train,
                   test = data_test,
                   cl = train_labels,
                   #k=11)
                   k = round(sqrt(nrow(data_train))))
## Plot values
plot_predict <- data.frame(
  data_test$x,
  data_test$y,
  predicted = predict_rec)

colnames(plot_predict) <- c("x","y",'predicted')
ggplot(plot_predict, aes(x,y,
                         color = predicted, fill = predicted)) +
  geom_point(size = 4) +
  geom_text(aes(label = test_labels), hjust = 1, vjust = 2) +
  ggtitle("KNN Binary Classifier")
```



```
bianary_euclidean <- euclidean(data_test$x,data_test$y)
```

## Find KNN with Euclidean Distance trinary classifier

```r
euclidean <- function(a, b) sqrt(sum((a - b)^2))
row_labels <- tcdata_df$label
## Split recs 80/20
set.seed(123)
size <- floor(0.8 * nrow(tcdata_df))

## Get training data
train_ind <- sample(seq_len(nrow(tcdata_df)), size = size)
train_labels <- tcdata_df[train_ind, 1]
test_labels <- row_labels[-train_ind]
data_train <- tcdata_df[train_ind,2:3]
data_test <- tcdata_df[-train_ind,2:3]

# Test Run
predict_rec <- knn(train = data_train,
                   test = data_test,
                   cl = train_labels,
                   #k=11)
                   k = round(sqrt(nrow(data_train))))
## Plot values
plot_predict <- data.frame(
  data_test$x,
  data_test$y,
  predicted = predict_rec)

colnames(plot_predict) <- c("x","y",'predicted')
ggplot(plot_predict, aes(x,y,
                         color = predicted, fill = predicted)) +
  geom_point(size = 4) +
  geom_text(aes(label = test_labels), hjust = 1, vjust = 2) +
  ggtitle("KNN Trinary Classifier")
```



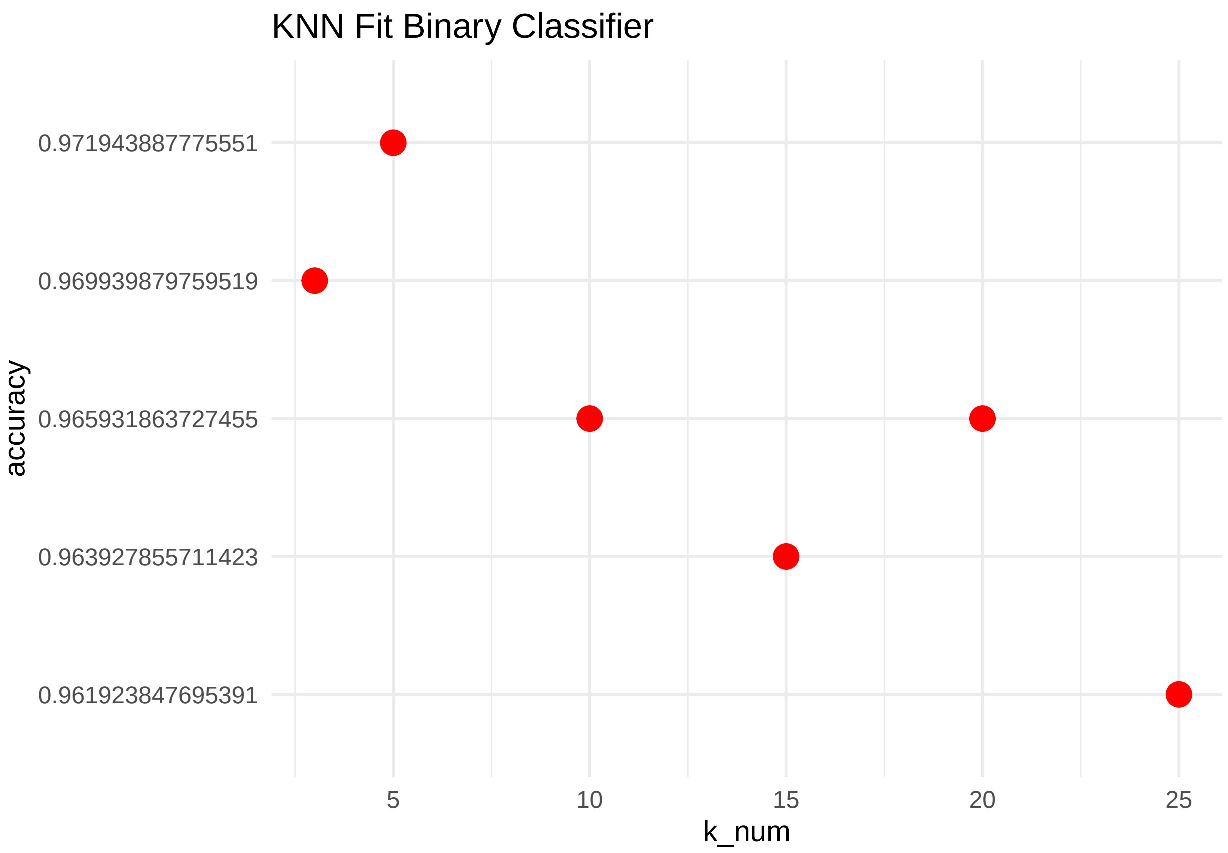
```r
trianary_euclidean <- euclidean(data_test$x,data_test$y)
```

## Accuracy with Fit KNN

## Binary Classifier

```
split <- sample.split(bcdata_df, SplitRatio = 0.8)
train_cl <- subset(bcdata_df, split == "TRUE")
test_cl <- subset(bcdata_df, split == "FALSE")

# Feature Scaling
train_scale <- scale(train_cl[, 2:3])
test_scale <- scale(test_cl[, 2:3])

# Fitting KNN Model
# to training dataset
classifier_knn <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 1)
#classifier_knn

# Confusiin Matrix
cm <- table(test_cl$label, classifier_knn)

# Model Evaluation - Choosing K
# Calculate out of Sample error
accuracy_1 <- 1-(mean(classifier_knn != test_cl$label))

# K = 3
classifier_knn_3 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 3)
accuracy_3 <- 1-(mean(classifier_knn_3 != test_cl$label))

# K = 5
classifier_knn_5 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 5)
accuracy_5 <- 1-(mean(classifier_knn_5 != test_cl$label))

# K = 10
classifier_knn_10 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 10)
accuracy_10 <- 1-(mean(classifier_knn_10 != test_cl$label))

# K = 15
classifier_knn_15 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 15)
```

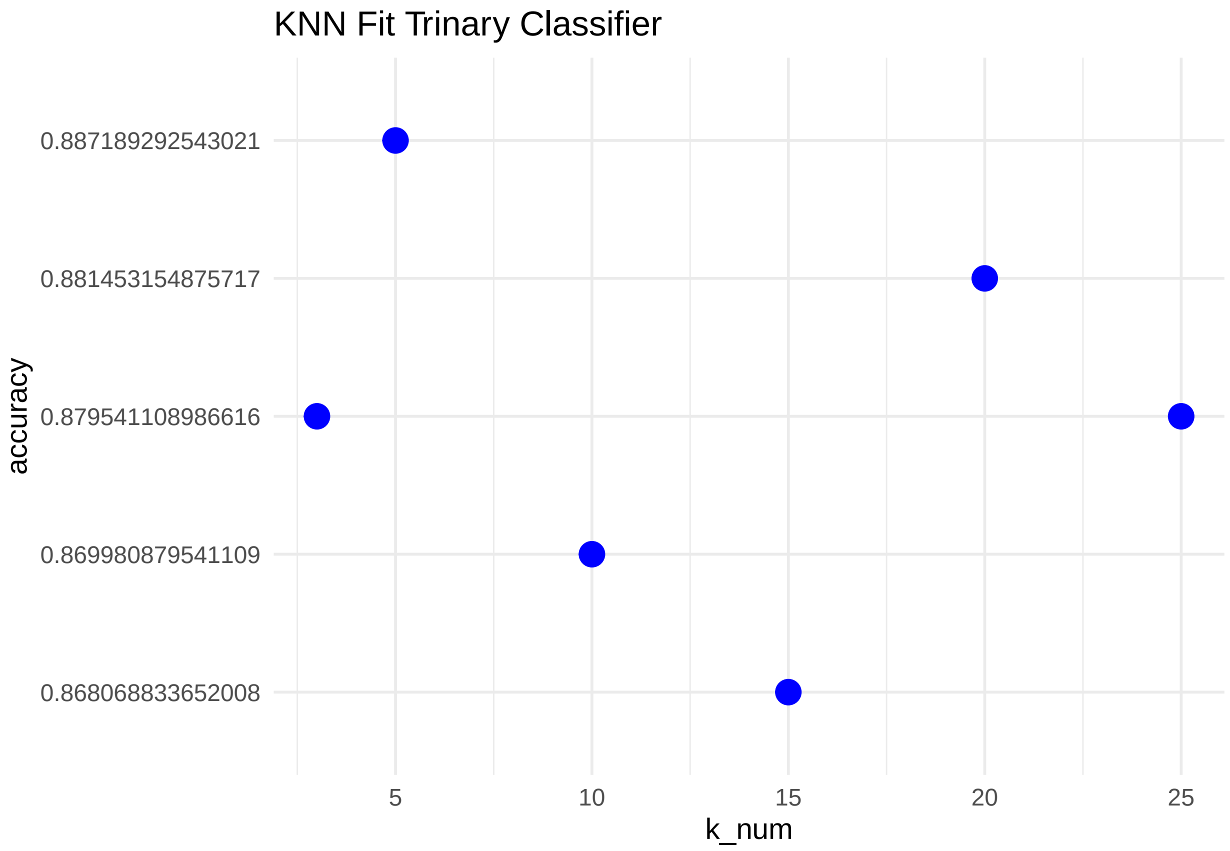
```r
accuracy_15 <- 1-(mean(classifier_knn_15 != test_cl$label))

# K = 20
classifier_knn_20 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 20)
accuracy_20 <- 1-(mean(classifier_knn_20 != test_cl$label))

# K = 25
classifier_knn_25 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 25)
accuracy_25 <- 1-(mean(classifier_knn_25 != test_cl$label))

# Accuracy Dataframe
biplot_df <- data.frame(k_num=c(3, 5, 10, 15, 20, 25))

accuracyl <-
  list(accuracy_3, accuracy_5, accuracy_10, accuracy_15,accuracy_20, accuracy_25)
biplot_df$accuracy <- as.character(accuracyl)

ggplot(biplot_df, aes(x=k_num,y=accuracy)) +
  geom_point(size = 4,color = "red") +
  ggtitle("KNN Fit Binary Classifier")
```


KNN Fit Binary Classifier
accuracy
0.971943887775551
0.969939879759519
0.965931863727455
0.963927855711423
0.961923847695391
5
10
15
20
25
k_num

### Trinary Classifier

```
split <- sample.split(tcdata_df, SplitRatio = 0.8)
train_cl <- subset(tcdata_df, split == "TRUE")
test_cl <- subset(tcdata_df, split == "FALSE")

# Feature Scaling
train_scale <- scale(train_cl[, 2:3])
test_scale <- scale(test_cl[, 2:3])

# Fitting KNN Model
# to training dataset
classifier_knn <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 1)
#classifier_knn

# Confusiin Matrix
cm <- table(test_cl$label, classifier_knn)

# Model Evaluation - Choosing K
# Calculate out of Sample error
accuracy_1 <- 1-(mean(classifier_knn != test_cl$label))

# K = 3
classifier_knn_3 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 3)
accuracy_3 <- 1-(mean(classifier_knn_3 != test_cl$label))

# K = 5
classifier_knn_5 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 5)
accuracy_5 <- 1-(mean(classifier_knn_5 != test_cl$label))

# K = 10
classifier_knn_10 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 10)
accuracy_10 <- 1-(mean(classifier_knn_10 != test_cl$label))

# K = 15
classifier_knn_15 <- knn(train = train_scale,
                      test = test_scale,
                      cl = train_cl$label,
                      k = 15)
accuracy_15 <- 1-(mean(classifier_knn_15 != test_cl$label))
```

```r
# K = 20
classifier_knn_20 <- knn(train = train_scale,
                         test = test_scale,
                         cl = train_cl$label,
                         k = 20)
accuracy_20 <- 1-(mean(classifier_knn_20 != test_cl$label))

# K = 25
classifier_knn_25 <- knn(train = train_scale,
                         test = test_scale,
                         cl = train_cl$label,
                         k = 25)
accuracy_25 <- 1-(mean(classifier_knn_25 != test_cl$label))

# Accuracy Dataframe
biplot_df <- data.frame(k_num=c(3, 5, 10, 15, 20, 25))

accuracyl <-
  list(accuracy_3, accuracy_5, accuracy_10, accuracy_15,accuracy_20, accuracy_25)
biplot_df$accuracy <- as.character(accuracyl)

ggplot(biplot_df, aes(x=k_num,y=accuracy)) +
  geom_point(size = 4,color = "blue") +
  ggtitle("KNN Fit Trinary Classifier")
```

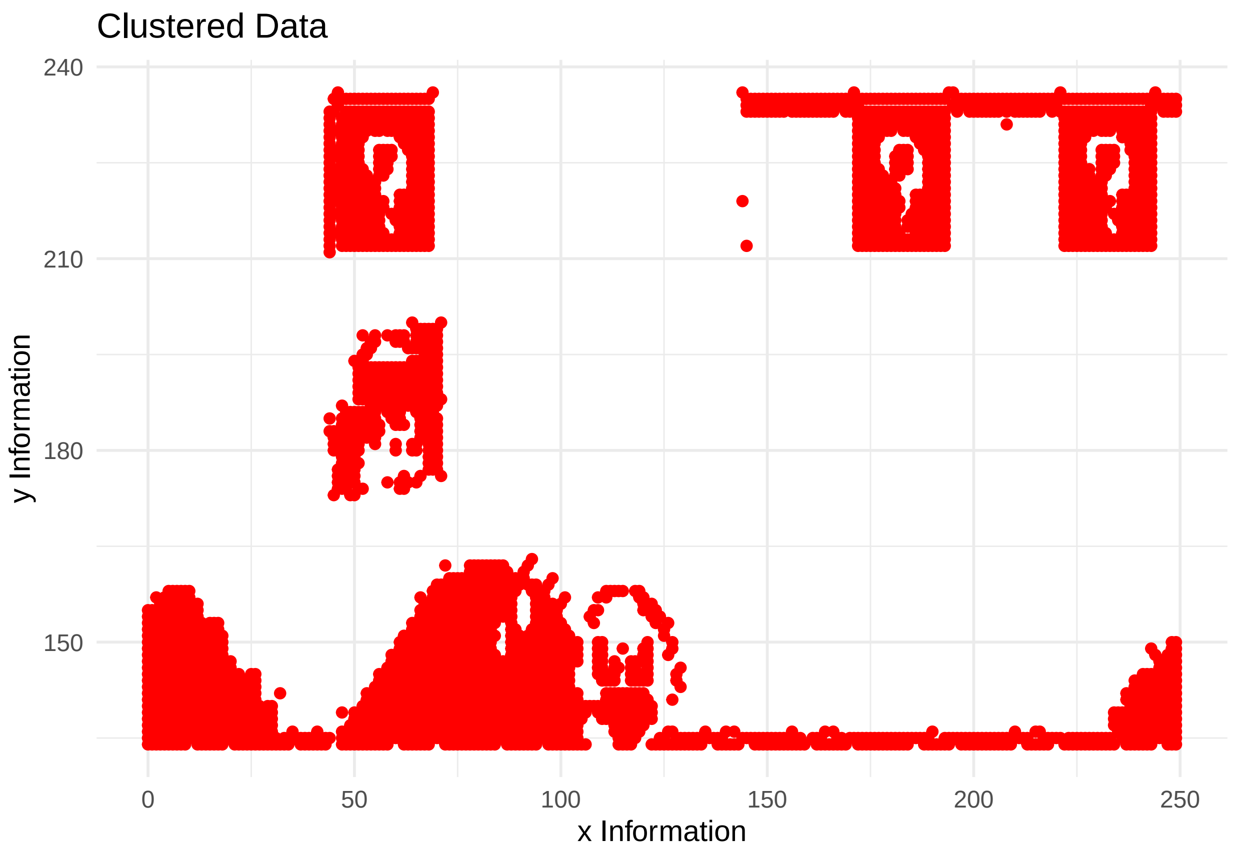


## KNN vs GLM

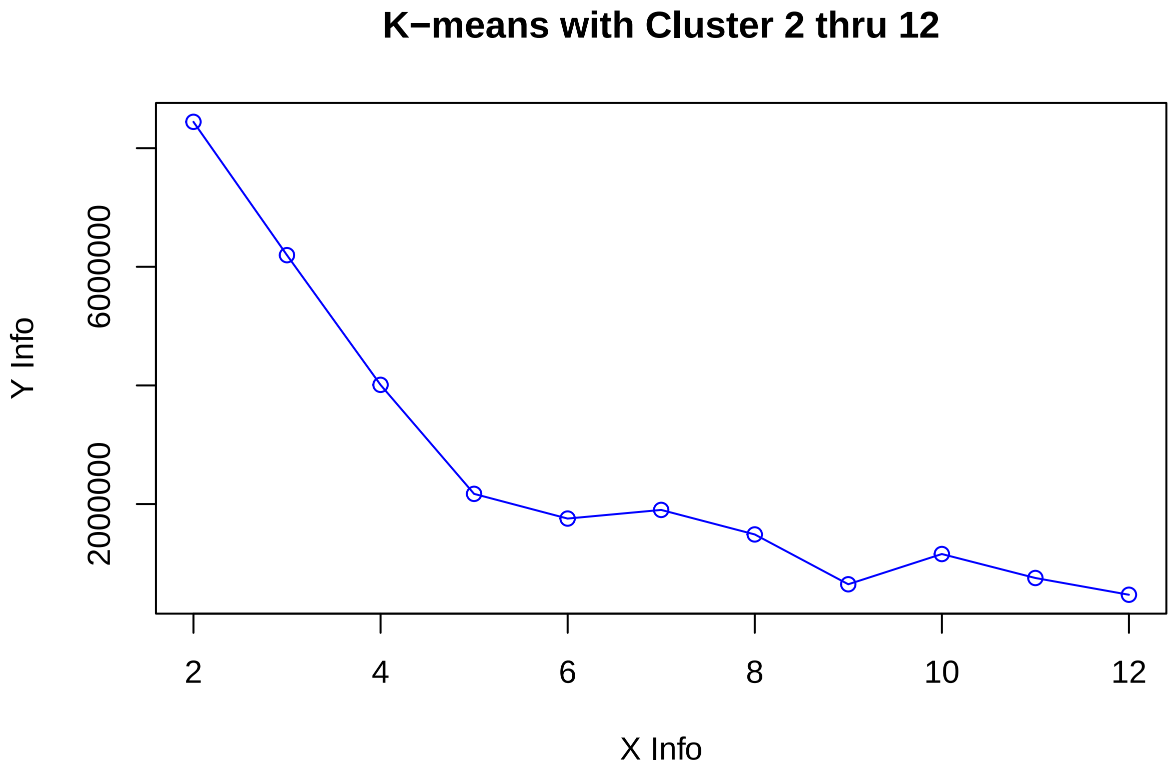
The models from week 10 and 11/12 are 2 completely different ways of looking at the same dataset. Week 10 is a Generalized Linear Model which is a regression model and Week 11/12 is a K Nearest Neighbor. GLM uses a prior information to predict a class. KNN uses proximity to make predictions. It's not just about the data, it's also the algorithm you use. Different algorithms ccan produce widely different results.

### Clustering

```
## Plot Clustering Data
ggplot(cldata_df, aes(x=x, y=y)) + geom_point(color= "red") +
  ggtitle('Clustered Data') + xlab('x Information') +
  ylab('y Information')
```



```
# Fitting K-Means clustering Model
set.seed(123)
wss <- NULL
for (i in 2:12){
  fit <- kmeans(cldata_df,centers = i)
  wss <- c(wss, fit$tot.withinss)
}

plot(2:12, wss, type = "o", main = "K-means with Cluster 2 thru 12",
     xlab = "X Info", ylab = "Y Info", col="blue")
```



```
distance <- mean(fit$betweenss/fit$totss)
```

The Distance to Center is 0.983536446946917

The Elbow Observation is K = 5 on the X Axis.

### Where Am I?

I'm at the spot where I'm understanding the statistical concepts and associating R code with those concepts. The biggest issue is what code to use and when. For example, with K-Mean Clusters, I found at least 4 completely different code samples to use and each one produced a different result. I'm not sure I've chosen the correct code.