Extra Credit Clinical Decision Making and Pattern Recognition in Health Care

Avery Holloman

2024-08-20

```
# First I utlized the "readxl" package to read in my Excel file that I converted to ".xlsx" from common delimited format to handle the data better
library(readxl)
# Second, I utilized the dplyr because it is part of a family of functional tools in the Tidyverse that allows myself to filter, arrange, summarize, and manipulate the data frame as I am going through my statistical analysis.
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# Thirdly, I made revisions to my code multiple times as I was receiving syntax errors in the data so I researched the "anomalize" package for better utilization of the anomaly detection for my time series data because it helped identify outliers in my time series data by decomposing specific features within my data to normalize the points for better anomaly detection method attributes for data visualization.
library(anomalize)
#tibbletime extends the tibble data structure to better handle time series data. It enables time-based manipulation of tibbles, making it easier to work with date and time indexes.
library(tibbletime)
```

```
##
## Attaching package: 'tibbletime'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

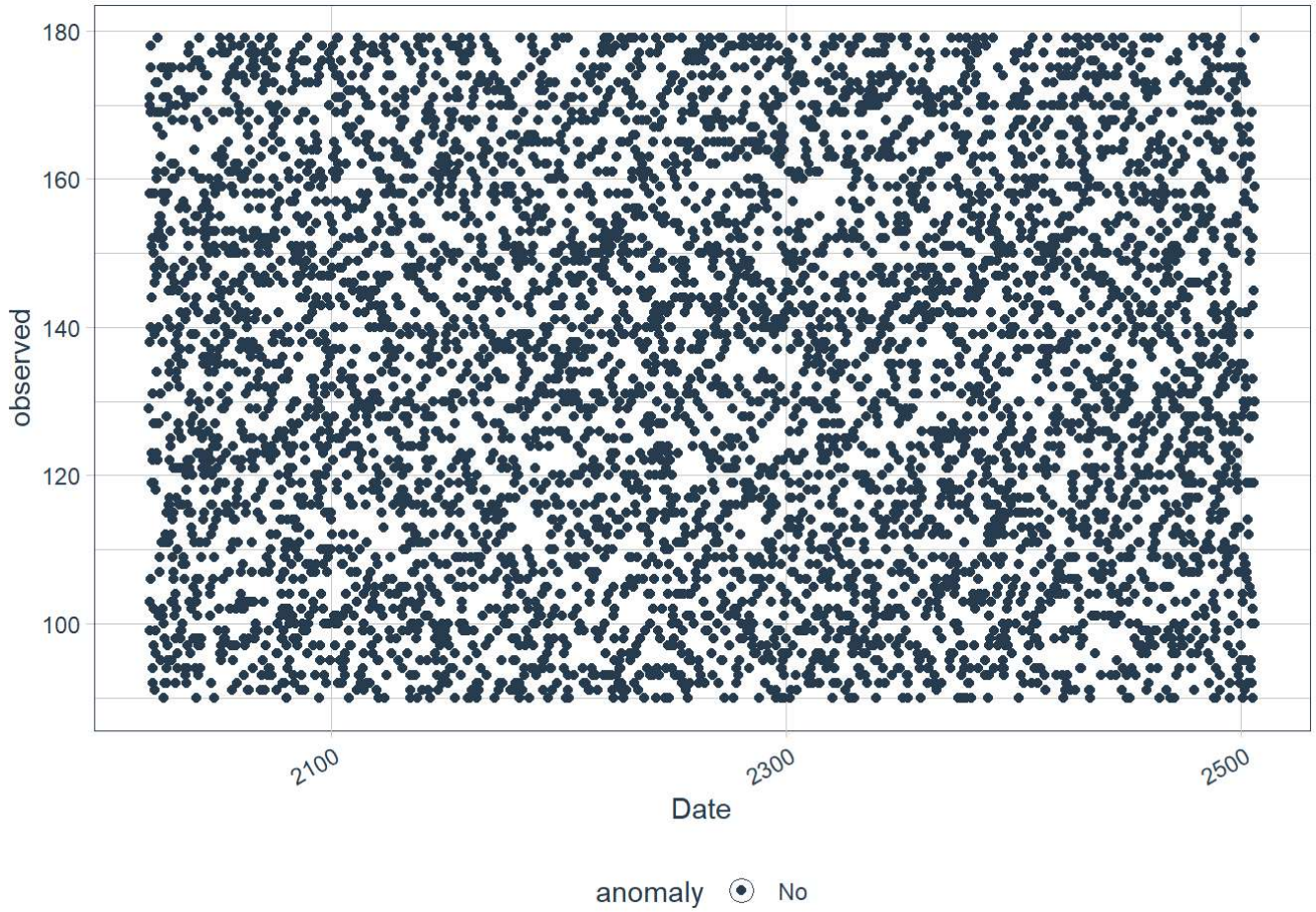
```
library(ggplot2)

# Loaded my data from the Excel file within my onedrive
healthcare <- read_excel("C:/Users/jacob/OneDrive/Desktop/R Studio Projects 2024/Datasets/health
care.xlsx")

# I Added a Date column to my data to represents monthly records starting from January 2020
healthcare <- healthcare %>%
  mutate(Date = seq(as.Date("2020-01-01"), by = "month", length.out = n()))

# Next I converted my data to a tibble
healthcare_tbl <- as_tibble(healthcare)

# Then, I had to convert my data to a resonable time-based tibble
healthcare_time_tbl <- healthcare_tbl %>%
  as_tbl_time(index = Date)

# Now it is time to perform anomaly detection on the 'Blood_Pressure_mmHg' column within the dat
aset
anomaly_detection <- healthcare_time_tbl %>%
  time_decompose(Blood_Pressure_mmHg, method = "stl") %>%
  anomalize(remainder, method = "iqr") %>%
  time_recompose()
```

```
## frequency = 12 months
```

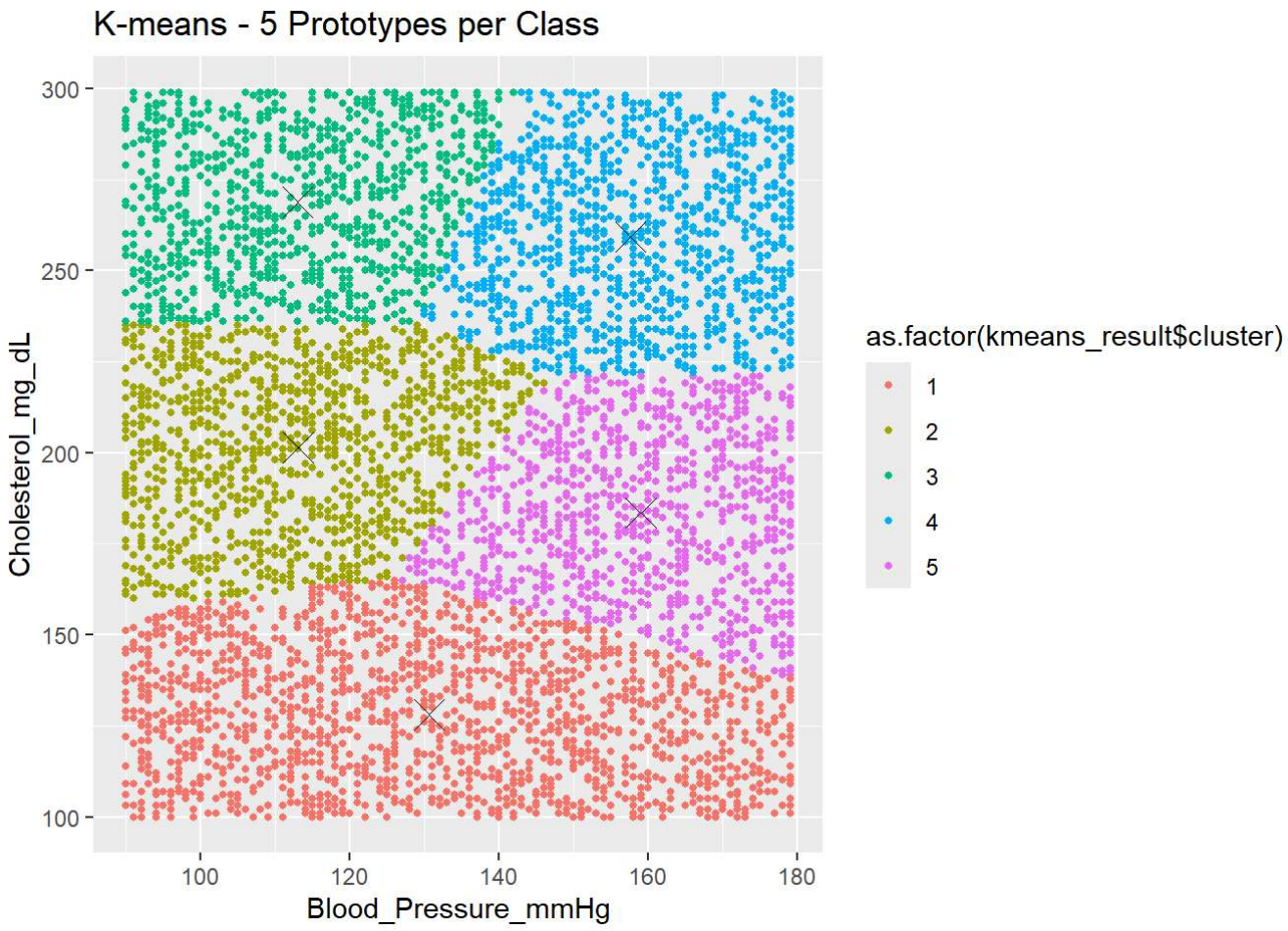
```
## trend = 60 months
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
# Let take a look with a plot of the anomalies within the data
plot_anomalies(anomaly_detection)
```




```
# For knitting purposes I loaded the library again
library(readxl)
library(cluster)
library(ggplot2)

# Loading data as I was receiving syntax erros in knitting
healthcare <- read_excel("C:/Users/jacob/OneDrive/Desktop/R Studio Projects 2024/Datasets/healthcare.xlsx")

# Now it is time to perform K-means clustering based on 'Blood_Pressure_mmHg' and 'Cholesterol_mg_dL' in the dataset
set.seed(123)
kmeans_result <- kmeans(healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers = 5)

# Lets take a look at the plot with ggplot2 by plotting K-means
ggplot(healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.factor(kmeans_result$cluster))) +
  geom_point(size = 1.0) +
  geom_point(data = as.data.frame(kmeans_result$centers), aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL), color = "black", size = 5, shape = 4) +
  ggtitle("K-means - 5 Prototypes per Class")
```

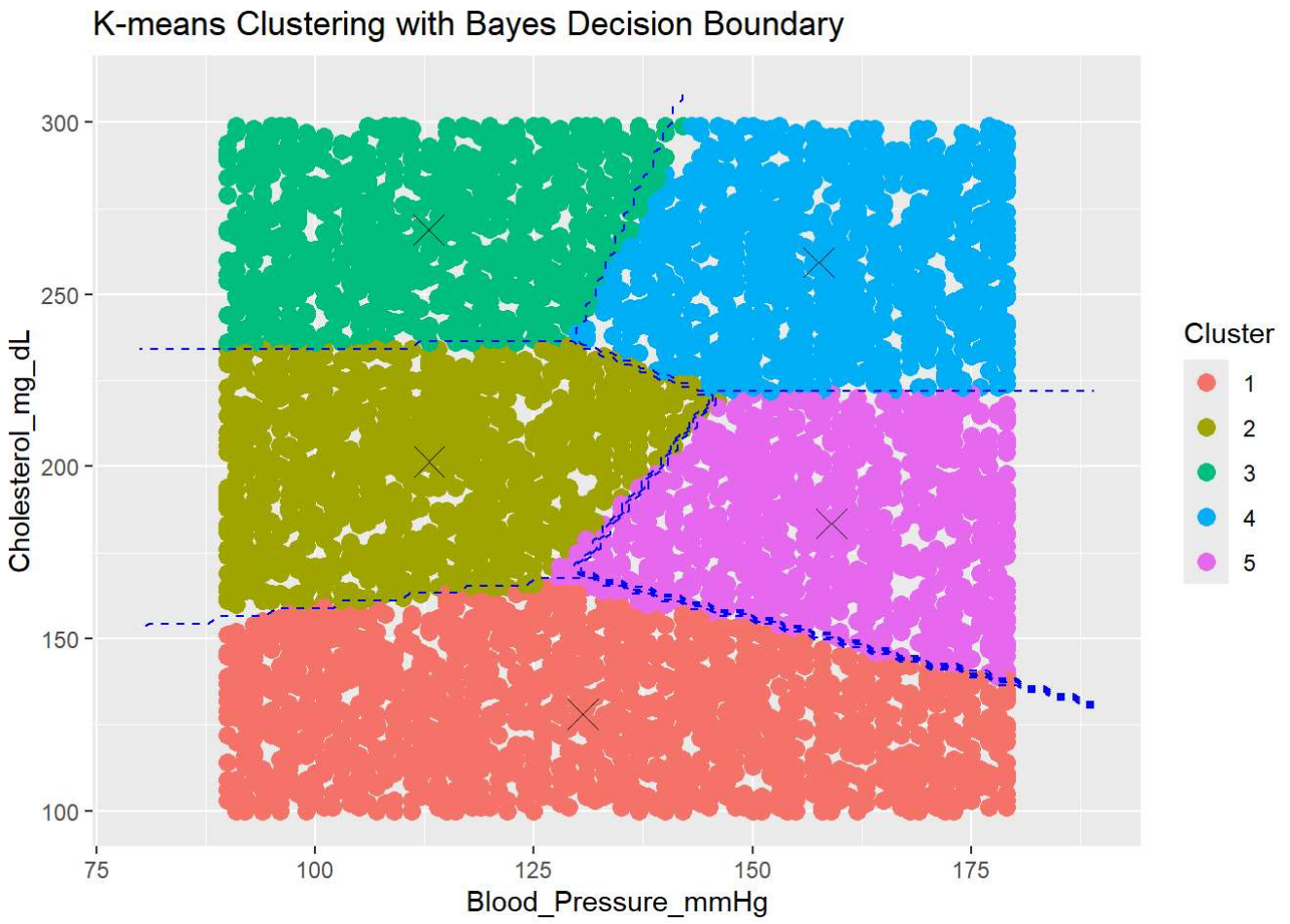



```
# Loading libraries to help but "MASS" and "ggplot" were giving me warning messages when trying to utilize the Bayes decision boundary but I was able to manage to still produce the plots
library(readxl)
library(cluster)
library(ggplot2)
library(MASS)
```

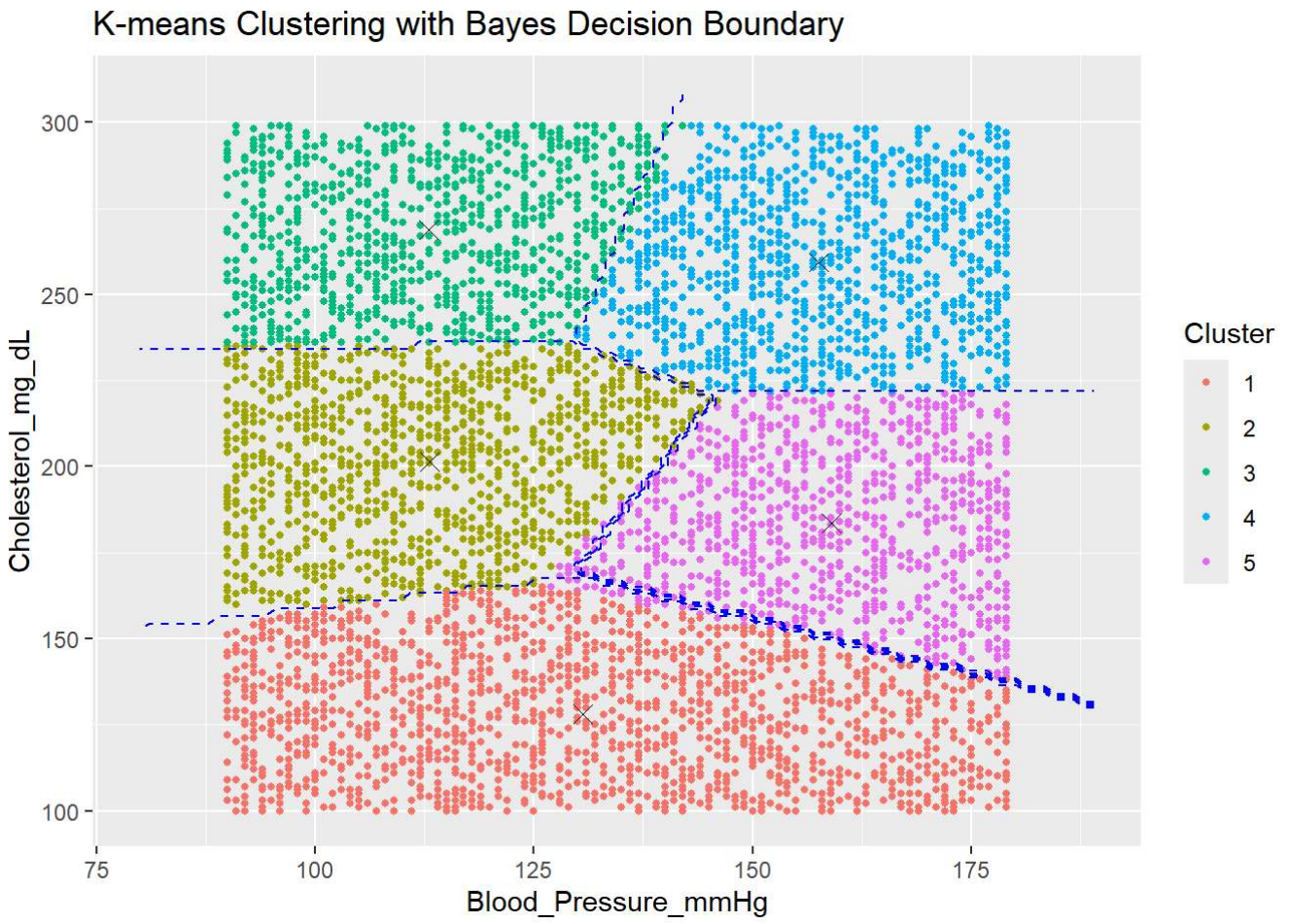
```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
# Now lets take a look at the results of the K-means clustering based on 'Blood_Pressure_mmHg' a
nd 'Cholesterol_mg_dL' within the dataset
set.seed(123)
kmeans_result <- kmeans(healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers =
5)

# In order to accomplish this I decided to create a grid of points spanning the feature space wi
thin the dataset
x_min <- min(healthcare$Blood_Pressure_mmHg) - 10
x_max <- max(healthcare$Blood_Pressure_mmHg) + 10
y_min <- min(healthcare$Cholesterol_mg_dL) - 10
y_max <- max(healthcare$Cholesterol_mg_dL) + 10

grid <- expand.grid(Blood_Pressure_mmHg = seq(x_min, x_max, length.out = 100),
                    Cholesterol_mg_dL = seq(y_min, y_max, length.out = 100))

# Now I need to assign a specific function to assign each point in the grid to the nearest clust
er center of the data points
assign_cluster <- function(x, centers) {
  distances <- apply(centers, 1, function(center) sqrt(sum((x - center)^2)))
  return(which.min(distances))
}

# Next I will assign each grid point to the nearest cluster
grid$cluster <- apply(grid, 1, assign_cluster, centers = kmeans_result$centers)
grid$cluster <- as.factor(grid$cluster)

# Next I will estimate the Bayes decision boundary utilizing the linear discriminant analysis me
thod
lda_model <- lda(cluster ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = data.frame(healthcar
e, cluster = kmeans_result$cluster))
grid$decision_boundary <- predict(lda_model, grid)$class

# Now let me use ggplot2 but first I need to adjust the geom_point to size 3 as I was having tro
uble with the "MASS" library and "ggplot" warning messages that was producing blank plots for de
tecting my K-means results with decision boundary method
ggplot(healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.factor(kmeans_
result$cluster))) +
  geom_point(size = 3) +
  geom_point(data = as.data.frame(kmeans_result$centers), aes(x = Blood_Pressure_mmHg, y = Chole
sterol_mg_dL), color = "black", size = 5, shape = 4) +
  geom_contour(data = grid, aes(z = as.numeric(decision_boundary)), bins = 5, color = "blue", li
netype = "dashed") +
  ggtitle("K-means Clustering with Bayes Decision Boundary") +
  scale_color_discrete(name = "Cluster")
```


K-means Clustering with Bayes Decision Boundary
Cholesterol_mg_dL
300
250
200
150
100
75
100
125
150
175
Blood_Pressure_mmHg
Cluster
1
2
3
4
5

```
# Loading the libraries in each markdown file for my Bayes decision boundary to prevent warning
messages and knitting complications
library(readxl)
library(cluster)
library(ggplot2)
library(MASS)

# Now let lets take a look at the K-means clustering results based on 'Blood_Pressure_mmHg' and
'Cholesterol_mg_dL' within the dataset
set.seed(123)
kmeans_result <- kmeans(healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers =
5)

# First I need to create a grid of points spanning the feature space
x_min <- min(healthcare$Blood_Pressure_mmHg) - 10
x_max <- max(healthcare$Blood_Pressure_mmHg) + 10
y_min <- min(healthcare$Cholesterol_mg_dL) - 10
y_max <- max(healthcare$Cholesterol_mg_dL) + 10

grid <- expand.grid(Blood_Pressure_mmHg = seq(x_min, x_max, length.out = 100),
                    Cholesterol_mg_dL = seq(y_min, y_max, length.out = 100))

# Next I need a function to assign to each particular point in the grid to the nearest cluster c
enter for better accuracy
assign_cluster <- function(x, centers) {
  distances <- apply(centers, 1, function(center) sqrt(sum((x - center)^2)))
  return(which.min(distances))
}

# Next I will assign each grid point to the nearest cluster
grid$cluster <- apply(grid, 1, assign_cluster, centers = kmeans_result$centers)
grid$cluster <- as.factor(grid$cluster)

# Now I will utilize the Bayes decision boundary using linear discriminant analysis to estimate
my results more accurately
lda_model <- lda(cluster ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = data.frame(healthcar
e, cluster = kmeans_result$cluster))
grid$decision_boundary <- predict(lda_model, grid)$class

# Now I will use ggplot2 with a color schema and geom_point of 1 because 3 made it to blurry and
hard to define the plot for my K-means results with the decision boundary to allow smaller data
points
ggplot(healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.factor(kmeans_
result$cluster))) +
  geom_point(size = 1.0) +  # Adjusted size for smaller points
  geom_point(data = as.data.frame(kmeans_result$centers), aes(x = Blood_Pressure_mmHg, y = Chole
sterol_mg_dL), color = "black", size = 3, shape = 4) +  # Adjusted size for cluster centers
  geom_contour(data = grid, aes(z = as.numeric(decision_boundary)), bins = 5, color = "blue", li
netype = "dashed") +
  ggtitle("K-means Clustering with Bayes Decision Boundary") +
  scale_color_discrete(name = "Cluster")
```




```
# Loading more libraries but this time I added "mclust" for the Gaussian Mixture Model and EM al
gorithm for better knitting purposes as well
library(cluster)
library(ggplot2)
library(mclust)
```

```
## Package 'mclust' version 6.1.1
## Type 'citation("mclust")' for citing this R package in publications.
```

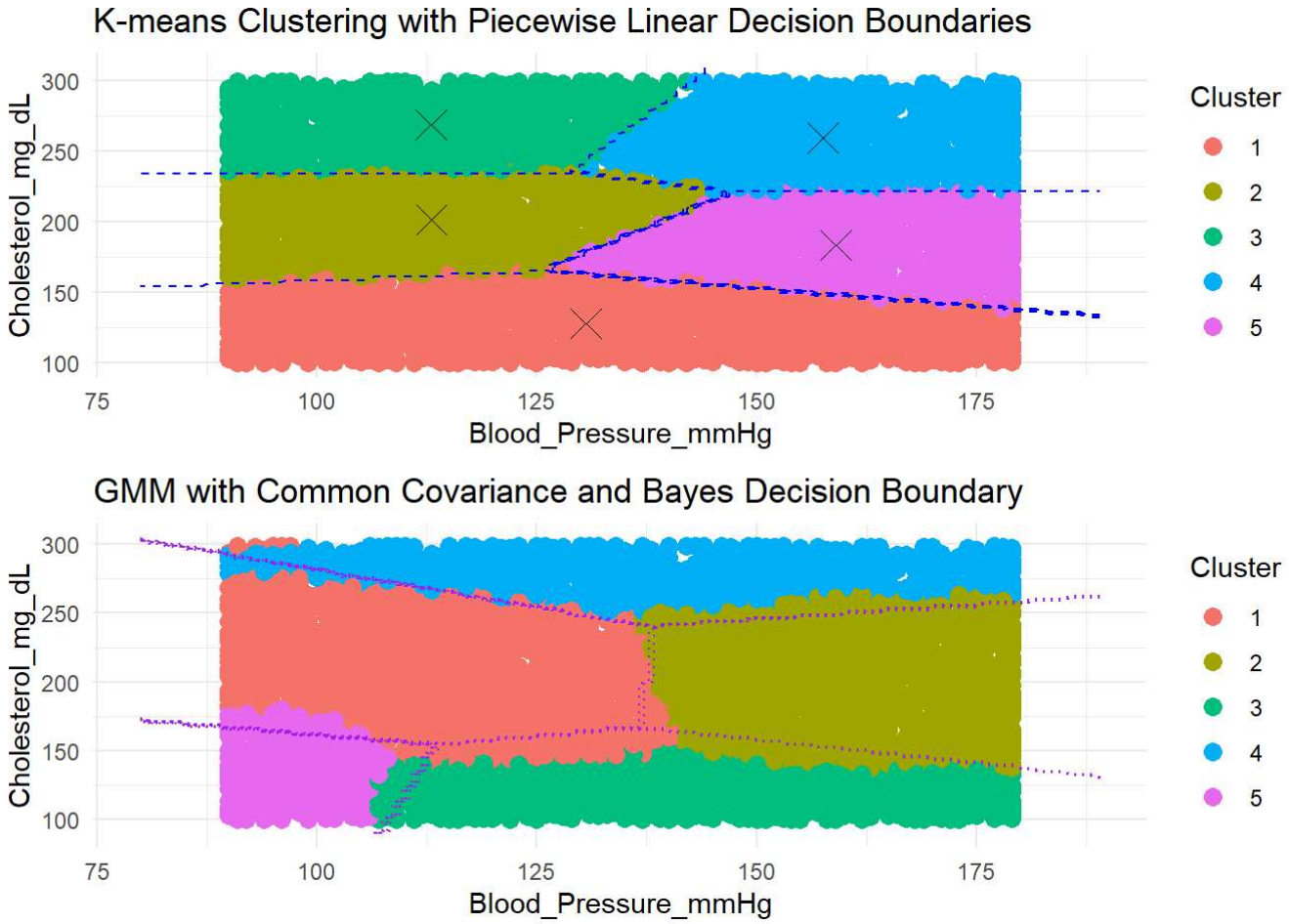
```
library(MASS)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

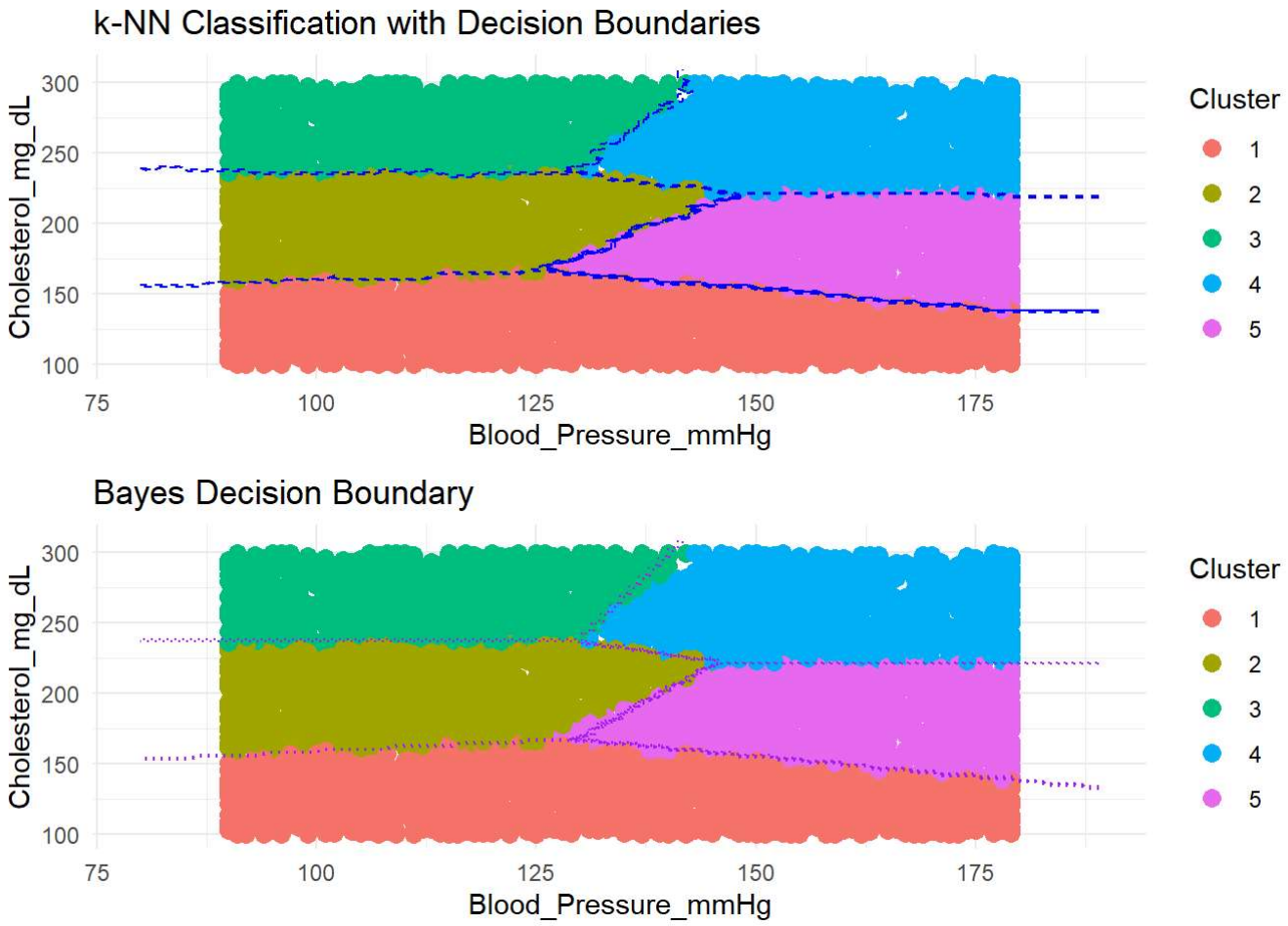
```
# Now lets take a look at the K-means clustering based on 'Blood_Pressure_mmHg' and 'Cholesterol
_mg_dL' within my data
set.seed(123)
kmeans_result <- kmeans(healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers =
5)

# First I need to assign a cluster to each particular grid point for the K-means model for bette
r accuracy
x_min <- min(healthcare$Blood_Pressure_mmHg) - 10
x_max <- max(healthcare$Blood_Pressure_mmHg) + 10
y_min <- min(healthcare$Cholesterol_mg_dL) - 10
y_max <- max(healthcare$Cholesterol_mg_dL) + 10

grid <- expand.grid(Blood_Pressure_mmHg = seq(x_min, x_max, length.out = 100),
                    Cholesterol_mg_dL = seq(y_min, y_max, length.out = 100))

assign_cluster <- function(x, centers) {
  distances <- apply(centers, 1, function(center) sqrt(sum((x - center)^2)))
  return(which.min(distances))
}

grid$kmeans_cluster <- apply(grid, 1, assign_cluster, centers = kmeans_result$centers)
grid$kmeans_cluster <- as.factor(grid$kmeans_cluster)

# Now it is time to fit my Gaussian Mixture Model (GMM)
gmm_result <- Mclust(healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], G = 5)

# Next I will assign cluster to each grid point for the GMM model
grid$gmm_cluster <- predict(gmm_result, newdata = grid[, c("Blood_Pressure_mmHg", "Cholesterol_m
g_dL")])$classification
grid$gmm_cluster <- as.factor(grid$gmm_cluster)

# Next I will estimate the Bayes decision boundary utilizing LDA for the GMM clusters within my
dataset
lda_model <- lda(gmm_cluster ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = grid)
grid$decision_boundary <- predict(lda_model, grid)$class

# To have a better visualization I will plot the results in a two-panel approach

# The Upper panel will represent the K-means clustering with piecewise linear decision boundarie
s
p1 <- ggplot(healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.factor(k
means_result$cluster))) +
  geom_point(size = 3) +
  geom_point(data = as.data.frame(kmeans_result$centers), aes(x = Blood_Pressure_mmHg, y = Chole
sterol_mg_dL), color = "black", size = 5, shape = 4) +
  geom_contour(data = grid, aes(z = as.numeric(kmeans_cluster)), bins = 5, color = "blue", linet
ype = "dashed") +
  ggtitle("K-means Clustering with Piecewise Linear Decision Boundaries") +
  scale_color_discrete(name = "Cluster") +
  theme_minimal()
```

```
# Next the Lower panel will have the GMM clustering with Bayes decision boundary
p2 <- ggplot(healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.factor(g
mm_result$classification))) +
  geom_point(size = 3) +
  geom_contour(data = grid, aes(z = as.numeric(decision_boundary)), color = "purple", linetype =
"dotted") +
  ggtitle("GMM with Common Covariance and Bayes Decision Boundary") +
  scale_color_discrete(name = "Cluster") +
  theme_minimal()

# Now it is time to combine the plots into a two-panel for better visulziation
grid.arrange(p1, p2, ncol = 1)
```



```
# Here I added a new library called "class" I found online for better k-NN classification for th
e Bayes decision boundary
library(cluster)
library(ggplot2)
library(class)
library(MASS)
library(gridExtra)

# For better comparison later on I will now start with the K-means clustering of Blood Pressure
and Cholesterol
set.seed(123)
kmeans_result <- kmeans(healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers =
5)

# I will utilize 5 for the number of neighbors in my K-NN classification
k <- 5
grid <- expand.grid(Blood_Pressure_mmHg = seq(min(healthcare$Blood_Pressure_mmHg) - 10, max(heal
thcare$Blood_Pressure_mmHg) + 10, length.out = 100),
                    Cholesterol_mg_dL = seq(min(healthcare$Cholesterol_mg_dL) - 10, max(healthca
re$Cholesterol_mg_dL) + 10, length.out = 100))

# Now I will train the healthcare data in order to utilize the k-NN algorithm to classify each p
oint in the grid from the "class" library
knn_result <- knn(train = healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")],
                  test = grid,
                  cl = kmeans_result$cluster,
                  k = k)

grid$knn_cluster <- as.factor(knn_result)

# Now lets estimate the Bayes decision boundary utilizing the LDA for the k-NN clusters by assig
ning it lda_model
lda_model <- lda(knn_cluster ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = grid)
grid$decision_boundary <- predict(lda_model, grid)$class

# Now I will combine the plots as I did earliar to display the results better in a two-panel dim
ension

# First I have the Upper panel that represents the k-NN classification and added decision bounda
ries to seperate the data points
p1 <- ggplot() +
  geom_point(data = healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.f
actor(kmeans_result$cluster)), size = 3) +
  geom_contour(data = grid, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, z = as.numeric(k
nn_cluster)), color = "blue", linetype = "dashed") +
  ggtitle("k-NN Classification with Decision Boundaries") +
  scale_color_discrete(name = "Cluster") +
  theme_minimal()

# Next I will do the same for the Lower panel for my Bayes decision boundary utilzing a purple t
hem
p2 <- ggplot() +
```

```
  geom_point(data = healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = as.f
actor(kmeans_result$cluster)), size = 3) +
  geom_contour(data = grid, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, z = as.numeric(d
ecision_boundary)), color = "purple", linetype = "dotted") +
  ggtitle("Bayes Decision Boundary") +
  scale_color_discrete(name = "Cluster") +
  theme_minimal()

# Now I will combine p1 and p2
grid.arrange(p1, p2, ncol = 1)
```

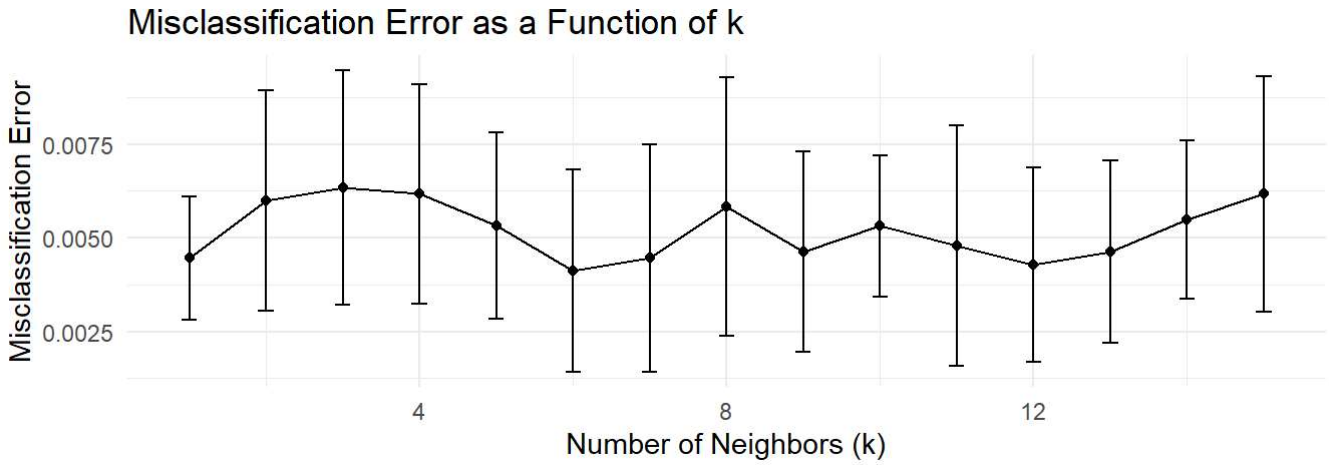


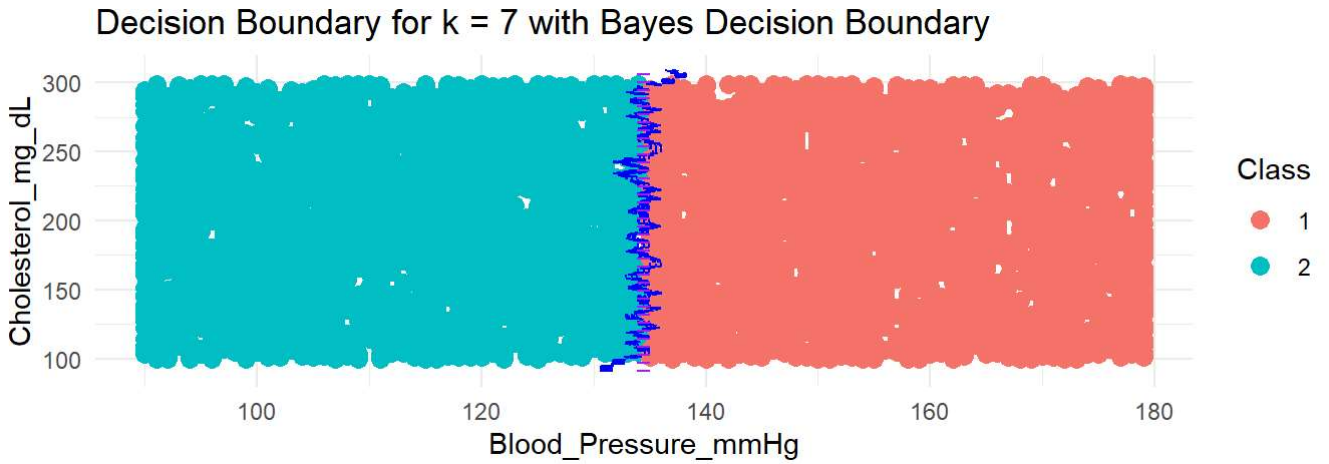

```
# Now I have added the caret function for better cross-validation for k-NN classification and be
tter Bayes decision boundary
library(cluster)
library(ggplot2)
library(class)
library(caret)
```

```
## Loading required package: lattice
```

```
library(MASS)
library(gridExtra)

# My data types need to represent a binary output so I created a  synthetic variable called 'Cla
ss' for binary classification when training my data
healthcare$Class <- as.factor(ifelse(healthcare$Blood_Pressure_mmHg > median(healthcare$Blood_Pr
essure_mmHg), 1, 2))

# To resolve the syntax errors I was receiving I research how to create a function to compute mi
s-classification error for the different values of k in my data from stackoverflow in order to t
rain my model better
calculate_knn_cv_error <- function(k) {
  train_control <- trainControl(method = "cv", number = 10)
  knn_model <- train(Class ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = healthcare, method
= "knn",
                     trControl = train_control, tuneGrid = data.frame(.k = k))
  return(knn_model$results)
}

# Now its time to look at the statistical analysis of my misclassification error for the differe
nt values of k ranging between (1 to 15) and utilizing rbind and lapply function
k_values <- 1:15
cv_results <- do.call(rbind, lapply(k_values, calculate_knn_cv_error))

# Now it is time to plot utilizing "Upper panel" for plotting my  misclassification error as a f
unction of k
p1 <- ggplot(cv_results, aes(x = k, y = 1 - Accuracy, ymin = 1 - (Accuracy + AccuracySD), ymax =
1 - (Accuracy - AccuracySD))) +
  geom_errorbar(width = 0.2) +
  geom_line() +
  geom_point() +
  labs(title = "Misclassification Error as a Function of k",
       x = "Number of Neighbors (k)",
       y = "Misclassification Error") +
  theme_minimal()

# Now for the "Lower panel" I decided to utilize the Decision boundary for k=7 instead of typica
l k=5 with the Bayes decision boundary as the data points were not visible because they were to
small on the "x" and "y" axis
k_optimal <- 7
grid <- expand.grid(Blood_Pressure_mmHg = seq(min(healthcare$Blood_Pressure_mmHg) - 10, max(heal
thcare$Blood_Pressure_mmHg) + 10, length.out = 100),
                    Cholesterol_mg_dL = seq(min(healthcare$Cholesterol_mg_dL) - 10, max(healthca
re$Cholesterol_mg_dL) + 10, length.out = 100))

knn_result <- knn(train = healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")],
                  test = grid,
                  cl = healthcare$Class,
                  k = k_optimal)

grid$knn_cluster <- as.factor(knn_result)
```

```
# Tome to take a look by estimating my data with the Bayes decision boundary utilizing the LDA
lda_model <- lda(Class ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = healthcare)
grid$decision_boundary <- predict(lda_model, grid)$class

# Now lets assign the plots with "Lower panel" for plotting the  decision boundary for k=7 with ggplot2 and assigning it to p2 with a size of 3 and setting it to numeric and giving it a title and some colors
p2 <- ggplot() +
  geom_point(data = healthcare, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, color = Class), size = 3) +
  geom_contour(data = grid, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, z = as.numeric(knn_cluster)), color = "blue", linetype = "dashed") +
  geom_contour(data = grid, aes(x = Blood_Pressure_mmHg, y = Cholesterol_mg_dL, z = as.numeric(decision_boundary)), color = "purple", linetype = "dotted") +
  ggtitle("Decision Boundary for k = 7 with Bayes Decision Boundary") +
  scale_color_discrete(name = "Class") +
  theme_minimal()

# Lets combine the plots to save space
grid.arrange(p1, p2, ncol = 1)
```





```
# Adding a new library function called "purr" for better mapping functions in my plot to match m
y LVQ in the "caret" package because when I created my "Class" variable for better binary classi
fications when splitting the data I was able to successully run the libraries at the same time w
hen setting my variables to a facto
library(cluster)
library(ggplot2)
library(class)
library(caret)
library(MASS)
library(purrr)
```

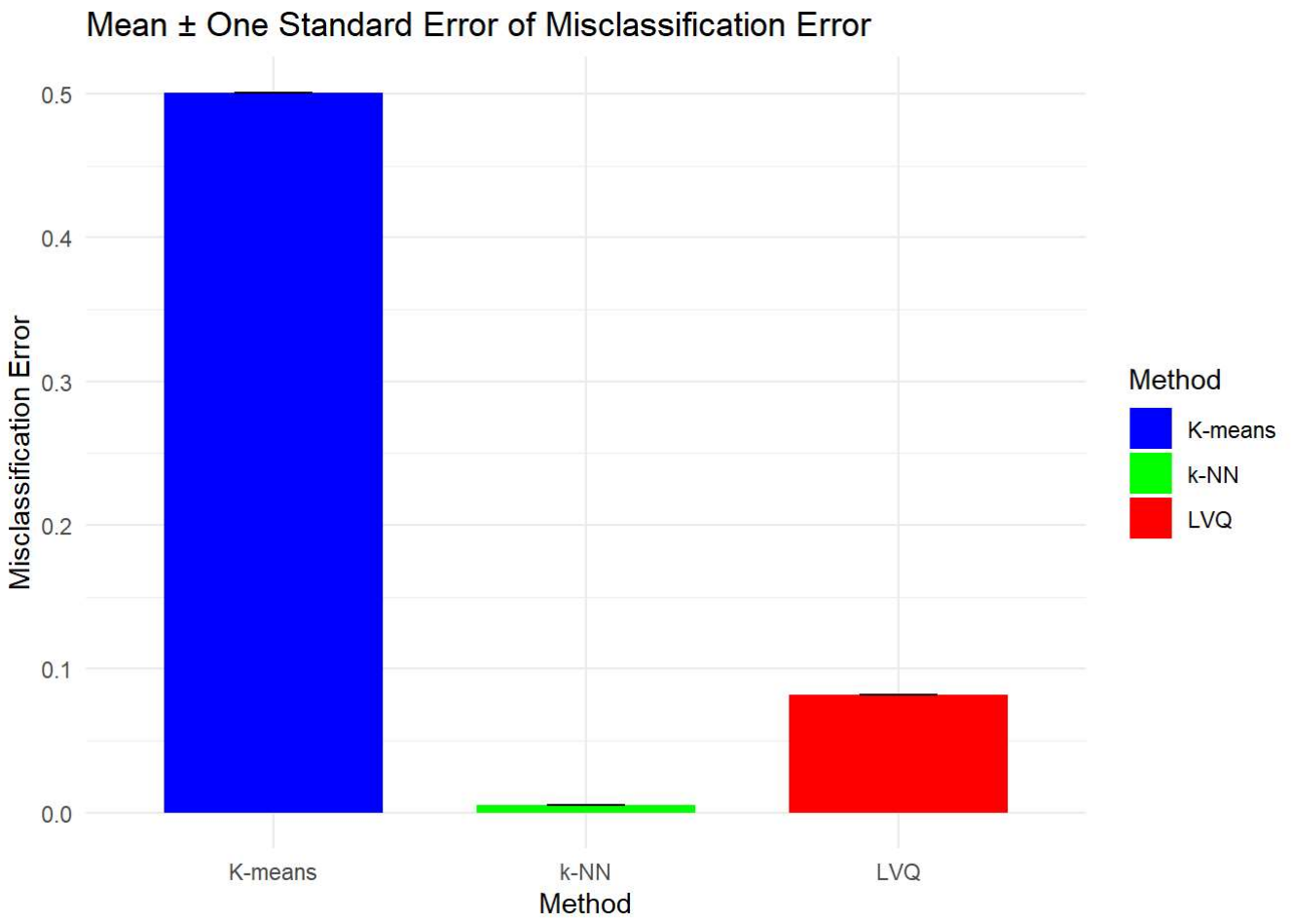
```
##
## Attaching package: 'purrr'
```

```
## The following object is masked from 'package:caret':
##
##     lift
```

```
## The following object is masked from 'package:mclust':
##
##     map
```

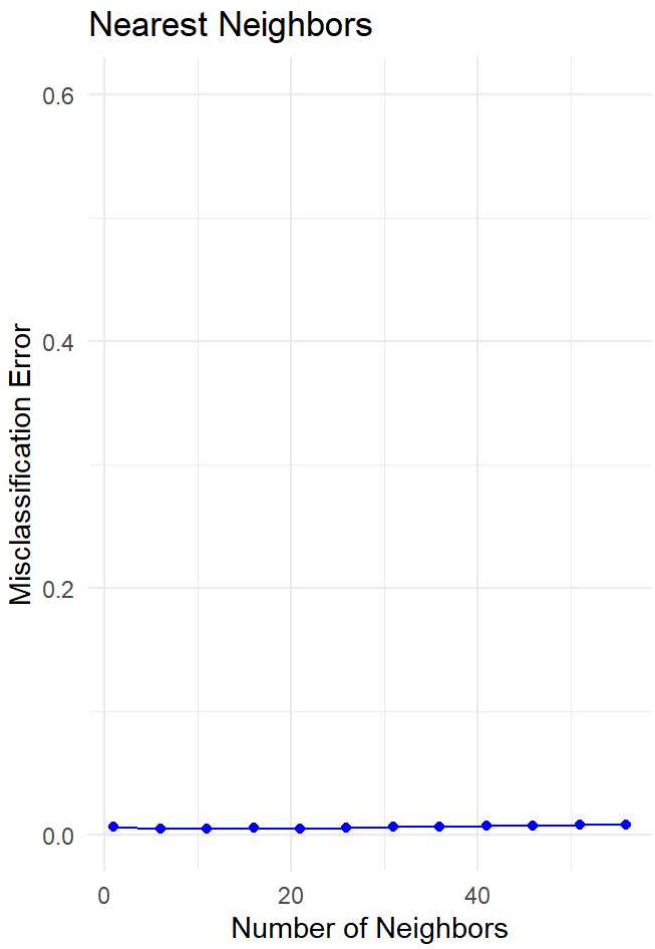
```
# Here is were I added my synthetic 'Class' variable
healthcare$Class <- as.factor(ifelse(healthcare$Blood_Pressure_mmHg > median(healthcare$Blood_Pr
essure_mmHg), 1, 2))

# Now lets name my fucntion calculate_errors in order to compute the  misclassification error ra
te for k-NN, K-means, and LVQ
calculate_errors <- function(data) {
  set.seed(123)

  # k-NN
  train_control <- trainControl(method = "cv", number = 10)
  knn_model <- train(Class ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = data, method = "kn
n",
                     trControl = train_control, tuneGrid = data.frame(.k = 7))
  knn_error <- 1 - max(knn_model$results$Accuracy)

  # K-means
  kmeans_result <- kmeans(data[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers = 2)
  kmeans_pred <- as.factor(ifelse(kmeans_result$cluster == 1, "1", "2"))
  kmeans_error <- mean(kmeans_pred != data$Class)

  # LVQ from the "caret" package
  lvq_model <- train(Class ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = data, method = "lv
q",
                     trControl = train_control)
  lvq_error <- 1 - max(lvq_model$results$Accuracy)

  return(c(knn_error, kmeans_error, lvq_error))
}

# I decided on just 10 calculations for the dataset
errors_realizations <- replicate(10, calculate_errors(healthcare))

# Now lets calculate the mean ± standard error rate for each method "k-NN, K-means, LVQ"
error_stats <- function(errors) {
  mean_errors <- rowMeans(errors)
  se_errors <- apply(errors, 1, sd) / sqrt(ncol(errors))
  return(data.frame(
    Method = c("k-NN", "K-means", "LVQ"),
    Mean = mean_errors,
    SE = se_errors
  ))
}

error_stats_combined <- error_stats(errors_realizations)

# Lets color code the results of my plot with "k-NN" = "green", "K-means" = "blue", "LVQ" = "re
d"
ggplot(error_stats_combined, aes(x = Method, y = Mean, fill = Method)) +
  geom_bar(stat = "identity", position = position_dodge(), width = 0.7) +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE), position = position_dodge(0.7), width =
0.25) +
```

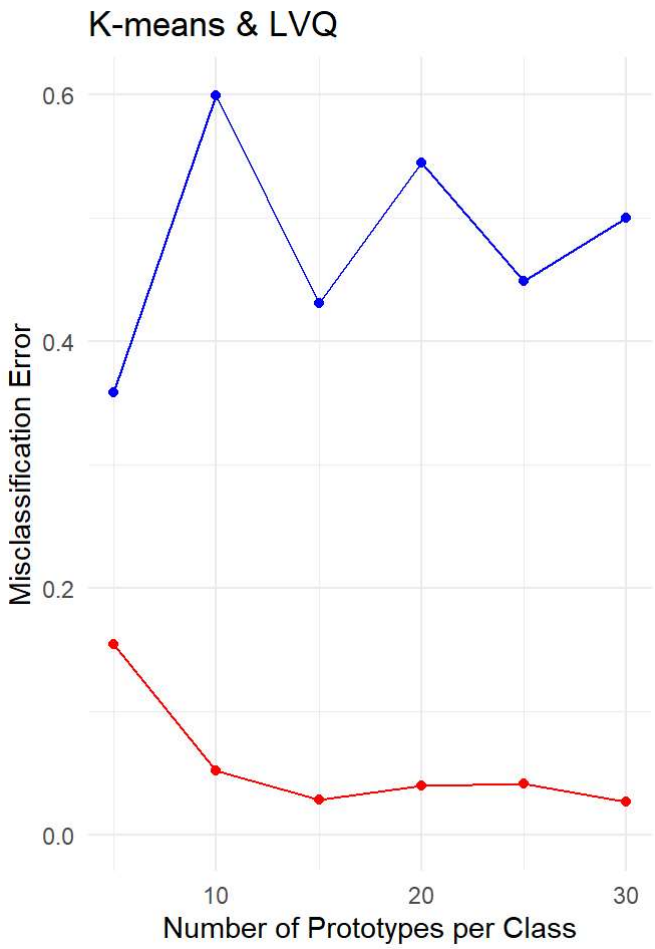
```
labs(title = "Mean ± One Standard Error of Misclassification Error",
     y = "Misclassification Error",
     x = "Method") +
scale_fill_manual(values = c("k-NN" = "green", "K-means" = "blue", "LVQ" = "red")) +
theme_minimal()
```



```
# The "caret" package and "class" package was necessary in order for the k-NN classification, LV
Q and K-means during my training of the dataset and ggplot2 for plotting the binary classificati
on when splitting the data
library(cluster)
library(ggplot2)
library(class)  # For k-NN classification
library(caret)  # For LVQ and K-means through training
library(MASS)
library(gridExtra)

#health$Class is the binary classification represented by 1s and 2s
healthcare$Class <- as.factor(ifelse(healthcare$Blood_Pressure_mmHg > median(healthcare$Blood_Pr
essure_mmHg), 1, 2))

# The function I created is assigned to "calculate_knn_errors" to calculate k-NN misclassificati
on errorsrates of my two variables
calculate_knn_errors <- function(data, k_values) {
  errors <- sapply(k_values, function(k) {
    knn_model <- train(Class ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = data, method = "k
nn",
                       trControl = trainControl(method = "cv", number = 10),
                       tuneGrid = data.frame(.k = k))
    return(1 - max(knn_model$results$Accuracy))
  })
  return(errors)
}

# The function I created is assigned to "calculate_kmeans_errors" to calculate k-NN misclassific
ation errors rates of my two variables
calculate_kmeans_errors <- function(data, proto_values) {
  errors <- sapply(proto_values, function(proto) {
    kmeans_result <- kmeans(data[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL")], centers = pro
to)
    cluster_assignment <- as.factor(ifelse(kmeans_result$cluster == 1, "1", "2"))
    return(mean(cluster_assignment != data$Class))
  })
  return(errors)
}

# I named the function "calculate_lvq_errors" to calculate LVQ misclassification errors
calculate_lvq_errors <- function(data, proto_values) {
  errors <- sapply(proto_values, function(proto) {
    lvq_model <- train(Class ~ Blood_Pressure_mmHg + Cholesterol_mg_dL, data = data, method = "l
vq",
                       trControl = trainControl(method = "cv", number = 10),
                       tuneGrid = data.frame(size = proto, k = 5))
    return(1 - max(lvq_model$results$Accuracy))
  })
  return(errors)
}

# I just played witht he results and decided to utilize random values for my test
```

```
k_values <- seq(1, 60, by = 5)
proto_values <- seq(5, 30, by = 5)

# Now lets calculate the errors
knn_errors <- calculate_knn_errors(healthcare, k_values)
kmeans_errors <- calculate_kmeans_errors(healthcare, proto_values)
lvq_errors <- calculate_lvq_errors(healthcare, proto_values)

# Plotting with ggplot2 and assigning colors to the points and lines
p1 <- ggplot(data.frame(k_values, knn_errors), aes(x = k_values, y = knn_errors)) +
  geom_line(color = "blue") +
  geom_point(color = "blue") +
  ggtitle("Nearest Neighbors") +
  xlab("Number of Neighbors") +
  ylab("Misclassification Error") +
  ylim(0, 0.6) +
  theme_minimal()

p2 <- ggplot(data.frame(proto_values, kmeans_errors, lvq_errors), aes(x = proto_values)) +
  geom_line(aes(y = kmeans_errors), color = "blue") +
  geom_point(aes(y = kmeans_errors), color = "blue") +
  geom_line(aes(y = lvq_errors), color = "red") +
  geom_point(aes(y = lvq_errors), color = "red") +
  ggtitle("K-means & LVQ") +
  xlab("Number of Prototypes per Class") +
  ylab("Misclassification Error") +
  ylim(0, 0.6) +
  theme_minimal()

# The grid function aligns the plots in a 1 by 2 view
grid.arrange(p1, p2, ncol = 2)
```


Nearest Neighbors
Misclassification Error
0.6
0.4
0.2
0.0
0
20
40
Number of Neighbors


K-means & LVQ
Misclassification Error
0.6
0.4
0.2
0.0
10
20
30
Number of Prototypes per Class

```
# I had to add "tidyr" package since I reinstalled r studio due to computer crashing due to lack of memory
library(class)
library(ggplot2)
library(tidyr)
library(gridExtra)

# I choose three variables
data <- healthcare[, c("Blood_Pressure_mmHg", "Cholesterol_mg_dL", "BMI", "Dosage_Med1_mg")]

# I had to double check to see if my variables were numeric and if not I converted from characte r to binary numeric format
data <- data.frame(lapply(data, function(x) as.numeric(as.character(x))))

# Data cleaning of missing values and replacing them with column mean
data <- data.frame(lapply(data, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x)))

# Based off my results I decided to utilize a fake 'LandUsage' class to help identify better
data$LandUsage <- factor(ifelse(data$Blood_Pressure_mmHg > median(data$Blood_Pressure_mmHg), 1, 2))

# Now I will predict with the k-NN classification to help with converting the matrix to a data f rame latter on because of my ggplot2 package was giving syntax errors
predicted_land_usage <- knn(train = data[, 1:4], test = data[, 1:4], cl = data$LandUsage, k = 5)

# matrix conversion
matrix_to_df <- function(mat, name) {
  df <- as.data.frame(mat)
  df$row <- 1:nrow(df)
  df <- pivot_longer(df, cols = -row, names_to = "col", values_to = name)
  df$col <- as.integer(gsub("V", "", df$col))
  return(df)
}

# Now I will convert the "actual and predicted land usage" of the data into matrices for better plotting and data visualization in order to calcuate the grid size and plot the actual land usag e
n <- sqrt(nrow(data))
df_actual <- matrix_to_df(matrix(as.numeric(data$LandUsage), nrow = n), "value")
```

```
## Warning in matrix(as.numeric(data$LandUsage), nrow = n): data length [5834] is
## not a sub-multiple or multiple of the number of rows [76]
```

```
df_predicted <- matrix_to_df(matrix(as.numeric(predicted_land_usage), nrow = n), "value")
```

```
## Warning in matrix(as.numeric(predicted_land_usage), nrow = n): data length
## [5834] is not a sub-multiple or multiple of the number of rows [76]
```

```
# Actual land usage plot and assigning p5
p5 <- ggplot(df_actual, aes(x = col, y = row, fill = as.factor(value))) +
  geom_tile() +
  scale_fill_manual(values = c("#FF0000", "#00FF00")) +
  ggtitle("Actual Land Usage") +
  theme_minimal() +
  theme(axis.text = element_blank(), axis.title = element_blank())

# Predicted land usage plot and assigning p6
p6 <- ggplot(df_predicted, aes(x = col, y = row, fill = as.factor(value))) +
  geom_tile() +
  scale_fill_manual(values = c("#FF0000", "#00FF00")) +
  ggtitle("Predicted Land Usage (5-NN)") +
  theme_minimal() +
  theme(axis.text = element_blank(), axis.title = element_blank())

# Lets assign p5, p6 side by side for better comparison
grid.arrange(p5, p6, ncol = 2)
```