# Cotiviti Healthcare

2024-09-18

```
# Libraries
library(caret)
```

```
## Loading required package: ggplot2
```

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

```
library(randomForest)
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
# Loading my data
data <- read.csv("C:\\Users\\jacob\\OneDrive\\Desktop\\R Studio Projects 2024\\Datasets\\healthc
are.csv")

# First I need to convert the categorical variables to factors to get it ready for training and
splitting the data
data$Gender <- as.factor(data$Gender)
data$Outcome <- as.factor(data$Outcome)

# Now it is time to split the data into training and testing sets
set.seed(42)
trainIndex <- createDataPartition(data$Outcome, p = 0.7, list = FALSE)
trainData <- data[trainIndex, ]
testData <- data[-trainIndex, ]

# Now lets train my Random Forest model of my healthcare data
set.seed(42)
rf_model <- train(Outcome ~ Age + Gender + Dosage_Med1_mg + Dosage_Med2_mg + Days_On_Treatment,
                  data = trainData,
                  method = "rf",
                  trControl = trainControl(method = "cv", number = 5))

# Now I need to predict on test data
rf_predictions <- predict(rf_model, testData)

# Lets take a look at the consfusion matrix
conf_matrix <- confusionMatrix(rf_predictions, testData$Outcome)
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Improved No Change Worsened
##   Improved       797       473      337
##   No Change       61        31       22
##   Worsened        13         8        7
##
## Overall Statistics
##
##                Accuracy : 0.4774
##                  95% CI : (0.4538, 0.5011)
##     No Information Rate : 0.498
##     P-Value [Acc > NIR] : 0.9596
##
##                   Kappa : -0.005
##
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: Improved Class: No Change Class: Worsened
## Sensitivity                  0.91504          0.06055        0.019126
## Specificity                  0.07745          0.93290        0.984816
## Pos Pred Value               0.49596          0.27193        0.250000
## Neg Pred Value               0.47887          0.70581        0.791400
## Prevalence                   0.49800          0.29274        0.209262
## Detection Rate               0.45569          0.01772        0.004002
## Detection Prevalence         0.91881          0.06518        0.016009
## Balanced Accuracy            0.49624          0.49672        0.501971
```

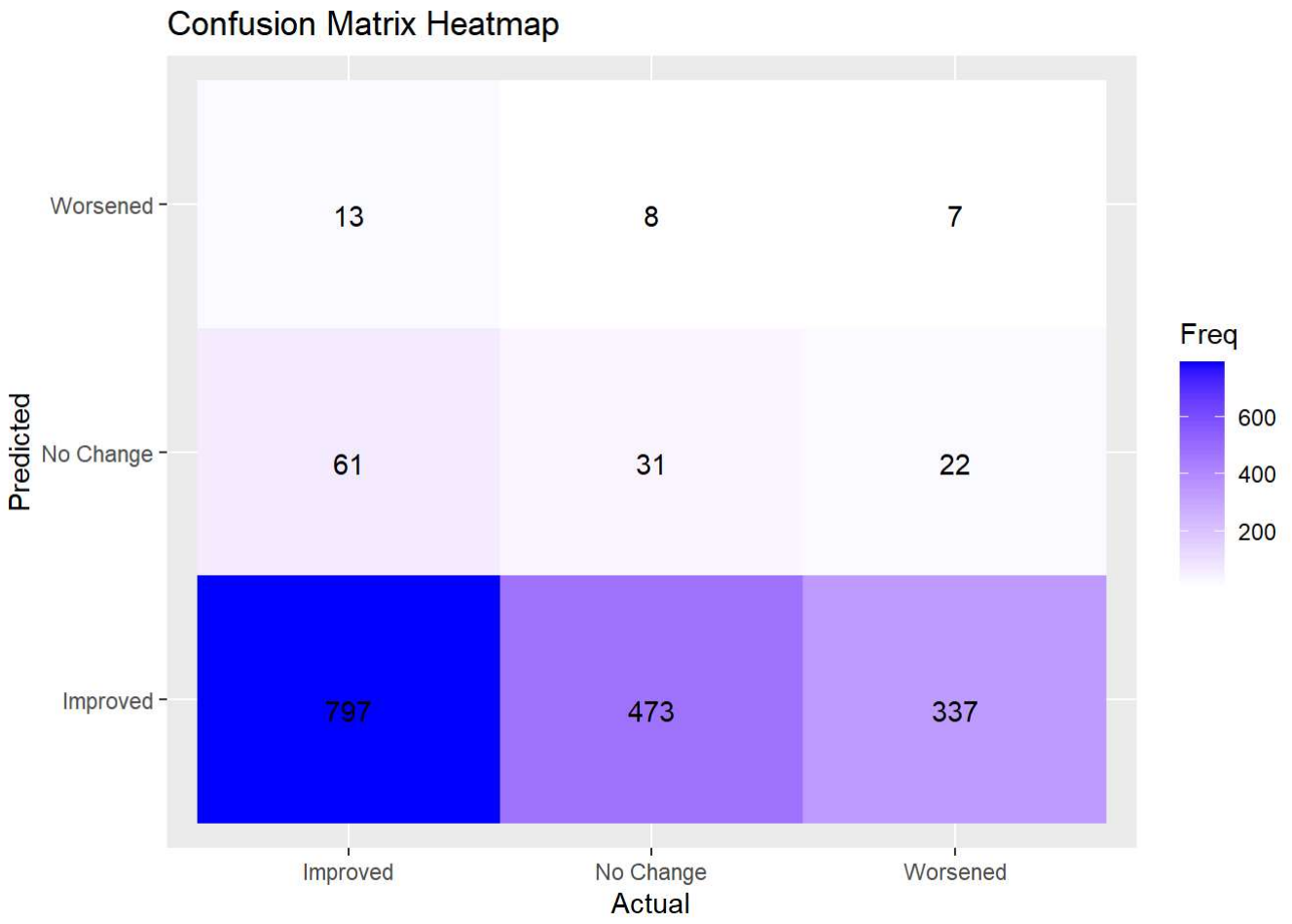
```
# Now I want to see it in ggplot2 to get a better understanding
conf_matrix <- confusionMatrix(rf_predictions, testData$Outcome)
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Improved No Change Worsened
##   Improved       797       473      337
##   No Change       61        31       22
##   Worsened        13         8        7
##
## Overall Statistics
##
##                Accuracy : 0.4774
##                  95% CI : (0.4538, 0.5011)
##     No Information Rate : 0.498
##     P-Value [Acc > NIR] : 0.9596
##
##                   Kappa : -0.005
##
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: Improved Class: No Change Class: Worsened
## Sensitivity                  0.91504          0.06055        0.019126
## Specificity                  0.07745          0.93290        0.984816
## Pos Pred Value               0.49596          0.27193        0.250000
## Neg Pred Value               0.47887          0.70581        0.791400
## Prevalence                   0.49800          0.29274        0.209262
## Detection Rate               0.45569          0.01772        0.004002
## Detection Prevalence         0.91881          0.06518        0.016009
## Balanced Accuracy            0.49624          0.49672        0.501971
```

```
# I was having issues so I just load the library just in case for knitting purposes
library(ggplot2)

# Now I need to convert the confusion matrix to a data frame becauase it will not configure corr
ectly
conf_df <- as.data.frame(conf_matrix$table)

# Time for a plot in ggplot2 with a level of frequency to show the variations in colors to make
the confusion matrix easier to visualize for novice users
ggplot(data = conf_df, aes(x = Reference, y = Prediction, fill = Freq)) +
  geom_tile() +
  geom_text(aes(label = Freq), vjust = 1) +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "Confusion Matrix Heatmap", x = "Actual", y = "Predicted")
```




```
# My linear regression model
model <- train(Days_On_Treatment ~ Age + Dosage_Med1_mg + Dosage_Med2_mg + Social_Interaction_Sc
ore,
               data = trainData,
               method = "lm")

# Let begin with predicting the test data first
predictions <- predict(model, testData)

# This will give myself a better feel for the data
results <- postResample(predictions, testData$Days_On_Treatment)
print(results)
```

```
##         RMSE     Rsquared          MAE
## 1.048782e+02 2.783348e-04 9.109412e+01
```

```
# Now it is time to see if the model is significant
model <- lm(Days_On_Treatment ~ Age + Dosage_Med1_mg + Dosage_Med2_mg + Social_Interaction_Scor
e, data = trainData)

# I begin with a summary of the model
summary(model)
```

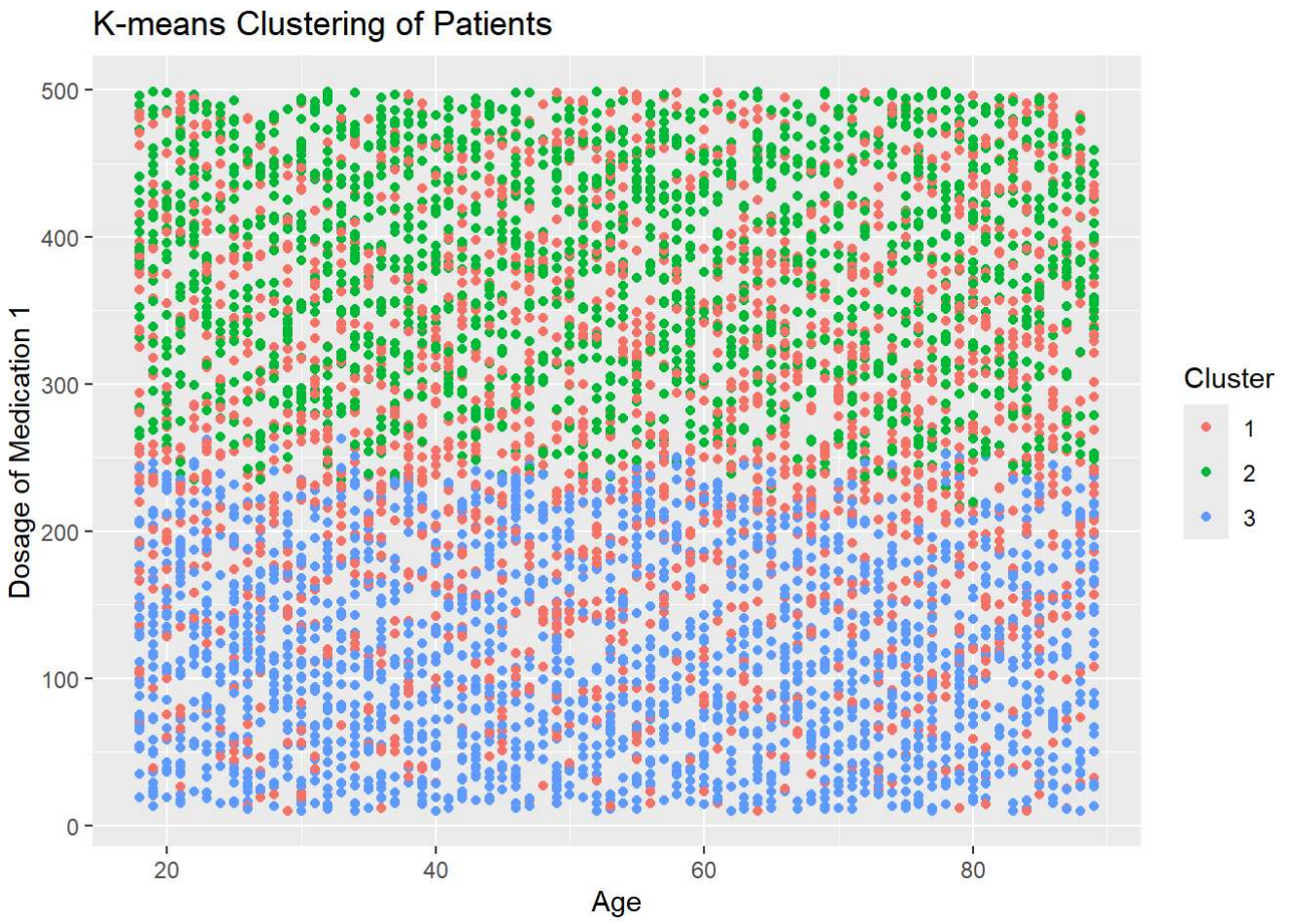
```
##
## Call:
## lm(formula = Days_On_Treatment ~ Age + Dosage_Med1_mg + Dosage_Med2_mg +
##     Social_Interaction_Score, data = trainData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -189.721  -91.927    0.659   90.946  187.988
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              173.841548   6.743021  25.781   <2e-16 ***
## Age                        0.047466   0.079360   0.598   0.5498
## Dosage_Med1_mg            -0.002237   0.011629  -0.192   0.8475
## Dosage_Med2_mg             0.019421   0.011748   1.653   0.0984 .
## Social_Interaction_Score   0.082860   0.056731   1.461   0.1442
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 105.1 on 4080 degrees of freedom
## Multiple R-squared:  0.001314,   Adjusted R-squared:  0.0003346
## F-statistic: 1.342 on 4 and 4080 DF,  p-value: 0.2519
```

```
# Now lets scale the data down to size
scaled_data <- scale(data[, c("Age", "Dosage_Med1_mg", "Dosage_Med2_mg", "Social_Interaction_Sco
re")])

# I will utilize K-means clustering
set.seed(123)
kmeans_result <- kmeans(scaled_data, centers = 3)

# Next I need to add the cluster assignments to the data as a factor
data$Cluster <- factor(kmeans_result$cluster)

# I was having problems knitting to pdf so I just used the ggplot2 library to make sure to Visua
lize the clusters from the data and I used geom_point because the dots were to huge
library(ggplot2)
ggplot(data, aes(x = Age, y = Dosage_Med1_mg, color = Cluster)) +
  geom_point() +
  labs(title = "K-means Clustering of Patients", x = "Age", y = "Dosage of Medication 1")
```




```
# My libraries but I could haved used more but do to time constraints
library(readxl)
library(dplyr)
```

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

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

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

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

```
library(anomalize)
library(tibbletime)
```

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

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

```
library(ggplot2)

# Loading my data from my excel file but first I set the working directory
healthcare <- read_excel("C:/Users/jacob/OneDrive/Desktop/R Studio Projects 2024/Datasets/healthcare.xlsx")

# I decided to add the Date column by months in a numerical format as I was having trouble fitting all the months on the plot so the data represents monthly records starting from January 2020 so each number represents a month, 12 moinths in a year is simply 12 counting from January 2020.
healthcare <- healthcare %>%
  mutate(Date = seq(as.Date("2020-01-01"), by = "month", length.out = n()))

# My table need to be converted using the tibble library packed I implemented prior to calculate the time for the months.
healthcare_tbl <- as_tibble(healthcare)

# Convert to a time-based tibble
healthcare_time_tbl <- healthcare_tbl %>%
  as_tbl_time(index = Date)

# Now its is time to perform my anomaly detection on the 'Blood_Pressure_mmHg' column
anomaly_detection <- healthcare_time_tbl %>%
  time_decompose(Blood_Pressure_mmHg, method = "stl", frequency = 12, trend = 60) %>%
  anomalize(remainder, method = "iqr") %>%
  time_recompose()
```

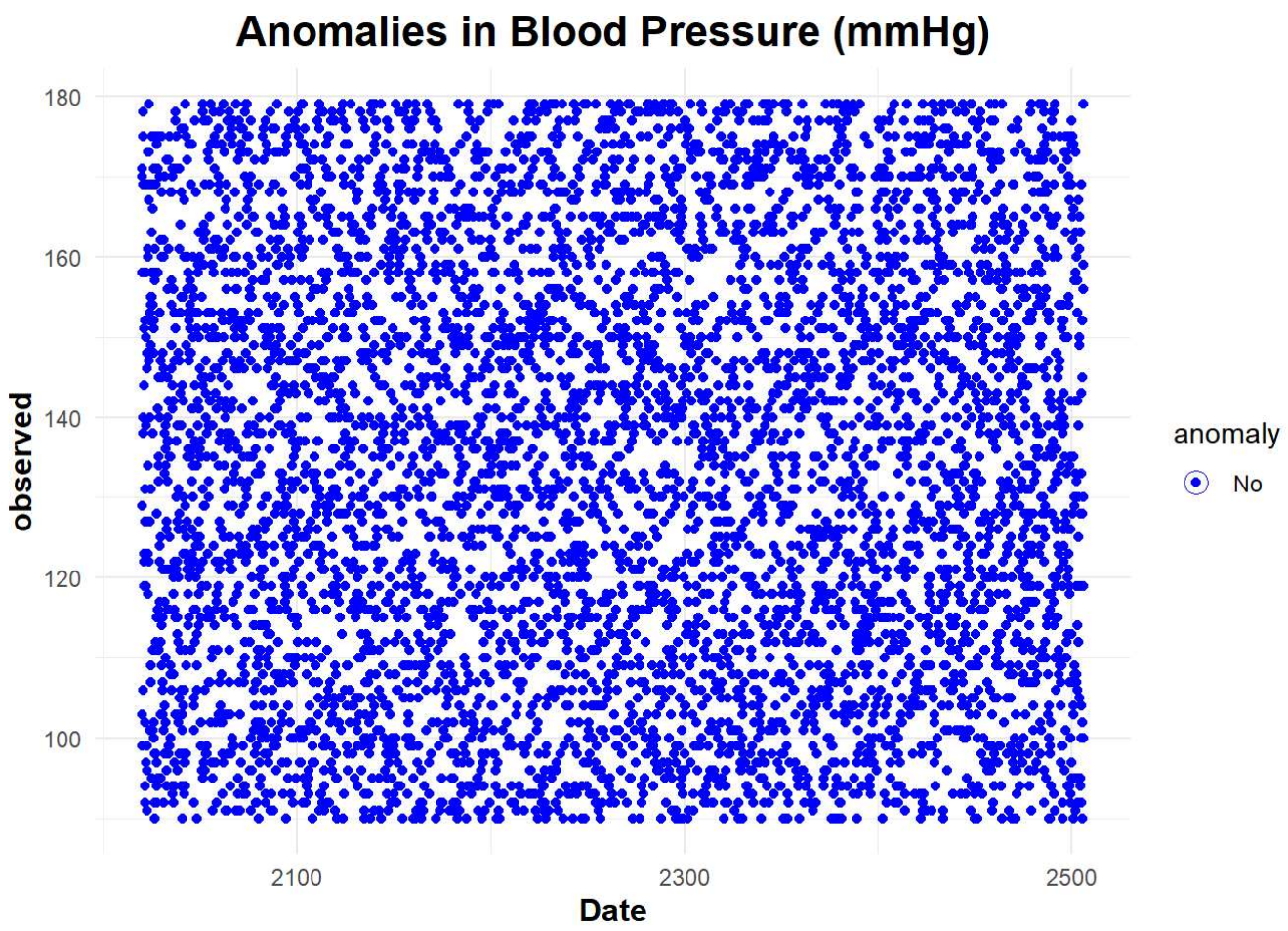
```
## frequency = 12 months
```

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

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

```
# I used the colors of red and blue to see if I could see anythin in my plot for various anomali
es for my color scheme
plot_anomaly <- plot_anomalies(anomaly_detection,
               color_no = "blue",    # Normal data points
               color_yes = "red",    # Anomalies
               fill_ribbon = "lightblue") +
  theme_minimal() +
  ggtitle("Anomalies in Blood Pressure (mmHg)") +  # Add title using ggtitle
  theme(plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
        axis.title.x = element_text(size = 12, face = "bold"),
        axis.title.y = element_text(size = 12, face = "bold"))

# Lets take a peak
print(plot_anomaly)
```




```
#Extra credit: I was just curious what it would look like as a 3d plot: Here is the html link to
my plot #####https://plotly.com/~averymarcellusfalumni2022/3/#/
library(plotly)  # 3D plotting
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

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

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(cluster)  # clustering algorithms
library(ggplot2)  # plotting
library(mclust)   # Gaussian Mixture Model and EM algorithm
```

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

```
library(MASS)     # Bayes decision boundary
```

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

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

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

```
# I had to first set Up my Plotly Credentials in R Studio
# By creating a Plotly username and API key from the website
Sys.setenv("plotly_username" = "averymarcellusfalumni2022")
Sys.setenv("plotly_api_key" = "VYKPa2fBMcdjXY718C6d")

# Next I had to make sure my data was properly formatted. #'healthcare' with 'Blood_Pressure_mmH
g' and 'Cholesterol_mg_dL'
# My data is already loaded in the environment

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

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

# Clustering time for plotting in 3D
healthcare$kmeans_cluster <- as.factor(kmeans_result$cluster)
healthcare$gmm_cluster <- as.factor(gmm_result$classification)

# 3D Plot for K-means clustering with Plotly but I need to make sure "z" is numeric because cate
rgorical variables is most of the time binary and will not work with my "x" and "y" numerical va
riables. I mean I could convert but I am running out of time researching the syntac errors on st
ackoverflow.com
kmeans_plot <- plot_ly(
  healthcare,
  x = ~Blood_Pressure_mmHg,
  y = ~Cholesterol_mg_dL,
  z = ~as.numeric(kmeans_cluster),
  type = 'scatter3d',
  mode = 'markers',
  marker = list(size = 4, color = ~as.numeric(kmeans_cluster), colorscale = 'Viridis')
) %>%
  layout(
    scene = list(
      xaxis = list(title = 'Blood Pressure (mmHg)'),
      yaxis = list(title = 'Cholesterol (mg/dL)'),
      zaxis = list(title = 'K-means Cluster')
    ),
    title = "3D K-means Clustering"
  )

# Now it is time to use 3D plotting for GMM clustering statistical analysis but once again "z" n
eeds to be numerical
gmm_plot <- plot_ly(
  healthcare,
  x = ~Blood_Pressure_mmHg,
  y = ~Cholesterol_mg_dL,
  z = ~as.numeric(gmm_cluster),
  type = 'scatter3d',
```

```
  mode = 'markers',
  marker = list(size = 4, color = ~as.numeric(gmm_cluster), colorscale = 'Viridis')
) %>%
  layout(
    scene = list(
      xaxis = list(title = 'Blood Pressure (mmHg)'),
      yaxis = list(title = 'Cholesterol (mg/dL)'),
      zaxis = list(title = 'GMM Cluster')
    ),
    title = "3D GMM Clustering"
  )

# Next I created a "api_create" function I researched online for my plot to be public and a url link to access to play around with the plot

# Time to upload my K-means results to my plot with my Plotly account but I must set it to TRUE (set public = TRUE) because setting it to FALSE will result in my plot being private and we do n ot want that.
kmeans_link <- api_create(kmeans_plot, filename = "Kmeans_Clustering_3D", sharing = "public")
```

```
## Found a grid already named: 'Kmeans_Clustering_3D Grid'. Since fileopt='overwrite', I'll try to update it
```

```
## Found a plot already named: 'Kmeans_Clustering_3D'. Since fileopt='overwrite', I'll try to up date it
```

```
# Time to upload my GMM plot to my Plotly account by once again setting it to TRUE (set public = TRUE)
gmm_link <- api_create(gmm_plot, filename = "GMM_Clustering_3D", sharing = "public")
```

```
## Found a grid already named: 'GMM_Clustering_3D Grid'. Since fileopt='overwrite', I'll try to update it
```

```
## Found a plot already named: 'GMM_Clustering_3D'. Since fileopt='overwrite', I'll try to updat e it
```

```
# I am so excited I can't wait to see the results, Time to print the URL to access my plots to a nalysis the results
print(kmeans_link)
print(gmm_link)
```