**Performance Assessment: Task 1**

**A1. Proposal of Question** My research question for this performance assessment is, “Can the k-nearest neighbors (KNN) classification method be used to predict hospital readmissions?”

**A2. Defined Goal**

The goal of this analysis is to develop a machine learning model using KNN to help the hospital identify patients that are at risk of readmitting to the hospital.

**B1. Explanation of Classification Method**

The KNN method classifies an unknown data point by examining the ‘k’ closest labeled data points, with ‘k’ being the number of labeled data points near the unknown and using them to predict how the unknown data point should be classified. The unknown data point is typically assigned the label of the points with the nearest standard Euclidean distance to the unknown.

**B2. Summary of Method Assumption**

One assumption of the KNN method is that the data is in a feature space that is represented in a quantitative or categorical manner. This implies that the space can be measured using a distance metric such as Euclidean distance.

**B3. Packages or Libraries List**

For this assignment, I chose to use the R coding language via R Studio. The *readr* package was used to load the data into the data frame. The *dplyr* package was used for the glimpse function, which allowed me to get a quick summary of the full data set. The *corrr* package was used to visualize the correlation between my independent variables. The *ggplot2* package was also used for correlation. The *caret* package was used to split the data. The *class* package was used for KNN. Lastly, the *pROC* package was used to plot the ROC curve and calculate the AUC score.

**C1. Data Preprocessing**

One data preprocessing goal that was used for this analysis was to encode categorical variables to numeric. This is a necessary step for analysis as KNN requires numeric values. Since the readmission field was Yes/No, I was able to encode the values as 1/0.

**C2. Data Set Variables**

|  |  |
| --- | --- |
| Readmis | Categorical |
| Children | Numeric |
| Age | Numeric |
| Income | Numeric |
| VitD\_levels | Numeric |
| Doc\_visits | Numeric |
| Full\_meals\_eaten | Numeric |
| vitD\_supp | Numeric |
| Initial\_days | Numeric |
| TotalCharge | Numeric |
| Additional\_charges | Numeric |
| Timely\_treatment | Numeric |
| Active\_listening | Numeric |

**C3. Steps for Analysis**

The first step in preparing the data was to import the necessary libraries for analysis. Next, I loaded the raw medical data set into the data frame. I chose to rename the survey response variables for easier understanding. I assessed for duplicates, nulls. I assessed for outliers using histograms. After that, I encoded the categorical variable in my initial data set as 1/0. I created a new data frame with the initial variables I would be using in my analysis. I created a visualization of the correlation between the independent variables and chose to drop TotalCharges. I visualized the correlation again and kept the remaining variables since all correlation values were under my threshold of 0.75. I normalized my numeric variables and converted the target variable to numeric. Finally, I exported the cleaned data set to a CSV file.

See below for data preprocessing code:

#Import necessary packages

library(readr)

library(dplyr)

library(ggplot2)

library(corrr)

library(caret)

library(class)

library(pROC)

#Load the data

med <- read\_csv("WGU/D209\_medical\_raw.csv")

# Get summary of data

str(med)

#Rename survey response variables

colnames(med)[colnames(med) == 'Item1'] <- 'Timely\_admission'

colnames(med)[colnames(med) == 'Item2'] <- 'Timely\_treatment'

colnames(med)[colnames(med) == 'Item3'] <- 'Timely\_visits'

colnames(med)[colnames(med) == 'Item4'] <- 'Reliability'

colnames(med)[colnames(med) == 'Item5'] <- 'Options'

colnames(med)[colnames(med) == 'Item6'] <- 'Hours\_of\_treatment'

colnames(med)[colnames(med) == 'Item7'] <- 'Courteous\_staff'

colnames(med)[colnames(med) == 'Item8'] <- 'Active\_listening'

#Assess for duplicates

sum(duplicated(med))

#Assess for nulls

sum(is.na(med))

#Assess for outliers with histograms for numeric variables

par(mar=c(1,1,1,1))

hist(med$Children)

hist(med$Age)

hist(med$Income)

hist(med$VitD\_levels)

hist(med$Doc\_visits)

hist(med$Full\_meals\_eaten)

hist(med$vitD\_supp)

hist(med$Initial\_days)

hist(med$TotalCharge)

hist(med$Additional\_charges)

hist(med$Timely\_treatment)

hist(med$Active\_listening)

#Encode categorical dependent variable as numeric

med$ReAdmis[med$ReAdmis == 'Yes'] <- 1

med$ReAdmis[med$ReAdmis == 'No'] <- 0

#New data frame with KNN variables

med\_reduced <- med[, c("ReAdmis", "Children", "Age", "Income", "VitD\_levels",

"Doc\_visits", "Full\_meals\_eaten", "vitD\_supp",

"Initial\_days", "TotalCharge", "Additional\_charges",

"Timely\_treatment", "Active\_listening")]

#Visualize correlation

med\_reduced %>%

dplyr::select(where(is.numeric)) %>%

correlate() %>%

shave() %>%

rplot(print\_cor = TRUE) +

theme(axis.text.x = element\_text(angle = 90, hjust = 1))

A screenshot of a graph

Description automatically generated

#Correlation between Additional charges and Age under 0.75 threshold, keep

#Correlation between Total charge & Initial days over 0.75, dropping Total charge

med\_reduced <- subset(med\_reduced, select = -c(TotalCharge))

#Confirm drop

str(med\_reduced)

#Visualize correlation again

med\_reduced %>%

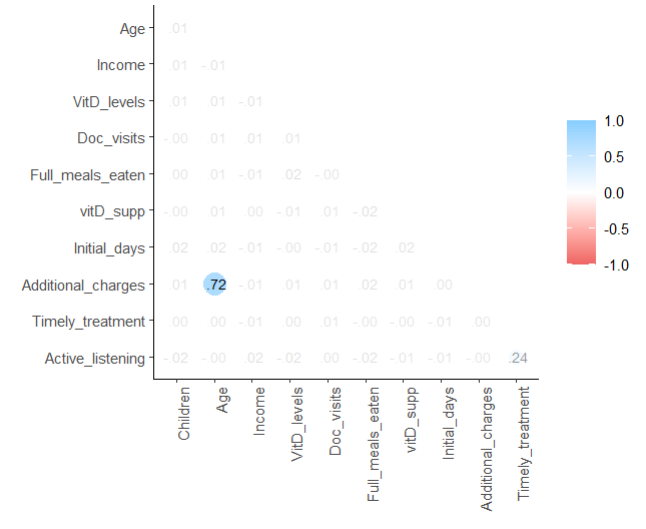
dplyr::select(where(is.numeric)) %>%

correlate() %>%

shave() %>%

rplot(print\_cor = TRUE) +

theme(axis.text.x = element\_text(angle = 90, hjust = 1))



#Normalize and scale the data, convert to numeric

normalize <- function(x) {

return((x-min(x))/(max(x)-min(x)))

}

med\_norm <- as.data.frame(sapply(med\_reduced, function(x) if(is.numeric(x)) normalize(x) else x))

med\_clean <- as.data.frame(lapply(med\_norm, function(x) if(is.character(x)) as.numeric(x) else x))

#Confirm data set normalized

str(med\_clean)

glimpse(med\_clean)

#Export cleaned data set

write\_csv(med\_clean, "WGU/D209\_T1\_MV\_clean.csv")

**C4. Cleaned Data Set**

See ‘D209\_T1\_MV\_clean’ file for cleaned data set.

**D1. Splitting the Data**

See below for data split code:

#Set seed to reproduce same split

set.seed(123)

#Create indices

train\_index <- createDataPartition(med\_clean$ReAdmis, p=0.8, list=FALSE)

#Create the training and test data sets

train\_set <-med\_clean[train\_index, ]

test\_set <- med\_clean[-train\_index, ]

#Further split data (y as data frame so I can export to CSV)

x\_train <- train\_set[, -which(names(train\_set) %in% "ReAdmis")]

y\_train <- as.data.frame(train\_set$ReAdmis)

x\_test <- test\_set[, -which(names(test\_set) %in% "ReAdmis")]

y\_test <- as.data.frame(test\_set$ReAdmis)

#Export as training & testing data

write\_csv(x\_train, "WGU/D209\_T1\_MV\_xtrain.csv")

write\_csv(x\_test, "WGU/D209\_T1\_MV\_xtest.csv")

write\_csv(y\_train, "WGU/D209\_T1\_MV\_ytrain.csv")

write\_csv(y\_test, "WGU/D209\_T1\_MV\_ytest.csv")

See the following files for training and test data sets:

1. ‘D209\_T1\_MV\_xtrain’
2. ‘D209\_T1\_MV\_xtest’
3. ‘D209\_T1\_MV\_ytrain’
4. ‘D209\_T1\_MV\_ytest’

**D2. Calculations**

I did not perform any intermediate calculations in this analysis.

**D3. Code Execution**

See below for KNN model code:

# Convert y back to vector

y\_train <- as.vector(unlist(y\_train))

y\_test <- as.vector(unlist(y\_test))

# Calculate k-value

n <- nrow(train\_set)

k <- sqrt(n)

print(k)



#Fit KNN models

knn\_89 <- knn(train=x\_train, test=x\_test, cl=y\_train, k=89, prob=TRUE)

knn\_90 <- knn(train=x\_train, test=x\_test, cl=y\_train, k=90, prob=TRUE)

#Confusion matrix

cfm\_89 <- table(knn\_89, y\_test)

cfm\_89

A close-up of numbers

Description automatically generated

cfm\_90 <- table(knn\_90, y\_test)

cfm\_90

A close-up of numbers

Description automatically generated

#Accuracy

acc\_89 <- mean(y\_test == knn\_89)

acc\_89



acc\_90 <- mean(y\_test == knn\_90)

acc\_90



#Create y\_test\_numeric for ROC/AUC

y\_test\_numeric <- ifelse(y\_test == "1", 1, 0)

#Plot ROC/AUC for knn\_89

probs\_89 <- attr(knn\_89, "prob")

readmit\_prob\_89 <- ifelse(knn\_89 == "1", probs\_89, 1-probs\_89)

roc\_89 <- roc(response=y\_test\_numeric, predictor=readmit\_prob\_89)

plot(roc\_89, main="ROC Curve for knn\_89")

A graph of a function

Description automatically generated

auc(roc\_89)



#Plot ROC/AUC for knn\_90

probs\_90 <- attr(knn\_90, "prob")

readmit\_prob\_90 <- ifelse(knn\_90 == "1", probs\_90, 1-probs\_90)

roc\_90 <- roc(response=y\_test\_numeric, predictor=readmit\_prob\_90)

plot(roc\_90, main="ROC Curve for knn\_90")

A graph of a function

Description automatically generated

auc(roc\_90)



**E1. Accuracy and AUC**

The k-value of my training set was calculated as 89.44, so I chose to run two KNN models where k=89 and k=90. The accuracy of both models was 0.90, which suggests that both models can accurately predict readmissions. The AUC score of both models was about 1, which also suggests that the models can predict readmissions.

**E2. Results and Implications**

In conclusion, readmissions can be accurately predicted using the variables included in these KNN models. In addition, the results of the confusion matrix show that the models include zero false negatives in their predictions.

**E3. Limitation**

One limitation of this analysis was that there were false positives shown in the confusion matrix.

**E4. Course of Action**

I recommended expanding the analysis to include all variables in the data set, which may lead to a different set of final variables. This could reduce the number of false positives in the confusion matrix.

**F. Panopto Recording**

Link to video included with submission.

**G. Sources for Third-Party Code**

**H. Sources**