Heart Failure Prediction Analysis using Statistical methods

Group Members:

Kanimozhi Subramanian (T00708068)

Keerthana Senthilkumar (T00711122)

Dr. Mateen Shaikh

STAT 5320 - Applied Stats in Data Mining

Department of Mathematics and Statistics

Thompson Rivers University

April 06, 2023

Table of Contents

Introduction

Background and Context

One of the most prevalent illnesses is heart disease (HD), and many healthcare professionals believe that early diagnosis of HD is essential to preventing and saving lives for their patients. The number one cause of death in the world is heart illness.

By using Linear Regression techniques to anticipate existing diseases like coronary heart disease, the health sector can help with early decision-making. Some common risk factors that may be included in a linear regression model for predicting heart disease include age, gender, blood pressure, cholesterol levels, and body mass index (BMI).

Hence, we will use the "heart.csv" dataset from Kaggle to conduct Prediction analysis for early detection and treatment. We will try to predict the future risk of a heart attack in those 918 samples of the dataset.

Objective

The primary goal of this study is to use statistical techniques to estimate the chance of a heart attack in a patient based on health factors like age, gender, chest pain, blood pressure, sugar level, cholesterol amount, etc.

Logistic Regression

The heart disease prediction analysis was performed using logistic regression, which is a statistical method used to model the probability of a binary response variable based on one or more predictor variables. In this case, the response variable was the presence or absence of heart disease, and the predictor variables were various clinical and demographic characteristics, including age, sex, cholesterol level, resting blood pressure, fasting blood sugar level, resting electrocardiogram results, and exercise-induced ST segment depression.

Data pre-processing

Removing missing values: The na.omit function was used to remove rows with missing values from both the training and testing datasets.We checked to see if any values were missing; if so, we removed the affected rows in order to avoid errors.Rows with a Cholesterol value of 0 were removed from the dataset.This was likely done because a cholesterol value of 0 is not realistic and may indicate missing data.

Splitting the data into training and testing sets: The createDataPartition function from the caret package was used to randomly split the data into a training set (50% of the original data) and a testing set (the remaining 50%). This allows for the model to be trained on a subset of the data and tested on an independent subset to evaluate its performance.

Scaling the features: The pre-process function from the caret package can be used to scale the features in the dataset to have zero mean and unit variance. Scaling can help to prevent features with large values from dominating the model and skewing the results.

Overall, these pre-processing steps are important for preparing the data for analysis and can help to improve the accuracy of the machine learning model.

Analysis

- The logistic regression model was fitted to the training data using the glm() function from the stats package. We have used variables Cholestrol, Sex,Age,Fasting Blood Sugar, Resting Blood Pressure as a predictors for predicting the heart disease for increased accuracy.

- The model was then used to make predictions on the testing set, and the predicted probabilities were converted to binary class labels using a threshold of 0.6.

- Heart disease is more likely to occur if the likelihood is greater than 0.6 and less likely if the probability is less than 0.6.

- The performance of the model was evaluated using a confusion matrix, which compares the predicted class labels to the true class labels in the testing set.

- The output displays the estimated coefficients of the 53% accurate logistic regression model that was constructed using our training set of data.

- The coefficients represent the log-odds of the probability of having heart disease (the outcome variable) associated with each predictor variable.

Result

The output shows the coefficients estimated by a logistic regression model, along with their standard errors, and corresponding p-values. The model is predicting the probability of a binary outcome (e.g., the presence of a disease) based on several predictor variables.

The coefficients indicate the direction and strength of the relationship between each predictor variable and the outcome, while holding all other variables constant.

- The Intercept represents the log odds of the outcome of female patients when all predictor variables are zero.

- Cholesterol has a positive coefficient of 0.2054, which means that higher levels of cholesterol are associated with higher log odds of the outcome, but the p-value (0.0925) suggests this association is not statistically significant at the 0.05 level.

- SexMale has a positive coefficient of 1.7099, which means that being male is associated with higher log odds of the outcome, and the low p-value (1.11e-07) suggests this association is statistically significant.

- Age has a positive coefficient of 0.5957, which means that older age is associated with higher log odds of the outcome, and the low p-value (3.23e-06) suggests this association is statistically significant.

- FastingBlood Sugar has a positive coefficient of 0.2381, which means that higher levels of fasting blood sugar are associated with higher log odds of the outcome, but the p-value (0.0588) suggests this association is not statistically significant at the 0.05 level.

- RestingBlood Pressure has a positive coefficient of 0.1772, which means that higher levels of resting blood pressure are associated with higher log odds of the outcome, but the p-value (0.1527) suggests this association is not statistically significant at the 0.05 level.

- The p-values indicate the probability of observing a z-value as extreme or more extreme than the one observed, assuming the null hypothesis that the coefficient is zero. If the p-value is less than the chosen significance level (e.g., 0.05), then the coefficient is considered statistically significant and unlikely to be due to chance. In this case, the variables SexMale and Age have statistically significant coefficients, while Cholesterol, FastingBlood Sugar, and RestingBlood Pressure do not.

- The output is related to the evaluation of a binary classification model using a confusion matrix and various statistics. The confusion matrix shows the number of true positives (139), false positives (124), false negatives (50), and true negatives (60) that the model has produced. The rows represent the predicted class (0 or 1), while the columns represent the true class (0 or 1).

The statistics calculated from the confusion matrix are as follows:

Accuracy: The proportion of correct predictions (139+60) divided by the total number of predictions (139+124+50+60), which is 0.5335 or 53.35%.

95% CI: The 95% confidence interval for the accuracy, which ranges from 0.4815 to 0.585.

P-Value [Acc > NIR]: The p-value for the hypothesis that the model's accuracy is greater than the no information rate. In this case, it is 0.1626, which is not significant at the 0.05 level.

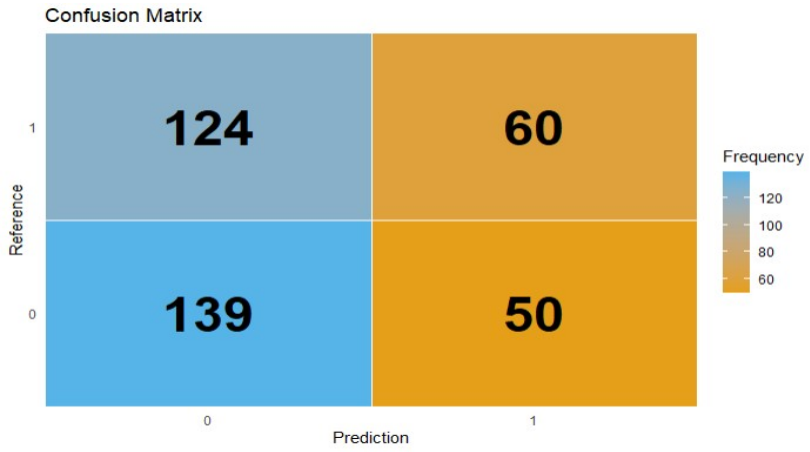


Figure1: Confusion Matrix of logistic Regression

Confusion matrix for a binary classification problem is shown in Figure 1 above, where 0 denotes no risk of heart disease and 1 denotes risk of heart disease.

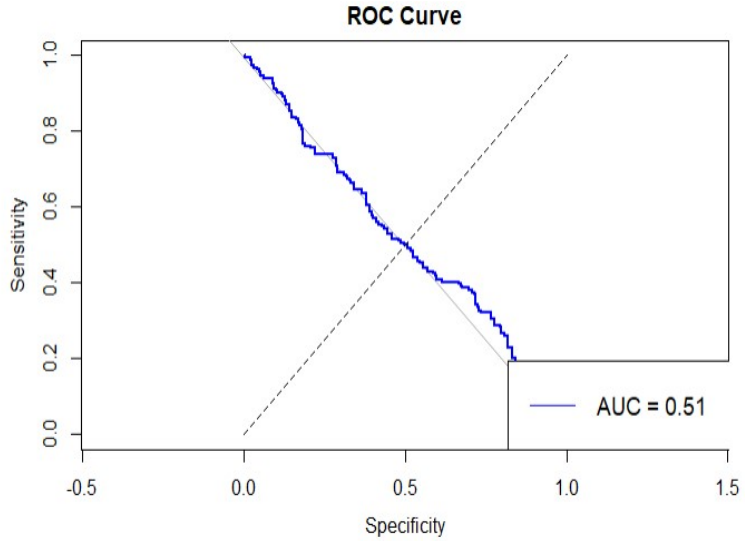


Figure2: ROC Curve for Logistic Regression

Given that the ROC curve is at a 45-degree angle and the AUC value is 0.51 in Figure 2, the model's accuracy is lower as indicated by the ROC curve. More accuracy is achieved with a higher AUC value. Our model's accuracy is lower.

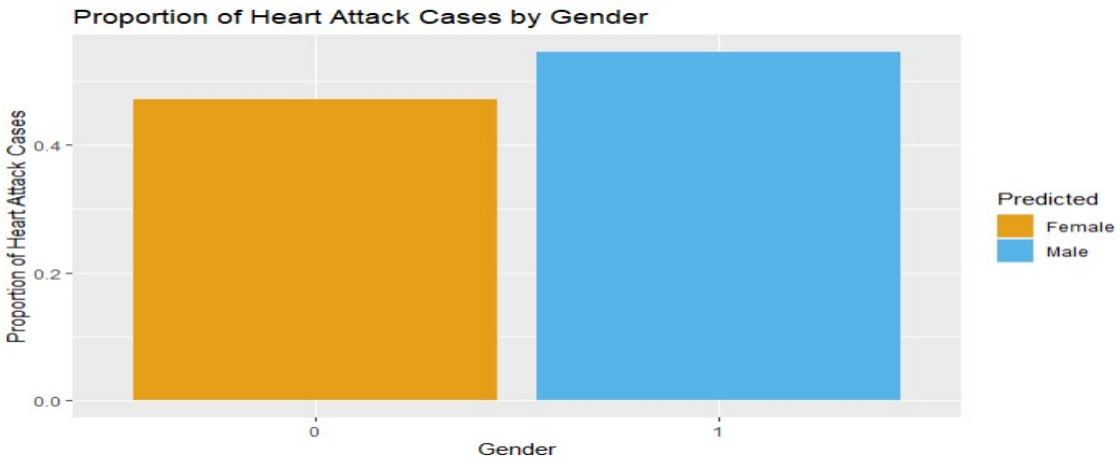


Figure3: Proportion of Heart Attack cases by Gender

In Figure 3 above, it is predicted that men have a higher risk of developing heart disease than women do.

SVM

We are conducting a heart disease prediction analysis using a statistical technique known as support vector machine (SVM). Heart disease is a common and serious medical condition that affects millions of individuals globally. SVM is a powerful machine learning algorithm that can be used for binary classification tasks such as predicting the presence or absence of heart disease.

In the case of this heart disease dataset, the SVM with a linear kernel attempts to find the best hyperplane in the feature space that separates patients with heart disease from those without heart disease. It tries to find a decision boundary that separates the positive and negative classes using a linear function of the input variables. The optimization problem that SVM solves is to find the hyperplane with the maximum margin that correctly classifies all training samples.Oncethe SVM model is trained on the dataset, it can be used to predict whether a new patient has heart disease or not based on their clinical and demographic characteristics.

Data Pre-processing

Removing missing values: We checked for missing values in the heart disease dataset and removed any rows with missing values. We also removed rows with a cholesterol value of 0. Handling missing values is important because the SVM algorithm cannot handle missing values, and they may cause errors in the analysis.

Splitting data into input features and output target: The heart disease dataset is split into input features (X) and output target (y). The input features are all the columns except the last

one, which contains the output target. The output target is the presence or absence of heart disease, which is stored in the 'HeartDisease' column.

Scaling the input features: The input features are scaled by using the preProcess() function from the caret package. Scaling is an important step in data pre-processing because it ensures that all input features are on the same scale. This helps to avoid any bias towards variables that have a larger range of values. The predict() function is then used to apply the scaling transformation to the input features.

Analysis

- An SVM model is created by specifying a train control object with 10-fold cross-validation. Train control specifies the type of resampling method used to evaluate the model's performance.

- The output target variable is converted into a factor variable with levels 0 and 1. This is important because SVM requires the output target variable to be a factor with levels indicating the presence or absence of heart disease.

- The SVM model is trained on the training data using the train() function and considered the output target variable "y" to be predicted using all the input features in "X". The "svmLinear" algorithm was chosen to predict the SVM model.

- Using the SVM model, we predicted the output target variable that indicates the presence or absence of heart disease based on the scaled input features of the test data.

- The confusion matrixes of the predicted and actual output target variables are calculated.

Result

- Accuracy: The accuracy of the model is reported as 0.8442 or 84.42%. This represents the proportion of correctly classified samples out of the total number of samples. In the heart disease dataset, the SVM model correctly predicted the presence or absence of heart disease in approximately 84.42% of the samples.

- 95% CI: The 95% confidence interval for the accuracy, which ranges from 0.8191 to 0.8671.

- No Information Rate: The accuracy that would be achieved by always predicting the majority class. In this case, the majority class is '0', so the NIR is 55.34%.

- P-Value [Acc > NIR]: The p-value for the hypothesis that the model's accuracy is greater than the no information rate. In this case, it is 2.2e- 16, which is significant at the 0.05 level

- Confusion Matrix: The confusion matrix displays the counts of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) predictions. In this case, the confusion matrix shows that the SVM model predicted 298 samples as class 0 (no heartdisease) correctly (TP), while misclassifying 31 samples as class 1 (heart disease) (FP). It also predicted 477 samples as class 1 correctly (TN), while misclassifying 112 samples as class 0 (FN).

- Sensitivity: Sensitivity, also known as true positive rate or recall, is the proportion of actual positive cases (heart disease) that are correctly predicted by the model. In this case, the sensitivity is reported as 0.7268 or 72.68%. This indicates that the SVM model correctly predicted approximately 72.68% of the cases with heart disease.

- Specificity: Specificity is the proportion of actual negative cases (no heart disease) that are correctly predicted by the model. In this case, the specificity is reported as 0.9390 or 93.90%. This indicates that the SVM model correctly predicted approximately 93.90% of the cases without heart disease.

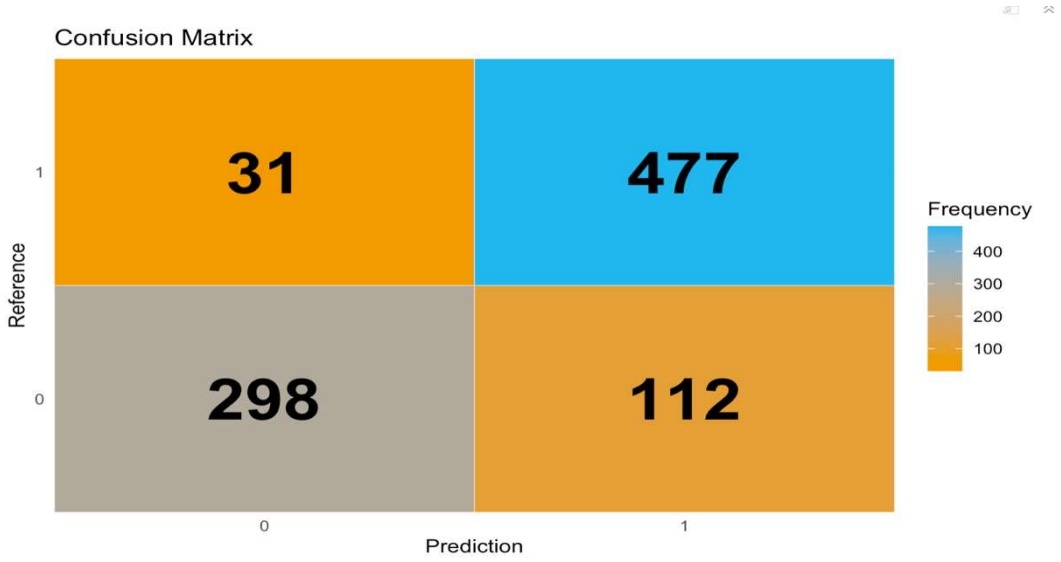


Figure 4: Confusion Matrix of support vector machine Regression.

Confusion matrix for a binary classification problem is shown in Figure 4 above, where 0 denotes no risk of heart disease and 1 denotes risk of heart disease.

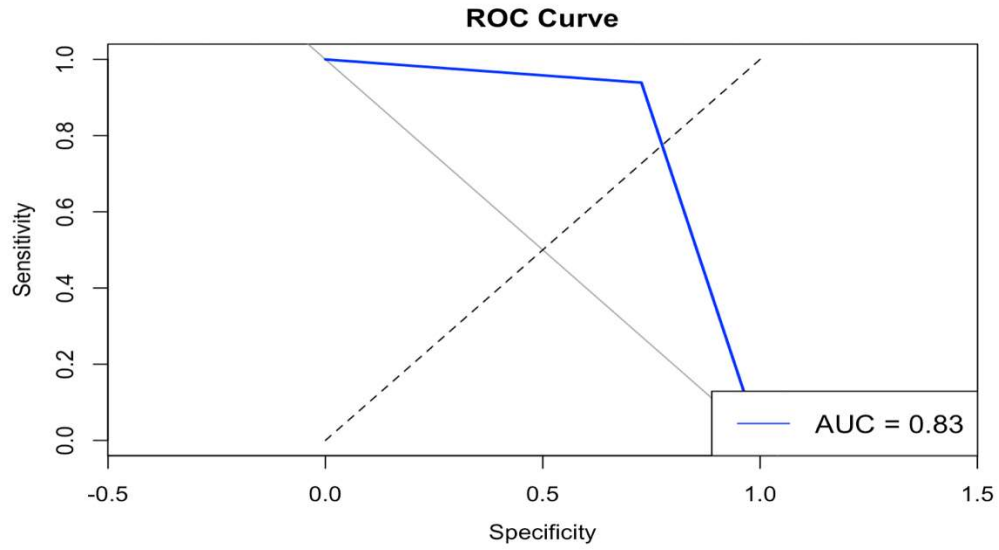


Figure 5 : ROC Curve for Support Vector Machine Regression

More accuracy is achieved with a higher AUC value. Our model's accuracy is higher, as our model's AUC value is 0.83 .

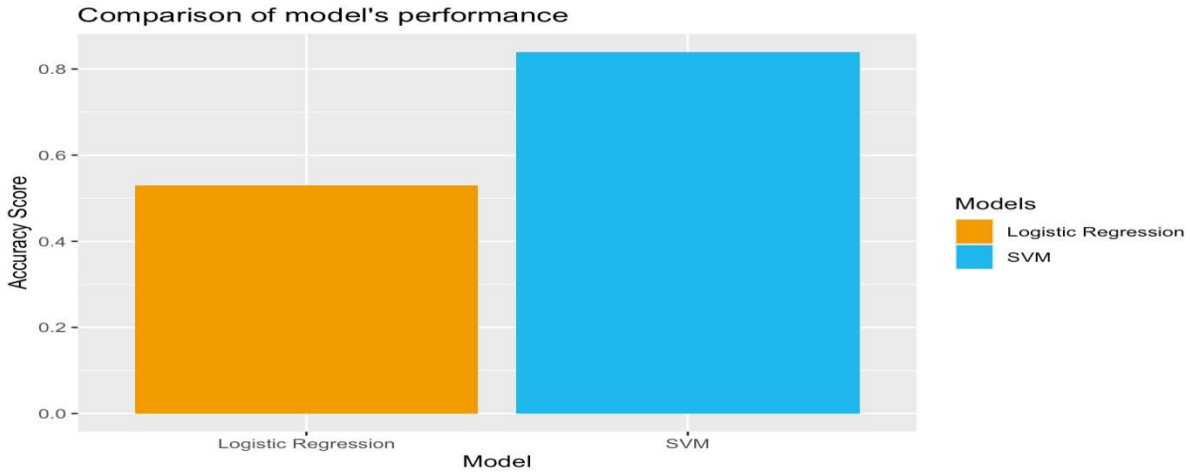


Figure 6 : Comparison of Logistic Regression and Support Vector Machine Regression Model.

By using the model's accuracy of both methods, SVM outperforms logistic regression with the accuracy of 0.84.

Data Collection Method

We took the dataset from Kaggle with the file name "heart.csv." It includes columns for age, gender, type of chest pain, resting blood pressure, cholesterol level, whether fasting blood sugar is greater than 120 or not, whether resting ECG is normal or not, and so on. The dataset contains 8 columns and 918 observations.

The data available in the heart failure dataset and its characteristics are given below.

- Age: age of the patient
- Sex: gender of the patient
- Chest Pain: chest pain (0 – No chest pain, 1 – Chest Pain)
- RestingBP: resting blood pressure (0 – Normal BP, 1 – High BP)
- Cholesterol: total cholesterol [mm/dl]
- FastingBS: fasting blood sugar [0: normal , 1: High blood sugar]
- RestingECG: resting electrocardiogram results [0: Normal, 1: At risk]
- MaxHR: maximum heart rate achieved

Conclusion

Heart disease prediction analysis can help identify individuals who are at high risk for developing heart disease before any symptoms appear and can provide personalized care that is tailored to the individual patient's needs. This can allow for early interventions to prevent or slow the progression of heart disease.

Overall, heart disease prediction analysis can help improve patient outcomes, reduce healthcare costs, and promote public health. It is an important tool for healthcare providers and policymakers in the fight against heart disease, which remains one of the leading causes of death worldwide.

References

- Rahman, R. (2021, March 22). *Heart attack analysis & prediction dataset*. Kaggle. Retrieved February 10, 2023, from https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset
- Devastator, T. (2023, January 12). *Predicting heart disease using clinical variables*. Kaggle. Retrieved February 10, 2023, from https://www.kaggle.com/datasets/thedevastator/predicting-heart-disease-risk-using-clinical-var
- Fedesoriano (2021) *Heart failure prediction dataset*, *Kaggle*. Available at: https://www.kaggle.com/datasets/fedesoriano/heart-failure-prediction (Accessed: February 10, 2023).
- Shreyasajal (2021) *Inferential statistics on Heart Attack Analysis*, *Kaggle*. Kaggle. Available at: https://www.kaggle.com/code/shreyasajal/inferential-statistics-on-heart-attack-analysis#Mann-Whitney-U-test (Accessed: February 10, 2023).

- (2020). Prediction Of Heart Disease Using Hybrid Linear Regression. *European Journal of Molecular & Clinical Medicine*, 7(5), 1159-1171.

Appendix:

Acronym:

- HD :Heart Disease
- BMI :Body Mass Index
- Acc :Accuracy
- NIR :No Information Rate
- CI :Confidence Interval
- ROC :Receiver Operating Characteristic
- AUC :Area Under Curve
- SVM :Support Vector Machine
- TN :True Negative
- TP :True Positive
- FP :False Positive
- FN :False Negative
- ECG: Electrocardiogram

LinearProject

2023-03-21

Logistic Regression Model

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.1     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(dplyr)

# Import the dataset
df <- read.csv("C:/Users/RENJITH/Downloads/heart.csv")
sum(is.na(df))
```

```
## [1] 0
```

```
# Removing missing values of Cholesterol
set.seed(42)
df<- df[df$Cholesterol != 0,]

# Split the data into training and testing sets
trainIndex <- createDataPartition(df$HeartDisease, p = .5, list = FALSE, times = 1)
train_df <- df[trainIndex, ]
test_df <- df[-trainIndex, ]

# Setting target and predictors for training and testing
X_train <- train_df %>%
  select(-HeartDisease)
y_train <- train_df$HeartDisease
X_test <- test_df %>%
  select(-HeartDisease)
y_test <- test_df$HeartDisease

# Remove rows with missing values
X_train <- na.omit(X_train)
y_train <- y_train[!is.na(X_train[,1])]
X_test <- na.omit(X_test)
y_test <- y_test[!is.na(X_test[,1])]

# Scale the features
scaler <- preProcess(X_train, method="scale")
X_train <- predict(scaler, X_train)
X_test <- predict(scaler, X_test)

# Build the model using logistic regression
model <- glm(y_train ~ Cholesterol+Sex+Age+FastingBS+RestingBP, data = as.data.frame(X_train), f
amily = "binomial")
summary(model)
```

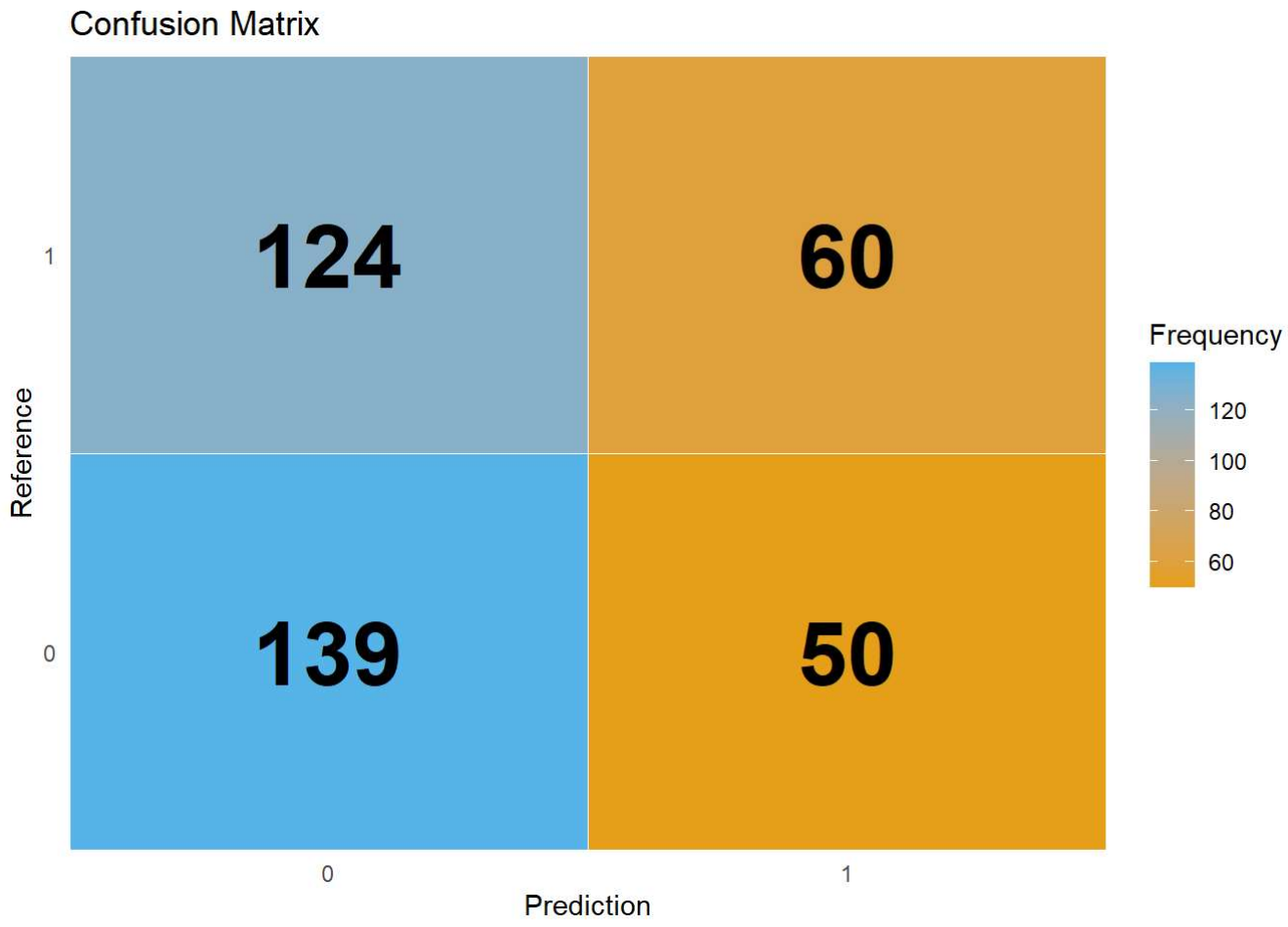
```
##
## Call:
## glm(formula = y_train ~ Cholesterol + Sex + Age + FastingBS +
##     RestingBP, family = "binomial", data = as.data.frame(X_train))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0417  -0.9648  -0.4494   0.9790   2.5530
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -7.1743     1.2626  -5.682 1.33e-08 ***
## Cholesterol   0.2054     0.1221   1.682   0.0925 .
## SexM          1.7099     0.3222   5.308 1.11e-07 ***
## Age           0.5957     0.1279   4.656 3.23e-06 ***
## FastingBS     0.2381     0.1260   1.889   0.0588 .
## RestingBP     0.1772     0.1239   1.430   0.1527
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 514.83  on 372  degrees of freedom
## Residual deviance: 434.15  on 367  degrees of freedom
## AIC: 446.15
##
## Number of Fisher Scoring iterations: 4
```

```
# Make predictions on the testing set
y_pred <- predict(model, newx = as.data.frame(X_test), type = "response")

# Convert predicted probabilities to class labels
y_pred_class <- ifelse(y_pred > 0.6, 1, 0)

# convert to factors with same levels
y_pred_class <-factor(y_pred_class, levels = c(0,1))
y_test <- factor(y_test, levels = c(0, 1))

# create confusion matrix
confusionMatrix(y_pred_class, y_test)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 139 124
##          1  50  60
##
##                Accuracy : 0.5335
##                  95% CI : (0.4815, 0.585)
##     No Information Rate : 0.5067
##     P-Value [Acc > NIR] : 0.1626
##
##                   Kappa : 0.0619
##
##  Mcnemar's Test P-Value : 3.128e-08
##
##             Sensitivity : 0.7354
##             Specificity : 0.3261
##          Pos Pred Value : 0.5285
##          Neg Pred Value : 0.5455
##              Prevalence : 0.5067
##          Detection Rate : 0.3727
##    Detection Prevalence : 0.7051
##       Balanced Accuracy : 0.5308
##
##        'Positive' Class : 0
##
```

Confusion Matrix Plot

```
library(ggplot2)

# Confusion matrix data
cm <- confusionMatrix(y_pred_class, y_test)

# create the confusion matrix plot
library(ggplot2)
cm_plot <- ggplot(data = as.data.frame(cm$table),
                  aes(x = Prediction, y = Reference, fill = as.numeric(Freq))) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "#E69F18", high = "#56B4E9") +
  theme_minimal() +
  labs(title = "Confusion Matrix", x = "Prediction", y = "Reference") +
  geom_text(aes(label = Freq), size = 12, fontface = "bold") +
  scale_x_discrete(expand = c(0, 0.1)) +
  scale_y_discrete(expand = c(0, 0.1))+
  guides(fill = guide_colorbar(title = "Frequency"))
cm_plot
```



ROC curve for Logistic Regression

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

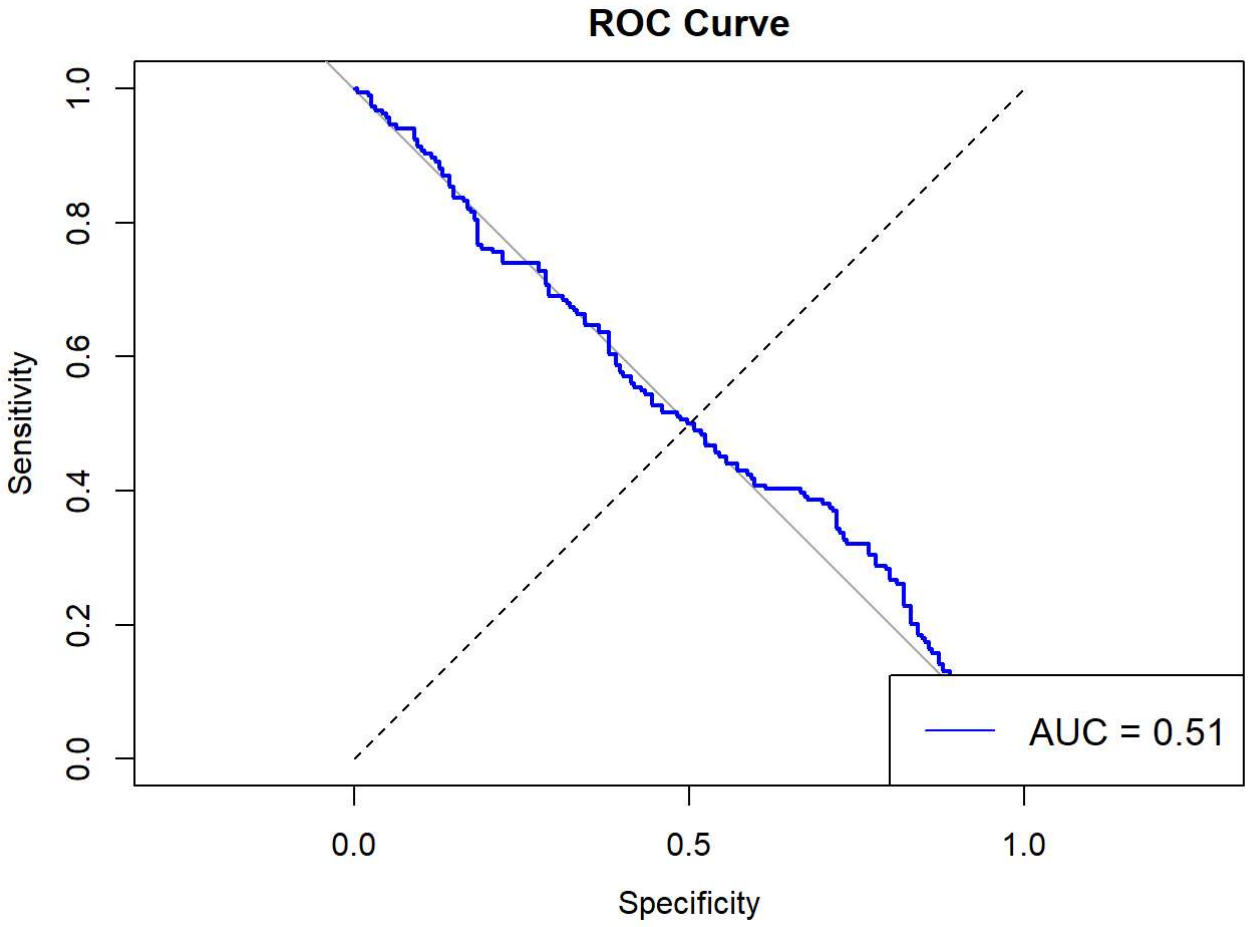
```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
# Calculate the ROC curve using predicted probabilities
roc_curve <- roc(y_test, y_pred)
```

```
## Setting levels: control = 0, case = 1
```

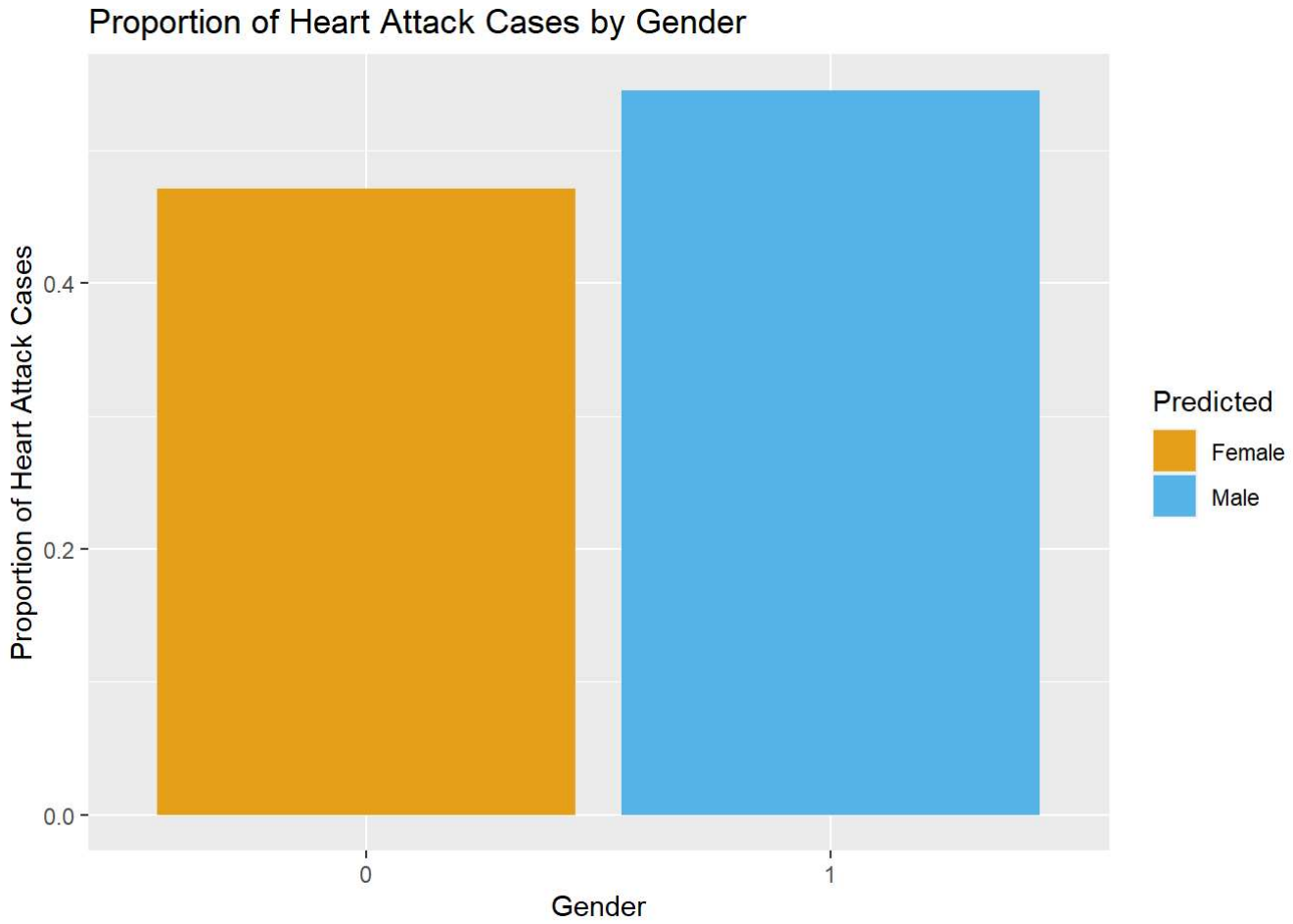
```
## Setting direction: controls < cases
```

```
# Plot the ROC curve
plot(roc_curve, col = "blue", main = "ROC Curve",xlim=c(0,1))
lines(x = c(0, 1), y = c(0, 1), lty = 2)
legend("bottomright", legend = paste0("AUC = ", round(auc(roc_curve), 2)), col = "blue", lty =
1, cex = 1.2)
```



Plot for Proportion of heart attack cases for each gender

```
library(tidyverse)

# Create a new data frame with predicted values and actual values
predictions_df <- data.frame(Predicted = y_pred_class, Actual = y_test)

# Add a column to count the number of heart attack cases for each gender
predictions_df <- predictions_df %>%
  mutate(Heart_Attack = ifelse(Actual == 1, 1, 0))

# Calculate the proportion of heart attack cases for each gender
prop_heart_attack <- predictions_df %>%
  group_by(Predicted) %>%
  summarise(prop_heart_attack = mean(Heart_Attack))


ggplot(prop_heart_attack, aes(x = Predicted, y = prop_heart_attack, fill = Predicted)) +
  geom_bar(stat = "identity") +
 scale_fill_manual(values = c("#E69F18",  "#56B4E9"), labels = c("Female", "Male")) +
  labs(title = "Proportion of Heart Attack Cases by Gender", x = "Gender", y = "Proportion of He
art Attack Cases")
```



LINEAR MODEL_SVM

2023-04-09

Support Vector Machine Regression Model:

```
# Load necessary libraries
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.1     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.4     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(e1071)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
# Load dataset
data <- read.csv("/Users/keerthanasenthilkumar/Downloads/heart.csv")
```

Exploratory data analysis

```
# Summary statistics
summary(data)
```

```
##       Age            Sex            ChestPainType        RestingBP
##  Min.   :28.00   Length:918         Length:918         Min.   :  0.0
##  1st Qu.:47.00   Class :character   Class :character   1st Qu.:120.0
##  Median :54.00   Mode  :character   Mode  :character   Median :130.0
##  Mean   :53.51                                         Mean   :132.4
##  3rd Qu.:60.00                                         3rd Qu.:140.0
##  Max.   :77.00                                         Max.   :200.0
##   Cholesterol      FastingBS       RestingECG            MaxHR
##  Min.   :  0.0   Min.   :0.0000   Length:918         Min.   : 60.0
```

```
##  1st Qu.:173.2   1st Qu.:0.0000   Class :character   1st Qu.:120.0
##  Median :223.0   Median :0.0000   Mode  :character   Median :138.0
##  Mean   :198.8   Mean   :0.2331                      Mean   :136.8
##  3rd Qu.:267.0   3rd Qu.:0.0000                      3rd Qu.:156.0
##  Max.   :603.0   Max.   :1.0000                      Max.   :202.0
##  ExerciseAngina        Oldpeak          ST_Slope          HeartDisease
##  Length:918         Min.   :-2.6000   Length:918         Min.   :0.0000
##  Class :character   1st Qu.: 0.0000   Class :character   1st Qu.:0.0000
##  Mode  :character   Median : 0.6000   Mode  :character   Median :1.0000
##                     Mean   : 0.8874                      Mean   :0.5534
##                     3rd Qu.: 1.5000                      3rd Qu.:1.0000
##                     Max.   : 6.2000                      Max.   :1.0000
```

```
names(data)
```

```
##  [1] "Age"            "Sex"            "ChestPainType"  "RestingBP"
##  [5] "Cholesterol"    "FastingBS"      "RestingECG"     "MaxHR"
##  [9] "ExerciseAngina" "Oldpeak"        "ST_Slope"       "HeartDisease"
```

```
# Check for missing values
heart <-na.omit(data)
sum(is.na(data))
```

```
## [1] 0
```

```
# Splitting the data into input features and output target
X <- data[, -12]
y <- data$HeartDisease
# Removing rows with missing values in the input features
X <- na.omit(X)
y <- y[!is.na(X[,1])]
# Scaling the input features
scaler <- preProcess(X, method="scale")
X <- predict(scaler, X)
# Creating a train control object with 10-fold cross-validation
train_control <- trainControl(method = "cv", number = 10)
y <- factor(y, levels=c(0,1))
# Defining the SVM model
svm_model <- train(y ~ ., data = cbind(X, y), method = "svmLinear", trControl = train_control)
# Printing the tuned model's performance
print(svm_model)
```

```
## Support Vector Machines with Linear Kernel
##
## 918 samples
##  11 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 826, 826, 826, 827, 826, 826, ...
## Resampling results:
```

```
##
##   Accuracy   Kappa
##   0.8660774  0.7276522
##
## Tuning parameter 'C' was held constant at a value of 1
```

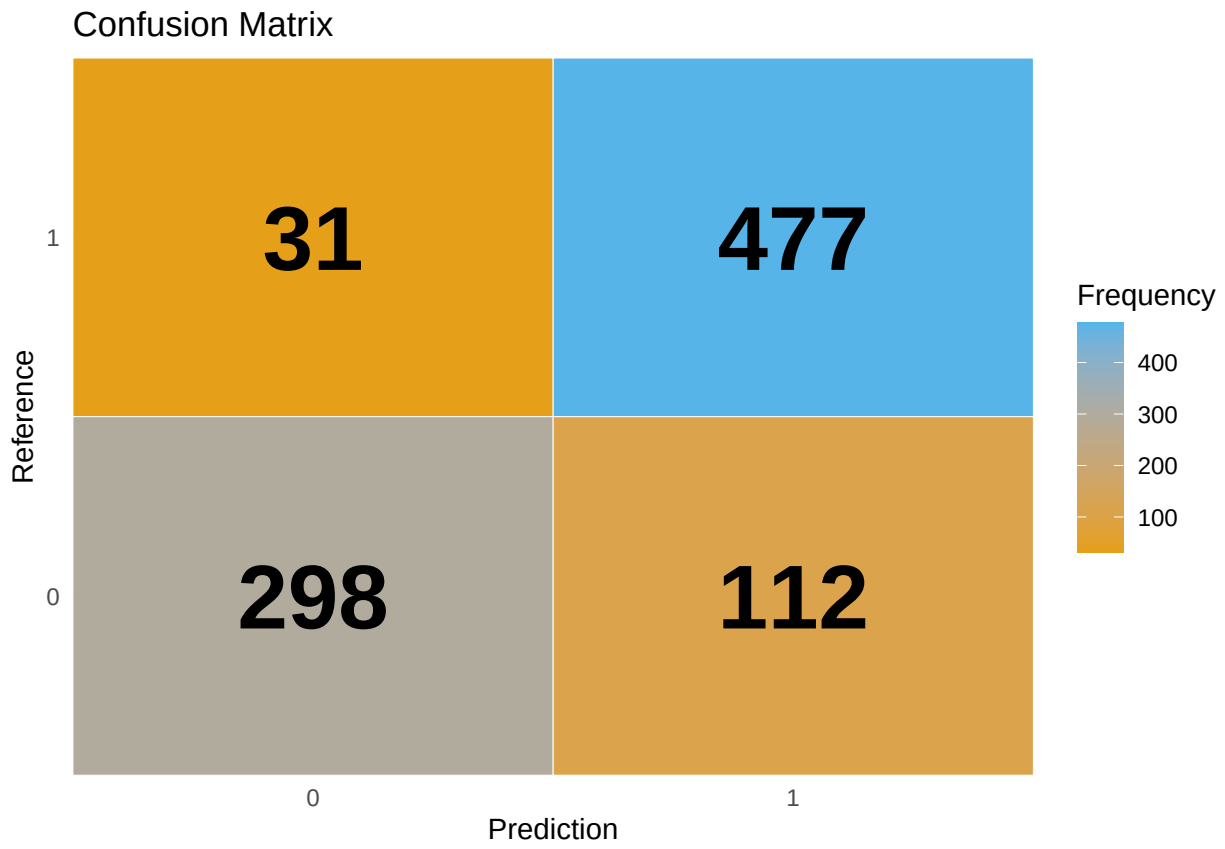
```
# Making predictions on the test data
y_pred <- predict(svm_model, newdata = predict(scaler, X))
y_pred <- factor(y_pred, levels=c(0,1))
y <- factor(y, levels=c(0,1))
#calculating the accuarcy of the model:
cm <-confusionMatrix(data = y_pred, reference = y)
cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 298  31
##          1 112 477
##
##                Accuracy : 0.8442
##                  95% CI : (0.8191, 0.8671)
##     No Information Rate : 0.5534
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6787
##
##  Mcnemar's Test P-Value : 2.233e-11
##
##             Sensitivity : 0.7268
##             Specificity : 0.9390
##          Pos Pred Value : 0.9058
##          Neg Pred Value : 0.8098
##              Prevalence : 0.4466
##          Detection Rate : 0.3246
##    Detection Prevalence : 0.3584
##       Balanced Accuracy : 0.8329
##
##        'Positive' Class : 0
##
```

Confusion Matrix for SVM

```
# create the confusion matrix plot
library(ggplot2)
cm_plot <- ggplot(data = as.data.frame(cm$table),
                  aes(x = Prediction, y = Reference, fill = as.numeric(Freq))) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "#E69F18", high = "#56B4E9") +
  theme_minimal() +
  labs(title = "Confusion Matrix", x = "Prediction", y = "Reference") +
  geom_text(aes(label = Freq), size = 12, fontface = "bold") +
```

```
  scale_x_discrete(expand = c(0, 0.1)) +
  scale_y_discrete(expand = c(0, 0.1))+
  guides(fill = guide_colorbar(title = "Frequency"))
cm_plot
```



ROC curve for SVM

```
library(pROC)
```

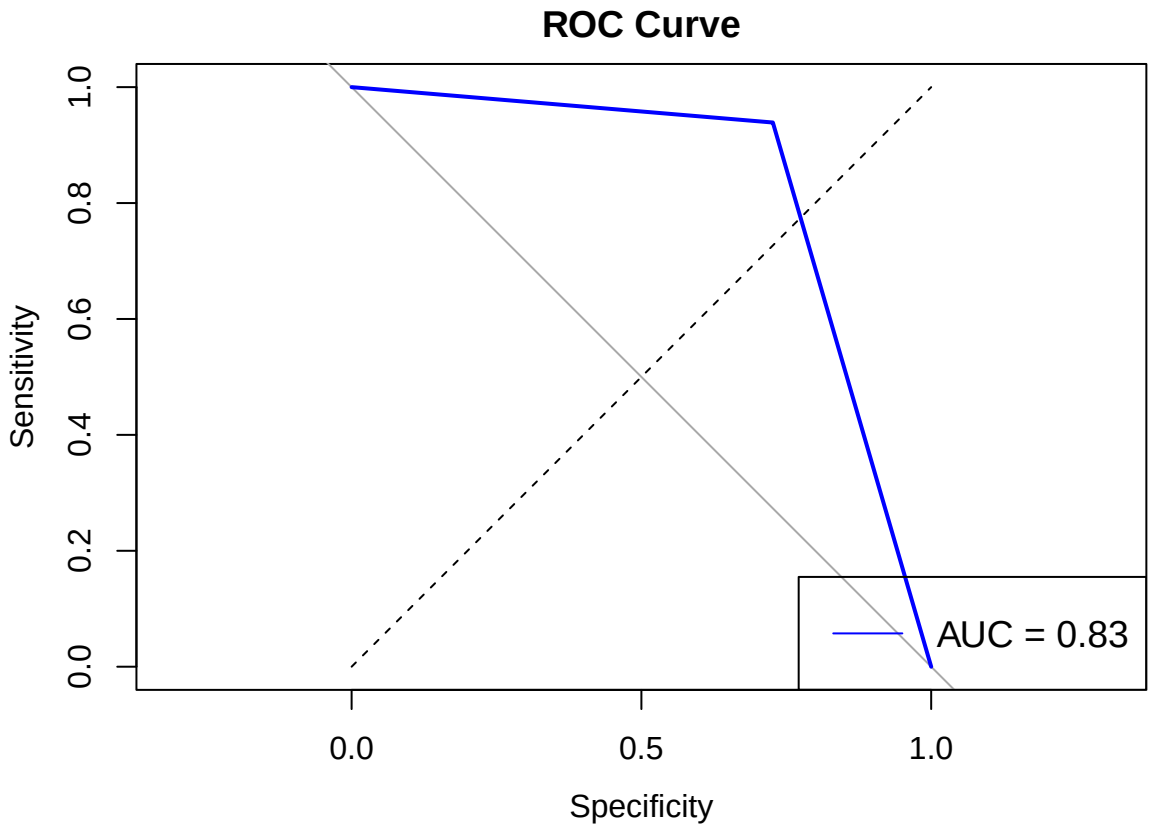
```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
# Calculate the ROC curve using predicted probabilities
# Convert y_pred to numeric
y_pred <- as.numeric(y_pred)
roc_curve <- roc(y, y_pred)
```

```
## Setting levels: control = 0, case = 1
```

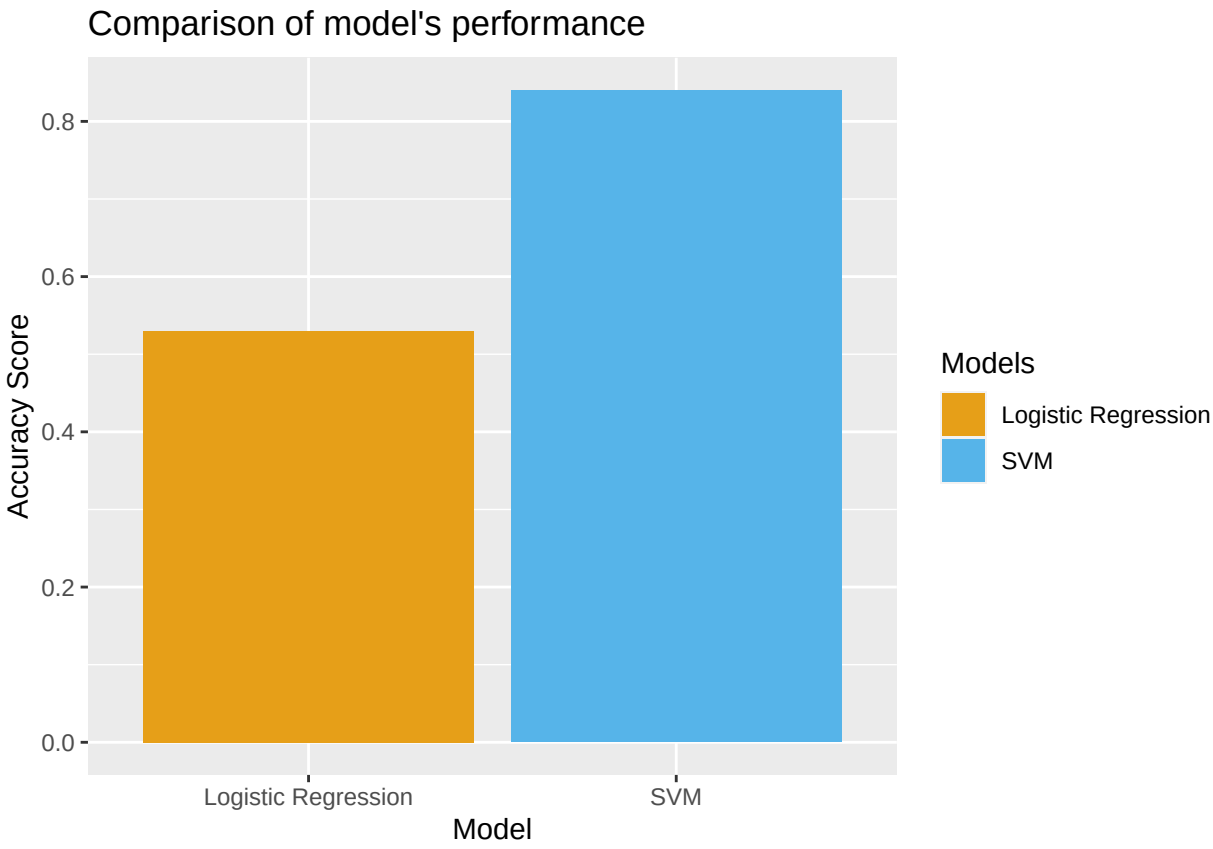
```
## Setting direction: controls < cases
```

```
# Plot the ROC curve
plot(roc_curve, col = "blue", main = "ROC Curve",xlim=c(0,1))
lines(x = c(0, 1), y = c(0, 1), lty = 2)
legend("bottomright", legend = paste0("AUC = ", round(auc(roc_curve), 2)), col = "blue", lty = 1, cex =
```



comparison of logistic regression and SVM

```
logistic_regression_accuracy <- 0.53
svm_accuracy <- 0.84
model_names <- c("Logistic Regression", "SVM")
accuracy_scores <- c(logistic_regression_accuracy, svm_accuracy)
data <- data.frame(model_names, accuracy_scores)
library(ggplot2)
ggplot(data, aes(x = model_names, y = accuracy_scores,fill=model_names)) +
  geom_bar(stat = "identity") +
   scale_fill_manual(values = c("#E69F18",  "#56B4E9"), labels = c("Logistic Regression", "SVM"),name="
  labs(title = "Comparison of model's performance",
       x = "Model",
       y = "Accuracy Score")
```


Comparison of model's performance
Accuracy Score
0.0
0.2
0.4
0.6
0.8
Logistic Regression
SVM
Model
Models
Logistic Regression
SVM