Predictive Modeling Assignment #8

Ryan Canfield

2023-12-15

### R Markdown

### Module 08: Assignment 01 - Final Project

### Due: Fri Dec 15, 2023 11:59pmDue: Fri Dec 15, 2023 11:59pm

### Instructions

## The final project assignment requires students to select a data science topic of interest, propose a research question that may be answered using predictive modeling methods, locate a suitable data file related to the data science topic and research question, analyze the data file using appropriate statistical methods and computational procedures and prepare a report summarizing your findings and documenting the procedures used. Students are strongly encouraged to begin plans for this project early in the course and discuss topics of interest with the instructor and employers.

# Opening the dataset and getting some different previews of the data.

heart <- read.csv("../Datasets/heart.csv")  
head(heart)

About this dataset:

Age : Age of the patient

Sex : Sex of the patient

exang : exercise induced angina (1 = yes; 0 = no)

ca : number of major vessels (0-3)

cp : Chest Pain type chest pain type Value 1: typical angina Value 2: atypical angina Value 3: non-anginal pain Value 4: asymptomatic

trtbps : resting blood pressure (in mm Hg)

chol : cholestoral in mg/dl fetched via BMI sensor

fbs : (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)

rest\_ecg : resting electrocardiographic results Value 0: normal Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) Value 2: showing probable or definite left ventricular hypertrophy by Estes’ criteria thalach : maximum heart rate achieved

target : 0 = less chance of heart attack 1= more chance of heart attack

# Exploratory data analysis

summary(heart)

cat("\n")

glimpse(heart)

# Correlation matrix   
pairs(heart)

# Looking at different histograms to get a better idea of the data  
par(mfrow = c(2,3))  
hist(heart$age, xlab = "Age of patients", main = "Age Distribution")  
hist(heart$sex, col = 4, breaks = 20, xlab = "Sex of patients", main = "Sex Distribution")  
hist(heart$chol, col = 3, xlab = "Cholestoral Levels", main = "Cholestoral Distribution")  
hist(heart$restecg, col = 2, breaks = 10, xlab = "Heart Rate Group", main = "Resting Electrocardiographic Results")  
hist(heart$trtbps, col = 5, xlab = "Blood Pressure", main = "Resting Blood Pressure Levels")  
hist(heart$thalach , col = 6, breaks = 8, xlab = "Heart Rate", main = "Maximum Heart Rate")

## Preprocessing

# count total missing values   
print("Count of total missing values - ")

sum(is.na(heart))

### Objective 1 (Quantitative Response Variable): To identify and quantify the factors that contribute to variations in cholesterol levels.

## First Model: Multiple Regression – Best Subset Selection

# Preliminary linear regression model to see what variables are statsitcally significiant.  
first\_Model <- lm(chol ~ ., data = heart)  
summary.lm(first\_Model)

# Taking the significant variables to produce the find BSS multiple linear regression model.  
first\_Model\_Final <- lm(chol ~ age + sex + restecg, data = heart)  
summary.lm(first\_Model\_Final)

## Second Model: Forward Stepwise Regression.

# To run stepwise regression:  
# First, defining the null model. We need this for forward stepwise regression:  
second\_Model\_Intercept\_Only <- lm(chol ~ 1, data = heart)  
  
# Next, have a model with all explanatory variables included.  
second\_Model\_All\_Variables <- lm(chol ~ ., data = heart)  
# The "." is a symbol that lets us include all the variables.  
  
# Perform forward stepwise regression here, using the step function.  
# We start from the intercept\_only model and try every combonation.   
# We tell "step" the direction of stepwise regression we want. .   
second\_Model <- step(second\_Model\_Intercept\_Only, direction = 'forward', scope = formula(second\_Model\_All\_Variables), trace = 0)  
  
# If we want to see the output of the forward stepwise regression, we can use this command:  
second\_Model$anova

# This gets us our coefficients for the model.  
second\_Model$coefficients

# Adding the variables above to find the best linear model.  
second\_Model\_Final <- lm(chol ~ age + sex + restecg + thall, data = heart)  
summary.lm(second\_Model\_Final)

## For future models.

## Splitting the data into training and testing set.

# Split the data into training and test set.  
set.seed(310)  
  
# This has all the data   
training\_Samples <- heart$output %>%   
 createDataPartition(p = 0.8, list = FALSE)  
  
heart\_Train\_Data <- heart[training\_Samples, ]  
heart\_Test\_Data <- heart[-training\_Samples, ]  
  
# If need here is the split separated by the data and then the target variable.  
heart\_TrainX <- heart\_Train\_Data[c(1:13)]  
heart\_Trainy <- heart\_Train\_Data[c(14)]  
heart\_TestX <- heart\_Test\_Data[c(1:13)]   
heart\_Testy <- heart\_Test\_Data[c(14)]

## Third Model: Ridge Regression

set.seed(310)  
  
# Setting up variables for Ridge Regression.  
heart\_TrainSet\_Matrix <- model.matrix(chol ~ ., data = heart\_Train\_Data)  
heart\_TestSet\_Matrix <- model.matrix(chol ~ ., data = heart\_Test\_Data)  
grid <- 10 ^ seq(10, -2, length = 100)  
  
# Fitting the model following the book.  
third\_Model\_RidgeR.fit <- glmnet(heart\_TrainSet\_Matrix, heart\_Train\_Data$chol, alpha = 0, lambda = grid, thresh = 1e-12)  
third\_Model\_RidgeR.cv <- cv.glmnet(heart\_TrainSet\_Matrix, heart\_Train\_Data$chol, alpha = 0, lambda = grid, thresh = 1e-12)  
third\_Model\_RidgeR.lambda.cv <- third\_Model\_RidgeR.cv$lambda.min  
  
third\_Model\_RidgeR.pred <- predict(third\_Model\_RidgeR.fit, s = third\_Model\_RidgeR.lambda.cv, newx = heart\_TestSet\_Matrix)  
print("The MSE on the Ridge Regression's testing set is")

mean((third\_Model\_RidgeR.pred - heart\_Test\_Data$chol)^2)

## 

## Fourth Model: Lasso Regression

set.seed(310)  
  
# Fitting a lasso model on the training set  
fourth\_Model\_Lasso.fit <- glmnet(heart\_TrainSet\_Matrix, heart\_Train\_Data$chol, alpha = 1, lambda = grid, thresh = 1e-12)  
  
# Choosing lambda by crossvalidation  
fourth\_Model\_Lasso.cv <- cv.glmnet(heart\_TrainSet\_Matrix, heart\_Train\_Data$chol, alpha = 1, lambda = grid, thresh = 1e-12)  
fourth\_Model\_Lasso.lambda.cv <- fourth\_Model\_Lasso.cv$lambda.min  
  
# Reporting the testing error  
fourth\_Model\_Lasso.pred <- predict(fourth\_Model\_Lasso.fit, s = fourth\_Model\_Lasso.lambda.cv, newx = heart\_TestSet\_Matrix)  
print("The MSE on the Lasso Model's testing set is")

mean((fourth\_Model\_Lasso.pred - heart\_Test\_Data$chol)^2)

# Finding the Lasso's coefficients  
fourth\_Model\_Lasso.coef <- predict(fourth\_Model\_Lasso.fit, s = fourth\_Model\_Lasso.lambda.cv, type = "coefficients")  
print("The Lasso coefficients are:")

## [1] "The Lasso coefficients are:"

round(fourth\_Model\_Lasso.coef, 3)

## 

## Fifth Model: Partial Least Squares

# Fitting the PLS model on the training set and looking at which M to choose.  
set.seed(310)  
fifth\_Model\_PLS.fit <- plsr(chol ~ ., data = heart\_Train\_Data , scale = TRUE, validation = "CV")  
summary(fifth\_Model\_PLS.fit)

# Finding out where dimensional stops being reduced used for ncomp below.  
validationplot(fifth\_Model\_PLS.fit, val.type = "MSEP")

# Finding the PLR testing error.  
fifth\_Model\_PLS.pred <- predict(fifth\_Model\_PLS.fit, heart\_Test\_Data, ncomp = 2)  
print("The MSE on the PLS Model's testing set is")

mean((fifth\_Model\_PLS.pred - heart\_Test\_Data$chol)^2)

## 

## Sixth Model: Regression Trees

# Fitting the regression trees.  
sixth\_Model\_Reg.Tree.Model <- tree(chol ~ ., heart\_Train\_Data)  
  
# Plotting the regression trees.  
plot(sixth\_Model\_Reg.Tree.Model)  
text(sixth\_Model\_Reg.Tree.Model, pretty = 0, cex = 0.65)

# Summarizing the results.  
summary(sixth\_Model\_Reg.Tree.Model)

# Getting the test MSE   
sixth\_Model\_Test.Pred <- predict(sixth\_Model\_Reg.Tree.Model, heart\_Test\_Data)  
mean((sixth\_Model\_Test.Pred - heart\_Test\_Data$chol)^2)

# Getting the train MSE.  
sixth\_Model\_Train.Pred <- mean(heart\_Train\_Data$chol)  
mean((sixth\_Model\_Train.Pred - heart\_Test\_Data$chol)^2)

# Using cross-validation in order to determine the optimal level of tree complexity and seeing if it helped improve the MSE.  
sixth\_Model\_.CV.Reg.Tree.Model <- cv.tree(sixth\_Model\_Reg.Tree.Model)  
plot(sixth\_Model\_.CV.Reg.Tree.Model$size , sixth\_Model\_.CV.Reg.Tree.Model$dev, type = "b")

sixth\_Model\_Pruned.Tree.Model <- prune.tree(sixth\_Model\_Reg.Tree.Model, best = 3)  
  
sixth\_Model\_Test.Pred <- predict(sixth\_Model\_Pruned.Tree.Model, heart\_Test\_Data)  
mean((sixth\_Model\_Test.Pred - heart\_Test\_Data$chol)^2)

## Seventh Model: Bagging

set.seed(310)  
  
#Creating the bagging model.  
Seventh\_Model\_Bagged.Trees.Model <- randomForest(y = heart\_Train\_Data$chol, x = heart\_Train\_Data[ , -5],  
 mtry = ncol(heart\_Train\_Data) - 5, importance = T)   
# Looking at the preformance of the model.  
Seventh\_Model\_Test.Pred <- predict(Seventh\_Model\_Bagged.Trees.Model, heart\_Test\_Data)  
mean((Seventh\_Model\_Test.Pred - heart\_Test\_Data$chol)^2)

## [1] 2087.262

# Getting the node purity here.  
importance(Seventh\_Model\_Bagged.Trees.Model) %>%  
 as.data.frame() %>%  
 rownames\_to\_column() %>%  
 arrange(desc(IncNodePurity))

## 

## Eighth Model Random Forests

set.seed(310)  
eighth\_Model\_RF.High <- randomForest(chol ~ ., data = heart\_Train\_Data, mtry = 3, importance = TRUE)  
  
eigth\_Model\_Yhat.RF <- predict(eighth\_Model\_RF.High, heart\_Test\_Data)  
mean((eigth\_Model\_Yhat.RF - heart\_Test\_Data$chol)^2)

importance(eighth\_Model\_RF.High)

varImpPlot(eighth\_Model\_RF.High)

## Ninth Model: Boosting

set.seed(310)  
  
ninth\_Model\_Lambda.Seq <- 10^seq(-5, 0, 0.1)  
  
ninth\_Model\_Heart.Train.MSE <- c()  
ninth\_Model\_Heart.Test.MSE <- c()  
  
for (i in 1:length(ninth\_Model\_Lambda.Seq)) {  
   
ninth\_Model\_Boost.Heart <- gbm(chol ~ . - chol, data = heart\_Train\_Data, distribution = "gaussian", n.trees = 1000, interaction.depth = 4,   
 shrinkage = ninth\_Model\_Lambda.Seq[i])  
   
 ninth\_Model\_Heart.Train.MSE[i] <- mean((predict(ninth\_Model\_Boost.Heart, heart\_Train\_Data, n.trees = 1000) - heart\_Train\_Data$chol)^2)  
 ninth\_Model\_Heart.Test.MSE[i] <- mean((predict(ninth\_Model\_Boost.Heart, heart\_Test\_Data, n.trees = 1000) - heart\_Test\_Data$chol)^2)  
   
}  
  
summary(ninth\_Model\_Boost.Heart)

### 

### Object 2 (Qualitative Response Variable): To see what factors contribute the most to having a more likely of a chance for a heart attack or a less likely of a chance for a heart attack.

## Tenth Model: KNN

set.seed(310)  
tenth\_Train.X = data.frame(heart$cp)  
tenth\_Test.X = data.frame(heart$cp)  
heart.output = heart$output  
  
# Creating the modeling.  
tenth\_Model\_KNN.Model = knn(tenth\_Train.X, tenth\_Test.X, heart.output, k = 1)  
table(tenth\_Model\_KNN.Model, heart.output)

## 

## 11th Model: Logistic Regression

# Forward Stepwise Logestic Regression   
  
# Fit an intercept-only model  
heart\_Null\_Model <- glm(output ~ 1, data = heart\_Train\_Data, family = binomial)  
  
# fit a model with everything  
heart\_All\_Model <- glm(output ~ ., data = heart\_Train\_Data, family = binomial)  
  
# Forward stepwise selection using AIC with both null and full models  
heart\_Final\_Model <- stepAIC(heart\_Null\_Model, scope = list(lower = heart\_Null\_Model, upper = heart\_All\_Model), direction = "forward", trace = 0)  
  
# Display the final model summary  
summary(heart\_Final\_Model)

# Obtain predicted probabilities on the testing set  
predicted\_probs <- predict(heart\_Final\_Model, newdata = heart\_Test\_Data, type = "response")  
  
# Assuming you have the true outcomes for the testing set (test\_data$output)  
observed\_responses <- as.factor(heart\_Test\_Data$output)  
  
# Convert predicted probabilities to binary predictions (e.g., using a threshold of 0.5)  
predicted\_classes <- as.factor(ifelse(predicted\_probs >= 0.5, 1, 0))  
  
# Create and displaying the confusion matrix  
conf\_matrix <- confusionMatrix(predicted\_classes, observed\_responses)  
conf\_matrix

## 

## 12th Model: LDA

# Creating the model based on variables from logistic regression  
twelfth\_Model\_LDA.Model = lda(output ~ cp + oldpeak + caa + sex + thalachh + thall + exng + trtbps + chol, data = heart\_Train\_Data)  
twelfth\_Model\_LDA.Model

# Making a confusion matrix to check accuracy.  
twelfth\_Model\_LDA.Model.Pred = predict(twelfth\_Model\_LDA.Model, heart\_Test\_Data)  
table(twelfth\_Model\_LDA.Model.Pred$class, heart\_Test\_Data$output)

## 13th Model: QDA

# Creating the model.  
thirteenth\_Model\_QDA.Model = qda(output ~ cp + oldpeak + caa + sex + thalachh + thall + exng + trtbps + chol, data = heart\_Train\_Data)  
thirteenth\_Model\_QDA.Model

# Checking its accuracy.  
thirteenth\_Model\_QDA.Model.Pred = predict(thirteenth\_Model\_QDA.Model, heart\_Test\_Data)  
table(thirteenth\_Model\_QDA.Model.Pred$class, heart\_Test\_Data$output)

## 14th Model: Classification Trees

# Creating the model  
fourteenth\_Model\_Class.Tree <- tree(output ~ .-output, data = heart\_Train\_Data)  
summary(fourteenth\_Model\_Class.Tree)

# Plotting the tree.  
plot(fourteenth\_Model\_Class.Tree)  
text(fourteenth\_Model\_Class.Tree , pretty = 0, cex = 0.65)

# Checking the accuracy with a confusion matrix.  
fourteenth\_Model\_Class.Tree.Pred <- predict(fourteenth\_Model\_Class.Tree, heart\_Test\_Data)  
fourteenth\_Model\_Class.Tree.Pred <- as.factor(ifelse(fourteenth\_Model\_Class.Tree.Pred >= 0.5, 1, 0))  
fourteenth\_Model\_Heart.Test.Data <- as.factor(heart\_Test\_Data$output)  
table(fourteenth\_Model\_Class.Tree.Pred, fourteenth\_Model\_Heart.Test.Data)

## 

## 15th Model: Bagging

set.seed(310)  
  
heart\_Train\_Data.factor <- as.factor(heart\_Train\_Data$output)  
heart\_Test\_Data.factor <- as.factor(heart\_Test\_Data$output)  
  
#Creating the bagging model.  
fifteenth\_Model\_Bagging <- randomForest(y = heart\_Train\_Data.factor, x = heart\_Train\_Data[ , -14], ntree = 100, importance = T)   
  
# Looking at the performance of the model.  
fifteenth\_Model\_Bagging.Pred <- predict(fifteenth\_Model\_Bagging, heart\_Test\_Data)  
table(fifteenth\_Model\_Bagging.Pred, heart\_Test\_Data.factor)

## 

## 16th Model: Random Forests

set.seed(310)  
  
  
  
sixteenth\_Model.Random.Forests.Train <- randomForest(heart\_Train\_Data.factor ~ cp + oldpeak + caa + sex + thalachh + thall + exng + trtbps + chol,  
 data = heart\_Train\_Data, ntree = 200, mtry = 3)   
sixteenth\_Model.Random.Forests.Train

sixteenth\_Model.Random.Forests.Test <- randomForest(heart\_Test\_Data.factor ~ cp + oldpeak + caa + sex + thalachh + thall + exng + trtbps + chol,  
 data = heart\_Test\_Data, ntree = 200, mtry = 3)  
sixteenth\_Model.Random.Forests.Test

### Objective 3 (Principal Components Regression)

## Final Model: Principal Components Regression

# Fitting the PCR model on the training set and looking at which M to choose.  
set.seed(310)  
final\_Model\_PCR.Fit <- pcr(output ~ ., data = heart\_Train\_Data , scale = TRUE, validation = "CV")  
summary(final\_Model\_PCR.Fit)

# Finding out where dimensional stops being reduced used for ncomp below.  
validationplot(final\_Model\_PCR.Fit, val.type = "MSEP")

# Finding the PCR testing error.  
final\_Model\_PCR.Pred <- predict(final\_Model\_PCR.Fit, heart\_Test\_Data, ncomp = 4)  
print("The MSE on the PCR Model's testing set is")

mean((final\_Model\_PCR.Pred - heart\_Test\_Data$output)^2)