IE 7275 Project

Chenghan Yue

4/17/2021

library(dummies)

## dummies-1.5.6 provided by Decision Patterns

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

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rpart)  
library(rpart.plot)  
library(forecast)

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

library(ggplot2)  
library(InformationValue)

##   
## Attaching package: 'InformationValue'

## The following objects are masked from 'package:caret':  
##   
## confusionMatrix, precision, sensitivity, specificity

library(ISLR)  
library("Hmisc")

## Loading required package: survival

##   
## Attaching package: 'survival'

## The following object is masked from 'package:caret':  
##   
## cluster

## Loading required package: Formula

##   
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':  
##   
## src, summarize

## The following objects are masked from 'package:base':  
##   
## format.pval, units

library(corrplot)

## corrplot 0.84 loaded

library(psych)

##   
## Attaching package: 'psych'

## The following object is masked from 'package:Hmisc':  
##   
## describe

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

library(reshape2)  
library(grid)  
library(gridExtra)

##   
## Attaching package: 'gridExtra'

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

library(randomForest)

## randomForest 4.6-14

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

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':  
##   
## combine

## The following object is masked from 'package:psych':  
##   
## outlier

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

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

library(pROC)

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

##   
## Attaching package: 'pROC'

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

library(e1071)

##   
## Attaching package: 'e1071'

## The following object is masked from 'package:Hmisc':  
##   
## impute

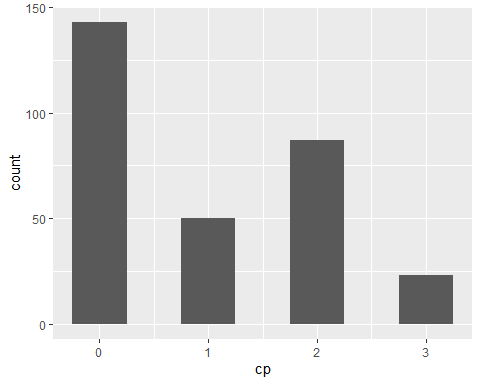
library(ROCR)  
library(FNN)

heart = read.csv("C:/IE 7275/Project/heart.csv", header = T)  
  
## Create dummy variables for the categorical predictors (Sex and chest pain type)  
heart.dummy = dummy.data.frame(heart, sep = ".", dummy.classes = "factor")  
heart.dummy = heart.dummy[,-c(1,2)]

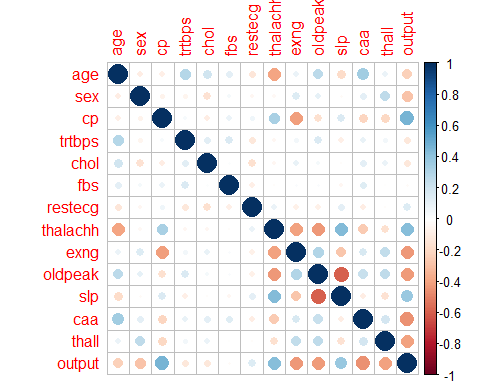
## Visualized data  
### heart attack output vs. age  
ggplot(heart) + geom\_histogram(aes(x = age), binwidth = 0.5)



### heart attack output vs. chest pain  
ggplot(heart) + geom\_histogram(aes(x = cp), binwidth = 0.5)



## Data correlation Plot  
heart.cor = cor(heart)  
heart.cor = cor(heart, method = c("spearman"))  
corrplot(heart.cor)

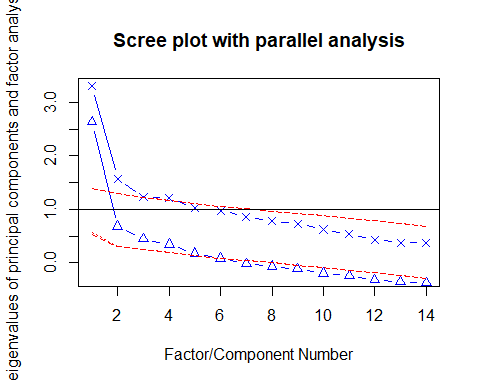


## Parallel Analysis Scree Plots\  
fa.parallel(heart, n.iter = 100,show.legend = F, main = "Scree plot with parallel analysis")

## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :  
## The estimated weights for the factor scores are probably incorrect. Try a  
## different factor score estimation method.  
  
## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :  
## The estimated weights for the factor scores are probably incorrect. Try a  
## different factor score estimation method.

## Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate = rotate, : An  
## ultra-Heywood case was detected. Examine the results carefully

## Warning in fa.stats(r = r, f = f, phi = phi, n.obs = n.obs, np.obs = np.obs, :  
## The estimated weights for the factor scores are probably incorrect. Try a  
## different factor score estimation method.



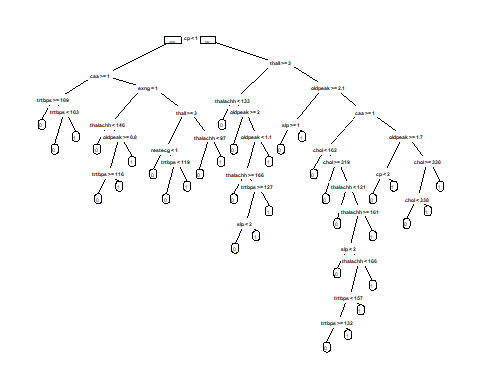
## Parallel analysis suggests that the number of factors = 5 and the number of components = 2

## PCA  
pca\_heart = prcomp(heart, center = T, scale. = T)  
summary(pca\_heart)

## Importance of components:  
## PC1 PC2 PC3 PC4 PC5 PC6 PC7  
## Standard deviation 1.8170 1.2539 1.1100 1.09847 1.0110 0.9850 0.92910  
## Proportion of Variance 0.2358 0.1123 0.0880 0.08619 0.0730 0.0693 0.06166  
## Cumulative Proportion 0.2358 0.3481 0.4361 0.52231 0.5953 0.6646 0.72627  
## PC8 PC9 PC10 PC11 PC12 PC13 PC14  
## Standard deviation 0.88096 0.85393 0.78913 0.73103 0.65577 0.60982 0.60658  
## Proportion of Variance 0.05544 0.05209 0.04448 0.03817 0.03072 0.02656 0.02628  
## Cumulative Proportion 0.78170 0.83379 0.87827 0.91644 0.94716 0.97372 1.00000

## Split the data into training (80%), validation(20%)  
set.seed(7275)  
train\_index = sample(nrow(heart.dummy), 0.8\*dim(heart))  
valid\_index = sample(setdiff(rownames(heart.dummy), train\_index), 0.2\*dim(heart)[1])  
valid\_index = as.numeric(valid\_index)  
train.df = heart.dummy[train\_index,]  
valid.df = heart.dummy[valid\_index,]

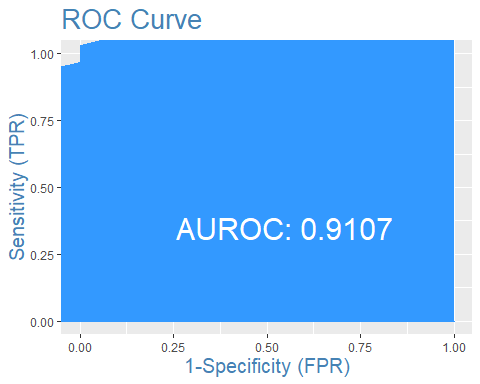
## fitting decision tree classification model  
  
## Run regression tree  
rt = rpart(output~ cp + trtbps + chol + fbs + restecg + thalachh + exng + oldpeak + slp + caa + thall, data = train.df, method = "class", minbucket = 1, maxdepth = 30, cp = 0.001)  
prp(rt)



## find the three or four most important car specifications for predicting the heart attack output  
t(t(rt$variable.importance))

## [,1]  
## cp 32.90058209  
## thalachh 30.96112021  
## oldpeak 25.55918489  
## thall 24.60120894  
## exng 23.23695835  
## slp 17.09906921  
## caa 15.94304334  
## trtbps 13.81436673  
## chol 13.41792713  
## restecg 2.34761905  
## fbs 0.04370581

## Predicting Model on Test Data Set  
predrt = predict(rt, newdata = valid.df, type = "prob")  
##(table\_pred = table(valid.df$output, predrt))  
#plot the ROC curve  
plotROC(valid.df$output, predrt)



## Logistic Regression  
logistic = glm(output~., data = train.df, family = "binomial")  
summary(logistic)

##   
## Call:  
## glm(formula = output ~ ., family = "binomial", data = train.df)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.8335 -0.4074 0.2428 0.6196 2.2072   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.4500096 2.1408207 1.144 0.252447   
## cp 0.7541070 0.2002025 3.767 0.000165 \*\*\*  
## trtbps -0.0183355 0.0111416 -1.646 0.099830 .   
## chol 0.0003841 0.0042853 0.090 0.928584   
## fbs -0.5049392 0.6102147 -0.827 0.407966   
## restecg 0.4847281 0.3706001 1.308 0.190889   
## thalachh 0.0166368 0.0096754 1.720 0.085523 .   
## exng -1.0008449 0.4331140 -2.311 0.020843 \*   
## oldpeak -0.7114880 0.2377588 -2.992 0.002767 \*\*   
## slp 0.6336053 0.3649336 1.736 0.082525 .   
## caa -0.7365536 0.1973551 -3.732 0.000190 \*\*\*  
## thall -1.1779443 0.3258444 -3.615 0.000300 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 333.10 on 241 degrees of freedom  
## Residual deviance: 181.75 on 230 degrees of freedom  
## AIC: 205.75  
##   
## Number of Fisher Scoring iterations: 6

exp(coef(logistic))

## (Intercept) cp trtbps chol fbs restecg   
## 11.5884576 2.1257125 0.9818316 1.0003841 0.6035423 1.6237335   
## thalachh exng oldpeak slp caa thall   
## 1.0167760 0.3675688 0.4909132 1.8843921 0.4787611 0.3079111

## propensities  
logistic\_pred = predict(logistic, valid.df, type = "response")  
#find optimal cutoff probability to use to maximize accuracy  
optimal = optimalCutoff(valid.df$output, logistic\_pred)[1]  
confusionMatrix(valid.df$output, logistic\_pred)

## 0 1  
## 0 22 2  
## 1 7 29

#calculate sensitivity  
sensitivity(valid.df$output, logistic\_pred)

## [1] 0.9354839

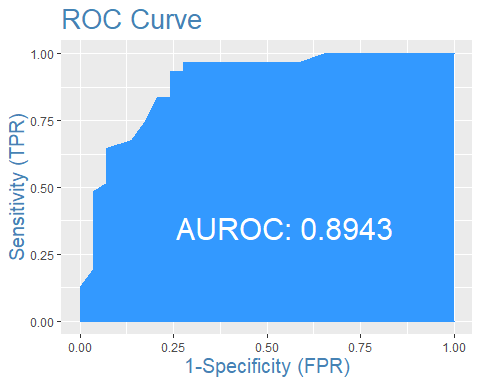
#calculate specificity  
specificity(valid.df$output, logistic\_pred)

## [1] 0.7586207

#calculate total misclassification error rate  
misClassError(valid.df$output, logistic\_pred, threshold = optimal)

## [1] 0.15

#plot the ROC curve  
plotROC(valid.df$output, logistic\_pred)



## SVM  
train.df$output = as.factor(train.df$output)  
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)  
  
svm\_Linear <- train(output ~., data = train.df, method = "svmLinear", trControl=trctrl, preProcess = c("center", "scale"), tuneLength = 10)  
svm\_Linear ## Therefore, it just tested at value “C” =1.

## Support Vector Machines with Linear Kernel   
##   
## 242 samples  
## 11 predictor  
## 2 classes: '0', '1'   
##   
## Pre-processing: centered (11), scaled (11)   
## Resampling: Cross-Validated (10 fold, repeated 3 times)   
## Summary of sample sizes: 218, 218, 218, 218, 218, 218, ...   
## Resampling results:  
##   
## Accuracy Kappa   
## 0.819901 0.632644  
##   
## Tuning parameter 'C' was held constant at a value of 1

## KNN  
## Normalized  
norm.values <- preProcess(train.df[, -c(12)], method=c("center", "scale"))  
train.df[, -c(12)] <- predict(norm.values, train.df[, -c(12)])  
valid.df[, -c(12)] <- predict(norm.values, valid.df[, -c(12)])  
heart.dummy[, -c(12)] = predict(norm.values, heart.dummy[, -c(12)])  
  
cl = train.df$output  
accuracy.df = data.frame(k = seq(1,14,1), accuracy = rep(0,14))  
for (i in 1:14) {  
 KNN\_b = knn(train = train.df[,-12], test = valid.df[,-12], cl, k = i, prob   
= T )  
## accuracy.df[i,2] = confusionMatrix(KNN\_b, as.factor(valid.df[,12]))$  
 accuracy.df[i,2] = sum(KNN\_b==valid.df[,12])/nrow(valid.df)  
}  
accuracy.df

## k accuracy  
## 1 1 0.7166667  
## 2 2 0.7333333  
## 3 3 0.8000000  
## 4 4 0.7833333  
## 5 5 0.7666667  
## 6 6 0.8000000  
## 7 7 0.8000000  
## 8 8 0.8000000  
## 9 9 0.8166667  
## 10 10 0.8166667  
## 11 11 0.8000000  
## 12 12 0.8166667  
## 13 13 0.7666667  
## 14 14 0.7833333

plot(accuracy.df,type = "l")

