MachineLearning\_Assignment 6

2023-02-28

Loading the necessary libraries:

library(tidyverse)   
library(readr)  
library(caret)  
library(klaR)  
library(rpart)  
library(rpart.plot)  
library(pROC)  
library(NHANES)  
options(tibble.print\_min = 5)  
  
knitr::opts\_chunk$set(  
 fig.width = 6,  
 fig.asp = .6,  
 out.width = "90%"  
)

## Q1 Loading NHANES data and limiting to 11 required variables; partitioning the data to 70/30:

set.seed(123)  
data(NHANES)  
   
nhanes\_hw6 = NHANES[c("Age", "Race1", "Education", "HHIncome", "Weight", "Height", "Pulse", "Diabetes", "BMI", "PhysActive", "Smoke100")] %>%  
 na.omit()  
  
summary(nhanes\_hw6)

## Age Race1 Education HHIncome   
## Min. :20.00 Black : 676 8th Grade : 351 more 99999 :1553   
## 1st Qu.:33.00 Hispanic: 347 9 - 11th Grade: 764 75000-99999: 770   
## Median :46.00 Mexican : 492 High School :1333 25000-34999: 660   
## Mean :47.12 White :4391 Some College :1991 35000-44999: 605   
## 3rd Qu.:60.00 Other : 450 College Grad :1917 45000-54999: 548   
## Max. :80.00 55000-64999: 455   
## (Other) :1765   
## Weight Height Pulse Diabetes BMI   
## Min. : 37.00 Min. :139.9 Min. : 40.00 No :5697 Min. :15.02   
## 1st Qu.: 67.60 1st Qu.:161.9 1st Qu.: 64.00 Yes: 659 1st Qu.:24.15   
## Median : 80.00 Median :169.2 Median : 72.00 Median :27.80   
## Mean : 82.88 Mean :169.1 Mean : 72.41 Mean :28.88   
## 3rd Qu.: 94.80 3rd Qu.:176.2 3rd Qu.: 80.00 3rd Qu.:32.35   
## Max. :230.70 Max. :200.4 Max. :128.00 Max. :81.25   
##   
## PhysActive Smoke100   
## No :2934 No :3544   
## Yes:3422 Yes:2812   
##   
##   
##   
##   
##

summary(nhanes\_hw6$Diabetes)

## No Yes   
## 5697 659

diab\_split = createDataPartition(nhanes\_hw6$Diabetes, p = 0.7, list = FALSE)  
train = nhanes\_hw6[diab\_split, ]  
test = nhanes\_hw6[-diab\_split, ]

#### The outcome Diabetes is unbalanced.

## Q2 Creating prediction models

### A. Classification tree

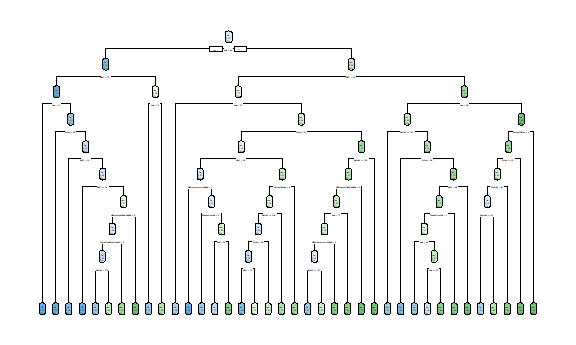
set.seed(123)  
#Creating 10-fold cross-validation and using down-sampling because of imbalance in data  
ctrl = trainControl(method = "cv", number = 10, sampling = "down")  
  
#Create sequence of cp parameters  
grid.2 = expand.grid(cp=seq(0.001, 0.3, by=0.01))  
  
#Train model  
class\_tree = train(Diabetes ~ ., data = train, method = "rpart", trControl = ctrl, tuneGrid=grid.2)  
  
class\_tree$bestTune

## cp  
## 1 0.001

class\_tree

## CART   
##   
## 4450 samples  
## 10 predictor  
## 2 classes: 'No', 'Yes'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 4005, 4005, 4004, 4005, 4005, 4006, ...   
## Addtional sampling using down-sampling  
##   
## Resampling results across tuning parameters:  
##   
## cp Accuracy Kappa   
## 0.001 0.7044851 0.2225926  
## 0.011 0.6692093 0.2027177  
## 0.021 0.6426893 0.1774506  
## 0.031 0.6269443 0.1755479  
## 0.041 0.6090081 0.1640054  
## 0.051 0.5793452 0.1431098  
## 0.061 0.5793452 0.1431098  
## 0.071 0.5793452 0.1431098  
## 0.081 0.5793452 0.1431098  
## 0.091 0.5793452 0.1431098  
## 0.101 0.5793452 0.1431098  
## 0.111 0.5793452 0.1431098  
## 0.121 0.5793452 0.1431098  
## 0.131 0.5793452 0.1431098  
## 0.141 0.5793452 0.1431098  
## 0.151 0.5793452 0.1431098  
## 0.161 0.5793452 0.1431098  
## 0.171 0.5793452 0.1431098  
## 0.181 0.5793452 0.1431098  
## 0.191 0.5793452 0.1431098  
## 0.201 0.5793452 0.1431098  
## 0.211 0.5793452 0.1431098  
## 0.221 0.5793452 0.1431098  
## 0.231 0.5793452 0.1431098  
## 0.241 0.5793452 0.1431098  
## 0.251 0.5793452 0.1431098  
## 0.261 0.5793452 0.1431098  
## 0.271 0.5793452 0.1431098  
## 0.281 0.5793452 0.1431098  
## 0.291 0.5793452 0.1431098  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was cp = 0.001.

rpart.plot(class\_tree$finalModel)

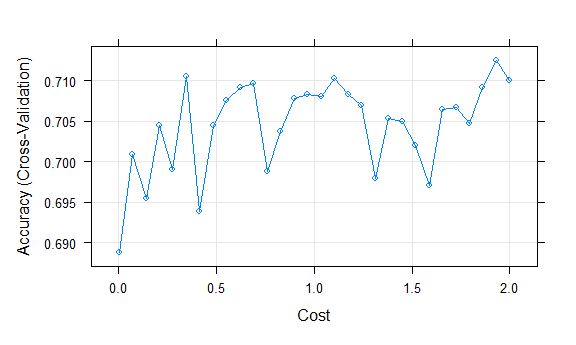


#Important variable  
varImp(class\_tree)

## rpart variable importance  
##   
## only 20 most important variables shown (out of 35)  
##   
## Overall  
## Age 100.0000  
## BMI 69.7194  
## Weight 49.7904  
## Height 29.0285  
## Pulse 28.3578  
## EducationCollege Grad 22.7297  
## PhysActiveYes 11.6649  
## HHIncomemore 99999 11.2164  
## Race1White 4.5234  
## Smoke100Yes 4.4698  
## HHIncome45000-54999 4.0866  
## Race1Mexican 3.7162  
## Race1Hispanic 1.9055  
## EducationSome College 1.8275  
## HHIncome55000-64999 1.1010  
## HHIncome65000-74999 1.0431  
## EducationHigh School 0.8472  
## HHIncome75000-99999 0.7452  
## HHIncome20000-24999 0.5552  
## `HHIncome55000-64999` 0.0000

### B. Support vector classifier

set.seed(123)  
#Creating 10-fold cross-validation   
cv =trainControl(method="cv", number=10, sampling = "down")  
  
#Train model  
svm\_mod = train(Diabetes ~ ., data = train, method = "svmLinear", trControl = cv, preProcess=c("center", "scale"), tuneGrid=expand.grid(C=seq(0.001,2, length=30)))  
  
#Visualizing accuracy versus values of C  
plot(svm\_mod)



#information about final model  
svm\_mod$finalModel

## Support Vector Machine object of class "ksvm"   
##   
## SV type: C-svc (classification)   
## parameter : cost C = 1.93106896551724   
##   
## Linear (vanilla) kernel function.   
##   
## Number of Support Vectors : 544   
##   
## Objective Function Value : -1011.328   
## Training error : 0.252165

### C. logistic regression

set.seed(123)  
cv = trainControl(method = "cv", number = 10, sampling = "down")  
  
lr\_mod = train(  
 Diabetes ~ .,   
 data = train,   
 method = "glm",   
 trControl = cv,   
 family = "binomial",  
 preProcess=c("center", "scale"),  
 tuneLength = 10  
)

## Q3 comparing the models in the training data

confusionMatrix(class\_tree)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction No Yes  
## No 62.6 2.6  
## Yes 27.0 7.8  
##   
## Accuracy (average) : 0.7045

confusionMatrix(svm\_mod)

## Cross-Validated (10 fold) Confusion Matrix   
##   
## (entries are percentual average cell counts across resamples)  
##   
## Reference  
## Prediction No Yes  
## No 63.0 2.1  
## Yes 26.7 8.3  
##   
## Accuracy (average) : 0.7126

lr\_mod$results

## parameter Accuracy Kappa AccuracySD KappaSD  
## 1 none 0.7148238 0.2362721 0.03751177 0.05495224

#### The model with highest accuracy is the optimal predictor model. Logistic regression algorithm will be used as the optimal predictor model because it has the highest accuracy among the 3 models (0.7148). The Suuport vector classifier algorithm model has the accuracy of 0.7126, and classification tree algorithm accuracy is 0.7045.

## Q4 Applying optimal prediction model in test data

set.seed(123)  
lr\_pred = predict(lr\_mod, test)  
lr\_cm = confusionMatrix(data= lr\_pred, reference=test$Diabetes, positive="Yes")  
  
lr\_cm

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 1194 39  
## Yes 515 158  
##   
## Accuracy : 0.7093   
## 95% CI : (0.6884, 0.7297)  
## No Information Rate : 0.8966   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : 0.242   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.8020   
## Specificity : 0.6987   
## Pos Pred Value : 0.2348   
## Neg Pred Value : 0.9684   
## Prevalence : 0.1034   
## Detection Rate : 0.0829   
## Detection Prevalence : 0.3531   
## Balanced Accuracy : 0.7503   
##   
## 'Positive' Class : Yes   
##

#### On applying the logistic regression model within the test data set, the accuracy is 0.7093, kappa is 0.242 and sensitivity is 0.8020.

## Q5 2 limitations/considerations of the model generated by this analysis.

#### One limitation or consideration can be that the model generated by this analysis may not be applicable to other populations or future time points, as the original NHANES data is cross-sectional. The variables included in the model also might not be as important predictors as others. Secondly, any innate biases in the data can affect the accuracy of the model and require further studies to validate it.