Demonstration of Simple Support Vector Classifier

JAS

null

## Demonstration of Support Vector Classifiers

Data Citation: We are using a dataset containing features related to heart disease. There are 13 features and the outcome variable is a binary, classification variable indicating the presence of heart disease.

### Step 1: Load packages

e1071 contains the svm function. Caret contains the data partitioning functions for creating of our training and testing datasets. Remember to install the packages if this is your first time utilizing them.

library(e1071)  
library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

##Step 2: Load data and perform minor cleaning, check and recode missings etc. 1. How to load a flat text file 2. How to assign column names when none are provided 3. How to check variable types across the dataframe 4. How to recode missing indicators, change variable types and explore variable distributions

heart.data = read.csv("/Users/ashleytseng/OneDrive - cumc.columbia.edu/MPH/Spring 2020/EPID P8451\_Machine Learning/Sessions/Session 6/p8451\_session6/processed.cleveland.data", header = FALSE)  
  
var.names = c("age", "sex", "pain\_type", "resting\_sysbp", "chol", "fast\_blsugar\_gt120", "rest\_ecg", "max\_hr", "exerc\_angina", "ST\_depression", "ST\_slope", "vessels\_colorflu", "defect", "heart\_disease\_present")  
  
colnames(heart.data) = var.names  
str(heart.data)

## 'data.frame': 303 obs. of 14 variables:  
## $ age : num 63 67 67 37 41 56 62 57 63 53 ...  
## $ sex : num 1 1 1 1 0 1 0 0 1 1 ...  
## $ pain\_type : num 1 4 4 3 2 2 4 4 4 4 ...  
## $ resting\_sysbp : num 145 160 120 130 130 120 140 120 130 140 ...  
## $ chol : num 233 286 229 250 204 236 268 354 254 203 ...  
## $ fast\_blsugar\_gt120 : num 1 0 0 0 0 0 0 0 0 1 ...  
## $ rest\_ecg : num 2 2 2 0 2 0 2 0 2 2 ...  
## $ max\_hr : num 150 108 129 187 172 178 160 163 147 155 ...  
## $ exerc\_angina : num 0 1 1 0 0 0 0 1 0 1 ...  
## $ ST\_depression : num 2.3 1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 ...  
## $ ST\_slope : num 3 2 2 3 1 1 3 1 2 3 ...  
## $ vessels\_colorflu : Factor w/ 5 levels "?","0.0","1.0",..: 2 5 4 2 2 2 4 2 3 2 ...  
## $ defect : Factor w/ 4 levels "?","3.0","6.0",..: 3 2 4 2 2 2 2 2 4 4 ...  
## $ heart\_disease\_present: int 0 2 1 0 0 0 3 0 2 1 ...

heart.data[heart.data=="?"] = NA  
  
heart.data$defect = as.numeric(factor(heart.data$defect))  
heart.data$vessels\_colorflu = as.numeric(factor(heart.data$vessels\_colorflu))  
  
heart.data$outcome = ifelse(heart.data$heart\_disease\_present==0, 0,1)  
heart.data$heart\_disease\_present = NULL  
heart.data$outcome = factor(heart.data$outcome)  
levels(heart.data$outcome) = c("HD Not Present", "HD Present")  
str(heart.data)

## 'data.frame': 303 obs. of 14 variables:  
## $ age : num 63 67 67 37 41 56 62 57 63 53 ...  
## $ sex : num 1 1 1 1 0 1 0 0 1 1 ...  
## $ pain\_type : num 1 4 4 3 2 2 4 4 4 4 ...  
## $ resting\_sysbp : num 145 160 120 130 130 120 140 120 130 140 ...  
## $ chol : num 233 286 229 250 204 236 268 354 254 203 ...  
## $ fast\_blsugar\_gt120: num 1 0 0 0 0 0 0 0 0 1 ...  
## $ rest\_ecg : num 2 2 2 0 2 0 2 0 2 2 ...  
## $ max\_hr : num 150 108 129 187 172 178 160 163 147 155 ...  
## $ exerc\_angina : num 0 1 1 0 0 0 0 1 0 1 ...  
## $ ST\_depression : num 2.3 1.5 2.6 3.5 1.4 0.8 3.6 0.6 1.4 3.1 ...  
## $ ST\_slope : num 3 2 2 3 1 1 3 1 2 3 ...  
## $ vessels\_colorflu : num 1 4 3 1 1 1 3 1 2 1 ...  
## $ defect : num 2 1 3 1 1 1 1 1 3 3 ...  
## $ outcome : Factor w/ 2 levels "HD Not Present",..: 1 2 2 1 1 1 2 1 2 2 ...

summary(heart.data)

## age sex pain\_type resting\_sysbp   
## Min. :29.00 Min. :0.0000 Min. :1.000 Min. : 94.0   
## 1st Qu.:48.00 1st Qu.:0.0000 1st Qu.:3.000 1st Qu.:120.0   
## Median :56.00 Median :1.0000 Median :3.000 Median :130.0   
## Mean :54.44 Mean :0.6799 Mean :3.158 Mean :131.7   
## 3rd Qu.:61.00 3rd Qu.:1.0000 3rd Qu.:4.000 3rd Qu.:140.0   
## Max. :77.00 Max. :1.0000 Max. :4.000 Max. :200.0   
##   
## chol fast\_blsugar\_gt120 rest\_ecg max\_hr   
## Min. :126.0 Min. :0.0000 Min. :0.0000 Min. : 71.0   
## 1st Qu.:211.0 1st Qu.:0.0000 1st Qu.:0.0000 1st Qu.:133.5   
## Median :241.0 Median :0.0000 Median :1.0000 Median :153.0   
## Mean :246.7 Mean :0.1485 Mean :0.9901 Mean :149.6   
## 3rd Qu.:275.0 3rd Qu.:0.0000 3rd Qu.:2.0000 3rd Qu.:166.0   
## Max. :564.0 Max. :1.0000 Max. :2.0000 Max. :202.0   
##   
## exerc\_angina ST\_depression ST\_slope vessels\_colorflu  
## Min. :0.0000 Min. :0.00 Min. :1.000 Min. :1.000   
## 1st Qu.:0.0000 1st Qu.:0.00 1st Qu.:1.000 1st Qu.:1.000   
## Median :0.0000 Median :0.80 Median :2.000 Median :1.000   
## Mean :0.3267 Mean :1.04 Mean :1.601 Mean :1.672   
## 3rd Qu.:1.0000 3rd Qu.:1.60 3rd Qu.:2.000 3rd Qu.:2.000   
## Max. :1.0000 Max. :6.20 Max. :3.000 Max. :4.000   
## NA's :4   
## defect outcome   
## Min. :1.000 HD Not Present:164   
## 1st Qu.:1.000 HD Present :139   
## Median :1.000   
## Mean :1.837   
## 3rd Qu.:3.000   
## Max. :3.000   
## NA's :2

#Remove the missings  
heart.data.nomiss = na.omit(heart.data)  
  
#Set No Heart Disease as Reference Level  
heart.data.nomiss$outcome = relevel(heart.data.nomiss$outcome, ref = "HD Not Present")

### Step 3: Partition data into training and testing

set.seed(100)  
train.indices = createDataPartition(y = heart.data.nomiss$outcome, p = 0.7, list = FALSE)  
  
training = heart.data.nomiss[train.indices,]  
testing = heart.data.nomiss[-train.indices,]

### Step 4: Construct and tune the Support Vector Machine with a linear classifier (Support Vector Classifier)

SVM requires us to set the hyperparameter C or cost. The smaller the value of C, the less misclassification the SVM will accept (i.e. data that crosses the hyperplane). We also set the kernel as linear to fit a support vector classifier. By using scale=TRUE, we ask the svm to standardize the variables.

set.seed(100)  
svm.heart = svm(outcome ~ ., data = training, kernel = "linear", cost = 1, scale = TRUE)  
print(svm.heart) # There are 74 support vectors, indicating that the observations are likely close to each other.

##   
## Call:  
## svm(formula = outcome ~ ., data = training, kernel = "linear",   
## cost = 1, scale = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 1   
##   
## Number of Support Vectors: 74

svm.pred = predict(svm.heart, newdata = training[,1:13])  
table(svm.pred, training$outcome)

##   
## svm.pred HD Not Present HD Present  
## HD Not Present 101 18  
## HD Present 11 78

misClasificError = mean(svm.pred != training$outcome, na.rm = T)  
print(paste('Accuracy Model 1', 1-misClasificError))

## [1] "Accuracy Model 1 0.860576923076923"

features = training[,1:13]  
outcome = training$outcome  
  
svm\_tune = tune(svm, train.x = features, train.y = outcome, kernel = "linear", range = list(cost = 10^(-1:2)))  
  
summary(svm\_tune)

##   
## Parameter tuning of 'svm':  
##   
## - sampling method: 10-fold cross validation   
##   
## - best parameters:  
## cost  
## 0.1  
##   
## - best performance: 0.1490476   
##   
## - Detailed performance results:  
## cost error dispersion  
## 1 0.1 0.1490476 0.1041843  
## 2 1.0 0.1492857 0.1044250  
## 3 10.0 0.1590476 0.1154799  
## 4 100.0 0.1590476 0.1154799

svm.heart.new = svm(outcome ~ ., data = training, kernel = "linear", cost = 0.1, scale = TRUE)  
  
print(svm.heart.new)

##   
## Call:  
## svm(formula = outcome ~ ., data = training, kernel = "linear",   
## cost = 0.1, scale = TRUE)  
##   
##   
## Parameters:  
## SVM-Type: C-classification   
## SVM-Kernel: linear   
## cost: 0.1   
##   
## Number of Support Vectors: 86

svm.pred.new = predict(svm.heart.new, newdata = training[,1:13])  
table(svm.pred.new, training$outcome)

##   
## svm.pred.new HD Not Present HD Present  
## HD Not Present 102 18  
## HD Present 10 78

misClasificError.new = mean(svm.pred.new != training$outcome, na.rm = T)  
print(paste('Accuracy Model 1',1-misClasificError.new))

## [1] "Accuracy Model 1 0.865384615384615"

### Group Exercise: Modify the range for the tuning parameter, c, to explore values less than 0.1. Choose the optimal c identified and then apply the final model in the test set and obtain evaluation metrics.