Model Training for Prediction: KNN and Logistic Regression

JGWJ

8/26/2021

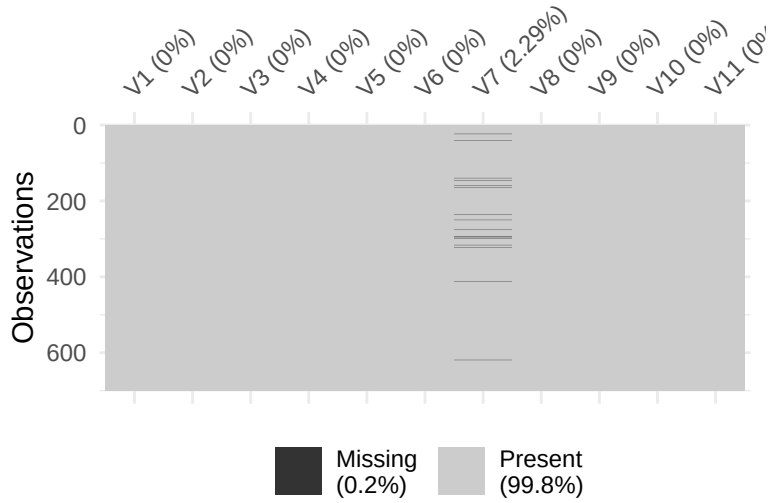
#1. R libraries and file imports

```
#----------1. Libraries----------
if(!require(tidyverse)) install.packages("tidyverse")
library(tidyverse)
if(!require(visdat)) install.packages("visdat")
library(visdat)
if(!require(stringr)) install.packages("stringr")
library(stringr)
if(!require(caret)) install.packages("caret")
library(caret)
if(!require(e1071)) install.packages("e1071")
library(e1071)
if(!require(kknn)) install.packages("kknn")
library(kknn)
if(!require(DAAG)) install.packages("DAAG")
library(DAAG)


#----------2. File Import----------
getwd()
list.files("")
file <- file.path("breast-cancer-wisconsin.data.txt")

data <- read.csv(file, stringsAsFactors=FALSE, header=FALSE, sep=",")

#----------3. Initial Explore----------
head(data)
str(data)
```

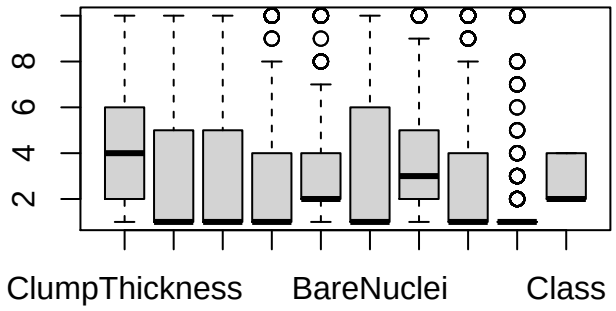
#2. Data Cleaning

```
#----------1. Missing Data----------
data[is.na(data)] #checking NA values

summary(data) #check all columns are numerical
data$V7
sum(data$V7=="?")

data$V7 <- as.integer(str_replace_all(data$V7,"\\?","NA"))
summary(data$V7)
```

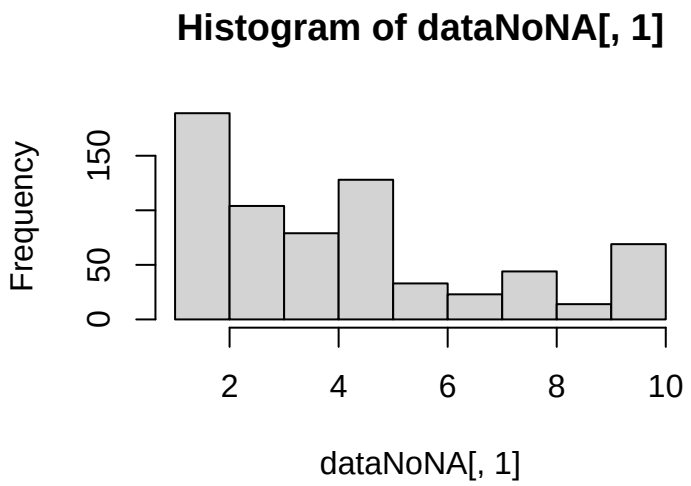
```
sum(is.na(data))
vis_miss(data)
```




```
#----------2. Duplicates----------
nrow(distinct(data))
data %>% count(V1) %>% filter(n>1) #V1 is not a distinct field


#----------3. Remove Data----------
data[,1] <- NULL #remove first column

missingData <- data %>% filter(is.na(V7)) #remove NA rows
dataNoNA <- anti_join(data, missingData, by="V7")
summary(dataNoNA)


#----------4. Renaming----------
str(dataNoNA)

colnames(dataNoNA) <- c("ClumpThickness","CellSizeUniformity",
                        "CellShapeUniformity","MarginalAdhesion",
                        "SingleEpithelialCellSize","BareNuclei",
                        "BlandChromatin","NormalNucleoli",
                        "Mitoses","Class")

str(dataNoNA)

#----------5. Statistics----------
cor(dataNoNA) #correlation

summary(dataNoNA) #min max mean median quartiles

boxplot(dataNoNA) #boxplot
```




```
hist(dataNoNA[,1]) #histogram
```

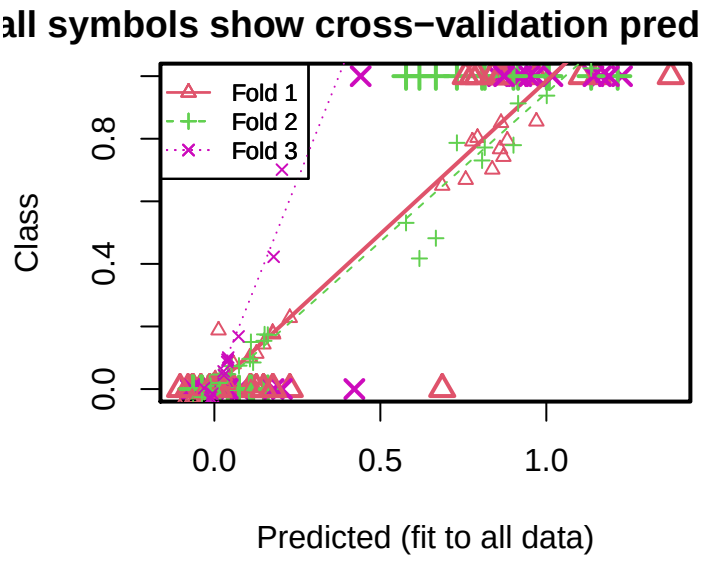



```
#qqnorm();qqline()
#skewness()
#shapiro.test()
#log10()
#grubbs.test()
```

#3. Modeling

```
#----------1. train/ validate/ test split----------

#function for splitting 6 sets of data
traintestsplit <- function(data) {
  set.seed(54)
```

```
  rand_row <- sample(rep(1:3,diff(floor(nrow(data) * c(0,0.7,0.85,1)))))

  train <- data[rand_row==1,]
  valid <- data[rand_row==2,]
  test <- data[rand_row==3,]

  return(list("train"=train, "valid"=valid, "test"=test))

}

dfnona <- traintestsplit(dataNoNA)


#----------2. logistic regression model----------
#changing response values to 0, 1 for logistic regression
#can be done at data cleaning step instead
dfnona$train[dfnona$train$Class==2,10] <- 0
dfnona$train[dfnona$train$Class==4,10] <- 1
dfnona$valid[dfnona$valid$Class==2,10] <- 0
dfnona$valid[dfnona$valid$Class==4,10] <- 1
dfnona$test[dfnona$test$Class==2,10] <- 0
dfnona$test[dfnona$test$Class==4,10] <- 1

logrModel <- glm(Class~., family=binomial(link="logit"),
                        data=dfnona$train) #logistic regression model
summary(logrModel)
coef(logrModel) #coefficients

pred <- ifelse(predict(logrModel, dfnona$valid, type="response")>0.5, 1, 0)
pred #predicted response

cm <- table(actual=dfnona$valid$Class,predicted=pred)
cm #confusion matrix

logrAccuracy <- confusionMatrix(cm, positive="0")
logrAccuracy$overall["Accuracy"]


#----------3. k-nearest neighbor model----------
knnModel <- kknn(Class~., dfnona$train, dfnona$valid, k=5, distance=2,
                 kernel="optimal",scale=FALSE)
accuracy <- sum(round(fitted(knnModel))== dfnona$valid[,10])/nrow(dfnona$valid)

print(accuracy)


#----------4. Cross Validation----------

cvLM <- cv.lm(form.lm=Class~., data=dfnona$valid)
```




```
attr(cvLM,"ms")


cvKNN <- cv.kknn(Class~., dfnona$valid, k=5, distance=2,
                 kernel="optimal",scale=FALSE)
cvKNNAccuracy <- sum(cvKNN[[1]][,2]==dfnona$valid$Class)/nrow(dfnona$valid)
print(cvKNNAccuracy) #0.804


#----------5. Test Data----------

#logistic regression
predFin <- ifelse(predict(logrModel, dfnona$test, type="response")>0.5, 1, 0)
predFin #predicted response

cmFin <- table(actual=dfnona$test$Class, predicted=predFin)
cmFin #confusion matrix

logrAccuracyFin <- confusionMatrix(cmFin, positive="0")
logrAccuracyFin$overall["Accuracy"] #0.981


#k-nearest neighbor
knnModelFin <- kknn(Class~., dfnona$train, dfnona$test, k=5, distance=2,
                 kernel="optimal",scale=FALSE)
knnAccuracyFin <- sum(round(fitted(knnModelFin))== dfnona$test[,10])/nrow(dfnona$test)
print(knnAccuracyFin) #0.971
```