Assignment 3 - Machine learning

Rohith chandra koyyala

06/03/2022

```r
#Installing the libraries
library(reshape2)
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
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v stringr 1.4.0
## v tidyr   1.1.4     v forcats 0.5.1
## v readr   2.1.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
library(ggplot2)
library(e1071)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(gmodels)
```

```
#Loading the universalbank dataset into UniversalBankdata
UniversalBankdata<- read.csv("C:/Users/kramr/Desktop/Fund ML/UniversalBank.csv", header = TRUE)
head(UniversalBankdata)
```

```
##   ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage
## 1  1  25          1     49    91107      4   1.6         1        0
## 2  2  45         19     34    90089      3   1.5         1        0
## 3  3  39         15     11    94720      1   1.0         1        0
## 4  4  35          9    100    94112      1   2.7         2        0
## 5  5  35          8     45    91330      4   1.0         2        0
## 6  6  37         13     29    92121      4   0.4         2      155
##   Personal.Loan Securities.Account CD.Account Online CreditCard
## 1             0                  1          0      0          0
## 2             0                  1          0      0          0
## 3             0                  0          0      0          0
## 4             0                  0          0      0          0
## 5             0                  0          0      0          1
## 6             0                  0          0      1          0
```

```
#partitioning data into training set(60) and validation set(40)

set.seed(64060)
split<- createDataPartition(UniversalBankdata$Personal.Loan, p=0.6, list = FALSE)
training<- UniversalBankdata[1:3000, ]
Valid<- UniversalBankdata[3001:5000, ]
NROW(training)
```

```
## [1] 3000
```

```
NROW(Valid)
```

```
## [1] 2000
```

```
#Feature Scaling: Scale is a inbuilt method to scale the columns of a numeric matrix
train_scale<- scale(training[,1:14])
Valid_scale<- scale(Valid[,1:14])
```

```
#1) creating pivot table using melt() and cast(). "melt" data so that each row is a unique id-variable
Melt.UBbank<- melt(training, id=c("CreditCard","Personal.Loan"), variable = "Online")
cast.UBbank<-dcast(Melt.UBbank, CreditCard+Personal.Loan~Online)
```

```
## Aggregation function missing: defaulting to length
```

```
cast.UBbank[,c(1:2,14)]
```

```
##   CreditCard Personal.Loan Online
## 1          0             0   1921
## 2          0             1    220
## 3          1             0    770
## 4          1             1     89
```

```r
#2) This is the probability of loan acceptance (Loan = 1) conditional on having a bank credit card (CC
# is 54/(54+477)= 54/531 = 0.101
table(training[,c(13,14,10)])
```

```
## , , Personal.Loan = 0
##
##       CreditCard
## Online    0    1
##      0  781  293
##      1 1140  477
##
## , , Personal.Loan = 1
##
##       CreditCard
## Online    0    1
##      0   81   35
##      1  139   54
```

```r
#3) Create two separate pivot tables for the training data. One will have Loan (rows) as a function of

Melt.UBbankc1 = melt(training, id=c("Online"), variable = "Personal.Loan") # Loan (rows) as a function
Melt.UBbankc2 = melt(training, id=c("CreditCard"), variable = "Personal.Loan") # Loan (rows) as a funct

recast.UBbankc1=dcast(Melt.UBbankc1, Online~Personal.Loan) #cast the melted data
```

```
## Aggregation function missing: defaulting to length
```

```r
recast.UBbankc2=dcast(Melt.UBbankc2, CreditCard~Personal.Loan)
```

```
## Aggregation function missing: defaulting to length
```

```r
#4) Compute the following quantities [P(A | B) means "the probability of A given B"]:

table(training[,c(14,10)])#credit card against personal.loan
```

```
##           Personal.Loan
## CreditCard    0    1
##          0 1921  220
##          1  770   89
```

```r
table(training[,c(13,10)])
```

```
##       Personal.Loan
## Online    0    1
##      0 1074  116
##      1 1617  193
```

```r
table(training[,c(13,14)])#online against CC
```

```
##       CreditCard
## Online    0    1
##      0  862  328
##      1 1279  531
```

```r
table(training[,c(13)])#Online
```

```
##
##    0    1
## 1190 1810
```

```r
table(training[,c(14)])#credit card
```

```
##
##    0    1
## 2141  859
```

```r
table(training[,c(10)])#personal.loan
```

```
##
##    0    1
## 2691  309
```

```r
# Calculating values based on above outputs
#i. P(CC = 1 | Loan = 1) (the proportion of credit card holders among the loan acceptors) = 89/(89+220)
#ii. P(Online = 1 | Loan = 1) = 193/(193+116) = 193/309 = 0.624
#iii. P(Loan = 1) (the proportion of loan acceptors) = 309/(309+2691) = 309/3000 = 0.103
#iv. P(CC = 1 | Loan = 0) = 770/(770+1921) = 770/2691 = 0.286
#v. P(Online = 1 | Loan = 0) = 1617/(1617+1074) = 1617/2691 = 0.6
#vi. P(Loan = 0) = 2691/(2691+309) = 2691/3000 = 0.897
```

```r
#5) Use the quantities computed above to compute the naive Bayes probability P(Loan = 1 | CC = 1, Onlin

#p(loan=0) = 2691
#p(loan=1) = 309
#p(cc,loan = 1)=89
#p(cc=1,loan=0)=770
#p(cc=0,loan=1)=220
#p(online,loan = 1)=193
#p(online=1,loan=0)=1617
#p(online=0,loan=1)=116

# P(Loan = 1 | CC = 1, Online = 1) = ((89/89+220)*(220/220+89)*(309/309+2691))/((89/89+220)*(220/220+89
# =((89/309)*(220/2141)*(309/3000))/((89/309)*(220/2141)*(309/3000))+((770/2691)*(1617/2691)*(2691/3000
# =(0.288*0.711*0.103)/((0.288*0.711*0.103)+(0.286*0.6*0.897)) = 0.021/0.021+0.154 = 0.021/0.175 =0.125
```

```r
# 6) Compare this value with the one obtained from the pivot table in (2). Which is a more accurate est
#the value obtained from pivot table is 0.101~0.125 . Both the vales are approximately equal. The value
```

```r
# 7) Implementing Naive Bayes Model
set.seed(64060)
N_model<- naiveBayes(Personal.Loan ~ CreditCard+Online, data = training)
N_model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##     0     1
## 0.897 0.103
##
## Conditional probabilities:
##    CreditCard
## Y        [,1]      [,2]
##   0 0.2861390 0.4520392
##   1 0.2880259 0.4535778
##
##    Online
## Y        [,1]      [,2]
##   0 0.6008919 0.4898061
##   1 0.6245955 0.4850126
```

```r
# the Navie Bayes out is nearly equal to the output derived from e.
#0.103~0.125
```