ASSIGNMENT3

deepak

2023-10-15

knitr::opts\_chunk$set(echo = TRUE)

library(ggplot2)  
library(lattice)  
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(readr)  
library(caret)  
library(dplyr)  
library(knitr)  
library(e1071)  
library(class)  
library(ISLR)  
library(reshape2)  
library(data.table)

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:reshape2':  
##   
## dcast, melt

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

#library(tydir)

#Importing Data set

#importing Data set and converting   
getwd()

## [1] "C:/Users/durga/OneDrive/Documents"

UB<-read.csv("C:/Users/durga/Downloads/UniversalBank.csv")  
#summarize the Data  
str(UB)

## 'data.frame': 5000 obs. of 14 variables:  
## $ ID : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Age : int 25 45 39 35 35 37 53 50 35 34 ...  
## $ Experience : int 1 19 15 9 8 13 27 24 10 9 ...  
## $ Income : int 49 34 11 100 45 29 72 22 81 180 ...  
## $ ZIP.Code : int 91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...  
## $ Family : int 4 3 1 1 4 4 2 1 3 1 ...  
## $ CCAvg : num 1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...  
## $ Education : int 1 1 1 2 2 2 2 3 2 3 ...  
## $ Mortgage : int 0 0 0 0 0 155 0 0 104 0 ...  
## $ Personal.Loan : int 0 0 0 0 0 0 0 0 0 1 ...  
## $ Securities.Account: int 1 1 0 0 0 0 0 0 0 0 ...  
## $ CD.Account : int 0 0 0 0 0 0 0 0 0 0 ...  
## $ Online : int 0 0 0 0 0 1 1 0 1 0 ...  
## $ CreditCard : int 0 0 0 0 1 0 0 1 0 0 ...

head(UB)

## 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

#Checking for Missing Values

colMeans(is.na(UB))

## ID Age Experience Income   
## 0 0 0 0   
## ZIP.Code Family CCAvg Education   
## 0 0 0 0   
## Mortgage Personal.Loan Securities.Account CD.Account   
## 0 0 0 0   
## Online CreditCard   
## 0 0

#Converting & Summary online variables

DF\_UB<-UB%>% select(Age,Experience,Income,Family,CCAvg,Education,Mortgage,Personal.Loan,Securities.Account,CD.Account,Online,CreditCard)  
  
DF\_UB$CreditCard <- as.factor(DF\_UB$CreditCard)  
summary(DF\_UB$CreditCard)

## 0 1   
## 3530 1470

is.factor(DF\_UB$CreditCard)

## [1] TRUE

DF\_UB$Personal.Loan <- as.factor((DF\_UB$Personal.Loan))  
summary(DF\_UB$Personal.Loan)

## 0 1   
## 4520 480

is.factor(DF\_UB$Personal.Loan)

## [1] TRUE

DF\_UB$Online <- as.factor(DF\_UB$Online)  
summary(DF\_UB$Online)

## 0 1   
## 2016 2984

is.factor(DF\_UB$Online)

## [1] TRUE

#split data 60% Training and 40% validation

selected.var <- c(8,11,12)  
set.seed(1)  
Train\_Index = createDataPartition(DF\_UB$Personal.Loan, p=0.60, list=FALSE)   
Train\_Data = DF\_UB[Train\_Index,selected.var]  
Validation\_Data = DF\_UB[-Train\_Index,selected.var]

#A.Pivot Table for credit card, Loan & Online

attach(Train\_Data)  
ftable(CreditCard,Personal.Loan,Online)

## Online 0 1  
## CreditCard Personal.Loan   
## 0 0 780 1126  
## 1 77 120  
## 1 0 303 503  
## 1 39 52

detach(Train\_Data)

The pivot table is now created with online as a column, Credit Card and LOAN as rows. #B) (probability not using Naive Bayes) With Online=1 and Credit Card=1, we can calculate the likelihood that Loan=1 by , we add 52(Loan=1 from ftable) and 503(Loan=0 from ftable) which gives us 555. Probability= 52/555 = 0.09369 or 9.36% . Hence the probability is 9.36%

prop.table(ftable(Train\_Data$CreditCard,Train\_Data$Online,Train\_Data$Personal.Loan),margin=1)

## 0 1  
##   
## 0 0 0.91015169 0.08984831  
## 1 0.90369181 0.09630819  
## 1 0 0.88596491 0.11403509  
## 1 0.90630631 0.09369369

The above table shows chances of geting a loan if you have a credit card and you apply online

#C.pivot table between personal loan and online , personal loan & credit card

attach(Train\_Data)  
ftable(Personal.Loan,Online)

## Online 0 1  
## Personal.Loan   
## 0 1083 1629  
## 1 116 172

ftable(Personal.Loan,CreditCard)

## CreditCard 0 1  
## Personal.Loan   
## 0 1906 806  
## 1 197 91

detach(Train\_Data)

The two pivot tables of above written as follows 1.In First pivot table: Online as a column & personal loan as row 2.In second Pivot table: Credit card as column & personal row as row

#D Propotion Pivot table

prop.table(ftable(Train\_Data$Personal.Loan,Train\_Data$CreditCard),margin=1)

## 0 1  
##   
## 0 0.7028024 0.2971976  
## 1 0.6840278 0.3159722

prop.table(ftable(Train\_Data$Personal.Loan,Train\_Data$Online),margin=1)

## 0 1  
##   
## 0 0.3993363 0.6006637  
## 1 0.4027778 0.5972222

The code above displays a proportion pivot table that can assist in answering question D. D1) 91/288 = 0.3159 or 31.59%  
D2) 172/288 = 0.5972 or 59.72% D3) total loans= 1 from table (288) is now divided by total count from table (3000) = 0.096 or 9.6% D4) 806/2712 = 0.2971 or 29.71% D5) 1629/2712 = 0.6006 or 60.06% D6) total loans=0 from table(2712) which is divided by total count from table (3000) = 0.904 or 90.4%

#E)Naive Bayes calculation (0.3159 \* 0.5972 \* 0.096)/[(0.3159 \* 0.5972 \* 0.096)+(0.2971 \* 0.6006 \* 0.904)] = 0.0528913646 or 5.29%

#F) Compare this value with the one obtained from the pivot table in (B). Which is a more accurate estimate? While E uses probability for each of the counts, B does a direct computation based on a count. As a result, B is more exact, but E is best for broad generality.

##G)Which of the entries in this table are needed for computing P(Loan = 1 | CC = 1, Online = 1)?Run naive Bayes on the data. Examine the model output on training data, and find the entrythat corresponds to P(Loan = 1 | CC = 1, Online = 1). Compare this to the number you obtained in (E).

Universal.nb <- naiveBayes(Personal.Loan ~ ., data = Train\_Data)  
Universal.nb

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## 0 1   
## 0.904 0.096   
##   
## Conditional probabilities:  
## Online  
## Y 0 1  
## 0 0.3993363 0.6006637  
## 1 0.4027778 0.5972222  
##   
## CreditCard  
## Y 0 1  
## 0 0.7028024 0.2971976  
## 1 0.6840278 0.3159722

While understanding how you’re computing P(LOAN=1|CC=1,Online=1) using the Naive Bayes model is made straightforward by utilizing the two tables created in step C, you can also rapidly compute P(LOAN=1|CC=1,Online=1) using the pivot table created in step B.

#NB confusion matrix for Train\_Data

pred.class <- predict(Universal.nb, newdata = Train\_Data)  
confusionMatrix(pred.class, Train\_Data$Personal.Loan)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2712 288  
## 1 0 0  
##   
## Accuracy : 0.904   
## 95% CI : (0.8929, 0.9143)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : 0.5157   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 1.000   
## Specificity : 0.000   
## Pos Pred Value : 0.904   
## Neg Pred Value : NaN   
## Prevalence : 0.904   
## Detection Rate : 0.904   
## Detection Prevalence : 1.000   
## Balanced Accuracy : 0.500   
##   
## 'Positive' Class : 0   
##

##Validation set

pred.prob <- predict(Universal.nb, newdata=Validation\_Data, type="raw")  
pred.class <- predict(Universal.nb, newdata = Validation\_Data)  
confusionMatrix(pred.class, Validation\_Data$Personal.Loan)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1808 192  
## 1 0 0  
##   
## Accuracy : 0.904   
## 95% CI : (0.8902, 0.9166)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : 0.5192   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 1.000   
## Specificity : 0.000   
## Pos Pred Value : 0.904   
## Neg Pred Value : NaN   
## Prevalence : 0.904   
## Detection Rate : 0.904   
## Detection Prevalence : 1.000   
## Balanced Accuracy : 0.500   
##   
## 'Positive' Class : 0   
##

#ROC

library(pROC)

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

##   
## Attaching package: 'pROC'

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

roc(Validation\_Data$Personal.Loan,pred.prob[,1])

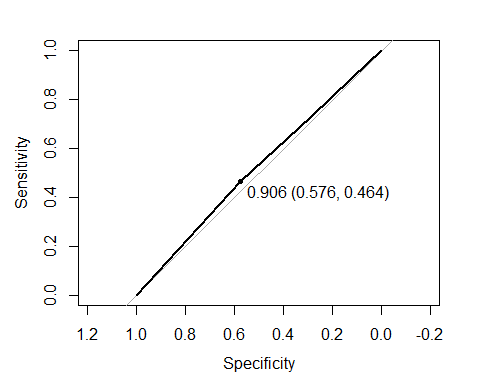
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##   
## Call:  
## roc.default(response = Validation\_Data$Personal.Loan, predictor = pred.prob[, 1])  
##   
## Data: pred.prob[, 1] in 1808 controls (Validation\_Data$Personal.Loan 0) < 192 cases (Validation\_Data$Personal.Loan 1).  
## Area under the curve: 0.5193

plot.roc(Validation\_Data$Personal.Loan,pred.prob[,1],print.thres="best")

## Setting levels: control = 0, case = 1  
## Setting direction: controls < cases

 Setting a threshold of 0.906 improves the model by decreasing sensitivity to 0.464 and improving specificity to 0.576. ```