Assignment 3

Priyanka Chillakuru

3/6/2022

```
UniversalBank <- read.csv("C:/Users/cpriy/Downloads/UniversalBank (1).csv")
summary(UniversalBank)
```

```
##        ID            Age          Experience       Income          ZIP.Code    
##  Min.   :   1   Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   : 9307  
##  1st Qu.:1251   1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:91911  
##  Median :2500   Median :45.00   Median :20.0   Median : 64.00   Median :93437  
##  Mean   :2500   Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :93153  
##  3rd Qu.:3750   3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:94608  
##  Max.   :5000   Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :96651  
##      Family          CCAvg          Education        Mortgage    
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0  
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0  
##  Median :2.000   Median : 1.500   Median :2.000   Median :  0.0  
##  Mean   :2.396   Mean   : 1.938   Mean   :1.881   Mean   : 56.5  
##  3rd Qu.:3.000   3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0  
##  Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :635.0  
##  Personal.Loan   Securities.Account   CD.Account         Online      
##  Min.   :0.000   Min.   :0.0000     Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.000   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :0.000   Median :0.0000     Median :0.0000   Median :1.0000  
##  Mean   :0.096   Mean   :0.1044     Mean   :0.0604   Mean   :0.5968  
##  3rd Qu.:0.000   3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000  
##  Max.   :1.000   Max.   :1.0000     Max.   :1.0000   Max.   :1.0000  
##    CreditCard   
##  Min.   :0.000  
##  1st Qu.:0.000  
##  Median :0.000  
##  Mean   :0.294  
##  3rd Qu.:1.000  
##  Max.   :1.000
```

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Warning in register(): Can't find generic `scale_type` in package ggplot2 to
## register S3 method.
```

```
## Loading required package: lattice
```

```r
library(ISLR)
library(e1071)
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(class)
library(reshape2)
library(ggplot2)
library(gmodels)
library(lattice)

#converting variables
UniversalBank$Personal.Loan <- factor(UniversalBank$Personal.Loan)
UniversalBank$Online <- factor(UniversalBank$Online)
UniversalBank$CreditCard <- factor(UniversalBank$CreditCard)
df= UniversalBank

#TASK1
set.seed(64060)
Train_index <- createDataPartition(df$Personal.Loan, p = 0.6, list = FALSE)
train.df = df[Train_index,]
validation.df = df[-Train_index,]

mytable <- xtabs(~ CreditCard + Online + Personal.Loan , data = train.df)
ftable(mytable)
```

```
##                      Personal.Loan    0    1
## CreditCard Online
## 0          0                        772   75
##            1                       1152  120
## 1          0                        309   34
##            1                        479   59
```

```r
#TASK2

probability = 59/(59+479)
probability
```

```
## [1] 0.1096654
```

```r
#TASK3

table(Personal.Loan = train.df$Personal.Loan, Online = train.df$Online)
```

```
##              Online
## Personal.Loan    0    1
##             0 1081 1631
##             1  109  179
```

```r
table(Personal.Loan = train.df$Personal.Loan, CreditCard = train.df$CreditCard)
```

```
##              CreditCard
## Personal.Loan    0    1
##             0 1924  788
##             1  195   93
```

```r
table(Personal.Loan = train.df$Personal.Loan)
```

```
## Personal.Loan
##    0    1
## 2712  288
```

```r
#TASK4

#i. P(CC = 1 | Loan = 1) (the proportion of credit card holders among the loan
#acceptors)
Probablity1 <- 93/(93+195)
Probablity1
```

```
## [1] 0.3229167
```

```r
#ii. P(Online = 1 | Loan = 1)
Probablity2 <- 179/(179+109)
Probablity2
```

```
## [1] 0.6215278
```

```r
#iii. P(Loan = 1) (the proportion of loan acceptors)
Probablity3 <- 288/(288+2712)
Probablity3
```

```
## [1] 0.096
```

```r
#iv. P(CC = 1 | Loan = 0)
Probablity4 <- 788/(788+1924)
Probablity4
```

```
## [1] 0.2905605
```

```r
#v. P(Online = 1 | Loan = 0)
Probablity5 <- 1631/(1631+1081)
Probablity5
```

```
## [1] 0.6014012
```

```r
#vi. P(Loan = 0)
Probablity6 <- 2712/(2712+288)
Probablity6
```

```
## [1] 0.904
```

```r
#TASK5

Task5Probablity <- (Probablity1*Probablity2*Probablity3)/
((Probablity1*Probablity2*Probablity3) +(Probablity4*Probablity5*Probablity6))

Task5Probablity
```

```
## [1] 0.1087106
```

```r
#TASK6

#Compare this value with the one obtained from the pivot table in (B). Which is a more
#accurate estimate?

##Value we got from question 2 was 0.1096654 and in the question 5  is 0.1087106 are almost same.
#The only difference #between by the exact method and naive bayes method is the exact method
#would need the exact same #independent variable #classification to predict, whereas the naive bayes
#method does not. We can confirm that the #value get from the #question 2 is more accurate
#since we have taken the exact values from the pivot table.

#Task7

#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 entry
#that corresponds to P(Loan = 1 | CC = 1, Online = 1). Compare this to the number you
#obtained in (E).

nb.model <- naiveBayes(Personal.Loan~ Online + CreditCard, data = train.df)
To_Predict=data.frame(Online=1, CreditCard= 1)
predict(nb.model, To_Predict,type = 'raw')
```

```
## Warning in predict.naiveBayes(nb.model, To_Predict, type = "raw"): Type mismatch
## between training and new data for variable 'Online'. Did you use factors with
## numeric labels for training, and numeric values for new data?
```

```
## Warning in predict.naiveBayes(nb.model, To_Predict, type = "raw"): Type mismatch
## between training and new data for variable 'CreditCard'. Did you use factors
## with numeric labels for training, and numeric values for new data?
```

```
##               0          1
## [1,] 0.9153656 0.08463445
```

```
#The value we got from question 7  is 0.08463445 and value derived from the task 5 is 0.1087106.
# the result is almost same that we got from Task5.
# There is only a minute difference because of the rounding.
#The difference will not effect the rank order of the output.
```