FML ASSIGNMENT 2

Rohit Vuradi

##Installing Required packages,Calling a library and loading the dataset universal bank data csv file.

```
#install.packages("tidyverse")
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
#install.packages("reshape")
library(reshape)
```

```
##
## Attaching package: 'reshape'
##
## The following object is masked from 'package:dplyr':
##
##     rename
##
## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
```

```
#install.packages("caret")
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
#install.packages("e1071")
library(e1071)
UB<- read_csv("C:/Users/girne/Downloads/UniversalBank - Copy.csv")
```

```
## Rows: 5000 Columns: 14
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (14): ID, Age, Experience, Income, ZIP Code, Family, CCAvg, Education, M...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(UB)
```

```
## # A tibble: 6 x 14
##      ID   Age Experience Income `ZIP Code` Family CCAvg Educat~1 Mortg~2 Perso~3
##   <dbl> <dbl>      <dbl>  <dbl>      <dbl>  <dbl> <dbl>    <dbl>   <dbl>   <dbl>
## 1     1    25          1     49      91107      4   1.6        1       0       0
## 2     2    45         19     34      90089      3   1.5        1       0       0
## 3     3    39         15     11      94720      1   1          1       0       0
## 4     4    35          9    100      94112      1   2.7        2       0       0
## 5     5    35          8     45      91330      4   1          2       0       0
## 6     6    37         13     29      92121      4   0.4        2     155       0
## # ... with 4 more variables: `Securities Account` <dbl>, `CD Account` <dbl>,
## #   Online <dbl>, CreditCard <dbl>, and abbreviated variable names
## #   1: Education, 2: Mortgage, 3: `Personal Loan`
```

```r
tail(UB)
```

```
## # A tibble: 6 x 14
##      ID   Age Experience Income `ZIP Code` Family CCAvg Educat~1 Mortg~2 Perso~3
##   <dbl> <dbl>      <dbl>  <dbl>      <dbl>  <dbl> <dbl>    <dbl>   <dbl>   <dbl>
## 1  4995    64         40     75      94588      3   2          3       0       0
## 2  4996    29          3     40      92697      1   1.9        3       0       0
## 3  4997    30          4     15      92037      4   0.4        1      85       0
## 4  4998    63         39     24      93023      2   0.3        3       0       0
## 5  4999    65         40     49      90034      3   0.5        2       0       0
## 6  5000    28          4     83      92612      3   0.8        1       0       0
## # ... with 4 more variables: `Securities Account` <dbl>, `CD Account` <dbl>,
## #   Online <dbl>, CreditCard <dbl>, and abbreviated variable names
## #   1: Education, 2: Mortgage, 3: `Personal Loan`
```

```r
colnames(UB)
```

```
##  [1] "ID"                 "Age"                "Experience"
##  [4] "Income"             "ZIP Code"           "Family"
##  [7] "CCAvg"              "Education"          "Mortgage"
## [10] "Personal Loan"      "Securities Account" "CD Account"
## [13] "Online"             "CreditCard"
```

#Transforming data into factors (categorical).

```
UB$`Personal Loan` = as.factor(UB$`Personal Loan`)
UB$Online = as.factor(UB$Online)
UB$CreditCard = as.factor(UB$CreditCard)
```

#Spliting the data into two the 60% of data in training set and 40% into validation set

```
set.seed(456)
UB.train.data <- sample(row.names(UB), 0.6*dim(UB)[1]) # 60 % training
UB.valid.data <- setdiff(row.names(UB), UB.train.data) # 40 % validation
UB.train <- UB[UB.train.data, ] # assigning the UB.train.data into data frame
UB.valid <- UB[UB.valid.data, ] # assigning the validation index into data frame
train <- UB[UB.train.data, ] # Duplicating the data frame UB.train
valid = UB[UB.train.data,] # Duplicating the data frame UB.valid
```

#A. Create a pivot table for the training data with Online as a column variable, CC as a row variable, and Loan as a secondary row variable. The values inside the table should convey the count. In R use functions melt() and cast(), or function table().

Pivot table

```
#install.packages("reshape2")
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following objects are masked from 'package:reshape':
##
##     colsplit, melt, recast

## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
melt = melt(train,id=c("CreditCard","Personal Loan"),variable= "Online")# to organize the data
```

```
## Warning: attributes are not identical across measure variables; they will be
## dropped
```

```
cast = dcast(melt,CreditCard+`Personal Loan`~Online) # dcast is the process of turning online, personal
```

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

```
cast[,c(1,2,3,14)] # Casting column number 14: Personal loan, ID, and credit card, respectively
```

```
##   CreditCard Personal Loan   ID Online
## 1          0             0 1917   1917
## 2          0             1  200    200
## 3          1             0  794    794
## 4          1             1   89     89
```

#B. Consider the task of classifying a customer who owns a bank credit card and is actively using online banking services. Looking at the pivot table, what is the probability that this customer will accept the loan offer? [This is the probability of loan acceptance (Loan = 1) conditional on having a bank credit card (CC = 1) and being an active user of online banking services (Online = 1)].

```
UB.Loan.CC1 <- 89/3000#According to the pivot table, the value for Loan is 89, and the value for CC is
UB.Loan.CC1 # which is 29 %.
```

```
## [1] 0.02966667
```

#C. Create two separate pivot tables for the training data. One will have Loan (rows) as a function of Online (columns) and the other will have Loan (rows) as a function of CC.

```
melt1 = melt(train,id=c("Personal Loan"),variable = "Online") # Melting Personal loan and Online data i
```

```
## Warning: attributes are not identical across measure variables; they will be
## dropped
```

```
melt2 = melt(train,id=c("CreditCard"),variable = "Online") # CREDIT CARD DATA MELTING WITH REFERENCE TO
```

```
## Warning: attributes are not identical across measure variables; they will be
## dropped
```

```
cast1 =dcast(melt1,`Personal Loan`~Online) # Casting Personal loan and online values
```

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

```
cast2=dcast(melt2,CreditCard~Online) # Casting Personal loan and online values
```

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

```
UB.Loanonline=cast1[,c(1,13)]
UB.LoanCC = cast2[,c(1,14)]
UB.Loanonline #shows the number of personal loans in reference to online
```

```
##   Personal Loan Online
## 1             0   2711
## 2             1    289
```

```
UB.LoanCC # shows the number of credit cards in reference to internet.
```

```
##   CreditCard Online
## 1          0   2117
## 2          1    883
```

D. Compute the following quantities [P (A | B) means "the probability of A given B"]: 1.P (CC = 1 | Loan = 1) (the proportion of credit card holders among the loan acceptors) 2.P(Online=1|Loan=1) 3.P (Loan = 1) (the proportion of loan acceptors) 4.P(CC=1|Loan=0) 5.P(Online=1|Loan=0) 6.P(Loan=0)

```r
table(train[,c(14,10)])   # creating a pivot table with the columns 14 and 10 representing personal loa
```

```
##           Personal Loan
## CreditCard    0    1
##          0 1917  200
##          1  794   89
```

```r
table(train[,c(13,10)])   # Creating a pivot table for column 13 and 10 which is online and personal lo
```

```
##       Personal Loan
## Online    0    1
##      0 1046  112
##      1 1665  177
```

```r
table(train[,c(10)])     # Personal loan pivot table There are 2725 and 275 from training, respectively
```

```
## Personal Loan
##    0    1
## 2711  289
```

1. P(CC = 1 | Loan = 1)

```r
UB.CCUB.Loan1 = 89/(89+200)   # We can obtain the CC= 1 and Loan = 1 values by referring to the above p
UB.CCUB.Loan1
```

```
## [1] 0.3079585
```

2. P(Online=1|Loan=1)

```r
UB.ONUB.Loan1 =177/(177+112) # We can get the online = 1 and loan = 1 values from the pivot table above
UB.ONUB.Loan1
```

```
## [1] 0.6124567
```

3. P(Loan = 1)

```r
UB.Loan1 =289/(289+2711) # By referring the above pivot table we can get the Loan = 1
UB.Loan1
```

```
## [1] 0.09633333
```

4. P(CC=1|Loan=0)

```r
UB.CCLoan.01= 794/(794+1917) #Using the pivot table above, we can obtain the CC = 1 and Loan = 0 values
UB.CCLoan.01
```

```
## [1] 0.2928809
```

5. P(Online=1|Loan=0)

```r
UB.ON1.L0= 1665/(1665+1046) # We can get the online = 1 and loan = 0 values from the pivot table above.
UB.ON1.L0
```

```
## [1] 0.6141645
```

6. P(Loan=0)

```r
UB.Loan0= 2711/(2711+289) # We can obtain the Loan = 0 values by  the pivot table above.
UB.Loan0
```

```
## [1] 0.9036667
```

E. Use the quantities computed above to compute the naive Ba1 probability P(Loan = 1 | CC = 1, Online = 1).

```r
UB.Naivebayes = ((89/(89+200))*(177/(177+112))*(289/(289+2711)))/(((89/(89+200))*(177/(177+112))*(289/(2
UB.Naivebayes # 100 % is the probability
```

```
## [1] 0.1005407
```

F. Compare this value with the one obtained from the pivot table in (b). Which is a more accurate estimate? 9.05% are very similar to the 9.7% the difference between the exact method and the naive-baise method is the exact method would need the the exact same independent variable classifications to predict, where the naive bayes method does not.

```r
library(caret)
library(e1071)
UB.nb.train = UB.train[,c(10,13,14)] # Personal loan, credit card, and online column training dataUB.nb
UB.naivebayes.1 = naiveBayes(`Personal Loan`~.,data=UB.nb.train) #Using the naivebayes algorithm to per
UB.naivebayes.1
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##          0          1
## 0.90366667 0.09633333
##
## Conditional probabilities:
##    Online
## Y           0         1
##   0 0.3858355 0.6141645
##   1 0.3875433 0.6124567
##
##    CreditCard
## Y           0         1
##   0 0.7071191 0.2928809
##   1 0.6920415 0.3079585
```