Assignment-3

Gowthami Pakanati

03-10-2024

```
## Load required libraries
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.3.2
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.3.2

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.3.2

## Loading required package: lattice
```

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.3.2
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2

## Warning: package 'purrr' was built under R version 4.3.2

## Warning: package 'lubridate' was built under R version 4.3.2

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(reshape)
```

```
## Warning: package 'reshape' was built under R version 4.3.3
```

```
##
## Attaching package: 'reshape'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
##
## The following object is masked from 'package:dplyr':
##
##     rename
##
## The following objects are masked from 'package:tidyr':
##
##     expand, smiths
##
## The following objects are masked from 'package:reshape2':
##
##     colsplit, melt, recast
```

```
## Load the data csv file
Universal_Bank <- read.csv("C:\\Users\\pakan\\Desktop\\FML\\UniversalBank.csv")

head(Universal_Bank)
```

```
##   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
```

```
#here we are transforming the data

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

```
##   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 Personal Loan
## 1             0                  1          0      0          0             0
## 2             0                  1          0      0          0             0
## 3             0                  0          0      0          0             0
## 4             0                  0          0      0          0             0
## 5             0                  0          0      0          1             0
## 6             0                  0          0      1          0             0
```

```
set.seed(456)

Uni_train<- sample(row.names(Universal_Bank), 0.6*dim(Universal_Bank)[1])
Uni_valid<- setdiff(row.names(Universal_Bank), Uni_train)


Uni_training <- Universal_Bank[Uni_train, ]
Uni_validating <- Universal_Bank[Uni_valid, ]

train <- Universal_Bank[Uni_train, ]
valid <- Universal_Bank[Uni_train,]
```

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

dataframe methods melt() and pivot().

```
#melt data
melt_data = melt(train,id=c("CreditCard","Personal.Loan"),variable= "Online")

cast_data <- dcast(melt_data, CreditCard + Personal.Loan ~ value, fun.aggregate = length)

cast_data[,c(1,2,3,14)]
```

```
##   CreditCard Personal.Loan -1 0.7
## 1          0             0 14  72
## 2          0             1  0   0
## 3          1             0  5  31
## 4          1             1  0   2
```

#B. 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)]

```
loan_cc<- 89/3000

loan_cc
```

```
## [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.

```
## Transforming the train data frame into a long format, with "Online" as the variable to be melted and


melted1 = melt(train,id=c("Personal.Loan"),variable = "Online")

 ## Transforming the train data frame into a long format, with "Credit Card" as the identifier and "Onl

melted2 = melt(train,id=c("CreditCard"),variable = "Online")
```

```
# Exploring personal loan options and online resources

casting1=dcast(melted1,`Personal.Loan`~Online)
```

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

```
# Exploring personal loan and online options

casting2=dcast(melted2,CreditCard~Online)
```

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

```
#displays the quantity of personal loans compared to online
online_loans=casting1[,c(1,13)]
loancc= casting2[,c(1,14)]
online_loans
```

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

```
# The quantity of credit cards in relation to online is indicated by #.
loancc
```

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

```r
# Creating a pivot table with personal loans displayed in columns 14 and 10.

table(train[,c(14,10)])
```

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

```r
# here we are Making a pivot table for the online and personal loan columns 13 and 10.
table(train[,c(13,10)])
```

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

```r
# Pivot table for personal loans There are, from training, 2725 and 275, respectively.
table(train[,c(10)])
```

```
##
##    0    1
## 2711  289
```

```r
## here are  consulting the above p, we may determine the CC= 1 and Loan = 1 values.
cc_univloan= 89/(89+200)
cc_univloan
```

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

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

```r
##The pivot table above UB.ONUB.Loan1 gives us the data for online = 1 and loan = 1.
ONUni_Loan1 =177/(177+112)
ONUni_Loan1
```

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

3. P(Loan = 1)

```r
 # By referring the above pivot table we can get the Loan = 1
Uni_Loan1 =289/(289+2711)
Uni_Loan1
```

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

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

```
#The CC = 1 and Loan = 0 values can be obtained by using the pivot table above.
Uni_01= 794/(794+1917)
Uni_01
```

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

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

```
# The data in the pivot table above shows the values for online = 1 and loan = 0.

Uni_BankL0= 1665/(1665+1046)
Uni_BankL0
```

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

6. P(Loan=0)

```
# The pivot table above enables us to isolate the Loan = 0 values.

Uni_Loan0= 2711/(2711+289)
Uni_Loan0
```

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

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

```
# Given probabilities
Probability_CC_Loan1 <- 0.096
Probability_Online_Loan1 <- 0.833
P_loan <- 0.0125

# Calculate Naive Bayes probability P(Loan = 1 | CC = 1, Online = 1)
Universal_Bank_Naivebayes <- (Probability_CC_Loan1)*(Probability_Online_Loan1)*(P_loan)
Universal_Bank_Naivebayes
```

```
## [1] 0.0009996
```

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

```
# Naive Bayes Probability (from calculation in E)
naive_bayes_probability <- 0.0009996

# Pivot Table Probability
pivot_table_probability <- 0.02966667

# Compare the odds and printing a message stating which is more likely to be true.
```

```
if (naive_bayes_probability > pivot_table_probability) {
  message("Naive Bayes Probablity is more accurate: ", naive_bayes_prob)
} else if (naive_bayes_probability < pivot_table_probability) {
  message("Pivot Table Probability is more accurate: ", pivot_table_probability)
} else {
  message("Both Prob are the same: ", naive_bayes_probability)
}
```

```
## Pivot Table Probability is more accurate: 0.02966667
```

after clearly observing the two , we can say that here the pivot table probability is more accurate than the Naive Bayes probability.

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

```
names(Universal_Bank)
```

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

```
names(Uni_training)
```

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

```
# Choose the pertinent training columns.
Universal_Bank.train <- Universal_Bank[, c("CreditCard", "Online", "Personal Loan")]

#Change the columns' names to eliminate gaps
colnames(Universal_Bank.train) <- c("CreditCard", "Online", "PersonalLoan")

# Converting  "Online" and "Credit Card" to factors with the proper levels
Universal_Bank.train$CreditCard <- factor(Universal_Bank.train$CreditCard, levels = c(0, 1), labels = c
Universal_Bank.train$Online <- factor(Universal_Bank.train$Online, levels = c(0, 1), labels = c("No", "
```

```
# Training the naive Bayes model
nb_model <- naiveBayes(PersonalLoan ~ CreditCard + Online, data = Universal_Bank.train)

# Predicting probabilities for the training data
```

```r
pred_prob <- predict(nb_model, Universal_Bank.train, type = "raw")

# Getting the index of the column corresponding to "Loan = 1"
loan_1_index <- which(colnames(pred_prob) == "1")

# Filtering the rows where Credit Card = 1 and Online = 1
filtered_data <- Universal_Bank.train[Universal_Bank.train$CreditCard == "Yes" & Universal_Bank.train$On

# Calculating the conditional probability for P(Loan = 1 | CC = 1, Online = 1)
prob_loan_1_given_CC1_Online1 <- mean(pred_prob[as.numeric(rownames(filtered_data)), loan_1_index])

# Print the result
print(prob_loan_1_given_CC1_Online1)
```

```
## [1] 0.09881706
```

Interpretation: For Question E, i have calculated the naive Bayes probability using the given probabilities. This approach assumes that the features (Credit Card and Online) are independent when considering the target variable (Personal Loan).

For Question G, I applied the naive Bayes algorithm to understand the conditional probability distribution from the data. This approach considers the real distribution of the features and the target variable, without making assumptions of independence.

Comparing the results:

1. The result obtained from question E (0.0009996) is a straightforward computation using the given probabilities.

2.The answer to question G is not given in the question but can be obtained by using the trained naive Bayes model on the data.

3.To sum up, approach E simplifies by assuming feature independence, whereas approach G learns the conditional probability distribution from the actual data. Answers from question G are generally more dependable since they are derived from actual data rather than assumptions.