BA 64060 Assignment 3”

Kyle Breeden

2025-10-12

## R Markdown

#rm(list = ls())  
#Use readr package for faster and more efficient reading  
#install.packages("readr") #not needed if already installed  
library(readr)

## Warning: package 'readr' was built under R version 4.5.1

assign3\_dataset <- read\_csv("C:/Users/kylej/OneDrive/Documents/BA 64060/Assignment 3/Data/UniversalBank.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.

#Use Spec function to see column specifications  
spec(assign3\_dataset)

## cols(  
## ID = col\_double(),  
## Age = col\_double(),  
## Experience = col\_double(),  
## Income = col\_double(),  
## `ZIP Code` = col\_double(),  
## Family = col\_double(),  
## CCAvg = col\_double(),  
## Education = col\_double(),  
## Mortgage = col\_double(),  
## `Personal Loan` = col\_double(),  
## `Securities Account` = col\_double(),  
## `CD Account` = col\_double(),  
## Online = col\_double(),  
## CreditCard = col\_double()  
## )

# create a subset to focus on 2 predictors (Online & Credit card) and the outcome (Personal Loan) per the directions  
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(caret)

## Loading required package: ggplot2

## Loading required package: lattice

ubank\_df <- assign3\_dataset %>% select(Online, CreditCard, `Personal Loan`)  
#partition data into training (60%) and validation (40%) sets  
set.seed(15) #can be any integer but must stay consistent so I can reproduce results  
Index\_Train <- createDataPartition(ubank\_df$`Personal Loan`,p = 0.6, list=FALSE)   
Train <- ubank\_df[Index\_Train, ]  
Validation <- ubank\_df[-Index\_Train, ]

#use functions melt() and cast() to create a pivot table with online as column variable, CC as row variable, and Loan as secondary row variable.  
#The values inside the table should convey the count  
#install.packages("reshape2")  
library(reshape2)

## Warning: package 'reshape2' was built under R version 4.5.1

trainmelt <- melt(Train, id.vars = c("CreditCard", "Online"), measure.vars = "Personal Loan", variable.name = "Measure", value.name = "Personal Loan")  
#using dcast instead of cast because it is in the reshape2 package  
pivot1 <- dcast(trainmelt, CreditCard + `Personal Loan` ~ Online, fun.aggregate = length)

## Using Personal Loan as value column: use value.var to override.

pivot1

## CreditCard Personal Loan 0 1  
## 1 0 0 785 1118  
## 2 0 1 79 130  
## 3 1 0 309 489  
## 4 1 1 40 50

#classifying a customer who owns a bank credit card and is actively using online banking services  
#using pivot1, what is the probability that this customer will accept the loan offer  
prob\_loan\_when\_cc1\_online1 <- 50 / (489 + 50)   
prob\_loan\_when\_cc1\_online1

## [1] 0.09276438

#Customers who have a bank credit card and use online banking have about a 9.3% chance of accepting the personal loan offer.

#Create pivot table with Loan (rows) as a function of Online (columns)  
trainmelt1 <- melt(Train, id.vars = "Online", measure.vars = "Personal Loan", variable.name = "Measure", value.name = "Personal Loan")  
  
pivot2 <- dcast(trainmelt1, `Personal Loan` ~ Online, fun.aggregate = length)

## Using Personal Loan as value column: use value.var to override.

pivot2

## Personal Loan 0 1  
## 1 0 1094 1607  
## 2 1 119 180

#Create pivot table with Loan (rows) as a function of CC.  
trainmelt2 <- melt(Train, id.vars = "CreditCard", measure.vars = "Personal Loan", variable.name = "Measure", value.name = "Personal Loan")  
  
Pivot3 <- dcast(trainmelt2, `Personal Loan` ~ CreditCard, fun.aggregate = length)

## Using Personal Loan as value column: use value.var to override.

Pivot3

## Personal Loan 0 1  
## 1 0 1903 798  
## 2 1 209 90

#i. P(CC = 1 | Loan = 1)  
prob\_cc1\_when\_loan1 <- 90 / (209 + 90)   
prob\_cc1\_when\_loan1

## [1] 0.3010033

percent\_cc1\_when\_loan1 <- paste(round(prob\_cc1\_when\_loan1 \* 100, 1), "%")  
percent\_cc1\_when\_loan1

## [1] "30.1 %"

#ii. P(Online = 1 | Loan = 1)  
prob\_Online1\_when\_Loan1 <- 180 / (119 + 180)   
prob\_Online1\_when\_Loan1

## [1] 0.6020067

percent\_Online1\_when\_Loan1 <- paste(round(prob\_Online1\_when\_Loan1 \* 100, 1), "%")  
percent\_Online1\_when\_Loan1

## [1] "60.2 %"

#iii. P(Loan = 1) (the proportion of loan acceptors)  
prob\_loan1 <- 299 / 3000  
prob\_loan1

## [1] 0.09966667

percent\_loan1 <- paste(round(prob\_loan1 \* 100, 1), "%")  
percent\_loan1

## [1] "10 %"

#iV. P(CC = 1 | Loan = 0)  
prob\_cc1\_when\_loan0 <- 798 / (1903 + 798)  
prob\_cc1\_when\_loan0

## [1] 0.2954461

percent\_cc1\_when\_loan0 <- paste(round(prob\_cc1\_when\_loan0 \* 100, 1), "%")  
percent\_cc1\_when\_loan0

## [1] "29.5 %"

#v. P(Online = 1 | Loan = 0)  
prob\_Online1\_when\_Loan0 <- 1607 / (1094 + 1607)  
prob\_Online1\_when\_Loan0

## [1] 0.5949648

percent\_Online1\_when\_Loan0 <- paste(round(prob\_Online1\_when\_Loan0 \* 100, 1), "%")  
percent\_Online1\_when\_Loan0

## [1] "59.5 %"

#vi. P(Loan = 0)  
prob\_loan0 <- 2701 / 3000  
prob\_loan0

## [1] 0.9003333

percent\_loan0 <-paste(round(prob\_loan0 \* 100, 1), "%")  
percent\_loan0

## [1] "90 %"

#Use the quantities computed above to compute the naive Bayes probability P(Loan = 1 | CC = 1, Online = 1)  
numerator <- 0.301 \* 0.602 \* 0.0997  
denominator <- numerator + (0.295 \* 0.595 \* 0.9003)  
nb\_model <- numerator / denominator  
nb\_model

## [1] 0.1025938

percent\_nb\_model <- paste(round(nb\_model \* 100, 3), "%")  
percent\_nb\_model

## [1] "10.259 %"

#Compare naive Bayes probability & pivot table probability for P(Loan = 1 | CC = 1, Online = 1) #10.3% vs. 9.3% #Which is more aaccurate? #The pivot-table estimate (~9.3%) is a more accurate estimate because it does not rely on the Naive Bayes independence assumption between CreditCard and Online given Loan.

#question G seems like a repeat of what is above, but I will use it as an opportunity to run nb with the e1017 package  
library(e1071)

## Warning: package 'e1071' was built under R version 4.5.1

#transform the variables into factor  
Train$`Personal Loan` <- factor(Train$`Personal Loan`, levels = c(0,1))  
Train$Online <- factor(Train$Online, levels = c(0,1))  
Train$CreditCard <- factor(Train$CreditCard, levels = c(0,1))  
  
nb\_model2 <- naiveBayes(`Personal Loan` ~ Online + CreditCard, data = Train)  
nb\_model2

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## 0 1   
## 0.90033333 0.09966667   
##   
## Conditional probabilities:  
## Online  
## Y 0 1  
## 0 0.4050352 0.5949648  
## 1 0.3979933 0.6020067  
##   
## CreditCard  
## Y 0 1  
## 0 0.7045539 0.2954461  
## 1 0.6989967 0.3010033

Predicted\_Test\_labels <-predict(nb\_model2,Train)  
library("gmodels")

## Warning: package 'gmodels' was built under R version 4.5.1

CrossTable(x=Train$"Personal Loan",y=Predicted\_Test\_labels, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 3000   
##   
##   
## | Predicted\_Test\_labels   
## Train$"Personal Loan" | 0 | Row Total |   
## ----------------------|-----------|-----------|  
## 0 | 2701 | 2701 |   
## | 0.900 | |   
## ----------------------|-----------|-----------|  
## 1 | 299 | 299 |   
## | 0.100 | |   
## ----------------------|-----------|-----------|  
## Column Total | 3000 | 3000 |   
## ----------------------|-----------|-----------|  
##   
##

Predicted\_Test\_labels\_raw <- predict(nb\_model2,Train, type = "raw")  
head(Predicted\_Test\_labels\_raw)

## 0 1  
## [1,] 0.9025946 0.09740545  
## [2,] 0.9025946 0.09740545  
## [3,] 0.9002349 0.09976507  
## [4,] 0.8999876 0.10001244  
## [5,] 0.9025946 0.09740545  
## [6,] 0.9025946 0.09740545

#find the first row where CC=1 and Online = 1   
idx <- with(Train, CreditCard == 1 & Online == 1)  
i <- which(idx)[1]  
Predicted\_Test\_labels\_raw[i, "1"]

## 1   
## 0.1024281

#$check to see if a manual calculation matches  
numerator2 <- 0.3010033 \* 0.6020067 \* 0.09966667   
denominator2 <- numerator2 + (0.2954461 \* 0.5949648 \* 0.90033333)  
nb\_model3 <- numerator2 / denominator2  
nb\_model3

## [1] 0.1024281

percent\_nb\_model3 <- paste(round(nb\_model3 \* 100, 1), "%")  
percent\_nb\_model3

## [1] "10.2 %"

#This number (0.1024281) is nearly the same as the number calculated in E (0.1025938).