Assignment 3

Dillon Steiger

2025-10-10

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

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(tidyr)

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

library(caret)

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

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.5.1

## Loading required package: lattice

library(e1071)

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

##   
## Attaching package: 'e1071'

## The following object is masked from 'package:ggplot2':  
##   
## element

ub\_raw<- read.csv("C:/Users/steig/Downloads/UniversalBank.csv")  
  
names(ub\_raw)

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

ub<- ub\_raw %>% transmute(  
 Loan = Personal.Loan,  
 CC = CreditCard,  
 Online = Online  
)  
  
#train/validation split 60/40  
set.seed(123)  
in\_train<- createDataPartition(ub$Loan, p = 0.60, list = FALSE)  
train<- ub[in\_train, ]  
valid<- ub[-in\_train, ]  
  
cat("Train dims:", paste(dim(train), collapse = " x "), "\n")

## Train dims: 3000 x 3

cat("Valid dims:", paste(dim(valid), collapse = " x "), "\n")

## Valid dims: 2000 x 3

cat("Training class proportions (Loan):\n")

## Training class proportions (Loan):

print(prop.table(table(train$Loan)))

##   
## 0 1   
## 0.90733333 0.09266667

#Part A  
#pivot table: Online (cols) x CC (rows)  
train\_fac<- train %>% mutate(across(c(Loan, CC, Online), ~ factor(.x, levels = c(0,1))))  
  
tab\_A<- xtabs(~ CC + Loan + Online, data = train\_fac)  
cat("\n(A) 3-way table: CC x Loan X Online (counts)\n")

##   
## (A) 3-way table: CC x Loan X Online (counts)

print(tab\_A)

## , , Online = 0  
##   
## Loan  
## CC 0 1  
## 0 785 65  
## 1 317 34  
##   
## , , Online = 1  
##   
## Loan  
## CC 0 1  
## 0 1145 122  
## 1 475 57

cat("\n(A) Pretty pivot (rows: CC,Loan; cols: Online)\n")

##   
## (A) Pretty pivot (rows: CC,Loan; cols: Online)

pretty\_A<- as.data.frame(tab\_A) %>% arrange(CC, Loan, Online) %>%  
 pivot\_wider(names\_from = Online, values\_from = Freq, names\_prefix = "Online\_") %>%  
 arrange(CC, Loan)  
print(pretty\_A)

## # A tibble: 4 × 4  
## CC Loan Online\_0 Online\_1  
## <fct> <fct> <int> <int>  
## 1 0 0 785 1145  
## 2 0 1 65 122  
## 3 1 0 317 475  
## 4 1 1 34 57

#Part B  
#P(Loan+1 | CC=1, Online=1)  
num\_B<- tab\_A["1", "1", "1"]  
den\_B<- tab\_A["1", "0", "1"] + tab\_A["1", "1", "1"]  
piv\_prob<- as.numeric(num\_B / den\_B)  
cat("\n(B) P(Loan=1 | CC=1, Online=1) from pivot =", piv\_prob, "\n")

##   
## (B) P(Loan=1 | CC=1, Online=1) from pivot = 0.1071429

#Part C  
#Tables (Loan x Online), (Loan x CC)  
tab\_L\_online<- table(Loan = train$Loan, Online = train$Online)  
tab\_L\_cc<- table(Loan = train$Loan, CC = train$CC)  
  
cat("\n(C) Loan x Online (counts)\n")

##   
## (C) Loan x Online (counts)

print(tab\_L\_online)

## Online  
## Loan 0 1  
## 0 1102 1620  
## 1 99 179

cat("\n(C) Loan x CC (counts)\n")

##   
## (C) Loan x CC (counts)

print(tab\_L\_cc)

## CC  
## Loan 0 1  
## 0 1930 792  
## 1 187 91

#Part D  
#P(CC=1 | Loan=1)  
p\_i<- tab\_L\_cc["1", "1"] / sum(tab\_L\_cc["1", ])  
  
#P(Online=1 | Loan=1)  
p\_ii<- tab\_L\_online["1", "1"] / sum(tab\_L\_online["1", ])  
  
#P(Loan=1)  
p\_iii<- mean(train$Loan == 1)  
  
#P(CC=1 | Loan=0)  
p\_iv<- tab\_L\_cc["0", "1"] / sum(tab\_L\_cc["0", ])  
  
#P(Online=1 | Loan=0)  
p\_v<- tab\_L\_online["0", "1"] / sum(tab\_L\_online["0", ])  
  
#P(Loan=0)  
p\_vi<- 1 - p\_iii  
  
cat("\n(D) Requested Probabilities:\n")

##   
## (D) Requested Probabilities:

print(c(`i = P(CC=1 | Loan=1)` = p\_i,  
 `ii = P(Online=1 | Loan=1)` = p\_ii,  
 `iii = P(Loan=1)` = p\_iii,  
 `iv = P(CC=1 | Loan=0)` = p\_iv,  
 `v = P(Online=1 | Loan=0)` = p\_v,  
 `vi = P(Loan=0)` = p\_vi))

## i = P(CC=1 | Loan=1) ii = P(Online=1 | Loan=1) iii = P(Loan=1)   
## 0.32733813 0.64388489 0.09266667   
## iv = P(CC=1 | Loan=0) v = P(Online=1 | Loan=0) vi = P(Loan=0)   
## 0.29096253 0.59515062 0.90733333

#Part E  
num1<- p\_i \* p\_ii \* p\_iii  
num0<- p\_iv \* p\_v \* p\_vi  
nb\_prob\_from\_counts<- as.numeric(num1 / (num1 + num0))  
cat("\n(E) Naive Bayes P(Loan=1 | CC=1, Online=1) =", nb\_prob\_from\_counts, "\n")

##   
## (E) Naive Bayes P(Loan=1 | CC=1, Online=1) = 0.1105637

#Part F  
cat("\n(F) Comparison:\n")

##   
## (F) Comparison:

comparison<- data.frame(  
 Method = c("Direct from pivot (B)", "Naive Bayes from (D) + (E)"),  
 Probability = c(piv\_prob, nb\_prob\_from\_counts)  
)  
print(comparison, row.names = FALSE)

## Method Probability  
## Direct from pivot (B) 0.1071429  
## Naive Bayes from (D) + (E) 0.1105637

#Part G  
cat("\n(G) Needed entries for Naive Bayes calculation:\n")

##   
## (G) Needed entries for Naive Bayes calculation:

cat("- From Loan x CC: P(CC=1 | Loan=1) and P(CC=1 | Loan=0)\n")

## - From Loan x CC: P(CC=1 | Loan=1) and P(CC=1 | Loan=0)

cat("- From Loan x Online: P(Online=1 | Loan=1) and P(Online=1 | Loan=0)\n")

## - From Loan x Online: P(Online=1 | Loan=1) and P(Online=1 | Loan=0)

cat("- Class priors: P(Loan=1) and P(Loan=0)\n")

## - Class priors: P(Loan=1) and P(Loan=0)

train\_fac\_for\_nb<- train\_fac #already factors w/ levels c(0,1)  
nb\_fit<- naiveBayes(Loan ~ CC + Online, data = train\_fac\_for\_nb, laplace = 0)  
newx<- data.frame(CC = factor(1, levels = c(0,1)),  
 Online = factor(1, levels = c(0,1)))  
post<- predict(nb\_fit, newdata = newx, type = "raw")  
cat("\nModel posterior P(Loan=1 | CC=1, Online=1) from e1071::naiveBayes =", post[, "1"], "\n")

##   
## Model posterior P(Loan=1 | CC=1, Online=1) from e1071::naiveBayes = 0.1105637