Assignment\_3v1a

Bryan Mack

2025-06-10

The purpose of this assignment is to use Naive Bayes for classification.

The file UniversalBank.csv contains data on 5000 customers of Universal Bank. The data include customer demographic information (age, income, etc.), the customer’s relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan). Among these 5000 customers, only 480 (= 9.6%) accepted the personal loan that was offered to them in the earlier campaign. In this exercise, we focus on two predictors: Online (whether or not the customer is an active user of online banking services) and Credit Card (abbreviated CC below) (does the customer hold a credit card issued by the bank), and the outcome Personal Loan (abbreviated Loan below).

Partition the data into training (60%) and validation (40%) sets.

Step 1 install packages

#install.packages("dplyr")  
#install.packages("ISLR")  
#install.packages("e1071")  
#install.packages("tidyr")  
#install.packages("class")  
#install.packages("ggplot2")

Step 2 Load libraries

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(ISLR)  
library(e1071)  
library(tidyr)  
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(ggplot2)  
library(gmodels)  
library(pROC)

## Type 'citation("pROC")' for a citation.

##   
## Attaching package: 'pROC'

## The following object is masked from 'package:gmodels':  
##   
## ci

## The following objects are masked from 'package:stats':  
##   
## cov, smooth, var

#learn to use require() to look for a required library

Step 3 Working Directory

setwd("D:/R\_DATA")  
getwd()

## [1] "D:/R\_DATA"

#setwd() #used to set R working directory   
# i.e. D:/R\_DATA or (note / for windows OS)   
# setwd("D:\\\\R\_DATA\\\\")  
#setwd("D:/R\_DATA")

Step 4 Load Working Data UniversalBank.csv

#data.df <- read.csv("UniversalBank.csv")  
#data <- read.csv("UniversalBank.csv")  
#Remove a variable Example  
#MyData<-data[,-2] Example  
#MyData <- data[,-1] #Remove column 1 ID  
data <- read.csv("UniversalBank.csv", stringsAsFactors = FALSE)  
data$CreditCard <- as.factor(data$CreditCard)  
data$Personal.Loan <- as.factor(data$Personal.Loan)  
data$Online <- as.factor(data$Online)  
#MyData <- data[,-1] #Remove column 1 ID

Step 5 Read Original Data

#head(data)  
#summary(data)  
#Find the number of missing values in the data set  
#sum(is.na(data$Online))   
#sum(is.na(data$CreditCard))  
#sum(is.na(data$Personal.Loan))

Step 5 Partition the data into training (60%) and validation (40%) sets.

set.seed(123)  
#Divide data into test and train  
Index\_Train <- createDataPartition(data$Personal.Loan, p=0.6, list=FALSE)  
Train <-data[Index\_Train,]  
Test <-data[-Index\_Train,]

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(). In Python, use panda dataframe methods melt() and pivot().

#DATA MINING FOR BUSINESS ANALYTICS Page 98 pivot tables using functions melt() and cast()  
#install(reshape)  
#library(reshape2)  
#create bins of size 1

pivot\_table <- table(data$CreditCard, data$Personal.Loan, data$Online)  
#pivot\_table\_train <- table(Train$CreditCard, Train$Personal.Loan, Train$Online)  
print(pivot\_table)

## , , = 0  
##   
##   
## 0 1  
## 0 1300 128  
## 1 527 61  
##   
## , , = 1  
##   
##   
## 0 1  
## 0 1893 209  
## 1 800 82

pivot\_table\_2 <- table(CreditCard = data$CreditCard,   
 Personal.Loan = data$Personal.Loan,   
 Online = data$Online)  
print(pivot\_table\_2)

## , , Online = 0  
##   
## Personal.Loan  
## CreditCard 0 1  
## 0 1300 128  
## 1 527 61  
##   
## , , Online = 1  
##   
## Personal.Loan  
## CreditCard 0 1  
## 0 1893 209  
## 1 800 82

Training Data Pivot Table

pivot\_table\_3 <- table(CreditCard = Train$CreditCard,   
 Personal.Loan = Train$Personal.Loan,   
 Online = Train$Online)  
print(pivot\_table\_3)

## , , Online = 0  
##   
## Personal.Loan  
## CreditCard 0 1  
## 0 791 79  
## 1 310 33  
##   
## , , Online = 1  
##   
## Personal.Loan  
## CreditCard 0 1  
## 0 1144 125  
## 1 467 51

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)].

Probability that Personal.Loan = 1 CreditCard = 1 Online = 1

Credit Card = 1, Online = 1, Loan = 1 = 48 (from table)

Total number of customers Credit Card = 1, Personal Loan = 1 = 477 (from table)

Total Customers with CC = 48 + 477 = 525

48/525 = 0.0914285714285714 = 9.14%

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.

function of Online (columns):

pivot\_loan\_online <- table(Personal.Loan = Train$Personal.Loan,   
 Online = Train$Online)  
print(pivot\_loan\_online)

## Online  
## Personal.Loan 0 1  
## 0 1101 1611  
## 1 112 176

function of CreditCard (columns):

pivot\_loan\_cc <- table(Personal.Loan = Train$Personal.Loan,   
 CreditCard = Train$CreditCard)  
print(pivot\_loan\_cc)

## CreditCard  
## Personal.Loan 0 1  
## 0 1935 777  
## 1 204 84

D. Compute the following quantities [P(A | B) means “the probability of A given B”]: i. P(CC = 1 | Loan = 1) (the proportion of credit card holders among the loan acceptors) (CC =1 | Loan = 1) = 86 /(196 + 86) 86/282 = 0.3049645390070922 30.496% of Loan Acceptors have a Credit Card ii. P(Online = 1 | Loan = 1) (Online = 1 | Loan = 1) 174 / (108 + 174) 174/282 = 0.6170212765957447 61.7021% Of Loan Acceptors have an Online Account iii. P(Loan = 1) (the proportion of loan acceptors) # 1 108 + 174 = 282 Online # 0 1091 1627 = 2,718 Online # 1 196 + 86 = 282 CreditCard # 0 1951 767 = 2,718 Credit Card # 3000 Total Records 282 / 3000 = 0.094 9.4% Have a Personal Loan iv. P(CC = 1 | Loan = 0) P(CC = 1 | Loan = 0) = 767 / (1951 + 767) 767 / (2,718) = 0.2821927888153054 28.2192% of non-Loan Acceptors have a Credit Card v. P(Online = 1 | Loan = 0) P(Online = 1 | Loan = 0) = 1627 / (1091 + 1627) 1627/2718 = 0.5986019131714496 59.8601% of non-Loan Acceptors have an Online Account vi. P(Loan = 0) # 1 108 + 174 = 282 Online # 0 1091 1627 = 2,718 Online # 1 196 + 86 = 282 CreditCard # 0 1951 767 = 2,718 Credit Card # 3000 Total Records 2,718 / 3000 = 0.906 90.6% of Customers do not have a Personal Loan

#summary(data)

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

naive Bayes probability: P(Loan = 1 | CC = 1, Online = 1) = 0.2916667 \* 0.6111111 = 0.17824075787037 17.82% Probability of a loan acceptor has both an online account and credit card

# Build a naïve Bayes classifier  
nb\_model <- naiveBayes(Personal.Loan~CreditCard+Online,data = Train)  
nb\_model

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## 0 1   
## 0.904 0.096   
##   
## Conditional probabilities:  
## CreditCard  
## Y 0 1  
## 0 0.7134956 0.2865044  
## 1 0.7083333 0.2916667  
##   
## Online  
## Y 0 1  
## 0 0.4059735 0.5940265  
## 1 0.3888889 0.6111111

# Predict the default status of test dataset   
Predicted\_Test\_labels <-predict(nb\_model,Test)  
  
#library("gmodels")  
  
# Show the confusion matrix of the classifier  
CrossTable(x=Test$Personal.Loan,y=Predicted\_Test\_labels, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 2000   
##   
##   
## | Predicted\_Test\_labels   
## Test$Personal.Loan | 0 | Row Total |   
## -------------------|-----------|-----------|  
## 0 | 1808 | 1808 |   
## | 0.904 | |   
## -------------------|-----------|-----------|  
## 1 | 192 | 192 |   
## | 0.096 | |   
## -------------------|-----------|-----------|  
## Column Total | 2000 | 2000 |   
## -------------------|-----------|-----------|  
##   
##

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

naive Bayes probability value: 17.82% Pivot Table Question B probability value: 9.14%

The Naive Bayes looks more accurate because it is a higher probability of Personal Loan acceptance when looking at customers with both an active Credit Card and Online Account. The directions stated that 9.6% of the total population accepted the last Personal Loan offer but we see a higher probability with acceptance if the customer has a Credit Card and Online Account.

naive Bayes probability: P(Loan = 1 | CC = 1, Online = 1) = 0.2916667 \* 0.6111111 = 0.17824075787037 17.82% Probability of a loan acceptor has both an online account and credit card

Probability that Personal.Loan = 1 CreditCard = 1 Online = 1 Credit Card = 1, Online = 1, Loan = 1 = 48 (from table) Total number of customers Credit Card = 1, Personal Loan = 1 = 477 (from table) Total Customers with CC = 48 + 477 = 525 48/525 = 0.0914285714285714 = 9.14%

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).

The entries needed in the table are the

#nb\_model <- naiveBayes(default~balance+income,data = Train)  
  
  
#Make predictions and return probability of each class  
Predicted\_Test\_labels <-predict(nb\_model,Test, type = "raw")  
  
#show the first few values   
head(Predicted\_Test\_labels)

## 0 1  
## [1,] 0.9082737 0.09172629  
## [2,] 0.9021538 0.09784623  
## [3,] 0.9061594 0.09384060  
## [4,] 0.9082737 0.09172629  
## [5,] 0.9082737 0.09172629  
## [6,] 0.8999139 0.10008606

#install.packages("pROC") # install if necessary  
#library(pROC)  
  
#Passing the second column of the predicted probabilities   
#That column contains the probability associate to ‘yes’  
roc(Test$Personal.Loan, Predicted\_Test\_labels[,2])

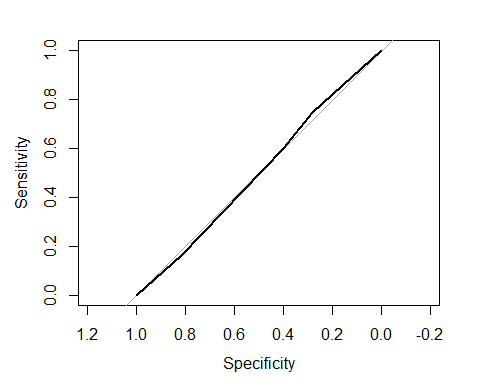
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##   
## Call:  
## roc.default(response = Test$Personal.Loan, predictor = Predicted\_Test\_labels[, 2])  
##   
## Data: Predicted\_Test\_labels[, 2] in 1808 controls (Test$Personal.Loan 0) < 192 cases (Test$Personal.Loan 1).  
## Area under the curve: 0.4986

plot.roc(Test$Personal.Loan,Predicted\_Test\_labels[,2])

## Setting levels: control = 0, case = 1  
## Setting direction: controls < cases



#Create a Box-Cox Transformation Model  
library(ISLR)  
library(caret)  
Box\_Cox\_Transform<-preProcess(data,method = "BoxCox")  
Box\_Cox\_Transform

## Created from 5000 samples and 9 variables  
##   
## Pre-processing:  
## - Box-Cox transformation (6)  
## - ignored (3)  
##   
## Lambda estimates for Box-Cox transformation:  
## 0.7, 0.8, 0.3, 2, 0.4, 0

# PersonalLoan\_Transformed=predict(Box\_Cox\_Transform,data)  
# y <- PersonalLoan\_Transformed$Personal.Loan  
# h<-hist(y, breaks=10, col="red", xlab="Personal Loan",  
# main="Histogram before Transformation")  
# xfit<-seq(min(y),max(y),length=40)  
# yfit<-dnorm(xfit,mean=mean(y),sd=sd(y))  
# yfit <- yfit\*diff(h$mids[1:2])\*length(y)  
# lines(xfit, yfit, col="blue", lwd=2)