HW\_3\_ML

2022-10-26

set.seed(999)  
#Loading the data and converting the independent variables to factors  
UniversalBank <- read\_csv("/Users/nawwaf/Desktop/Kent/Kent Master\_s/Machine Learning/UniversalBank.csv")  
Orignal\_Data <- UniversalBank  
  
Orignal\_Data$CreditCard <- as.factor(Orignal\_Data$CreditCard)  
Orignal\_Data$`Personal Loan` <- as.factor(Orignal\_Data$`Personal Loan`)  
Orignal\_Data$Online <- as.factor(Orignal\_Data$Online)

#Split the data into training and testing set  
  
Index\_Train<-createDataPartition(Orignal\_Data$`Personal Loan`, p=0.6, list=FALSE)  
Train <-Orignal\_Data[Index\_Train,]  
Test <-Orignal\_Data[-Index\_Train,]

#Removing the predictor variables and Normalizing the data   
Normlaized\_Data <- preProcess(Train[,-c(10,13:14)],   
 method=c("center","scale"))  
Training\_predictions <- predict(Normlaized\_Data,Train)  
Test\_predictions <- predict(Normlaized\_Data,Test)

#(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  
table <- ftable(Training\_predictions[,c("CreditCard","Personal Loan","Online")])  
table

## Online 0 1  
## CreditCard Personal Loan   
## 0 0 769 1130  
## 1 76 120  
## 1 0 319 494  
## 1 39 53

#(B)  
# Consider the task of classifying a customer who owns a bank credit card and is actively using   
# online banking services  
  
  
#Total of people that accepted a personal loan =   
Total\_Loan = 76 + 120 + 39 + 53  
#Total number of people with No Loan  
Total\_No\_Loan = 769 + 1130 + 319 + 494  
ALL\_Loan\_And\_No\_Loan = Total\_Loan + Total\_No\_Loan  
#Total of people that have CC  
CC\_total = 319 + 494 + 39 + 53  
#Total of people that have CC among those with loan   
Loan\_And\_CC = 39 + 53  
No\_loan\_And\_CC = 319 + 494  
#Total of people that have online banking   
Total\_Online = 1130 + 120 + 494 + 53  
#Total of people that have online banking among those that accepted a loan  
Loan\_And\_Online = 120 + 53  
No\_Loan\_And\_Online = 1130 + 494  
  
Numenator = (Loan\_And\_CC / Total\_Loan) \* (Loan\_And\_Online / Total\_Loan) \* (Total\_Loan/ALL\_Loan\_And\_No\_Loan)  
Denomentor = (No\_loan\_And\_CC/Total\_No\_Loan)\*(No\_Loan\_And\_Online/Total\_No\_Loan)\*(Total\_No\_Loan/ALL\_Loan\_And\_No\_Loan)  
P= Numenator/Denomentor  
cat("the probability of a customer to accept a loan while being a credit card holder and an online banking user is",P)

## the probability of a customer to accept a loan while being a credit card holder and an online banking user is 0.1135153

#(C) 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.   
Online\_And\_Loan\_Table = ftable(Training\_predictions[,c(10,13)])  
Online\_And\_Loan\_Table

## Online 0 1  
## Personal Loan   
## 0 1088 1624  
## 1 115 173

CC\_And\_Loan\_Table = ftable(Training\_predictions[,c(10,14)])  
CC\_And\_Loan\_Table

## CreditCard 0 1  
## Personal Loan   
## 0 1899 813  
## 1 196 92

ftable(Training\_predictions[,10])

## Personal Loan 0 1  
##   
## 2712 288

#(D) Compute the following quantities [P(A | B) means “the probability ofA given B”]:   
#i. P(CC = 1 | Loan = 1) (the proportion of credit card holders among the loan acceptors)   
CC\_And\_Loan = 92/(92+196)  
CC\_And\_Loan

## [1] 0.3194444

#ii. P(Online = 1 | Loan = 1)   
Online\_And\_Loan =173/(173+115)  
Online\_And\_Loan

## [1] 0.6006944

#iii. P(Loan = 1) (the proportion of loan acceptors)   
Loan\_to\_ALL = 288/(2712+288)  
Loan\_to\_ALL

## [1] 0.096

#iv. P(CC = 1 | Loan = 0)   
CC\_And\_N\_Loan = 813/(813+1899)  
CC\_And\_N\_Loan

## [1] 0.2997788

#v. P(Online = 1 | Loan = 0)   
Online\_And\_No\_loan =1624/(1624+1088)  
Online\_And\_No\_loan

## [1] 0.5988201

#vi. P(Loan = 0)   
No\_Loan\_To\_ALL = 2712/(2712+288)  
No\_Loan\_To\_ALL

## [1] 0.904

#(E). Use the quantities computed above to compute the naive Bayes probability  
#P(Loan = 1 | CC = 1, Online = 1).   
(CC\_And\_Loan\*Online\_And\_Loan\*Loan\_to\_ALL)/((CC\_And\_Loan\*Online\_And\_Loan\*Loan\_to\_ALL)+(CC\_And\_N\_Loan\*Online\_And\_No\_loan\*No\_Loan\_To\_ALL))

## [1] 0.1019432

#(F) Compare this value with the one obtained from the pivot table in (B). Which is a more   
#accurate estimate?  
cat("Naive bayes account for the conditional events against Ci where in the direct method we dont account for Ci", NULL)

## Naive bayes account for the conditional events against Ci where in the direct method we dont account for Ci

#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).   
  
NaiveB <- naiveBayes(`Personal Loan`~Online+CreditCard,data=Training\_predictions)  
NaiveB

##   
## 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:  
## Online  
## Y 0 1  
## 0 0.4011799 0.5988201  
## 1 0.3993056 0.6006944  
##   
## CreditCard  
## Y 0 1  
## 0 0.7002212 0.2997788  
## 1 0.6805556 0.3194444

#(G)Which of the entries in this table are needed for computing P(Loan = 1 | CC = 1, Online = 1)?  
cat("The entries needed are A-priori probabilities (0):904, Conditional probabilities:  
 Online(0,1):0.5988201, Conditional probabilities:CreditCard(0,1): 0.2997788, A-priori probabilities:  
Y:0.096, Conditional probabilities:  
 Online(1,1):0.6006944, Conditional probabilities:CreditCard 0.3194444",NULL)

## The entries needed are A-priori probabilities (0):904, Conditional probabilities:  
## Online(0,1):0.5988201, Conditional probabilities:CreditCard(0,1): 0.2997788, A-priori probabilities:  
## Y:0.096, Conditional probabilities:  
## Online(1,1):0.6006944, Conditional probabilities:CreditCard 0.3194444

#Compare this to the number you   
#obtained in (E).  
cat("The numbers are exactly the same as the ones in E")

## The numbers are exactly the same as the ones in E