# MichaelBasta_Assignment_3

Michael Basta

2022-10-16

```
banks <- read.csv("/Users/michaelbasta/Documents/Fundmentals of Machine Learning /Module 5/UniversalBank

#install.packages("reshape")
#install.packages("reshape2")

library(reshape)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following objects are masked from 'package:reshape':
##
##     colsplit, melt, recast
```

```
# A.
table(banks$CreditCard, banks$Personal.Loan, banks$Online)
```

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

```
prop.table(table(banks$CreditCard, banks$Personal.Loan, banks$Online))
```

```
## , ,  = 0
##
##
##          0      1
##   0 0.2600 0.0256
```

```
##   1 0.1054 0.0122
##
## , ,  = 1
##
##
##          0      1
##   0 0.3786 0.0418
##   1 0.1600 0.0164
```

```
# B.
print("The pobability that the customer will accept the loan conditional on customer has Credit Card and
```

```
## [1] "The pobability that the customer will accept the loan conditional on customer has Credit Card an
```

```
# C.
print("Table row: loan  Col: Online")
```

```
## [1] "Table row: loan  Col: Online"
```

```
table(banks$Personal.Loan, banks$Online)
```

```
##
##        0    1
##   0 1827 2693
##   1  189  291
```

```
prop.table(table(banks$Personal.Loan, banks$Online))
```

```
##
##          0      1
##   0 0.3654 0.5386
##   1 0.0378 0.0582
```

```
print("Table row: loan  Col: Credit Card ")
```

```
## [1] "Table row: loan  Col: Credit Card "
```

```
table(banks$Personal.Loan, banks$CreditCard)
```

```
##
##        0    1
##   0 3193 1327
##   1  337  143
```

```
prop.table(table(banks$Personal.Loan, banks$CreditCard))
```

```
##
##          0      1
##   0 0.6386 0.2654
##   1 0.0674 0.0286
```

```
# D.
## i.
print("i. P(CC = 1 | loan = 1) = 0.286")
```

```
## [1] "i. P(CC = 1 | loan = 1) = 0.286"
```

```
## ii.
print("ii. P(online = 1 | loan = 1) = 0.0582")
```

```
## [1] "ii. P(online = 1 | loan = 1) = 0.0582"
```

```
## iii.
print("iii. P(loan = 1) = 0.0378 + 0.0582 = 0.096")
```

```
## [1] "iii. P(loan = 1) = 0.0378 + 0.0582 = 0.096"
```

```
## iv
print("iv. P(CC = 1 | loan = 0) = 0.2654")
```

```
## [1] "iv. P(CC = 1 | loan = 0) = 0.2654"
```

```
## v.
print("v. P(online = 1 | loan = 0) = 0.5386")
```

```
## [1] "v. P(online = 1 | loan = 0) = 0.5386"
```

```
## vi.
print("vi. P(loan = 0) = 0.3654 + 0.5386 = 0.904")
```

```
## [1] "vi. P(loan = 0) = 0.3654 + 0.5386 = 0.904"
```

```
# E.
print("Using the long formula 8.3 in textbook")
```

```
## [1] "Using the long formula 8.3 in textbook"
```

```
print("P(loan = 1 | CC = 1, Online = 1) = (P(loan=1)[P(CC=1 | loan=1) * P(Online=1 | loan=1)] \ (P(loan=
```

```
## [1] "P(loan = 1 | CC = 1, Online = 1) = (P(loan=1)[P(CC=1 | loan=1) * P(Online=1 | loan=1)]  (P(loan=
```

```
sprintf("Numerator = 0.096*[0.286*0.0582] = %1.4f",(0.096*0.286*0.0582))
```

```
## [1] "Numerator = 0.096*[0.286*0.0582] = 0.0016"
```

```
sprintf("Denominator = 0.002 + 0.904*[0.2654*0.5386] = %1.4f",(0.0016 + (0.904*0.2654*0.5386)))
```

```
## [1] "Denominator = 0.002 + 0.904*[0.2654*0.5386] = 0.1308"
```

```
sprintf("Naive Bayes Probability = 0.002/0.131 = %1.4f",(0.002/0.131))
```

```
## [1] "Naive Bayes Probability = 0.002/0.131 = 0.0153"
```

```
# F.
print("Both values are very close B is 0.016 & E is 0.0153")
```

```
## [1] "Both values are very close B is 0.016 & E is 0.0153"
```

```
#install.packages("caret")
#install.packages("ISLR")
#install.packages("e1071")

library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(ISLR)
library(e1071)

# G.
# Divide data into test and train (60% - 40%)
Index_Train <- createDataPartition(banks$Personal.Loan, p=0.6, list=FALSE)
Train <- banks[Index_Train,]
Test <- banks[-Index_Train,]

# Build a naive Bayes classifier
nb_model <- naiveBayes(Personal.Loan~CreditCard+Online, data = Train)

# Predict the default status of test dataset
Predicted_Test_labels <- predict(nb_model,Train, type = "raw")

head(cbind(Predicted_Test_labels,Train$CreditCard, Train$Online))
```

```
##              0          1
## [1,] 0.9042180 0.09578198 0 0
## [2,] 0.9017867 0.09821328 0 1
## [3,] 0.9017867 0.09821328 0 1
## [4,] 0.9042180 0.09578198 0 0
## [5,] 0.9042180 0.09578198 0 0
## [6,] 0.9042180 0.09578198 0 0
```

```
p <- predict(nb_model,Train)
t1 <- table(p,Train$CreditCard)
prop.table(t1)
```

```
##
## p       0     1
##   0 0.708 0.292
##   1 0.000 0.000
```

```r
t2 <- table(p,Train$Online)
prop.table(t2)
```

```
##
## p           0         1
##   0 0.3986667 0.6013333
##   1 0.0000000 0.0000000
```

```r
#install.packages("gmodels")
library("gmodels")

p <- predict(nb_model,Test)
CrossTable(x=Test$Personal.Loan,y=p,prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##                     | p
## Test$Personal.Loan |         0 | Row Total |
## -------------------|-----------|-----------|
##                  0 |      1800 |      1800 |
##                    |     0.900 |           |
## -------------------|-----------|-----------|
##                  1 |       200 |       200 |
##                    |     0.100 |           |
## -------------------|-----------|-----------|
##       Column Total |      2000 |      2000 |
## -------------------|-----------|-----------|
##
##
```