BA 64060 - Assignment 3

Disney Maxwell

2025-10-12

Install packages

```
library(readr)
library(class)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```
library(ISLR)
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)
library(e1071)
```

```
##
## Attaching package: 'e1071'

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

Dataset: UniversalBank.csv - Data on a Bank's 5000 customers

Import UniversalBank Data into R Data Transforms

```
universal_bank_init<-read_csv("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.
```

```
#Remove other columns, Columns to remove
#ID, Age, Experience, Income, ZIP code, Family, CCAvg, Education, Mortgage, Securities Account, CD Acco
cols_to_remove <- c("ID", "Age", "Experience", "Income", "ZIP Code", "Family", "CCAvg", "Education", "M
universal_bank_data<-universal_bank_init[,!(names(universal_bank_init) %in% cols_to_remove)]
#Rename Personal Loan column name to Loan
universal_bank_data <- universal_bank_data %>% rename(Loan = 'Personal Loan', CC = 'CreditCard')
```

Partition Data into Training (60%) and Validation (40%)

```
set.seed(123)
Train_Index<-createDataPartition(universal_bank_data$Loan, p=0.6, list=FALSE)
Train_Data<-universal_bank_data[Train_Index,]
Validn_Data<-universal_bank_data[-Train_Index,]
```

QA. Create a Pivot Table for Training Data

Online as column, CC as row and Loan as secondary row

```
# Create the pivot table
Pivot_Table_Train_Data <- Train_Data %>%
  group_by(CC, Loan, Online) %>%
  summarise(Count = n()) %>%
  pivot_wider(names_from = Online, values_from = Count, values_fill = 0)
```

```
## `summarise()` has grouped output by 'CC', 'Loan'. You can override using the
## `.groups` argument.
```

```
# Rename Online Columns
colnames(Pivot_Table_Train_Data)[3]<- "Online = 0"
colnames(Pivot_Table_Train_Data)[4]<- "Online = 1"

# Print the pivot table
print(Pivot_Table_Train_Data)
```

```
## # A tibble: 4 x 4
## # Groups:   CC, Loan [4]
##      CC  Loan `Online = 0` `Online = 1`
##   <dbl> <dbl>        <int>        <int>
## 1     0     0          785         1145
## 2     0     1           65          122
## 3     1     0          317          475
## 4     1     1           34           57
```

```
Check_Data <- Train_Data %>%
  group_by(CC, Loan, Online) %>%
  summarise(Count = n(), .groups = "rowwise")
```

QB. Classifying a customer who owns a credit card and is using online services

From Pivot_Table_Train_Data Probability that this customer will accept the loan offer = 57/(475+57)

```
print(paste0("Probability that this customer will accept the loan offer = 57/(475+57)=", (57/(475+57))))
```

```
## [1] "Probability that this customer will accept the loan offer = 57/(475+57)=0.107142857142857"
```

QC. Create Two Pivot Tables

```
#Create First Pivot Table
Pivot_Table_C1_Train_Data <- Train_Data %>%
  group_by(Loan, Online) %>%
  summarise(Count = n()) %>%
  pivot_wider(names_from = Online, values_from = Count, values_fill = 0)
```

```
## `summarise()` has grouped output by 'Loan'. You can override using the
## `.groups` argument.
```

```
# Rename Online Columns
colnames(Pivot_Table_C1_Train_Data)[2]<- "Online = 0"
colnames(Pivot_Table_C1_Train_Data)[3]<- "Online = 1"

# Print First pivot table - Loan as a function of Online (columns)
print(paste0("First pivot table - Loan as a function of Online (columns)"))
```

```
## [1] "First pivot table - Loan as a function of Online (columns)"
```

```
print(Pivot_Table_C1_Train_Data)
```

```
## # A tibble: 2 x 3
## # Groups:   Loan [2]
```

```
##    Loan `Online = 0` `Online = 1`
##   <dbl>        <int>        <int>
## 1     0         1102         1620
## 2     1           99          179
```

```
#Create Second Pivot Table
Pivot_Table_C2_Train_Data <- Train_Data %>%
  group_by(Loan, CC) %>%
  summarise(Count = n()) %>%
  pivot_wider(names_from = CC, values_from = Count, values_fill = 0)
```

```
## `summarise()` has grouped output by 'Loan'. You can override using the
## `.groups` argument.
```

```
# Rename Online Columns
colnames(Pivot_Table_C2_Train_Data)[2]<- "CC = 0"
colnames(Pivot_Table_C2_Train_Data)[3]<- "CC = 1"

# Print Second pivot table - Loan as a function of Credit Card (columns)
print(paste0("Second pivot table - Loan as a function of Credit Card (columns)"))
```

```
## [1] "Second pivot table - Loan as a function of Credit Card (columns)"
```

```
print(Pivot_Table_C2_Train_Data)
```

```
## # A tibble: 2 x 3
## # Groups:   Loan [2]
##    Loan `CC = 0` `CC = 1`
##   <dbl>    <int>    <int>
## 1     0     1930      792
## 2     1      187       91
```

QD. Probability Computations

```
# i. P(CC=1 | Loan =1) = 91/(187+91)
print(paste0("P(CC=1 | Loan =1) = 91/(187+91) = ", (91/(187+91))))
```

```
## [1] "P(CC=1 | Loan =1) = 91/(187+91) = 0.327338129496403"
```

```
# ii. P(Online=1 | Loan=1) = 179/(179+99)
print(paste0("P(Online=1 | Loan=1) = 179/(179+99) = ", (179/(179+99))))
```

```
## [1] "P(Online=1 | Loan=1) = 179/(179+99) = 0.643884892086331"
```

```
# iii. P(Loan=1) = 278/3000
print(paste0("P(Loan=1) = 278/3000 = ", (278/3000)))
```

```
## [1] "P(Loan=1) = 278/3000 = 0.0926666666666667"
```

```r
# iv. P(CC=1 | Loan=0) = 792/(1930+792)
print(paste0("P(CC=1 | Loan=0) = 792/(1930+792) = ", (792/(1930+792))))
```

```
## [1] "P(CC=1 | Loan=0) = 792/(1930+792) = 0.29096252755327"
```

```r
# v. P(Online=1 | Loan=0 ) = 1620/(1102+1620)
print(paste0("P(Online=1 | Loan=0 ) = 1620/(1102+1620) = ", (1620/(1102+1620))))
```

```
## [1] "P(Online=1 | Loan=0 ) = 1620/(1102+1620) = 0.595150624540779"
```

```r
# vi. P(Loan=0) = 2722/3000
print(paste0("P(Loan=0) = 2722/3000 = ", (2722/3000)))
```

```
## [1] "P(Loan=0) = 2722/3000 = 0.907333333333333"
```

QE. Naive Bayes Probability

```r
# P(Loan=1 | CC=1, Online=1) = [P(CC=1, Online=1 | Loan=1) * P(Loan=1)] / P(CC=1, Online=1)
# The naive part is the assumption that the predictor features are independent of each other.
# Therefore, P(CC=1, Online=1 | Loan=1) = P(CC=1 | Loan=1) * P(Online=1 | Loan=1)
# Substituting above:
# [P(CC=1 | Loan=1) * P(Online=1 | Loan=1) * P(Loan=1)] / P(CC=1, Online=1)
# = (91/278) * (179/278) * (278/3000) / (532/1799) * (1799/3000)
# = (91*179) / (532*278)
print(paste0("Computing naive Bayes probability, P(Loan=1 | CC=1, Online=1) = ", ((91*179)/(532*278))))
```

```
## [1] "Computing naive Bayes probability, P(Loan=1 | CC=1, Online=1) = 0.110138205225293"
```

QF. Compare QB and QE - Naive Bayes probability calculations

```r
# In the naive Bayes method, we had to assume that the features are independent of each other.
# In the Pivot table in QB approach, we calculated the probability using the numbers in the data set.
# As this used actual numbers, the answer from QB is a more accurate estimate which equals 0.1071.
```

QG. naive Bayes Model

From Pivot_Table_Train_Data P(Loan=1 | CC=1, Online=1) = 57/(475+57) The entries required are: - Given CC=1 and Online=1, No. of Loan = 1 which equals 57 - Given CC=1 and Online=1, No. of Loan possibilities which equals (475+57)

```r
# Transform variables to categorical
Train_Data$Loan<-as.factor(Train_Data$Loan)
Train_Data$Online<-as.factor(Train_Data$Online)
Train_Data$CC<-as.factor(Train_Data$CC)

# Build a naive Bayes classifier
nb_model<-naiveBayes(Loan~Online+CC, data=Train_Data)

#Make predictions and return probability of each class
Predicted_Train_labels<-predict(nb_model,Train_Data, type="raw")

#Examine Model output on Training data
head(Predicted_Train_labels)
```

```
##              0          1
## [1,] 0.9214655 0.07853447
## [2,] 0.9214655 0.07853447
## [3,] 0.9082092 0.09179078
## [4,] 0.9051206 0.09487943
## [5,] 0.9214655 0.07853447
## [6,] 0.9051206 0.09487943
```

```r
#Entries which correspond to P(Loan=1 | CC=1, Online=1)
data_loan1_entries<-data.frame(CC = factor(1, levels = c(0, 1)), Online = factor(1, levels = c(0, 1)))
probabilities<-predict(nb_model,data_loan1_entries, type="raw")
p_loan1_given_cc1_online1<-probabilities[1, "1"]
print(paste0("Using the Naive Bayes model, P(Loan=1 | CC=1, Online=1) = ", p_loan1_given_cc1_online1))
```

```
## [1] "Using the Naive Bayes model, P(Loan=1 | CC=1, Online=1) = 0.110563663384569"
```

```r
# The probability in QE and QG are 0.1101. This shows that the model is using naive Bayes probability.
print(paste0("The probability in QE and QG are 0.1101. This shows that the model is using naive Bayes p
```

```
## [1] "The probability in QE and QG are 0.1101. This shows that the model is using naive Bayes probabil
```