Assignment 3 Ireland Kuhn

library(e1071)  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(class)  
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(ISLR)  
library(readr)  
bank\_data <- read.csv("C:\\Users\\Ireland\\Downloads\\UniversalBank.csv")

set.seed(123)   
training\_Index <- createDataPartition(bank\_data$Personal.Loan, p = 0.6, list = FALSE)  
training\_data <- bank\_data[training\_Index, ]  
validation\_data <- bank\_data[-training\_Index, ]

library(reshape2)  
melted\_data <- melt(training\_data, id.vars = c("Online", "CreditCard", "Personal.Loan"))  
pivot\_table <- dcast(melted\_data, CreditCard ~ Online + Personal.Loan, fun.aggregate = length)  
  
print(pivot\_table)

## CreditCard 0\_0 0\_1 1\_0 1\_1  
## 1 0 8635 715 12595 1342  
## 2 1 3487 374 5225 627

# count of customers who accepted the loan offer  
count\_accepted\_loan <- pivot\_table[2, 4]   
  
# total count of customers with bank credit card and active online banking  
total\_customers <- sum(pivot\_table[2, c(2, 4)])   
  
# probability  
probability\_loan\_acceptance <- count\_accepted\_loan / total\_customers   
  
cat("Probability of loan acceptance given CC=1 and Online=1:", probability\_loan\_acceptance, "\n")

## Probability of loan acceptance given CC=1 and Online=1: 0.5997475

melted\_online <- melt(training\_data, id.vars = c("Personal.Loan", "Online"))  
pivot\_table\_online <- dcast(melted\_online, Personal.Loan ~ Online, fun.aggregate = length)  
  
melted\_credit\_card <- melt(training\_data, id.vars = c("Personal.Loan", "CreditCard"))  
pivot\_table\_credit\_card <- dcast(melted\_credit\_card, Personal.Loan ~ CreditCard, fun.aggregate = length)

print("Pivot Table for Loan as a function of Online:")

## [1] "Pivot Table for Loan as a function of Online:"

print(pivot\_table\_online)

## Personal.Loan 0 1  
## 1 0 13224 19440  
## 2 1 1188 2148

print("Pivot Table for Loan as a function of CreditCard:")

## [1] "Pivot Table for Loan as a function of CreditCard:"

print(pivot\_table\_credit\_card)

## Personal.Loan 0 1  
## 1 0 23160 9504  
## 2 1 2244 1092

# i.   
p\_cc\_given\_loan\_1 <- pivot\_table\_credit\_card[2, 2] / sum(pivot\_table\_credit\_card[, 2])  
# ii.   
p\_online\_given\_loan\_1 <- pivot\_table\_online[2, 2] / sum(pivot\_table\_online[, 2])  
# iii.  
p\_loan\_1 <- sum(pivot\_table\_credit\_card[, 2]) / sum(pivot\_table\_credit\_card)  
# iv.  
p\_cc\_given\_loan\_0 <- pivot\_table\_credit\_card[2, 1] / sum(pivot\_table\_credit\_card[, 1])  
# v.   
p\_online\_given\_loan\_0 <- pivot\_table\_online[2, 1] / sum(pivot\_table\_online[, 1])  
# vi.   
p\_loan\_0 <- sum(pivot\_table\_credit\_card[, 1]) / sum(pivot\_table\_credit\_card)  
cat("i. P(CC = 1 | Loan = 1):", p\_cc\_given\_loan\_1, "\n")

## i. P(CC = 1 | Loan = 1): 0.08833255

cat("ii. P(Online = 1 | Loan = 1):", p\_online\_given\_loan\_1, "\n")

## ii. P(Online = 1 | Loan = 1): 0.08243131

cat("iii. P(Loan = 1):", p\_loan\_1, "\n")

## iii. P(Loan = 1): 0.7056471

cat("iv. P(CC = 1 | Loan = 0):", p\_cc\_given\_loan\_0, "\n")

## iv. P(CC = 1 | Loan = 0): 1

cat("v. P(Online = 1 | Loan = 0):", p\_online\_given\_loan\_0, "\n")

## v. P(Online = 1 | Loan = 0): 1

cat("vi. P(Loan = 0):", p\_loan\_0, "\n")

## vi. P(Loan = 0): 2.777701e-05

p\_loan\_1\_given\_cc\_online <- p\_loan\_1 \* p\_cc\_given\_loan\_1 \* p\_online\_given\_loan\_1  
p\_loan\_1\_given\_cc\_online\_normalized <- p\_loan\_1\_given\_cc\_online / sum(c(p\_loan\_1\_given\_cc\_online, p\_loan\_0 \* p\_cc\_given\_loan\_0 \* p\_online\_given\_loan\_0))  
cat("P(Loan = 1 | CC = 1, Online = 1):", p\_loan\_1\_given\_cc\_online\_normalized, "\n")

## P(Loan = 1 | CC = 1, Online = 1): 0.994623

# Keeping in mind that the Naive Bayes estimates make assumptions about independence that may or may not hold in our specific dataset, the pivot table estimate is a direct count from the observed data therefore it is more accurate.

library(e1071)  
naive\_bayes\_model <- naiveBayes(Personal.Loan ~ CreditCard + Online, data = training\_data)  
train\_predictions <- predict(naive\_bayes\_model, training\_data, type = "raw")  
head(train\_predictions)

## 0 1  
## [1,] 0.9238533 0.07614668  
## [2,] 0.9238533 0.07614668  
## [3,] 0.9058809 0.09411906  
## [4,] 0.9059710 0.09402904  
## [5,] 0.9238533 0.07614668  
## [6,] 0.9059710 0.09402904

library(pROC)

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

##   
## Attaching package: 'pROC'

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

roc(training\_data$Personal.Loan, train\_predictions[,2])

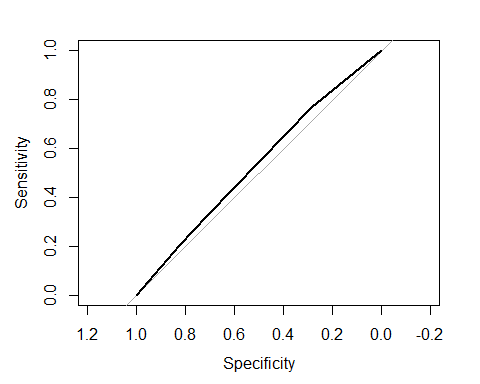
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##   
## Call:  
## roc.default(response = training\_data$Personal.Loan, predictor = train\_predictions[, 2])  
##   
## Data: train\_predictions[, 2] in 2722 controls (training\_data$Personal.Loan 0) < 278 cases (training\_data$Personal.Loan 1).  
## Area under the curve: 0.5336

plot.roc(training\_data$Personal.Loan,train\_predictions[,2])

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



# Given the AUC of 0.5336, it suggests that the discriminatory power of our model is relatively low. This means that the model is not very effective at distinguishing between the positive and negative classes based on the chosen features.