Assignment_3

Olayinka Sikiru

2024-02-29

```
# To load relevant packages
library(class)
library(caret)
library(dplyr)
library(tidyr)
library(e1071)
```

```
#To import the dataset
library(readr)
UniversalBank <- read_csv("C:/Users/DELL/Dataset/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.
```

```
# To display the first few rows of the dataset 'UniversalBank'
head(UniversalBank)
```

```
## # A tibble: 6 × 14
##      ID   Age Experience Income `ZIP Code` Family CCAvg Education Mortgage
##   <dbl> <dbl>      <dbl>  <dbl>      <dbl>  <dbl> <dbl>     <dbl>    <dbl>
## 1     1    25          1     49      91107      4   1.6         1        0
## 2     2    45         19     34      90089      3   1.5         1        0
## 3     3    39         15     11      94720      1   1           1        0
## 4     4    35          9    100      94112      1   2.7         2        0
## 5     5    35          8     45      91330      4   1           2        0
## 6     6    37         13     29      92121      4   0.4         2      155
## # ℹ 5 more variables: `Personal Loan` <dbl>, `Securities Account` <dbl>,
## #   `CD Account` <dbl>, Online <dbl>, CreditCard <dbl>
```

```
# Set seed for reproducibility
set.seed(123)

# Partition the data into training (60%) and validation (40%) sets
train_indices <- sample(1:nrow(UniversalBank), 0.6*nrow(UniversalBank))
train_data <- UniversalBank[train_indices, ]
valid_data <- UniversalBank[-train_indices, ]
```

```
# Change column names to 'CC' and 'Loan'
colnames(train_data)[colnames(train_data) == "CreditCard"] <- "CC"
colnames(train_data)[colnames(train_data) == "Personal Loan"] <- "Loan"
```

Solution A - Create the Pivot Table

```
#To create and print a pivot table from 'train_data' with 'Online' as column, 'CC' and 'Loan' as row variables.
pivot_table <- table(train_data$Online, train_data$CC, train_data$Loan)
print(pivot_table)
```

```
## , ,  = 0
##
##
##        0    1
##   0  785  317
##   1 1145  475
##
## , ,  = 1
##
##
##        0    1
##   0   65   34
##   1  122   57
```

Solution B - Probability Computation

```
prob_Loan_CC_Online <- pivot_table[2, 2, 2]

# Extract the count for (CC = 1, Online = 1)
prob_CC_Online <- sum(pivot_table[ , 2, 2])

# Calculate the probability
prob_Loan_given_CC_Online <- prob_Loan_CC_Online / prob_CC_Online

# Print the result
print(prob_Loan_given_CC_Online)
```

```
## [1] 0.6263736
```

Solution C - Create two separate pivot tables for the training data

```
# Pivot table for Loan as a function of Online
pivot_Loan_Online <- table(train_data$Loan, train_data$Online)

# Pivot table for Loan as a function of CC
pivot_Loan_CC <- table(train_data$Loan, train_data$CC)

print(pivot_Loan_Online)
```

```
##
##        0    1
##   0 1102 1620
##   1   99  179
```

```
print(pivot_Loan_CC)
```

```
##
##        0    1
##   0 1930  792
##   1  187   91
```

Solution D(i) - Computation of the Quantities

```
#P(CC = 1 | Loan = 1)
prob_CC1_Loan1 <- pivot_Loan_CC[2, 2]
prob_CC1_Loan1_count <- sum(pivot_Loan_CC[, 2])
prob_CC1_given_Loan1 <- prob_CC1_Loan1/prob_CC1_Loan1_count
print(prob_CC1_given_Loan1)
```

```
## [1] 0.1030578
```

D(ii)

```
#P(Online = 1 | Loan = 1)
prob_Online1_Loan1 <- pivot_Loan_Online[2, 2]
prob_Online1_Loan1_count <- sum(pivot_Loan_Online[, 2])
prob_Online1_given_Loan1 <- prob_Online1_Loan1/prob_Online1_Loan1_count
print(prob_Online1_given_Loan1)
```

```
## [1] 0.09949972
```

D(iii)

```
#P(Loan = 1)
total_count <- nrow(train_data)
count_loan1 <- sum(train_data$Loan == 1)

# Calculate the proportion
prob_Loan1 <- count_loan1 / total_count
print(prob_Loan1)
```

```
## [1] 0.09266667
```

D(iv)

```
#P(CC = 1 | Loan = 0)
prob_CC1_Loan0 <- pivot_Loan_CC[1, 2]
prob_CC1_Loan0_count <- sum(pivot_Loan_CC[1,])
prob_CC1_given_Loan0 <- prob_CC1_Loan0/prob_CC1_Loan0_count
print(prob_CC1_given_Loan0)
```

```
## [1] 0.2909625
```

D(v)

```
#P(Online = 1 | Loan = 0)
prob_Online1_Loan0 <- pivot_Loan_Online[1, 2]
prob_Online1_Loan0_count <- sum(pivot_Loan_Online[1,])
prob_given_Online1_Loan0 <- prob_Online1_Loan0/prob_Online1_Loan0_count
print(prob_given_Online1_Loan0)
```

```
## [1] 0.5951506
```

D(vi)

```
#P(Loan = 0)
total_count1 <- nrow(train_data)
count_loan0 <- sum(train_data$Loan == 0)

# Calculate the proportion
prob_Loan0 <- count_loan0 / total_count1
print(prob_Loan0)
```

```
## [1] 0.9073333
```

Solution E - Naive Bayes probability computation

```
# Using the quantities computed above to compute the naive Bayes probability P(Loan = 1 | CC = 1, Online = 1).
prob_Loan1_CC1_Online1 <- (prob_CC1_given_Loan1 * prob_Online1_given_Loan1 * prob_Loan1) /    (prob_CC1_given_Loan0 * prob_given_Online1_Loan0)

print(prob_Loan1_CC1_Online1)
```

```
## [1] 0.005487343
```

Solution F - Value Comparison of (E) and (B)

```
#To compare Naive Bayes probability with pivot table value in result_compared.
result_compared <- c(Naive_Bayes = prob_Loan1_CC1_Online1, pivot_table = prob_Loan_given_CC_Online)

print(result_compared)
```

```
## Naive_Bayes pivot_table
## 0.005487343 0.626373626
```

Accurate estimate between compared values

```
# Ensure that train_data$Loan is a factor with specific levels
train_data$Loan <- factor(train_data$Loan, levels = c("0", "1"))

# Train the Naive Bayes model
model <- naiveBayes(Loan ~ ., data = train_data)

# Re-run the Naive Bayes prediction to ensure compatibility
nb_predictions <- predict(model, newdata = train_data, type = "class")

# Make sure nb_predictions is a factor with the same levels as train_data$Loan
nb_predictions <- factor(nb_predictions, levels = levels(train_data$Loan))

# Calculate accuracy metrics for Naive Bayes
confusionMatrix(nb_predictions, train_data$Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2506  121
##          1  216  157
##
##                Accuracy : 0.8877
##                  95% CI : (0.8758, 0.8988)
##     No Information Rate : 0.9073
##     P-Value [Acc > NIR] : 0.9999
##
##                   Kappa : 0.4208
##
##  Mcnemar's Test P-Value : 3.047e-07
##
##             Sensitivity : 0.9206
##             Specificity : 0.5647
##          Pos Pred Value : 0.9539
##          Neg Pred Value : 0.4209
##              Prevalence : 0.9073
##          Detection Rate : 0.8353
##    Detection Prevalence : 0.8757
##       Balanced Accuracy : 0.7427
##
##        'Positive' Class : 0
##
```

```
# Generate pivot table predictions
pivot_predictions <- ifelse(train_data$CC == 1 & train_data$Online == 1, "1", "0")

# Ensure pivot_predictions is a factor with the same levels as train_data$Loan
pivot_predictions <- factor(pivot_predictions, levels = levels(train_data$Loan))

# Calculate accuracy metrics for Pivot Table predictions
confusionMatrix(pivot_predictions, train_data$Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2247  221
##          1  475   57
##
##                Accuracy : 0.768
##                  95% CI : (0.7525, 0.783)
##     No Information Rate : 0.9073
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.0217
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.8255
##             Specificity : 0.2050
##          Pos Pred Value : 0.9105
##          Neg Pred Value : 0.1071
##              Prevalence : 0.9073
##          Detection Rate : 0.7490
##    Detection Prevalence : 0.8227
##       Balanced Accuracy : 0.5153
##
##        'Positive' Class : 0
##
```

Solution G

```
#Train the model using the 'naiveBayes' function
model <- naiveBayes(Loan ~ CC + Online, data = train_data)

# Assuming 'train_data' contains at least one row with CC = 1 and Online = 1
probabilities <- predict(model, newdata = train_data, type = "raw")

# Extracting the probability of Loan = 1 for the specific condition (CC = 1, Online = 1)
# Assuming the first row of train_data corresponds to this condition
prob_Loan1_given_CC1_given_Online1 <- probabilities[1, "1"]
print(prob_Loan1_given_CC1_given_Online1)
```

```
##         1
## 0.09402904
```

```r
# Compare the manually computed Naive Bayes probability with the model's prediction
manual_prob <- prob_Loan1_CC1_Online1
model_prob <- prob_Loan1_given_CC1_given_Online1

# To calculate and print the absolute difference between the manual and model probabilities
difference <- abs(manual_prob - model_prob)
print(paste("Difference between manual and model probability:", difference))
```

```
## [1] "Difference between manual and model probability: 0.0885416954936669"
```