Assignment 3 - Naive Bayes Classification

Stephen Gombos

2025-10-11

## Introduction

Use the UniversalBank.csv dataset to build a Naive Bayes classifier. The goal is to predict whether a customer will accept a personal loan (Personal.Loan) based on two predictor variables: if they are an online banking user (Online) and if they hold a credit card with the bank (CreditCard).

### 1. Load Libraries

First, we load the necessary R libraries.

library(e1071)  
library(caret)

## Loading required package: ggplot2

##   
## Attaching package: 'ggplot2'

## The following object is masked from 'package:e1071':  
##   
## element

## Loading required package: lattice

library(reshape2)

### 2. Load and Prepare Data

We load the dataset and prepare the categorical variables for analysis by converting them to factors. This ensures that R treats them as distinct categories rather than numerical values.

# Load the dataset  
data <- read.csv("/Users/stephengombos/Documents/KSU MBA PROGRAM/Fall 2025 FUNDAMENTALS OF MACHINE LEARNING (BA-64060-002)/CSV Files/UniversalBank.CSV")  
  
# Rename columns for easier use and convert predictors to factors  
data$CC <- as.factor(data$CreditCard)  
data$Online <- as.factor(data$Online)  
data$Loan <- as.factor(data$Personal.Loan)

### 3. Partition Data

The data is partitioned into a 60% training set and a 40% validation set. The training set will be used to build our model and perform manual calculations, while the validation set would typically be used to test the model’s performance on unseen data. We set a seed for reproducibility.

set.seed(100)  
trainIndex <- createDataPartition(data$Loan, p = 0.6, list = FALSE)  
train <- data[trainIndex, ]  
valid <- data[-trainIndex, ]

### A. Create Pivot Table

**Question:** 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[cite: 10].

**Explanation:** We use the melt() function to convert the data from a wide format to a long format, which is an intermediate step. Then, dcast() is used to reshape the long-format data into the desired pivot table, with length as the aggregation function to count the occurrences of each combination.

# Melt the data to prepare for casting  
melted\_data <- melt(train, id.vars = c("Online", "CC", "Loan"))  
  
# Create the pivot table by casting the melted data. 'length' counts the number of records.  
pivot\_A <- dcast(melted\_data, CC + Loan ~ Online, fun.aggregate = length)  
  
# Output the pivot table as a nicely formatted table  
knitr::kable(pivot\_A, caption = "Pivot Table: CC (rows), Loan (subrows), Online (columns)")

Pivot Table: CC (rows), Loan (subrows), Online (columns)

| CC | Loan | 0 | 1 |
| --- | --- | --- | --- |
| 0 | 0 | 9932 | 14963 |
| 0 | 1 | 1027 | 1573 |
| 1 | 0 | 4108 | 6253 |
| 1 | 1 | 455 | 689 |

**Result:** The pivot table shows the number of customers for each of the 8 possible combinations of Credit Card ownership (CC), Loan acceptance (Loan), and Online banking usage (Online).For example, there are 689 customers who have a credit card (CC=1), accepted the loan (Loan=1), and use online banking (Online=1). Similarly, there are 455 customers who have a credit card (CC=1), accepted the loan (Loan=1), but do not use online banking (Online=0).

### B. Empirical Probability from Pivot Table

**Question:** Looking at the pivot table, what is the probability that a customer who owns a bank credit card and is actively using online banking services will accept the loan offer?

**Explanation:** This is the empirical probability, calculated directly from the data. We find the total number of customers who have CC=1 and Online=1, and then find what fraction of that group also has Loan=1.

# Count of customers with Loan=1, CC=1, and Online=1  
num\_L1\_CC1\_O1 <- pivot\_A[pivot\_A$CC == 1 & pivot\_A$Loan == 1, "1"]  
  
# Count of all customers with CC=1 and Online=1  
# This is the sum of those with Loan=0 and Loan=1 for that category  
num\_CC1\_O1 <- sum(pivot\_A[pivot\_A$CC == 1, "1"])  
  
# Calculate the conditional probability  
prob\_L1\_given\_CC1\_Online1 <- num\_L1\_CC1\_O1 / num\_CC1\_O1  
  
print(paste("B. The empirical probability P(Loan=1 | CC=1, Online=1) is:", prob\_L1\_given\_CC1\_Online1))

## [1] "B. The empirical probability P(Loan=1 | CC=1, Online=1) is: 0.099250936329588"

**Result:** The probability based on direct observation is approximately 9.92%. This is our baseline “true” probability from the training data.

### C. Separate Pivot Tables

**Question:** Create two separate pivot tables for the training data.One for Loan vs. Online, and one for Loan vs. CC.

**Explanation:** These tables are required to compute the individual probabilities needed for the Naive Bayes formula. The table() function is a straightforward way to get these counts.

# Create a contingency table of Loan vs. Online  
pivot\_online <- table(train$Loan, train$Online)  
knitr::kable(pivot\_online, caption = "Pivot Table of Loan by Online")

Pivot Table of Loan by Online

|  | 0 | 1 |
| --- | --- | --- |
| 0 | 1080 | 1632 |
| 1 | 114 | 174 |

# Create a contingency table of Loan vs. CC  
pivot\_cc <- table(train$Loan, train$CC)  
knitr::kable(pivot\_cc, caption = "Pivot Table of Loan by CC")

Pivot Table of Loan by CC

|  | 0 | 1 |
| --- | --- | --- |
| 0 | 1915 | 797 |
| 1 | 200 | 88 |

### D. Compute Conditional and Marginal Probabilities

**Question:** Compute the six specified probabilities using the tables from part C.

**Explanation:** We calculate each probability by dividing the relevant count from the pivot tables by the appropriate total. These are the building blocks for our manual Naive Bayes calculation.

# i. P(CC=1 | Loan=1) = (Loan acceptors with CC) / (Total loan acceptors)  
p\_cc1\_loan1 <- pivot\_cc["1", "1"] / sum(pivot\_cc["1", ])  
  
# ii. P(Online=1 | Loan=1) = (Loan acceptors who are online) / (Total loan acceptors)  
p\_online1\_loan1 <- pivot\_online["1", "1"] / sum(pivot\_online["1", ])  
  
# iii. P(Loan=1) = (Total loan acceptors) / (Total customers)  
p\_loan1 <- sum(pivot\_cc["1", ]) / nrow(train)  
  
# iv. P(CC=1 | Loan=0) = (Loan non-acceptors with CC) / (Total loan non-acceptors)  
p\_cc1\_loan0 <- pivot\_cc["0", "1"] / sum(pivot\_cc["0", ])  
  
# v. P(Online=1 | Loan=0) = (Loan non-acceptors who are online) / (Total loan non-acceptors)  
p\_online1\_loan0 <- pivot\_online["0", "1"] / sum(pivot\_online["0", ])  
  
# vi. P(Loan=0) = (Total loan non-acceptors) / (Total customers)  
p\_loan0 <- sum(pivot\_cc["0", ]) / nrow(train)  
  
# Create a summary table for the probabilities  
prob\_table <- data.frame(  
 Probability = c("P(CC=1|Loan=1)", "P(Online=1|Loan=1)", "P(Loan=1)",   
 "P(CC=1|Loan=0)", "P(Online=1|Loan=0)", "P(Loan=0)"),  
 Value = c(p\_cc1\_loan1, p\_online1\_loan1, p\_loan1, p\_cc1\_loan0, p\_online1\_loan0, p\_loan0)  
)  
  
knitr::kable(prob\_table, caption = "Naive Bayes Probabilities (Part D)")

Naive Bayes Probabilities (Part D)

| Probability | Value |
| --- | --- |
| P(CC=1|Loan=1) | 0.3055556 |
| P(Online=1|Loan=1) | 0.6041667 |
| P(Loan=1) | 0.0960000 |
| P(CC=1|Loan=0) | 0.2938791 |
| P(Online=1|Loan=0) | 0.6017699 |
| P(Loan=0) | 0.9040000 |

### E. Manual Naive Bayes Calculation

**Question:** Use the quantities computed above to compute the naive Bayes probability .

**Explanation:** The Naive Bayes classifier calculates a “score” for each possible outcome (class) by multiplying the prior probability of that class by the conditional probabilities of the evidence. The “naive” assumption is that the predictors (CC and Online) are independent of each other, given the class (Loan). After calculating the scores for Loan=1 and Loan=0, we normalize them to get the final probability.

# Step 1: Calculate the 'score' for the Loan=1 class  
# This is P(Loan=1) \* P(CC=1|Loan=1) \* P(Online=1|Loan=1)  
score\_loan1 <- p\_loan1 \* p\_cc1\_loan1 \* p\_online1\_loan1  
  
# Step 2: Calculate the 'score' for the Loan=0 class  
# This is P(Loan=0) \* P(CC=1|Loan=0) \* P(Online=1|Loan=0)  
score\_loan0 <- p\_loan0 \* p\_cc1\_loan0 \* p\_online1\_loan0  
  
# Step 3: Normalize the scores to get the final probability for Loan=1  
prob\_nb\_manual <- score\_loan1 / (score\_loan1 + score\_loan0)  
  
print(paste("E. Naive Bayes Estimate P(Loan=1|CC=1, Online=1):", prob\_nb\_manual))

## [1] "E. Naive Bayes Estimate P(Loan=1|CC=1, Online=1): 0.0997915415131373"

### F. Compare Empirical vs. Naive Bayes Estimate

**Question:** Compare this value with the one obtained from the pivot table in (B). Which is a more accurate estimate?

print("F. Comparison of Estimates:")

## [1] "F. Comparison of Estimates:"

print(paste(" - Empirical Probability (from B):", prob\_L1\_given\_CC1\_Online1))

## [1] " - Empirical Probability (from B): 0.099250936329588"

print(paste(" - Naive Bayes Estimate (from E):", prob\_nb\_manual))

## [1] " - Naive Bayes Estimate (from E): 0.0997915415131373"

## **Explanation:** The empirical probability (≈0.0993) is a more accurate estimate for the training data itself. This is because it is a direct measurement of the proportion of customers in that specific subgroup (CC=1 and Online=1) who accepted the loan (Loan=1). The Naive Bayes estimate (≈0.0998) is a model-based approximation that relies on the strong assumption of conditional independence between CC and Online, which is rarely perfectly true in reality.

### G. Pivot Cells Needed

**Question:** Which of the entries in this table are needed for computing ?

**Explanation:** To perform the Naive Bayes calculation, we did not use the combined pivot table from part A. Instead, we needed all the counts from the two separate pivot tables created in part C. These tables allowed us to calculate the six key probabilities in part D (the prior probabilities of the loan status and the conditional probabilities of the predictors given the loan status).

### H. Run Naive Bayes Model and Find Probability

**Question:** Run naive Bayes on the data. [cite\_start]Examine the model output on training data, and find the entry that corresponds to [cite: 32, 33].

**Explanation:** We use the naiveBayes() function from the e1071 library to create the model. We then use the predict() function with type="raw" to get the model’s computed probabilities for each record in the training set. Finally, we filter for the records where CC=1 and Online=1 and find the corresponding probability for the Loan=1 class.

# Run the Naive Bayes model using the training data  
nb\_model <- naiveBayes(Loan ~ CC + Online, data = train)  
  
# Predict probabilities on the training data  
pred\_train <- predict(nb\_model, train, type = "raw")  
  
# Isolate the predicted probability for the specific case where CC=1 and Online=1  
# The '1' column corresponds to the probability of Loan=1  
# We take the mean because it will be the same value for all such instances.  
model\_prob <- mean(pred\_train[train$CC == 1 & train$Online == 1, "1"])  
  
print(paste("H. Model Output for P(Loan=1|CC=1, Online=1):", model\_prob))

## [1] "H. Model Output for P(Loan=1|CC=1, Online=1): 0.0997915415131373"

### I. Final Comparison

**Question:** Compare this to the number you obtained in (E).

# Create a summary table for the probability comparisons  
compare\_probs <- data.frame(  
 Estimate = c("Empirical Probability (Part B)", "Manual NB Calculation (Part E)", "Model Output (Part H)"),  
 Value = c(round(prob\_L1\_given\_CC1\_Online1,4), round(prob\_nb\_manual,4), round(model\_prob,4))  
)  
  
knitr::kable(compare\_probs, caption = "Comparison of Probabilities (Parts B, E, H)")

Comparison of Probabilities (Parts B, E, H)

| Estimate | Value |
| --- | --- |
| Empirical Probability (Part B) | 0.0993 |
| Manual NB Calculation (Part E) | 0.0998 |
| Model Output (Part H) | 0.0998 |

**Conclusion:** The manual Naive Bayes calculation from Part E (0.0998) and the R model’s output from Part H (0.0998) are identical. This confirms that our manual calculation correctly replicated the algorithm used by the naiveBayes function. The small difference between this model-based probability and the empirical probability (0.0993) is due to the model’s simplifying “naive” assumption of conditional independence.