The PiñacoDatas

Data Analysis for Business - Midterm Project

09-04-2024

Chiara Baldoni 285441, Sofia Bruni 285231, Francesca Romana Sanna 282491, Mattia Sebastiani 288071

Part I: Classification

Exploratory Data Analysis

IMPORTING THE DATASETS

```
# Importing the train data-set
bank_accounts_train = read.csv("csv/bank_accounts_train.csv", sep = ",", dec = ".", header = T, colClass

print("Train dataframe:")
str(bank_accounts_train)


# Importing the test data-set
bank_accounts_test <- read.csv("csv/bank_accounts_test.csv", sep = ",", dec = ".", header = T, colClasse

print("Test dataframe:")
str(bank_accounts_test)
```

CLEANING THE DATASETS

```
# Defining categorical variables
catVar <- c('Gender', 'Education_Level', 'Marital_Status', 'Card_Category')

# Replacing all occurrences of "Unknown" with NA in the specified categorical features using the dplyr
cleaned_bank_accounts_train <- bank_accounts_train %>% mutate_at(catVar, ~na_if(., "Unknown"))
cleaned_bank_accounts_test <- bank_accounts_test %>% mutate_at(catVar, ~na_if(., "Unknown"))

# Checking for duplicate IDs
length(unique(cleaned_bank_accounts_train$CLIENTNUM))
n_occur <- data.frame(table(cleaned_bank_accounts_train$CLIENTNUM))
n_occur[n_occur$Freq > 1,]
cleaned_bank_accounts_train[cleaned_bank_accounts_train$CLIENTNUM %in% n_occur$Var1[n_occur$Freq > 1],]
cleaned_bank_accounts_train <- slice(cleaned_bank_accounts_train, 1:(n()-2))
```

```r
length(unique(cleaned_bank_accounts_test$CLIENTNUM))
n_occur <- data.frame(table(cleaned_bank_accounts_test$CLIENTNUM))
n_occur[n_occur$Freq > 1,] # no duplicates here

# Converting numerical variables from character strings to numeric values by iterating through the list
numVarInChar <- c('Customer_Age', 'Dependent_count', 'Months_on_book', 'Total_Relationship_Count', 'Mon
for (var in numVarInChar) {
  cleaned_bank_accounts_train[[var]] <- as.numeric(cleaned_bank_accounts_train[[var]])
  cleaned_bank_accounts_test[[var]] <- as.numeric(cleaned_bank_accounts_test[[var]])
}
cleaned_bank_accounts_train[['Closed_Account']] <- as.numeric(cleaned_bank_accounts_train[['Closed_Acco

# Printing the cleaned data-frames
print("Cleaned train dataframe:")
str(cleaned_bank_accounts_train)

print("Cleaned test dataframe:")
str(cleaned_bank_accounts_test)
```

ENCODING CATEGORICAL VALUES IN THE DATASETS

```r
# Factorizing categorical values in the data frames by iterating through the list of categorical variab
encoded_bank_accounts_train <- cleaned_bank_accounts_train
encoded_bank_accounts_test <- cleaned_bank_accounts_test

fact_bank_accounts_train <- cleaned_bank_accounts_train
fact_bank_accounts_test <- cleaned_bank_accounts_test

for (var in catVar) {
  fact_bank_accounts_train[[var]] <- as.factor(encoded_bank_accounts_train[[var]])
  fact_bank_accounts_test[[var]] <- as.factor(encoded_bank_accounts_test[[var]])
}

# One-hot encoding the categorical variables using the mltools library.
encoded_bank_accounts_train <- one_hot(as.data.table(fact_bank_accounts_train))
encoded_bank_accounts_test <- one_hot(as.data.table(fact_bank_accounts_test))

# Printing the encoded data frames.
str(encoded_bank_accounts_train)
str(encoded_bank_accounts_test)
```

DATA VISUALIZATION

```r
# Visualizing number of NA values count for each variable
colSums(is.na(cleaned_bank_accounts_train))
```

```
##                CLIENTNUM             Customer_Age                   Gender
##                        0                        0                        0
##          Dependent_count          Education_Level           Marital_Status
##                        0                     1127                      565
##            Card_Category           Months_on_book Total_Relationship_Count
##                        0                        0                        0
##   Months_Inactive_12_mon    Contacts_Count_12_mon             Credit_Limit
##                        0                        0                        0
##      Total_Revolving_Bal          Avg_Open_To_Buy     Total_Amt_Chng_Q4_Q1
##                        0                        0                        0
##          Total_Trans_Amt           Total_Trans_Ct      Total_Ct_Chng_Q4_Q1
##                        0                        0                        0
##    Avg_Utilization_Ratio                   Income           Closed_Account
##                        0                        0                        0
```

```
colSums(is.na(cleaned_bank_accounts_test))
```

```
##                CLIENTNUM             Customer_Age                   Gender
##                        0                        0                        0
##          Dependent_count          Education_Level           Marital_Status
##                        0                      392                      184
##            Card_Category           Months_on_book Total_Relationship_Count
##                        0                        0                        0
##   Months_Inactive_12_mon    Contacts_Count_12_mon             Credit_Limit
##                        0                        0                        0
##      Total_Revolving_Bal          Avg_Open_To_Buy     Total_Amt_Chng_Q4_Q1
##                        0                        0                        0
##          Total_Trans_Amt           Total_Trans_Ct      Total_Ct_Chng_Q4_Q1
##                        0                        0                        0
##    Avg_Utilization_Ratio                   Income
##                        0                        0
```

SUMMARIZING THE DATA

```
# printing the summary of the cleaned data-frames
summary(cleaned_bank_accounts_train)
```

```
##   CLIENTNUM          Customer_Age      Gender          Dependent_count
##  Length:7595        Min.   :26.00   Length:7595        Min.   :0.000
##  Class :character   1st Qu.:41.00   Class :character   1st Qu.:1.000
##  Mode  :character   Median :46.00   Mode  :character   Median :2.000
##                     Mean   :46.27                      Mean   :2.354
##                     3rd Qu.:52.00                      3rd Qu.:3.000
##                     Max.   :73.00                      Max.   :5.000
##  Education_Level    Marital_Status     Card_Category      Months_on_book
##  Length:7595        Length:7595        Length:7595        Min.   :13.00
##  Class :character   Class :character   Class :character   1st Qu.:31.00
##  Mode  :character   Mode  :character   Mode  :character   Median :36.00
##                                                           Mean   :35.86
##                                                           3rd Qu.:40.00
##                                                           Max.   :56.00
##  Total_Relationship_Count Months_Inactive_12_mon Contacts_Count_12_mon
##  Min.   :1.000            Min.   :0.000          Min.   :0.000
##  1st Qu.:3.000            1st Qu.:2.000          1st Qu.:2.000
##  Median :4.000            Median :2.000          Median :3.000
##  Mean   :3.809            Mean   :2.332          Mean   :2.469
##  3rd Qu.:5.000            3rd Qu.:3.000          3rd Qu.:3.000
##  Max.   :6.000            Max.   :6.000          Max.   :6.000
##   Credit_Limit   Total_Revolving_Bal Avg_Open_To_Buy Total_Amt_Chng_Q4_Q1
##  Min.   : 1439   Min.   :   0.0      Min.   :   14   Min.   :   0.0
##  1st Qu.: 2767   1st Qu.: 297.5      1st Qu.: 1462   1st Qu.: 579.0
##  Median : 5349   Median :1265.0      Median : 4199   Median : 715.0
##  Mean   : 9255   Mean   :1154.9      Mean   : 7974   Mean   : 691.7
##  3rd Qu.:13478   3rd Qu.:1776.0      3rd Qu.:11428   3rd Qu.: 846.0
##  Max.   :34516   Max.   :2517.0      Max.   :34516   Max.   :3355.0
##  Total_Trans_Amt Total_Trans_Ct   Total_Ct_Chng_Q4_Q1 Avg_Utilization_Ratio
##  Min.   :  510   Min.   : 10.00   Min.   :   0.0      Min.   :  0
##  1st Qu.: 2143   1st Qu.: 45.00   1st Qu.: 444.0      1st Qu.:  3
##  Median : 3895   Median : 67.00   Median : 656.0      Median :132
##  Mean   : 4405   Mean   : 64.76   Mean   : 593.7      Mean   :249
##  3rd Qu.: 4742   3rd Qu.: 81.00   3rd Qu.: 786.0      3rd Qu.:463
##  Max.   :18484   Max.   :139.00   Max.   :3714.0      Max.   :994
##      Income          Closed_Account
##  Min.   :  0.01425   Min.   :0.0000
##  1st Qu.: 53.85319   1st Qu.:0.0000
##  Median : 79.61139   Median :0.0000
##  Mean   : 90.99486   Mean   :0.1602
##  3rd Qu.:121.86149   3rd Qu.:0.0000
##  Max.   :199.94190   Max.   :1.0000
```

```
#summary(cleaned_bank_accounts_test)
```

PLOTTING GRAPHS

```
# Plotting the correlation matrix

#GGally::ggpairs(cleaned_bank_accounts_train, cardinality_threshold = 20, columns = (2:20))
```

DATA ANALYSIS

```
# Distribution of the variables answering 3,4,5
table(cleaned_bank_accounts_train$Closed_Account)
```

```
##
##    0    1
## 6378 1217
```

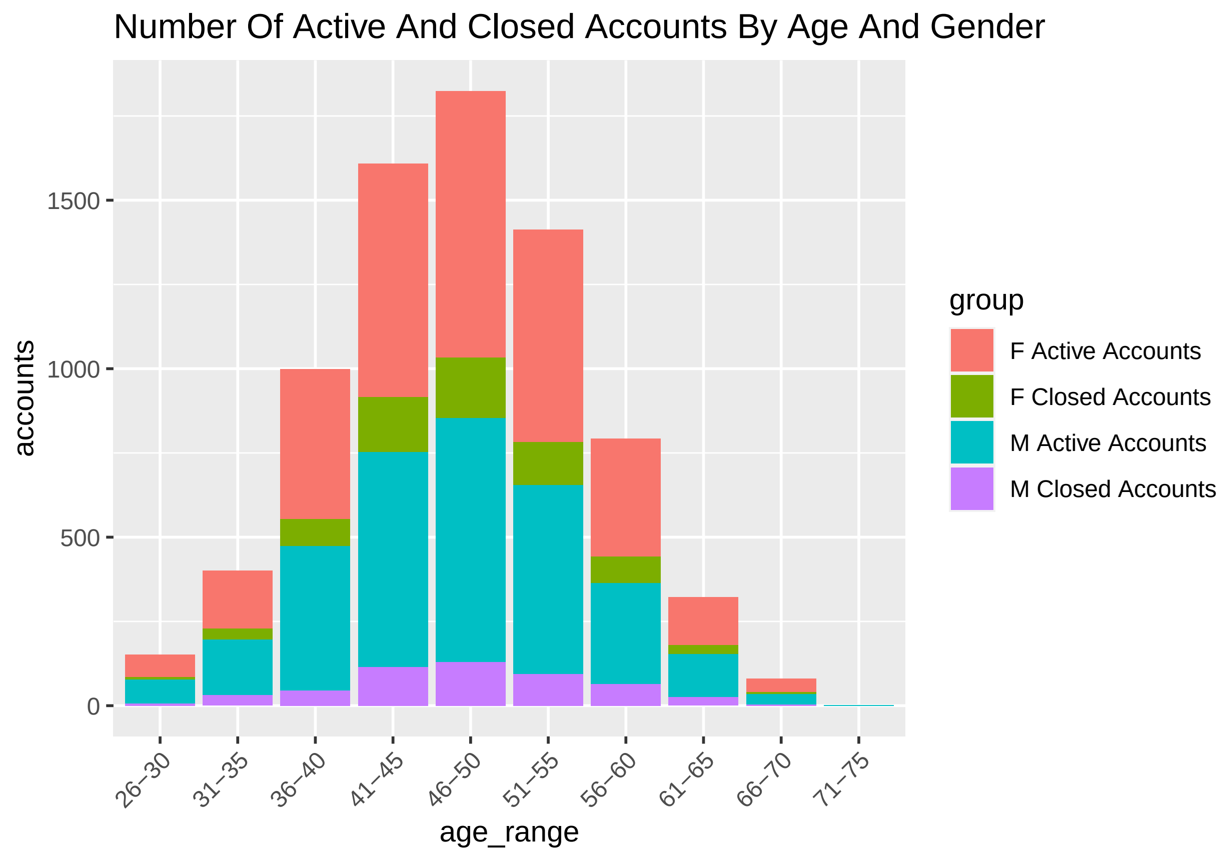
```r
prop.table(table(cleaned_bank_accounts_train$Closed_Account))
```

```
##
##        0        1
## 0.839763 0.160237
```

DATA PLOTS

We observe a peak in active accounts at ages 46-50 for both genders, indicating heightened *banking engagement in midlife*. Males generally have more active accounts than females, but the difference lessens around the age of 56-60. Closed accounts slightly increase around ages 46-50 but do not show a significant rise with age. The distribution of active accounts is slightly right-skewed, suggesting a **near-normal distribution** with a decrease in activity after age 50, reflecting fewer active accounts among older customers.

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



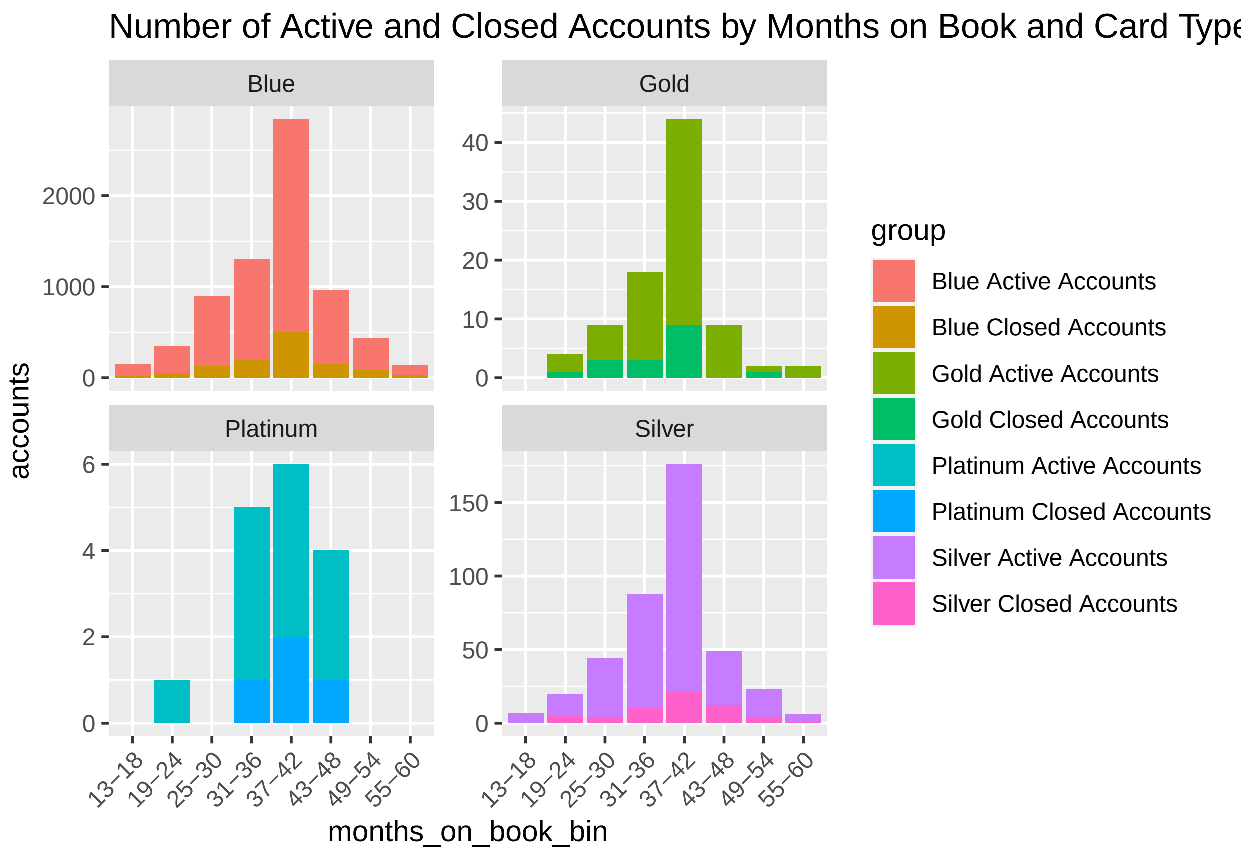
DATA PLOTS

Here we use stacked bar charts to explore potential correlations between the months an account has been open (`months_on_book_bin`), the account status (`Closed_Account`), and the card type (`Card_Category`).

Our goal is to statistically determine if these variables have a significant relationship and whether they influence account activity. The charts generated help us visually represent the trends based on the card type, thus we can observe:

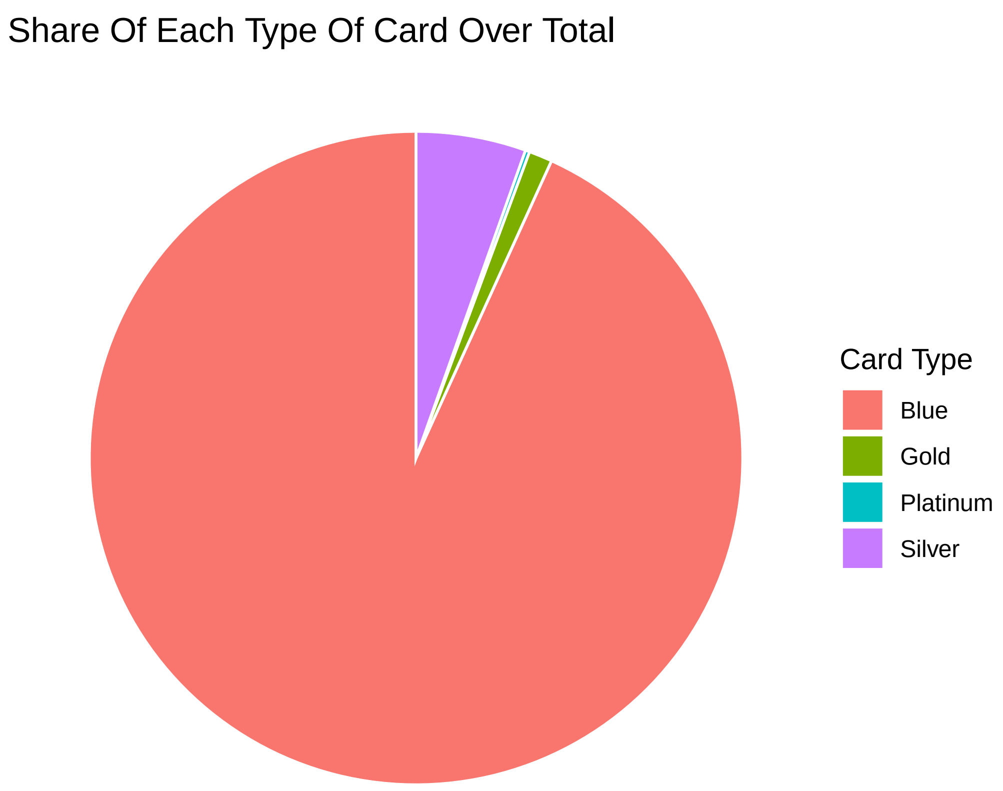
- for the *Blue Card*, the distribution of active accounts shows a peak around the 37-42 months interval; this suggests a mode of account activity indicating customer retention within this timeframe; the relatively flat distribution of closed accounts implies that most accounts tend to remain active over time.
- *Silver Card* accounts as well show a spike in activity within the same range and then drop off; the number of closed accounts seems consistent, indicating a turnover around two years.
- looking at the *Gold Card* we can see a peak in active accounts around 37-42 months, just like the Blue and Silver Card.
- *Platinum Card* accounts are fewer overall, with a peak in the number of total accounts between 31-42 months, while showing less fluctuation in activity, maintaining a steady number of closed accounts.

In general, active cards outnumber closed ones in time, with account activity peaking at 37-42 phase followed by a gradual decline, meaning a longer retention rate. The first three types of cards present a **near normal distribution graph**, slightly left skewed, while the Platinum card has two observable spikes, both in active and closed accounts, showing that most clients sign during the peak of their career.



DATA PLOTS

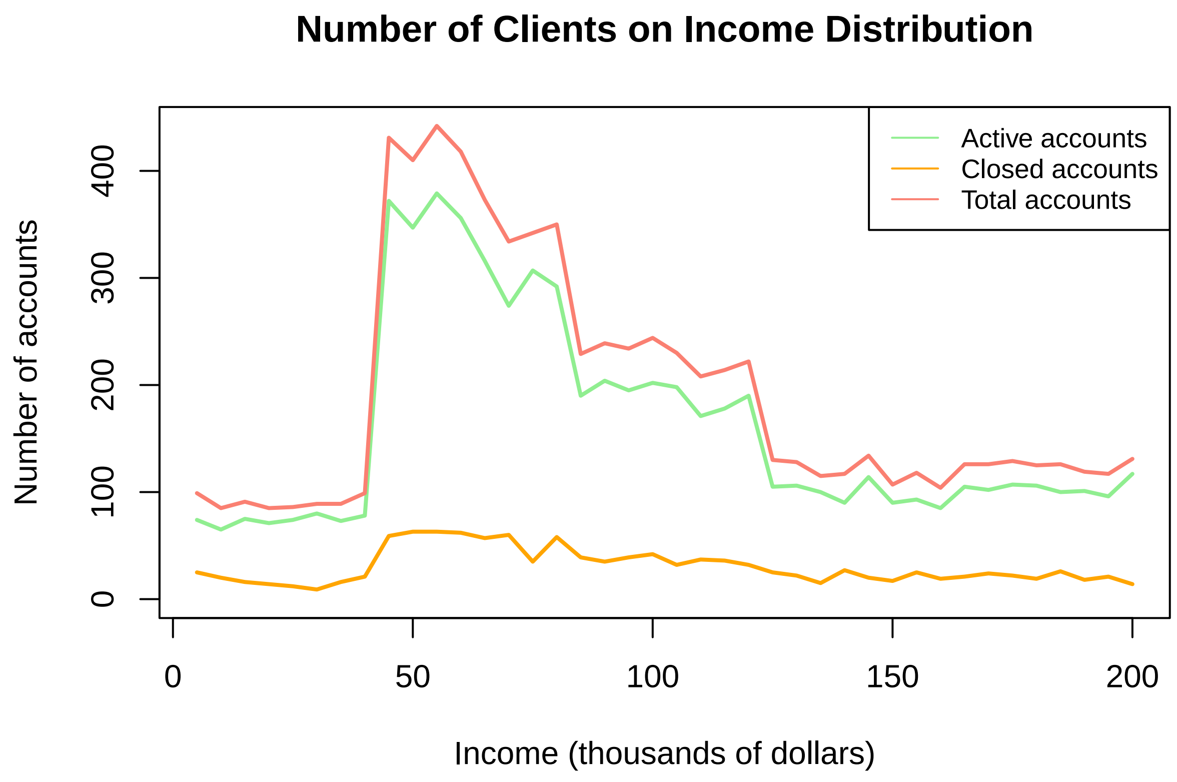
The *Blue card* is unsurprisingly the most common (93.2% share), followed by the *Silver card* (5.4% share), the *Gold* (1.2% share) and the *Platinum* (0.2% share). There is a **weak positive correlation** between the months on books and the type of card.



```
## # A tibble: 4 x 2
##   Card_type Percentage
##   <chr>          <dbl>
## 1 Blue          93.2
## 2 Gold           1.16
## 3 Platinum       0.211
## 4 Silver         5.44
```

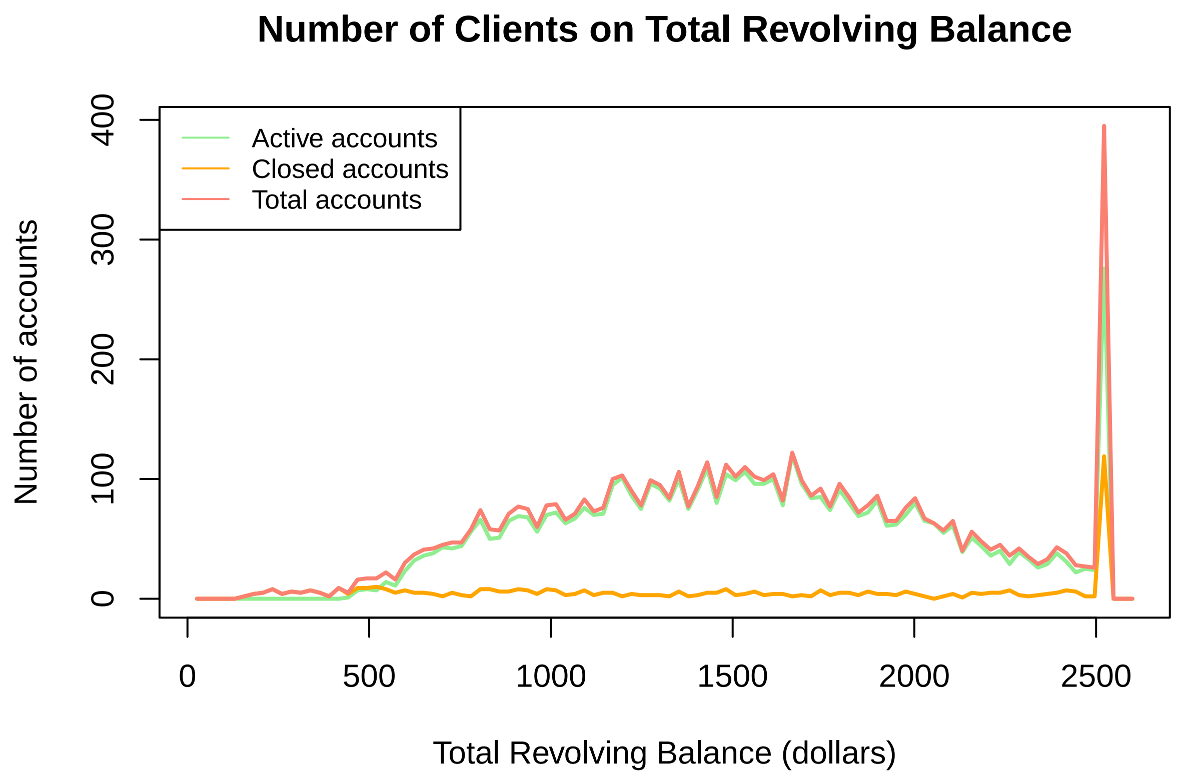
DATA PLOTS

The graph demonstrates that the percentage of closed accounts in relation to total accounts exhibits a slight *dependency on income* levels. There's a substantial drop in active accounts past the $50,000 mark, while closed accounts reduce at a slower rate. The middle-income range between $45,000 and $120,000 shows a lower ratio of closed to total accounts, indicating more stable account activity in this bracket. The most accounts are concentrated in the $50,000 to $75,000 income range, and beyond $120,000, total accounts become much rarer.



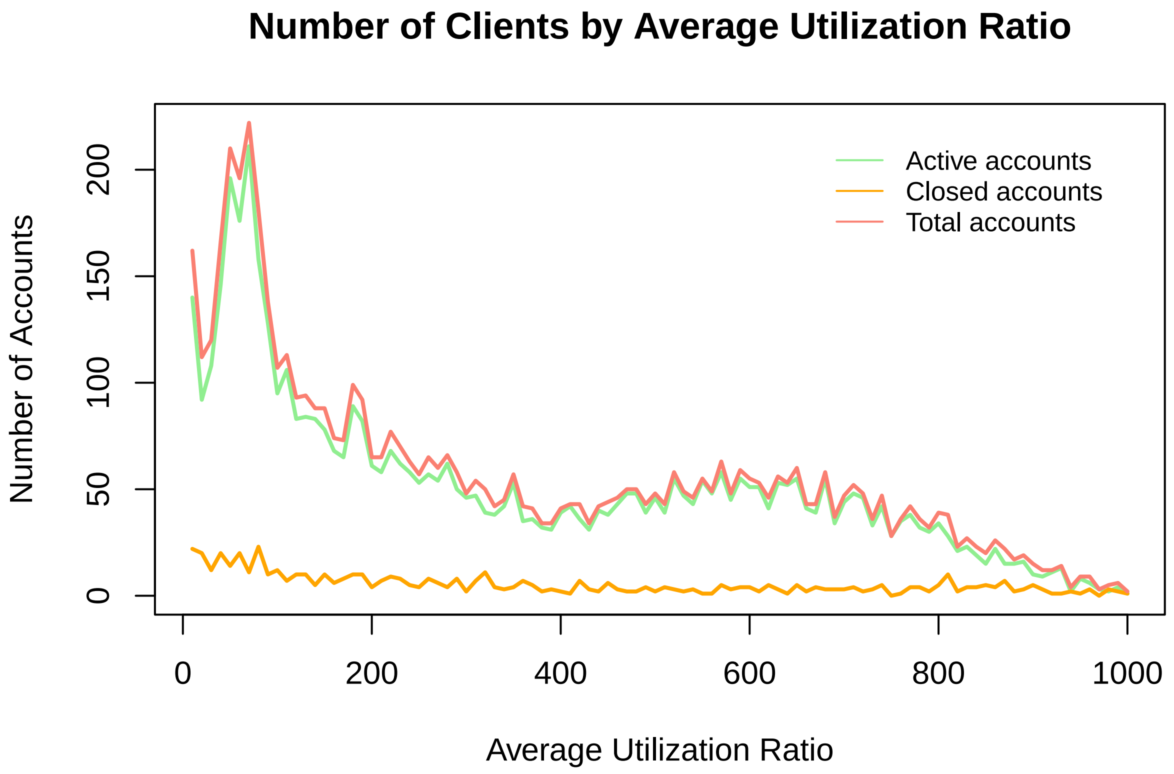
DATA PLOTS

Bank accounts are divided by their *total revolving balance* and categorized based on account status - either closed or active - incrementing ranges by $26 to formula the bins. Grouping the *number of accounts* within each bin, allows to visualize the *distribution patterns* across the balance variety. The following plot highlights the concentration of accounts around the median (going towards higher values) and reveals a notable spike at the higher end of the balance scale, possibly indicating outliers. The graph also shows a higher number of active accounts compared to closed accounts, suggesting a **positive correlation** between the total revolving balance and account activity.



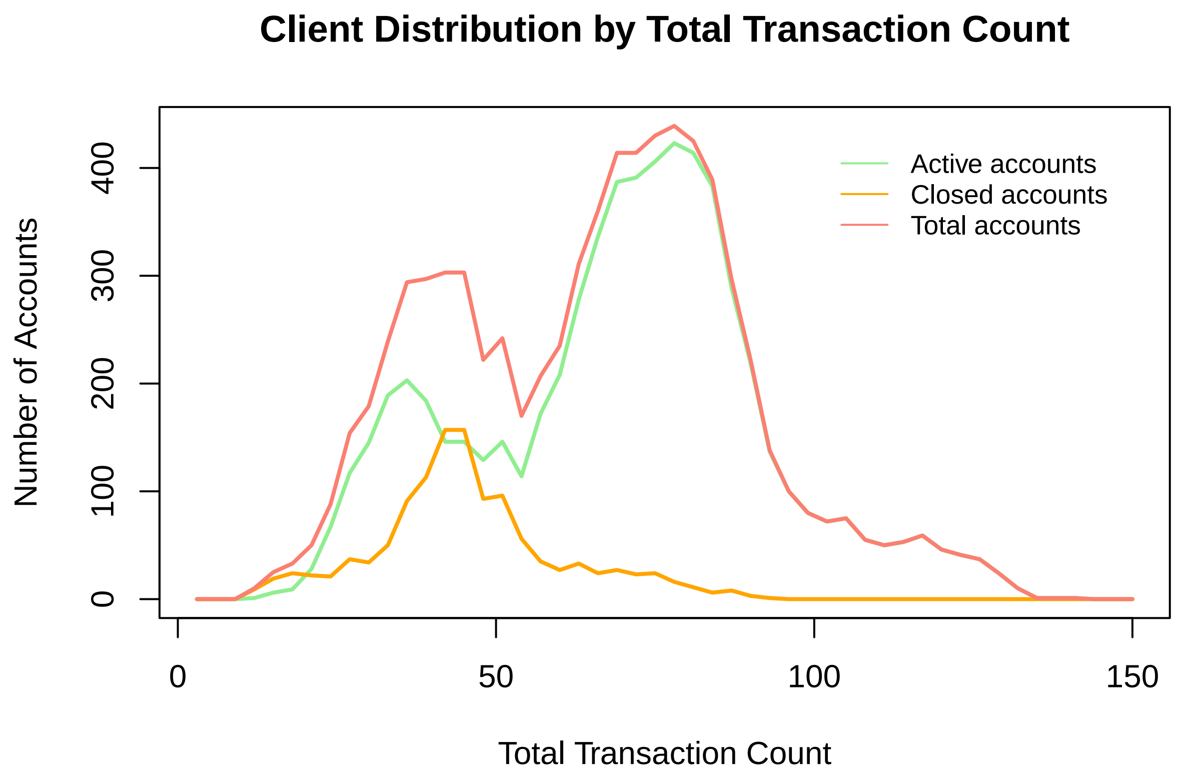
DATA PLOTS

The graph illustrates the distribution of accounts based on the *average utilization ratio*, which is the proportion of the *total revolving balance* to the *credit limit*. The utilization ratio is divided into bins of 10, and the number of accounts in each bin is calculated. The plot shows a higher number of active accounts compared to closed accounts, indicating a **positive correlation** between the utilization ratio and account activity. The distribution of active accounts is *right-skewed*, with a peak around the 10-20% utilization ratio, while the distribution of closed accounts is relatively *flat*, suggesting that most accounts remain active regardless of the utilization ratio.



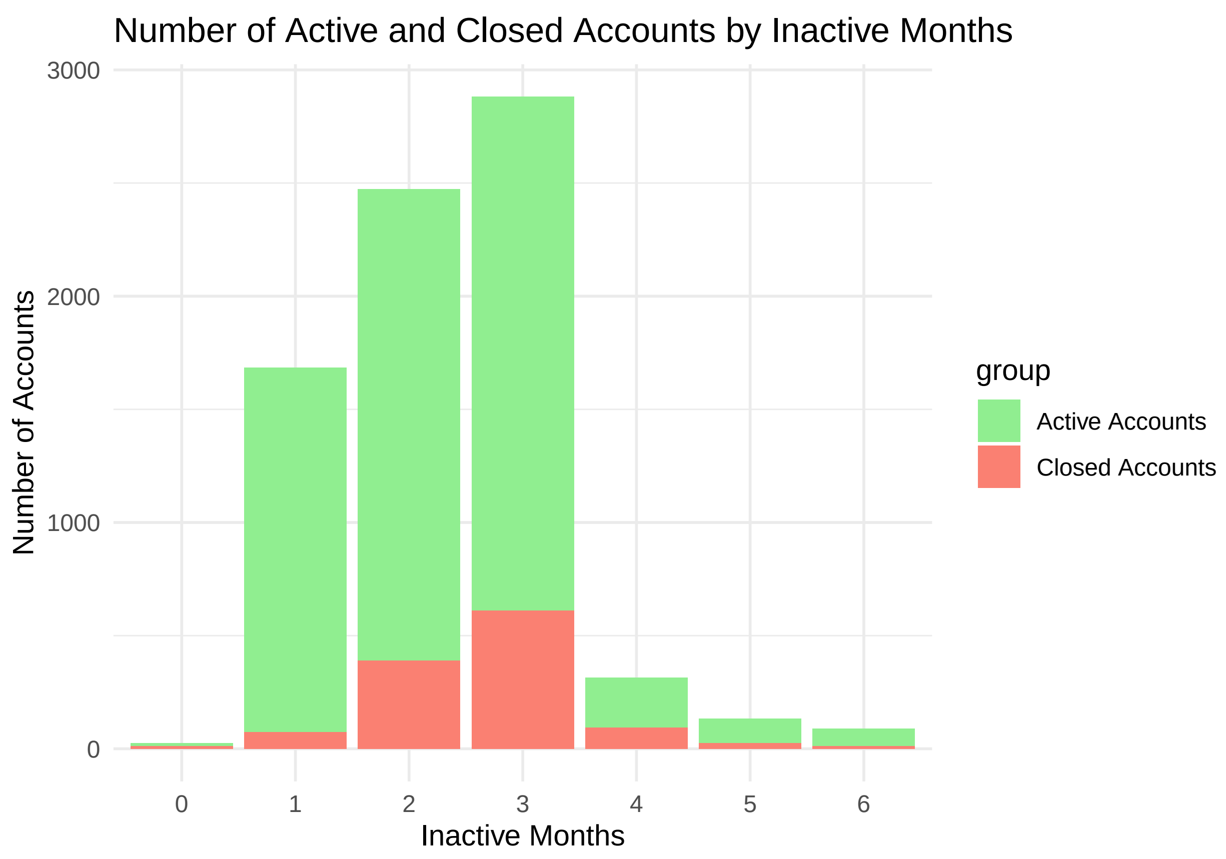
DATA PLOTS

Bank accounts are divided by their *total transaction count*, with the bins incremented by three transactions to categorize the accounts. The plot visualizes the *distribution patterns* of the number of accounts across varying transaction volumes. It reveals a prominent spike right at the median, suggesting a common *transactional behavior* among clients. Toward the higher transaction counts, there's a significant drop off, which may point to outliers or less frequent high-volume transactors. Notably, the graph indicates a higher number of active accounts in most transaction ranges, implying **positive correlation** between transaction activity and the likelihood of an account being active.



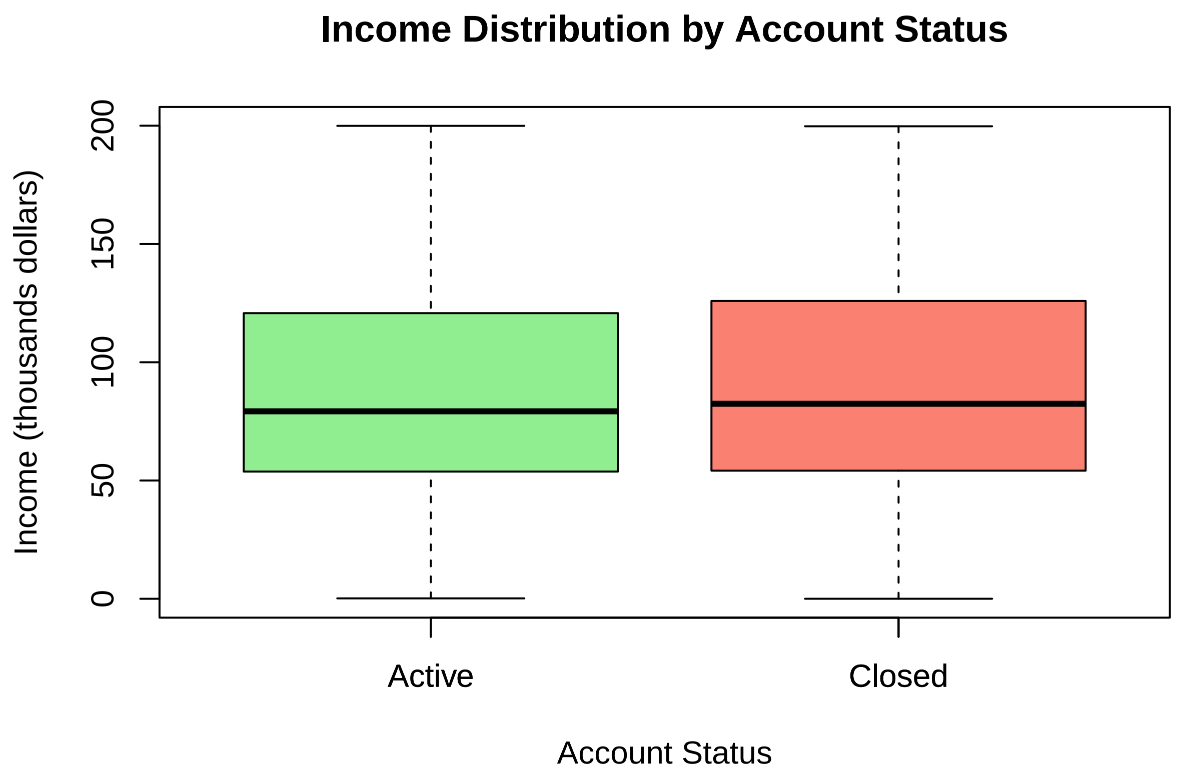
DATA PLOTS

The graph illustrates the distribution of accounts based on the number of *inactive months*, categorized as either active or closed. The plot shows a higher number of active accounts compared to closed accounts, indicating a **positive correlation** between the number of inactive months and account activity. The distribution of active accounts is *right-skewed*, with a peak around 2 inactive months, while the distribution of closed accounts is relatively *flat*, suggesting that most accounts remain active regardless of the number of inactive months.



DATA PLOTS

The boxplot compares the *income distribution* between active and closed accounts. The plot shows that the median income for active accounts is higher than that for closed accounts, suggesting a **positive correlation** between income and account activity. The income distribution for active accounts is *right-skewed*, with a peak around the median income, while the income distribution for closed accounts is relatively *flat*, indicating that most accounts remain active regardless of income level.



In conclusion, the data analysis demonstrates that account activity is related to demographic factors like age and income, as well as account features like the type of card, balance, utilization ratio, transaction count, and inactive months. The predominant number of active accounts compared to closed accounts indicates the data is unbalanced, which may affect the performance of predictive models.

LOGISTIC REGRESSION

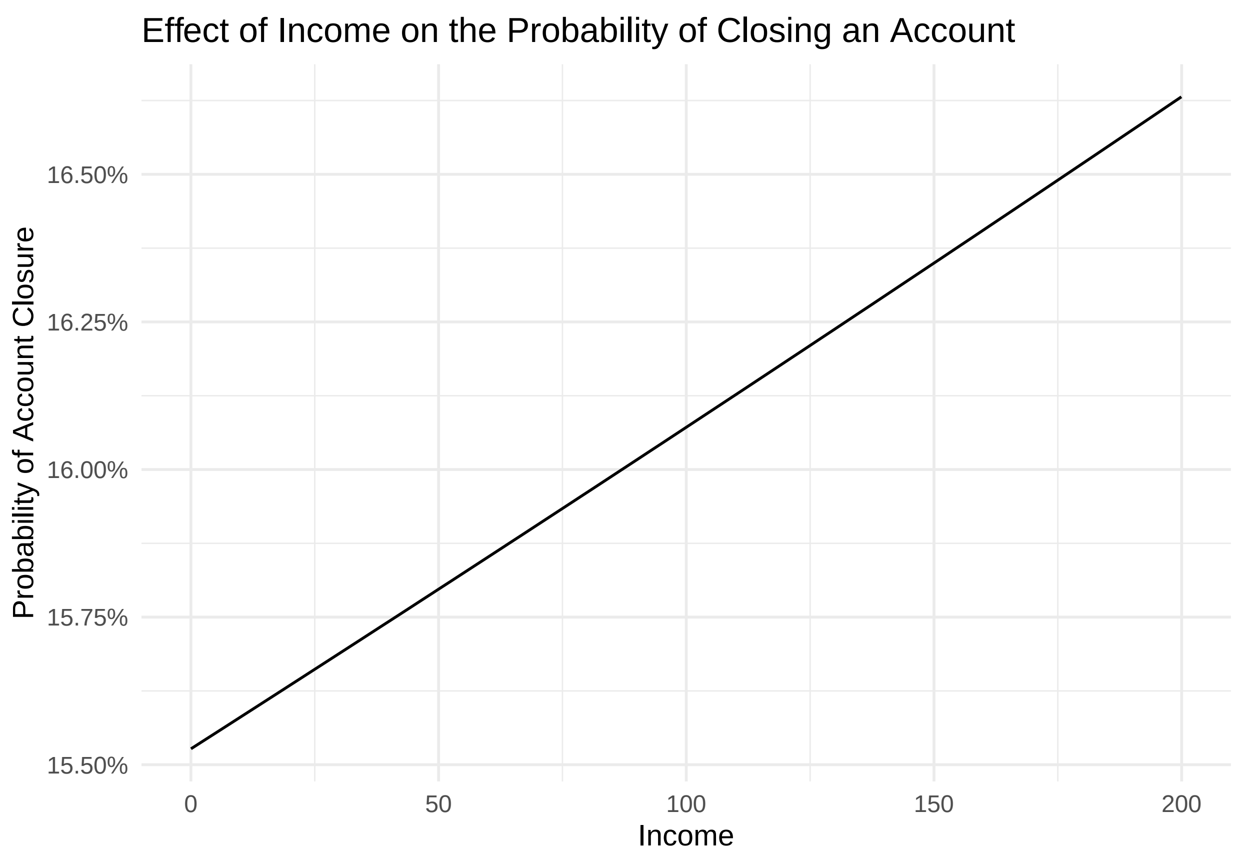
The **logistic regression model** is used to predict the probability of an account being closed based on the *income* of the account holder. The model is fitted using the `glm()` function with the *binomial family* and *logit link* function. The summary of the model provides information about the coefficients and significance levels of the predictors. The plot shows the effect of income on the probability of account closure, with income on the x-axis and the probability of account closure on the y-axis, demonstrating how the probability of account closure increases with higher income levels.

INCOME AS REGRESSOR

```
# Fit the logistic regression model
logit_model <- glm(Closed_Account ~ Income, data = cleaned_bank_accounts_train, family = binomial(link=

# Summarize the model
summary(logit_model)
```

```
##
## Call:
## glm(formula = Closed_Account ~ Income, family = binomial(link = "logit"),
##     data = cleaned_bank_accounts_train)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.6938616  0.0663566 -25.527   <2e-16 ***
## Income       0.0004095  0.0006385   0.641    0.521
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6684.6  on 7594  degrees of freedom
## Residual deviance: 6684.1  on 7593  degrees of freedom
## AIC: 6688.1
##
## Number of Fisher Scoring iterations: 3
```

```r
# Plot the logistic regression model
income_range <- with(cleaned_bank_accounts_train, seq(min(Income, na.rm = TRUE), max(Income, na.rm = TRU
newdata <- data.frame(Income = income_range)
newdata$Probability <- predict(logit_model, newdata = newdata, type = "response")

ggplot(newdata, aes(x = Income, y = Probability)) +
  geom_line() +
  labs(title = "Effect of Income on the Probability of Closing an Account",
       x = "Income",
       y = "Probability of Account Closure") +
  scale_y_continuous(labels = scales::percent_format()) +
  theme_minimal()
```



```
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold"),
  )
```

```
## List of 4
##  $ axis.title.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : chr "bold"
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y:List of 11
##   ..$ family       : NULL
##   ..$ face         : chr "bold"
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ plot.title  :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

GENDER AND INCOME

As seen before, we are now using logistic regression to look into the relationship between the **probability of account closure** and two predictors: *income* and *gender*. Logistic regression is particularly suited in modeling *binary outcomes*(0 when active, 1 when closed) by estimating probabilities with a logistic function. Including income and gender as independent variables, and their interaction term, we can assess the impact of a person's income and gender on the likelihood of closing their account and whether the income has a different influence for different genders.

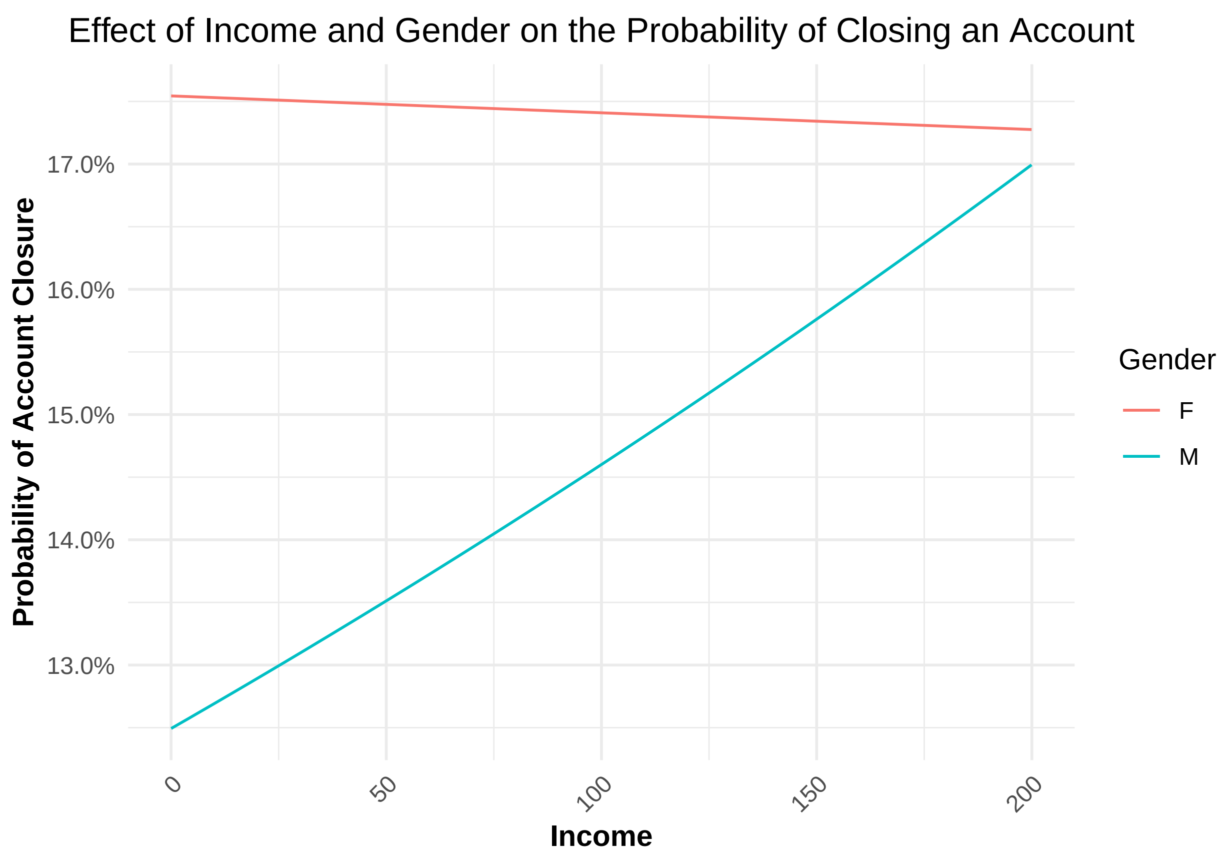
The summary gives us coefficients for income, gender, and the interaction between them, indicating how a unit change in these predictors alters the *log odds* of an account being closed, while keeping all other variables unchanged.

The plotted curve, visually represents the probability of account closure coming from predictions for a range of *income across gender groups*.

```r
# Fit the logistic regression model with gender and income as regressors
logit_model <- glm(Closed_Account ~ Income * Gender,
                   data = cleaned_bank_accounts_train,
                   family = binomial(link = "logit"))

# Summarize the model
summary(logit_model)
```

```
##
## Call:
## glm(formula = Closed_Account ~ Income * Gender, family = binomial(link = "logit"),
##     data = cleaned_bank_accounts_train)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.548e+00  7.877e-02 -19.646  < 2e-16 ***
## Income         -9.346e-05  7.426e-04  -0.126  0.89985
## GenderM        -3.990e-01  1.446e-01  -2.760  0.00577 **
## Income:GenderM  1.896e-03  1.402e-03   1.353  0.17606
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6684.6  on 7594  degrees of freedom
## Residual deviance: 6669.7  on 7591  degrees of freedom
## AIC: 6677.7
##
## Number of Fisher Scoring iterations: 4
```

```r
# Plot the logistic regression model
income_range <- with(cleaned_bank_accounts_train,
                     seq(min(Income, na.rm = TRUE),
                         max(Income, na.rm = TRUE),
                         length.out = 100))
gender_levels <- levels(fact_bank_accounts_train$Gender)
newdata <- expand.grid(Income = income_range, Gender = gender_levels)
newdata$Probability <- predict(logit_model, newdata = newdata, type = "response")

ggplot(newdata, aes(x = Income, y = Probability, color = Gender)) +
  geom_line() +
  labs(title = "Effect of Income and Gender on the Probability of Closing an Account",
       x = "Income",
       y = "Probability of Account Closure",
       color = "Gender") +
  scale_y_continuous(labels = scales::percent_format()) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text.x = element_text(angle = 45, hjust = 1),
    axis.title.x = element_text(face = "bold"),
    axis.title.y = element_text(face = "bold"),
  )
```



KNN

The graphs illustrate the performance of the **k-NN model** based on the *misclassification error* and the *Area Under the Curve (AUC)* for both the training and validation sets. The plots show the misclassification error and AUC values for different values of k. The best value of k is determined based on the minimum misclassification error and the maximum AUC on the validation set.

The best one based on the misclassification error is calculated by finding the k value that minimizes the misclassification error (7) on the validation set, while similarly, the best one based on the AUC is determined by finding the k value that maximizes the AUC on the validation set (20).

By applying the *elbow rule*, we observe that past these optimal values, additional increases in k do not substantially enhance the model's discriminative power. This ensures that the identified values of k best encapsulate the trade-off between model complexity and generalizability.

DATA PREPARATION

```
set.seed(42)

knn_cleaned_bank_accounts <- encoded_bank_accounts_train %>% select(Total_Trans_Amt, Total_Trans_Ct, Cl
```

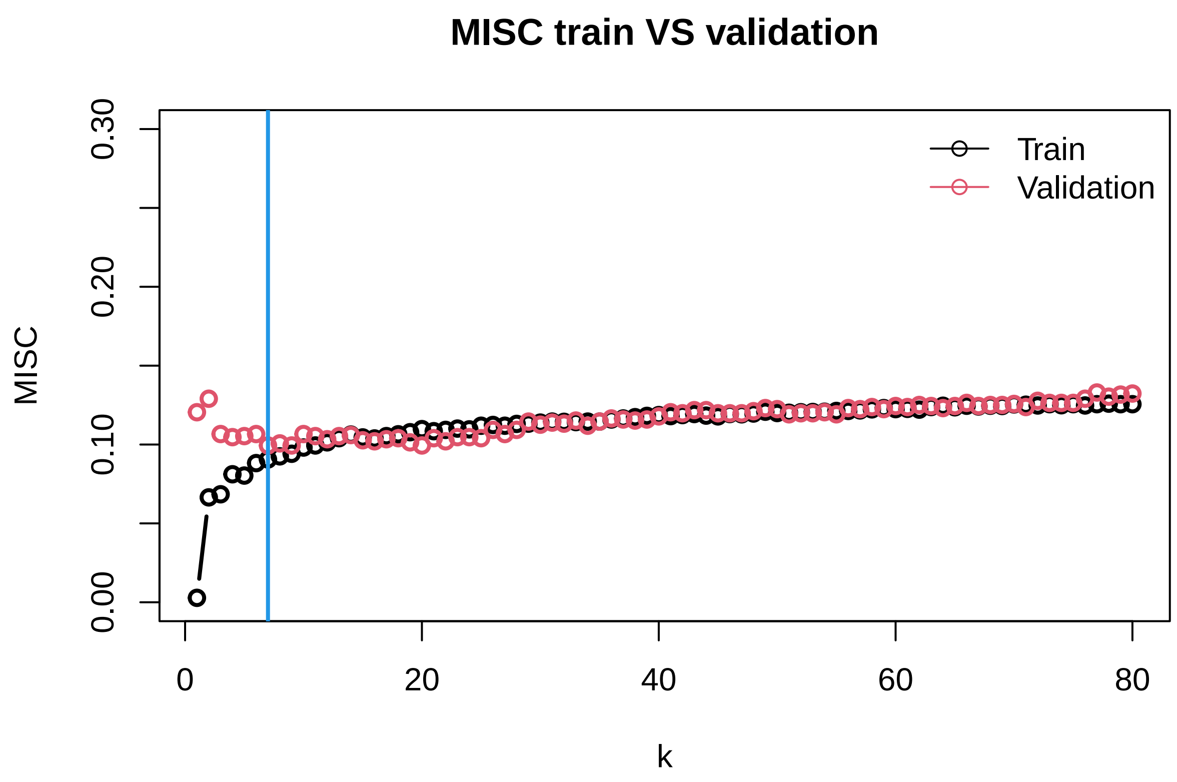
```r
p = 0.8
division <- createDataPartition(knn_cleaned_bank_accounts$Closed_Account, times = 1, p = p, list = FALS

# New sets of covariates and response variables
knn_train <- knn_cleaned_bank_accounts[division,]
knn_val <- knn_cleaned_bank_accounts[-division,]

x_train <- knn_train %>% select(Total_Trans_Amt, Total_Trans_Ct)
y_train <- knn_train$Closed_Account

x_val <- knn_val %>% select(Total_Trans_Amt, Total_Trans_Ct)
y_val <- knn_val$Closed_Account

# Standardization of training set
x_train_sc <- scale(x_train)

# Standardization of validation set
train_m <- apply(x_train, MARGIN = 2, FUN = mean)
train_s <- apply(x_train, MARGIN = 2, FUN = sd)
x_val_sc <- scale(x_val, center = train_m, scale = train_s)
```

IMPLEMENTATION OF K-NN MODEL

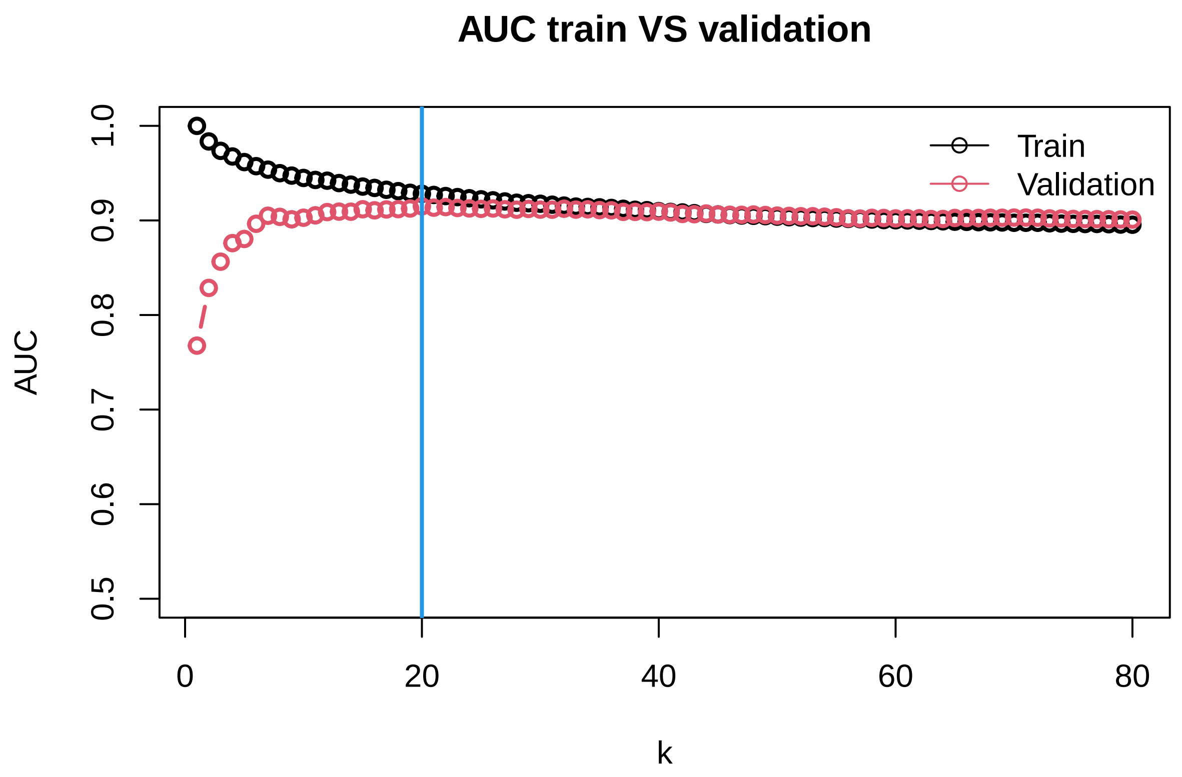
```r
k_grid <- 1:80

# Model fitting (MISC/AUC calculation) for each value of k
k_results <- sapply(k_grid, FUN = function (kk) {
  # Model fitting
  kk_fit_train <- knn(train = x_train, test = x_train, cl = y_train, k = kk, prob = TRUE)
  kk_fit_val <- knn(train = x_train, test = x_val, cl = y_train, k = kk, prob = TRUE)

  # Misclassification error
  kk_misc_in <- 1 - mean(y_train == kk_fit_train)
  kk_misc_out <- 1 - mean(y_val == kk_fit_val)

  # AUC
  kk_probyes_in <- ifelse(kk_fit_train == 1, attr(kk_fit_train, "prob"), 1 - attr(kk_fit_train, "prob")
  kk_probyes_out <- ifelse(kk_fit_val == 1, attr(kk_fit_val, "prob"), 1 - attr(kk_fit_val, "prob"))
  kk_auc_in <- pROC::auc(y_train == 1, kk_probyes_in)
  kk_auc_out <- pROC::auc(y_val == 1, kk_probyes_out)

  # Results
  return(c(kk_misc_in, kk_misc_out, kk_auc_in, kk_auc_out))
})
```

EVALUATION

```r
# Evaluation metrics on the grid
misc_in <- k_results[1, ]
misc_out <- k_results[2, ]
auc_in <- k_results[3, ]
auc_out <- k_results[4, ]

# Plot the performance (MISC)
plot(k_grid, misc_in, type = "b", lwd = 2, ylim = c(0, 0.3), main = "MISC train VS validation", ylab =
lines(k_grid, misc_out, type = "b", lwd = 2, col = 2, lty = 1)
legend("topright", c("Train", "Validation"), col = c(1, 2), lty = c(1), bty = "n", pch = 21)

# Best k using MISC
k_best_misc <- k_grid[which.min(misc_out)]
abline(v = k_best_misc, col = 4, lwd = 2)
```



```r
best_misc <- min(misc_out)

# Plot the performance (AUC)
plot(k_grid, auc_in, type = "b", lwd = 2, ylim = c(0.5, 1), main = "AUC train VS validation", ylab = "A
lines(k_grid, auc_out, type = "b", lwd = 2, col = 2, lty = 1)
legend("topright", c("Train", "Validation"), col = c(1,2), lty = c(1), bty = "n", pch = 21)

# Best k using AUC
k_best_auc <- k_grid[which.max(auc_out)]
abline(v = k_best_auc, col = 4, lwd = 2)
```



```
best_auc <- max(auc_out)

k_best_auc
```

```
## [1] 20
```

```
k_best_misc
```

```
## [1] 7
```

FINDING THE BEST MODELS

We focused on the predictive modeling of closed bank accounts using logistic regression, appropriate for binary outcomes. The initial phase revolves around *data preprocessing*, where irrelevant or non-predictive variables (like `CLIENTNUM`, `Education_Level`, and `Marital_Status`) get removed to prepare the dataset.

In the next steps data is divided into *training* and *validation* sets, hightlighting the importance of cross-validation in assessing model applicability to data. Using various logistic regression models, each employing different variable selection techniques (forward, backward, and bidirectional), implies the necessary balance between *simplicity* and *precision in forecasts*. This takes us to choosing a model that lets us avoid excess complexity and is accurate enough when predicting outcomes. For this purpose, the *Akaike Information Criterion (AIC)* and the *Bayesian Information Criterion (BIC)* are leveraged to select the best model based on the lowest value of the respective criterion. The selected model is then evaluated using the *Area Under the Curve (AUC)* metric, providing a comprehensive measure of model performance across different classification thresholds and considering both *sensitivity* and *specificity* in binary classification, especially important when the costs of false positives (wrong prediction of account closure) and false negatives (failed prediction of account closure) are varying so much.

In the end, we then select the optimal model based on the *highest AUC value*, applying it to the test dataset to generate predictive probabilities of account closure, generating the predictive probabilities csv file: it is the `aic1 with AUC of 0.9161982`.

```
factorized_bank_accounts = fact_bank_accounts_train %>% select(-c(CLIENTNUM, Education_Level, Marital_S

# New sets of covariates and response variables
model_train <- factorized_bank_accounts[division,]
model_val <- factorized_bank_accounts[-division,]

x_train <- model_train %>% select(-Closed_Account)
y_train <- model_train$Closed_Account

x_val <- model_val %>% select(-Closed_Account)
y_val <- model_val$Closed_Account
```

```
# MULTIPLE LOGISTIC REGRESSION

# Full analysis using all the predictors.

logit_fit1 <- glm(Closed_Account ~ .,
                  family = "binomial",
                  data = model_train)
#summary(logit_fit1)


# Stepwise variable selection (based on AIC):

# Forward
logit_fit_aic1 <- step(glm(Closed_Account ~ 1,
                           family = "binomial",
                           data = model_train),
                       scope = formula(logit_fit1),
                       direction = "forward")

# Backward
logit_fit_aic2 <- step(logit_fit1,
                       direction = "backward")

# Both directions
logit_fit_aic3 <- step(logit_fit1,
                       direction = "both")

logit_fit_aic1$aic
logit_fit_aic2$aic
logit_fit_aic3$aic
```

```r
names(model_train)
names(logit_fit_aic1$coefficients)


# Stepwise variable selection (based on BIC):

# Forward
logit_fit_bic1 <- step(glm(Closed_Account ~ 1,
                           family = "binomial",
                           data = model_train),
                       scope = formula(logit_fit1),
                       direction = "forward",
                       k = log(nrow(model_train)))

# Backward
logit_fit_bic2 <- step(logit_fit1,
                       direction = "backward",
                       k = log(nrow(model_train)))

# Both directions
logit_fit_bic3 <- step(logit_fit1,
                       direction = "both",
                       k = log(nrow(model_train)))

logit_fit_bic1$aic
logit_fit_bic2$aic
logit_fit_bic3$aic

names(model_train)
names(logit_fit_bic1$coefficients)

setdiff(names(logit_fit_aic1$coefficients),names(logit_fit_aic2$coefficients))
setdiff(names(logit_fit_aic2$coefficients),names(logit_fit_aic3$coefficients))
setdiff(names(logit_fit_aic3$coefficients),names(logit_fit_aic1$coefficients))

best_model = logit_fit_aic1
best_model2 = logit_fit_bic1



# Function to calculate AUC for a given model and validation data
calculate_auc <- function(model, x_val, y_val) {
  probabilities <- predict(model, newdata = x_val, type = "response")
  roc_obj <- roc(y_val, probabilities)
  return(auc(roc_obj))
}

# Calculate AUC for each model
auc_aic1 <- calculate_auc(logit_fit_aic1, model_val, y_val)
```

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

## Setting direction: controls < cases
```

```r
auc_aic2 <- calculate_auc(logit_fit_aic2, model_val, y_val)
```

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

```r
auc_aic3 <- calculate_auc(logit_fit_aic3, model_val, y_val)
```

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

```r
auc_bic1 <- calculate_auc(logit_fit_bic1, model_val, y_val)
```

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

```r
auc_bic2 <- calculate_auc(logit_fit_bic2, model_val, y_val)
```

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

```r
auc_bic3 <- calculate_auc(logit_fit_bic3, model_val, y_val)
```

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

```r
# Print AUC values for comparison
cat("AUC AIC1:", auc_aic1, "\n",
    "AUC AIC2:", auc_aic2, "\n",
    "AUC AIC3:", auc_aic3, "\n",
    "AUC BIC1:", auc_bic1, "\n",
    "AUC BIC2:", auc_bic2, "\n",
    "AUC BIC3:", auc_bic3, "\n")

# Higher AUC is better, selecting the model with the highest AUC
auc_values <- c(auc_aic1, auc_aic2, auc_aic3, auc_bic1, auc_bic2, auc_bic3)
names(auc_values) <- c("aic1", "aic2", "aic3", "bic1", "bic2", "bic3")

best_model_name <- names(which.max(auc_values))
best_model <- get(paste("logit_fit", best_model_name, sep = "_"))

cat("Best model is:", best_model_name, "with AUC of", max(auc_values), "\n")


# Predict probabilities using the best model
predicted_probabilities <- predict(best_model, newdata = cleaned_bank_accounts_test, type = "response")

#Print or save the predicted probabilities
#print(predicted_probabilities)
write.csv(data.frame(CLIENTNUM = bank_accounts_test$CLIENTNUM, PredictedProbability = predicted_probabil
```

THRESHOLD FOR GAINS

THRESHOLD AT 0.5

```r
tt <- 0.5

# Prediction with threshold 0.5
prob_out_best <- predict(best_model, newdata = model_val, type = "response")
pred_out_best_binary <- ifelse(prob_out_best > tt, 1, 0)

# Compute Confusion Matrix with '1' as the positive class
CM <- table(Predicted = pred_out_best_binary, Actual = model_val$Closed_Account)

# Adjusted definitions based on '1' being the positive class
TP <- CM["1", "1"]  # True Positives
TN <- CM["0", "0"]  # True Negatives
FP <- CM["1", "0"]  # False Positives
FN <- CM["0", "1"]  # False Negatives

# Output the confusion matrix
print(CM)
```

```
##          Actual
## Predicted    0    1
##         0 1236  120
##         1   35  128
```

```r
# Financial outcome and metrics calculation can be adjusted as per these definitions
# Example:
gain_TP <- TP * 20  # Adjust gain from True Positives as per your model specifics
gain_TN <- TN * 50  # Adjust gain from True Negatives
loss_FP <- FP * 20  # Adjust loss from False Positives
loss_FN <- FN * (-50)  # Adjust loss from False Negatives

total_financial_outcome <- gain_TP + gain_TN + loss_FP + loss_FN

# Relevant Metrics
accuracy <- (TP + TN) / sum(CM)
precision <- TP / (TP + FP)
recall <- TP / (TP + FN)  # Also known as sensitivity
F1_score <- 2 * (precision * recall) / (precision + recall)

# Output the metrics
cat("Treshold: ", tt, "\n",
    "Financial Gain: ", total_financial_outcome, "\n",
    "Accuracy: ", accuracy, "\n",
    "Precision: ", precision, "\n",
    "Recall (Sensitivity): ", recall, "\n",
    "F1 Score: ", F1_score, "\n")
```

```
## Treshold:  0.5
##  Financial Gain:  59060
##  Accuracy:  0.8979592
##  Precision:  0.7852761
##  Recall (Sensitivity):  0.516129
##  F1 Score:  0.622871
```

```
# Calculate the ROC curve
roc_obj <- roc(response = model_val$Closed_Account, predictor = prob_out_best, levels = c("0", "1"))
```

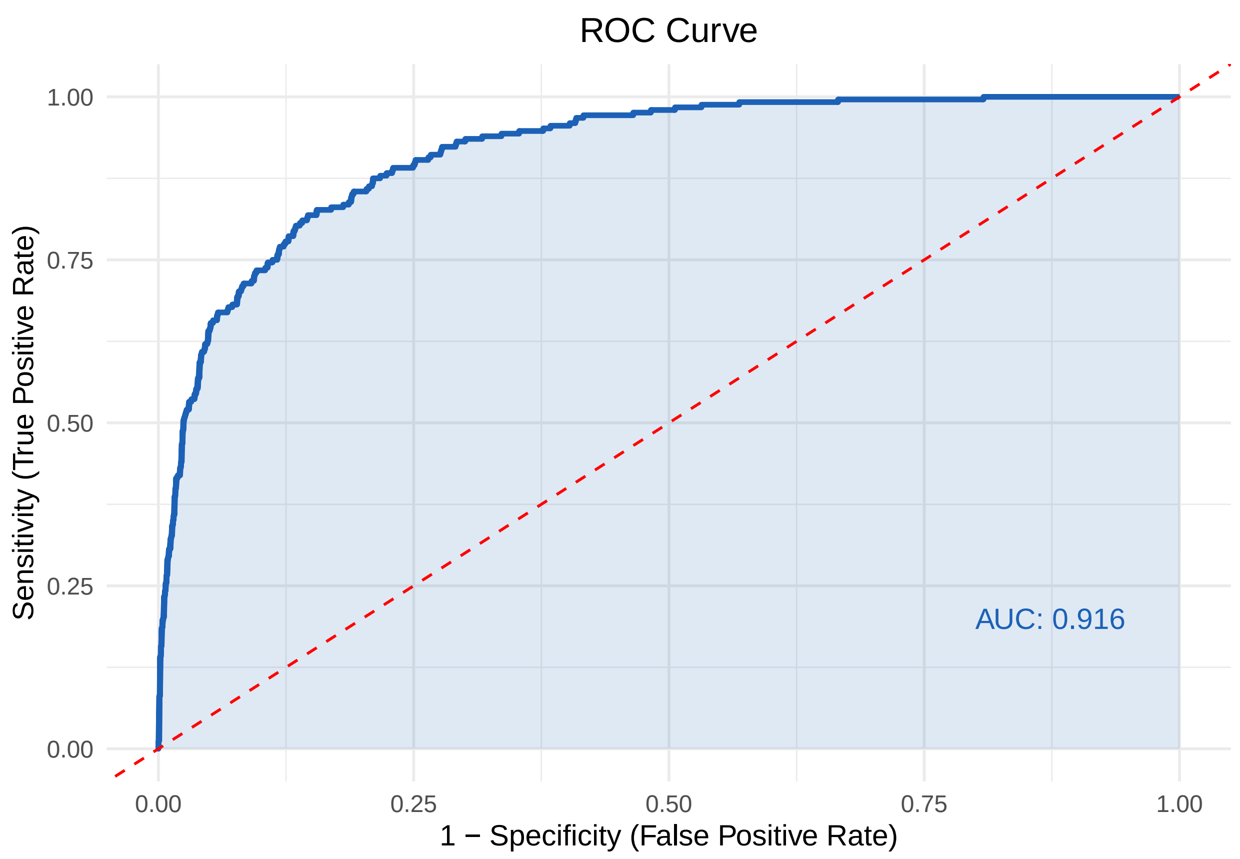
```
## Setting direction: controls < cases
```

```
# Print AUC
auc_value2 <- auc(roc_obj)
print(paste("AUC:", auc_value2))
```

```
## [1] "AUC: 0.916198192939266"
```

```
# Create a data frame from the roc object
roc_data <- data.frame(
  TPR = roc_obj$sensitivities,
  FPR = roc_obj$specificities,
  Thresholds = roc_obj$thresholds
)

# Plot the ROC curve
ggplot(roc_data, aes(x = 1 - FPR, y = TPR)) +
  geom_line(color = "#1c61b6", size = 1) +
  geom_area(fill = "#1c61b624") +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed", color = "red") +
  annotate("text", x = 0.8, y = 0.2, label = paste("AUC:", round(auc(roc_obj), 3)), hjust = 0, color =
  theme_minimal() +
  labs(
    title = "ROC Curve",
    x = "1 - Specificity (False Positive Rate)",
    y = "Sensitivity (True Positive Rate)"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5),
    legend.position = "none"
  )
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



THRESHOLD AT 0.2

```r
tt <- 0.2

# Prediction with threshold 0.2
prob_out_best <- predict(best_model, newdata = model_val, type = "response")
pred_out_best_binary <- ifelse(prob_out_best > tt, 1, 0)

# Compute Confusion Matrix with '1' as the positive class
CM <- table(Predicted = pred_out_best_binary, Actual = model_val$Closed_Account)

# Adjusted definitions based on '1' being the positive class
TP <- CM["1", "1"]  # True Positives
TN <- CM["0", "0"]  # True Negatives
FP <- CM["1", "0"]  # False Positives
FN <- CM["0", "1"]  # False Negatives

# Output the confusion matrix
print(CM)
```

```
##          Actual
## Predicted    0    1
##         0 1111   55
##         1  160  193
```

```
# Financial outcome and metrics calculation can be adjusted as per these definitions
# Example:
gain_TP <- TP * 20  # Adjust gain from True Positives as per your model specifics
gain_TN <- TN * 50  # Adjust gain from True Negatives
loss_FP <- FP * 20  # Adjust loss from False Positives
loss_FN <- FN * (-50)  # Adjust loss from False Negatives

total_financial_outcome <- gain_TP + gain_TN + loss_FP + loss_FN

# Relevant Metrics
accuracy <- (TP + TN) / sum(CM)
precision <- TP / (TP + FP)
recall <- TP / (TP + FN)  # Also known as sensitivity
F1_score <- 2 * (precision * recall) / (precision + recall)

# Output the metrics
cat("Treshold: ", tt, "\n",
    "Financial Gain: ", total_financial_outcome, "\n",
    "Accuracy: ", accuracy, "\n",
    "Precision: ", precision, "\n",
    "Recall (Sensitivity): ", recall, "\n",
    "F1 Score: ", F1_score, "\n")
```

```
## Treshold:  0.2
##  Financial Gain:  59860
##  Accuracy:  0.8584595
##  Precision:  0.5467422
##  Recall (Sensitivity):  0.7782258
##  F1 Score:  0.6422629
```

THRESHOLD AT 0.8

```
tt <- 0.8

# Prediction with threshold 0.8
prob_out_best <- predict(best_model, newdata = model_val, type = "response")
pred_out_best_binary <- ifelse(prob_out_best > tt, 1, 0)

# Compute Confusion Matrix with '1' as the positive class
CM <- table(Predicted = pred_out_best_binary, Actual = model_val$Closed_Account)

# Adjusted definitions based on '1' being the positive class
TP <- CM["1", "1"]  # True Positives
TN <- CM["0", "0"]  # True Negatives
FP <- CM["1", "0"]  # False Positives
FN <- CM["0", "1"]  # False Negatives

# Output the confusion matrix
print(CM)
```

```
##          Actual
## Predicted    0    1
##         0 1262  187
##         1    9   61
```

```
# Financial outcome and metrics calculation can be adjusted as per these definitions
# Example:
gain_TP <- TP * 20  # Adjust gain from True Positives as per your model specifics
gain_TN <- TN * 50  # Adjust gain from True Negatives
loss_FP <- FP * 20  # Adjust loss from False Positives
loss_FN <- FN * (-50)  # Adjust loss from False Negatives

total_financial_outcome <- gain_TP + gain_TN + loss_FP + loss_FN

# Relevant Metrics
accuracy <- (TP + TN) / sum(CM)
precision <- TP / (TP + FP)
recall <- TP / (TP + FN)  # Also known as sensitivity
F1_score <- 2 * (precision * recall) / (precision + recall)

# Output the metrics
cat("Treshold: ", tt, "\n",
    "Financial Gain: ", total_financial_outcome, "\n",
    "Accuracy: ", accuracy, "\n",
    "Precision: ", precision, "\n",
    "Recall (Sensitivity): ", recall, "\n",
    "F1 Score: ", F1_score, "\n")
```

```
## Treshold:  0.8
##  Financial Gain:  55150
##  Accuracy:  0.8709677
##  Precision:  0.8714286
##  Recall (Sensitivity):  0.2459677
##  F1 Score:  0.3836478
```

THRESHOLD AT 0.28

```
tt <- 0.28

# Prediction with threshold 0.28
prob_out_best <- predict(best_model, newdata = model_val, type = "response")
pred_out_best_binary <- ifelse(prob_out_best > tt, 1, 0)

# Compute Confusion Matrix with '1' as the positive class
CM <- table(Predicted = pred_out_best_binary, Actual = model_val$Closed_Account)

# Adjusted definitions based on '1' being the positive class
TP <- CM["1", "1"]  # True Positives
TN <- CM["0", "0"]  # True Negatives
FP <- CM["1", "0"]  # False Positives
FN <- CM["0", "1"]  # False Negatives
```

```r
# Output the confusion matrix
print(CM)
```

```
##          Actual
## Predicted    0    1
##         0 1167   72
##         1  104  176
```

```r
# Financial outcome and metrics calculation can be adjusted as per these definitions
# Example:
gain_TP <- TP * 20  # Adjust gain from True Positives as per your model specifics
gain_TN <- TN * 50  # Adjust gain from True Negatives
loss_FP <- FP * 20  # Adjust loss from False Positives
loss_FN <- FN * (-50)  # Adjust loss from False Negatives

total_financial_outcome <- gain_TP + gain_TN + loss_FP + loss_FN

# Relevant Metrics
accuracy <- (TP + TN) / sum(CM)
precision <- TP / (TP + FP)
recall <- TP / (TP + FN)  # Also known as sensitivity
F1_score <- 2 * (precision * recall) / (precision + recall)

# Output the metrics
cat("Treshold: ", tt, "\n",
    "Financial Gain: ", total_financial_outcome, "\n",
    "Accuracy: ", accuracy, "\n",
    "Precision: ", precision, "\n",
    "Recall (Sensitivity): ", recall, "\n",
    "F1 Score: ", F1_score, "\n")
```

```
## Treshold:  0.28
##  Financial Gain:  60350
##  Accuracy:  0.8841343
##  Precision:  0.6285714
##  Recall (Sensitivity):  0.7096774
##  F1 Score:  0.6666667
```

In the context of an imbalanced dataset where the class of interest (closed accounts) is the minority class, optimizing for sensitivity is crucial. Sensitivity, also known as the true positive rate or recall, measures the proportion of actual positives that are correctly identified by the model (the accounts that were correctly predicted to close). This focus is essential because missing out on identifying an account that will close (a false negative) is costlier than incorrectly predicting that an account will close (a false positive). High sensitivity ensures the bank can proactively retain these customers, avoiding a potential loss for each customer who leaves.

• Threshold of 0.5 yields decent accuracy but falls short in maximizing financial gain due to moderate sensitivity. • Threshold of 0.2 boosts sensitivity, catching more customers who might leave, but also increases false positives, where offers are made to customers who would stay regardless. • Threshold of 0.8 sharply decreases sensitivity, reducing potential gains significantly. • Threshold of 0.28 strikes the best balance, leading to the highest financial gain by achieving a good balance between sensitivity and precision.

The ROC curve and the AUC value of 0.916 confirm the model's strong discriminative ability. The curve well above the baseline indicates a good separation between the true positive and false positive rates across various thresholds, affirming the reliability of the model in differentiating between the customers who will close their accounts and those who will not.

In conclusion, the model performs well across all thresholds, but the best financial outcome is achieved at a threshold of 0.28, where the balance between sensitivity and the financial implications of intervention (incentives offered versus customer retention) is optimal.