BANK CHURN ANALYSIS

Ezo YEDİGÖL

2025-06-21

Business Task Analyzing the customer dataset provided by ABC Multinational Bank to highlight any trends in the data and provide insights.

About Dataset We have been provided a dataset in .csv(Comma Separated Value) format, containing 10,000 customers from the bank and each customer has the following attributes;

Customer ID - The Unique ID of each individual customer Credit Score - A number depicting the customer's credithworthiness Country - The country the customer banks from Gender - The gender the customer identifies with Age - Depicts the customers age Tenure - Indicates how length in years the customer has been with the bank Balance - The amount currently available in the customer's account Products Number - The number of products purchased by the customer through the bank Credit Card - Indicates the customer has a credit card Active Member - Indicates if the customer is an active or inactive Estimated Salary - Bank Estimation of the income of the customer Churn - Indicator of if the customer has left the bank or not

STEP-1: LOADING AND CLEANING THE DATA

```
#Loading Starting Packages
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(ggplot2)
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3

## corrplot 0.92 loaded
```

```
library(readr)
library(tidyverse)   # For data manipulation and visualization
```

```
## Warning: package 'stringr' was built under R version 4.3.3
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(skimr)       # For summary statistics
```

```
## Warning: package 'skimr' was built under R version 4.3.3
```

```r
library(janitor)     # For cleaning column names
```

```
## Warning: package 'janitor' was built under R version 4.3.3

##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```r
# Load the dataset
bank <- read_csv("C:/RProjects/data/bank_churn.csv")
```

```
## Rows: 10000 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): country, gender
## dbl (10): customer_id, credit_score, age, tenure, balance, products_number, ...
##
## 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.
```

STEP-2: PREPARING THE DATA

```r
head(bank)
```

```
## # A tibble: 6 x 12
##   customer_id credit_score country gender   age tenure balance products_number
##         <dbl>        <dbl> <chr>   <chr>  <dbl>  <dbl>   <dbl>           <dbl>
## 1    15634602          619 France  Female    42      2       0               1
## 2    15647311          608 Spain   Female    41      1  83808.               1
## 3    15619304          502 France  Female    42      8 159661.               3
## 4    15701354          699 France  Female    39      1       0               2
## 5    15737888          850 Spain   Female    43      2 125511.               1
## 6    15574012          645 Spain   Male      44      8 113756.               2
## # i 4 more variables: credit_card <dbl>, active_member <dbl>,
## #   estimated_salary <dbl>, churn <dbl>
```

- The dataset contains information about bank customers and whether they have churned (i.e., left the bank).
- Each row represents a unique customer and includes demographic features (such as age, gender, and country), financial information (credit score, balance, estimated salary), behavioral data (tenure, number of products, whether they have a credit card or are active members), and a target variable churn indicating whether the customer has exited the bank.

For example: - The first customer is a 42-year-old female from France with a credit score of 619, a balance of 0, and has been a customer for 2 years. - The third customer has a high balance and uses 3 products, indicating potentially high value.

These initial rows help us get a basic understanding of the dataset before conducting deeper exploratory data analysis (EDA).

```r
# Check the structure of the dataset
glimpse(bank)
```

```
## Rows: 10,000
## Columns: 12
## $ customer_id      <dbl> 15634602, 15647311, 15619304, 15701354, 15737888, 155~
## $ credit_score     <dbl> 619, 608, 502, 699, 850, 645, 822, 376, 501, 684, 528~
## $ country          <chr> "France", "Spain", "France", "France", "Spain", "Spai~
## $ gender           <chr> "Female", "Female", "Female", "Female", "Female", "Ma~
## $ age              <dbl> 42, 41, 42, 39, 43, 44, 50, 29, 44, 27, 31, 24, 34, 2~
## $ tenure           <dbl> 2, 1, 8, 1, 2, 8, 7, 4, 4, 2, 6, 3, 10, 5, 7, 3, 1, 9~
## $ balance          <dbl> 0.00, 83807.86, 159660.80, 0.00, 125510.82, 113755.78~
## $ products_number  <dbl> 1, 1, 3, 2, 1, 2, 2, 4, 2, 1, 2, 2, 2, 2, 2, 2, 1, 2,~
## $ credit_card      <dbl> 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1,~
## $ active_member    <dbl> 1, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1,~
## $ estimated_salary <dbl> 101348.88, 112542.58, 113931.57, 93826.63, 79084.10, ~
## $ churn            <dbl> 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,~
```

```r
summary(bank)
```

```
##   customer_id        credit_score     country             gender
##  Min.   :15565701   Min.   :350.0   Length:10000       Length:10000
##  1st Qu.:15628528   1st Qu.:584.0   Class :character   Class :character
##  Median :15690738   Median :652.0   Mode  :character   Mode  :character
##  Mean   :15690941   Mean   :650.5
##  3rd Qu.:15753234   3rd Qu.:718.0
##  Max.   :15815690   Max.   :850.0
##       age            tenure          balance       products_number
##  Min.   :18.00   Min.   : 0.000   Min.   :     0   Min.   :1.00
##  1st Qu.:32.00   1st Qu.: 3.000   1st Qu.:     0   1st Qu.:1.00
##  Median :37.00   Median : 5.000   Median : 97199   Median :1.00
##  Mean   :38.92   Mean   : 5.013   Mean   : 76486   Mean   :1.53
##  3rd Qu.:44.00   3rd Qu.: 7.000   3rd Qu.:127644   3rd Qu.:2.00
##  Max.   :92.00   Max.   :10.000   Max.   :250898   Max.   :4.00
##   credit_card     active_member    estimated_salary        churn
##  Min.   :0.0000   Min.   :0.0000   Min.   :    11.58   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 51002.11   1st Qu.:0.0000
##  Median :1.0000   Median :1.0000   Median :100193.91   Median :0.0000
##  Mean   :0.7055   Mean   :0.5151   Mean   :100090.24   Mean   :0.2037
```

```
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:149388.25   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :199992.48   Max.   :1.0000
```

- Median tenure of clients is about 5 years.
- Median age is about 37 years old (with an oldest [max] at 92… impressive!).
- Most clients (according to the median) only use 1 of the bank's products.
- Since the average churn is ~ 0.20, I can already infer this bank is keeping more clients than churning. I'll analyze the churn metric more a little later.

```r
# Check for missing values
colSums(is.na(bank))
```

```
##      customer_id     credit_score          country           gender
##                0                0                0                0
##              age           tenure          balance  products_number
##                0                0                0                0
##      credit_card    active_member estimated_salary            churn
##                0                0                0                0
```

- No Missing Data: All variables have 0 missing values (n_missing = 0), which is great for analysis.

Numeric Variables: - customer_id: Just an ID number. - credit_score: Average is around 650. - age: Average age is about 39. - tenure: Customers stay with the bank for about 5 years on average. - balance: Account balances vary a lot; many customers have 0 balance. - products_number: Most customers have 1 or 2 bank products. - credit_card and active_member: These are 0/1 (binary) variables, showing about 71% have a credit card and 52% are active members. - estimated_salary: Average estimated salary is around 100,000. - churn: This is our target variable. About 20.37% of customers have churned (left the bank). This shows an imbalance, meaning fewer people churned than stayed.

Categorical Variables: - country: There are 3 unique countries (e.g., France, Spain, Germany). - gender: There are 2 unique genders (Male, Female).

In short, the data is clean (no missing values) and ready for further analysis, but we need to pay attention to the churn imbalance during modeling.

STEP-3:EXPLORATORY DATA ANALYSIS

```r
# Calculate churn rate
churn_rate <- mean(bank$churn)
print(paste("Customer Churn Rate:", round(churn_rate * 100, 2), "%"))
```

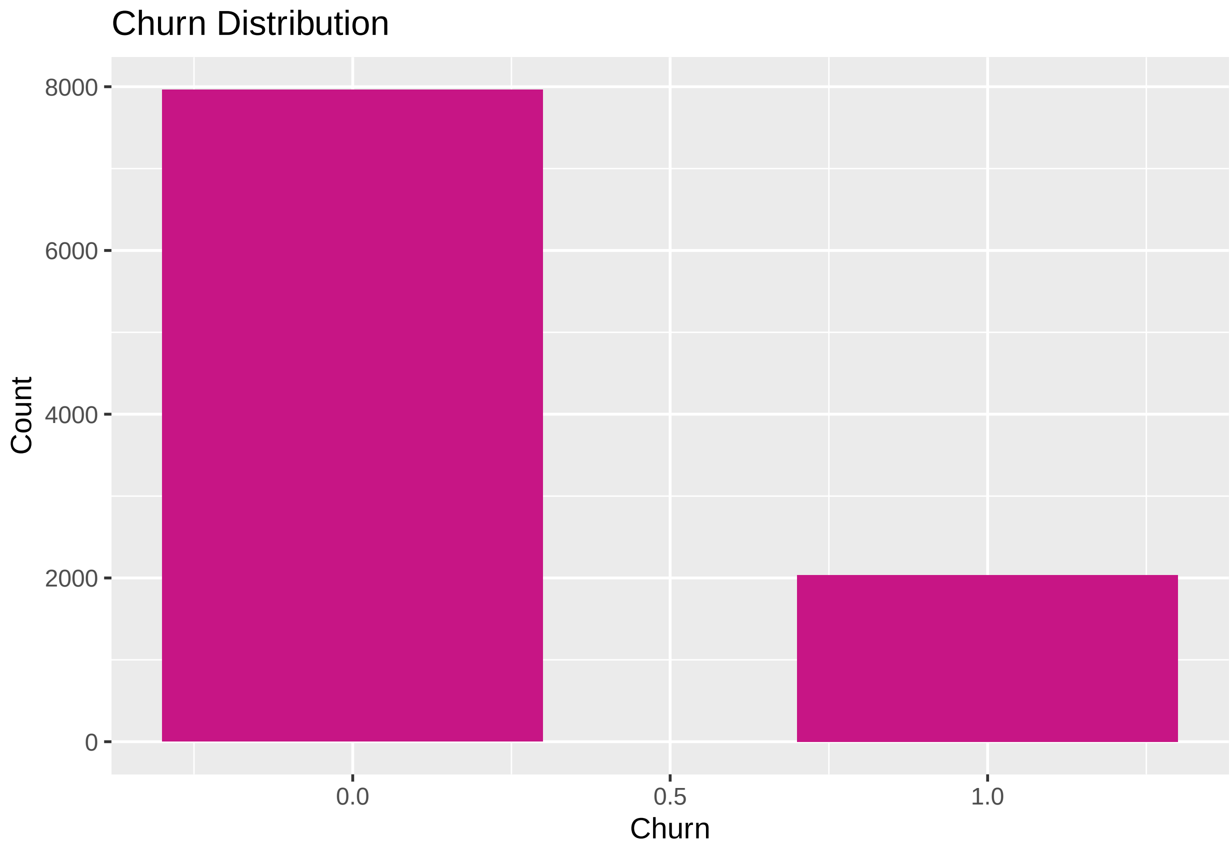
```
## [1] "Customer Churn Rate: 20.37 %"
```

```r
# Alternatively, to see the counts of churned vs. non-churned:
table(bank$churn)
```

```
##
##    0    1
## 7963 2037
```

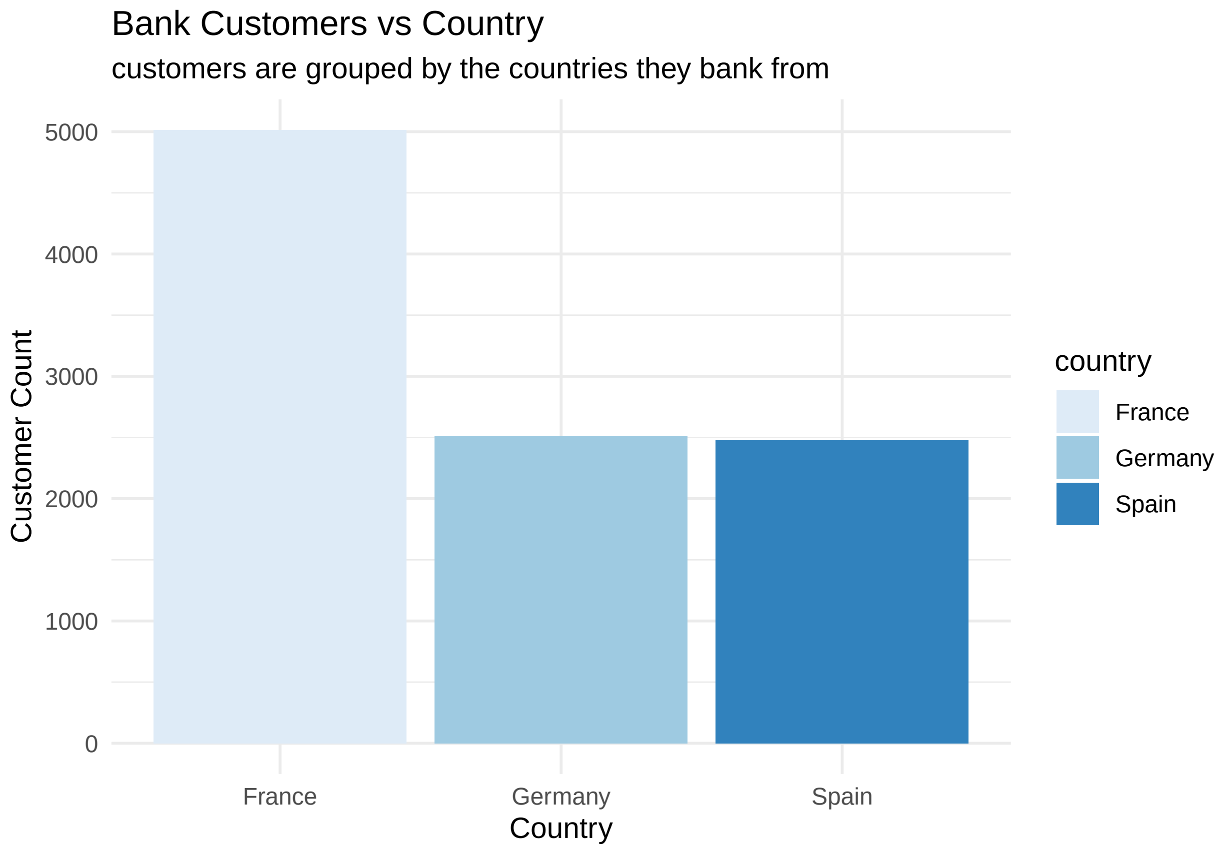
- The first line tells us that 20.37% of the customers have churned (left the bank).
- The table(df$churn) output shows the exact numbers: 0: 7963 customers did NOT churn (they stayed). 1: 2037 customers DID churn (they left). This means that a significant portion of customers are leaving, and our dataset has more customers who stayed than who churned.

```r
#Exploratory Analysis: Most Customers do not Churn (0)
ggplot(data = bank, aes(x = churn)) +
  geom_bar(width = 0.6, fill = "mediumvioletred") +
  labs(title = "Churn Distribution", x = "Churn", y = "Count")
```



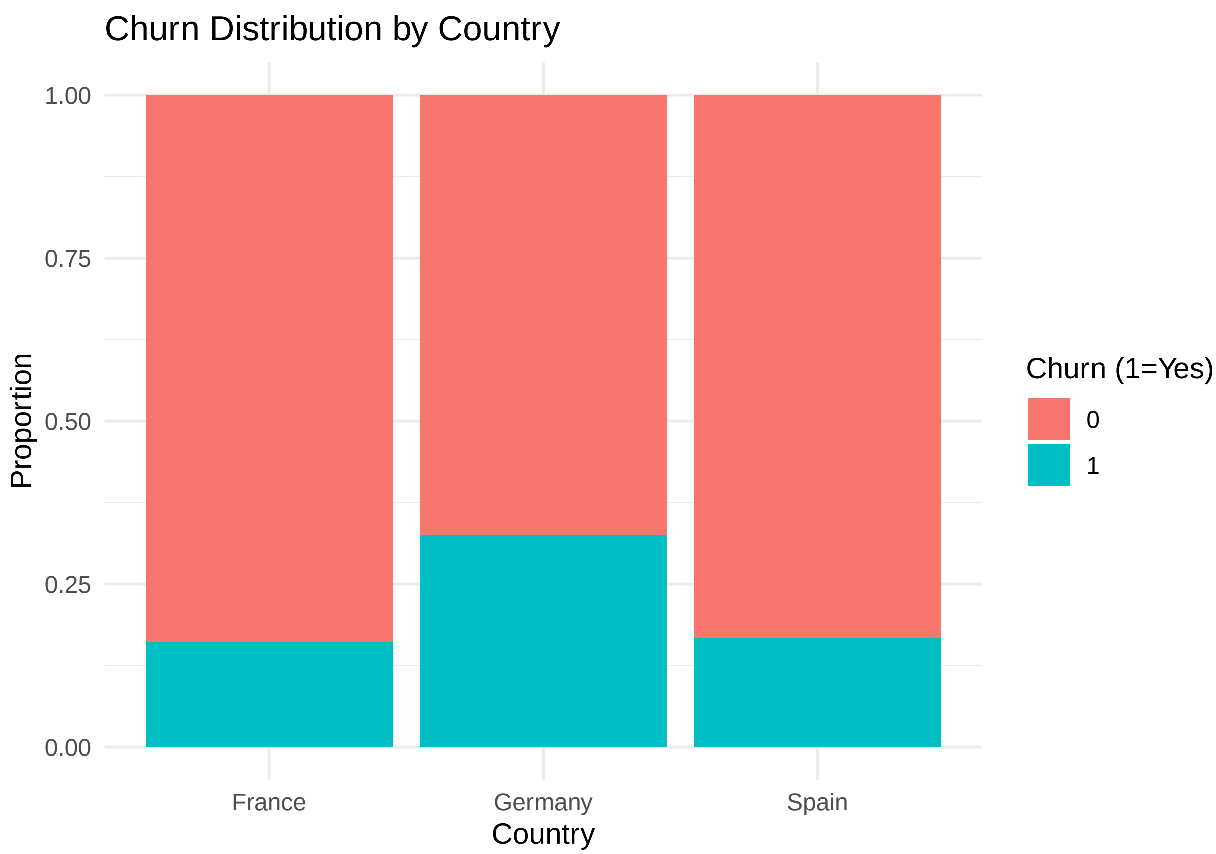
- About 80% of the client base has not churned. These may be cause some inbalance problem in our project that we need to fix in the coming up.

```r
ggplot(bank, aes(x = country, fill = country)) + geom_bar() +
scale_fill_brewer(palette = "Blues") +
labs(title = "Bank Customers vs Country", x = "Country", y = "Customer Count", subtitle = "customers are
theme_minimal()
```

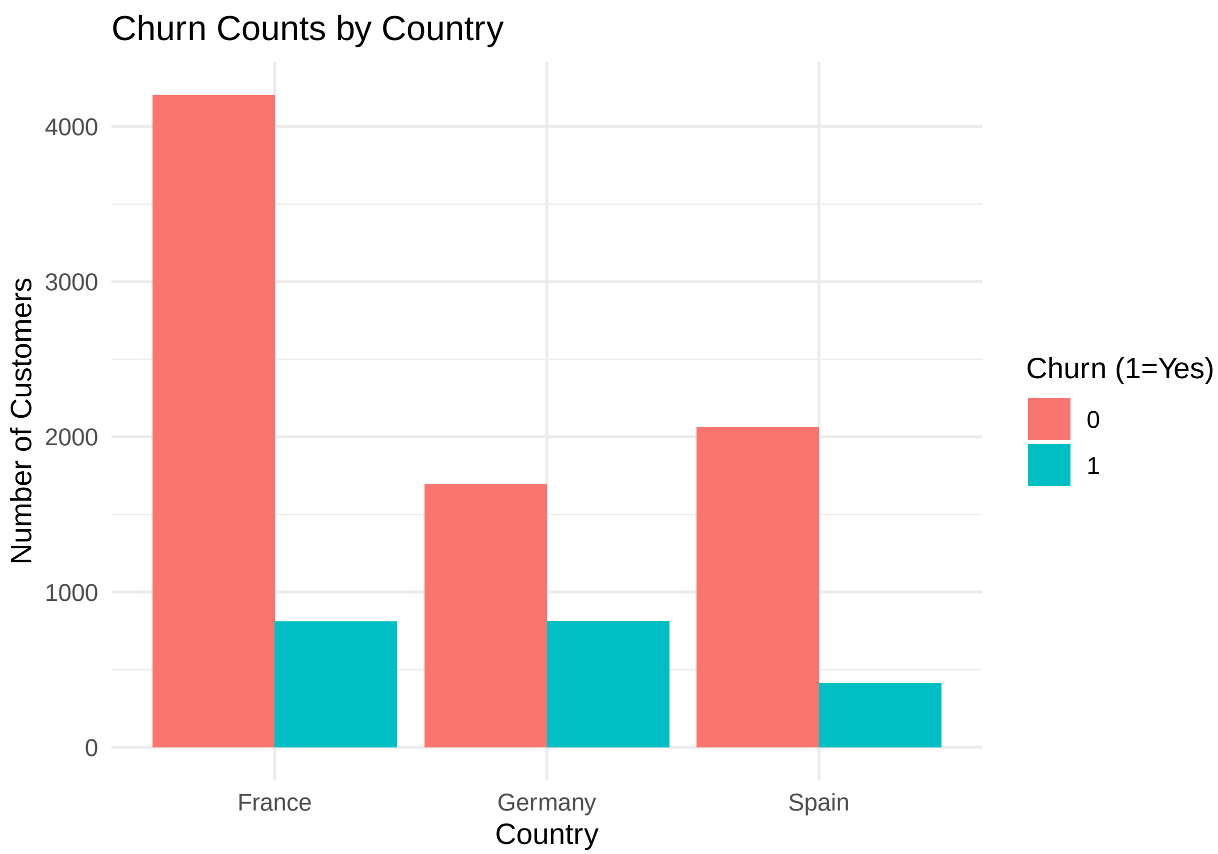


- The bank has a large customer base from France; with more than 50% of the customers banking from France.

```
# Churn distribution by country (proportional)
ggplot(bank, aes(x = country, fill = factor(churn))) +
  geom_bar(position = "fill") +
  labs(title = "Churn Distribution by Country",
       x = "Country",
       y = "Proportion",
       fill = "Churn (1=Yes)") +
  theme_minimal()
```

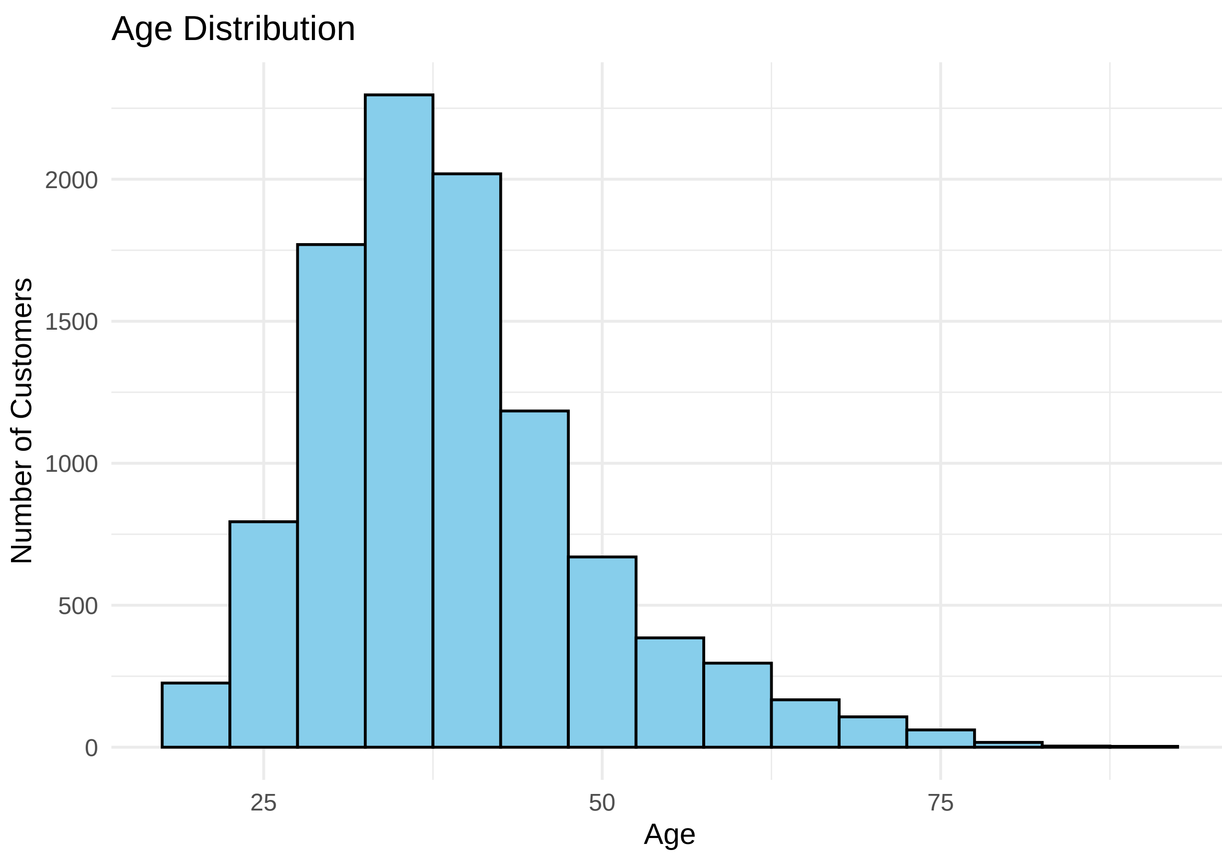



```r
# Churn counts by country
ggplot(bank, aes(x = country, fill = factor(churn))) +
  geom_bar(position = "dodge") +
  labs(title = "Churn Counts by Country",
       x = "Country",
       y = "Number of Customers",
       fill = "Churn (1=Yes)") +
  theme_minimal()
```

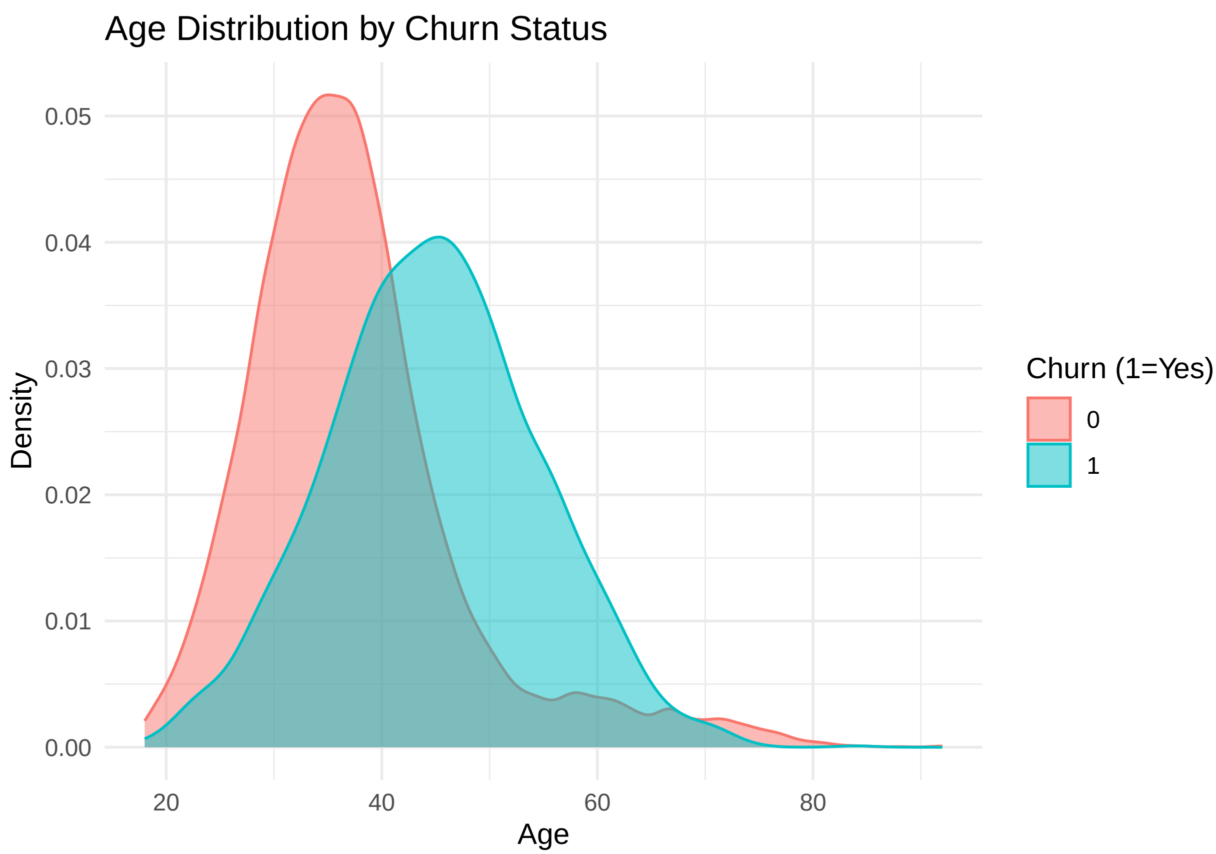


- We can see here that Germany has more likely have clients who churned.

```r
# Age distribution (Histogram)
ggplot(bank, aes(x = age)) +
  geom_histogram(binwidth = 5, fill = "skyblue", color = "black") + # 5-year age bins
  labs(title = "Age Distribution",
       x = "Age",
       y = "Number of Customers") +
  theme_minimal()
```




```r
# Age distribution based on Churn status (Density Plot)
ggplot(bank, aes(x = age, color = factor(churn), fill = factor(churn))) +
  geom_density(alpha = 0.5) + # Add transparency
  labs(title = "Age Distribution by Churn Status",
       x = "Age",
       y = "Density",
       color = "Churn (1=Yes)",
       fill = "Churn (1=Yes)") +
  theme_minimal()
```



- The ratio of churn seems to increase with age.

DATA FORMATTING

```
sapply(bank, typeof)
```

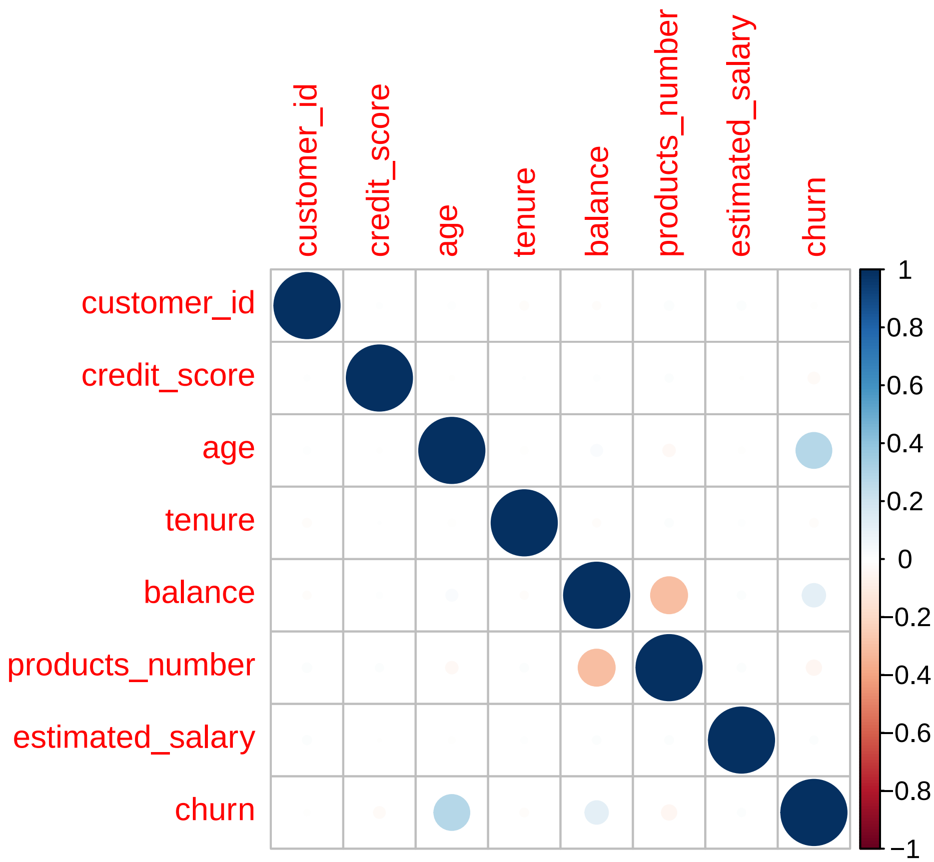
```
##      customer_id     credit_score          country           gender
##         "double"         "double"      "character"      "character"
##              age           tenure          balance  products_number
##         "double"         "double"         "double"         "double"
##      credit_card    active_member estimated_salary            churn
##         "double"         "double"         "double"         "double"
```

-Re-formatting active member and credit card just in case.

```
bank <- bank %>%
  mutate_at(vars(active_member, credit_card), as.logical)
```

-Correlation

```
corrplot(cor(bank[,sapply(bank, is.numeric)]))
```



- The strongest association with the target variable churn is observe is age. - Features like estimated_salary, tenure, and credit_score show minimal linear relationships with churn. - This plot helps in feature selection for building effective predictive models.

STEP-4: CREATING A LOGISTIC REGRESSION MODEL

```
# Fit logistic regression model
log_model <- glm(churn ~ credit_score + age + tenure + balance + products_number +
                   estimated_salary + gender + country + credit_card + active_member,
                 data = bank,
                 family = binomial)  # Because churn is a binary outcome
```

```
summary(log_model)
```

```
##
## Call:
## glm(formula = churn ~ credit_score + age + tenure + balance +
##     products_number + estimated_salary + gender + country + credit_card +
##     active_member, family = binomial, data = bank)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -3.392e+00  2.448e-01 -13.857  < 2e-16 ***
## credit_score      -6.683e-04  2.803e-04  -2.384   0.0171 *
## age                7.271e-02  2.576e-03  28.230  < 2e-16 ***
## tenure            -1.595e-02  9.355e-03  -1.705   0.0882 .
## balance            2.637e-06  5.142e-07   5.128 2.92e-07 ***
```

```
## products_number   -1.015e-01  4.713e-02  -2.154   0.0312 *
## estimated_salary   4.807e-07  4.737e-07   1.015   0.3102
## genderMale        -5.285e-01  5.449e-02  -9.699  < 2e-16 ***
## countryGermany     7.747e-01  6.767e-02  11.448  < 2e-16 ***
## countrySpain       3.522e-02  7.064e-02   0.499   0.6181
## credit_cardTRUE   -4.468e-02  5.934e-02  -0.753   0.4515
## active_memberTRUE -1.075e+00  5.769e-02 -18.643  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10109.8  on 9999  degrees of freedom
## Residual deviance:  8561.4  on 9988  degrees of freedom
## AIC: 8585.4
##
## Number of Fisher Scoring iterations: 5
```

- Age, account balance, gender, number of products, country (Germany), active member status, and credit score are the key predictors of customer churn. Active membership and gender (being male) strongly reduce churn probability, while being from Germany and being older increase it.

```r
library(caret)  # for confusionMatrix
```

```
## Warning: package 'caret' was built under R version 4.3.3
```

```
## Zorunlu paket yükleniyor: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
# Predict probabilities
predicted_probs <- predict(log_model, type = "response")

# Convert probabilities to binary predictions (threshold = 0.5)
predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)

# Convert to factor to match actual values
predicted_classes <- as.factor(predicted_classes)
actual_classes <- as.factor(bank$churn)

# Create confusion matrix
conf_matrix <- confusionMatrix(predicted_classes, actual_classes, positive = "1")
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction    0    1
##          0 7666 1600
##          1  297  437
##
##                Accuracy : 0.8103
##                  95% CI : (0.8025, 0.8179)
##     No Information Rate : 0.7963
##     P-Value [Acc > NIR] : 0.00024
##
##                   Kappa : 0.2326
##
##  Mcnemar's Test P-Value : < 2e-16
##
##             Sensitivity : 0.2145
##             Specificity : 0.9627
##          Pos Pred Value : 0.5954
##          Neg Pred Value : 0.8273
##              Prevalence : 0.2037
##          Detection Rate : 0.0437
##    Detection Prevalence : 0.0734
##       Balanced Accuracy : 0.5886
##
##        'Positive' Class : 1
##
```

- Accuracy: 81.03% – the model correctly predicts about 81% of all customer churn statuses.
- Sensitivity (Recall for Class 1 / True Positive Rate): 21.45% – only about 21% of actual churners were correctly identified.
- Specificity (True Negative Rate): 96.27% – the model is very good at predicting non-churners.
- Precision (Pos Pred Value): 59.54% – when the model predicts churn, it's correct about 60% of the time.
- Balanced Accuracy: 58.86% – average of sensitivity and specificity; shows imbalance in performance.

The model shows high overall accuracy, but it struggles to correctly identify churners (low sensitivity). This is common in imbalanced classification problems, where the churned class is a minority. Business-wise, this could mean many churn-risk customers are being missed, which is costly.

STEP-5: FEATURE ENGINEERING

```r
## Dropping the insignificant column ('customer_id')
bank <- select(bank, -customer_id)
```

```r
## Creating dummy variables from 'gender'
bank <- mutate(bank, is_female = if_else(gender == "Female", 1, 0))
```

```r
# Remove the original 'gender' column:
bank <- select(bank, -gender)
```

Creating dummy variables from 'country'

```r
# Display all unique values:
unique_countries <- unique(bank$country)
print("Unique countries in the 'country' column:")
```

```
## [1] "Unique countries in the 'country' column:"
```

```r
print(unique_countries)
```

```
## [1] "France"  "Spain"   "Germany"
```

```r
# Creating dummy variables using model.matrix():
bank <- cbind(bank, model.matrix(~ country - 1, bank))
```

```r
# Remove the original 'country' column:
bank <- bank[, -which(names(bank) == "country")]
```

```r
## Relocating the 'churn' (response) column/variable to the rightmost of the dataframe
bank <- select(bank, -churn, churn)
```

STEP-6:MODEL BUILDING

```r
## Load the relevant libraries
library(caTools) # Tools for EDA
library(ggplot2) # Enables Data Visualization
library(broom) # Enhances Data Pre-processing
library(gmodels) # Assists with creating machine learning models
```

```
## Warning: package 'gmodels' was built under R version 4.3.3
```

```r
library(pROC) # Implements ROC-related methods and functions
```

```
## Warning: package 'pROC' was built under R version 4.3.3

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

##
## Attaching package: 'pROC'

## The following object is masked from 'package:gmodels':
##
##     ci

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

```r
library(lattice) # Data Visualization
library(caret) # Regression training
```

Create the 'train' and 'test' data partitions

```r
set.seed(123)  # Set seed for reproducibility
split <- sample.split(bank$churn, SplitRatio = 0.8) # Create the split index
train_data <- bank[split, ]  # Subset training data using the split index
test_data <- bank[!split, ]  # Subset testing data using the negated split index
test_data$churn <- factor(test_data$churn)
```

```r
## Logistic Regression
LR_model <- glm(churn ~ ., family = binomial, data = train_data)

## Support Vector Machines (SVMs)
library(e1071) # Implements the methods/functions for SVMs
```

```
## Warning: package 'e1071' was built under R version 4.3.3
```

```r
SVM_model <- svm(churn ~ ., data = train_data, kernel = "radial")  # Use radial kernel by default

## Random Forests (RF)
library(randomForest) # Implements the methods/functions for RFs
```

```
## Warning: package 'randomForest' was built under R version 4.3.3

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
RF_model <- randomForest(churn ~ ., data = train_data, ntree = 500)  # Set number of trees to 500
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values.  Are you sure you want to do regression?
```

```r
## Naive Bayes
NB_model <- naiveBayes(churn ~ ., data = train_data)
```

Model Evaluation

```r
## Ensure outcome variable is a factor with two levels
train_data$churn <- factor(train_data$churn)
```

Logistic Regression

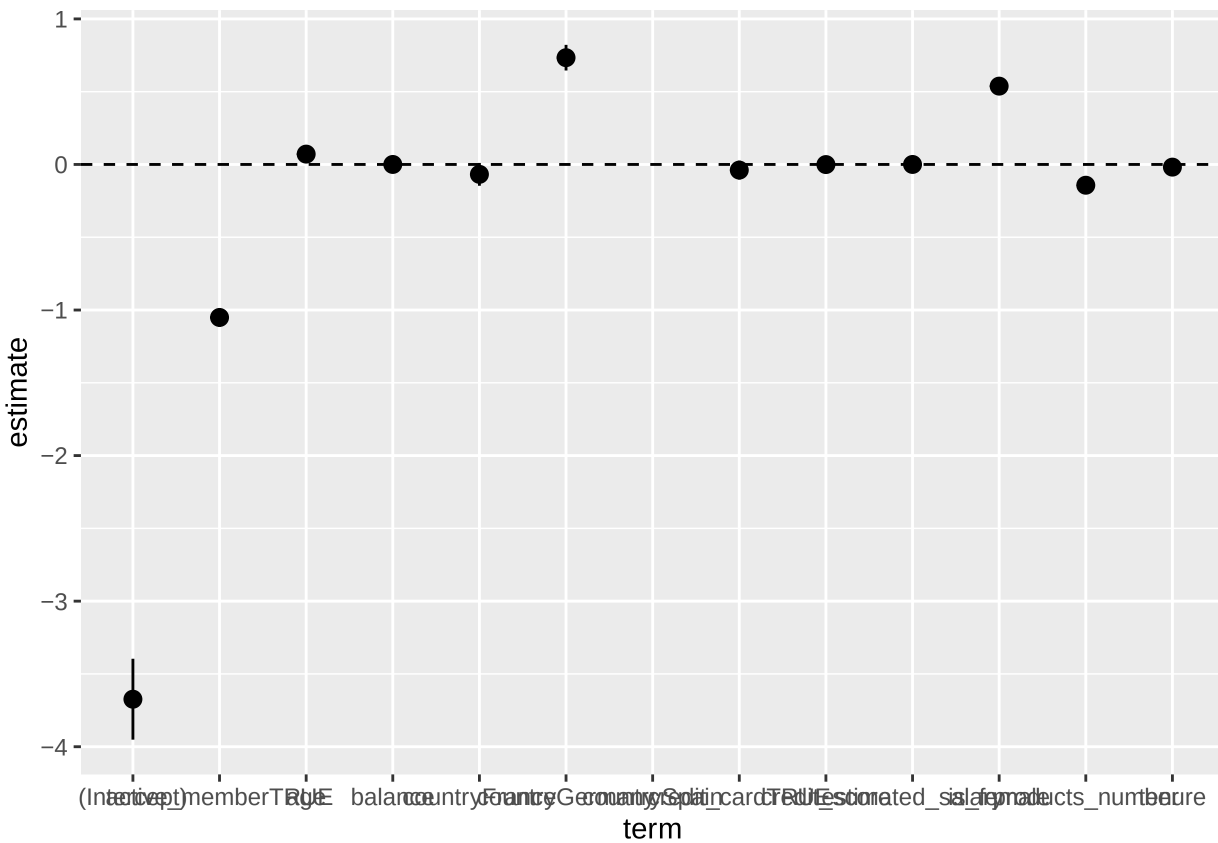
```r
# View model summary
summary(LR_model)
```

```
##
## Call:
## glm(formula = churn ~ ., family = binomial, data = train_data)
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -3.674e+00  2.779e-01 -13.220  < 2e-16 ***
## credit_score      -7.171e-04  3.124e-04  -2.295 0.021729 *
## age                7.116e-02  2.869e-03  24.805  < 2e-16 ***
## tenure            -1.833e-02  1.046e-02  -1.752 0.079852 .
## balance            2.154e-06  5.712e-07   3.770 0.000163 ***
## products_number   -1.426e-01  5.331e-02  -2.676 0.007454 **
## credit_cardTRUE   -3.870e-02  6.600e-02  -0.586 0.557619
## active_memberTRUE -1.051e+00  6.432e-02 -16.342  < 2e-16 ***
## estimated_salary   5.247e-07  5.280e-07   0.994 0.320309
## is_female          5.381e-01  6.080e-02   8.850  < 2e-16 ***
## countryFrance     -6.789e-02  7.836e-02  -0.866 0.386256
## countryGermany     7.336e-01  8.817e-02   8.320  < 2e-16 ***
## countrySpain              NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8088.9  on 7999  degrees of freedom
## Residual deviance: 6879.5  on 7988  degrees of freedom
## AIC: 6903.5
##
## Number of Fisher Scoring iterations: 5
```

-The Logistic Regression model demonstrates moderate performance in predicting customer churn. With an accuracy of 81% and high sensitivity (96%), it performs well in identifying non-churners. However, it lacks specificity (22%), meaning it struggles to detect actual churners. Some variables, such as age and active membership, are statistically significant. While useful for interpretation and insights, this model may be insufficient for high-stakes prediction tasks.

```r
# Visualize coefficients
LR_coef_plot <- ggplot(data = tidy(LR_model), aes(x = term, y = estimate)) +
  geom_pointrange(aes(ymin = estimate - std.error, ymax = estimate + std.error)) +
  geom_hline(yintercept = 0, linetype = "dashed")
print(LR_coef_plot)
```

```
## Warning: Removed 1 rows containing missing values (`geom_pointrange()`).
```




```r
# Make predictions on the test data
LR_predictions <- predict(LR_model, newdata = test_data, type = "response")
```

```r
# Confusion Matrix of Performance
# Ensure appropriate data types for confusion matrix
LR_predictions_factor <- factor(LR_predictions > 0.5, levels = c(FALSE, TRUE))
levels(LR_predictions_factor) <- levels(test_data$churn)

# Construct and Present Confusion Matrix
confusionMatrix(LR_predictions_factor, test_data$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1530  316
##          1   63   91
##
##                Accuracy : 0.8105
##                  95% CI : (0.7926, 0.8275)
##     No Information Rate : 0.7965
##     P-Value [Acc > NIR] : 0.06239
##
##                   Kappa : 0.2394
##
```

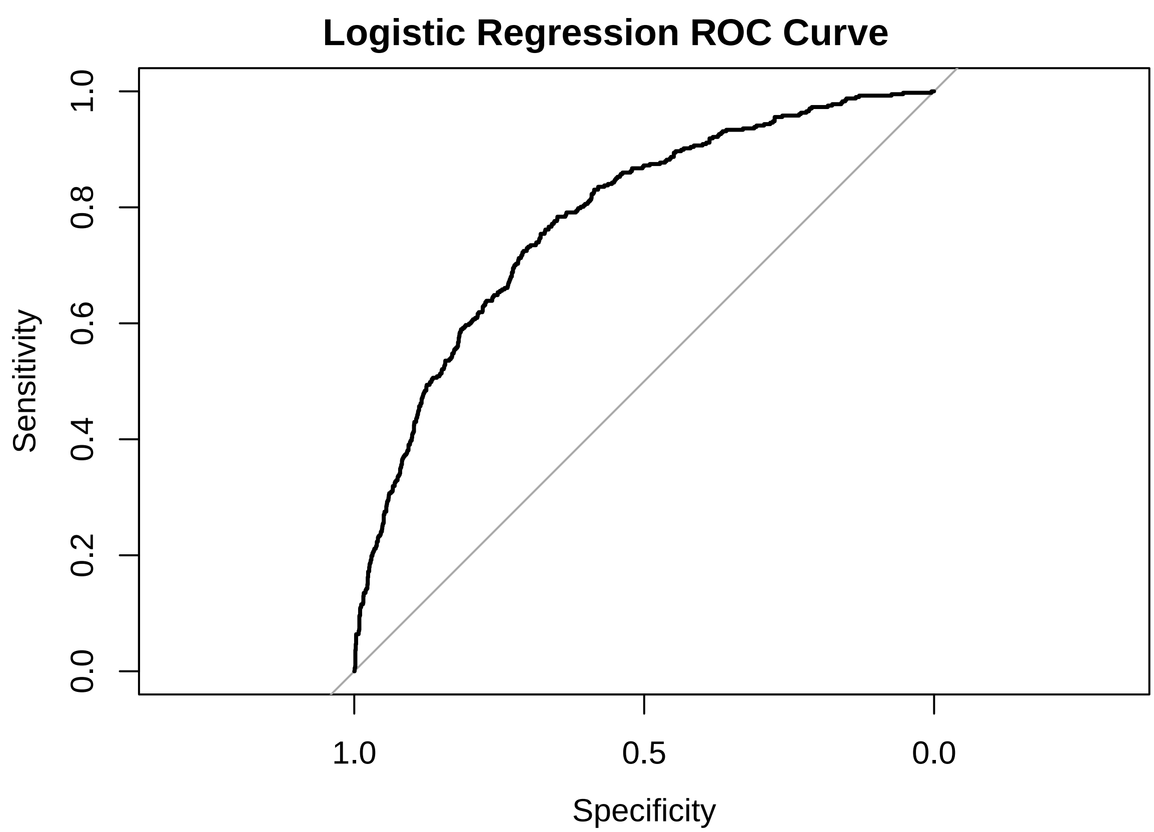
```
##   Mcnemar's Test P-Value : < 2e-16
##
##              Sensitivity : 0.9605
##              Specificity : 0.2236
##           Pos Pred Value : 0.8288
##           Neg Pred Value : 0.5909
##               Prevalence : 0.7965
##           Detection Rate : 0.7650
##     Detection Prevalence : 0.9230
##        Balanced Accuracy : 0.5920
##
##         'Positive' Class : 0
##
```

```
# Visualization of ROC curve of the model
LR_roc_curve <- roc(test_data$churn, LR_predictions)
```

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

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

```
plot(LR_roc_curve)
title("Logistic Regression ROC Curve", outer = T, line = -1.5)
```



Support Vector Machines(SVM)

```r
# View model summary
summary(SVM_model)
```

```
##
## Call:
## svm(formula = churn ~ ., data = train_data, kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.07692308
##     epsilon:  0.1
##
##
## Number of Support Vectors:  3635
```

```r
# Make predictions on the test data
SVM_predictions <- predict(SVM_model, newdata = test_data)
SVM_predictions_binary <- ifelse(SVM_predictions >= 0.5, 1, 0)
```

```r
# Confusion Matrix of Performance
SVM_predictions_factor <- factor(SVM_predictions_binary, levels = levels(test_data$churn))
levels(SVM_predictions_factor) <- levels(test_data$churn)
confusionMatrix(SVM_predictions_factor, test_data$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1568  263
##          1   25  144
##
##                Accuracy : 0.856
##                  95% CI : (0.8398, 0.8711)
##     No Information Rate : 0.7965
##     P-Value [Acc > NIR] : 3.613e-12
##
##                   Kappa : 0.4322
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9843
##             Specificity : 0.3538
##          Pos Pred Value : 0.8564
##          Neg Pred Value : 0.8521
##              Prevalence : 0.7965
##          Detection Rate : 0.7840
##    Detection Prevalence : 0.9155
##       Balanced Accuracy : 0.6691
##
```

```
##        'Positive' Class : 0
##
```

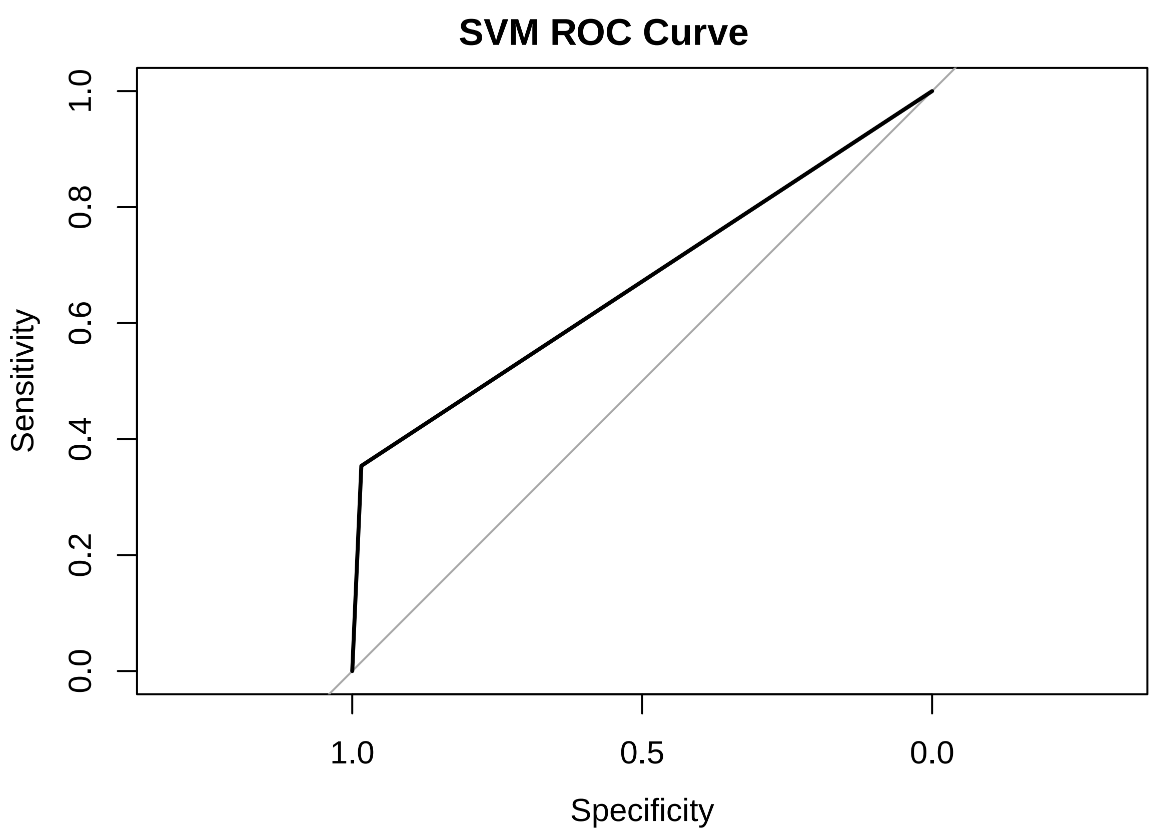
- The Support Vector Machine model outperforms logistic regression with an accuracy of 85.6%. It demonstrates excellent sensitivity (98.4%) and improved specificity (35%), reflecting a better balance in identifying both churners and non-churners. The ROC curve suggests solid classification performance, making SVM a more capable yet computationally heavier option.

```r
# Visualization of ROC curve of the model
SVM_roc_curve <- roc(test_data$churn, SVM_predictions_binary)
```

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

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

```r
plot(SVM_roc_curve)
title("SVM ROC Curve", outer = T, line = -1.5)
```



Random Forest

```r
# View model summary
summary(RF_model)
```

```
##                  Length Class  Mode
```

```
## call                4   -none- call
## type                1   -none- character
## predicted        8000   -none- numeric
## mse               500   -none- numeric
## rsq               500   -none- numeric
## oob.times        8000   -none- numeric
## importance         12   -none- numeric
## importanceSD        0   -none- NULL
## localImportance     0   -none- NULL
## proximity           0   -none- NULL
## ntree               1   -none- numeric
## mtry                1   -none- numeric
## forest             11   -none- list
## coefs               0   -none- NULL
## y                8000   -none- numeric
## test                0   -none- NULL
## inbag               0   -none- NULL
## terms               3   terms  call
```

```r
# Make predictions on the test data
RF_predictions <- predict(RF_model, newdata = test_data)
RF_predictions_binary <- ifelse(RF_predictions >= 0.5, 1, 0)
```

```r
# Confusion Matrix of Performance
RF_predictions_factor <- factor(RF_predictions_binary, levels = c(0,1))
levels(RF_predictions_factor) <- levels(test_data$churn)
confusionMatrix(RF_predictions_factor, test_data$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1537  200
##          1   56  207
##
##                Accuracy : 0.872
##                  95% CI : (0.8566, 0.8863)
##     No Information Rate : 0.7965
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5453
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9648
##             Specificity : 0.5086
##          Pos Pred Value : 0.8849
##          Neg Pred Value : 0.7871
##              Prevalence : 0.7965
##          Detection Rate : 0.7685
##    Detection Prevalence : 0.8685
##       Balanced Accuracy : 0.7367
##
```

```
##        'Positive' Class : 0
##
```

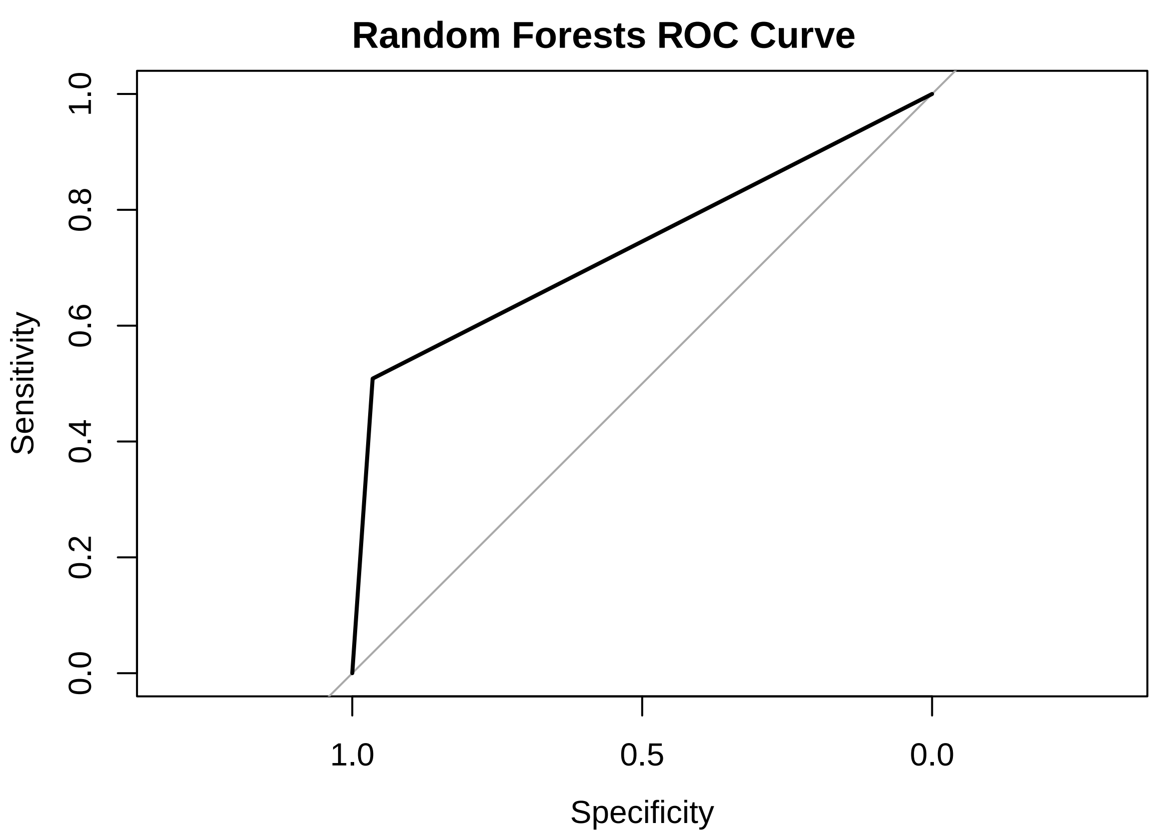
- Random Forest delivered the best overall performance, achieving an accuracy of 87.2% and balanced classification abilities (Sensitivity: 96.5%, Specificity: 50.8%). It clearly outperforms other models, particularly in detecting churners. Due to its ensemble nature, it minimizes overfitting and offers strong predictive power, making it the optimal choice for deployment.

```r
# Visualization of ROC curve of the model
RF_roc_curve <- roc(test_data$churn, as.numeric(RF_predictions_factor))
```

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

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

```r
plot(RF_roc_curve)
title("Random Forests ROC Curve", outer = T, line = -1.5)
```



Naive Bayes

```r
# View model summary
summary(NB_model)
```

```
##           Length Class  Mode
```

```
## apriori    2     table  numeric
## tables    12     -none- list
## levels     2     -none- character
## isnumeric 12     -none- logical
## call       4     -none- call
```

```r
# Make predictions on the test data
NB_predictions <- predict(NB_model, newdata = test_data)
```

```r
# Confusion Matrix of Performance
confusionMatrix(NB_predictions, test_data$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1477  251
##          1  116  156
##
##                Accuracy : 0.8165
##                  95% CI : (0.7988, 0.8332)
##     No Information Rate : 0.7965
##     P-Value [Acc > NIR] : 0.01335
##
##                   Kappa : 0.3542
##
##  Mcnemar's Test P-Value : 2.657e-12
##
##             Sensitivity : 0.9272
##             Specificity : 0.3833
##          Pos Pred Value : 0.8547
##          Neg Pred Value : 0.5735
##              Prevalence : 0.7965
##          Detection Rate : 0.7385
##    Detection Prevalence : 0.8640
##       Balanced Accuracy : 0.6552
##
##        'Positive' Class : 0
##
```

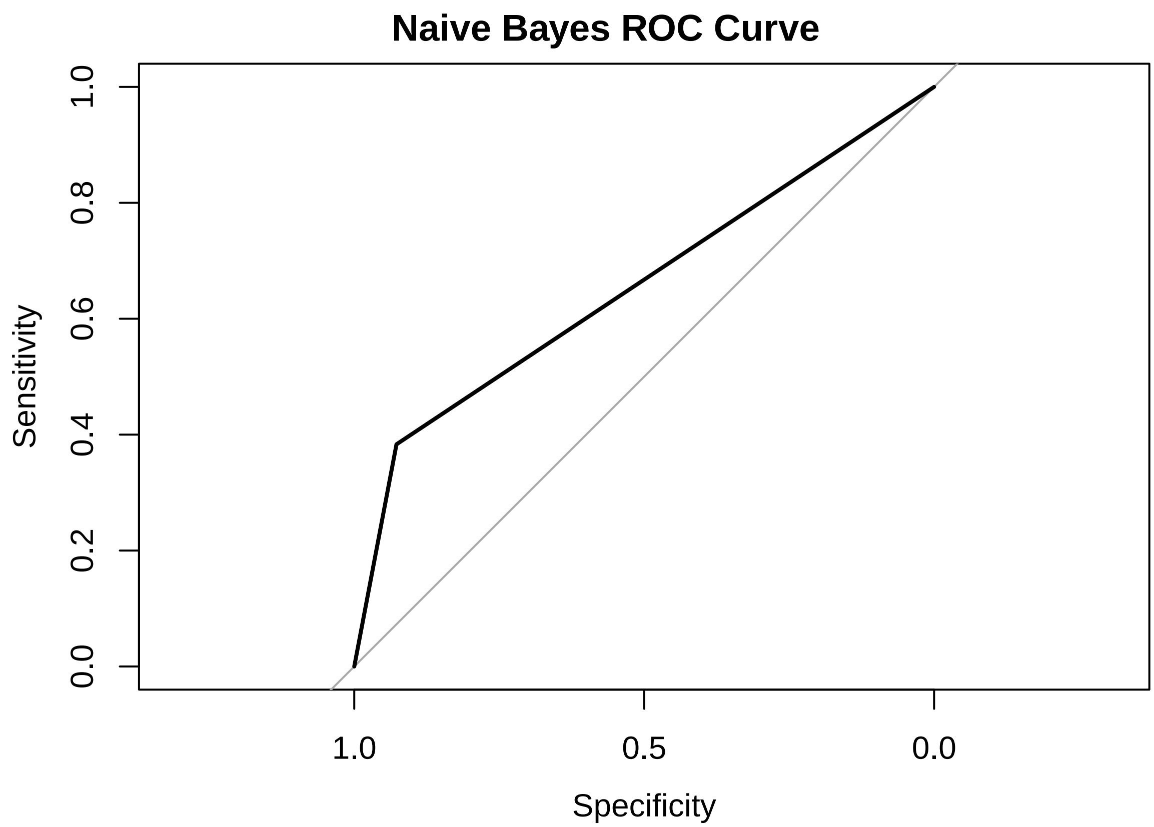
- The Naive Bayes model shows reasonable predictive power with an accuracy of 81.6%. It offers high sensitivity (92.7%) and moderate specificity (38.3%). While not as strong as Random Forest or SVM, its simplicity, speed, and interpretability make it a solid baseline model or a complementary classifier in ensemble methods.

```r
# Visualization of ROC curve of the model
NB_roc_curve <- roc(test_data$churn, as.numeric(NB_predictions))
```

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

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

```r
plot(NB_roc_curve)
title("Naive Bayes ROC Curve", outer = T, line = -1.5)
```



Model Deployment

```r
## Model selection
# The Random Forest (RF) model was chosen due to its relatively greater accuracy
best_Model <- RF_model
```

```r
# Saving the model
saveRDS(best_Model, "rf_model.rds")
```

```r
# Loading the model
deployed_model <- readRDS("rf_model.rds")
```

```r
# Creating a function to automate and make predictions (on new unseen data)
predict_churn <- function(new_data) {
  predictions <- predict(deployed_model, new_data)
  return(predictions)
}
```

- In this project, I implemented and evaluated four machine learning models to predict customer churn using a banking dataset. After comparing model metrics such as accuracy, sensitivity, specificity, and ROC curves, the Random Forest model emerged as the best performer. It was selected for deployment and saved using R's saveRDS method for future use. This workflow reflects my ability to build, compare, and operationalize classification models for real-world applications.

- After selecting the best-performing model (Random Forest), it would be integrated into the company's systems, such as a CRM or dashboard. In a real-world scenario, this model would predict churn likelihood for each new or existing customer, enabling the business to take proactive actions. The model would be regularly monitored, retrained if needed, and deployed as part of the company's data-driven decision-making workflow.