Week6_Tasks

Tran Anh Chuong

2025-10-22

Task 1 — Data Loading and Initial Exploration

First, we load the necessary libraries. The `tidyverse` is a collection of R packages for data science, including `dplyr` for data manipulation and `readr` for reading data.

```
library(tidyverse)
library(skimr) # For detailed summary statistics
```

Dataset Characteristics

- **Unit of analysis:** One row per customer; target = `Attrition_Flag`.
- **Numerics:** activity (`Total_Trans_Ct`, `Total_Trans_Amt`), exposure (`Credit_Limit`, `Total_Revolving_Bal`), ratios (`Avg_Utilization_Ratio`, quarterly change rates), tenure/inactivity (months).
- **Categoricals:** `Gender`, `Education_Level`, `Income_Category`, `Marital_Status`, `Card_Category`.
- **Leakage/Redundancy:** `Naive_Bayes_Classifier_*` (pre-scored), `Avg_Open_To_Buy  Credit_Limit - Total_Revolving_Bal`.
- **Assumed bounds/units:** utilization in [0,1]; counts non-negative; amounts 0; tenure in months.

```
# Load the dataset from the CSV file
df <- read_csv('BankChurners.csv')

# Display the first few rows
head(df)
```

```
## # A tibble: 6 x 23
##   CLIENTNUM Attrition_Flag   Customer_Age Gender Dependent_count Education_Level
##       <dbl> <chr>                   <dbl> <chr>            <dbl> <chr>
## 1 768805383 Existing Custom~           45 M                    3 High School
## 2 818770008 Existing Custom~           49 F                    5 Graduate
## 3 713982108 Existing Custom~           51 M                    3 Graduate
## 4 769911858 Existing Custom~           40 F                    4 High School
## 5 709106358 Existing Custom~           40 M                    3 Uneducated
## 6 713061558 Existing Custom~           44 M                    2 Graduate
## # i 17 more variables: Marital_Status <chr>, Income_Category <chr>,
## #   Card_Category <chr>, Months_on_book <dbl>, Total_Relationship_Count <dbl>,
## #   Months_Inactive_12_mon <dbl>, Contacts_Count_12_mon <dbl>,
## #   Credit_Limit <dbl>, Total_Revolving_Bal <dbl>, Avg_Open_To_Buy <dbl>,
## #   Total_Amt_Chng_Q4_Q1 <dbl>, Total_Trans_Amt <dbl>, Total_Trans_Ct <dbl>,
## #   Total_Ct_Chng_Q4_Q1 <dbl>, Avg_Utilization_Ratio <dbl>,
## #   Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Educati
```

The original dataset includes two long `Naive_Bayes_Classifier_*` columns that are pre-scored and represent data leakage. We will remove them immediately.

```
# Remove the two Naive Bayes classifier columns by selecting them out
# The 'select' function keeps columns, so we use '-' to remove them.
data <- df %>%
  select(
    -`Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Educati
    -`Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Educati
  )

# Display the dimensions and first few rows of the cleaned data
glimpse(data)
```

```
## Rows: 10,127
## Columns: 21
## $ CLIENTNUM                <dbl> 768805383, 818770008, 713982108, 769911858, 7~
## $ Attrition_Flag           <chr> "Existing Customer", "Existing Customer", "Ex~
## $ Customer_Age             <dbl> 45, 49, 51, 40, 40, 44, 51, 32, 37, 48, 42, 6~
## $ Gender                   <chr> "M", "F", "M", "F", "M", "M", "M", "M", "M", ~
## $ Dependent_count          <dbl> 3, 5, 3, 4, 3, 2, 4, 0, 3, 2, 5, 1, 1, 3, 2, ~
## $ Education_Level          <chr> "High School", "Graduate", "Graduate", "High ~
## $ Marital_Status           <chr> "Married", "Single", "Married", "Unknown", "M~
## $ Income_Category          <chr> "$60K - $80K", "Less than $40K", "$80K - $120~
## $ Card_Category            <chr> "Blue", "Blue", "Blue", "Blue", "Blue", "Blue~
## $ Months_on_book           <dbl> 39, 44, 36, 34, 21, 36, 46, 27, 36, 36, 31, 5~
## $ Total_Relationship_Count <dbl> 5, 6, 4, 3, 5, 3, 6, 2, 5, 6, 5, 6, 3, 5, 5, ~
## $ Months_Inactive_12_mon   <dbl> 1, 1, 1, 4, 1, 1, 1, 2, 2, 3, 3, 2, 6, 1, 2, ~
## $ Contacts_Count_12_mon    <dbl> 3, 2, 0, 1, 0, 2, 3, 2, 0, 3, 2, 3, 0, 3, 2, ~
## $ Credit_Limit             <dbl> 12691.0, 8256.0, 3418.0, 3313.0, 4716.0, 4010~
## $ Total_Revolving_Bal      <dbl> 777, 864, 0, 2517, 0, 1247, 2264, 1396, 2517,~
## $ Avg_Open_To_Buy          <dbl> 11914.0, 7392.0, 3418.0, 796.0, 4716.0, 2763.~
## $ Total_Amt_Chng_Q4_Q1     <dbl> 1.335, 1.541, 2.594, 1.405, 2.175, 1.376, 1.9~
## $ Total_Trans_Amt          <dbl> 1144, 1291, 1887, 1171, 816, 1088, 1330, 1538~
## $ Total_Trans_Ct           <dbl> 42, 33, 20, 20, 28, 24, 31, 36, 24, 32, 42, 2~
## $ Total_Ct_Chng_Q4_Q1      <dbl> 1.625, 3.714, 2.333, 2.333, 2.500, 0.846, 0.7~
## $ Avg_Utilization_Ratio    <dbl> 0.061, 0.105, 0.000, 0.760, 0.000, 0.311, 0.0~
```

Let's get a summary of the data types and check for missing values. `glimpse()` is great for this.

```
# glimpse() provides a transposed view of the dataframe, showing column types and first values.
glimpse(data)
```

```
## Rows: 10,127
## Columns: 21
## $ CLIENTNUM                <dbl> 768805383, 818770008, 713982108, 769911858, 7~
## $ Attrition_Flag           <chr> "Existing Customer", "Existing Customer", "Ex~
## $ Customer_Age             <dbl> 45, 49, 51, 40, 40, 44, 51, 32, 37, 48, 42, 6~
## $ Gender                   <chr> "M", "F", "M", "F", "M", "M", "M", "M", "M", ~
## $ Dependent_count          <dbl> 3, 5, 3, 4, 3, 2, 4, 0, 3, 2, 5, 1, 1, 3, 2, ~
## $ Education_Level          <chr> "High School", "Graduate", "Graduate", "High ~
## $ Marital_Status           <chr> "Married", "Single", "Married", "Unknown", "M~
## $ Income_Category          <chr> "$60K - $80K", "Less than $40K", "$80K - $120~
```

```
## $ Card_Category            <chr> "Blue", "Blue", "Blue", "Blue", "Blue", "Blue~
## $ Months_on_book           <dbl> 39, 44, 36, 34, 21, 36, 46, 27, 36, 36, 31, 5~
## $ Total_Relationship_Count <dbl> 5, 6, 4, 3, 5, 3, 6, 2, 5, 6, 5, 6, 3, 5, 5, ~
## $ Months_Inactive_12_mon   <dbl> 1, 1, 1, 4, 1, 1, 1, 2, 2, 3, 3, 2, 6, 1, 2, ~
## $ Contacts_Count_12_mon    <dbl> 3, 2, 0, 1, 0, 2, 3, 2, 0, 3, 2, 3, 0, 3, 2, ~
## $ Credit_Limit             <dbl> 12691.0, 8256.0, 3418.0, 3313.0, 4716.0, 4010~
## $ Total_Revolving_Bal      <dbl> 777, 864, 0, 2517, 0, 1247, 2264, 1396, 2517,~
## $ Avg_Open_To_Buy          <dbl> 11914.0, 7392.0, 3418.0, 796.0, 4716.0, 2763.~
## $ Total_Amt_Chng_Q4_Q1     <dbl> 1.335, 1.541, 2.594, 1.405, 2.175, 1.376, 1.9~
## $ Total_Trans_Amt          <dbl> 1144, 1291, 1887, 1171, 816, 1088, 1330, 1538~
## $ Total_Trans_Ct           <dbl> 42, 33, 20, 20, 28, 24, 31, 36, 24, 32, 42, 2~
## $ Total_Ct_Chng_Q4_Q1      <dbl> 1.625, 3.714, 2.333, 2.333, 2.500, 0.846, 0.7~
## $ Avg_Utilization_Ratio    <dbl> 0.061, 0.105, 0.000, 0.760, 0.000, 0.311, 0.0~
```

Now, let's generate descriptive statistics for all numeric columns. `summary()` is the base R function for this.

```r
# summary() provides min, median, mean, max, and quartiles for numeric columns
summary(data %>% select(where(is.numeric)))
```

```
##    CLIENTNUM          Customer_Age   Dependent_count Months_on_book
##  Min.   :708082083   Min.   :26.00   Min.   :0.000   Min.   :13.00
##  1st Qu.:713036770   1st Qu.:41.00   1st Qu.:1.000   1st Qu.:31.00
##  Median :717926358   Median :46.00   Median :2.000   Median :36.00
##  Mean   :739177606   Mean   :46.33   Mean   :2.346   Mean   :35.93
##  3rd Qu.:773143533   3rd Qu.:52.00   3rd Qu.:3.000   3rd Qu.:40.00
##  Max.   :828343083   Max.   :73.00   Max.   :5.000   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 :2.000
##  Mean   :3.813            Mean   :2.341          Mean   :2.455
##  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.   : 1438   Min.   :   0        Min.   :    3   Min.   :0.0000
##  1st Qu.: 2555   1st Qu.: 359        1st Qu.: 1324   1st Qu.:0.6310
##  Median : 4549   Median :1276        Median : 3474   Median :0.7360
##  Mean   : 8632   Mean   :1163        Mean   : 7469   Mean   :0.7599
##  3rd Qu.:11068   3rd Qu.:1784        3rd Qu.: 9859   3rd Qu.:0.8590
##  Max.   :34516   Max.   :2517        Max.   :34516   Max.   :3.3970
##  Total_Trans_Amt Total_Trans_Ct   Total_Ct_Chng_Q4_Q1 Avg_Utilization_Ratio
##  Min.   :  510   Min.   : 10.00   Min.   :0.0000      Min.   :0.0000
##  1st Qu.: 2156   1st Qu.: 45.00   1st Qu.:0.5820      1st Qu.:0.0230
##  Median : 3899   Median : 67.00   Median :0.7020      Median :0.1760
##  Mean   : 4404   Mean   : 64.86   Mean   :0.7122      Mean   :0.2749
##  3rd Qu.: 4741   3rd Qu.: 81.00   3rd Qu.:0.8180      3rd Qu.:0.5030
##  Max.   :18484   Max.   :139.00   Max.   :3.7140      Max.   :0.9990
```

We can also create a more detailed data dictionary to understand each column's role, type, and content.

```r
# This custom function replicates the Python notebook's data dictionary
# Note: This is a more advanced R implementation for demonstration
```

```r
# Define a function to identify the role of each column
get_role <- function(col_name) {
  if (col_name == "Attrition_Flag") return("target")
  if (col_name == "CLIENTNUM") return("id")
  if (str_starts(col_name, "Naive_Bayes_Classifier_")) return("leakage")
  return("feature")
}

# Define a function to get an example or range for a column
get_example_or_range <- function(s) {
  if (is.numeric(s)) {
    str_glue("min={min(s, na.rm=TRUE)}, max={max(s, na.rm=TRUE)}")
  } else {
    # Get top 3 unique non-NA values
    unique_vals <- s %>% na.omit() %>% unique() %>% head(3)
    str_c(unique_vals, collapse = ", ")
  }
}

data_dictionary <- tibble(
  column = names(data),
  role = map_chr(column, get_role),
  type = map_chr(data, ~class(.)[1]),
  n_missing = map_int(data, ~sum(is.na(.))),
  pct_missing = map_dbl(data, ~mean(is.na(.)) * 100),
  n_unique = map_int(data, n_distinct),
  example_or_range = map_chr(data, get_example_or_range)
) %>%
  arrange(role, column)

# Print the data dictionary
print(data_dictionary, n = 25)
```

```
## # A tibble: 21 x 7
##    column            role  type  n_missing pct_missing n_unique example_or_range
##    <chr>             <chr> <chr>     <int>       <dbl>    <int> <chr>
##  1 Avg_Open_To_Buy   feat~ nume~         0           0     6813 min=3, max=34516
##  2 Avg_Utilization_~ feat~ nume~         0           0      964 min=0, max=0.999
##  3 Card_Category     feat~ char~         0           0        4 Blue, Gold, Sil~
##  4 Contacts_Count_1~ feat~ nume~         0           0        7 min=0, max=6
##  5 Credit_Limit      feat~ nume~         0           0     6205 min=1438.3, max~
##  6 Customer_Age      feat~ nume~         0           0       45 min=26, max=73
##  7 Dependent_count   feat~ nume~         0           0        6 min=0, max=5
##  8 Education_Level   feat~ char~         0           0        7 High School, Gr~
##  9 Gender            feat~ char~         0           0        2 M, F
## 10 Income_Category   feat~ char~         0           0        6 $60K - $80K, Le~
## 11 Marital_Status    feat~ char~         0           0        4 Married, Single~
## 12 Months_Inactive_~ feat~ nume~         0           0        7 min=0, max=6
## 13 Months_on_book    feat~ nume~         0           0       44 min=13, max=56
## 14 Total_Amt_Chng_Q~ feat~ nume~         0           0     1158 min=0, max=3.397
## 15 Total_Ct_Chng_Q4~ feat~ nume~         0           0      830 min=0, max=3.714
## 16 Total_Relationsh~ feat~ nume~         0           0        6 min=1, max=6
## 17 Total_Revolving_~ feat~ nume~         0           0     1974 min=0, max=2517
```

```
## 18 Total_Trans_Amt   feat~ nume~         0           0     5033 min=510, max=18~
## 19 Total_Trans_Ct    feat~ nume~         0           0      126 min=10, max=139
## 20 CLIENTNUM         id    nume~         0           0    10127 min=708082083, ~
## 21 Attrition_Flag    targ~ char~         0           0        2 Existing Custom~
```

Assumptions

- `CLIENTNUM` is a customer identifier.
- `Avg_Utilization_Ratio` is within [0,1].
- Tenure fields are non-negative and measured in months.

Let's check for outlier share using the IQR method.

```r
# Function to calculate the share of outliers based on 1.5 * IQR rule
calculate_outlier_share <- function(x) {
  if (!is.numeric(x)) return(NA)
  q1 <- quantile(x, 0.25, na.rm = TRUE)
  q3 <- quantile(x, 0.75, na.rm = TRUE)
  iqr <- q3 - q1
  lower_bound <- q1 - 1.5 * iqr
  upper_bound <- q3 + 1.5 * iqr
  mean(x < lower_bound | x > upper_bound, na.rm = TRUE)
}

# Apply the function to all numeric columns and sort the result
data %>%
  select(where(is.numeric)) %>%
  map_dbl(calculate_outlier_share) %>%
  sort(decreasing = TRUE)
```

```
##             Credit_Limit          Avg_Open_To_Buy          Total_Trans_Amt
##             0.0971659919             0.0950923274             0.0884763504
##    Contacts_Count_12_mon     Total_Amt_Chng_Q4_Q1      Total_Ct_Chng_Q4_Q1
##             0.0621111879             0.0391033870             0.0389058951
##           Months_on_book   Months_Inactive_12_mon             Customer_Age
##             0.0381159277             0.0326849017             0.0001974919
##           Total_Trans_Ct                CLIENTNUM          Dependent_count
##             0.0001974919             0.0000000000             0.0000000000
## Total_Relationship_Count      Total_Revolving_Bal    Avg_Utilization_Ratio
##             0.0000000000             0.0000000000             0.0000000000
```

Task 2 — Cleaning

Strategy

- Trimmed/squished text, normalized case for key categoricals; parsed numeric strings to numbers.
- Median imputed numeric NAs; added `"missing"` category to categorical fields.
- Dropped full row duplicates; removed NB classifier scores and plan to drop `Naive_Bayes_Classifier_` columns due to redundancy, which is also mentioned by the author of the dataset.

- Preserved the original file (`df_raw`) and worked on a copy (`df`) for reproducibility.

```
# Start with the 'data' dataframe which has leakage columns removed
df_clean <- data

# Cleaning character columns: trim whitespace and normalize case
df_clean <- df_clean %>%
  mutate(
    across(where(is.character), str_trim),
    Gender = str_to_upper(Gender),
    Education_Level = str_to_title(Education_Level)
  )

# Convert character columns with a limited number of unique values to factors
df_clean <- df_clean %>%
  mutate(across(c(Attrition_Flag, Gender, Education_Level, Marital_Status, Income_Category, Card_Categor

# Remove duplicate rows
df_clean <- df_clean %>%
  distinct()

# Note: The original dataset had no missing values. The code below is how you would handle them.
# For demonstration: Median imputation for numerics and adding a "missing" category for factors.

# num_cols <- df_clean %>% select(where(is.numeric)) %>% names()
# cat_cols <- df_clean %>% select(where(is.factor)) %>% names()
#
# df_clean <- df_clean %>%
#   mutate(across(all_of(num_cols), ~replace_na(., median(., na.rm = TRUE)))) %>%
#   mutate(across(all_of(cat_cols), ~fct_explicit_na(., "missing")))

glimpse(df_clean)
```

```
## Rows: 10,127
## Columns: 21
## $ CLIENTNUM                <dbl> 768805383, 818770008, 713982108, 769911858, 7~
## $ Attrition_Flag           <fct> Existing Customer, Existing Customer, Existin~
## $ Customer_Age             <dbl> 45, 49, 51, 40, 40, 44, 51, 32, 37, 48, 42, 6~
## $ Gender                   <fct> M, F, M, F, M, M, M, M, M, M, M, M, M, F, ~
## $ Dependent_count          <dbl> 3, 5, 3, 4, 3, 2, 4, 0, 3, 2, 5, 1, 1, 3, 2, ~
## $ Education_Level          <fct> High School, Graduate, Graduate, High School,~
## $ Marital_Status           <fct> Married, Single, Married, Unknown, Married, M~
## $ Income_Category          <fct> $60K - $80K, Less than $40K, $80K - $120K, Le~
## $ Card_Category            <fct> Blue, Blue, Blue, Blue, Blue, Blue, Gold, Sil~
## $ Months_on_book           <dbl> 39, 44, 36, 34, 21, 36, 46, 27, 36, 36, 31, 5~
## $ Total_Relationship_Count <dbl> 5, 6, 4, 3, 5, 3, 6, 2, 5, 6, 5, 6, 3, 5, 5, ~
## $ Months_Inactive_12_mon   <dbl> 1, 1, 1, 4, 1, 1, 1, 2, 2, 3, 3, 2, 6, 1, 2, ~
## $ Contacts_Count_12_mon    <dbl> 3, 2, 0, 1, 0, 2, 3, 2, 0, 3, 2, 3, 0, 3, 2, ~
## $ Credit_Limit             <dbl> 12691.0, 8256.0, 3418.0, 3313.0, 4716.0, 4010~
## $ Total_Revolving_Bal      <dbl> 777, 864, 0, 2517, 0, 1247, 2264, 1396, 2517,~
## $ Avg_Open_To_Buy          <dbl> 11914.0, 7392.0, 3418.0, 796.0, 4716.0, 2763.~
## $ Total_Amt_Chng_Q4_Q1     <dbl> 1.335, 1.541, 2.594, 1.405, 2.175, 1.376, 1.9~
## $ Total_Trans_Amt          <dbl> 1144, 1291, 1887, 1171, 816, 1088, 1330, 1538~
## $ Total_Trans_Ct           <dbl> 42, 33, 20, 20, 28, 24, 31, 36, 24, 32, 42, 2~
```

```
## $ Total_Ct_Chng_Q4_Q1      <dbl> 1.625, 3.714, 2.333, 2.333, 2.500, 0.846, 0.7~
## $ Avg_Utilization_Ratio    <dbl> 0.061, 0.105, 0.000, 0.760, 0.000, 0.311, 0.0~
```

Task 3 — Transformations

Feature engineering logic

- `inactivity_ratio` reflects disengagement risk; `trans_intensity` reflects engagement.
- `age_group` indicate better generalization for the dataset since the age varies from many sectors. Thus, group in to groups such as <=30, 31-40 will helps.
- Log transforms reduce skew for spend/limits; age groups aid reporting and monotonic behavior.

```
df_fe <- df_clean %>%
  mutate(
    # Create age groups
    age_group = cut(
      Customer_Age,
      breaks = c(0, 30, 40, 50, 60, 200),
      labels = c("<=30", "31-40", "41-50", "51-60", "60+"),
      right = TRUE
    ),

    # Create new ratio features
    inactivity_ratio = if_else(Months_on_book > 0, Months_Inactive_12_mon / Months_on_book, NA_real_),
    trans_intensity = if_else(Months_on_book > 0, Total_Trans_Ct / Months_on_book, NA_real_),

    # Add log-transformed versions of skewed numeric columns
    Total_Trans_Amt_log = log1p(Total_Trans_Amt),
    Total_Trans_Ct_log = log1p(Total_Trans_Ct),
    Credit_Limit_log = log1p(Credit_Limit)
  )

# R's modeling functions (like lm, glm) handle factors automatically,
# so explicit one-hot encoding is often not needed for modeling.
# For other purposes, you could use libraries like 'fastDummies'.
# For this script, we'll keep the factor columns as they are.

glimpse(df_fe)
```

```
## Rows: 10,127
## Columns: 27
## $ CLIENTNUM                <dbl> 768805383, 818770008, 713982108, 769911858, 7~
## $ Attrition_Flag           <fct> Existing Customer, Existing Customer, Existin~
## $ Customer_Age             <dbl> 45, 49, 51, 40, 40, 44, 51, 32, 37, 48, 42, 6~
## $ Gender                   <fct> M, F, M, F, M, M, M, M, M, M, M, M, M, F, ~
## $ Dependent_count          <dbl> 3, 5, 3, 4, 3, 2, 4, 0, 3, 2, 5, 1, 1, 3, 2, ~
## $ Education_Level          <fct> High School, Graduate, Graduate, High School,~
## $ Marital_Status           <fct> Married, Single, Married, Unknown, Married, M~
## $ Income_Category          <fct> $60K - $80K, Less than $40K, $80K - $120K, Le~
## $ Card_Category            <fct> Blue, Blue, Blue, Blue, Blue, Blue, Gold, Sil~
```

```
## $ Months_on_book           <dbl> 39, 44, 36, 34, 21, 36, 46, 27, 36, 36, 31, 5~
## $ Total_Relationship_Count <dbl> 5, 6, 4, 3, 5, 3, 6, 2, 5, 6, 5, 6, 3, 5, 5, ~
## $ Months_Inactive_12_mon   <dbl> 1, 1, 1, 4, 1, 1, 1, 2, 2, 3, 3, 2, 6, 1, 2, ~
## $ Contacts_Count_12_mon    <dbl> 3, 2, 0, 1, 0, 2, 3, 2, 0, 3, 2, 3, 0, 3, 2, ~
## $ Credit_Limit             <dbl> 12691.0, 8256.0, 3418.0, 3313.0, 4716.0, 4010~
## $ Total_Revolving_Bal      <dbl> 777, 864, 0, 2517, 0, 1247, 2264, 1396, 2517,~
## $ Avg_Open_To_Buy          <dbl> 11914.0, 7392.0, 3418.0, 796.0, 4716.0, 2763.~
## $ Total_Amt_Chng_Q4_Q1     <dbl> 1.335, 1.541, 2.594, 1.405, 2.175, 1.376, 1.9~
## $ Total_Trans_Amt          <dbl> 1144, 1291, 1887, 1171, 816, 1088, 1330, 1538~
## $ Total_Trans_Ct           <dbl> 42, 33, 20, 20, 28, 24, 31, 36, 24, 32, 42, 2~
## $ Total_Ct_Chng_Q4_Q1      <dbl> 1.625, 3.714, 2.333, 2.333, 2.500, 0.846, 0.7~
## $ Avg_Utilization_Ratio    <dbl> 0.061, 0.105, 0.000, 0.760, 0.000, 0.311, 0.0~
## $ age_group                <fct> 41-50, 41-50, 51-60, 31-40, 31-40, 41-50, 51-~
## $ inactivity_ratio         <dbl> 0.02564103, 0.02272727, 0.02777778, 0.1176470~
## $ trans_intensity          <dbl> 1.0769231, 0.7500000, 0.5555556, 0.5882353, 1~
## $ Total_Trans_Amt_log      <dbl> 7.043160, 7.163947, 7.543273, 7.066467, 6.705~
## $ Total_Trans_Ct_log       <dbl> 3.761200, 3.526361, 3.044522, 3.044522, 3.367~
## $ Credit_Limit_log         <dbl> 9.448727, 9.018817, 8.137103, 8.105911, 8.458~
```

Task 4 — Aggregation & Final Dataset

Now, we can perform aggregations to derive insights. Let's analyze customer behavior by age group and attrition status.

```
by_age <- df_fe %>%
  group_by(age_group, Attrition_Flag) %>%
  summarize(
    client_count = n(),
    avg_trans = mean(Total_Trans_Ct, na.rm = TRUE),
    avg_spend = mean(Total_Trans_Amt, na.rm = TRUE),
    util_med = median(Avg_Utilization_Ratio, na.rm = TRUE),
    .groups = 'drop' # Drop grouping structure after summarizing
  )

head(by_age)
```

```
## # A tibble: 6 x 6
##   age_group Attrition_Flag    client_count avg_trans avg_spend util_med
##   <fct>     <fct>                    <int>     <dbl>     <dbl>    <dbl>
## 1 <=30      Attrited Customer           32      53.4     5215.    0
## 2 <=30      Existing Customer          233      64.4     4314.    0.252
## 3 31-40     Attrited Customer          310      46.0     3282.    0
## 4 31-40     Existing Customer         1822      67.7     4489.    0.243
## 5 41-50     Attrited Customer          779      45.2     3023.    0
## 6 41-50     Existing Customer         3873      71.7     4964.    0.179
```

Let's also calculate the churn rate for each card category.

```r
churn_by_card <- df_fe %>%
  group_by(Card_Category, Attrition_Flag) %>%
  summarize(n = n(), .groups = 'drop') %>%
  group_by(Card_Category) %>%
  mutate(churn_rate = n / sum(n)) %>%
  select(-n) %>%
  pivot_wider(names_from = Attrition_Flag, values_from = churn_rate, values_fill = 0)

head(churn_by_card)
```

```
## # A tibble: 4 x 3
## # Groups:   Card_Category [4]
##   Card_Category `Attrited Customer` `Existing Customer`
##   <fct>                       <dbl>               <dbl>
## 1 Blue                        0.161               0.839
## 2 Gold                        0.181               0.819
## 3 Platinum                    0.25                0.75
## 4 Silver                      0.148               0.852
```

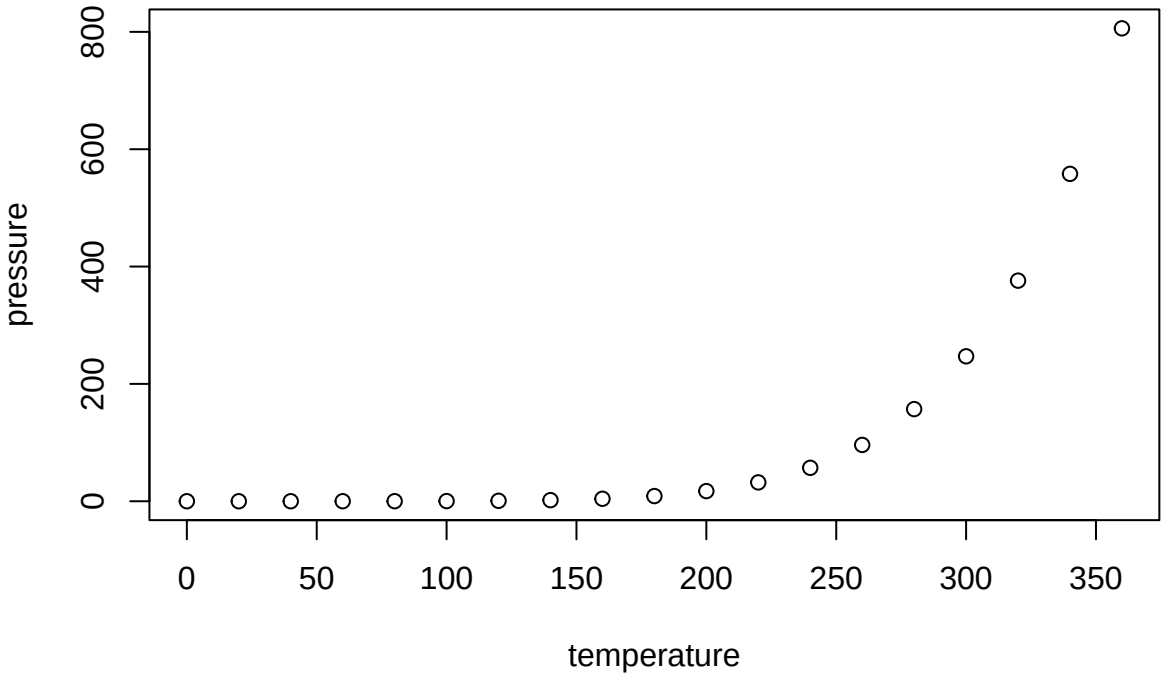
The final dataset is now ready for modeling or further analysis.

```r
# The final dataset is df_fe.
# If we were to save it for modeling, we might drop identifier columns.
final_df <- df_fe %>%
  select(-CLIENTNUM, -Avg_Open_To_Buy) # Drop ID and redundant column

# final_df %>% write_csv("bankchurn_final_clean_r.csv")

# Final shape of the dataset
dim(final_df)
```

```
## [1] 10127    25
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.