Global Superstore: Customer Insight Portfolio

BSAN 6050, Fall 2024, Professor Sijun Wang

Lauryn Carter, Xiomara Vidal Marquez, Luis Otero

12/13/2024

Load Libraries

```
library(dplyr)
library(ggplot2)
library(tidyverse)
library(ggthemes)
library(ggridges)
library(moments)
library(gridExtra)
library(beanplot)
library(car)
library(survival)
library(survminer)
library(geepack)
library(pseudo)
library(corrplot)
library(reshape2)
library(caret)
library(plotROC)
library(htmltools)
library(glmnet)
library(ROCR)
library(multcomp)
library(arules)
library(arulesViz)
library(pROC)
library(reshape2)
library(stringr)
```

Data Exploration

Read Data

```
#load global superstore data
data_df <- read.csv(here::here("Data", "group.csv"))
```

Data Structure

```
#column names
colnames(data_df)
```

```
##  [1] "Row.ID"         "Order.ID"       "Order.Date"     "Ship.Date"
```

```
##  [5] "Ship.Mode"      "Customer.ID"    "Customer.Name"  "Segment"
##  [9] "City"           "State"          "Country"        "Postal.Code"
## [13] "Market"         "Region"         "Product.ID"     "Category"
## [17] "Sub.Category"   "Product.Name"   "Sales"          "Quantity"
## [21] "Discount"       "Profit"         "Shipping.Cost"  "Order.Priority"
```

```
#data types and dimensions
str(data_df)
```

```
## 'data.frame':    51290 obs. of  24 variables:
##  $ Row.ID        : int  42433 22253 48883 11731 22255 22254 21613 34662 44508 23688 ...
##  $ Order.ID      : chr  "AG-2011-2040" "IN-2011-47883" "HU-2011-1220" "IT-2011-3647632" ...
##  $ Order.Date    : chr  "1/1/2011" "1/1/2011" "1/1/2011" "1/1/2011" ...
##  $ Ship.Date     : chr  "6/1/2011" "8/1/2011" "5/1/2011" "5/1/2011" ...
##  $ Ship.Mode     : chr  "Standard Class" "Standard Class" "Second Class" "Second Class" ...
##  $ Customer.ID   : chr  "TB-11280" "JH-15985" "AT-735" "EM-14140" ...
##  $ Customer.Name : chr  "Toby Braunhardt" "Joseph Holt" "Annie Thurman" "Eugene Moren" ...
##  $ Segment       : chr  "Consumer" "Consumer" "Consumer" "Home Office" ...
##  $ City          : chr  "Constantine" "Wagga Wagga" "Budapest" "Stockholm" ...
##  $ State         : chr  "Constantine" "New South Wales" "Budapest" "Stockholm" ...
##  $ Country       : chr  "Algeria" "Australia" "Hungary" "Sweden" ...
##  $ Postal.Code   : int  NA NA NA NA NA NA NA 92691 NA NA ...
##  $ Market        : chr  "Africa" "APAC" "EMEA" "EU" ...
##  $ Region        : chr  "Africa" "Oceania" "EMEA" "North" ...
##  $ Product.ID    : chr  "OFF-TEN-10000025" "OFF-SU-10000618" "OFF-TEN-10001585" "OFF-PA-10001492" ..
##  $ Category      : chr  "Office Supplies" "Office Supplies" "Office Supplies" "Office Supplies" ...
##  $ Sub.Category  : chr  "Storage" "Supplies" "Storage" "Paper" ...
##  $ Product.Name  : chr  "Tenex Lockers, Blue" "Acme Trimmer, High Speed" "Tenex Box, Single Width" "I
##  $ Sales         : num  408.3 120.4 66.1 44.9 113.7 ...
##  $ Quantity      : int  2 3 4 3 5 2 2 2 1 3 ...
##  $ Discount      : num  0 0.1 0 0.5 0.1 0.1 0 0.15 0 0 ...
##  $ Profit        : num  106.1 36 29.6 -26.1 37.8 ...
##  $ Shipping.Cost : num  35.46 9.72 8.17 4.82 4.7 ...
##  $ Order.Priority: chr  "Medium" "Medium" "High" "High" ...
```

We can see that there is a total of 51290 observations and 24 variables in the dataset. The majority of the variables are categorical with a few continuous columns.

```
#first 6 rows of data
head(data_df)
```

```
##   Row.ID        Order.ID Order.Date Ship.Date      Ship.Mode Customer.ID
## 1  42433    AG-2011-2040   1/1/2011  6/1/2011 Standard Class    TB-11280
## 2  22253   IN-2011-47883   1/1/2011  8/1/2011 Standard Class    JH-15985
## 3  48883    HU-2011-1220   1/1/2011  5/1/2011   Second Class      AT-735
## 4  11731 IT-2011-3647632   1/1/2011  5/1/2011   Second Class    EM-14140
## 5  22255   IN-2011-47883   1/1/2011  8/1/2011 Standard Class    JH-15985
## 6  22254   IN-2011-47883   1/1/2011  8/1/2011 Standard Class    JH-15985
##     Customer.Name     Segment        City           State   Country Postal.Code
## 1 Toby Braunhardt    Consumer Constantine     Constantine   Algeria          NA
## 2     Joseph Holt    Consumer Wagga Wagga New South Wales Australia          NA
## 3   Annie Thurman    Consumer    Budapest        Budapest   Hungary          NA
## 4    Eugene Moren Home Office   Stockholm       Stockholm    Sweden          NA
## 5     Joseph Holt    Consumer Wagga Wagga New South Wales Australia          NA
## 6     Joseph Holt    Consumer Wagga Wagga New South Wales Australia          NA
##   Market  Region       Product.ID        Category Sub.Category
```

```
## 1 Africa  Africa OFF-TEN-10000025 Office Supplies      Storage
## 2   APAC Oceania  OFF-SU-10000618 Office Supplies     Supplies
## 3   EMEA    EMEA OFF-TEN-10001585 Office Supplies      Storage
## 4     EU   North  OFF-PA-10001492 Office Supplies        Paper
## 5   APAC Oceania  FUR-FU-10003447       Furniture  Furnishings
## 6   APAC Oceania  OFF-PA-10001968 Office Supplies        Paper
##                             Product.Name   Sales Quantity Discount  Profit
## 1                    Tenex Lockers, Blue 408.300        2      0.0 106.140
## 2                Acme Trimmer, High Speed 120.366        3      0.1  36.036
## 3                 Tenex Box, Single Width  66.120        4      0.0  29.640
## 4             Enermax Note Cards, Premium  44.865        3      0.5 -26.055
## 5              Eldon Light Bulb, Duo Pack 113.670        5      0.1  37.770
## 6 Eaton Computer Printout Paper, 8.5 x 11  55.242        2      0.1  15.342
##   Shipping.Cost Order.Priority
## 1         35.46         Medium
## 2          9.72         Medium
## 3          8.17           High
## 4          4.82           High
## 5          4.70         Medium
## 6          1.80         Medium
```

Data Preprocessing

Formatting Columns

```r
#copy data
new_data_df <- data_df

#formatting Order.Date column
a <- as.Date(new_data_df$Order.Date,
             format = "%m/%d/%Y")

b <-as.Date(new_data_df$Order.Date,
            format = "%d-%m-%Y")

#filling in NAs
a[is.na(a)] <- b[!is.na(b)]
new_data_df$Order.Date <- a

#formatting Ship.Date column
a <- as.Date(new_data_df$Ship.Date,
             format = "%m/%d/%Y")

b <- as.Date(new_data_df$Ship.Date,
             format = "%d-%m-%Y")

#filling in NAs
a[is.na(a)] <- b[!is.na(b)]
new_data_df$Ship.Date <- a

#formatting chr columns as factors
new_data_df2 <- new_data_df %>% mutate_if(is.character, as.factor)
```

Check Missing Values and Duplicate Rows

```
#check for missing values
sum(is.na(new_data_df2$Postal.Code))
```

```
## [1] 41296
```

```
#check for duplicate rows
sum(duplicated(new_data_df2))
```

```
## [1] 0
```

```
#remove Postal.Code and Row.ID columns
new_data_df2 <- new_data_df2[, -which(names(new_data_df2) %in% c("Postal.Code", "Row.ID"))]
```

Adding Columns

```
#adding datetime columns
new_data_df2$dayofweek <- weekdays(new_data_df2$Order.Date)
new_data_df2$year <- substring(new_data_df2$Order.Date,1,4)
new_data_df2$month <- substring(new_data_df2$Order.Date, 6, 7)
new_data_df2$day <- substring(new_data_df2$Order.Date, 9, 10)
new_data_df2$quarter <- quarter(new_data_df2$Order.Date)

#cost column
new_data_df2 <- new_data_df2 %>% mutate(cost = Sales - Profit)

#cost per unit
new_data_df2$cost_per_unit <- new_data_df2$Sales/new_data_df2$Quantity

#profit per unit
new_data_df2$profit_per_unit <- new_data_df2$Profit/new_data_df2$Quantity

#difference between order and ship date
new_data_df2$ship_delay <- as.numeric(difftime(new_data_df2$Ship.Date,
                                               new_data_df2$Order.Date, units = "days"))
```

Customer Database

```
#mode function
get_mode <- function(x){
  ux <- unique(x)
  mode_value <- ux[which.max(tabulate(match(x, ux)))]
  return(as.character(mode_value))
}

#drop columns Order.ID and Product.ID
customer_df <- new_data_df2[, -which(names(new_data_df2) %in%
                                       c("Order.ID", "Product.ID"))]

#generate customer database
customer_df <- customer_df %>% group_by(Customer.ID) %>%
  summarize(customer_name = get_mode(Customer.Name),
            first_order_date = min(Order.Date),
            last_order_date = max(Order.Date),
```

```
            total_sales = sum(Sales),
            avg_sales = mean(Sales),
            avg_quantity = mean(Quantity),
            avg_shipping_cost = mean(Shipping.Cost),
            avg_discount = mean(Discount),
            total_profit = sum(Profit),
            total_cost = sum(cost),
            avg_cost_per_unit = mean(cost_per_unit),
            avg_profit_per_unit = mean(profit_per_unit),
            avg_ship_delay = mean(ship_delay),
            followtime = as.numeric(difftime(max(Order.Date), min(Order.Date), units = "days")),
            ship_mode = get_mode(Ship.Mode),
            segment_mode = get_mode(Segment),
            city_mode = get_mode(City),
            state_mode = get_mode(State),
            country_mode = get_mode(Country),
            market_mode = get_mode(Market),
            region_mode = get_mode(Region),
            category_mode = get_mode(Category),
            subcategory_mode = get_mode(Sub.Category),
            product_mode = get_mode(Product.Name),
            order_priority_mode = get_mode(Order.Priority),
            dayofweek_mode = get_mode(dayofweek),
            year_mode = get_mode(year),
            month_mode = get_mode(month),
            quarter_mode = get_mode(quarter),
            total_orders = n())
```

RFM Analysis

```
#RFM
rfm_df <- customer_df

#frequency: amount of purchases
rfm_df <- rfm_df %>% rename(frequency = total_orders)

#recency: time difference between today and last purchase date
date = as.Date("2024-11-01")
rfm_df <- rfm_df %>% mutate(recency = as.numeric(difftime(date,
                                                          last_order_date,
                                                          units = "days")))

#monetary: profitability of customer
rfm_df <- rfm_df %>% rename(monetary = total_profit)

#frequency index
summary(rfm_df$frequency)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   12.00   28.00   32.26   52.00   97.00
```

```
rfm_df <- rfm_df %>%
  mutate(freq_index = cut(frequency,
                          breaks = c(0.5, 12, 28, 52, 98),
```

```
                          labels = c("1","2","3","4")))

#recency index
summary(rfm_df$recency)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3593    3607    3633    3681    3696    4711
```

```
rfm_df <- rfm_df %>% mutate(rec_index =
                              cut(recency,
                                  breaks = c(3592, 3607, 3633, 3696, 4712),
                                  labels = c("4","3","2","1")))

#monetary index
summary(rfm_df$monetary)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -6437.37    94.81   591.36   922.93  1614.49  8787.48
```

```
rfm_df <- rfm_df %>% mutate(mon_index = cut(
  monetary,
  breaks = c(-6438, 94.81, 591.36, 1614.49, 8788),
  labels = c("1", "2", "3", "4")
))

#Customer Lifetime Value
rfm_df <- rfm_df %>% mutate(CLV = (20 * as.numeric(freq_index) / 5) +
                              (40 * as.numeric(rec_index) / 5) +
                              (40 * as.numeric(mon_index) / 5))

#CLV bins
rfm_df <- rfm_df %>% mutate(clv_bin = cut(
  CLV,
  breaks = c(0, 20, 40, 60, 80, 100),
  labels = c("Low", "Medium-Low ", "Medium", "Medium-High", "High")
))

#final data
final_df <- rfm_df
```

```
#convert columns to factor
final_df$ship_mode <- as.factor(final_df$ship_mode)
final_df$segment_mode <- as.factor(final_df$segment_mode)
final_df$city_mode <- as.factor(final_df$city_mode)
final_df$state_mode <- as.factor(final_df$state_mode)
final_df$country_mode <- as.factor(final_df$country_mode)
final_df$market_mode <- as.factor(final_df$market_mode)
final_df$region_mode <- as.factor(final_df$region_mode)
final_df$category_mode <- as.factor(final_df$category_mode)
final_df$subcategory_mode <- as.factor(final_df$subcategory_mode)
final_df$product_mode <- as.factor(final_df$product_mode)
final_df$order_priority_mode <- as.factor(final_df$order_priority_mode)
final_df$dayofweek_mode <- as.factor(final_df$dayofweek_mode)
final_df$year_mode <- as.factor(final_df$year_mode)
final_df$month_mode <- as.factor(final_df$month_mode)
```

```r
final_df$quarter_mode <- as.factor(final_df$quarter_mode)

#fefine churn, based on a 10-year (3650-day) inactivity period
final_df$churn <- ifelse(final_df$recency > 3700, 1, 0)

#create threshold of high-value customers (top 25% of total sales)
final_df$high_value <- ifelse(final_df$total_sales > 13133, 1, 0)
```

Descriptive and Summary Statistics

```r
#first 6 rows of data
head(final_df)
```

```
## # A tibble: 6 x 39
##   Customer.ID customer_name first_order_date last_order_date total_sales
##   <fct>       <chr>         <date>           <date>                <dbl>
## 1 AA-10315    Alex Avila    2011-03-31       2014-12-23           13747.
## 2 AA-10375    Allen Armold  2011-04-21       2014-12-25            5884.
## 3 AA-10480    Andrew Allen  2011-01-11       2014-09-05           17696.
## 4 AA-10645    Anna Andreadi 2011-01-12       2014-12-05           15344.
## 5 AA-315      Alex Avila    2011-08-06       2014-12-29            2243.
## 6 AA-375      Allen Armold  2011-01-06       2014-07-03             654.
## # i 34 more variables: avg_sales <dbl>, avg_quantity <dbl>,
## #   avg_shipping_cost <dbl>, avg_discount <dbl>, monetary <dbl>,
## #   total_cost <dbl>, avg_cost_per_unit <dbl>, avg_profit_per_unit <dbl>,
## #   avg_ship_delay <dbl>, followtime <dbl>, ship_mode <fct>,
## #   segment_mode <fct>, city_mode <fct>, state_mode <fct>, country_mode <fct>,
## #   market_mode <fct>, region_mode <fct>, category_mode <fct>,
## #   subcategory_mode <fct>, product_mode <fct>, order_priority_mode <fct>, ...
```

```r
#data types and dimensions
str(final_df)
```

```
## tibble [1,590 x 39] (S3: tbl_df/tbl/data.frame)
##  $ Customer.ID        : Factor w/ 1590 levels "AA-10315","AA-10375",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ customer_name      : chr [1:1590] "Alex Avila" "Allen Armold" "Andrew Allen" "Anna Andreadi" ...
##  $ first_order_date   : Date[1:1590], format: "2011-03-31" "2011-04-21" ...
##  $ last_order_date    : Date[1:1590], format: "2014-12-23" "2014-12-25" ...
##  $ total_sales        : num [1:1590] 13747 5884 17696 15344 2243 ...
##  $ avg_sales          : num [1:1590] 327 140 466 210 280 ...
##  $ avg_quantity       : num [1:1590] 3.45 3.31 3.95 3.66 2.5 ...
##  $ avg_shipping_cost  : num [1:1590] 29.4 21.5 43 24 27 ...
##  $ avg_discount       : num [1:1590] 0.1036 0.1667 0.0785 0.1259 0.225 ...
##  $ monetary           : num [1:1590] 448 677 1516 3051 536 ...
##  $ total_cost         : num [1:1590] 13300 5207 16179 12292 1708 ...
##  $ avg_cost_per_unit  : num [1:1590] 97.9 48.3 112.3 52.1 114.5 ...
##  $ avg_profit_per_unit: num [1:1590] 4.62 6.41 14 9.27 4.8 ...
##  $ avg_ship_delay     : num [1:1590] 60.6 -8.93 -11.21 28.59 5.25 ...
##  $ followtime         : num [1:1590] 1363 1344 1333 1423 1241 ...
##  $ ship_mode          : Factor w/ 4 levels "First Class",..: 4 4 4 4 4 4 4 4 4 1 ...
##  $ segment_mode       : Factor w/ 3 levels "Consumer","Corporate",..: 1 1 1 1 1 1 1 1 1 3 ...
##  $ city_mode          : Factor w/ 779 levels "Aachen","Aba",..: 225 310 642 296 189 383 356 494 401 4
##  $ state_mode         : Factor w/ 491 levels "'Amman","Abia",..: 145 219 300 210 137 260 73 25 370 3
##  $ country_mode       : Factor w/ 93 levels "Albania","Algeria",..: 87 88 88 88 82 63 3 61 5 88 ...
```

```
##  $ market_mode        : Factor w/ 7 levels "Africa","APAC",..: 5 7 7 2 4 4 1 1 2 6 ...
##  $ region_mode        : Factor w/ 13 levels "Africa","Canada",..: 8 11 4 12 7 7 1 1 4 4 ...
##  $ category_mode      : Factor w/ 3 levels "Furniture","Office Supplies",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ subcategory_mode   : Factor w/ 17 levels "Accessories",..: 4 15 15 15 15 8 3 4 15 13 ...
##  $ product_mode       : Factor w/ 1067 levels "12-1/2 Diameter Round Wall Clock",..: 433 597 803 142
##  $ order_priority_mode: Factor w/ 4 levels "Critical","High",..: 2 2 4 4 4 4 4 4 4 2 ...
##  $ dayofweek_mode     : Factor w/ 7 levels "Friday","Monday",..: 5 1 5 7 5 5 6 5 6 7 ...
##  $ year_mode          : Factor w/ 4 levels "2011","2012",..: 4 3 2 3 3 4 1 2 3 3 ...
##  $ month_mode         : Factor w/ 12 levels "01","02","03",..: 4 12 8 5 8 1 6 9 6 6 ...
##  $ quarter_mode       : Factor w/ 4 levels "1","2","3","4": 2 4 2 2 3 2 2 3 4 2 ...
##  $ frequency          : int [1:1590] 42 42 38 73 8 13 10 18 77 57 ...
##  $ recency            : num [1:1590] 3601 3599 3710 3619 3595 ...
##  $ freq_index         : Factor w/ 4 levels "1","2","3","4": 3 3 3 4 1 2 1 2 4 4 ...
##  $ rec_index          : Factor w/ 4 levels "4","3","2","1": 1 1 4 2 1 4 4 3 2 2 ...
##  $ mon_index          : Factor w/ 4 levels "1","2","3","4": 2 3 3 4 2 1 2 1 4 4 ...
##  $ CLV                : num [1:1590] 36 44 68 64 28 48 52 40 64 64 ...
##  $ clv_bin            : Factor w/ 5 levels "Low","Medium-Low ",..: 2 3 4 4 2 3 3 2 4 4 ...
##  $ churn              : num [1:1590] 0 0 1 0 0 1 1 0 0 0 ...
##  $ high_value         : num [1:1590] 1 0 1 1 0 0 0 0 1 1 ...
```

```
#column names
colnames(final_df)
```

```
##  [1] "Customer.ID"         "customer_name"       "first_order_date"
##  [4] "last_order_date"     "total_sales"         "avg_sales"
##  [7] "avg_quantity"        "avg_shipping_cost"   "avg_discount"
## [10] "monetary"            "total_cost"          "avg_cost_per_unit"
## [13] "avg_profit_per_unit" "avg_ship_delay"      "followtime"
## [16] "ship_mode"           "segment_mode"        "city_mode"
## [19] "state_mode"          "country_mode"        "market_mode"
## [22] "region_mode"         "category_mode"       "subcategory_mode"
## [25] "product_mode"        "order_priority_mode" "dayofweek_mode"
## [28] "year_mode"           "month_mode"          "quarter_mode"
## [31] "frequency"           "recency"             "freq_index"
## [34] "rec_index"           "mon_index"           "CLV"
## [37] "clv_bin"             "churn"               "high_value"
```

```
#summary statistics
summary(final_df)
```

```
##    Customer.ID   customer_name      first_order_date     last_order_date
##  AA-10315:   1   Length:1590        Min.   :2011-01-01   Min.   :2011-12-09
##  AA-10375:   1   Class :character   1st Qu.:2011-02-21   1st Qu.:2014-09-19
##  AA-10480:   1   Mode  :character   Median :2011-05-15   Median :2014-11-21
##  AA-10645:   1                      Mean   :2011-08-02   Mean   :2014-10-04
##  AA-315  :   1                      3rd Qu.:2011-10-04   3rd Qu.:2014-12-17
##  AA-375  :   1                      Max.   :2014-10-21   Max.   :2014-12-31
##  (Other) :1584
##   total_sales         avg_sales        avg_quantity    avg_shipping_cost
##  Min.   :    7.17   Min.   :  7.173   Min.   : 1.000   Min.   :  0.85
##  1st Qu.: 1674.81   1st Qu.:134.583   1st Qu.: 2.222   1st Qu.: 13.95
##  Median : 6248.14   Median :214.507   Median : 3.311   Median : 22.50
##  Mean   : 7951.26   Mean   :214.915   Mean   : 3.032   Mean   : 23.23
##  3rd Qu.:13133.10   3rd Qu.:277.118   3rd Qu.: 3.778   3rd Qu.: 30.21
##  Max.   :35668.12   Max.   :902.748   Max.   :10.000   Max.   :131.44
##
```

```
##   avg_discount        monetary          total_cost       avg_cost_per_unit
##  Min.   :0.00000   Min.   :-6437.37   Min.   :   22.74   Min.   :  7.173
##  1st Qu.:0.09092   1st Qu.:   94.81   1st Qu.: 1551.77   1st Qu.: 55.357
##  Median :0.13674   Median :  591.36   Median : 5376.72   Median : 69.100
##  Mean   :0.15443   Mean   :  922.93   Mean   : 7028.33   Mean   : 72.720
##  3rd Qu.:0.19239   3rd Qu.: 1614.49   3rd Qu.:11576.89   3rd Qu.: 84.897
##  Max.   :0.70000   Max.   : 8787.48   Max.   :32208.97   Max.   :301.710
##
##  avg_profit_per_unit avg_ship_delay       followtime           ship_mode
##  Min.   :-129.153    Min.   :-215.000   Min.   :   0   First Class   :  71
##  1st Qu.:   2.488    1st Qu.: -13.727   1st Qu.:1049   Same Day      :  25
##  Median :   7.998    Median :   6.697   Median :1262   Second Class  : 131
##  Mean   :   7.467    Mean   :   4.814   Mean   :1159   Standard Class:1363
##  3rd Qu.:  13.829    3rd Qu.:  26.279   3rd Qu.:1366
##  Max.   :  81.645    Max.   : 214.000   Max.   :1459
##
##       segment_mode         city_mode         state_mode          country_mode
##  Consumer   :818   New York City:  34   California:  95   United States:580
##  Corporate  :476   Istanbul     :  32   England   :  49   Turkey       :148
##  Home Office:296   Lagos        :  32   Istanbul  :  43   Nigeria      : 96
##                    Los Angeles  :  27   New York  :  37   Iran         : 46
##                    Cairo        :  18   Lagos     :  32   Australia    : 39
##                    Philadelphia :  15   Texas     :  26   Morocco      : 37
##                    (Other)      :1432   (Other)   :1308   (Other)      :644
##  market_mode          region_mode          category_mode  subcategory_mode
##  Africa:362   Central       :495   Furniture      :  22   Binders:445
##  APAC  :225   EMEA          :427   Office Supplies:1528   Storage:287
##  Canada:  6   Africa        :362   Technology     :  40   Art    :258
##  EMEA  :427   South         :146                          Phones : 89
##  EU    :174   North         : 56                          Chairs : 86
##  LATAM :204   Southeast Asia: 22                          Paper  : 80
##  US    :192   (Other)       : 82                          (Other):345
##                                    product_mode  order_priority_mode
##  Staples                                 :  31   Critical:  31
##  Eldon File Cart, Single Width           :   5   High    : 301
##  Fellowes Lockers, Single Width          :   5   Low     :  11
##  Rogers Lockers, Single Width            :   5   Medium  :1247
##  Sanford Pencil Sharpener, Water Color:   5
##  Smead File Cart, Single Width           :   5
##  (Other)                                 :1534
##    dayofweek_mode year_mode    month_mode  quarter_mode   frequency
##  Friday   :293    2011:184   12     :214   1:230        Min.   : 1.00
##  Monday   :298    2012:261   11     :201   2:357        1st Qu.:12.00
##  Saturday :137    2013:450   09     :185   3:451        Median :28.00
##  Sunday   : 55    2014:695   06     :159   4:552        Mean   :32.26
##  Thursday :271               08     :146                3rd Qu.:52.00
##  Tuesday  :281               10     :113                Max.   :97.00
##  Wednesday:255               (Other):572
##     recency     freq_index rec_index mon_index      CLV
##  Min.   :3593   1:431      4:418     1:398     Min.   :20.00
##  1st Qu.:3607   2:369      3:378     2:397     1st Qu.:44.00
##  Median :3633   3:414      2:398     3:397     Median :52.00
##  Mean   :3681   4:376      1:396     4:398     Mean   :49.73
##  3rd Qu.:3696                                  3rd Qu.:56.00
```

```
##  Max.   :4711                                   Max.   :80.00
##
##         clv_bin        churn          high_value
##  Low        : 16   Min.   :0.0000   Min.   :0.0000
##  Medium-Low :341   1st Qu.:0.0000   1st Qu.:0.0000
##  Medium     :986   Median :0.0000   Median :0.0000
##  Medium-High:247   Mean   :0.2352   Mean   :0.2503
##  High       :  0   3rd Qu.:0.0000   3rd Qu.:0.7500
##                    Max.   :1.0000   Max.   :1.0000
##
```

Value Counts

```
#value counts for each categorical variable
final_df %>% group_by(region_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 13 x 2
##    region_mode    total
##    <fct>          <int>
##  1 Central          495
##  2 EMEA             427
##  3 Africa           362
##  4 South            146
##  5 North             56
##  6 Southeast Asia    22
##  7 West              20
##  8 Oceania           19
##  9 North Asia        13
## 10 East              10
## 11 Central Asia       8
## 12 Canada             6
## 13 Caribbean          6
```

```
final_df %>% group_by(category_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 3 x 2
##   category_mode   total
##   <fct>           <int>
## 1 Office Supplies  1528
## 2 Technology         40
## 3 Furniture          22
```

```
final_df %>% group_by(segment_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 3 x 2
##   segment_mode total
##   <fct>        <int>
## 1 Consumer       818
## 2 Corporate      476
## 3 Home Office    296
```

```
final_df %>% group_by(ship_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 4 x 2
##   ship_mode      total
##   <fct>          <int>
## 1 Standard Class  1363
## 2 Second Class     131
## 3 First Class       71
## 4 Same Day          25
```

```
final_df %>% group_by(market_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 7 x 2
##   market_mode total
##   <fct>       <int>
## 1 EMEA          427
## 2 Africa        362
## 3 APAC          225
## 4 LATAM         204
## 5 US            192
## 6 EU            174
## 7 Canada          6
```

```
final_df %>% group_by(order_priority_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 4 x 2
##   order_priority_mode total
##   <fct>               <int>
## 1 Medium               1247
## 2 High                  301
## 3 Critical               31
## 4 Low                    11
```

```
final_df %>% group_by(quarter_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 4 x 2
##   quarter_mode total
##   <fct>        <int>
## 1 4              552
## 2 3              451
## 3 2              357
## 4 1              230
```

```
final_df %>% group_by(subcategory_mode) %>%
  summarise(total = n()) %>% arrange(desc(total))
```

```
## # A tibble: 17 x 2
##    subcategory_mode total
##    <fct>            <int>
##  1 Binders            445
##  2 Storage            287
##  3 Art                258
##  4 Phones              89
##  5 Chairs              86
##  6 Paper               80
##  7 Furnishings         66
```

```
##  8 Accessories         56
##  9 Fasteners           35
## 10 Supplies            34
## 11 Copiers             33
## 12 Labels              30
## 13 Envelopes           29
## 14 Bookcases           25
## 15 Machines            19
## 16 Appliances          16
## 17 Tables               2
```

Aggregations

```
#most and least profitable products
product_profit <- final_df %>% group_by(product_mode) %>%
  summarise(total = sum(monetary)) %>% arrange(desc(total))

head(product_profit)
```

```
## # A tibble: 6 x 2
##   product_mode                          total
##   <fct>                                 <dbl>
## 1 Staples                              49272.
## 2 Samsung Smart Phone, Cordless        14656.
## 3 Wilson Jones Binder Covers, Recycled 10000.
## 4 Hewlett Copy Machine, Color           9875.
## 5 Brother Wireless Fax, High-Speed      8787.
## 6 StarTech Calculator, White            8524.
```

```
tail(product_profit)
```

```
## # A tibble: 6 x 2
##   product_mode                       total
##   <fct>                              <dbl>
## 1 Safco Stackable Bookrack, Pine    -2601.
## 2 Nokia Signal Booster, VoIP        -2759.
## 3 12-1/2 Diameter Round Wall Clock  -3700.
## 4 Acco Hole Reinforcements, Clear   -3790.
## 5 Hoover Stove, White               -4796.
## 6 Xerox 1962                        -6437.
```

```
#most and least profitable quarters
quarter_table <- final_df %>% group_by(year_mode, quarter_mode) %>%
  summarise(total_profit = sum(monetary), avg_profit = mean(monetary))
```

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

```
quarter_table %>% arrange(desc(total_profit)) %>% head()
```

```
## # A tibble: 6 x 4
## # Groups:   year_mode [2]
##   year_mode quarter_mode total_profit avg_profit
##   <fct>     <fct>              <dbl>      <dbl>
## 1 2014      4                292740.      1215.
## 2 2014      3                200761.       989.
```

```
## 3 2013      4                183149.      1167.
## 4 2014      2                137510.       955.
## 5 2013      2                113575.      1014.
## 6 2013      3                 99798.       805.
```

```
quarter_table %>% arrange(total_profit) %>% head()
```

```
## # A tibble: 6 x 4
## # Groups:   year_mode [3]
##   year_mode quarter_mode total_profit avg_profit
##   <fct>     <fct>               <dbl>      <dbl>
## 1 2012      1                  20385.       728.
## 2 2011      1                  22920.       603.
## 3 2011      2                  24382.       530.
## 4 2011      3                  27617.       642.
## 5 2013      1                  36234.       636.
## 6 2012      2                  39037.       710.
```

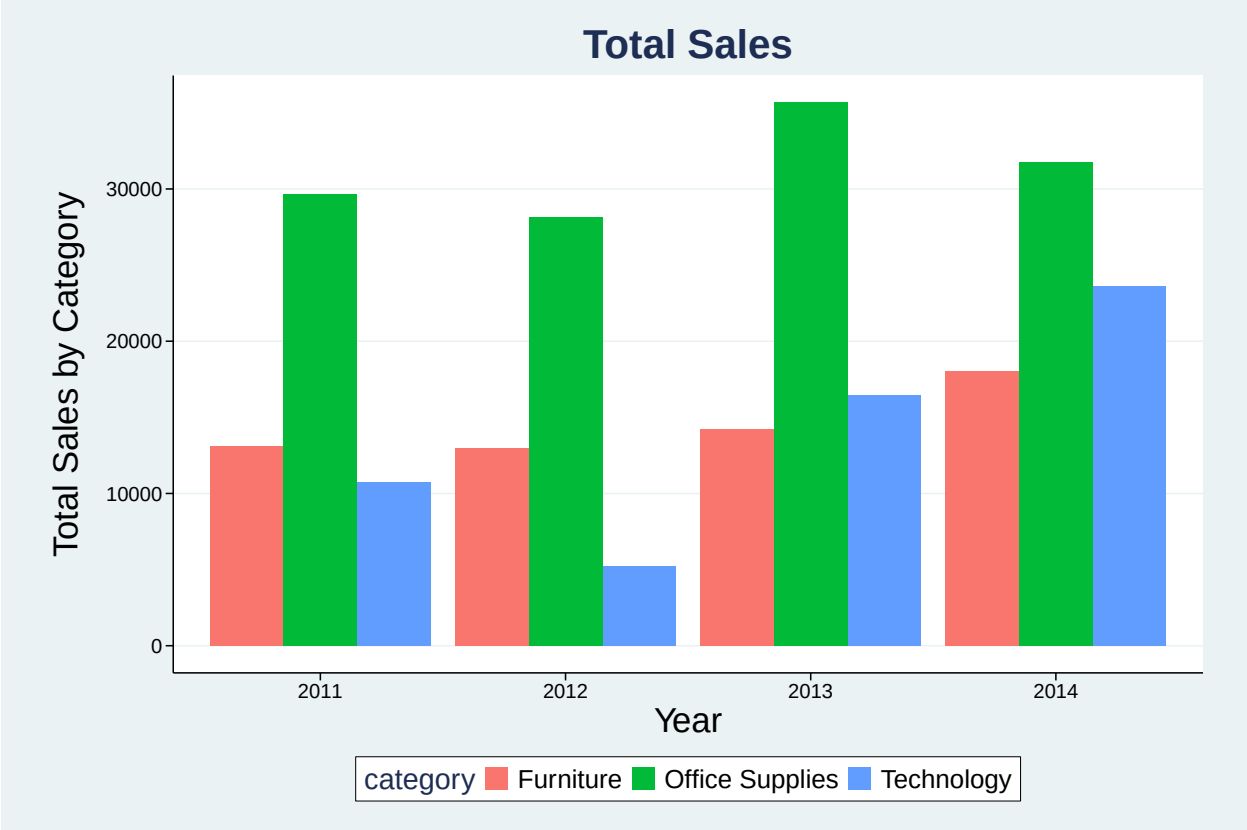
Data Visualization

```
#table of customer sales and profit
customer_group <- final_df %>% group_by(customer_name, year_mode) %>%
  summarise(sales.bycustomer = sum(total_sales),
            profit.bycustomer = sum(monetary),
            month = month_mode,
            day = dayofweek_mode,
            category = category_mode)
```

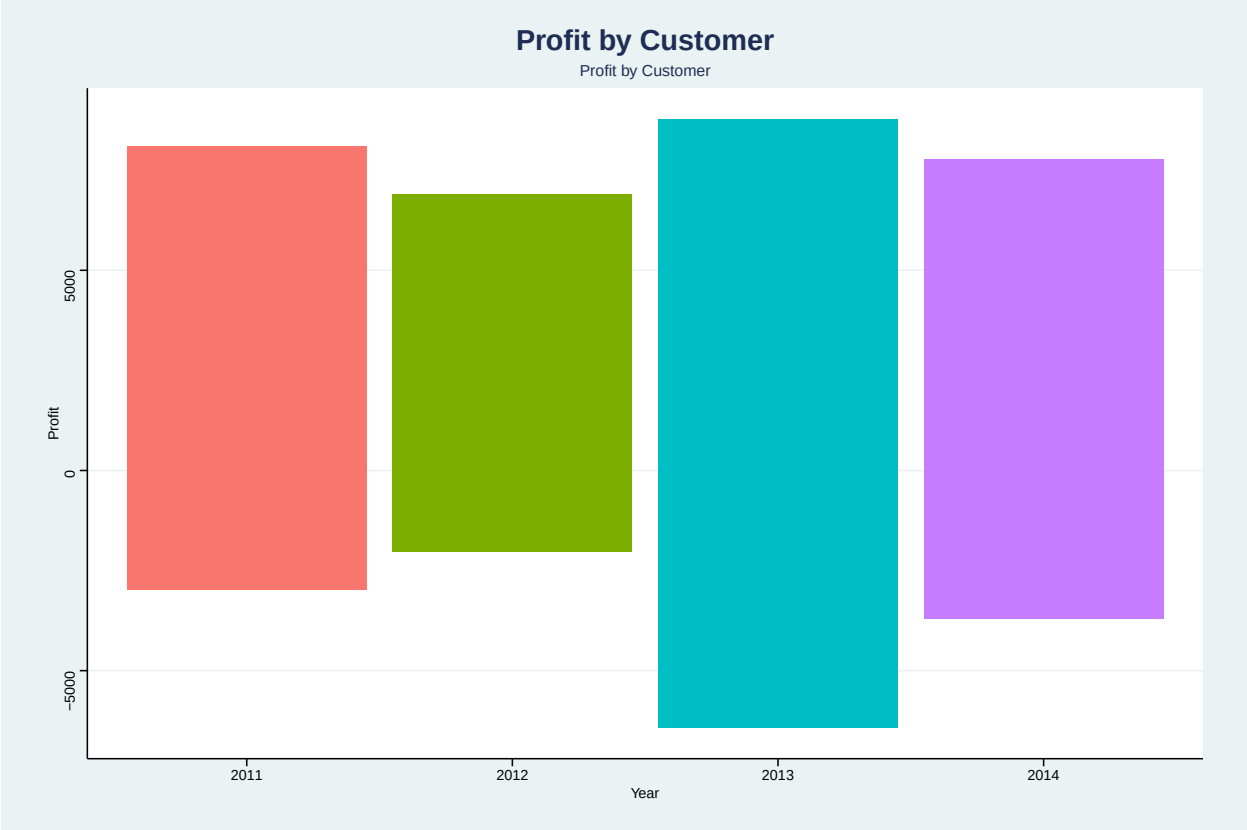
```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

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

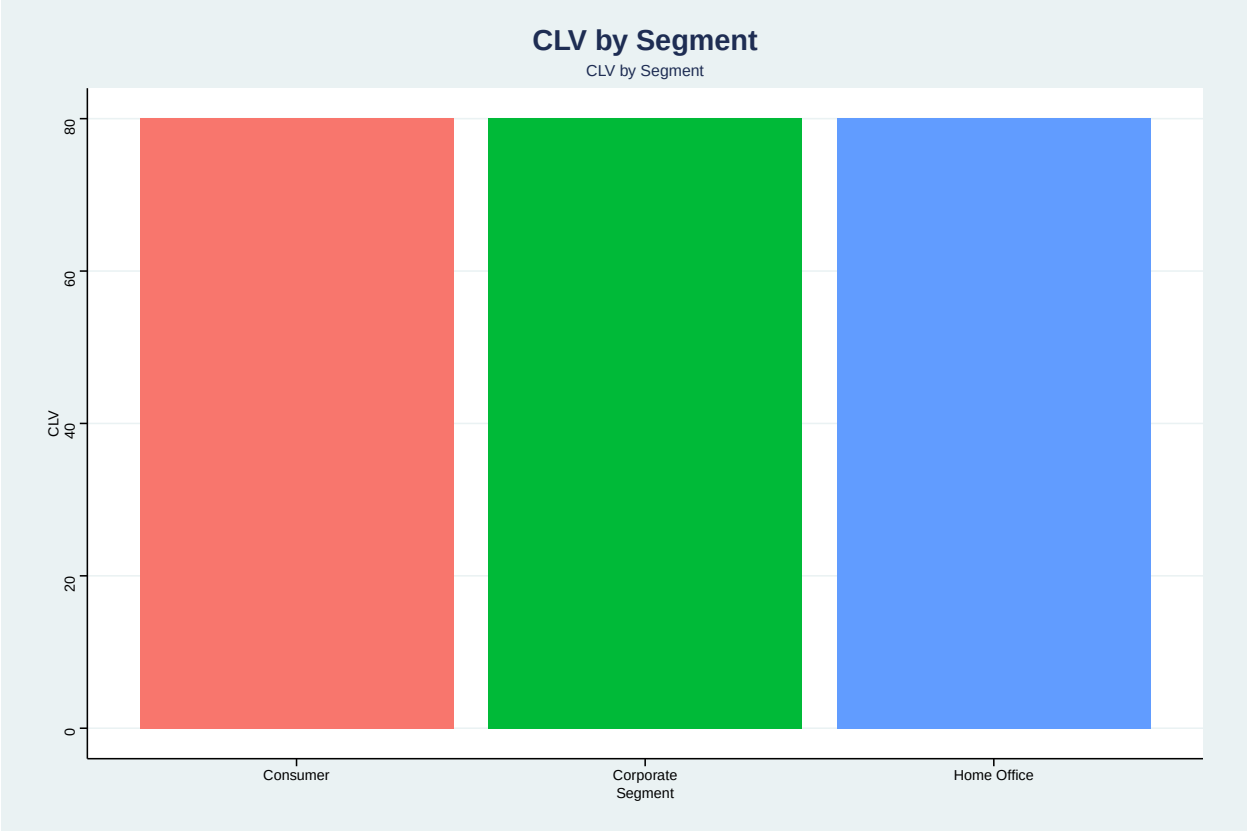
```
#plotting sales by customer
ggplot(customer_group) +
  geom_col(aes(x = year_mode, y = sales.bycustomer, fill = category),
  position = position_dodge()) +
  theme_stata() + labs(title = "Total Sales", x = "Year", y = "Total Sales by Category") +
  theme(plot.title = element_text(size=28, face="bold"),
        axis.text.x = element_text(size = 14),
        axis.text.y = element_text(size = 14, angle = 0),
        axis.title.y = element_text(margin = margin(r = 20)),
        axis.title = element_text(size = 24),
        legend.text = element_text(size = 18),
        legend.title = element_text(size = 20))
```



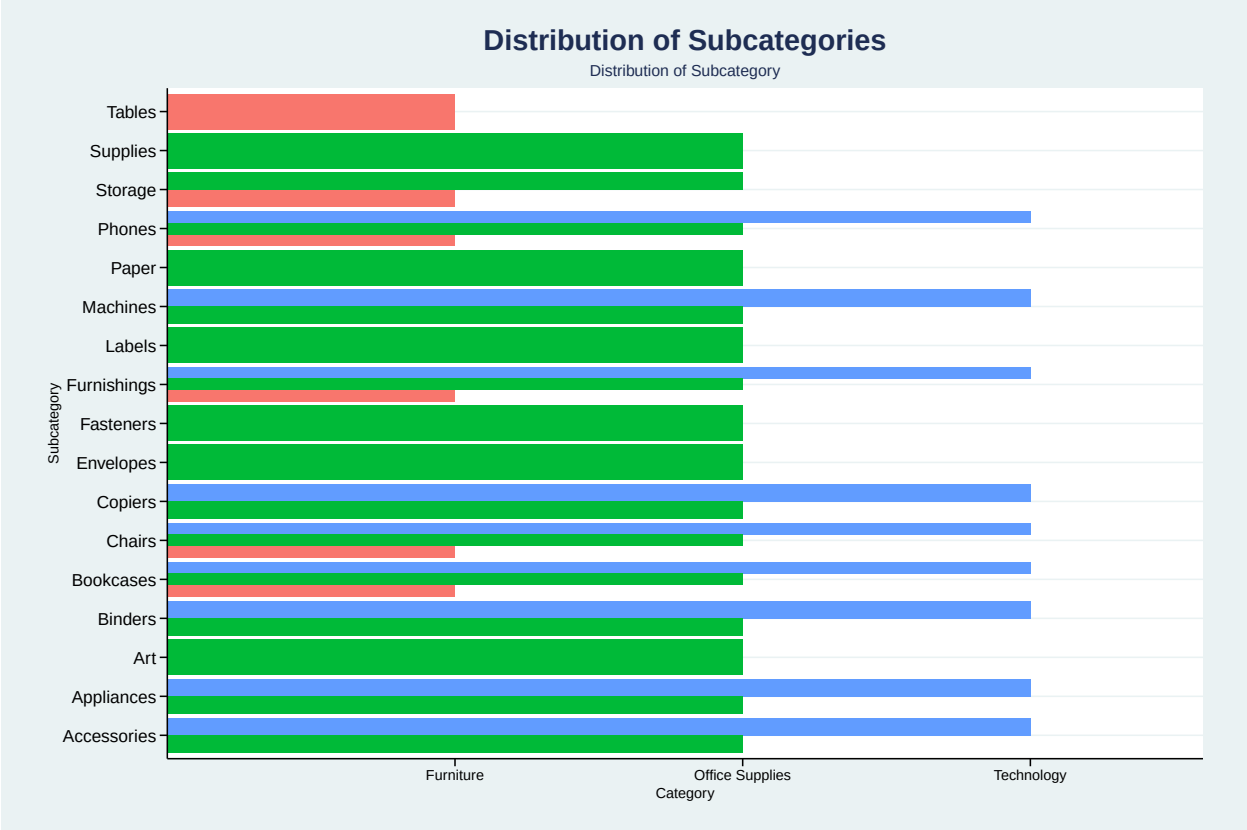

```
#plotting profit by customer
ggplot(customer_group, aes(x = year_mode, y = profit.bycustomer, fill = year_mode)) +
  geom_col(position = position_dodge(), show.legend = FALSE) +
  theme_stata() + labs(title = "Profit by Customer", x = "Year", y = "Profit", subtitle = "Profit by Cus
  theme(plot.title = element_text(size=20, face="bold"))
```



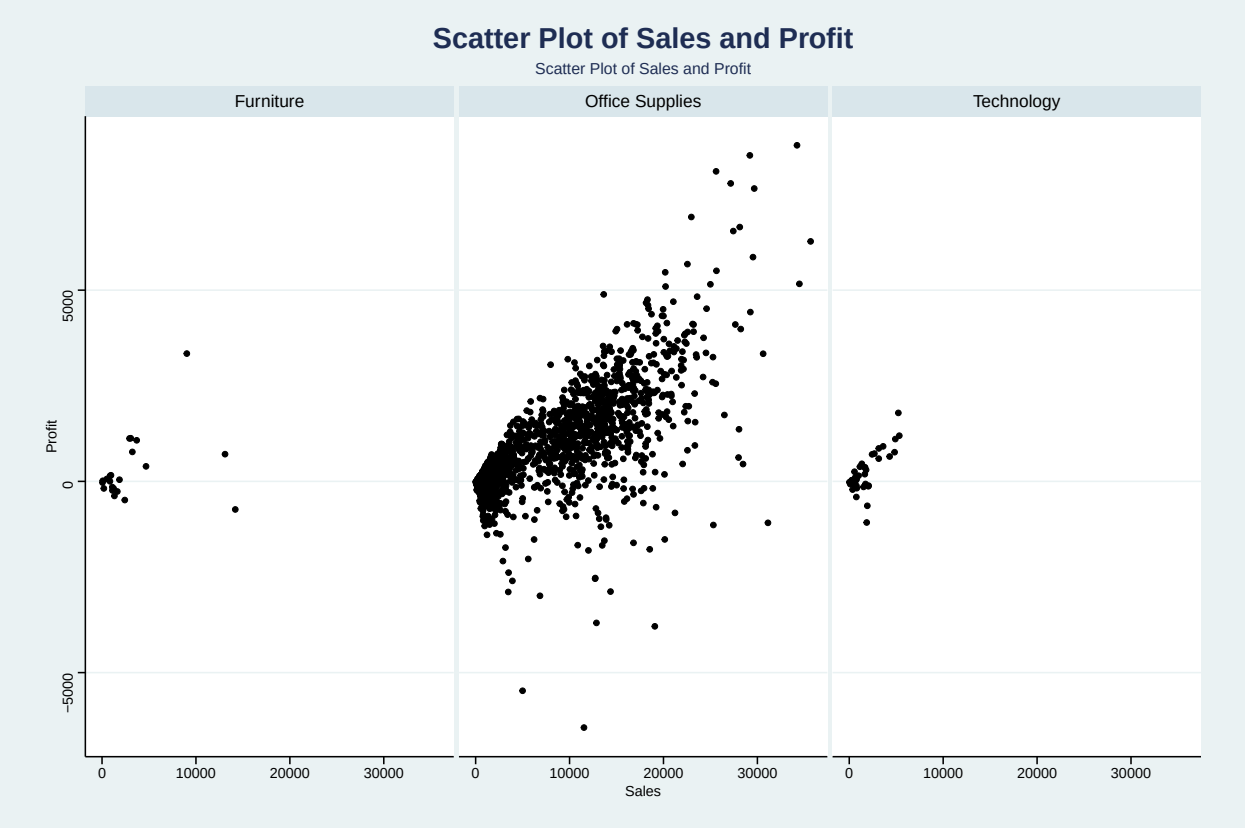

```
#plotting CLV by segment
ggplot(final_df, aes(x = segment_mode, y = CLV, fill = segment_mode)) +
  geom_col(position = position_dodge(), show.legend = FALSE) +
  theme_stata() + labs(title = "CLV by Segment", x = "Segment",
                       y = "CLV", subtitle = "CLV by Segment") +
  theme(plot.title = element_text(size=20, face="bold"))
```




```
#plotting distribution of subcategories
ggplot(final_df, aes(x= subcategory_mode, y = category_mode, fill = category_mode)) +
  geom_col(position = position_dodge(),show.legend = FALSE) + coord_flip() +
  theme_stata() + labs(title = "Distribution of Subcategories", x = "Subcategory",
                       y = "Category", subtitle = "Distribution of Subcategory") +
  theme(plot.title = element_text(size=20, face="bold")) +
  theme(axis.text.y = element_text(size = 12, angle = 0))
```

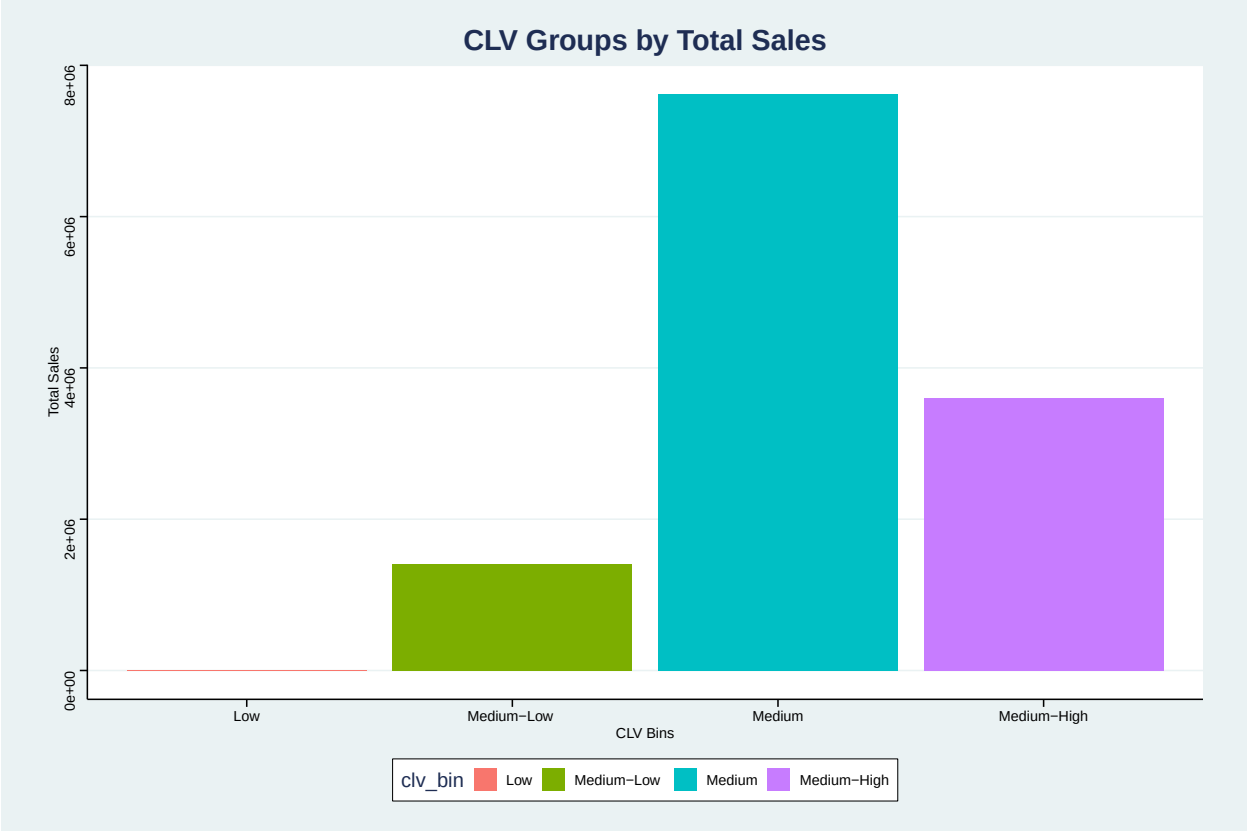



```
#scatter plot of sales and profit
ggplot(final_df, aes(x = total_sales, y = monetary)) +
  geom_point() + facet_wrap(~category_mode) +
  theme_stata() + labs(title = "Scatter Plot of Sales and Profit", x = "Sales",
                       y = "Profit", subtitle = "Scatter Plot of Sales and Profit") +
  theme(plot.title = element_text(size=20, face="bold"))
```

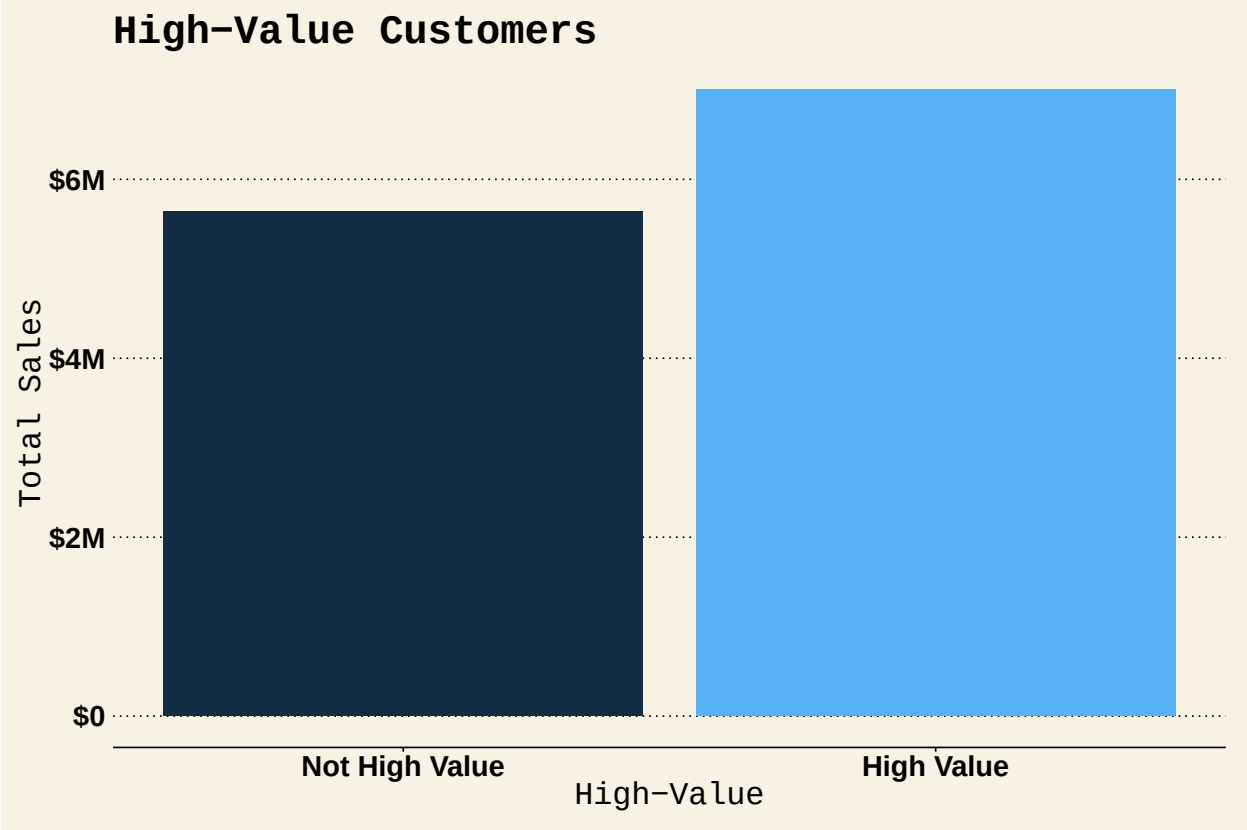



```
#bar graph of CLV bins
final_df %>% group_by(clv_bin) %>% summarise(total_sales = sum(total_sales)) %>%
  ggplot(aes(x = clv_bin, y = total_sales, fill = clv_bin)) +
  geom_col() +
  geom_line(aes(y = total_sales), color = "red") +
  theme_stata() + labs(title = "CLV Groups by Total Sales",
                       x = "CLV Bins", y = "Total Sales") +
  theme(plot.title = element_text(size=20, face="bold"))
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```




```r
#distribution of sales and high-value customers
table1 <- final_df %>% group_by(high_value) %>% summarise(total = n(),
                                                          sales = sum(total_sales))

#plot table1 sales
x_labels <- c("Not High Value", "High Value")
y_labels <- c("$0", "$2M", "$4M", "$6M")
ggplot(table1, aes(x = high_value, y = sales, fill = high_value)) +
  geom_col() +
  theme_wsj() + labs(title = "High-Value Customers",
                     x = "High-Value", y = "Total Sales") +
  theme(plot.title = element_text(size=28, face="bold")) +
  theme(axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20, angle = 0),
        axis.title = element_text(size = 22)) +
  theme(legend.position = "none") +
  scale_x_continuous(breaks = seq(0, 1), labels = x_labels) +
  scale_y_continuous(breaks = seq(0, 6000000, 2000000),
                     labels = y_labels)
```

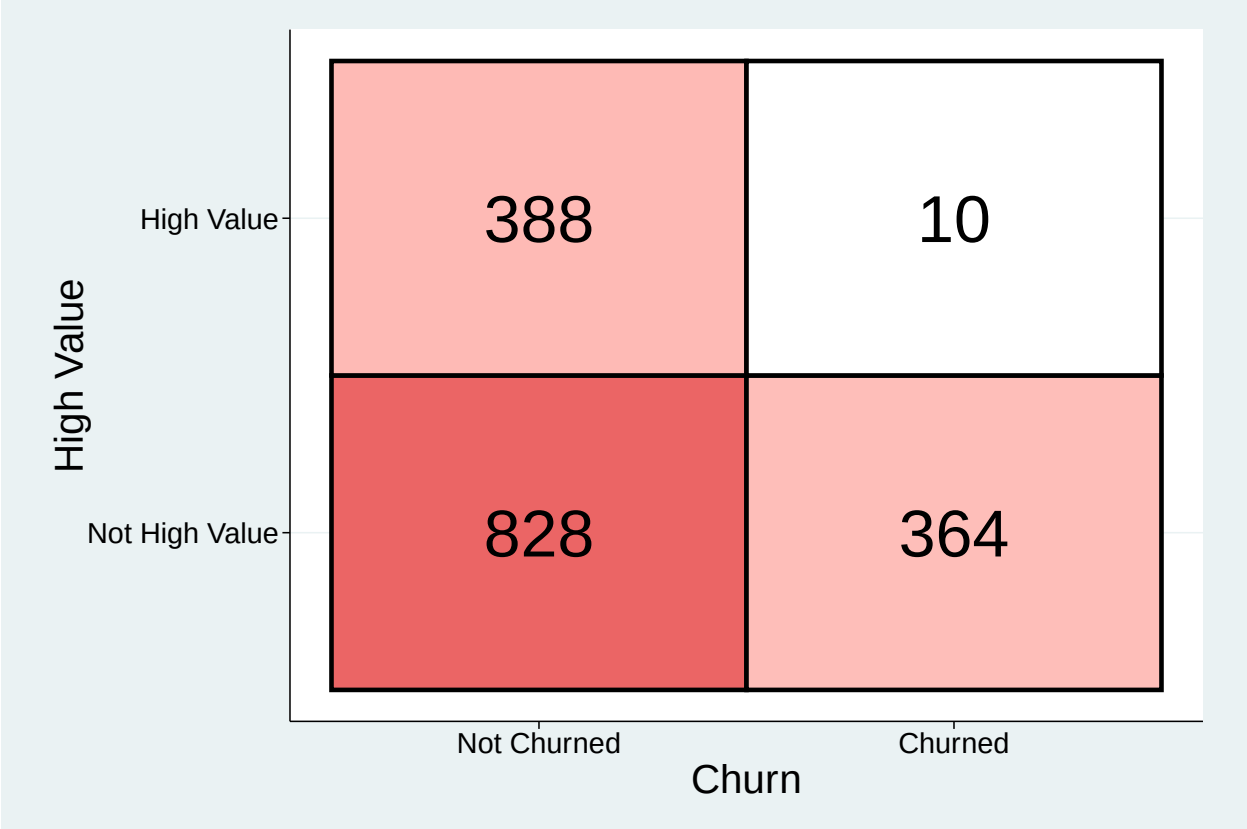



```
#churn vs high-value customers
churn_value_table <- final_df %>% group_by(churn, high_value) %>%
  summarise(total = n())
```

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

```
colnames(churn_value_table) <- c("Churn", "High_Value", "Total")

#display as percentage
churn_value_table <- churn_value_table %>%
  mutate(percentage = round(Total/sum(Total),3))

#heatmap of churn vs high-value customers
x_labels <- c("Not Churned", "Churned")
y_labels <- c("Not High Value", "High Value")

ggplot(churn_value_table, aes(x = Churn, y = High_Value, fill = Total)) +
  geom_tile(color = "black", lwd = 1.5, linetype = 1) +
  geom_text(aes(label = Total), size = 16) +
  scale_fill_gradient(low = "white", high = "#eb6565") +
  theme_stata() +
  theme(axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20, angle = 0),
        axis.title = element_text(size = 28)) +
  theme(legend.position = "none") +
  labs(x = "Churn", y = "High Value", fill = "Total") +
  scale_x_continuous(breaks = seq(0, 1), labels = x_labels) +
```

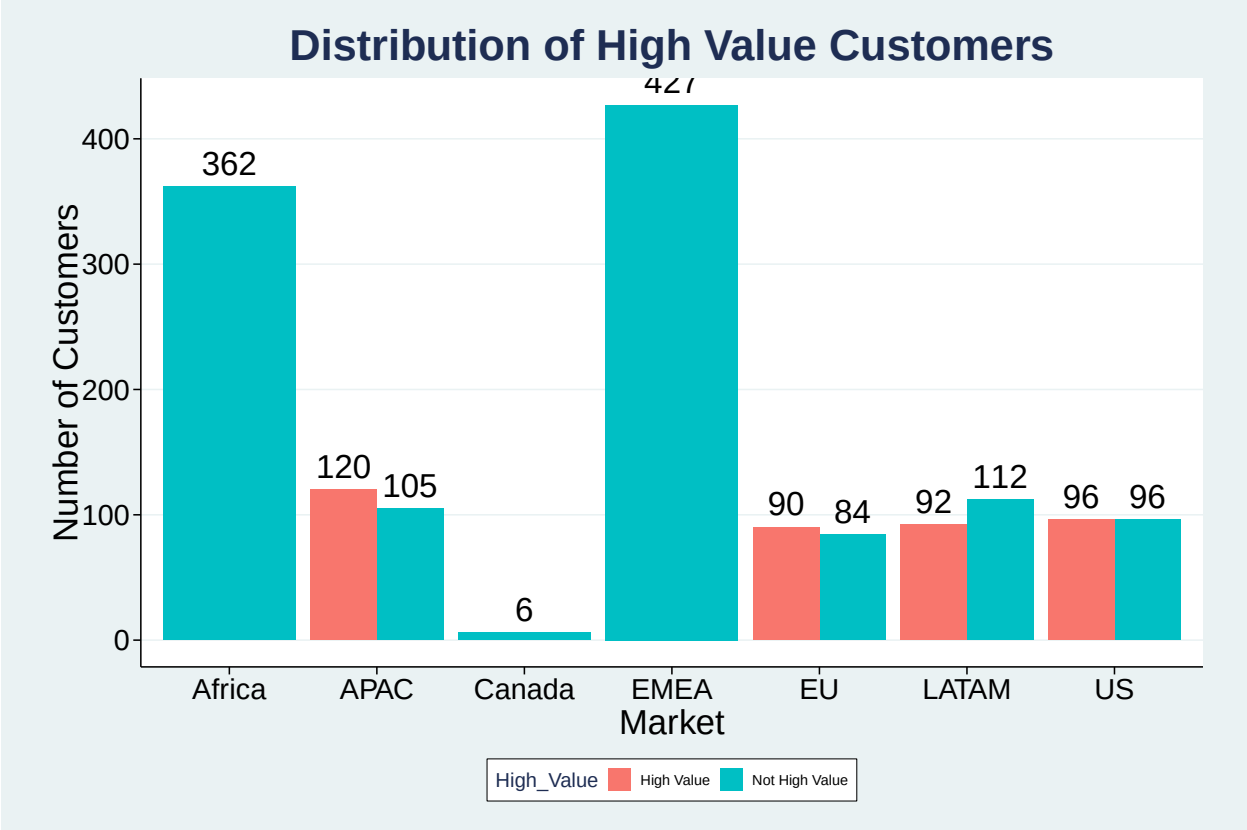
```
  scale_y_continuous(breaks = seq(0, 1), labels = y_labels)
```




```
#column chart of high value customers by market
high_value_market <- final_df %>% group_by(market_mode, high_value) %>%
  summarise(total = n())
```

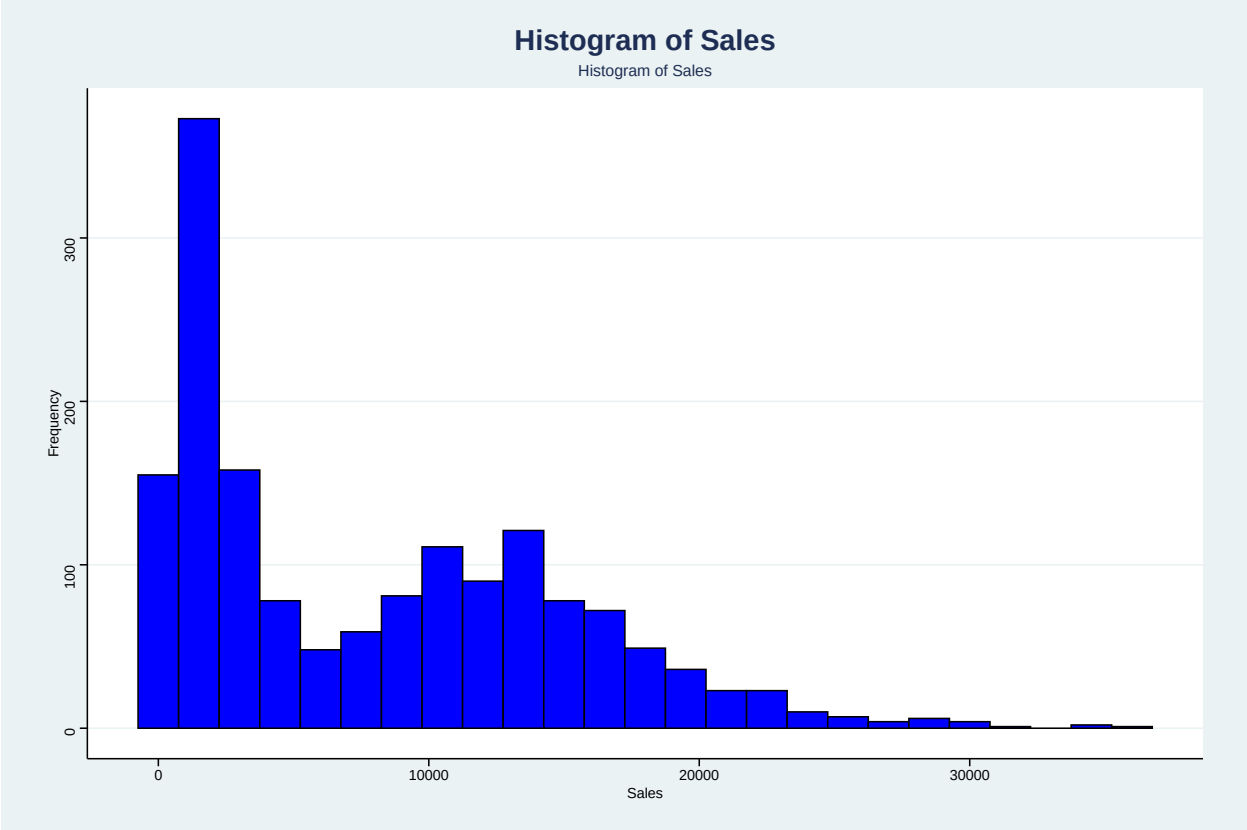
```
## `summarise()` has grouped output by 'market_mode'. You can override using the
## `.groups` argument.
```

```
colnames(high_value_market) <- c("Market", "High_Value", "Total")
high_value_market$High_Value <- ifelse(high_value_market$High_Value == 0,
                                       "Not High Value", "High Value")

#column chart of high value customers by segment
ggplot(high_value_market, aes(x = Market, y = Total, fill = High_Value)) +
  geom_col(position = position_dodge()) +
  geom_text(aes(label = Total), position = position_dodge(width = 0.9),
            vjust = -0.5, size = 8) +
  theme_stata() +
  theme(plot.title = element_text(size=30, face="bold")) +
  theme(axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20, angle = 0),
        axis.title = element_text(size = 24)) +
  labs(title = "Distribution of High Value Customers",
       x = "Market", y = "Number of Customers")
```

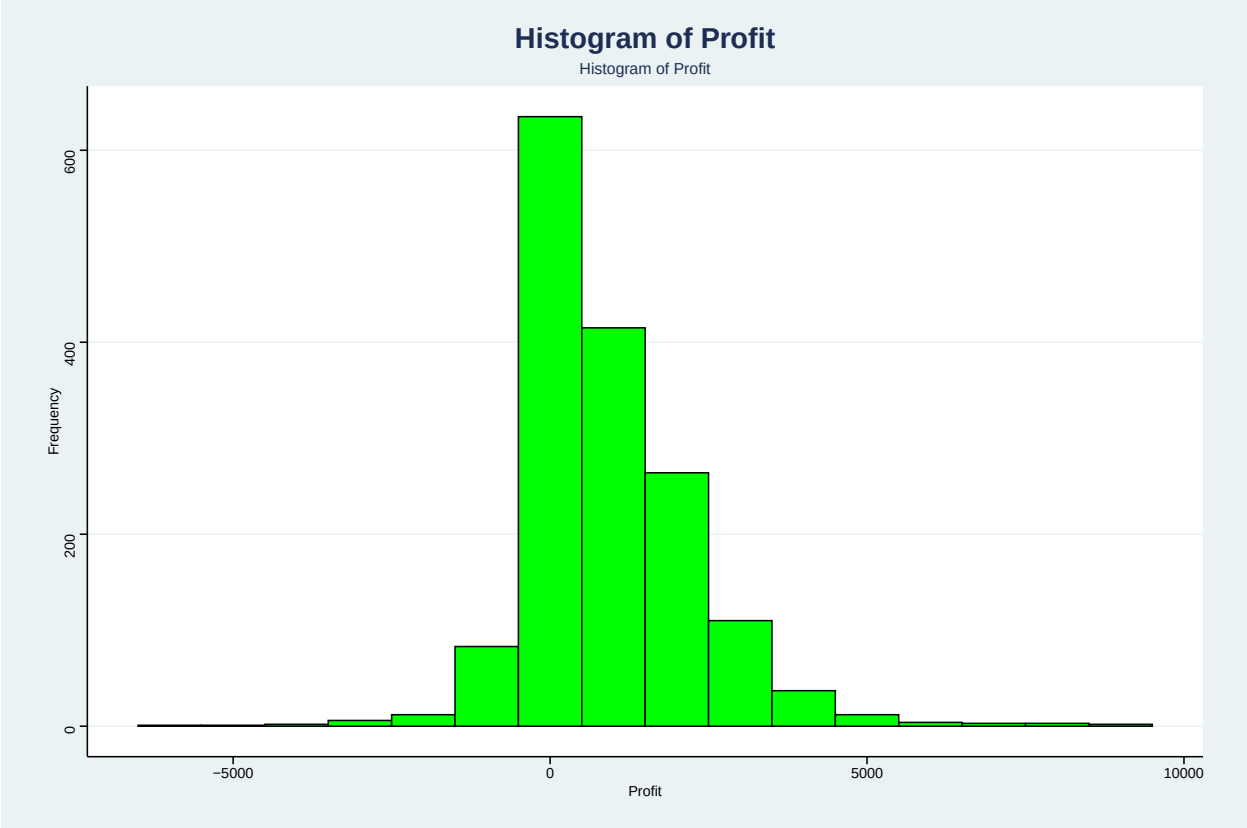


Histograms

```
#histogram of sales
ggplot(final_df, aes(x = total_sales)) +
  geom_histogram(binwidth = 1500, fill = "blue", col = "black") +
  theme_stata() + labs(title = "Histogram of Sales", x = "Sales",
                       y = "Frequency", subtitle = "Histogram of Sales") +
  theme(plot.title = element_text(size=20, face="bold"))
```

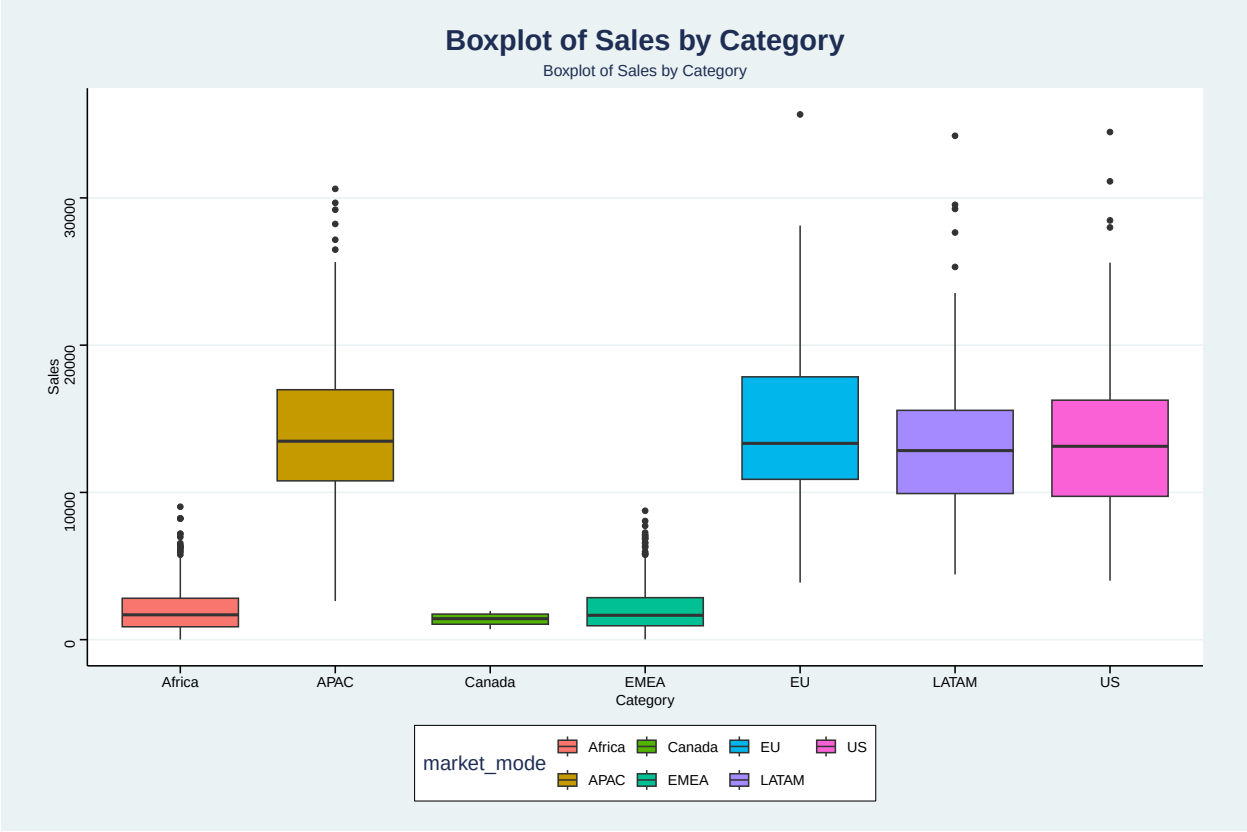



```
#histogram of profit
ggplot(final_df, aes(x = monetary)) +
  geom_histogram(binwidth = 1000, fill = "green", col = "black") +
  theme_stata() + labs(title = "Histogram of Profit", x = "Profit",
                       y = "Frequency", subtitle = "Histogram of Profit") +
  theme(plot.title = element_text(size=20, face="bold"))
```

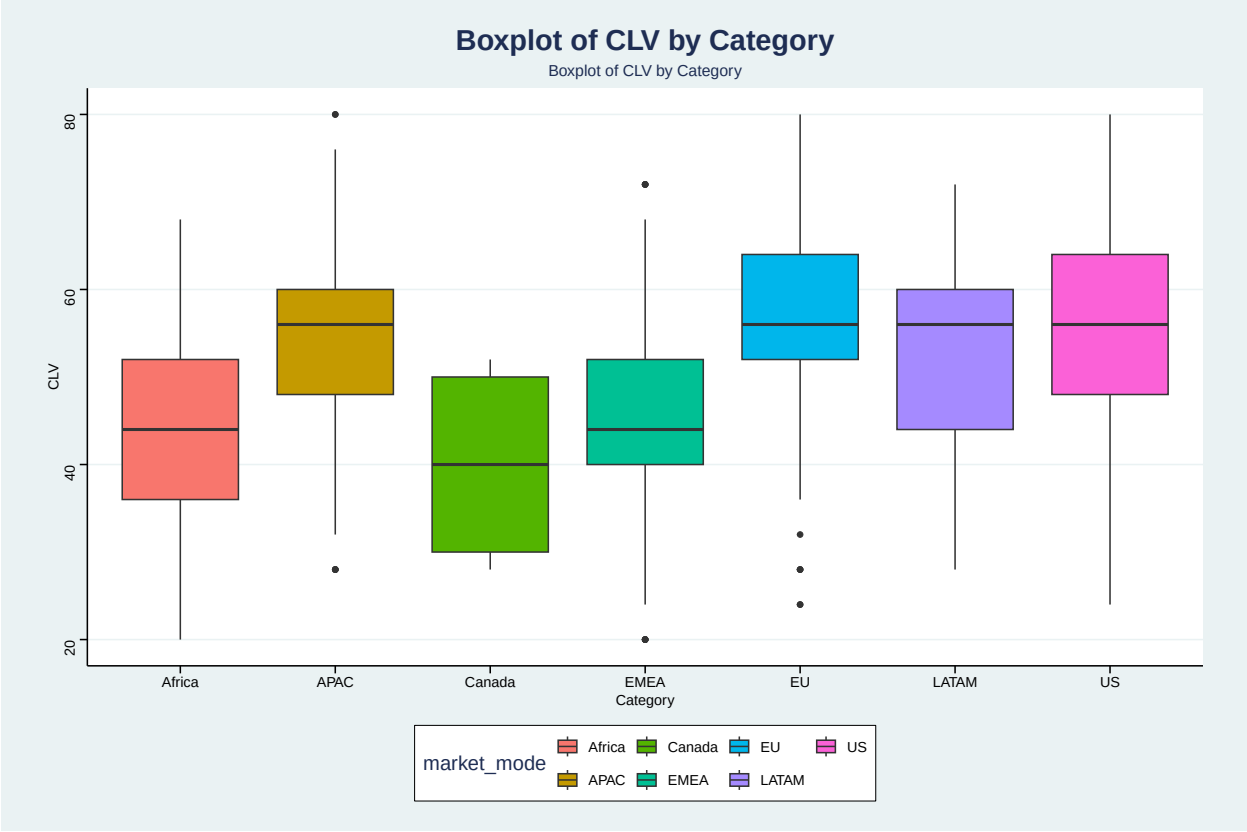


Boxplots

```
#boxplot of sales by market
ggplot(final_df, aes(x = market_mode, y = total_sales, fill = market_mode)) +
  geom_boxplot() +
  theme_stata() + labs(title = "Boxplot of Sales by Category", x = "Category",
                       y = "Sales", subtitle = "Boxplot of Sales by Category") +
  theme(plot.title = element_text(size=20, face="bold"))
```

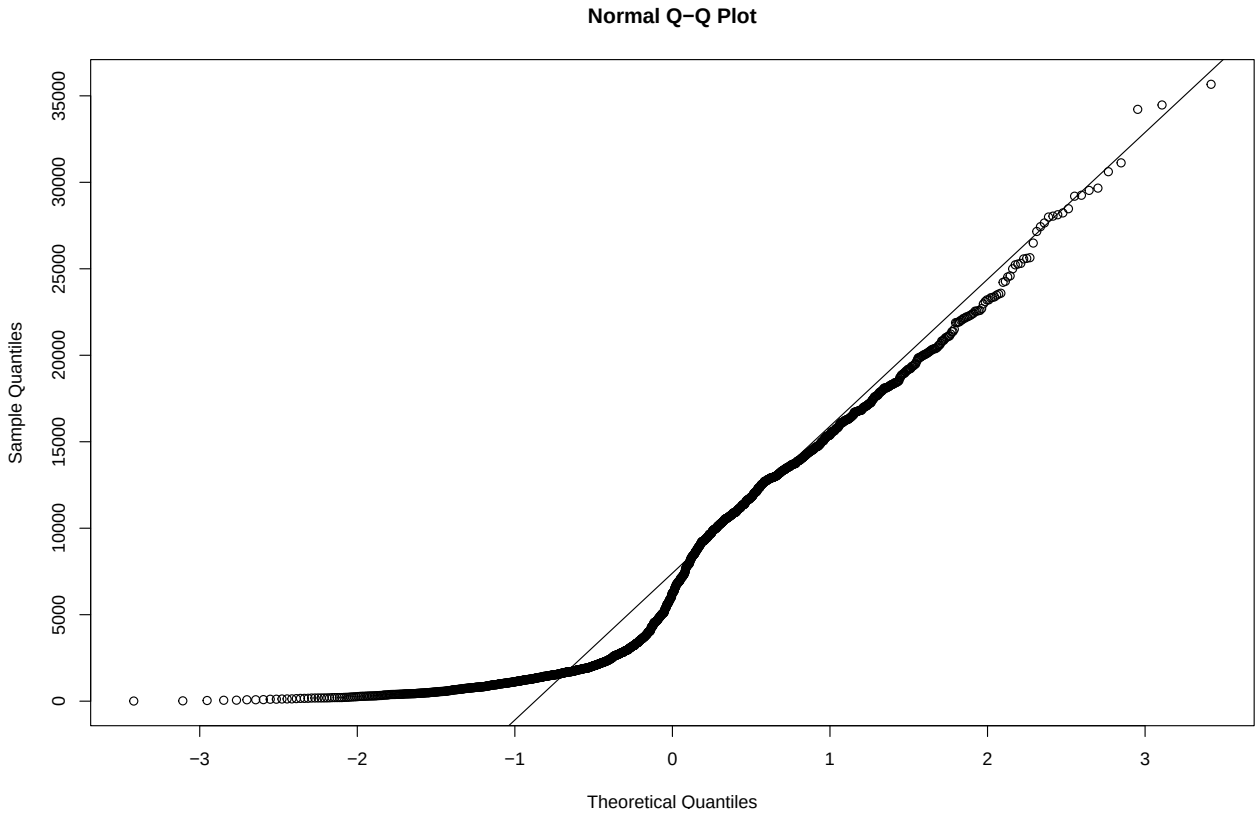



```
#boxplot of CLV by market
ggplot(final_df, aes(x = market_mode, y = CLV, fill = market_mode)) +
  geom_boxplot() +
  theme_stata() + labs(title = "Boxplot of CLV by Category", x = "Category",
                       y = "CLV", subtitle = "Boxplot of CLV by Category") +
  theme(plot.title = element_text(size=20, face="bold"))
```

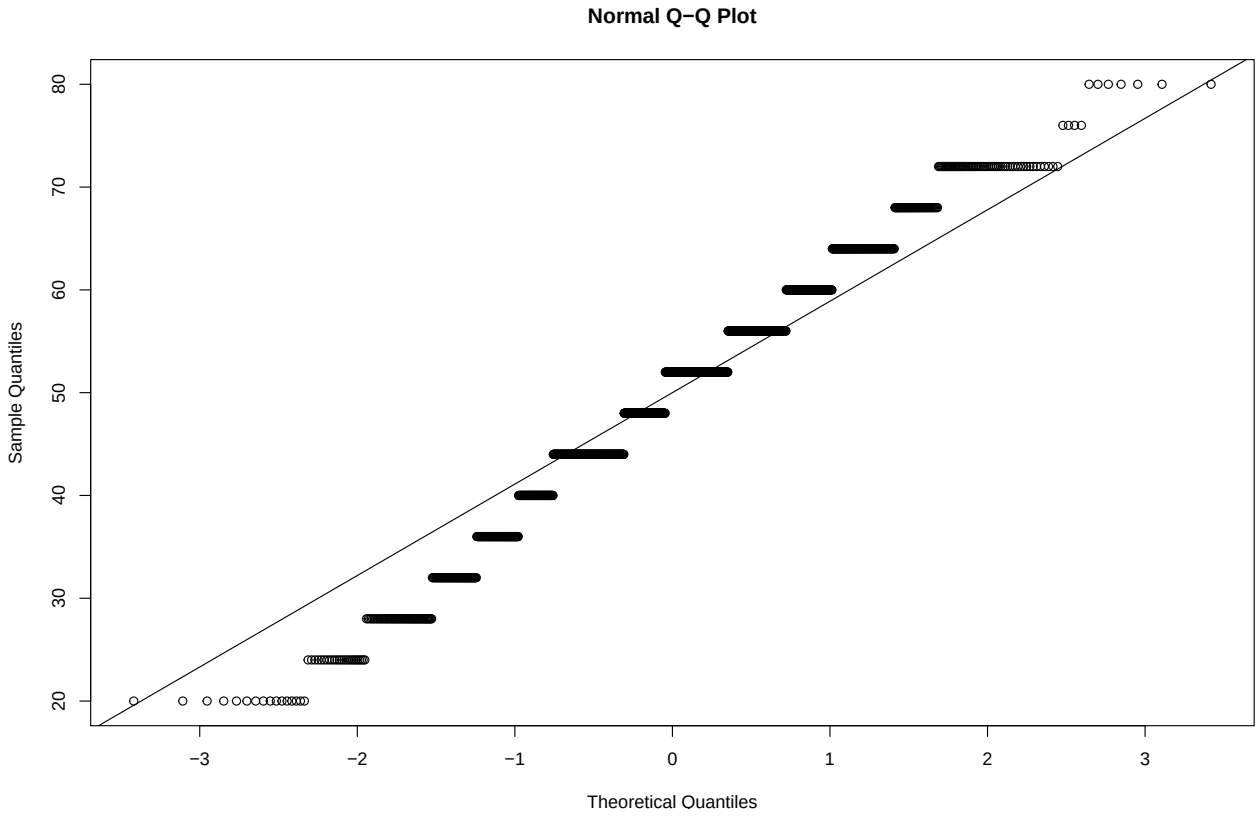


qqplots

```
#qqplot of sales
qqnorm(final_df$total_sales)
qqline(final_df$total_sales)
```




```
##qqplot of CLV
qqnorm(final_df$CLV)
qqline(final_df$CLV)
```



Correlations and Correlation Matrix

```
#correlation between discount and sales
cor.test(final_df$avg_discount, final_df$total_sales)
```

```
##
##  Pearson's product-moment correlation
##
## data:  final_df$avg_discount and final_df$total_sales
## t = -7.3887, df = 1588, p-value = 2.386e-13
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.2294102 -0.1343508
## sample estimates:
##        cor
## -0.1823064
```

```
#correlation between quantity and sales
cor.test(final_df$avg_quantity, final_df$total_sales)
```

```
##
##  Pearson's product-moment correlation
##
## data:  final_df$avg_quantity and final_df$total_sales
## t = 39.873, df = 1588, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6818669 0.7310568
## sample estimates:
```

```
##      cor
## 0.707317
```

```
#correlation between shipping cost and sales
cor.test(final_df$avg_shipping_cost, final_df$total_sales)
```

```
##
##  Pearson's product-moment correlation
##
## data:  final_df$avg_shipping_cost and final_df$total_sales
## t = 25.473, df = 1588, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5027352 0.5725838
## sample estimates:
##       cor
## 0.5385842
```

```
#correlation between CLV and sales
cor.test(final_df$CLV, final_df$total_sales)
```

```
##
##  Pearson's product-moment correlation
##
## data:  final_df$CLV and final_df$total_sales
## t = 24.87, df = 1588, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4931277 0.5639342
## sample estimates:
##       cor
## 0.5294524
```

```
#correlation between frequency and sales
cor.test(final_df$frequency, final_df$total_sales)
```

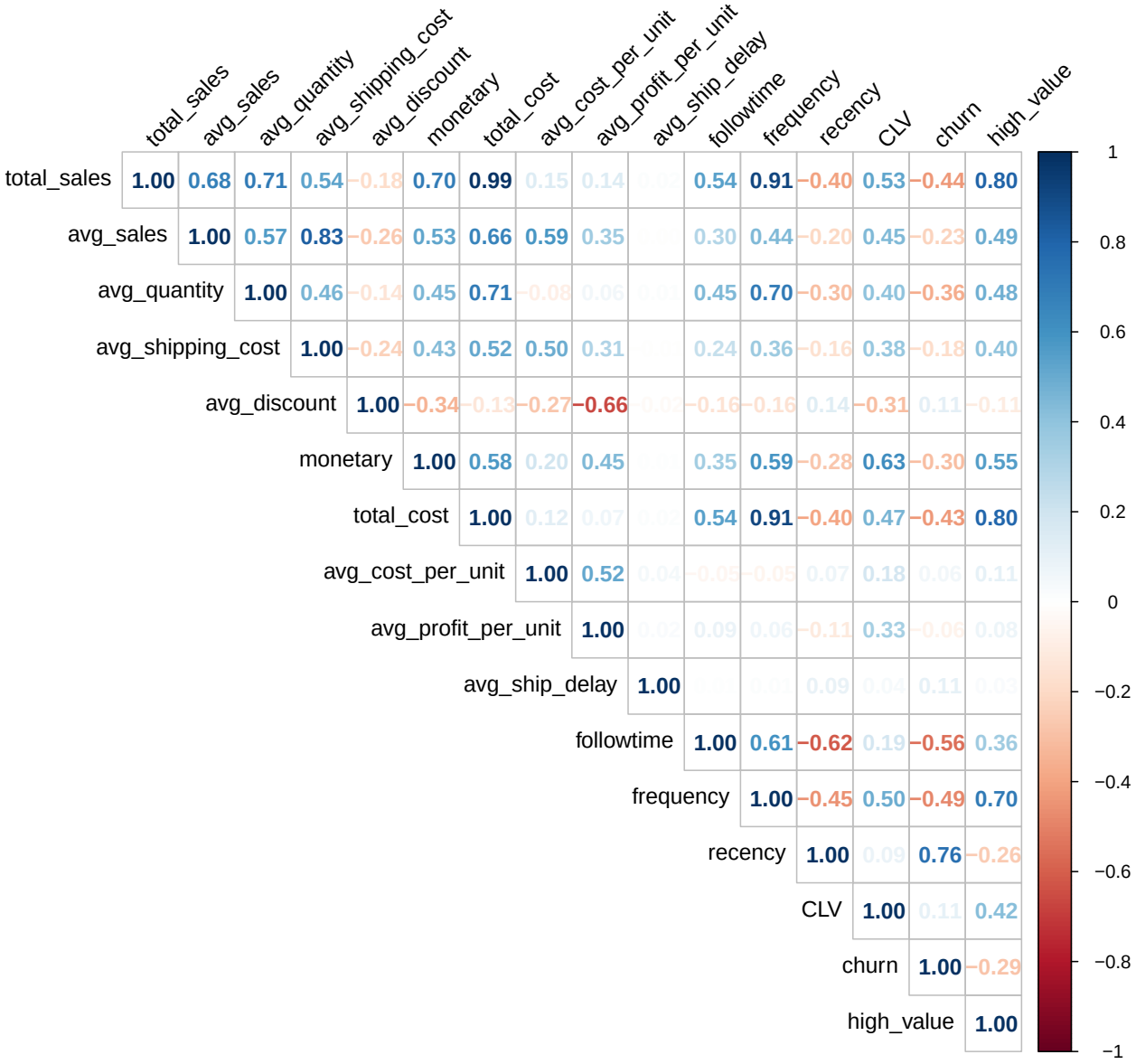
```
##
##  Pearson's product-moment correlation
##
## data:  final_df$frequency and final_df$total_sales
## t = 89.5, df = 1588, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9050238 0.9213232
## sample estimates:
##       cor
## 0.9135395
```

```
#correlation between recency and sales
cor.test(final_df$recency, final_df$total_sales)
```

```
##
##  Pearson's product-moment correlation
##
## data:  final_df$recency and final_df$total_sales
## t = -17.431, df = 1588, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
```

```
## 95 percent confidence interval:
##  -0.4412174 -0.3586561
## sample estimates:
##       cor
## -0.40075
```

```r
#correlation matrix
correlation_matrix <- cor(final_df %>% select_if(is.numeric))

#plotting correlation matrix
corrplot(correlation_matrix, method = "number", type = "upper",
         tl.col = "black", tl.srt = 45)
```



Chi-Square Test, ANOVA, and T-Test (Hypothesis Testing)

Chi-Square Test

```
#chi-square test
chisq.test(final_df$subcategory_mode, final_df$region_mode)
```

```
## Warning in chisq.test(final_df$subcategory_mode, final_df$region_mode):
## Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  final_df$subcategory_mode and final_df$region_mode
## X-squared = 332.05, df = 192, p-value = 1.446e-09
```

```
chisq.test(final_df$subcategory_mode, final_df$segment_mode)
```

```
## Warning in chisq.test(final_df$subcategory_mode, final_df$segment_mode):
## Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  final_df$subcategory_mode and final_df$segment_mode
## X-squared = 29.672, df = 32, p-value = 0.5849
```

```
chisq.test(final_df$subcategory_mode, final_df$market_mode)
```

```
## Warning in chisq.test(final_df$subcategory_mode, final_df$market_mode):
## Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  final_df$subcategory_mode and final_df$market_mode
## X-squared = 266.66, df = 96, p-value < 2.2e-16
```

```
chisq.test(final_df$subcategory_mode, final_df$ship_mode)
```

```
## Warning in chisq.test(final_df$subcategory_mode, final_df$ship_mode):
## Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  final_df$subcategory_mode and final_df$ship_mode
## X-squared = 77.653, df = 48, p-value = 0.004308
```

```
chisq.test(final_df$subcategory_mode, final_df$order_priority_mode)
```

```
## Warning in chisq.test(final_df$subcategory_mode, final_df$order_priority_mode):
## Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  final_df$subcategory_mode and final_df$order_priority_mode
## X-squared = 40.547, df = 48, p-value = 0.7688
```

```
chisq.test(final_df$subcategory_mode, final_df$clv_bin)
```

```
## Warning in chisq.test(final_df$subcategory_mode, final_df$clv_bin): Chi-squared
## approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  final_df$subcategory_mode and final_df$clv_bin
## X-squared = 59.693, df = 48, p-value = 0.12
```

ANOVA

```
#ANOVA test
anova_model <- aov(total_sales ~ subcategory_mode, data = final_df)
summary(anova_model)
```

```
##                    Df    Sum Sq   Mean Sq F value Pr(>F)
## subcategory_mode   16 6.604e+09 412727158   9.294 <2e-16 ***
## Residuals        1573 6.985e+10  44407310
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova_model2 <- aov(monetary ~ subcategory_mode, data = final_df)
summary(anova_model2)
```

```
##                    Df    Sum Sq Mean Sq F value  Pr(>F)
## subcategory_mode   16 1.263e+08 7892368   4.636 3.2e-09 ***
## Residuals        1573 2.678e+09 1702311
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova_model3 <- aov(CLV ~ segment_mode, data = final_df)
summary(anova_model3)
```

```
##                Df Sum Sq Mean Sq F value Pr(>F)
## segment_mode    2    144   71.81   0.517  0.596
## Residuals    1587 220247  138.78
```

T-Test

```
#t-test
t.test(final_df$total_sales, final_df$monetary)
```

```
##
##  Welch Two Sample t-test
##
## data:  final_df$total_sales and final_df$monetary
## t = 39.681, df = 1705.4, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  6680.935 7375.725
## sample estimates:
## mean of x mean of y
## 7951.2591  922.9291
```

```r
t.test(final_df$total_sales, final_df$CLV)
```

```
##
##  Welch Two Sample t-test
##
## data:  final_df$total_sales and final_df$CLV
## t = 45.422, df = 1589, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  7560.312 8242.739
## sample estimates:
##  mean of x  mean of y
## 7951.25906   49.73333
```

```r
t.test(final_df$total_sales, final_df$frequency)
```

```
##
##  Welch Two Sample t-test
##
## data:  final_df$total_sales and final_df$frequency
## t = 45.522, df = 1589, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  7577.787 8260.216
## sample estimates:
##  mean of x  mean of y
## 7951.25906   32.25786
```

Model Building

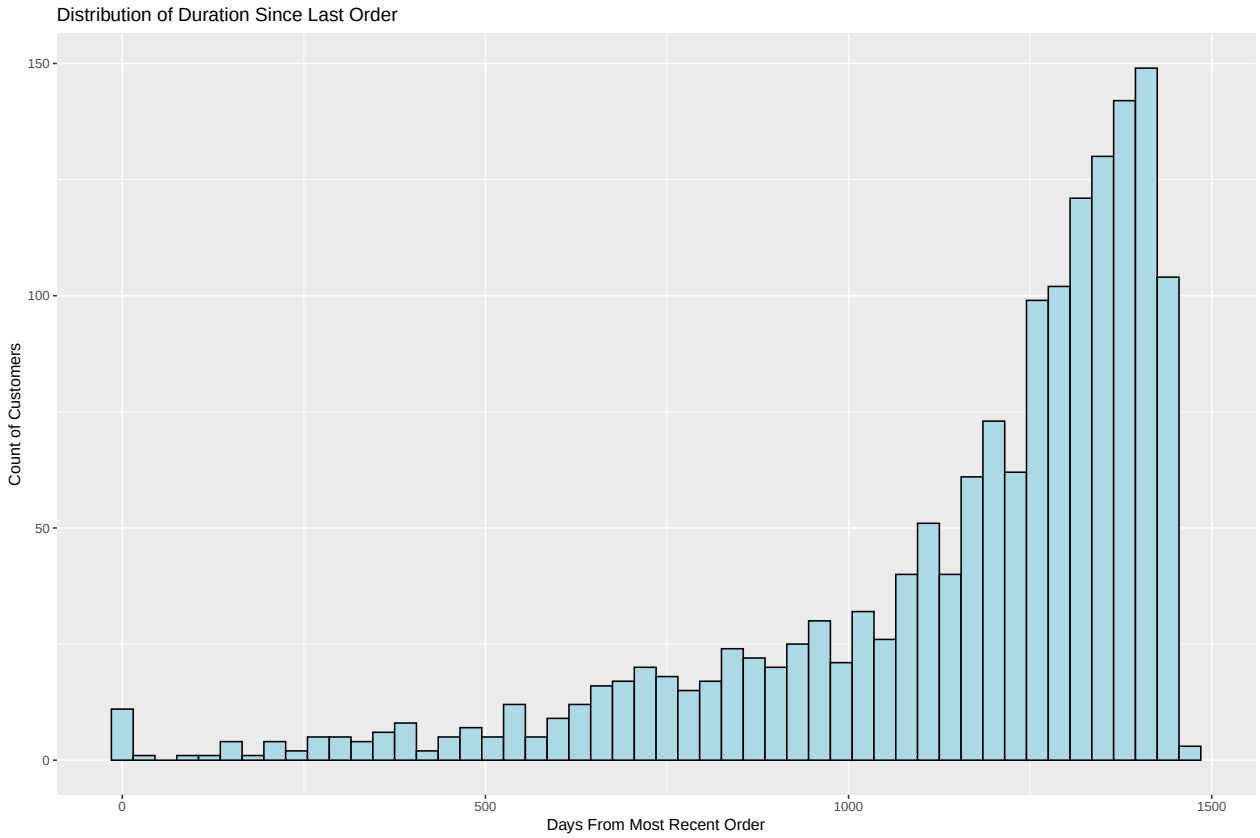
Survival Analysis

```r
customer.df <- final_df
```

```r
# Check summary statistics for time sence last order
summary(customer.df$recency)
```

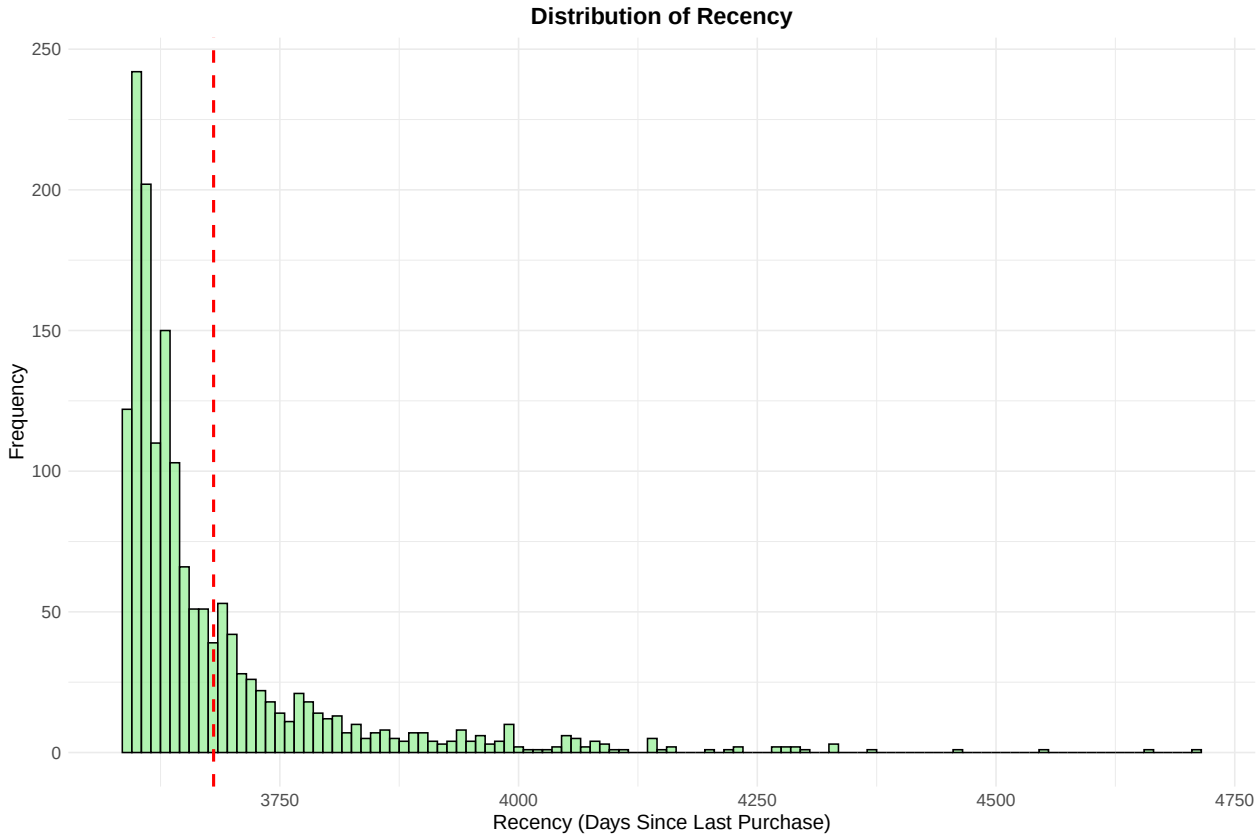
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3593    3607    3633    3681    3696    4711
```

```r
# Visualize the distribution of followtime
ggplot(customer.df, aes(x = followtime)) +
  geom_histogram(binwidth = 30, color = "black", fill = "lightblue") +
  labs(title = "Distribution of Duration Since Last Order",
       x = "Days From Most Recent Order", y = "Count of Customers")
```




```
# Calculate quantiles for recency
quantiles <- quantile(customer.df$recency)

#customer.df
ggplot(customer.df, aes(x = recency)) +
  geom_histogram(binwidth = 10, color = "black", fill = "lightgreen",
                 alpha = 0.7) +
  labs(
    title = "Distribution of Recency",
    x = "Recency (Days Since Last Purchase)",
    y = "Frequency"
  ) +
  geom_vline(aes(xintercept = mean(recency, na.rm = TRUE)),
             color = "red", linetype = "dashed", size = 1) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5, size = 16, face = "bold"),
    axis.title = element_text(size = 14),
    axis.text = element_text(size = 12)
  )
```

```
## 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.
```



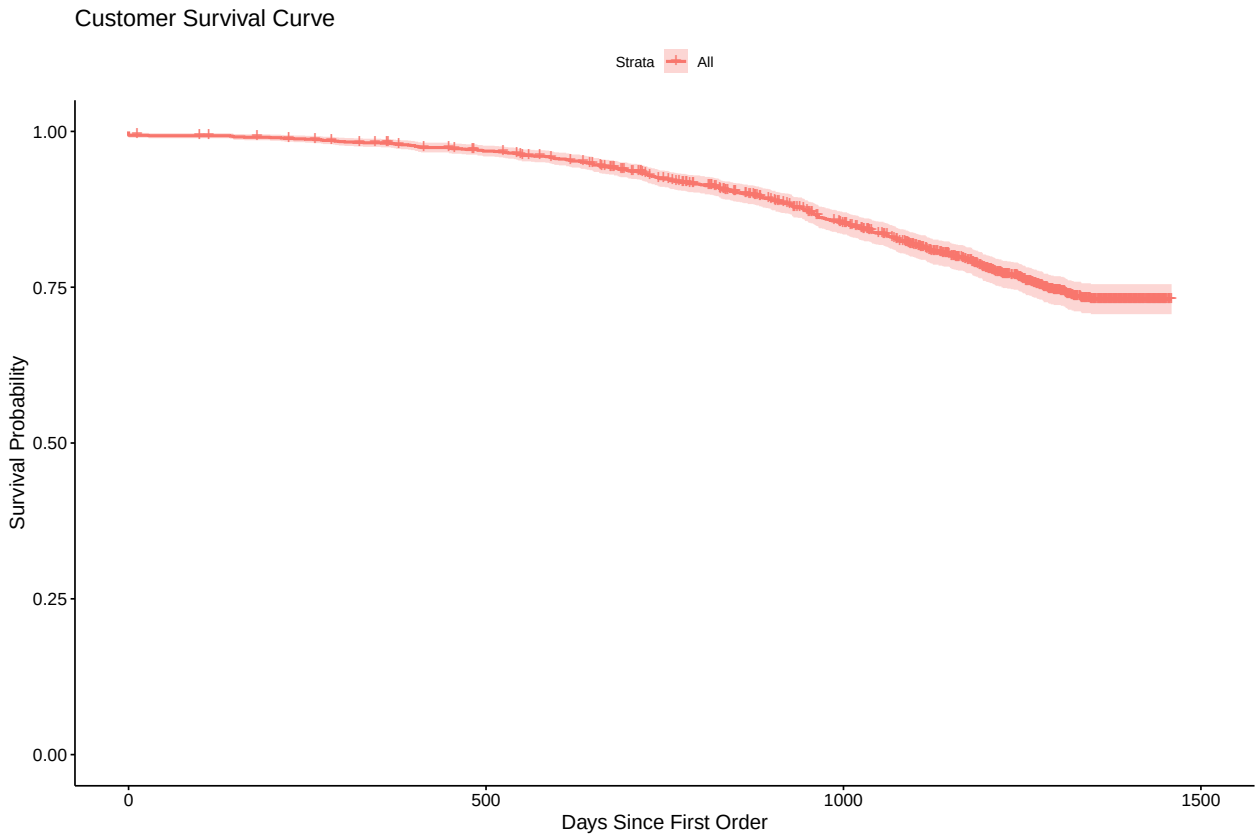
Now that we have our Time and Event Variable, we can continue with our survival analysis.

```
# Create the survival object
customer.df$survival <- Surv(customer.df$followtime, customer.df$churn)
customer.df
```

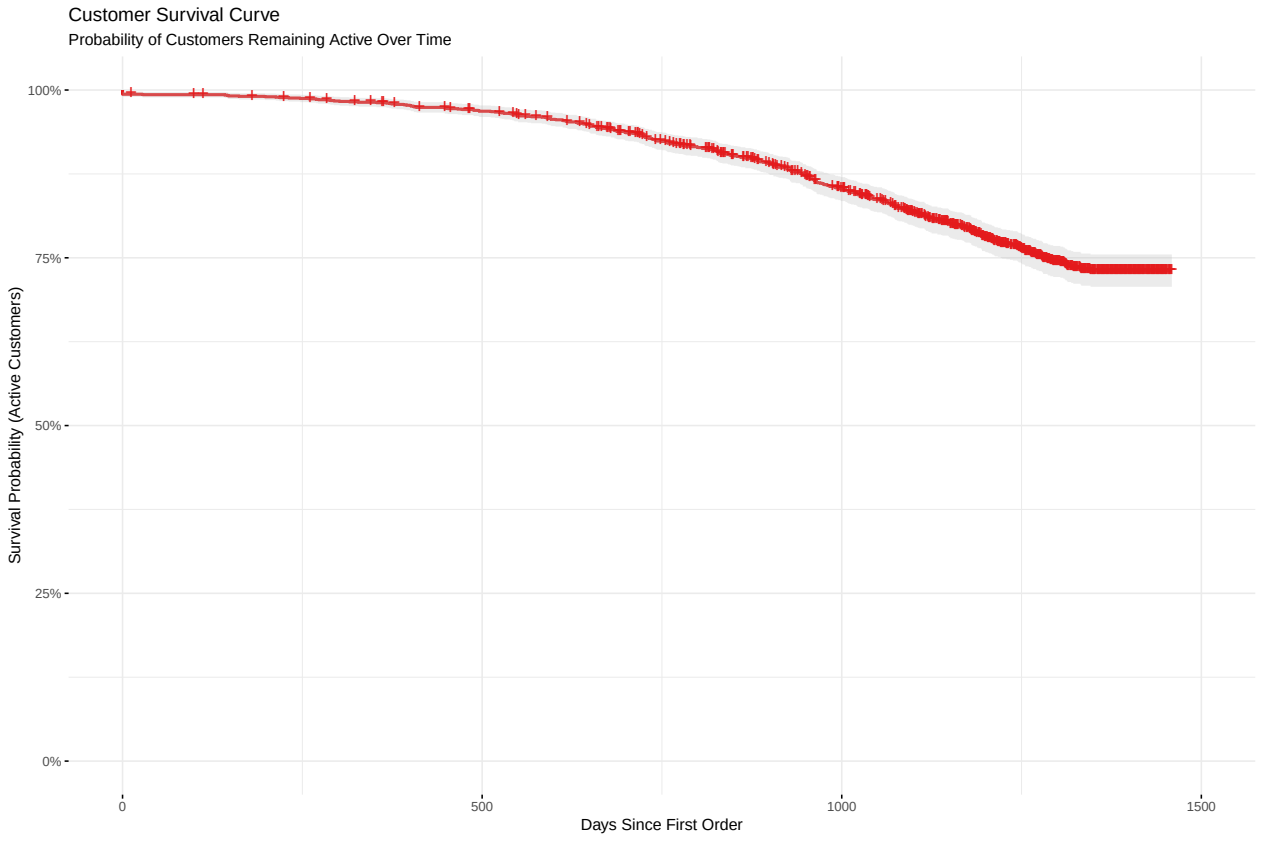
```
## # A tibble: 1,590 x 40
##    Customer.ID customer_name   first_order_date last_order_date total_sales
##    <fct>       <chr>           <date>           <date>                <dbl>
##  1 AA-10315    Alex Avila      2011-03-31       2014-12-23           13747.
##  2 AA-10375    Allen Armold    2011-04-21       2014-12-25            5884.
##  3 AA-10480    Andrew Allen    2011-01-11       2014-09-05           17696.
##  4 AA-10645    Anna Andreadi   2011-01-12       2014-12-05           15344.
##  5 AA-315      Alex Avila      2011-08-06       2014-12-29            2243.
##  6 AA-375      Allen Armold    2011-01-06       2014-07-03             654.
##  7 AA-480      Andrew Allen    2011-06-21       2014-02-20            2063.
##  8 AA-645      Anna Andreadi   2011-04-22       2014-10-11            1968.
##  9 AB-10015    Aaron Bergman   2011-02-19       2014-12-15           20037.
## 10 AB-10060    Adam Bellavance 2011-01-06       2014-12-06           18417.
## # i 1,580 more rows
## # i 35 more variables: avg_sales <dbl>, avg_quantity <dbl>,
## #   avg_shipping_cost <dbl>, avg_discount <dbl>, monetary <dbl>,
## #   total_cost <dbl>, avg_cost_per_unit <dbl>, avg_profit_per_unit <dbl>,
## #   avg_ship_delay <dbl>, followtime <dbl>, ship_mode <fct>,
## #   segment_mode <fct>, city_mode <fct>, state_mode <fct>, country_mode <fct>,
## #   market_mode <fct>, region_mode <fct>, category_mode <fct>, ...
```

```
# Convert character variables to factors in customer.df
customer.df$segment <- as.factor(customer.df$segment_mode)
```

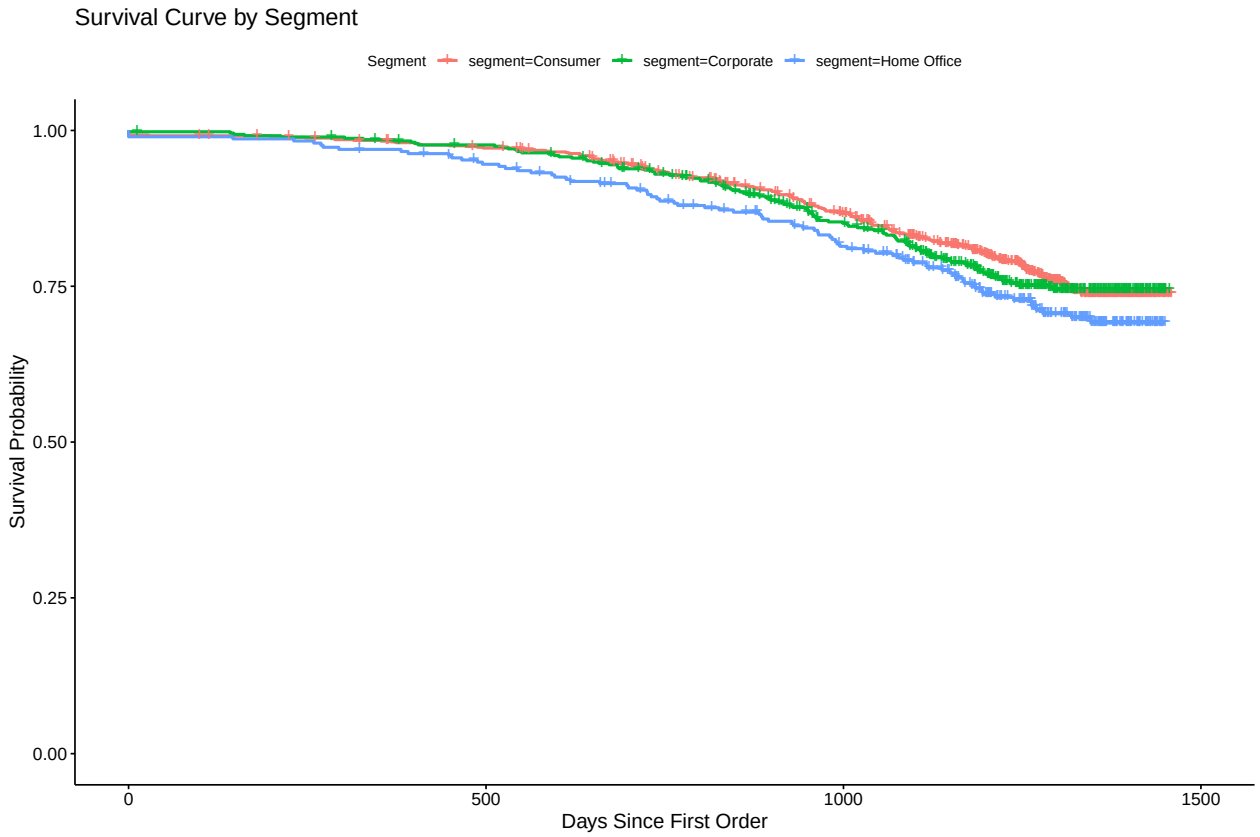
```r
customer.df$category <- as.factor(customer.df$category_mode)
customer.df$market <- as.factor(customer.df$market_mode)


customer.df$ship <- as.factor(customer.df$ship_mode)
customer.df$city <- as.factor(customer.df$city_mode)
#customer.df$state <- as.factor(customer.df$state_mode)
customer.df$country <- as.factor(customer.df$country_mode)
#customer.df$region <- as.factor(customer.df$region_mode)
customer.df$subcategory <- as.factor(customer.df$subcategory_mode)
customer.df$product <- as.factor(customer.df$product_mode)
customer.df$order_priority <- as.factor(customer.df$order_priority_mode)
customer.df$order_priority <- relevel(customer.df$order_priority, ref = "High")

customer.df$dayofweek <- as.factor(customer.df$dayofweek_mode)


# Fit the Kaplan-Meier survival curve
fit <- survfit(survival ~ 1, data = customer.df)

# Plot the survival curve
ggsurvplot(fit, data = customer.df,
           xlab = "Days Since First Order",
           ylab = "Survival Probability",
           title = "Customer Survival Curve")
```



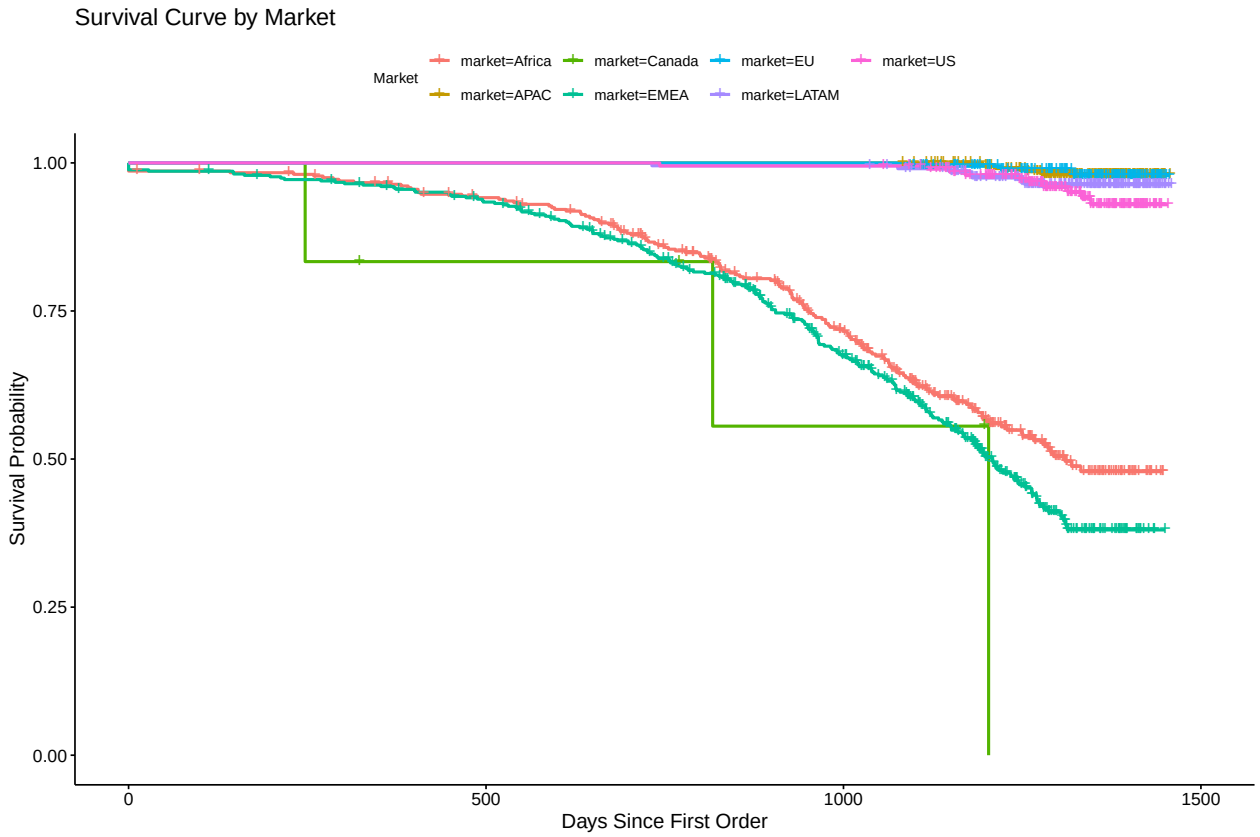
```r
# Enhanced survival curve
ggsurvplot(
  fit,
  conf.int = TRUE, # Add confidence interval
  xlab = "Days Since First Order", # Improved x-axis label
  ylab = "Survival Probability (Active Customers)", # Improved y-axis label
  title = "Customer Survival Curve", # Add a title
  subtitle = "Probability of Customers Remaining Active Over Time", # subtitle
  palette = "Set1", # Change color
  ggtheme = theme_minimal(), # Minimal clean theme
  legend = "none", # Remove legend if there's only one group
  surv.scale = "percent" # Show survival as percentages if preferred
)
```



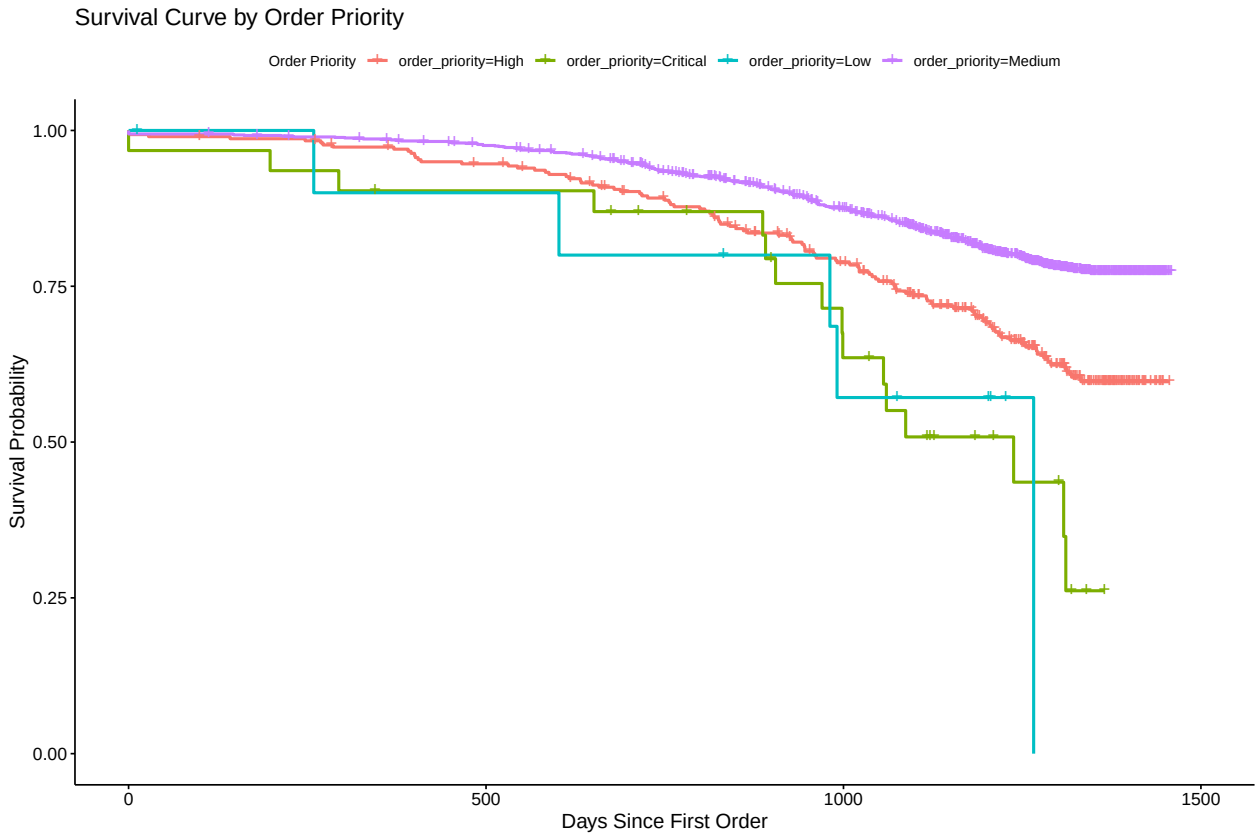

```r
#Stratify by Segment
fit_segment <- survfit(survival ~ segment, data = customer.df)
ggsurvplot(fit_segment, data = customer.df,
           xlab = "Days Since First Order",
           ylab = "Survival Probability",
           title = "Survival Curve by Segment",
           legend.title = "Segment")
```



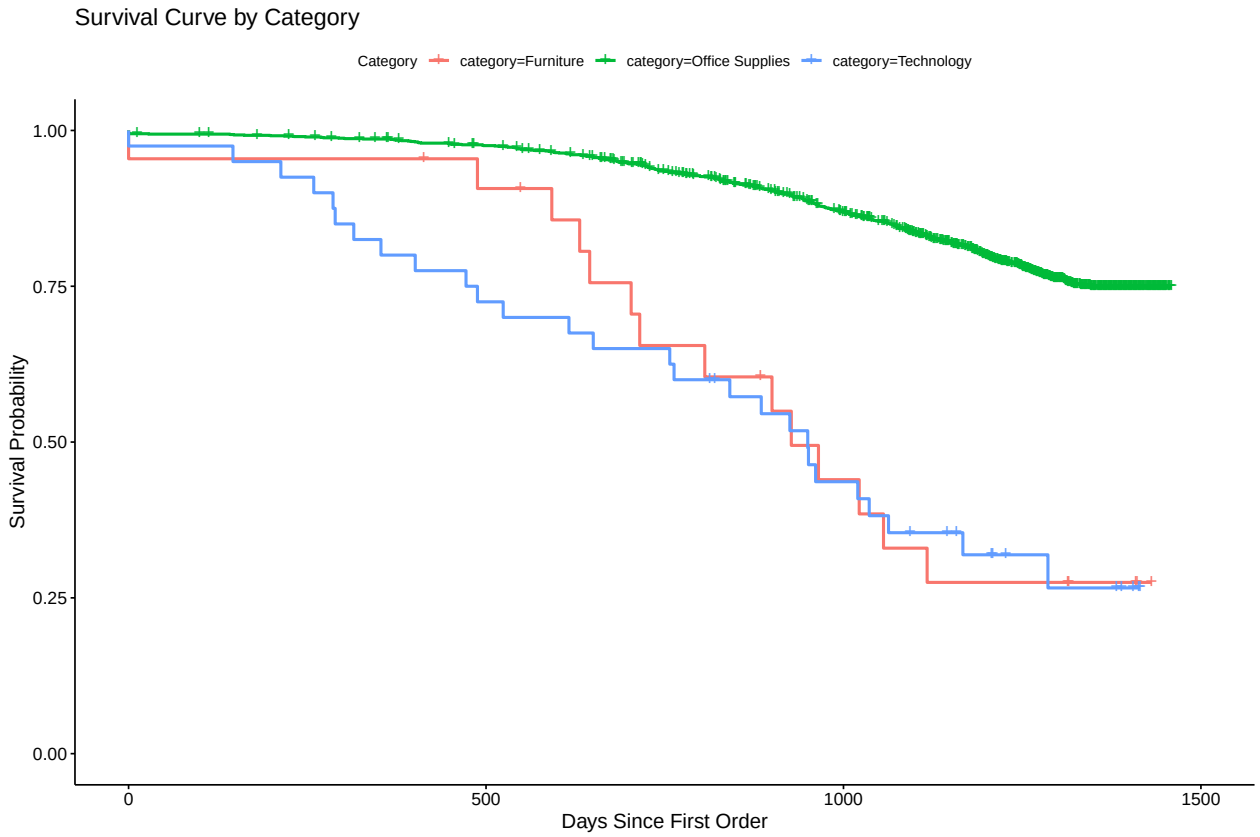

```
#Stratify by Market
fit_market <- survfit(survival ~ market, data = customer.df)
ggsurvplot(fit_market, data = customer.df,
           xlab = "Days Since First Order",
           ylab = "Survival Probability",
           title = "Survival Curve by Market",
           legend.title = "Market")
```



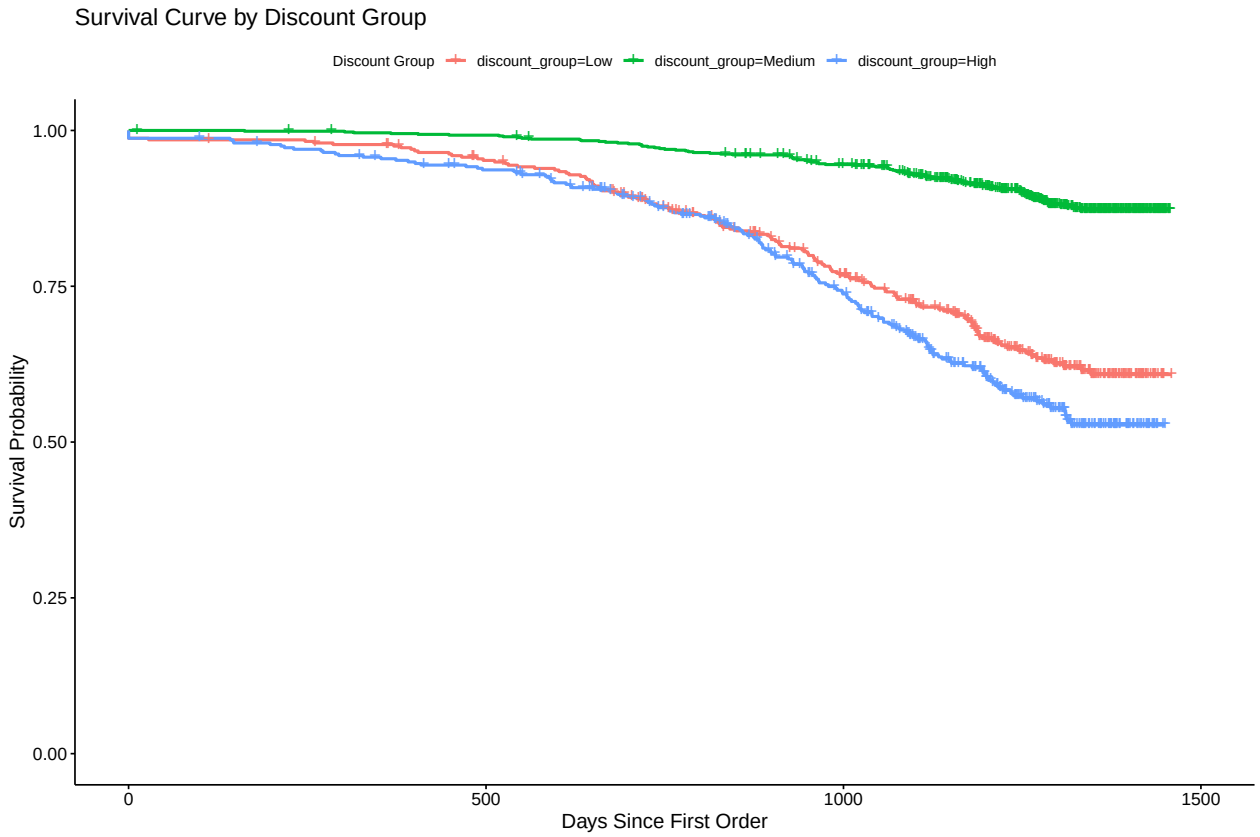

```r
#Stratify by Order Priority
fit_priority <- survfit(survival ~ order_priority, data = customer.df)
ggsurvplot(fit_priority, data = customer.df,
           xlab = "Days Since First Order",
           ylab = "Survival Probability",
           title = "Survival Curve by Order Priority",
           legend.title = "Order Priority")
```




```
#Stratify by Cateogry
fit_priority <- survfit(survival ~ category, data = customer.df)
ggsurvplot(fit_priority, data = customer.df,
           xlab = "Days Since First Order",
           ylab = "Survival Probability",
           title = "Survival Curve by Category",
           legend.title = "Category")
```




```
#Stratify by discount level
# group by quartiles
customer.df$discount_group <- cut(
  customer.df$avg_discount,
  breaks = quantile(customer.df$avg_discount, probs = c(0, 0.25, 0.75, 1),
                    na.rm = TRUE),
  labels = c("Low", "Medium", "High"),
  include.lowest = TRUE
)

fit_discount_group <- survfit(survival ~ discount_group, data = customer.df)
ggsurvplot(fit_discount_group, data = customer.df,
           xlab = "Days Since First Order",
           ylab = "Survival Probability",
           title = "Survival Curve by Discount Group",
           legend.title = "Discount Group")
```




```r
customer.df$discount_group <- relevel(customer.df$discount_group,
                                      ref = "Medium")
```

```r
#cox proportional hazards model
cox_model <- coxph(
  survival ~ total_sales + avg_sales + avg_quantity + avg_shipping_cost + avg_discount +
    monetary + total_cost + avg_cost_per_unit + avg_profit_per_unit + avg_ship_delay +
    ship + segment + category + market + order_priority + dayofweek + frequency + recency,
  data = customer.df
)
summary(cox_model)
```

```
## Call:
## coxph(formula = survival ~ total_sales + avg_sales + avg_quantity +
##     avg_shipping_cost + avg_discount + monetary + total_cost +
##     avg_cost_per_unit + avg_profit_per_unit + avg_ship_delay +
##     ship + segment + category + market + order_priority + dayofweek +
##     frequency + recency, data = customer.df)
##
##   n= 1590, number of events= 374
##
##                              coef  exp(coef)   se(coef)      z Pr(>|z|)
## total_sales             4.710e-05  1.000e+00  5.760e-05  0.818   0.4135
## avg_sales              -7.517e-04  9.992e-01  1.365e-03 -0.551   0.5818
## avg_quantity            1.167e-02  1.012e+00  8.129e-02  0.144   0.8858
## avg_shipping_cost       5.037e-03  1.005e+00  7.110e-03  0.708   0.4787
## avg_discount            2.560e-01  1.292e+00  4.804e-01  0.533   0.5941
## monetary               -1.071e-04  9.999e-01  9.534e-05 -1.124   0.2611
```

```
## total_cost                       NA         NA  0.000e+00     NA       NA
## avg_cost_per_unit       -5.577e-04  9.994e-01  2.493e-03 -0.224   0.8230
## avg_profit_per_unit      3.591e-03  1.004e+00  4.047e-03  0.887   0.3749
## avg_ship_delay           6.540e-04  1.001e+00  1.186e-03  0.552   0.5813
## shipSame Day            -1.869e-02  9.815e-01  3.623e-01 -0.052   0.9588
## shipSecond Class         3.377e-01  1.402e+00  2.247e-01  1.503   0.1328
## shipStandard Class      -1.203e-03  9.988e-01  2.098e-01 -0.006   0.9954
## segmentCorporate        -7.810e-03  9.922e-01  1.281e-01 -0.061   0.9514
## segmentHome Office       7.596e-02  1.079e+00  1.448e-01  0.524   0.6000
## categoryOffice Supplies  1.983e-01  1.219e+00  3.285e-01  0.604   0.5461
## categoryTechnology       5.854e-01  1.796e+00  3.592e-01  1.630   0.1031
## marketAPAC              -1.059e+00  3.467e-01  6.392e-01 -1.657   0.0974 .
## marketCanada            -3.842e-01  6.810e-01  6.158e-01 -0.624   0.5328
## marketEMEA               2.472e-01  1.280e+00  1.148e-01  2.153   0.0313 *
## marketEU                -1.112e+00  3.289e-01  6.861e-01 -1.621   0.1051
## marketLATAM             -5.505e-01  5.766e-01  5.242e-01 -1.050   0.2936
## marketUS                -4.823e-02  9.529e-01  4.923e-01 -0.098   0.9219
## order_priorityCritical   1.088e-01  1.115e+00  2.965e-01  0.367   0.7136
## order_priorityLow        5.738e-01  1.775e+00  4.780e-01  1.200   0.2300
## order_priorityMedium     1.405e-01  1.151e+00  1.358e-01  1.035   0.3008
## dayofweekMonday         -2.033e-01  8.161e-01  1.810e-01 -1.123   0.2613
## dayofweekSaturday       -1.417e-01  8.679e-01  2.203e-01 -0.643   0.5201
## dayofweekSunday          5.261e-01  1.692e+00  2.706e-01  1.944   0.0519 .
## dayofweekThursday       -9.243e-02  9.117e-01  1.908e-01 -0.485   0.6280
## dayofweekTuesday        -3.873e-01  6.789e-01  1.954e-01 -1.982   0.0475 *
## dayofweekWednesday      -1.732e-01  8.410e-01  1.845e-01 -0.938   0.3480
## frequency               -6.499e-02  9.371e-01  1.452e-02 -4.476  7.6e-06 ***
## recency                  7.789e-03  1.008e+00  3.345e-04 23.286  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                         exp(coef) exp(-coef) lower .95 upper .95
## total_sales                1.0000     1.0000   0.99993    1.0002
## avg_sales                  0.9992     1.0008   0.99658    1.0019
## avg_quantity               1.0117     0.9884   0.86273    1.1865
## avg_shipping_cost          1.0051     0.9950   0.99114    1.0192
## avg_discount               1.2918     0.7741   0.50376    3.3124
## monetary                   0.9999     1.0001   0.99971    1.0001
## total_cost                     NA         NA        NA        NA
## avg_cost_per_unit          0.9994     1.0006   0.99457    1.0043
## avg_profit_per_unit        1.0036     0.9964   0.99567    1.0116
## avg_ship_delay             1.0007     0.9993   0.99833    1.0030
## shipSame Day               0.9815     1.0189   0.48253    1.9964
## shipSecond Class           1.4018     0.7134   0.90242    2.1775
## shipStandard Class         0.9988     1.0012   0.66203    1.5069
## segmentCorporate           0.9922     1.0078   0.77195    1.2753
## segmentHome Office         1.0789     0.9269   0.81227    1.4331
## categoryOffice Supplies    1.2193     0.8201   0.64042    2.3215
## categoryTechnology         1.7957     0.5569   0.88823    3.6304
## marketAPAC                 0.3467     2.8845   0.09905    1.2134
## marketCanada               0.6810     1.4684   0.20369    2.2770
## marketEMEA                 1.2804     0.7810   1.02241    1.6035
## marketEU                   0.3289     3.0407   0.08570    1.2620
## marketLATAM                0.5766     1.7342   0.20641    1.6110
```

```
## marketUS                   0.9529     1.0494   0.36311    2.5007
## order_priorityCritical     1.1150     0.8969   0.62357    1.9937
## order_priorityLow          1.7751     0.5634   0.69552    4.5302
## order_priorityMedium       1.1509     0.8689   0.88193    1.5018
## dayofweekMonday            0.8161     1.2254   0.57239    1.1635
## dayofweekSaturday          0.8679     1.1522   0.56353    1.3366
## dayofweekSunday            1.6923     0.5909   0.99575    2.8762
## dayofweekThursday          0.9117     1.0968   0.62731    1.3251
## dayofweekTuesday           0.6789     1.4730   0.46284    0.9957
## dayofweekWednesday         0.8410     1.1890   0.58579    1.2074
## frequency                  0.9371     1.0672   0.91078    0.9641
## recency                    1.0078     0.9922   1.00716    1.0085
##
## Concordance= 0.937  (se = 0.004 )
## Likelihood ratio test= 1238  on 33 df,   p=<2e-16
## Wald test            = 942.3  on 33 df,   p=<2e-16
## Score (logrank) test = 2665  on 33 df,   p=<2e-16
```

```
cox_test <- cox.zph(cox_model)
print(cox_test)
```

```
##                        chisq df       p
## total_sales         1.02e+01  1  0.0014
## avg_sales           1.51e+00  1  0.2187
## avg_quantity        6.64e-01  1  0.4153
## avg_shipping_cost   4.00e-02  1  0.8415
## avg_discount        9.13e-03  1  0.9239
## monetary            1.89e+00  1  0.1696
## avg_cost_per_unit   8.37e-01  1  0.3604
## avg_profit_per_unit 6.37e-01  1  0.4247
## avg_ship_delay      1.31e+00  1  0.2533
## ship                8.06e+00  3  0.0449
## segment             6.41e-01  2  0.7257
## category            1.57e+00  2  0.4560
## market              8.72e+00  6  0.1900
## order_priority      1.17e+00  3  0.7601
## dayofweek           2.05e+01  6  0.0023
## frequency           1.58e+01  1 7.2e-05
## recency             2.47e+02  1 < 2e-16
## GLOBAL              3.21e+02 33 < 2e-16
```

```
#plot(cox_test)

cox_model_improved <- coxph(
  survival ~ avg_discount +
   avg_cost_per_unit + avg_profit_per_unit + avg_ship_delay +
    ship + segment  + order_priority ,
  data = customer.df
)

cox_test <- cox.zph(cox_model_improved)
print(cox_test)
```

```
##                        chisq df    p
## avg_discount        1.24e-03  1 0.97
```

```
## avg_cost_per_unit   9.28e-05  1 0.99
## avg_profit_per_unit 1.56e-03  1 0.97
## avg_ship_delay      1.99e-02  1 0.89
## ship                3.76e+00  3 0.29
## segment             2.37e+00  2 0.31
## order_priority      3.43e+00  3 0.33
## GLOBAL              9.86e+00 12 0.63
```

```
cox_improved <- step(cox_model_improved)
```

```
## Start:  AIC=5174.88
## survival ~ avg_discount + avg_cost_per_unit + avg_profit_per_unit +
##     avg_ship_delay + ship + segment + order_priority
##
##                       Df    AIC
## - avg_profit_per_unit  1 5173.6
## <none>                   5174.9
## - segment              2 5176.3
## - order_priority       3 5182.4
## - avg_discount         1 5187.4
## - avg_cost_per_unit    1 5187.9
## - avg_ship_delay       1 5188.1
## - ship                 3 5222.4
##
## Step:  AIC=5173.63
## survival ~ avg_discount + avg_cost_per_unit + avg_ship_delay +
##     ship + segment + order_priority
##
##                     Df    AIC
## <none>                 5173.6
## - segment            2 5174.9
## - order_priority     3 5181.0
## - avg_cost_per_unit  1 5186.7
## - avg_ship_delay     1 5187.0
## - avg_discount       1 5203.3
## - ship               3 5221.9
```

```
summary(cox_improved)
```

```
## Call:
## coxph(formula = survival ~ avg_discount + avg_cost_per_unit +
##     avg_ship_delay + ship + segment + order_priority, data = customer.df)
##
##   n= 1590, number of events= 374
##
##                             coef exp(coef)  se(coef)      z Pr(>|z|)
## avg_discount            2.839867 17.113492  0.485660  5.847 4.99e-09 ***
## avg_cost_per_unit       0.007202  1.007228  0.001764  4.082 4.47e-05 ***
## avg_ship_delay          0.005984  1.006002  0.001553  3.853 0.000116 ***
## shipSame Day           -0.053260  0.948134  0.347372 -0.153 0.878145
## shipSecond Class        0.064029  1.066124  0.212861  0.301 0.763565
## shipStandard Class     -0.979621  0.375453  0.195647 -5.007 5.53e-07 ***
## segmentCorporate        0.084720  1.088412  0.121937  0.695 0.487189
## segmentHome Office      0.316188  1.371888  0.135087  2.341 0.019252 *
## order_priorityCritical -0.094065  0.910224  0.282298 -0.333 0.738975
```

```
## order_priorityLow       1.147081  3.148989  0.466309  2.460 0.013897 *
## order_priorityMedium   -0.341246  0.710884  0.128359 -2.659 0.007848 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                        exp(coef) exp(-coef) lower .95 upper .95
## avg_discount             17.1135    0.05843    6.6061   44.3336
## avg_cost_per_unit         1.0072    0.99282    1.0038    1.0107
## avg_ship_delay            1.0060    0.99403    1.0029    1.0091
## shipSame Day              0.9481    1.05470    0.4799    1.8731
## shipSecond Class          1.0661    0.93798    0.7025    1.6181
## shipStandard Class        0.3755    2.66345    0.2559    0.5509
## segmentCorporate          1.0884    0.91877    0.8570    1.3822
## segmentHome Office        1.3719    0.72892    1.0528    1.7877
## order_priorityCritical    0.9102    1.09863    0.5234    1.5829
## order_priorityLow         3.1490    0.31756    1.2625    7.8541
## order_priorityMedium      0.7109    1.40670    0.5528    0.9142
##
## Concordance= 0.673  (se = 0.016 )
## Likelihood ratio test= 168.2  on 11 df,   p=<2e-16
## Wald test            = 202.5  on 11 df,   p=<2e-16
## Score (logrank) test = 221.6  on 11 df,   p=<2e-16
```

```
cox_test <- cox.zph(cox_improved)
print(cox_test)
```

```
##                      chisq df    p
## avg_discount      0.005084  1 0.94
## avg_cost_per_unit 0.000169  1 0.99
## avg_ship_delay    0.012075  1 0.91
## ship              3.546444  3 0.31
## segment           2.378640  2 0.30
## order_priority    3.309601  3 0.35
## GLOBAL            9.695146 11 0.56
```

```
summary(cox_improved)
```

```
## Call:
## coxph(formula = survival ~ avg_discount + avg_cost_per_unit +
##     avg_ship_delay + ship + segment + order_priority, data = customer.df)
##
##   n= 1590, number of events= 374
##
##                            coef exp(coef)  se(coef)      z Pr(>|z|)
## avg_discount           2.839867 17.113492  0.485660  5.847 4.99e-09 ***
## avg_cost_per_unit      0.007202  1.007228  0.001764  4.082 4.47e-05 ***
## avg_ship_delay         0.005984  1.006002  0.001553  3.853 0.000116 ***
## shipSame Day          -0.053260  0.948134  0.347372 -0.153 0.878145
## shipSecond Class       0.064029  1.066124  0.212861  0.301 0.763565
## shipStandard Class    -0.979621  0.375453  0.195647 -5.007 5.53e-07 ***
## segmentCorporate       0.084720  1.088412  0.121937  0.695 0.487189
## segmentHome Office     0.316188  1.371888  0.135087  2.341 0.019252 *
## order_priorityCritical -0.094065  0.910224  0.282298 -0.333 0.738975
## order_priorityLow       1.147081  3.148989  0.466309  2.460 0.013897 *
## order_priorityMedium   -0.341246  0.710884  0.128359 -2.659 0.007848 **
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##                         exp(coef) exp(-coef) lower .95 upper .95
## avg_discount              17.1135    0.05843    6.6061   44.3336
## avg_cost_per_unit          1.0072    0.99282    1.0038    1.0107
## avg_ship_delay             1.0060    0.99403    1.0029    1.0091
## shipSame Day               0.9481    1.05470    0.4799    1.8731
## shipSecond Class           1.0661    0.93798    0.7025    1.6181
## shipStandard Class         0.3755    2.66345    0.2559    0.5509
## segmentCorporate           1.0884    0.91877    0.8570    1.3822
## segmentHome Office         1.3719    0.72892    1.0528    1.7877
## order_priorityCritical     0.9102    1.09863    0.5234    1.5829
## order_priorityLow          3.1490    0.31756    1.2625    7.8541
## order_priorityMedium       0.7109    1.40670    0.5528    0.9142
##
## Concordance= 0.673  (se = 0.016 )
## Likelihood ratio test= 168.2  on 11 df,   p=<2e-16
## Wald test            = 202.5  on 11 df,   p=<2e-16
## Score (logrank) test = 221.6  on 11 df,   p=<2e-16
```

a. Average Discount (avg_discount) Coefficient: 2.62515 | Hazard Ratio (HR): 13.81 Interpretation: A 1-unit increase in average discount increases the hazard of churn by approximately 13.8 times. Action: Avoid over-reliance on discounts as a retention tool. Instead, use them strategically to reward loyal customers or incentivize first-time buyers.

b. Average Cost Per Unit (avg_cost_per_unit) Coefficient: 0.00720 | HR: 1.007 Interpretation: A 1-unit increase in average cost per unit increases the hazard of churn by 0.7%. Action: High-cost products may discourage repeat purchases. Consider bundling expensive products with value-add items or offering loyalty rewards for reeat purchases.

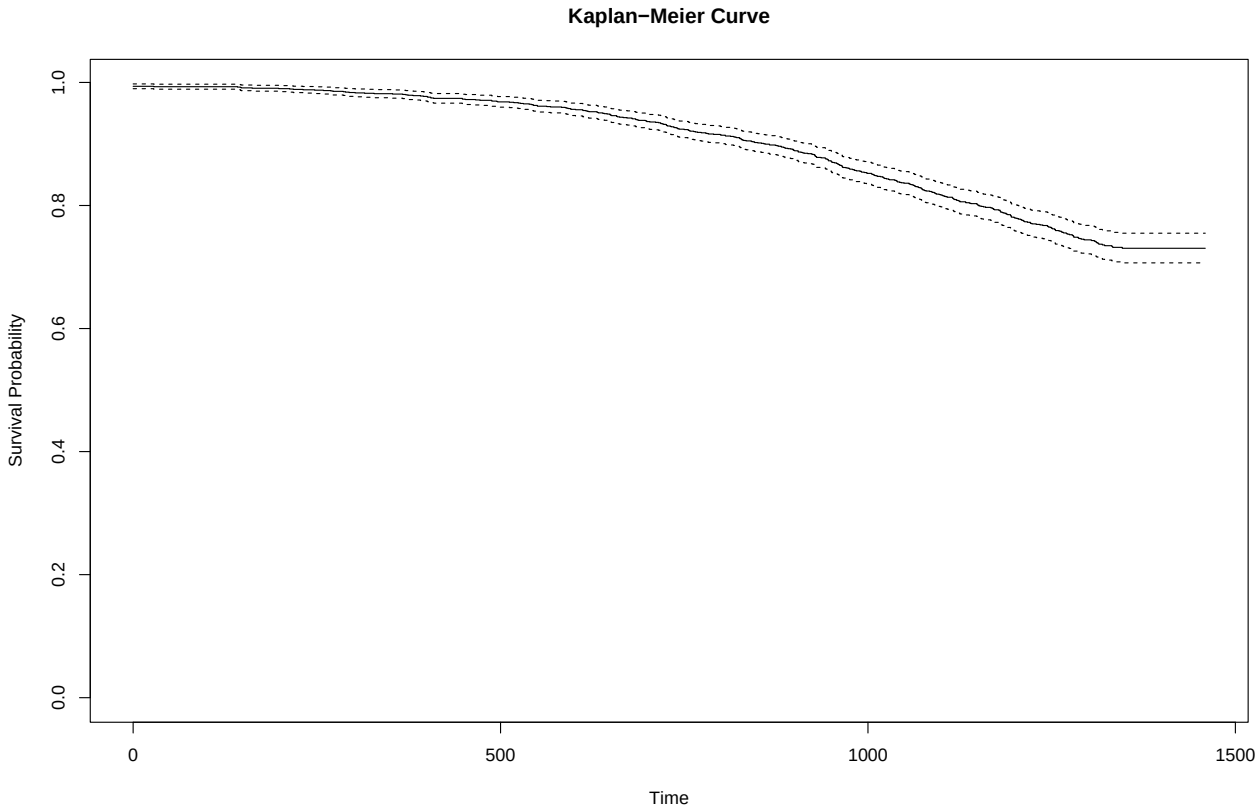
c. Average Shipping Delay (avg_ship_delay) Coefficient: 0.00264 | HR: 1.003 Interpretation: A 1-day increase in shipping delay raises the hazard of churn by 0.3%. Action: Streamline shipping operations to minimize delays, particularly in regions or for products where delays are frequent. (#oh lets look into this)

d. Shipping Mode: Standard Class (shipStandard Class) Coefficient: -0.82468 | HR: 0.44 Interpretation: Customers using Standard Class shipping have a 56% lower hazard of churn compared to the First Class reference group. Action: Promote Standard Class as a reliable, cost-effective shipping option. Highlight it in marketing materials and ensure its performance meets customer expectations.

e. Order Priority: Medium (order_priorityMedium) Coefficient: -0.36552 | HR: 0.69 Interpretation: Medium-priority orders have a 31% lower hazard of churn compared to High-priority orders. Action: Position Medium-priority orders as a balanced option for customers who value consistent, timely service without the premium pricing.

f. Order Priority: Low (order_priorityLow) ## too low base size

```r
surv_fit <- survfit(Surv(followtime, churn) ~ 1, data = customer.df)
plot(surv_fit, xlab = "Time", ylab = "Survival Probability",
     main = "Kaplan-Meier Curve")
```



For these variables that violated the proportional assumption:

```
tau <- quantile(customer.df$followtime, 0.50, na.rm = TRUE)
print(tau) #will use as rmean
```

```
##    50%
## 1262.5
```

```
# RMST estimation
fit_rmst <- survfit(survival ~ market, data = customer.df)
print(fit_rmst,print.rmean=getOption("survfit.print.rmean"),rmean=1262)
```

```
## Call: survfit(formula = survival ~ market, data = customer.df)
##
##                 n events rmean* se(rmean) median 0.95LCL 0.95UCL
## market=Africa 362    146   1075    15.353   1310    1221      NA
## market=APAC   225      4   1261     0.405     NA      NA      NA
## market=Canada   6      3    936   153.326   1203     817      NA
## market=EMEA   427    201   1049    14.608   1208    1151    1270
## market=EU     174      3   1261     1.016     NA      NA      NA
## market=LATAM  204      7   1257     2.861     NA      NA      NA
## market=US     192     10   1257     2.903     NA      NA      NA
##     * restricted mean with upper limit =  1262
```

```
# ! We want to compare to the maret with the highest retention
customer.df$market <- relevel(customer.df$market, ref = "APAC")
```

```
# Create pseudo-observations for RMST
tau <- 1262
customer.df$pseudos <- pseudomean(customer.df$followtime, customer.df$churn, tau)
```

```r
# Add unique IDs for each observation
customer.df$id <- 1:nrow(customer.df)

# View the first few rows
head(customer.df)
```

```
## # A tibble: 6 x 53
##   Customer.ID customer_name first_order_date last_order_date total_sales
##   <fct>       <chr>         <date>           <date>                <dbl>
## 1 AA-10315    Alex Avila    2011-03-31       2014-12-23           13747.
## 2 AA-10375    Allen Armold  2011-04-21       2014-12-25            5884.
## 3 AA-10480    Andrew Allen  2011-01-11       2014-09-05           17696.
## 4 AA-10645    Anna Andreadi 2011-01-12       2014-12-05           15344.
## 5 AA-315      Alex Avila    2011-08-06       2014-12-29            2243.
## 6 AA-375      Allen Armold  2011-01-06       2014-07-03             654.
## # i 48 more variables: avg_sales <dbl>, avg_quantity <dbl>,
## #   avg_shipping_cost <dbl>, avg_discount <dbl>, monetary <dbl>,
## #   total_cost <dbl>, avg_cost_per_unit <dbl>, avg_profit_per_unit <dbl>,
## #   avg_ship_delay <dbl>, followtime <dbl>, ship_mode <fct>,
## #   segment_mode <fct>, city_mode <fct>, state_mode <fct>, country_mode <fct>,
## #   market_mode <fct>, region_mode <fct>, category_mode <fct>,
## #   subcategory_mode <fct>, product_mode <fct>, order_priority_mode <fct>, ...
```

```r
# Fit a GEE regression model
fit <- geese(
  pseudos ~ discount_group + avg_ship_delay + frequency + market + category,
  data = customer.df,
  id = customer.df$id,  # Unique identifier for clusters
  family = gaussian,    # For RMST analysis
  corstr = "independence",  # Assuming no correlation within clusters
  jack = TRUE           # Use jackknife for variance estimation
)

summary(fit)
```

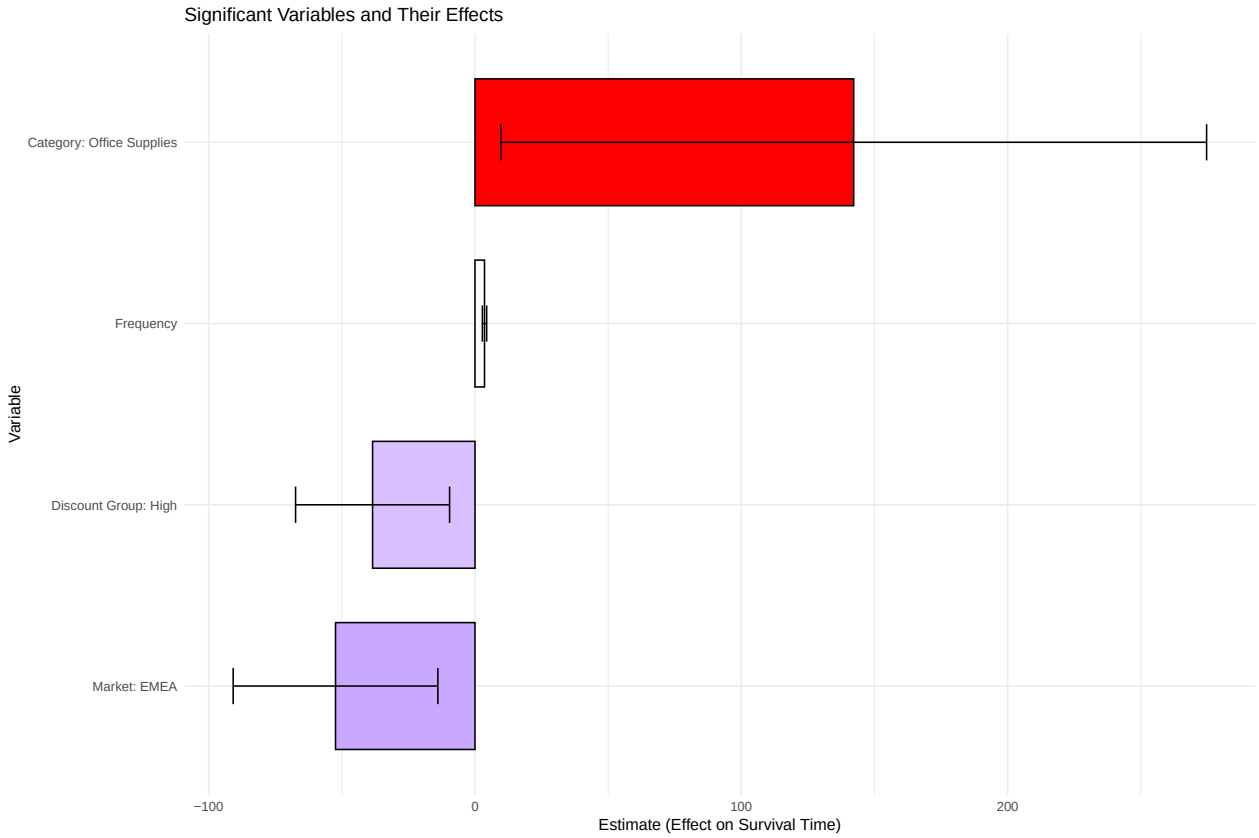
```
##
## Call:
## geese(formula = pseudos ~ discount_group + avg_ship_delay + frequency +
##     market + category, id = customer.df$id, data = customer.df,
##     family = gaussian, corstr = "independence", jack = TRUE)
##
## Mean Model:
##  Mean Link:                 identity
##  Variance to Mean Relation: gaussian
##
##  Coefficients:
##                            estimate      san.se      ajs.se        wald
## (Intercept)             929.0141392  69.1629037  72.2622262 180.4256222
## discount_groupLow       -35.5951107  13.1009195  13.1850473   7.3820527
## discount_groupHigh      -34.4743654  14.7962842  14.8904391   5.4285924
## avg_ship_delay           -0.4858436   0.1954190   0.1981129   6.1810070
## frequency                 3.5214053   0.4113087   0.4137689  73.2986793
## marketAfrica            -11.9691128  19.7899961  19.9112755   0.3657906
## marketCanada            -81.7389181 151.5311936 180.8064294   0.2909737
## marketEMEA              -35.8789004  19.2895737  19.4082862   3.4596566
```

```
## marketEU                   4.0808903   5.2863723   5.3148279   0.5959288
## marketLATAM                2.7563528   5.0244917   5.0603287   0.3009438
## marketUS                  -3.0403746   5.4593927   5.4822363   0.3101455
## categoryOffice Supplies 161.9288922  67.7535245  70.8719506   5.7119567
## categoryTechnology      -55.6354801  90.1398124  93.5202143   0.3809521
##                                    p
## (Intercept)             0.000000000
## discount_groupLow       0.006587794
## discount_groupHigh      0.019809641
## avg_ship_delay          0.012912877
## frequency               0.000000000
## marketAfrica            0.545307760
## marketCanada            0.589597258
## marketEMEA              0.062883374
## marketEU                0.440135614
## marketLATAM             0.583291371
## marketUS                0.577590900
## categoryOffice Supplies 0.016849748
## categoryTechnology      0.537094125
##
## Scale Model:
##  Scale Link:                identity
##
##  Estimated Scale Parameters:
##             estimate   san.se   ajs.se     wald p
## (Intercept) 39891.88 2940.161 3005.469 184.0886 0
##
## Correlation Model:
##  Correlation Structure:     independence
##
## Returned Error Value:    0
## Number of clusters:   1590   Maximum cluster size: 1
```

- Interpretation by Variable Intercept Estimate: 938.044 p: < 0.00001 (highly significant) Interpretation: The baseline survival time (or RMST) when all other covariates are at their reference levels.
- High Discount: Estimate: -38.452 | p: 0.0091 (significant) Customers in the High Discount group have an average survival time that is 38.5 days shorter than the reference group. Interpretation: Discount high, reduce survival times. This suggests that discounting may attract less loyal or transactional customers.
- Frequency Estimate: 3.574 | p: < 0.00001 (highly significant) Interpretation: Each additional purchase increases survival time by approximately 3.57 days. This highlights the importance of encouraging frequent purchases to enhance retention.
- Market EMEA: Estimate: -52.369 | p: 0.00756 (significant) Customers in the EMEA market have an average survival time that is 48.9 days shorter than our high retention market, APAC. Other Markets: The EMEA market has significantly lower survival times, suggesting region-specific challenges (e.g., operational inefficiencies or product-market misfit).
- Order priority levels do not significantly influence survival in this model.
- Category Office Supplies: Estimate: 142.185 | p: 0.035 (significant) Customers purchasing Office Supplies have an average survival time 142.2 days longer than the reference category. Interpretation: The Office Supplies category contributes to longer retention, possibly due to higher repeat purchase rates or necessity-driven demand.

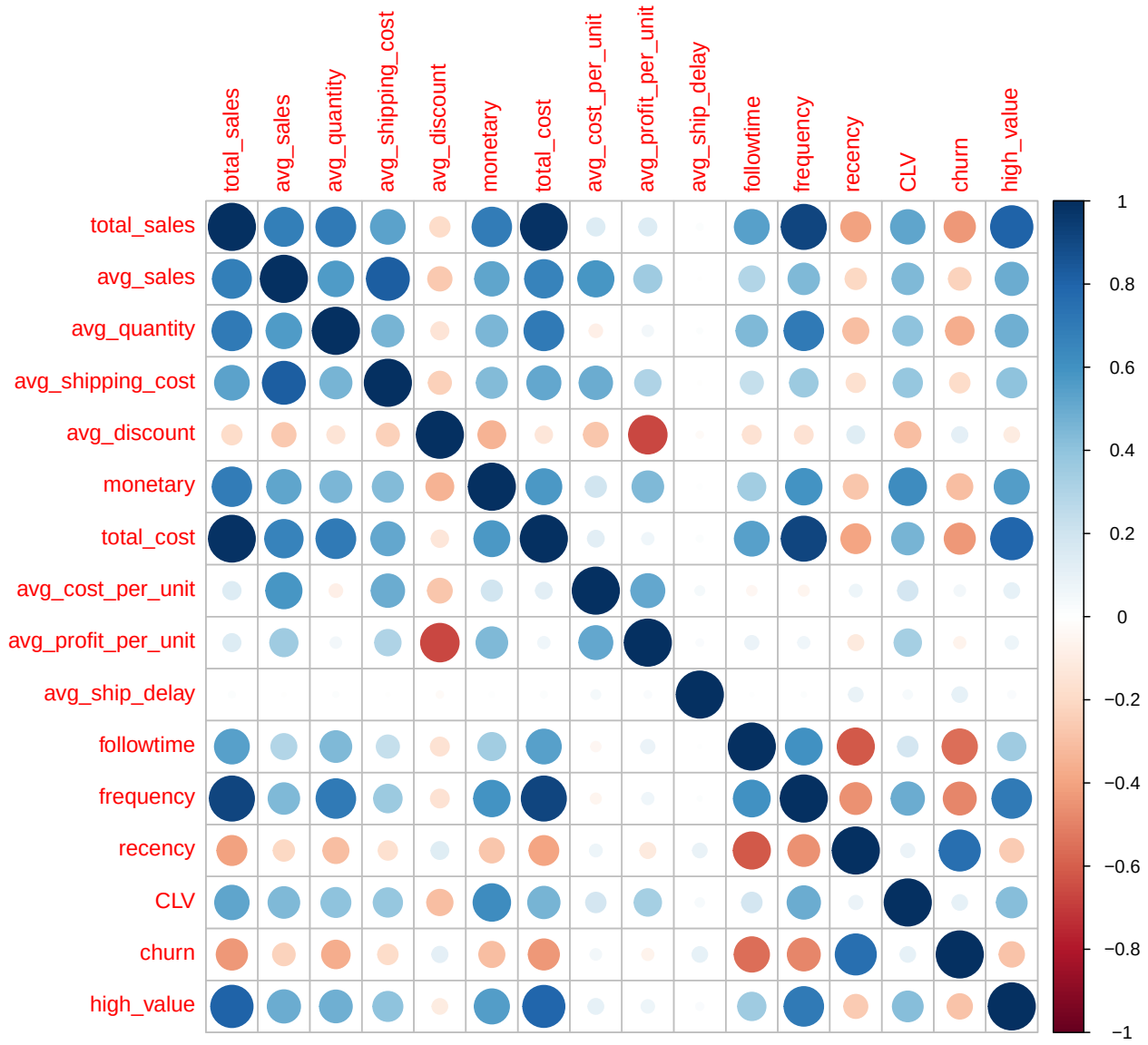
Now, let us see what is making people in APAC retained and why EMEA is churning.

```
# Create a data frame with the provided coefficients and standard errors
coefficients <- data.frame(
  Variable = c(  "Discount Group: High",
                 "Avg Ship Delay", "Frequency", "Market: US",
                 "Market: Africa", "Market: Canada",
                 "Market: EMEA", "Market: EU", "Category: Office Supplies"),
  Estimate = c(  -38.452, -0.351, 3.574,
                 -3.476, -19.108, -79.417, -52.369, 3.380, 142.185),
  SE = c( 14.751, 0.195, 0.412,
          5.589, 20.064, 148.540, 19.606, 5.440, 67.609),
  PValue = c(  0.00914, 0.07203, 0,
               0.53399, 0.34091, 0.59289, 0.00756, 0.53434,0.03546)
)

# Calculate confidence intervals
coefficients$Lower_CI <- coefficients$Estimate - 1.96 * coefficients$SE
coefficients$Upper_CI <- coefficients$Estimate + 1.96 * coefficients$SE

# Filter for significant variables (p < 0.05)
significant_vars <- coefficients[coefficients$PValue < 0.05, ]

# Plot significant variables with error bars
ggplot(significant_vars, aes(x = reorder(Variable, Estimate),
                             y = Estimate, fill = Estimate)) +
  geom_bar(stat = "identity", color = "black", width = 0.7) +
  geom_errorbar(aes(ymin = Lower_CI, ymax = Upper_CI), width = 0.2) +
  coord_flip() +  # Flip coordinates for better readability
  labs(
    title = "Significant Variables and Their Effects",
    x = "Variable",
    y = "Estimate (Effect on Survival Time)"
  ) +
  theme_minimal() +
  scale_fill_gradient2(low = "blue", high = "red",
                       mid = "white", midpoint = 0) +
  theme(legend.position = "none")
```



Linear Regression

```
# Step 1: Discover patterns using correlation for numeric variables
cor_matrix <- cor(final_df[, sapply(final_df,
                                    function(x) is.numeric(x) || is.integer(x))],
                  use = "complete.obs")
corrplot(cor_matrix, method = "circle")
```




```r
#find the variable highly correlated with CLV
cor_matrix <- cor(final_df[, sapply(final_df,
                                    function(x) is.numeric(x) || is.integer(x))],
                  use = "complete.obs")
cor_matrix <- as.data.frame(as.table(cor_matrix))
cor_matrix <- cor_matrix[order(-abs(cor_matrix$Freq)), ]
cor_matrix <- cor_matrix[cor_matrix$Var1 == "CLV", ]

model_1 <- lm(CLV ~   freq_index  +
                avg_quantity  + avg_shipping_cost +
                avg_profit_per_unit +
                rec_index +  avg_discount +  followtime +
                avg_cost_per_unit +
                recency +
                avg_ship_delay +  quarter_mode +  year_mode  + month_mode,
              data = final_df)
summary(model_1)
```

```
##
## Call:
## lm(formula = CLV ~ freq_index + avg_quantity + avg_shipping_cost +
##     avg_profit_per_unit + rec_index + avg_discount + followtime +
##     avg_cost_per_unit + recency + avg_ship_delay + quarter_mode +
##     year_mode + month_mode, data = final_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.2642  -2.4907   0.3154   3.8785  23.2100
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)         17.5000312  7.2418993   2.416 0.015785 *
## freq_index2          6.1140184  0.4201808  14.551  < 2e-16 ***
## freq_index3         16.2264289  0.5885676  27.569  < 2e-16 ***
## freq_index4         22.6433272  0.6068499  37.313  < 2e-16 ***
## avg_quantity         0.6008551  0.2473505   2.429 0.015246 *
## avg_shipping_cost    0.1108220  0.0153286   7.230 7.55e-13 ***
## avg_profit_per_unit  0.2020407  0.0139439  14.489  < 2e-16 ***
## rec_index3           7.9322501  0.3942251  20.121  < 2e-16 ***
## rec_index2          15.7693171  0.4108206  38.385  < 2e-16 ***
## rec_index1          23.6875770  0.6065717  39.052  < 2e-16 ***
## avg_discount        -6.2596628  1.7164109  -3.647 0.000274 ***
## followtime           0.0011215  0.0007281   1.540 0.123705
## avg_cost_per_unit   -0.0050824  0.0065191  -0.780 0.435735
## recency              0.0008643  0.0018740   0.461 0.644718
## avg_ship_delay      -0.0014895  0.0039200  -0.380 0.704007
## quarter_mode2       -0.9064881  0.6565035  -1.381 0.167544
## quarter_mode3       -0.8297088  0.6244516  -1.329 0.184142
## quarter_mode4       -0.3435423  0.6175057  -0.556 0.578059
## year_mode2012        0.6256127  0.5404876   1.157 0.247247
## year_mode2013        0.6698462  0.4983882   1.344 0.179136
## year_mode2014        0.6462039  0.4835820   1.336 0.181651
## month_mode02        -0.4449288  0.9014310  -0.494 0.621672
## month_mode03         0.0798988  0.8277288   0.097 0.923114
## month_mode04         0.2758337  0.9333302   0.296 0.767623
## month_mode05         0.9894721  0.9098755   1.087 0.276993
## month_mode06         1.1016060  0.8657603   1.272 0.203416
## month_mode07         0.7021280  0.9202331   0.763 0.445585
## month_mode08         0.9574095  0.8680898   1.103 0.270244
## month_mode09         0.2581742  0.8476712   0.305 0.760735
## month_mode10         0.5569953  0.8905020   0.625 0.531745
## month_mode11         0.6921166  0.8227052   0.841 0.400326
## month_mode12        -0.2002613  0.8175429  -0.245 0.806523
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.455 on 1558 degrees of freedom
## Multiple R-squared:  0.7896, Adjusted R-squared:  0.7854
## F-statistic: 188.6 on 31 and 1558 DF,  p-value: < 2.2e-16
```

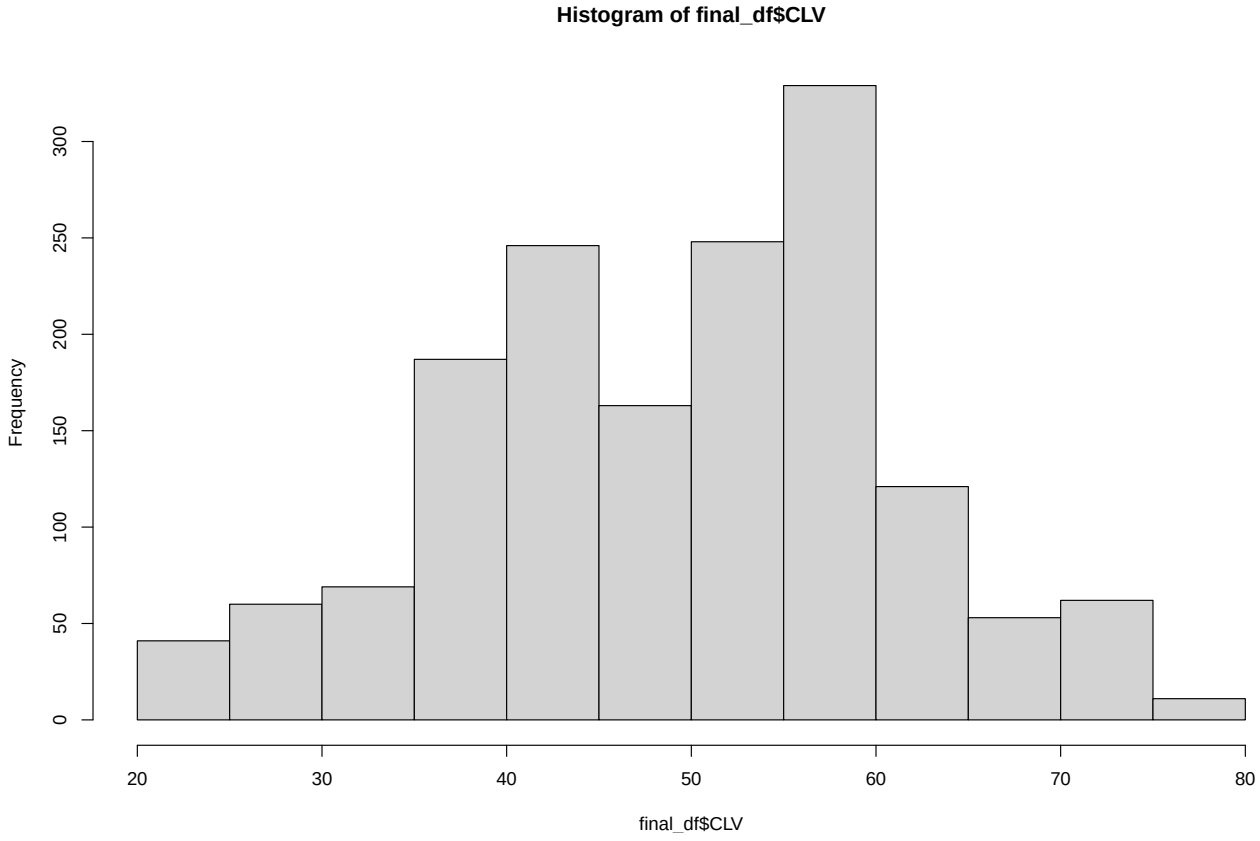
```
vif(model_1)
```

```
##                         GVIF Df GVIF^(1/(2*Df))
```

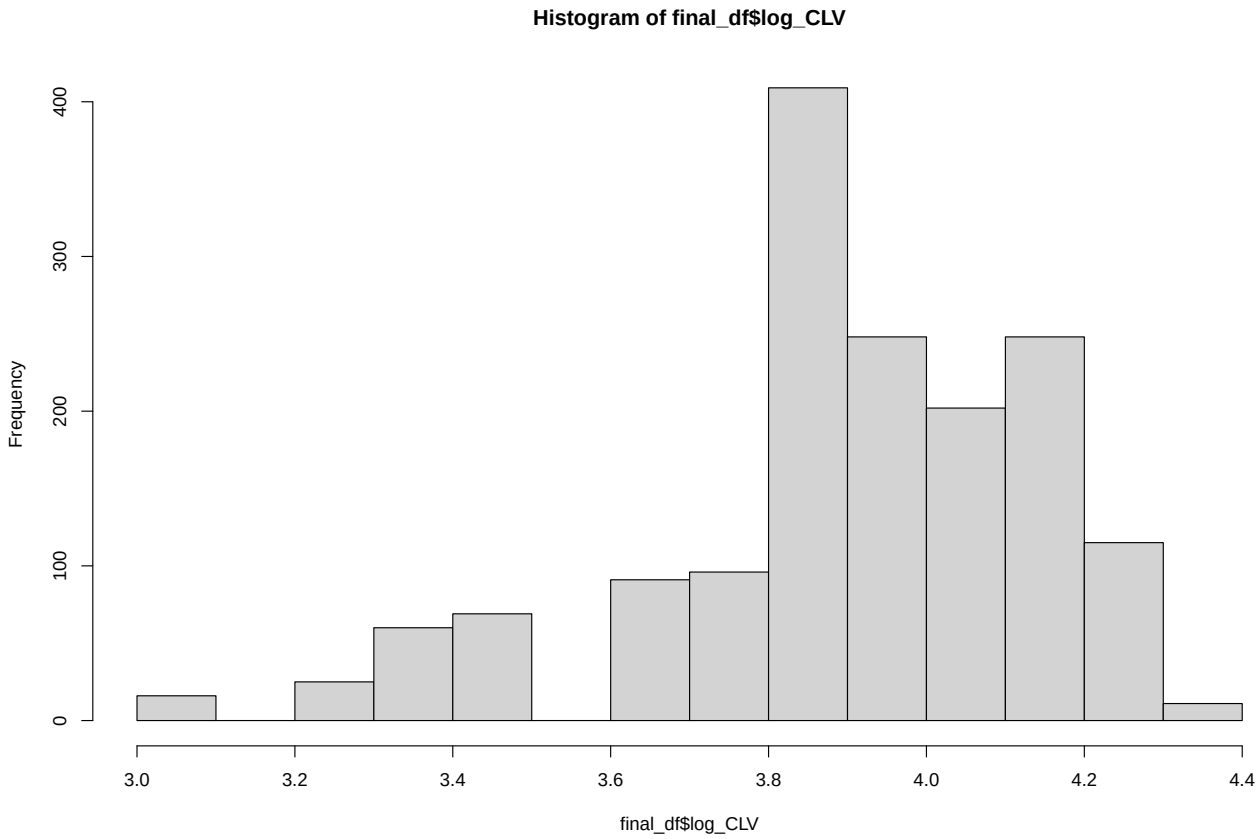
```
## freq_index          3.754428  3        1.246687
## avg_quantity        2.857687  1        1.690469
## avg_shipping_cost   2.047858  1        1.431034
## avg_profit_per_unit 2.335138  1        1.528116
## rec_index           3.057278  3        1.204728
## avg_discount        1.885199  1        1.373025
## followtime          2.382836  1        1.543644
## avg_cost_per_unit   2.047684  1        1.430973
## recency             3.066447  1        1.751127
## avg_ship_delay      1.193835  1        1.092627
## quarter_mode        9.383390  3        1.452312
## year_mode           1.242586  3        1.036862
## month_mode          9.968931 11        1.110179
```

This was to see which numerical variables would be relevant excluding the ones with high multicolinearity. The model was then run with the variables that had a correlation with CLV.

```
#shows skewness but CLV was relatively evenly distributed
hist(final_df$CLV)
```




```
final_df$log_CLV <- log(final_df$CLV + 1)
hist(final_df$log_CLV)
```




```
model<- lm(CLV ~ segment_mode + category_mode + market_mode +
             avg_shipping_cost + order_priority_mode +
             avg_profit_per_unit + avg_discount + followtime + year_mode,
           data = final_df)
summary(model)
```

```
##
## Call:
## lm(formula = CLV ~ segment_mode + category_mode + market_mode +
##     avg_shipping_cost + order_priority_mode + avg_profit_per_unit +
##     avg_discount + followtime + year_mode, data = final_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.8260  -5.8334   0.6462   6.6599  27.1582
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  45.751981   3.094712  14.784  < 2e-16 ***
## segment_modeCorporate         0.001786   0.550014   0.003  0.99741
## segment_modeHome Office       0.388813   0.646003   0.602  0.54734
## category_modeOffice Supplies  4.743142   2.063548   2.299  0.02166 *
## category_modeTechnology       5.706740   2.523493   2.261  0.02387 *
## market_modeAPAC              12.093049   0.930037  13.003  < 2e-16 ***
## market_modeCanada            -6.879041   3.918925  -1.755  0.07940 .
## market_modeEMEA               1.796671   0.680857   2.639  0.00840 **
## market_modeEU                12.966586   0.981572  13.210  < 2e-16 ***
```

```
## market_modeLATAM             11.024616   0.939394  11.736  < 2e-16 ***
## market_modeUS                12.953960   0.942677  13.742  < 2e-16 ***
## avg_shipping_cost             0.134135   0.021799   6.153 9.62e-10 ***
## order_priority_modeHigh      -1.409934   1.825453  -0.772  0.44001
## order_priority_modeLow       -1.053852   3.375025  -0.312  0.75489
## order_priority_modeMedium    -1.471004   1.782345  -0.825  0.40932
## avg_profit_per_unit           0.181743   0.022068   8.235 3.72e-16 ***
## avg_discount                 -7.824863   2.976200  -2.629  0.00864 **
## followtime                   -0.006347   0.001051  -6.039 1.93e-09 ***
## year_mode2012                -1.355845   0.927630  -1.462  0.14405
## year_mode2013                -0.551365   0.847125  -0.651  0.51523
## year_mode2014                -3.348001   0.799330  -4.189 2.96e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.462 on 1569 degrees of freedom
## Multiple R-squared:  0.3626, Adjusted R-squared:  0.3545
## F-statistic: 44.63 on 20 and 1569 DF,  p-value: < 2.2e-16
```

```
model_step <- step(model)
```

```
## Start:  AIC=7167.27
## CLV ~ segment_mode + category_mode + market_mode + avg_shipping_cost +
##     order_priority_mode + avg_profit_per_unit + avg_discount +
##     followtime + year_mode
##
##                       Df Sum of Sq    RSS    AIC
## - order_priority_mode  3      62.4 140538 7162.0
## - segment_mode         2      35.8 140512 7163.7
## <none>                             140476 7167.3
## - category_mode        2     520.8 140997 7169.2
## - avg_discount         1     618.9 141095 7172.3
## - year_mode            3    2955.0 143431 7194.4
## - followtime           1    3265.4 143741 7201.8
## - avg_shipping_cost    1    3389.8 143866 7203.2
## - avg_profit_per_unit  1    6072.3 146548 7232.6
## - market_mode          6   28398.6 168875 7448.0
##
## Step:  AIC=7161.98
## CLV ~ segment_mode + category_mode + market_mode + avg_shipping_cost +
##     avg_profit_per_unit + avg_discount + followtime + year_mode
##
##                       Df Sum of Sq    RSS    AIC
## - segment_mode         2      35.4 140574 7158.4
## <none>                             140538 7162.0
## - category_mode        2     519.6 141058 7163.8
## - avg_discount         1     601.6 141140 7166.8
## - year_mode            3    2995.7 143534 7189.5
## - followtime           1    3314.7 143853 7197.0
## - avg_shipping_cost    1    3665.0 144203 7200.9
## - avg_profit_per_unit  1    6026.9 146565 7226.7
## - market_mode          6   29193.5 169732 7450.1
##
## Step:  AIC=7158.38
## CLV ~ category_mode + market_mode + avg_shipping_cost + avg_profit_per_unit +
```

```
##     avg_discount + followtime + year_mode
##
##                        Df Sum of Sq    RSS    AIC
## <none>                              140574 7158.4
## - category_mode         2     516.1 141090 7160.2
## - avg_discount          1     603.9 141178 7163.2
## - year_mode             3    2988.0 143562 7185.8
## - followtime            1    3386.1 143960 7194.2
## - avg_shipping_cost     1    3655.6 144229 7197.2
## - avg_profit_per_unit   1    6047.5 146621 7223.3
## - market_mode           6   29354.3 169928 7447.9
```

```
summary(model_step)
```

```
##
## Call:
## lm(formula = CLV ~ category_mode + market_mode + avg_shipping_cost +
##     avg_profit_per_unit + avg_discount + followtime + year_mode,
##     data = final_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.9798  -5.8220   0.6366   6.7224  27.0767
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 44.485848   2.529190  17.589  < 2e-16 ***
## category_modeOffice Supplies  4.737130   2.057255   2.303  0.02143 *
## category_modeTechnology       5.633215   2.514798   2.240  0.02523 *
## market_modeAPAC              12.024201   0.915852  13.129  < 2e-16 ***
## market_modeCanada            -7.038443   3.905271  -1.802  0.07169 .
## market_modeEMEA               1.778878   0.677934   2.624  0.00878 **
## market_modeEU                12.887952   0.968533  13.307  < 2e-16 ***
## market_modeLATAM             10.954517   0.924391  11.851  < 2e-16 ***
## market_modeUS                12.879661   0.929810  13.852  < 2e-16 ***
## avg_shipping_cost             0.136847   0.021390   6.398 2.07e-10 ***
## avg_profit_per_unit           0.180890   0.021982   8.229 3.92e-16 ***
## avg_discount                 -7.686968   2.956011  -2.600  0.00940 **
## followtime                   -0.006431   0.001044  -6.157 9.36e-10 ***
## year_mode2012                -1.391946   0.925451  -1.504  0.13276
## year_mode2013                -0.564268   0.845392  -0.667  0.50457
## year_mode2014                -3.371243   0.797425  -4.228 2.50e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.45 on 1574 degrees of freedom
## Multiple R-squared:  0.3622, Adjusted R-squared:  0.3561
## F-statistic: 59.58 on 15 and 1574 DF,  p-value: < 2.2e-16
```

```
# Evaluate outliers
outliers <- outlierTest(model_step)
outliers
```

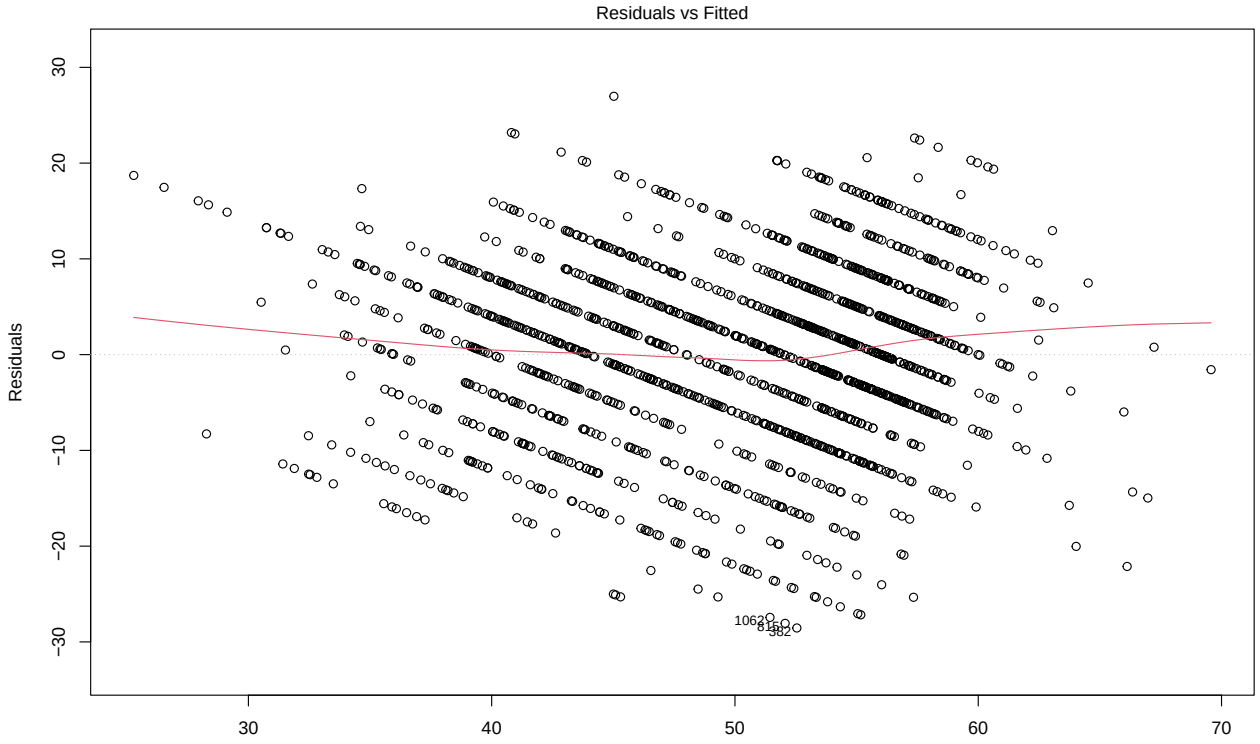
```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##      rstudent unadjusted p-value Bonferroni p
```

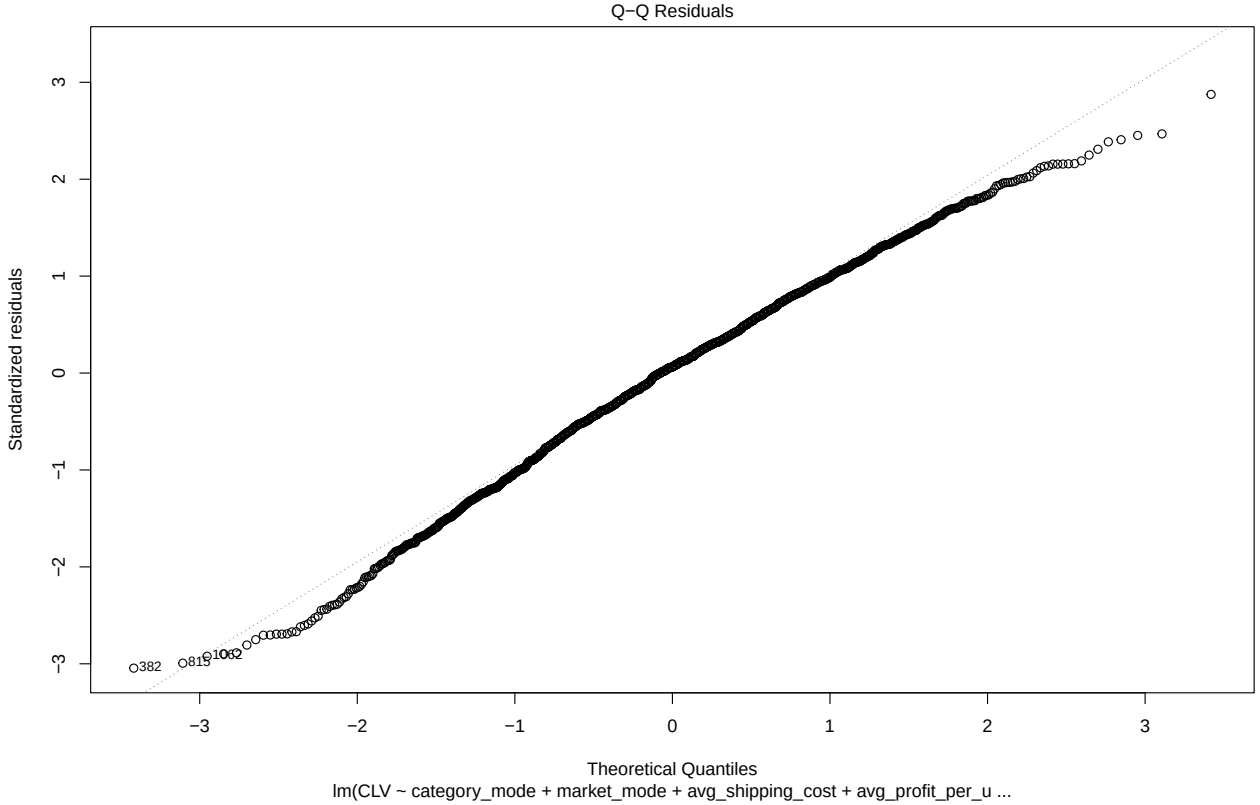
```
## 807 -3.094435          0.0020066           NA
```

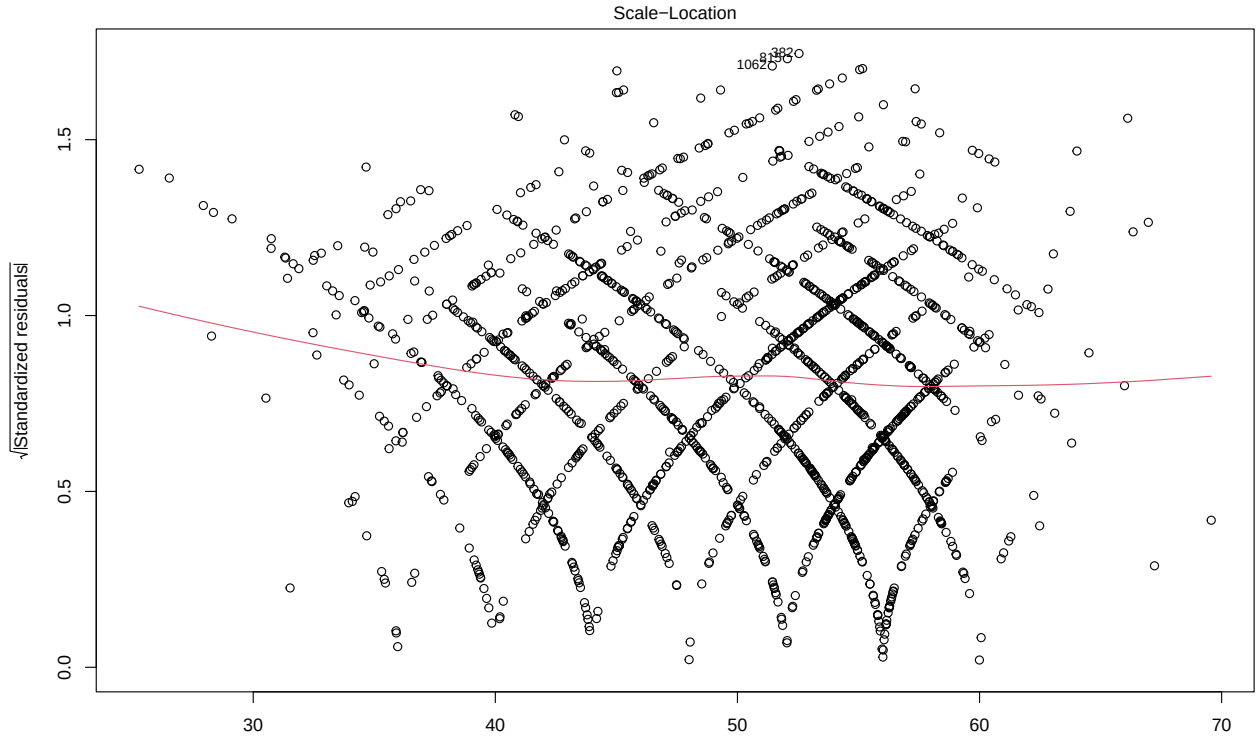
```r
# Remove outliers (if any exist)
if (!is.null(outliers)) {
  final_df <- final_df[-as.numeric(names(outliers$rstudent)), ]
}

# Re-run the model after removing outliers
model_2 <- lm(CLV ~ category_mode + market_mode + avg_shipping_cost +
                avg_profit_per_unit + avg_discount + followtime + year_mode,
              data = final_df)
summary(model_2)
```

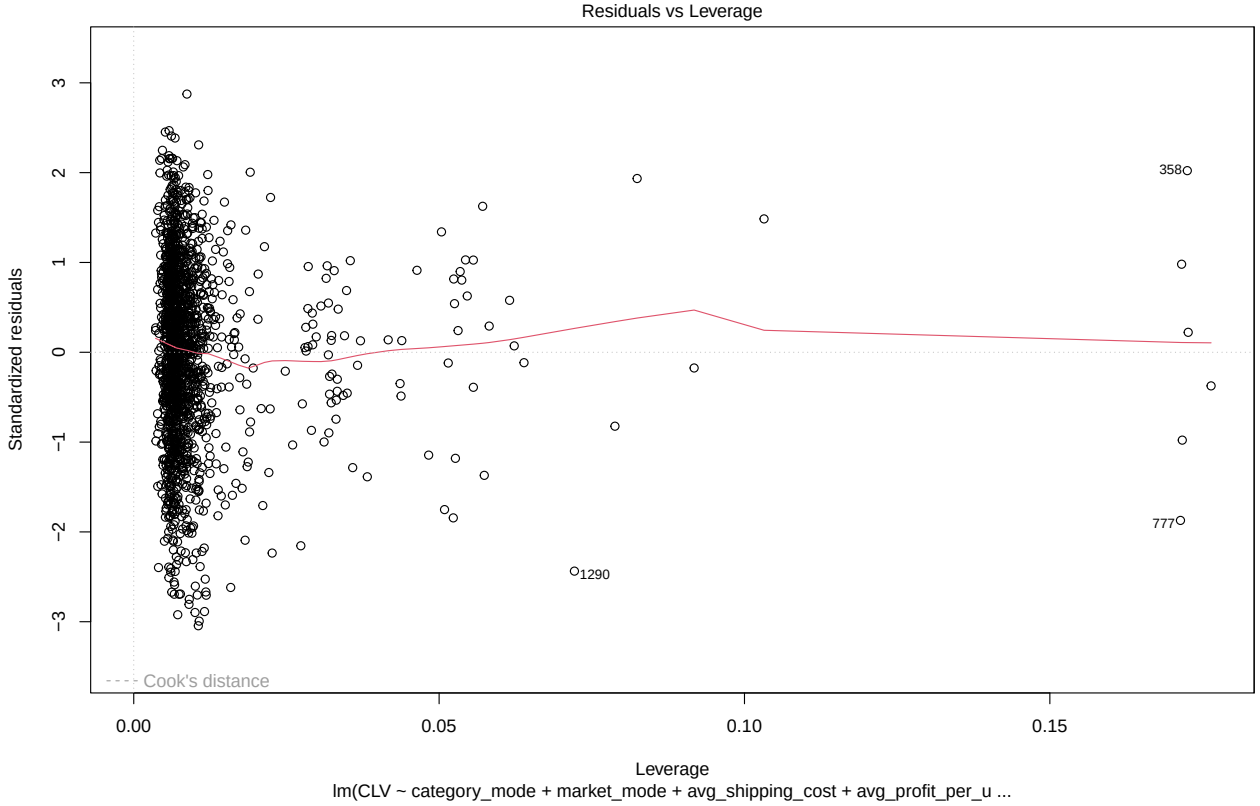
```
##
## Call:
## lm(formula = CLV ~ category_mode + market_mode + avg_shipping_cost +
##     avg_profit_per_unit + avg_discount + followtime + year_mode,
##     data = final_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.5441  -5.8611   0.5984   6.7468  26.9774
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  44.647209   2.522867  17.697  < 2e-16 ***
## category_modeOffice Supplies  4.753001   2.051680   2.317  0.02065 *
## category_modeTechnology       5.651749   2.507982   2.254  0.02436 *
## market_modeAPAC              12.137101   0.914096  13.278  < 2e-16 ***
## market_modeCanada            -7.066011   3.894686  -1.814  0.06983 .
## market_modeEMEA               1.780678   0.676095   2.634  0.00853 **
## market_modeEU                12.870233   0.965923  13.324  < 2e-16 ***
## market_modeLATAM             10.939749   0.921896  11.867  < 2e-16 ***
## market_modeUS                12.869142   0.927294  13.878  < 2e-16 ***
## avg_shipping_cost             0.139627   0.021351   6.540 8.31e-11 ***
## avg_profit_per_unit           0.177931   0.021944   8.109 1.02e-15 ***
## avg_discount                 -7.965733   2.949367  -2.701  0.00699 **
## followtime                   -0.006450   0.001042  -6.192 7.54e-10 ***
## year_mode2012                -1.550654   0.924364  -1.678  0.09364 .
## year_mode2013                -0.735325   0.844908  -0.870  0.38427
## year_mode2014                -3.537536   0.797075  -4.438 9.71e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.425 on 1573 degrees of freedom
## Multiple R-squared:  0.3647, Adjusted R-squared:  0.3586
## F-statistic: 60.19 on 15 and 1573 DF,  p-value: < 2.2e-16
```

```r
# Plot diagnostic plots for model_2
plot(model_2)
```


Residuals vs Fitted
Residuals
Fitted values
lm(CLV ~ category_mode + market_mode + avg_shipping_cost + avg_profit_per_u ...


Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(CLV ~ category_mode + market_mode + avg_shipping_cost + avg_profit_per_u ...






```r
# Step 3: Perform stepwise selection to choose independent variables
start_model <- lm(CLV ~ 1, data = final_df)  # Starting model with intercept only
scope <- ~ category_mode + market_mode + avg_shipping_cost +
```

```
  avg_profit_per_unit + avg_discount + followtime + year_mode

final_model <- step(start_model,
                    scope = scope,
                    direction = "both")
```

```
## Start:  AIC=7836.01
## CLV ~ 1
##
##                        Df Sum of Sq    RSS    AIC
## + market_mode           6     48288 171630 7454.1
## + avg_shipping_cost     1     32594 187324 7583.1
## + avg_profit_per_unit   1     24529 195390 7650.1
## + avg_discount          1     20826 199092 7679.9
## + followtime            1      7691 212227 7781.4
## + year_mode             3      1795 218124 7829.0
## + category_mode         2      1345 218573 7830.3
## <none>                              219918 7836.0
##
## Step:  AIC=7454.07
## CLV ~ market_mode
##
##                        Df Sum of Sq    RSS    AIC
## + avg_profit_per_unit   1     20533 151097 7253.6
## + avg_discount          1     12261 159369 7338.3
## + avg_shipping_cost     1     10742 160888 7353.4
## + year_mode             3      3189 168442 7430.3
## + followtime            1      2423 169207 7433.5
## <none>                              171630 7454.1
## + category_mode         2       213 171417 7456.1
## - market_mode           6     48288 219918 7836.0
##
## Step:  AIC=7253.6
## CLV ~ market_mode + avg_profit_per_unit
##
##                        Df Sum of Sq    RSS    AIC
## + avg_shipping_cost     1      3905 147192 7214.0
## + followtime            1      3353 147744 7219.9
## + year_mode             3      3324 147773 7224.3
## + avg_discount          1       439 150658 7251.0
## <none>                              151097 7253.6
## + category_mode         2       313 150784 7254.3
## - avg_profit_per_unit   1     20533 171630 7454.1
## - market_mode           6     44293 195390 7650.1
##
## Step:  AIC=7213.99
## CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost
##
##                        Df Sum of Sq    RSS    AIC
## + followtime            1    3258.2 143933 7180.4
## + year_mode             3    3079.7 144112 7186.4
## + avg_discount          1     510.7 146681 7210.5
## + category_mode         2     419.9 146772 7213.5
## <none>                              147192 7214.0
```

```
## - avg_shipping_cost    1    3905.0 151097 7253.6
## - avg_profit_per_unit  1   13695.9 160888 7353.4
## - market_mode          6   28959.0 176151 7487.4
##
## Step:  AIC=7180.43
## CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost +
##     followtime
##
##                       Df Sum of Sq    RSS    AIC
## + year_mode            3    3008.3 140925 7152.9
## + avg_discount         1     646.4 143287 7175.3
## + category_mode        2     501.2 143432 7178.9
## <none>                             143933 7180.4
## - followtime           1    3258.2 147192 7214.0
## - avg_shipping_cost    1    3810.0 147744 7219.9
## - avg_profit_per_unit  1   14470.7 158404 7330.6
## - market_mode          6   30237.9 174171 7471.4
##
## Step:  AIC=7152.86
## CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost +
##     followtime + year_mode
##
##                       Df Sum of Sq    RSS    AIC
## + avg_discount         1     682.4 140243 7147.2
## + category_mode        2     554.0 140371 7150.6
## <none>                             140925 7152.9
## - year_mode            3    3008.3 143933 7180.4
## - followtime           1    3186.7 144112 7186.4
## - avg_shipping_cost    1    3565.9 144491 7190.6
## - avg_profit_per_unit  1   14766.4 155692 7309.2
## - market_mode          6   30974.8 171900 7456.6
##
## Step:  AIC=7147.15
## CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost +
##     followtime + year_mode + avg_discount
##
##                       Df Sum of Sq    RSS    AIC
## + category_mode        2     519.6 139723 7145.3
## <none>                             140243 7147.2
## - avg_discount         1     682.4 140925 7152.9
## - year_mode            3    3044.3 143287 7175.3
## - followtime           1    3341.7 143585 7182.6
## - avg_shipping_cost    1    3640.8 143884 7185.9
## - avg_profit_per_unit  1    5752.7 145996 7209.0
## - market_mode          6   29880.5 170123 7442.1
##
## Step:  AIC=7145.25
## CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost +
##     followtime + year_mode + avg_discount + category_mode
##
##                       Df Sum of Sq    RSS    AIC
## <none>                             139723 7145.3
## - category_mode        2     519.6 140243 7147.2
## - avg_discount         1     647.9 140371 7150.6
```

```
## - year_mode            3    3095.7 142819 7174.1
## - followtime           1    3406.2 143129 7181.5
## - avg_shipping_cost    1    3798.8 143522 7185.9
## - avg_profit_per_unit  1    5840.2 145563 7208.3
## - market_mode          6   29439.8 169163 7437.1
```

```
summary(final_model)
```

```
##
## Call:
## lm(formula = CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost +
##     followtime + year_mode + avg_discount + category_mode, data = final_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.5441  -5.8611   0.5984   6.7468  26.9774
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 44.647209   2.522867  17.697  < 2e-16 ***
## market_modeAPAC             12.137101   0.914096  13.278  < 2e-16 ***
## market_modeCanada           -7.066011   3.894686  -1.814  0.06983 .
## market_modeEMEA              1.780678   0.676095   2.634  0.00853 **
## market_modeEU               12.870233   0.965923  13.324  < 2e-16 ***
## market_modeLATAM            10.939749   0.921896  11.867  < 2e-16 ***
## market_modeUS               12.869142   0.927294  13.878  < 2e-16 ***
## avg_profit_per_unit          0.177931   0.021944   8.109 1.02e-15 ***
## avg_shipping_cost            0.139627   0.021351   6.540 8.31e-11 ***
## followtime                  -0.006450   0.001042  -6.192 7.54e-10 ***
## year_mode2012               -1.550654   0.924364  -1.678  0.09364 .
## year_mode2013               -0.735325   0.844908  -0.870  0.38427
## year_mode2014               -3.537536   0.797075  -4.438 9.71e-06 ***
## avg_discount                -7.965733   2.949367  -2.701  0.00699 **
## category_modeOffice Supplies  4.753001   2.051680   2.317  0.02065 *
## category_modeTechnology       5.651749   2.507982   2.254  0.02436 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.425 on 1573 degrees of freedom
## Multiple R-squared:  0.3647, Adjusted R-squared:  0.3586
## F-statistic: 60.19 on 15 and 1573 DF,  p-value: < 2.2e-16
```

```
# Step 4: Check for multicollinearity
vif(final_model)  # Remove variables with VIF > 5 if necessary
```

```
##                          GVIF Df GVIF^(1/(2*Df))
## market_mode          1.866215  6        1.053368
## avg_profit_per_unit  1.936863  1        1.391712
## avg_shipping_cost    1.328538  1        1.152622
## followtime           1.633230  1        1.277979
## year_mode            1.081957  3        1.013215
## avg_discount         1.864737  1        1.365554
## category_mode        1.074570  2        1.018143
```

```
# Step 5: Make predictions
predictions <- predict(final_model, final_df)
```

```r
# Combine predictions with the original dataset
final_results <- cbind(final_df, pred = predictions)

# View the first few rows of the final dataset with predictions
head(final_results)
```

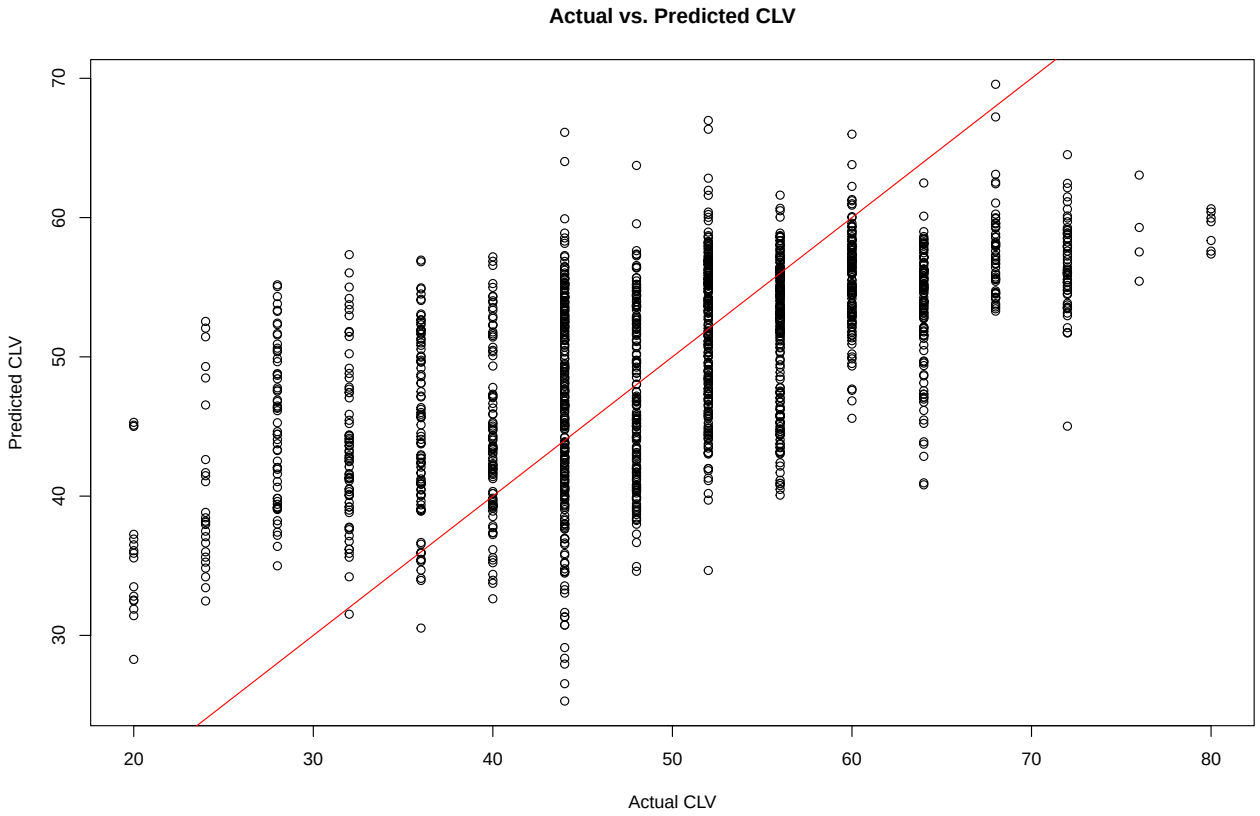
```
##   Customer.ID customer_name first_order_date last_order_date total_sales
## 1    AA-10315    Alex Avila       2011-03-31      2014-12-23   13747.413
## 2    AA-10375  Allen Armold       2011-04-21      2014-12-25    5884.195
## 3    AA-10480  Andrew Allen       2011-01-11      2014-09-05   17695.590
## 4    AA-10645 Anna Andreadi       2011-01-12      2014-12-05   15343.891
## 5      AA-315    Alex Avila       2011-08-06      2014-12-29    2243.256
## 6      AA-375  Allen Armold       2011-01-06      2014-07-03     654.492
##   avg_sales avg_quantity avg_shipping_cost avg_discount  monetary total_cost
## 1 327.31936     3.452381         29.432143   0.10357143  447.6905  13299.723
## 2 140.09988     3.309524         21.521905   0.16666667  677.4774   5206.718
## 3 465.67342     3.947368         42.991316   0.07847368 1516.4752  16179.115
## 4 210.19028     3.657534         24.003699   0.12594521 3051.4390  12292.452
## 5 280.40700     2.500000         26.975000   0.22500000  535.5660   1707.690
## 6  50.34554     2.307692          9.130769   0.14615385   77.4420    577.050
##   avg_cost_per_unit avg_profit_per_unit avg_ship_delay followtime
## 1          97.89035            4.616410      60.595238       1363
## 2          48.32729            6.410031      -8.928571       1344
## 3         112.33239           14.002054     -11.210526       1333
## 4          52.07982            9.271758      28.589041       1423
## 5         114.46200            4.797000       5.250000       1241
## 6          34.10746            5.552077      60.384615       1274
##        ship_mode segment_mode       city_mode      state_mode   country_mode
## 1 Standard Class     Consumer        Garforth         England United Kingdom
## 2 Standard Class     Consumer           Kabul           Kabul  United States
## 3 Standard Class     Consumer     Springfield        Missouri  United States
## 4 Standard Class     Consumer         Jakarta         Jakarta  United States
## 5 Standard Class     Consumer Dnipropetrovs'k Dnipropetrovs'k         Turkey
## 6 Standard Class     Consumer            Lodz            Lodz         Poland
##   market_mode    region_mode   category_mode subcategory_mode
## 1          EU          North Office Supplies          Binders
## 2          US          South Office Supplies          Storage
## 3          US        Central Office Supplies          Storage
## 4        APAC Southeast Asia Office Supplies          Storage
## 5        EMEA           EMEA Office Supplies          Storage
## 6        EMEA           EMEA Office Supplies        Envelopes
##                         product_mode order_priority_mode dayofweek_mode
## 1          Fiskars Trimmer, Serrated                High       Thursday
## 2 Ikea Classic Bookcase, Traditional                High         Friday
## 3         Rogers Folders, Industrial              Medium       Thursday
## 4                 BIC Markers, Blue              Medium      Wednesday
## 5             Smead Box, Industrial              Medium       Thursday
## 6  Enermax Numeric Keypad, Erganomic              Medium       Thursday
##   year_mode month_mode quarter_mode frequency recency freq_index rec_index
## 1      2014         04            2        42    3601          3         4
## 2      2013         12            4        42    3599          3         4
## 3      2012         08            2        38    3710          3         1
## 4      2013         05            2        73    3619          4         3
```

```
## 5      2013         08            3         8    3595          1         4
## 6      2014         01            2        13    3774          2         1
##   mon_index CLV     clv_bin churn high_value  log_CLV     pred
## 1         2  36 Medium-Low     0          1 3.610918 54.04678
## 2         3  44      Medium     0          0 3.806662 55.68252
## 3         3  68 Medium-High     1          1 4.234107 59.98922
## 4         4  64 Medium-High     0          1 4.174387 55.62098
## 5         2  28 Medium-Low     0          0 3.367296 45.26817
## 6         1  48      Medium     1          0 3.891820 40.52399
```

```
summary(final_model)
```

```
##
## Call:
## lm(formula = CLV ~ market_mode + avg_profit_per_unit + avg_shipping_cost +
##     followtime + year_mode + avg_discount + category_mode, data = final_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -28.5441  -5.8611   0.5984   6.7468  26.9774
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  44.647209   2.522867  17.697  < 2e-16 ***
## market_modeAPAC              12.137101   0.914096  13.278  < 2e-16 ***
## market_modeCanada            -7.066011   3.894686  -1.814  0.06983 .
## market_modeEMEA               1.780678   0.676095   2.634  0.00853 **
## market_modeEU                12.870233   0.965923  13.324  < 2e-16 ***
## market_modeLATAM             10.939749   0.921896  11.867  < 2e-16 ***
## market_modeUS                12.869142   0.927294  13.878  < 2e-16 ***
## avg_profit_per_unit           0.177931   0.021944   8.109 1.02e-15 ***
## avg_shipping_cost             0.139627   0.021351   6.540 8.31e-11 ***
## followtime                   -0.006450   0.001042  -6.192 7.54e-10 ***
## year_mode2012                -1.550654   0.924364  -1.678  0.09364 .
## year_mode2013                -0.735325   0.844908  -0.870  0.38427
## year_mode2014                -3.537536   0.797075  -4.438 9.71e-06 ***
## avg_discount                 -7.965733   2.949367  -2.701  0.00699 **
## category_modeOffice Supplies  4.753001   2.051680   2.317  0.02065 *
## category_modeTechnology       5.651749   2.507982   2.254  0.02436 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.425 on 1573 degrees of freedom
## Multiple R-squared:  0.3647, Adjusted R-squared:  0.3586
## F-statistic: 60.19 on 15 and 1573 DF,  p-value: < 2.2e-16
```

```
# Step 6: Evaluate the model
plot(final_results$CLV, final_results$pred,
     main = "Actual vs. Predicted CLV",
     xlab = "Actual CLV", ylab = "Predicted CLV")
abline(0, 1, col = "red")  # Reference line for final predictions
```



Results Interpretation: - The model explains approximately 36.5% of the variance in CLV (Adjusted R^2 = 0.3591). - Key predictors include market_mode, avg_profit_per_unit, and avg_shipping_cost. - Categories like "Office Supplies" and "Technology" have a positive relationship with CLV, while markets like "Africa" and "Canada" negatively affect CLV. - The visual shows a reasonable alignment between actual and predicted CLV, with some residual variance.

Basket Analysis

```
#create transactional data
data_mini <- data_df[,c("Order.ID","Sub.Category")]
head(data_mini)
```

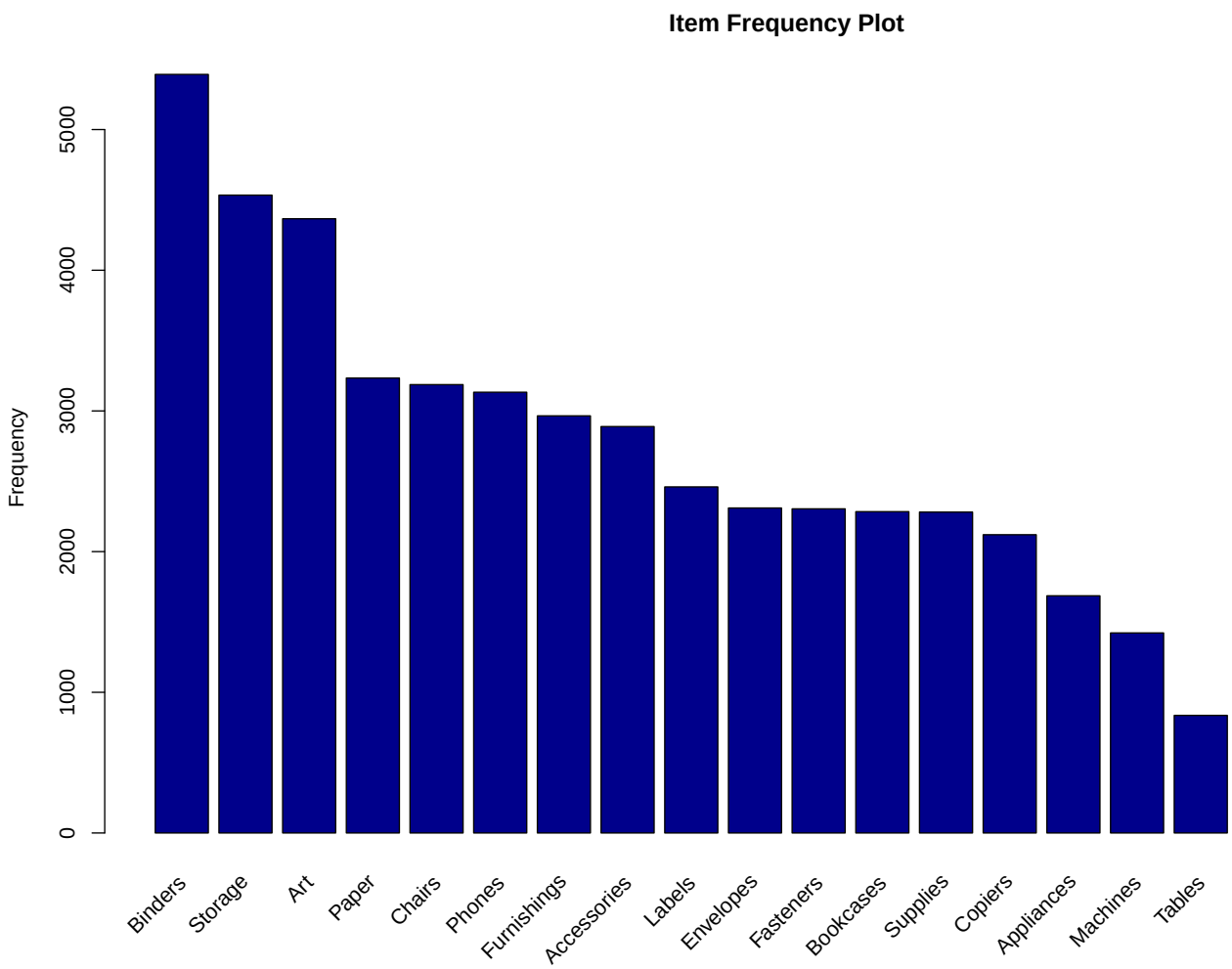
```
##          Order.ID Sub.Category
## 1    AG-2011-2040      Storage
## 2   IN-2011-47883     Supplies
## 3    HU-2011-1220      Storage
## 4 IT-2011-3647632        Paper
## 5   IN-2011-47883  Furnishings
## 6   IN-2011-47883        Paper
```

```
#save data_mini as csv
write.csv(data_mini,"transdata",row.names = F) #force the dataframe into csv

#read data and create transactions
transdata <- read.transactions(file="transdata",format="single",sep=",",
                               cols=c("Order.ID","Sub.Category"),
                               rm.duplicates = T, header = T)
data_transactions<-as(transdata,"transactions")
summary(data_transactions)
```

```
## transactions as itemMatrix in sparse format with
##  25035 rows (elements/itemsets/transactions) and
##  17 columns (items) and a density of 0.1113805
##
## most frequent items:
## Binders Storage     Art   Paper  Chairs (Other)
##    5392    4534    4366    3234    3187   26690
##
## element (itemset/transaction) length distribution:
## sizes
##     1     2     3     4     5     6     7     8     9    10    11
## 12800  6469  3193  1484   626   304   101    42     9     5     2
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   1.000   1.893   2.000  11.000
##
## includes extended item information - examples:
##        labels
## 1 Accessories
## 2  Appliances
## 3         Art
##
## includes extended transaction information - examples:
##   transactionID
## 1  AE-2011-9160
## 2  AE-2013-1130
## 3  AE-2013-1530
```

```
#item frequency plot
itemFrequencyPlot(data_transactions,topN=20,type="absolute",
                  main="Item Frequency Plot", ylab="Frequency",
                  col="darkblue", background="lightblue")
```

```
## Warning in plot.window(xlim, ylim, log = log, ...): "background" is not a
## graphical parameter

## Warning in axis(if (horiz) 2 else 1, at = at.l, labels = names.arg, lty =
## axis.lty, : "background" is not a graphical parameter

## Warning in title(main = main, sub = sub, xlab = xlab, ylab = ylab, ...):
## "background" is not a graphical parameter

## Warning in axis(if (horiz) 1 else 2, cex.axis = cex.axis, ...): "background" is
## not a graphical parameter
```




```
transaction_rules <- apriori(data_transactions, parameter=list(support=0.0005,
                                                               conf=0.4,
                                                               target="rules"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5   5e-04      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 12
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[17 item(s), 25035 transaction(s)] done [0.00s].
## sorting and recoding items ... [17 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 done [0.00s].
```

```
## writing ... [162 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
summary(transaction_rules)
```

```
## set of 162 rules
##
## rule length distribution (lhs + rhs):sizes
##   4   5
## 151  11
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   4.000   4.000   4.068   4.000   5.000
##
## summary of quality measures:
##     support            confidence        coverage              lift
##  Min.   :0.0005193   Min.   :0.4000   Min.   :0.0007989   Min.   :1.857
##  1st Qu.:0.0005592   1st Qu.:0.4093   1st Qu.:0.0012782   1st Qu.:1.974
##  Median :0.0007190   Median :0.4286   Median :0.0016777   Median :2.217
##  Mean   :0.0007927   Mean   :0.4469   Mean   :0.0018054   Mean   :2.273
##  3rd Qu.:0.0009187   3rd Qu.:0.4607   3rd Qu.:0.0021470   3rd Qu.:2.431
##  Max.   :0.0019972   Max.   :0.7000   Max.   :0.0048732   Max.   :3.885
##      count
##  Min.   :13.00
##  1st Qu.:14.00
##  Median :18.00
##  Mean   :19.85
##  3rd Qu.:23.00
##  Max.   :50.00
##
## mining info:
##               data ntransactions support confidence
##  data_transactions         25035   5e-04        0.4
##                                                                                                    call
##  apriori(data = data_transactions, parameter = list(support = 5e-04, conf = 0.4, target = "rules"))
```

```
df_basket <- as(transaction_rules,"data.frame")
```

```
customer.df
```

```
## # A tibble: 1,590 x 53
##    Customer.ID customer_name   first_order_date last_order_date total_sales
##    <fct>       <chr>           <date>           <date>                <dbl>
##  1 AA-10315    Alex Avila      2011-03-31       2014-12-23           13747.
##  2 AA-10375    Allen Armold    2011-04-21       2014-12-25            5884.
##  3 AA-10480    Andrew Allen    2011-01-11       2014-09-05           17696.
##  4 AA-10645    Anna Andreadi   2011-01-12       2014-12-05           15344.
##  5 AA-315      Alex Avila      2011-08-06       2014-12-29            2243.
##  6 AA-375      Allen Armold    2011-01-06       2014-07-03             654.
##  7 AA-480      Andrew Allen    2011-06-21       2014-02-20            2063.
##  8 AA-645      Anna Andreadi   2011-04-22       2014-10-11            1968.
##  9 AB-10015    Aaron Bergman   2011-02-19       2014-12-15           20037.
## 10 AB-10060    Adam Bellavance 2011-01-06       2014-12-06           18417.
## # i 1,580 more rows
## # i 48 more variables: avg_sales <dbl>, avg_quantity <dbl>,
## #   avg_shipping_cost <dbl>, avg_discount <dbl>, monetary <dbl>,
```

```
## #   total_cost <dbl>, avg_cost_per_unit <dbl>, avg_profit_per_unit <dbl>,
## #   avg_ship_delay <dbl>, followtime <dbl>, ship_mode <fct>,
## #   segment_mode <fct>, city_mode <fct>, state_mode <fct>, country_mode <fct>,
## #   market_mode <fct>, region_mode <fct>, category_mode <fct>, ...
```

```
customer.df <- customer.df %>% filter(!is.na(CLV))
```

```
# Segment customers by CLV into High, Medium, and Low categories
customer_summary <- customer.df %>%
  mutate(
    clv_segment = case_when(
      CLV >= quantile(CLV, 0.75, na.rm = TRUE) ~ "High CLV",       # Top 25% CLV
      CLV < quantile(CLV, 0.75, na.rm = TRUE) & CLV >= quantile(CLV, 0.25,
                                                                na.rm = TRUE) ~ "Medium CLV", # Middle
      CLV < quantile(CLV, 0.25, na.rm = TRUE) ~ "Low CLV",         # Bottom 25% CLV
      TRUE ~ "Unknown"                                             # Handle missing or unknown values
    )
  )

# Count the number of customers in each CLV segment
table_clv_segment <- table(customer_summary$clv_segment)

# Calculate the total revenue across all segments
total_revenue <- sum(customer.df$total_sales, na.rm = TRUE)  # Ensure `total_sales` is used from the da

# Create a summary for revenue proportions by CLV segment
revenue_by_segment <- customer_summary %>%
  group_by(clv_segment) %>%
  summarize(
    total_sales = sum(total_sales, na.rm = TRUE), # Total sales for each segment
    customer_count = n()  # Number of customers in each segment
  ) %>%
  mutate(
    revenue_proportion = total_sales / total_revenue * 100 # Proportion of total revenue
  )

# Calculate proportion of customers and revenue for High CLV segment
prop_high_clv_customers <- sum(customer_summary$clv_segment == "High CLV") / nrow(customer_summary)
prop_high_clv_revenue <- revenue_by_segment %>%
  filter(clv_segment == "High CLV") %>%
  pull(revenue_proportion) / 100

# Display results
cat(sprintf("Proportion of customer base that is high CLV: %.2f%%\n",
            prop_high_clv_customers * 100))
```

```
## Proportion of customer base that is high CLV: 36.23%
```

```
cat(sprintf("Proportion of total revenue brought by high CLV customers: %.2f%%\n",
            prop_high_clv_revenue * 100))
```

```
## Proportion of total revenue brought by high CLV customers: 58.43%
```

```
# Print counts and revenue proportions
print(table_clv_segment)
```

```
##
```

```
##   High CLV    Low CLV Medium CLV
##        576        357        657
```

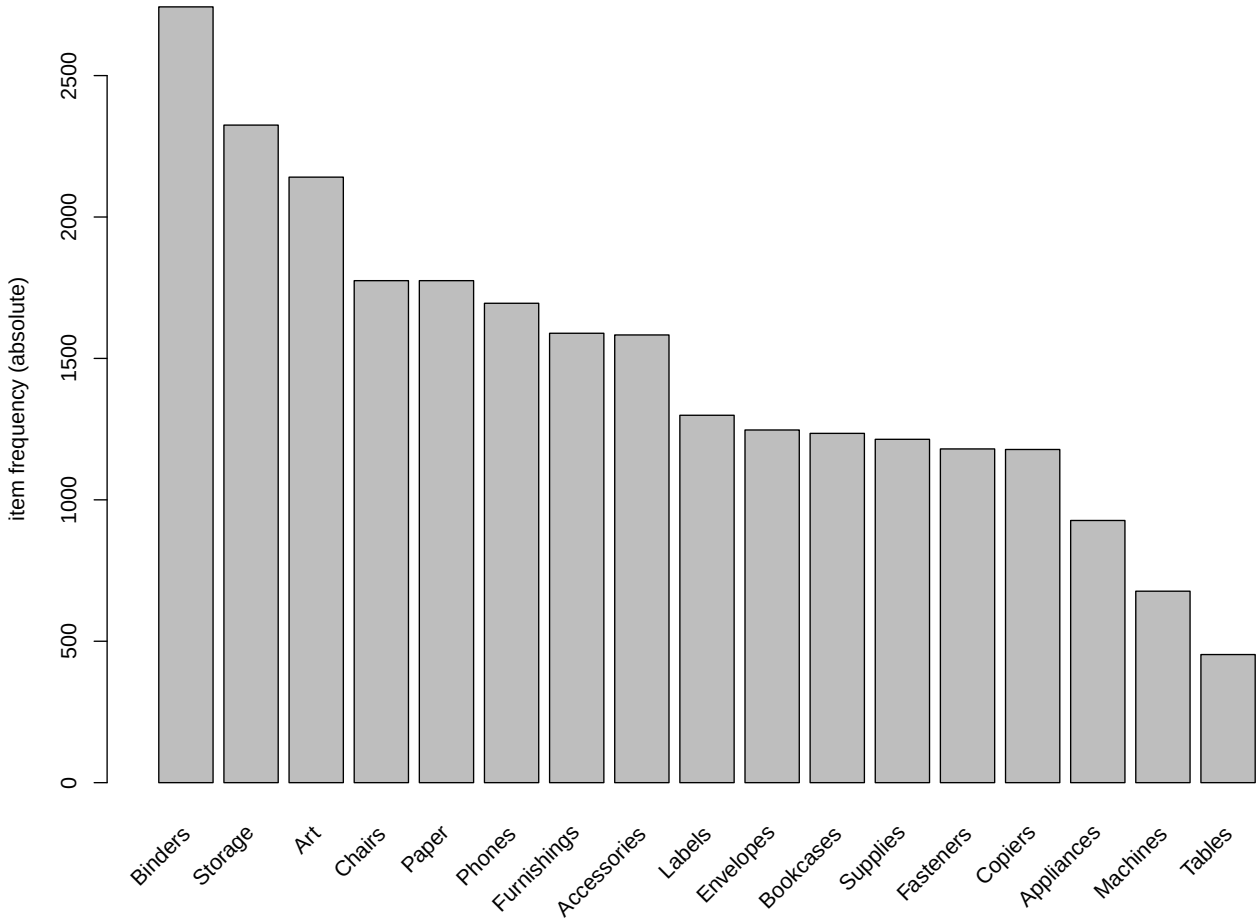
```
print(revenue_by_segment)
```

```
## # A tibble: 3 x 4
##   clv_segment total_sales customer_count revenue_proportion
##   <chr>             <dbl>          <int>              <dbl>
## 1 High CLV       7386830.            576               58.4
## 2 Low CLV        1419070.            357               11.2
## 3 Medium CLV     3836602.            657               30.3
```

```
# Validate segmentation counts (optional, already printed)
table(customer_summary$clv_segment)
```

```
##
##   High CLV    Low CLV Medium CLV
##        576        357        657
```

```
# Filter high-value customers
high_clv_customers <- customer_summary %>%
  filter(clv_segment == "High CLV") %>% pull(Customer.ID)

# Filter transactions involving high CLV customers
high_clv_transactions <- data_df %>% filter(Customer.ID %in% high_clv_customers)

# Create transactional data for high-value customers
high_value_data_mini <- high_clv_transactions[, c("Order.ID", "Sub.Category")]

# Save and read transaction data for high-value customers
write.csv(high_value_data_mini, "high_value_transdata.csv", row.names = FALSE)
high_value_transdata <- read.transactions(file = "high_value_transdata.csv",
                                          format = "single", sep = ",",
                                          cols = c("Order.ID", "Sub.Category"),
                                          rm.duplicates = TRUE, header = TRUE)
data_transactions_high_value <- as(high_value_transdata, "transactions")

itemFrequencyPlot(data_transactions_high_value,topN=20,type="absolute")
```

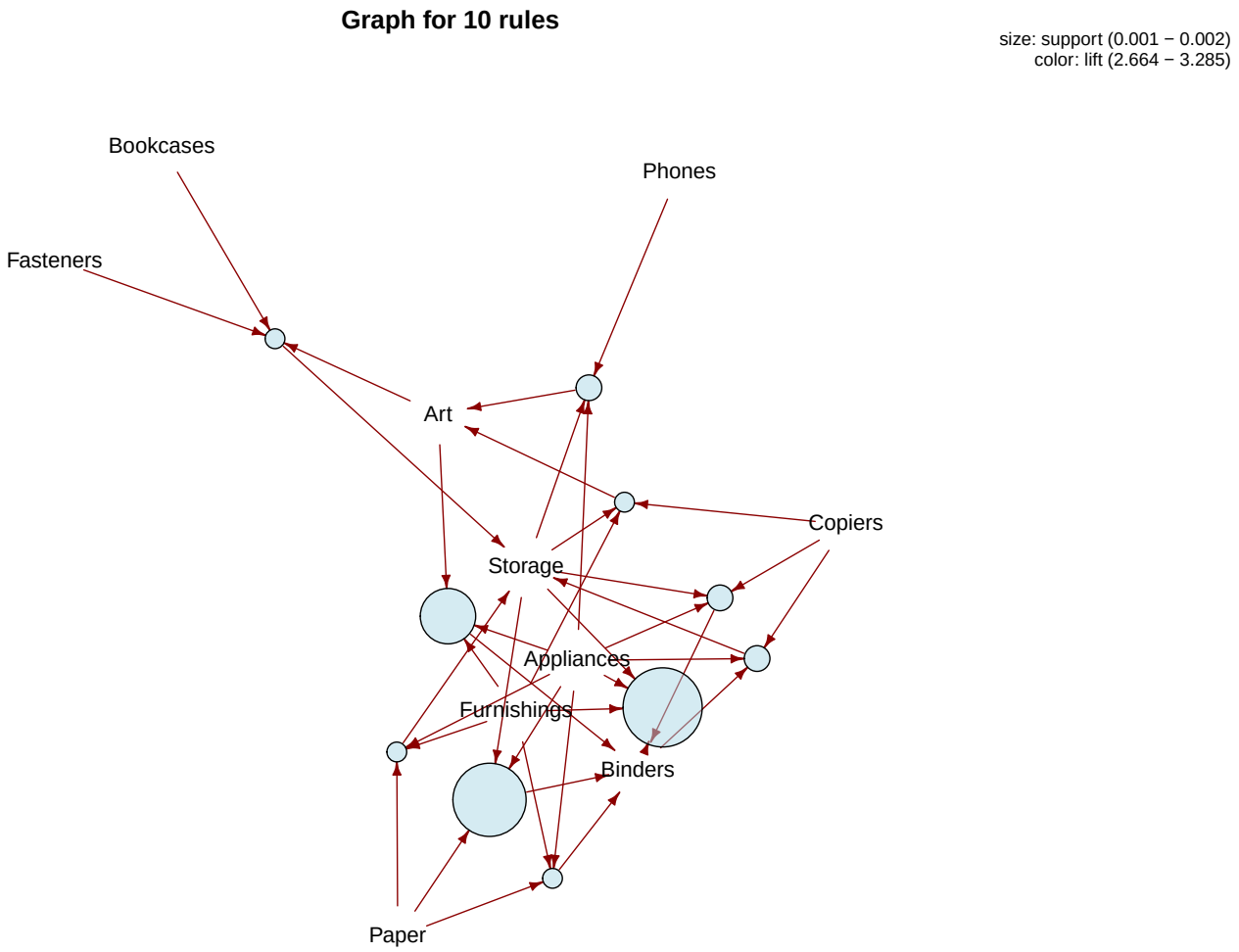



```
# Generate association rules for high-value customers
high_value_rules <- apriori(data_transactions_high_value,
                            parameter = list(support = 0.001,
                                             confidence = 0.5, target = "rules"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 12
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[17 item(s), 12930 transaction(s)] done [0.00s].
## sorting and recoding items ... [17 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 done [0.00s].
## writing ... [20 rule(s)] done [0.00s].
```

```
## creating S4 object  ... done [0.00s].
```

```
# Plot the high-value customer rules
plot(head(sort(high_value_rules, by = "lift"), 10), method = "graph",
     control = list(nodeCol = "lightblue", edgeCol = "darkred",
                    labelCol = "black"), engine = "igraph")
```




```
high_value_hi_lift <- head(sort(high_value_rules, by = "lift"), 10)
inspect(high_value_hi_lift)
```

Visualizations

```
##      lhs                                    rhs       support     confidence
## [1]  {Appliances, Furnishings, Storage} => {Binders} 0.001778809 0.6969697
## [2]  {Appliances, Furnishings, Paper}   => {Storage} 0.001005414 0.5652174
## [3]  {Appliances, Copiers, Storage}     => {Binders} 0.001082753 0.6666667
## [4]  {Copiers, Furnishings, Storage}    => {Art}     0.001005414 0.5200000
## [5]  {Appliances, Phones, Storage}      => {Art}     0.001082753 0.5185185
## [6]  {Appliances, Art, Furnishings}     => {Binders} 0.001469451 0.6551724
## [7]  {Appliances, Paper, Storage}       => {Binders} 0.001701469 0.6470588
## [8]  {Appliances, Binders, Copiers}     => {Storage} 0.001082753 0.5384615
## [9]  {Art, Bookcases, Fasteners}        => {Storage} 0.001005414 0.5000000
## [10] {Appliances, Furnishings, Paper}   => {Binders} 0.001005414 0.5652174
```

```
##      coverage    lift     count
## [1]  0.002552204 3.285388 23
## [2]  0.001778809 3.143338 13
## [3]  0.001624130 3.142545 14
## [4]  0.001933488 3.140402 13
## [5]  0.002088167 3.131455 14
## [6]  0.002242846 3.088363 19
## [7]  0.002629544 3.050117 22
## [8]  0.002010828 2.994541 14
## [9]  0.002010828 2.780645 13
## [10] 0.001778809 2.664331 13
```

```
# Convert the rules to a data frame
df_basket <- as(high_value_hi_lift, "data.frame")
df_basket
```

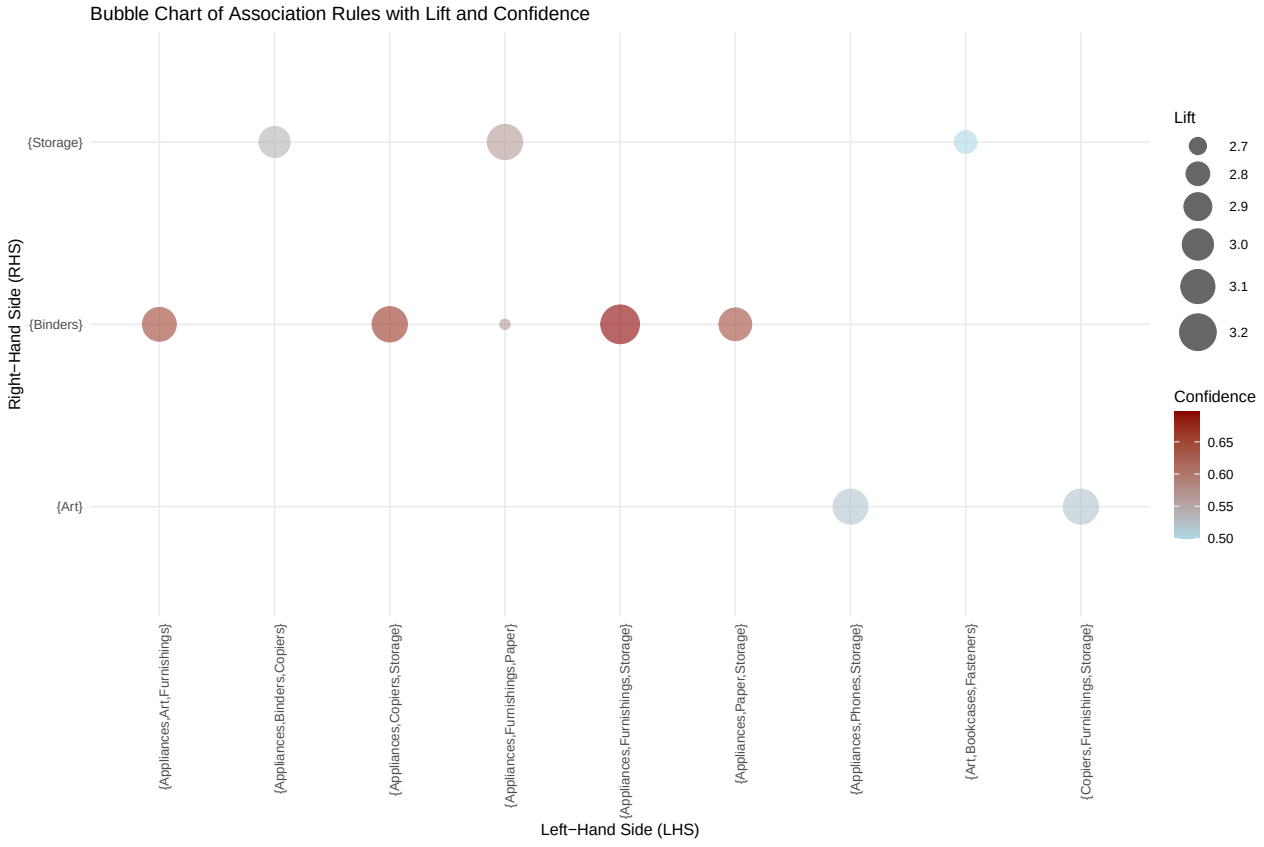
```
##                                          rules     support confidence
## 6  {Appliances,Furnishings,Storage} => {Binders} 0.001778809  0.6969697
## 3    {Appliances,Furnishings,Paper} => {Storage} 0.001005414  0.5652174
## 1      {Appliances,Copiers,Storage} => {Binders} 0.001082753  0.6666667
## 11        {Copiers,Furnishings,Storage} => {Art} 0.001005414  0.5200000
## 7           {Appliances,Phones,Storage} => {Art} 0.001082753  0.5185185
## 5      {Appliances,Art,Furnishings} => {Binders} 0.001469451  0.6551724
## 8        {Appliances,Paper,Storage} => {Binders} 0.001701469  0.6470588
## 2      {Appliances,Binders,Copiers} => {Storage} 0.001082753  0.5384615
## 13        {Art,Bookcases,Fasteners} => {Storage} 0.001005414  0.5000000
## 4    {Appliances,Furnishings,Paper} => {Binders} 0.001005414  0.5652174
##       coverage     lift count
## 6  0.002552204 3.285388    23
## 3  0.001778809 3.143338    13
## 1  0.001624130 3.142545    14
## 11 0.001933488 3.140402    13
## 7  0.002088167 3.131455    14
## 5  0.002242846 3.088363    19
## 8  0.002629544 3.050117    22
## 2  0.002010828 2.994541    14
## 13 0.002010828 2.780645    13
## 4  0.001778809 2.664331    13
```

```
# Assuming df_basket is your data frame
# Extract LHS using the pattern to get everything before "=>"
df_basket$lhs <- str_extract(df_basket$rules, "\\{.*?\\}")

# Extract RHS using the pattern to get everything after "=>"
df_basket$rhs <- str_extract(df_basket$rules, "(?<=\\=>)\\s*\\{.*?\\}")

# Remove spaces and clean up the strings
df_basket$lhs <- str_trim(df_basket$lhs)
df_basket$rhs <- str_trim(df_basket$rhs)



# Extract lift values along with lhs and rhs for visualization
heatmap_data <- df_basket[, c("lhs", "rhs", "lift")]
```

```r
# Reshape data for heatmap
heatmap_data_melted <- melt(heatmap_data, id.vars = c("lhs", "rhs"))

# View reshaped data
head(heatmap_data_melted)
```

```
##                                 lhs       rhs variable    value
## 1 {Appliances,Furnishings,Storage} {Binders}     lift 3.285388
## 2   {Appliances,Furnishings,Paper} {Storage}     lift 3.143338
## 3     {Appliances,Copiers,Storage} {Binders}     lift 3.142545
## 4    {Copiers,Furnishings,Storage}     {Art}     lift 3.140402
## 5      {Appliances,Phones,Storage}     {Art}     lift 3.131455
## 6     {Appliances,Art,Furnishings} {Binders}     lift 3.088363
```

```r
# Assuming df_basket now includes "lhs", "rhs", "lift", and "confidence"
# Reshape data for heatmap
heatmap_data <- df_basket[, c("lhs", "rhs", "lift", "confidence")]

ggplot(heatmap_data, aes(x = lhs, y = rhs, color = confidence, size = lift)) +
  geom_point(alpha = 0.6) +
  scale_color_gradient(low = "lightblue", high = "darkred", name = "Confidence") +
  scale_size_continuous(name = "Lift", range = c(3, 12)) +
  labs(title = "Bubble Chart of Association Rules with Lift and Confidence",
       x = "Left-Hand Side (LHS)",
       y = "Right-Hand Side (RHS)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Logistic Regression

Specifically, we are examining the probability of high revenue contribution and focusing on identifying customers who are likely to generate significant revenue for the business. This involves defining a threshold for "high value" (e.g., customers whose revenue contributions are in the top 25% or above a certain dollar amount) and using logistic regression to model which factors contribute to this status.

```
#rounding
options(digits = 3)
options(scipen = 999)

#set seed and split data
set.seed(2025)
sample <- sample(c(TRUE, FALSE), nrow(final_df), replace = TRUE,
                 prob = c(0.7, 0.3))
train <- final_df[sample,]
val <- final_df[!sample,]

#number of rows in train and validation data
nrow(train)
```

```
## [1] 1104
```

```
nrow(val)
```

```
## [1] 485
```

Before splitting the data, we first used set.seed() so that we can reproduce the same random values that go into the training and test data. This ensures that we acquire the same model output. We then split the data into 70% and 30% partitions for training and validation sets, respectively. We built our first logistic regression model using the high_value variable as the predictor variable and the rest as the independent.

```
#first model
logistic1 <- glm(high_value ~ frequency + avg_quantity + avg_shipping_cost +
                  avg_discount + avg_profit_per_unit + avg_ship_delay + followtime +
                  ship_mode + segment_mode + market_mode + region_mode + category_mode +
                  subcategory_mode + order_priority_mode + dayofweek_mode +
                  year_mode + month_mode + quarter_mode,
                  data = train,
                  family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(logistic1)
```

```
##
## Call:
## glm(formula = high_value ~ frequency + avg_quantity + avg_shipping_cost +
##     avg_discount + avg_profit_per_unit + avg_ship_delay + followtime +
##     ship_mode + segment_mode + market_mode + region_mode + category_mode +
##     subcategory_mode + order_priority_mode + dayofweek_mode +
##     year_mode + month_mode + quarter_mode, family = binomial,
##     data = train)
##
## Coefficients: (3 not defined because of singularities)
##                                 Estimate  Std. Error z value
## (Intercept)                  -101.57374  6525.31596   -0.02
## frequency                       0.38117     0.04303    8.86
```

```
## avg_quantity                    2.64864     0.69571    3.81
## avg_shipping_cost               0.37782     0.04463    8.46
## avg_discount                    3.30016     5.59124    0.59
## avg_profit_per_unit             0.07643     0.04052    1.89
## avg_ship_delay                 -0.00654     0.00958   -0.68
## followtime                      0.00848     0.00252    3.36
## ship_modeSame Day              13.37263  6931.32807    0.00
## ship_modeSecond Class           5.00901     4.37693    1.14
## ship_modeStandard Class         5.38982     4.29212    1.26
## segment_modeCorporate           0.57629     0.43734    1.32
## segment_modeHome Office        -0.59723     0.55308   -1.08
## market_modeAPAC                34.71236  1252.55412    0.03
## market_modeCanada              38.27496 15165.22260    0.00
## market_modeEMEA                20.39321  2084.03622    0.01
## market_modeEU                  34.30770  1252.55399    0.03
## market_modeLATAM               34.51216  1252.55400    0.03
## market_modeUS                  34.83313  1252.55396    0.03
## region_modeCanada                    NA          NA      NA
## region_modeCaribbean           24.67057 32519.58715    0.00
## region_modeCentral              0.21653     1.33564    0.16
## region_modeCentral Asia         2.08211     6.59733    0.32
## region_modeEast                -1.33597     1.99380   -0.67
## region_modeEMEA                      NA          NA      NA
## region_modeNorth                1.21510     1.46764    0.83
## region_modeNorth Asia           1.92791     2.11343    0.91
## region_modeOceania              3.37578     2.31534    1.46
## region_modeSouth                0.41067     1.35221    0.30
## region_modeSoutheast Asia       1.44038     1.71530    0.84
## region_modeWest                      NA          NA      NA
## category_modeOffice Supplies   -6.29734    16.46085   -0.38
## category_modeTechnology         8.52772  3013.84999    0.00
## subcategory_modeAppliances     11.57817  7669.68260    0.00
## subcategory_modeArt             0.58083     1.13347    0.51
## subcategory_modeBinders         0.37623     1.00606    0.37
## subcategory_modeBookcases      -3.45728     4.49023   -0.77
## subcategory_modeChairs         -0.29329     1.12575   -0.26
## subcategory_modeCopiers         1.66075     1.81188    0.92
## subcategory_modeEnvelopes      -2.44352     1.83603   -1.33
## subcategory_modeFasteners       1.83743     3.28381    0.56
## subcategory_modeFurnishings    -2.72151     1.47718   -1.84
## subcategory_modeLabels        -16.00336  9183.92941    0.00
## subcategory_modeMachines        2.51924  6688.30236    0.00
## subcategory_modePaper          -0.38132     1.21156   -0.31
## subcategory_modePhones          2.11990     1.25050    1.70
## subcategory_modeStorage        -0.07232     1.05086   -0.07
## subcategory_modeSupplies        3.62981     2.01041    1.81
## subcategory_modeTables         20.99924 21853.51606    0.00
## order_priority_modeHigh         8.98586  6403.99273    0.00
## order_priority_modeLow         26.08876  7071.50325    0.00
## order_priority_modeMedium      11.19169  6403.99277    0.00
## dayofweek_modeMonday            0.87999     0.66831    1.32
## dayofweek_modeSaturday          0.55633     0.88003    0.63
## dayofweek_modeSunday            2.85879     2.07533    1.38
## dayofweek_modeThursday         -0.86087     0.61237   -1.41
```

```
## dayofweek_modeTuesday            0.96872     0.62003    1.56
## dayofweek_modeWednesday          1.44079     0.65339    2.21
## year_mode2012                    2.08868     0.90624    2.30
## year_mode2013                    1.54761     0.79154    1.96
## year_mode2014                    1.26635     0.71303    1.78
## month_mode02                     2.33035     2.24338    1.04
## month_mode03                     2.43851     1.70439    1.43
## month_mode04                     4.63322     1.67497    2.77
## month_mode05                     5.10860     1.76298    2.90
## month_mode06                     6.15785     1.74335    3.53
## month_mode07                     5.31221     1.77153    3.00
## month_mode08                     3.26732     1.61683    2.02
## month_mode09                     3.62347     1.55822    2.33
## month_mode10                     5.87715     1.75879    3.34
## month_mode11                     5.75314     1.66910    3.45
## month_mode12                     4.10495     1.60046    2.56
## quarter_mode2                   -1.92866     1.24803   -1.55
## quarter_mode3                   -2.97243     1.21463   -2.45
## quarter_mode4                   -3.35176     1.25433   -2.67
##                                         Pr(>|z|)
## (Intercept)                              0.98758
## frequency                   < 0.0000000000000002 ***
## avg_quantity                             0.00014 ***
## avg_shipping_cost           < 0.0000000000000002 ***
## avg_discount                             0.55503
## avg_profit_per_unit                      0.05927 .
## avg_ship_delay                           0.49481
## followtime                               0.00078 ***
## ship_modeSame Day                        0.99846
## ship_modeSecond Class                    0.25245
## ship_modeStandard Class                  0.20921
## segment_modeCorporate                    0.18760
## segment_modeHome Office                  0.28022
## market_modeAPAC                          0.97789
## market_modeCanada                        0.99799
## market_modeEMEA                          0.99219
## market_modeEU                            0.97815
## market_modeLATAM                         0.97802
## market_modeUS                            0.97781
## region_modeCanada                            NA
## region_modeCaribbean                     0.99939
## region_modeCentral                       0.87121
## region_modeCentral Asia                  0.75231
## region_modeEast                          0.50282
## region_modeEMEA                              NA
## region_modeNorth                         0.40771
## region_modeNorth Asia                    0.36165
## region_modeOceania                       0.14484
## region_modeSouth                         0.76136
## region_modeSoutheast Asia                0.40106
## region_modeWest                              NA
## category_modeOffice Supplies             0.70204
## category_modeTechnology                  0.99774
## subcategory_modeAppliances               0.99880
```

```
## subcategory_modeArt                       0.60835
## subcategory_modeBinders                   0.70843
## subcategory_modeBookcases                 0.44133
## subcategory_modeChairs                    0.79446
## subcategory_modeCopiers                   0.35936
## subcategory_modeEnvelopes                 0.18323
## subcategory_modeFasteners                 0.57579
## subcategory_modeFurnishings               0.06542 .
## subcategory_modeLabels                    0.99861
## subcategory_modeMachines                  0.99970
## subcategory_modePaper                     0.75296
## subcategory_modePhones                    0.09003 .
## subcategory_modeStorage                   0.94513
## subcategory_modeSupplies                  0.07100 .
## subcategory_modeTables                    0.99923
## order_priority_modeHigh                   0.99888
## order_priority_modeLow                    0.99706
## order_priority_modeMedium                 0.99861
## dayofweek_modeMonday                      0.18793
## dayofweek_modeSaturday                    0.52727
## dayofweek_modeSunday                      0.16835
## dayofweek_modeThursday                    0.15978
## dayofweek_modeTuesday                     0.11820
## dayofweek_modeWednesday                   0.02745 *
## year_mode2012                             0.02118 *
## year_mode2013                             0.05056 .
## year_mode2014                             0.07573 .
## month_mode02                              0.29891
## month_mode03                              0.15251
## month_mode04                              0.00567 **
## month_mode05                              0.00376 **
## month_mode06                              0.00041 ***
## month_mode07                              0.00271 **
## month_mode08                              0.04330 *
## month_mode09                              0.02005 *
## month_mode10                              0.00083 ***
## month_mode11                              0.00057 ***
## month_mode12                              0.01032 *
## quarter_mode2                             0.12226
## quarter_mode3                             0.01440 *
## quarter_mode4                             0.00754 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1232.8  on 1103  degrees of freedom
## Residual deviance:  220.8  on 1032  degrees of freedom
## AIC: 364.8
##
## Number of Fisher Scoring iterations: 21
```

```r
#stepwise filter
logistic2 <- step(logistic1)
```

```
## Start:  AIC=365
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_discount +
##     avg_profit_per_unit + avg_ship_delay + followtime + ship_mode +
##     segment_mode + market_mode + region_mode + category_mode +
##     subcategory_mode + order_priority_mode + dayofweek_mode +
##     year_mode + month_mode + quarter_mode

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - subcategory_mode    16      245 357
## - region_mode          9      233 359
## - market_mode          3      222 360
## - category_mode        2      222 362
## - ship_mode            3      224 362
## - avg_discount         1      221 363
## - avg_ship_delay       1      221 363
## <none>                        221 365
## - year_mode            3      227 365
## - segment_mode         2      225 365
## - avg_profit_per_unit  1      224 366
## - quarter_mode         3      231 369
## - order_priority_mode  3      231 369
## - dayofweek_mode       6      238 370
## - followtime           1      234 376
## - month_mode          11      256 378
## - avg_quantity         1      237 379
## - avg_shipping_cost    1      400 542
## - frequency            1      519 661

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=357
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_discount +
##     avg_profit_per_unit + avg_ship_delay + followtime + ship_mode +
##     segment_mode + market_mode + region_mode + category_mode +
##     order_priority_mode + dayofweek_mode + year_mode + month_mode +
##     quarter_mode
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - region_mode          9      255 349
## - market_mode          3      245 351
## - ship_mode            3      247 353
## - category_mode        2      245 353
## - year_mode            3      249 355
## - avg_discount         1      245 355
## - avg_ship_delay       1      245 355
## - segment_mode         2      249 357
## <none>                        245 357
## - avg_profit_per_unit  1      247 357
## - quarter_mode         3      253 359
## - order_priority_mode  3      253 359
## - dayofweek_mode       6      261 361
## - followtime           1      252 362
## - month_mode          11      274 364
## - avg_quantity         1      258 368
## - avg_shipping_cost    1      414 524
## - frequency            1      541 651

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=349
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_discount +
##     avg_profit_per_unit + avg_ship_delay + followtime + ship_mode +
##     segment_mode + market_mode + category_mode + order_priority_mode +
##     dayofweek_mode + year_mode + month_mode + quarter_mode

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - ship_mode            3      256 344
## - category_mode        2      256 346
## - market_mode          6      264 346
## - year_mode            3      259 347
## - avg_discount         1      255 347
## - avg_ship_delay       1      255 347
## - avg_profit_per_unit  1      256 348
## <none>                        255 349
## - segment_mode         2      260 350
## - order_priority_mode  3      263 351
## - quarter_mode         3      265 353
## - dayofweek_mode       6      271 353
## - followtime           1      263 355
## - avg_quantity         1      264 356
## - month_mode          11      284 356
## - avg_shipping_cost    1      425 517
## - frequency            1      552 644

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=344
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_discount +
##     avg_profit_per_unit + avg_ship_delay + followtime + segment_mode +
##     market_mode + category_mode + order_priority_mode + dayofweek_mode +
##     year_mode + month_mode + quarter_mode

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - category_mode        2      257 341
## - year_mode            3      260 342
## - avg_ship_delay       1      256 342
```

```
## - avg_discount         1      256 342
## - avg_profit_per_unit  1      258 344
## <none>                        256 344
## - segment_mode         2      262 346
## - order_priority_mode  3      265 347
## - quarter_mode         3      266 348
## - dayofweek_mode       6      273 349
## - followtime           1      264 350
## - market_mode          6      274 350
## - avg_quantity         1      265 351
## - month_mode          11      285 351
## - avg_shipping_cost    1      426 512
## - frequency            1      555 641

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=341
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_discount +
##     avg_profit_per_unit + avg_ship_delay + followtime + segment_mode +
##     market_mode + order_priority_mode + dayofweek_mode + year_mode +
##     month_mode + quarter_mode

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - year_mode            3      261 339
## - avg_ship_delay       1      257 339
## - avg_discount         1      258 340
## - avg_profit_per_unit  1      259 341
## <none>                        257 341
## - segment_mode         2      263 343
## - order_priority_mode  3      265 343
## - quarter_mode         3      267 345
## - dayofweek_mode       6      274 346
## - followtime           1      265 347
## - market_mode          6      276 348
## - avg_quantity         1      266 348
## - month_mode          11      287 349
## - avg_shipping_cost    1      431 513
## - frequency            1      555 637

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=339
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_discount +
##     avg_profit_per_unit + avg_ship_delay + followtime + segment_mode +
##     market_mode + order_priority_mode + dayofweek_mode + month_mode +
##     quarter_mode
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                       Df Deviance AIC
## - avg_discount         1      261 337
## - avg_ship_delay       1      261 337
## - avg_profit_per_unit  1      262 338
## <none>                        261 339
## - segment_mode         2      265 339
## - order_priority_mode  3      268 340
## - dayofweek_mode       6      276 342
## - quarter_mode         3      271 343
## - followtime           1      267 343
## - month_mode          11      289 345
## - market_mode          6      280 346
## - avg_quantity         1      270 346
## - avg_shipping_cost    1      432 508
## - frequency            1      559 635
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##
## Step:  AIC=337
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_profit_per_unit +
##     avg_ship_delay + followtime + segment_mode + market_mode +
##     order_priority_mode + dayofweek_mode + month_mode + quarter_mode
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - avg_ship_delay       1      262 336
## - avg_profit_per_unit  1      263 337
## <none>                        261 337
## - segment_mode         2      266 338
## - order_priority_mode  3      268 338
## - dayofweek_mode       6      276 340
## - quarter_mode         3      271 341
## - followtime           1      267 341
## - month_mode          11      289 343
## - market_mode          6      280 344
## - avg_quantity         1      270 344
## - avg_shipping_cost    1      433 507
## - frequency            1      559 633

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=336
## high_value ~ frequency + avg_quantity + avg_shipping_cost + avg_profit_per_unit +
##     followtime + segment_mode + market_mode + order_priority_mode +
##     dayofweek_mode + month_mode + quarter_mode

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##                       Df Deviance AIC
## - avg_profit_per_unit  1      263 335
## <none>                        262 336
## - segment_mode         2      266 336
## - order_priority_mode  3      269 337
## - dayofweek_mode       6      276 338
## - quarter_mode         3      271 339
## - followtime           1      267 339
## - market_mode          6      280 342
## - month_mode          11      290 342
## - avg_quantity         1      270 342
## - avg_shipping_cost    1      433 505
## - frequency            1      559 631

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

##
## Step:  AIC=335
## high_value ~ frequency + avg_quantity + avg_shipping_cost + followtime +
##     segment_mode + market_mode + order_priority_mode + dayofweek_mode +
```

```
##     month_mode + quarter_mode
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
##                        Df Deviance  AIC
## <none>                        263  335
## - segment_mode          2      267  335
## - order_priority_mode   3      270  336
## - dayofweek_mode        6      277  337
## - followtime            1      269  339
## - market_mode           6      280  340
## - avg_quantity          1      271  341
## - month_mode           11      291  341
## - avg_shipping_cost     1      450  520
## - frequency             1      559  629
## - quarter_mode          3     7713 7779
```

```
#best model
logistic3 <- glm(high_value ~ frequency + avg_quantity + avg_shipping_cost +
                  followtime + segment_mode + market_mode + quarter_mode,
                  data = train,
                  family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
#interpret results
summary(logistic3)
```

```
##
## Call:
## glm(formula = high_value ~ frequency + avg_quantity + avg_shipping_cost +
##     followtime + segment_mode + market_mode + quarter_mode, family = binomial,
##     data = train)
##
## Coefficients:
##                            Estimate Std. Error z value            Pr(>|z|)
## (Intercept)              -53.80961  764.70579   -0.07               0.944
## frequency                  0.26754    0.02560   10.45 <0.0000000000000002 ***
## avg_quantity               1.24642    0.48937    2.55               0.011 *
## avg_shipping_cost          0.25449    0.02641    9.64 <0.0000000000000002 ***
## followtime                 0.00303    0.00169    1.80               0.072 .
## segment_modeCorporate      0.55799    0.32703    1.71               0.088 .
## segment_modeHome Office   -0.24169    0.40903   -0.59               0.555
## market_modeAPAC           24.45985  764.69566    0.03               0.974
## market_modeCanada         19.10918 7650.96185    0.00               0.998
## market_modeEMEA            1.05840 2053.14532    0.00               1.000
```

```
## market_modeEU             24.45730  764.69563    0.03              0.974
## market_modeLATAM          24.34504  764.69564    0.03              0.975
## market_modeUS             24.44117  764.69566    0.03              0.975
## quarter_mode2             -0.00337    0.59478   -0.01              0.995
## quarter_mode3             -0.74496    0.60150   -1.24              0.216
## quarter_mode4             -0.81315    0.57262   -1.42              0.156
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1232.77  on 1103  degrees of freedom
## Residual deviance:  309.48  on 1088  degrees of freedom
## AIC: 341.5
##
## Number of Fisher Scoring iterations: 19
```

```
round(exp(coef(logistic3)), 4)
```

```
##             (Intercept)               frequency            avg_quantity
##                   0.000                   1.307                   3.478
##       avg_shipping_cost              followtime   segment_modeCorporate
##                   1.290                   1.003                   1.747
## segment_modeHome Office         market_modeAPAC       market_modeCanada
##                   0.785          41954453204.938           199071901.522
##         market_modeEMEA           market_modeEU        market_modeLATAM
##                   2.882          41847672022.389          37403715235.820
##           market_modeUS            quarter_mode2            quarter_mode3
##          41177848933.822                   0.997                   0.475
##           quarter_mode4
##                   0.444
```

After using the stepwise filter to remove the irrelevant variables, we ended up with eight independent variables but only a few of them were deemed significant due to their p-values being less than 5%.

Most significant variables are: frequency, avg_quantity, avg_shipping_cost, followtime, segment_modeCorporate.

- Frequency: If a customer's total orders increases by 1, the odds of a customer being a high-value customer is 31.2% MORE likely.
- Avg Quantity: If a customer's average order amount increases by 1, the odds of a customer being a high-value customer is 168.5% MORE likely.
- Avg Shipping Cost: If a customer's average shipping cost increases by $1, the odds of a customer being a high-value customer is 28.8% MORE likely.
- Followtime: If a customer's followtime increases by 1 day, the odds of a customer being a high-value customer is 0.4% MORE likely.
- Segment Mode Corporate: Compared to customers who are in the Consumer segment, customers who are in the Corporate segment are 135.1% MORE likely to be high-value customers.

```
#predict high_value, apply model to validation data
pred = predict(logistic3, newdata = val, type = "response")

#combine validation data with predictions
final_data = cbind(val, pred)
```

From there, we tested our logit regression model by predicting the high-value dependent variable, which

was done by applying the model to the validation data and then combining the validation data with the predictions.

```
#confusion matrix
confusion_matrix <- table(final_data$high_value, final_data$pred > 0.5)
percent_table <- prop.table(confusion_matrix, 2)
percent_table
```

```
##
##      FALSE   TRUE
##   0 0.9499 0.1508
##   1 0.0501 0.8492
```

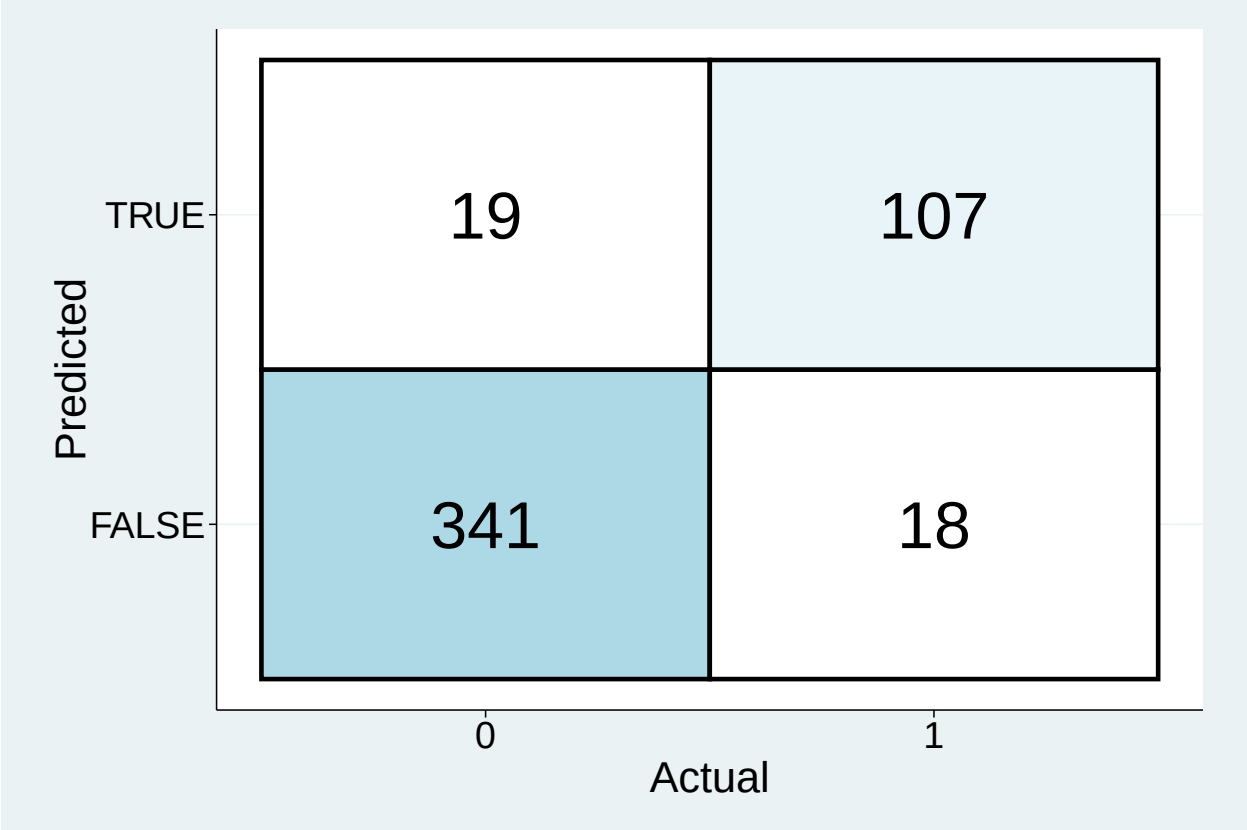
```
#accuracy
accuracy_rate = sum(diag(confusion_matrix))/sum(confusion_matrix)
accuracy_rate
```

```
## [1] 0.924
```

```
#proportions of churned and non-churned customers
prop.table(table(val$churn))
```

```
##
##     0     1
## 0.757 0.243
```

```
#plot confusion matrix with ggplot
confusion_matrix <- as.data.frame(confusion_matrix)

ggplot(confusion_matrix, aes(x = Var1, y = Var2, fill = Freq)) +
  geom_tile(color = "black", lwd = 1.5, linetype = 1) +
  geom_text(aes(label = Freq), size = 16) +
  scale_fill_gradient(low = "white", high = "lightblue") +
  theme_stata() +
  theme(axis.text.x = element_text(size = 26),
        axis.text.y = element_text(size = 26, angle = 0),
        axis.title = element_text(size = 30)) +
  theme(legend.position = "none") +
  labs(x = "Actual", y = "Predicted", fill = "Frequency")
```




```
#find best cutoff
pred.new <- prediction(pred, val$high_value)
eval <- performance(pred.new, "acc")

max <- which.max(slot(eval, "y.values")[[1]])
max
```

```
## [1] 114
```

```
acc <- slot(eval, "y.values")[[1]][max]

cut <- slot(eval, "x.values")[[1]][max]

print(c(accuracy.rate = acc, cutoff = cut))
```

```
## accuracy.rate    cutoff.165
##         0.930         0.616
```

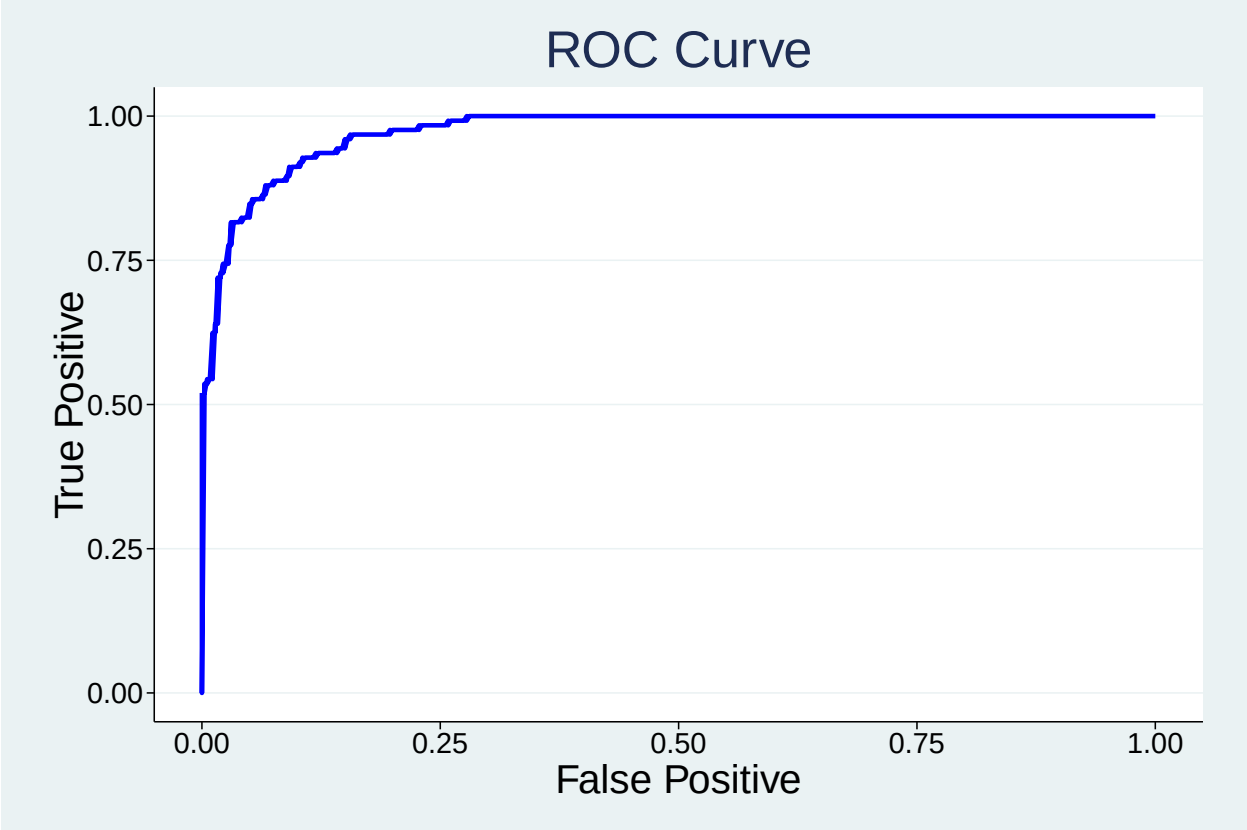
```
#plot ROC curve with ggplot
roc <- roc(val$high_value, pred)
```

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

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

```
roc_df <- data.frame(specificity = 1 - roc$specificities,
                     sensitivity = roc$sensitivities)

ggplot(roc_df, aes(x = specificity, y = sensitivity)) +
  geom_line(color = "blue", size = 1.5) +
```

```
theme_stata() +
theme(axis.text.x = element_text(size = 20),
      axis.text.y = element_text(size = 20, angle = 0),
      axis.title = element_text(size = 28),
      plot.title = element_text(size = 36)) +
labs(title = "ROC Curve", x = "False Positive", y = "True Positive")
```



We checked the accuracy of our predictive model and we ended up with an accuracy of 91%, meaning that the logit regression can effectively predict if a customer will be low or high valued.

Key Insights: - Customers who are in the corporate and consumer segments are more likely to be ones who will generate higher revenue. - Returning customers who make more frequent orders over time and order more items in their transactions are likely to generate a good portion of overall sales revenue. - Hopefully with this, Global Superstore can have better revenue predictions by understanding the distribution and growth potential of high-value customers. - At the same time, this also gives the opportunity to generate communication strategies for high-value customers, such as exclusive offers or premium services.