Exploring a Hypothetical Store's Shopper Data Dataset in Anticipation of the Black Friday Holiday Season

Data Science: Capstone Project for Harvardx Professional Data Science Certificate
(Choose your own project: PH125.9x)

Delpagodage Ama Nayanahari Jayaweera

05 June 2024

Contents

1 Chapter 1

1.1 Overview

The purpose of this project is to apply advanced machine learning techniques to a publicly available dataset as part of the ‘Data Science: Capstone’ course (PH125.9x)offered by edX HarvardX.

The project goes beyond focuses on exploring and analyzing the data using machine learning algorithms.The goal is to gain insights from the analysis and effectively communicate the process and findings [1].

1.2 Introduction

The term ‘Black Friday’ didn’t start as a shopping-related term. Instead, it emerged during a financial crisis.This explanation looks back at the historical context that led to the term’s creation, tracing its beginnings to a significant event that happened on a memorable day in September.The term ‘Black Friday’ first appeared in history on September 24, 1869 [1].

Black Friday is a major shopping event that happens on the day after Thanksgiving in the United States.It’s seen as the start of the Christmas shopping season and is known for big discounts and special offers from stores.Black Friday is one of the busiest shopping days of the year, with shoppers lining up early in the morning to get great deals. In recent years, online shopping’s popularity has led to the rise of Cyber Monday, which takes place the Monday after Black Friday and offers online sales and discounts. Black Friday is now observed in other parts of the world, like Canada, the United Kingdom, and Australia [2].

In retrospection, the origins of “Black Friday” emerge not as a celebration of consumerism, but rather as a historical marker of a financial crisis of considerable magnitude.The term’s inception, intertwined with the machinations of Gould and Fisk, serves as a poignant reminder of the intricacies and fragility inherent within the realm of finance.The events of that fateful September day in 1,869 stand as a testament to the far-reaching consequences of unscrupulous financial manipulation, casting a somber and profound shadow over the narrative of “Black Friday”.

The modern incarnation of Black Friday has evolved into a grand spectacle characterized by an array of sales, enticing promotions, and serpentine queues forming outside retail establishments.Esteemed retailers including Target, Best Buy, Amazon, among others, eagerly anticipate this annual occasion, anticipating that consumers will seize the opportunity to partake in extraordinary bargains and exclusive offers.

Beyond its immediate scope, the term “Black Friday” has also catalyzed the emergence of additional retail-themed observances, including “Cyber Monday,” “Small Business Saturday,” and “Giving Tuesday”[3]. Herein lie several noteworthy statistical highlights extracted from the events of “Black Friday” in the year 2018:

1. Foot traffic of patrons within physical stores experienced a marginal decline of 1.7% as compared to the preceding year of 2017.
2. Online consumer expenditures witnessed a robust surge, tallying an impressive $6.22 billion, reflecting a substantial increase of 23.6% from the corresponding figures of 2017.

1.3 Executive Summary

The main goal of this project is to harness the power of machine learning models and vector support techniques to analyze and forecast the sales volume of “Black Friday.” Intend to explore a variety of factors, including Gender, Top Sellers, Age, City, Marital Status, and occupation, to gain insights into what drives sales during this event.

To achieve this objective, employ data transformation and feature engineering techniques. These processes are crucial for enhancing the accuracy of predictions. We will strive to optimize the models by utilizing various approaches and methods.

As part of this project, delve into several modeling techniques to assess their effectiveness in predicting Black Friday sales.Throughout the project, utilize various evaluation metrics to measure the performance of each modeling approach.

Aim is to identify the most suitable model that provides the most accurate predictions.This project not only involves data analysis and machine learning but also the critical task of selecting the most effective model to improve understanding of Black Friday sales patterns.

2 Chapter 2

2.1 Method and Analysis

In Exploratory Data Analysis (EDA), methods and analysis refer to the techniques and approaches used to examine and understand the characteristics of a dataset without making any formal assumptions about the underlying distribution or relationships.

EDA is a crucial step in the data analysis process as it helps uncover patterns, trends, anomalies, and relationships within the data. Here's a brief overview of methods and analysis in EDA:

1. **Descriptive Statistics** Descriptive statistics provide a summary of the main aspects of the dataset.Measures like the mean (average), median (middle value), and mode (most frequent value) offer insights into the central tendency of the data.
2. **Range and Variability** Understanding the range (difference between the maximum and minimum values) and variability (standard deviation, interquartile range) helps gauge the spread of the data.
3. **Frequency Distribution** Creating histograms and frequency tables helps visualize the distribution of values in the dataset.
4. **Data Visualization**
   i. **Histograms:** A graphical representation of the distribution of a dataset, showing the frequency of different values.
   ii. **Box Plots (Box-and-Whisker Plots):** Box plots provide a visual summary of the distribution, including the median, quartiles, and potential outliers.
   iii. **Scatter Plots:** Used to explore relationships between two continuous variables. Each point on the plot represents a data point.
   iv. **Pair Plots:** For multivariate analysis, pair plots display scatter plots for all pairs of variables in the dataset.
   v. **Heatmaps:** Visualizing the correlation matrix to understand relationships between variables.
5. **Data Cleaning** Identifying and handling missing data.Dealing with outliers that might affect the analysis.
   i. **Dimensionality Reduction:**Techniques like Principal Component Analysis (PCA) or t-Distributed Stochastic Neighbor Embedding (t-SNE) can be applied to visualize high-dimensional data in a lower-dimensional space.
   ii. **Pattern Recognition:** Identifying patterns or trends in the data that might indicate interesting features or relationships.
   iii. **Statistical Tests:** Conducting basic statistical tests to check assumptions or explore relationships, though this is more common in confirmatory data analysis.
6. **Interactive Exploration:** Using tools like interactive dashboards or applications to explore the data dynamically.

The goal of EDA is to gain insights into the data, generate hypotheses, and inform the next steps in the analysis.It's a flexible and iterative process that allows analysts to adapt their approach based on the discoveries made during exploration.

2.1.1 Exploratory Data Analysis (EDA)

Commencing endeavor, initiate by loading the dataset that will serve as the foundation for forthcoming Exploratory Data Analysis (EDA).

```
dataset = read.csv("BlackFriday.csv")
```

Next, proceed to import the essential libraries that shall constitute the backbone of analytical framework within this kernel.

```
if (!require(tidyverse)) {
  install.packages("tidyverse", repos = "http://cran.us.r-project.org")
}

if (!require(scales)) {
  install.packages("scales", repos = "http://cran.us.r-project.org")
}

if (!require(arules)) {
  install.packages("arules", repos = "http://cran.us.r-project.org")
}

if (!require(gridExtra)) {
  install.packages("gridExtra", repos = "http://cran.us.r-project.org")
}

if (!require(purrr)) {
  install.packages("purrr", repos = "http://cran.us.r-project.org")
}

if (!require(readr)) {
  install.packages("readr", repos = "http://cran.us.r-project.org")
}

if (!require(tidyr)) {
  install.packages("tidyr", repos = "http://cran.us.r-project.org")
}

if (!require(dplyr)) {
  install.packages("dplyr", repos = "http://cran.us.r-project.org")
}

if (!require(arulesViz)) {
  install.packages("arulesViz", repos = "http://cran.us.r-project.org")
}


library(tidyverse)
library(scales)
library(arules)
library(gridExtra)
library(purrr)
library(readr)
```

```r
library(tidyr)
library(dplyr)
```

For the purpose of visualizing and delving into dataset, harness the capabilities of the “tidyverse” package.This package is renowned for its user-friendly syntax and an extensive array of valuable functions. To further enhance the visual representation of plots, also enlist the “scales” package, facilitating tailored adjustments to plot axes.In the culminating phase of this kernel, the “arules” package will come to the fore, playing an integral role in Association Rule Learning and the Apriori algorithm.

Comprehensive information pertaining to all the packages integrated into this EDA can be found in the Works Cited section of this kernel.Now, let us embark upon journey with an initial high-level overview of the entirety of the dataset.

```r
summary(dataset)
```

```
##     User_ID         Product_ID           Gender              Age
##  Min.   :1000001   Length:537577      Length:537577      Length:537577
##  1st Qu.:1001495   Class :character   Class :character   Class :character
##  Median :1003031   Mode  :character   Mode  :character   Mode  :character
##  Mean   :1002992
##  3rd Qu.:1004417
##  Max.   :1006040
##
##    Occupation     City_Category      Stay_In_Current_City_Years
##  Min.   : 0.000   Length:537577      Length:537577
##  1st Qu.: 2.000   Class :character   Class :character
##  Median : 7.000   Mode  :character   Mode  :character
##  Mean   : 8.083
##  3rd Qu.:14.000
##  Max.   :20.000
##
##  Marital_Status   Product_Category_1 Product_Category_2 Product_Category_3
##  Min.   :0.0000   Min.   : 1.000     Min.   : 2.00      Min.   : 3.0
##  1st Qu.:0.0000   1st Qu.: 1.000     1st Qu.: 5.00      1st Qu.: 9.0
##  Median :0.0000   Median : 5.000     Median : 9.00      Median :14.0
##  Mean   :0.4088   Mean   : 5.296     Mean   : 9.84      Mean   :12.7
##  3rd Qu.:1.0000   3rd Qu.: 8.000     3rd Qu.:15.00      3rd Qu.:16.0
##  Max.   :1.0000   Max.   :18.000     Max.   :18.00      Max.   :18.0
##                                      NA's   :166986     NA's   :373299
##     Purchase
##  Min.   :  185
##  1st Qu.: 5866
##  Median : 8062
##  Mean   : 9334
##  3rd Qu.:12073
##  Max.   :23961
##
```

```r
head(dataset)
```

```
##   User_ID Product_ID Gender   Age Occupation City_Category
## 1 1000001  P00069042      F  0-17         10             A
## 2 1000001  P00248942      F  0-17         10             A
## 3 1000001  P00087842      F  0-17         10             A
## 4 1000001  P00085442      F  0-17         10             A
## 5 1000002  P00285442      M   55+         16             C
## 6 1000003  P00193542      M 26-35         15             A
##   Stay_In_Current_City_Years Marital_Status Product_Category_1
## 1                          2              0                  3
## 2                          2              0                  1
## 3                          2              0                 12
## 4                          2              0                 12
## 5                         4+              0                  8
## 6                          3              0                  1
##   Product_Category_2 Product_Category_3 Purchase
## 1                 NA                 NA     8370
## 2                  6                 14    15200
## 3                 NA                 NA     1422
## 4                 14                 NA     1057
## 5                 NA                 NA     7969
## 6                  2                 NA    15227
```

Dataset reveals a total of 12 distinct columns, with each column aligning itself with a corresponding variable outlined below.

1. **User_ID:** Unique identifier of shopper.
2. **Product_ID:** Unique identifier of product. (No key given)
3. **Gender:** Sex of shopper.
4. **Age:** Age of shopper split into bins.
5. **Occupation:** Occupation of shopper. (No key given)
6. **City_Category:** Residence location of shopper. (No key given)
7. **Stay_In_Current_City_Years:** Number of years stay in current city.
8. **Marital_Status:** Marital status of shopper.
9. **Product_Category_1:** Product category of purchase.
10. **Product_Category_2:** Product may belong to other category.
11. **Product_Category_3:** Product may belong to other category.
12. **Purchase:** Purchase amount in dollars.

Upon perusing the initial rows of dataset, a distinctive pattern emerges:

1. Each row encapsulates a discrete transaction, signifying an individual item procured by a specific customer.

2. As an analysis advances, this delineation will assume paramount significance when aggregate transactions based on User_ID, culminating in an aggregation of purchases attributed to each unique customer.

A notable critique pertinent to this dataset pertains to the absence of a definitive key that correlates various Product_IDs with their corresponding item descriptions. This lack of explicit linkage (for instance, an inability to seamlessly correlate P00265242 with a readily identifiable item) potentially poses a challenge.

In a real-world context, the ideal scenario would involve a supplementary dataset furnishing comprehensive information, coupling item names with their respective Product_IDs.The integration of such supplementary data, while not directly influencing EDA, would substantially enhance the efficacy of the Apriori algorithm implementation and foster a more lucid interpretation of certain facets within EDA.

2.1.2 Gender

Initiating exploratory journey, initial focal point shall be the examination of the gender distribution among shoppers frequenting this establishment. Given that each row corresponds to a distinct transaction, a preliminary step entails grouping the data by User_ID to eliminate any duplicate entries. This process will serve to streamline the analysis and facilitate an accurate portrayal of the gender distribution within the shopper demographic.

```
dataset_gender = dataset %>%
                    select(User_ID, Gender) %>%
                    group_by(User_ID) %>%
                    distinct()

head(dataset_gender)
```

```
## # A tibble: 6 x 2
## # Groups:   User_ID [6]
##   User_ID Gender
##     <int> <chr>
## 1 1000001 F
## 2 1000002 M
## 3 1000003 M
## 4 1000004 M
## 5 1000005 M
## 6 1000006 F
```

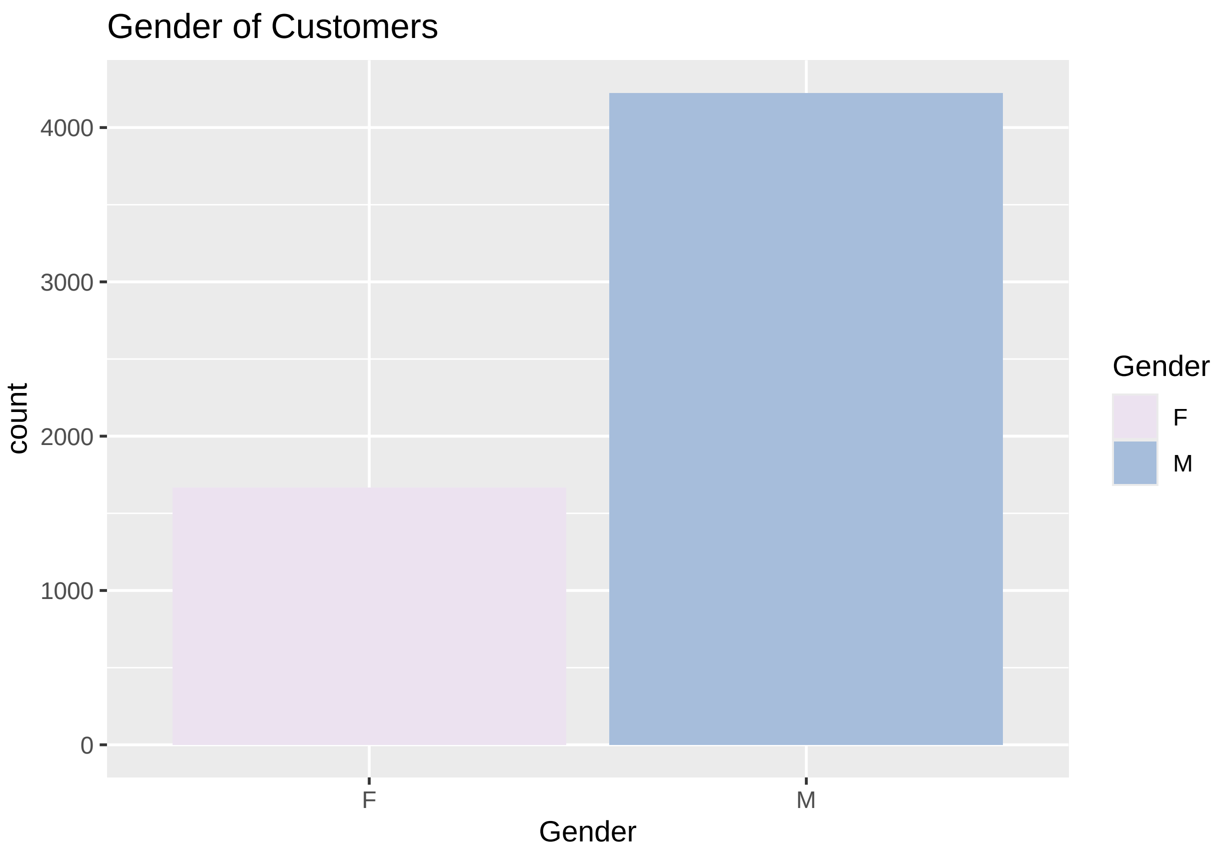
```
summary(dataset_gender$Gender)
```

```
##    Length     Class      Mode
##      5891 character character
```

With the requisite dataframe in place, encapsulating the correlation between each User_ID and their associated gender, coupled with the comprehensive counts for reference, primed to depict the gender distribution across dataset through an illustrative plot.

```
options(scipen=10000)    # To remove scientific numbering

genderDist  = ggplot(data = dataset_gender) +
                geom_bar(mapping = aes(x = Gender, y = ..count.., fill = Gender)) +
                labs(title = 'Gender of Customers') +
                scale_fill_brewer(palette = 'PuBuGn')
print(genderDist)
```



Evidently, the male demographic significantly outweighs the female counterpart in terms of shopping participation on Black Friday within store. This gender distribution metric assumes particular significance for retailers, as it can potentially steer decisions regarding store layout, product assortment, and other variables, contingent upon the proportion of male and female shoppers.

2.1.3 Purchases between Female and Male shoppers

Citing a research study from the Clothing and Textiles Research Journal, it is revealed that certain factors such as involvement, variety seeking, and the physical store environment are antecedents of shopping experience satisfaction.

The study further suggests the mediating role of hedonic shopping value in shopping satisfaction, a correlation confirmed among female subjects, yet not among male respondents (Chang, E., Burns, L. D., & Francis, S. K., 2004). While this may not yield immediate prescriptive insights for retail establishments, it underscores a gender-based disparity in the derived value of shopping and its nexus with gender, serving as a compelling consideration for retailers.

To delve deeper into analysis, let us proceed to compute the average spending amount in relation to gender.To facilitate interpretability and traceability, construct distinct tables before ultimately merging them for a holistic perspective.

```
total_purchase_user = dataset %>%
                        select(User_ID, Gender, Purchase) %>%
                        group_by(User_ID) %>%
                        arrange(User_ID) %>%
                        summarise(Total_Purchase = sum(Purchase))
```

```r
user_gender = dataset %>%
                select(User_ID, Gender) %>%
                group_by(User_ID) %>%
                arrange(User_ID) %>%
                distinct()

head(user_gender)
```

```
## # A tibble: 6 x 2
## # Groups:   User_ID [6]
##   User_ID Gender
##     <int> <chr>
## 1 1000001 F
## 2 1000002 M
## 3 1000003 M
## 4 1000004 M
## 5 1000005 M
## 6 1000006 F
```

```r
head(total_purchase_user)
```

```
## # A tibble: 6 x 2
##   User_ID Total_Purchase
##     <int>          <int>
## 1 1000001         333481
## 2 1000002         810353
## 3 1000003         341635
## 4 1000004         205987
## 5 1000005         821001
## 6 1000006         379450
```

```r
user_purchase_gender = full_join(total_purchase_user, user_gender, by = "User_ID")
head(user_purchase_gender)
```

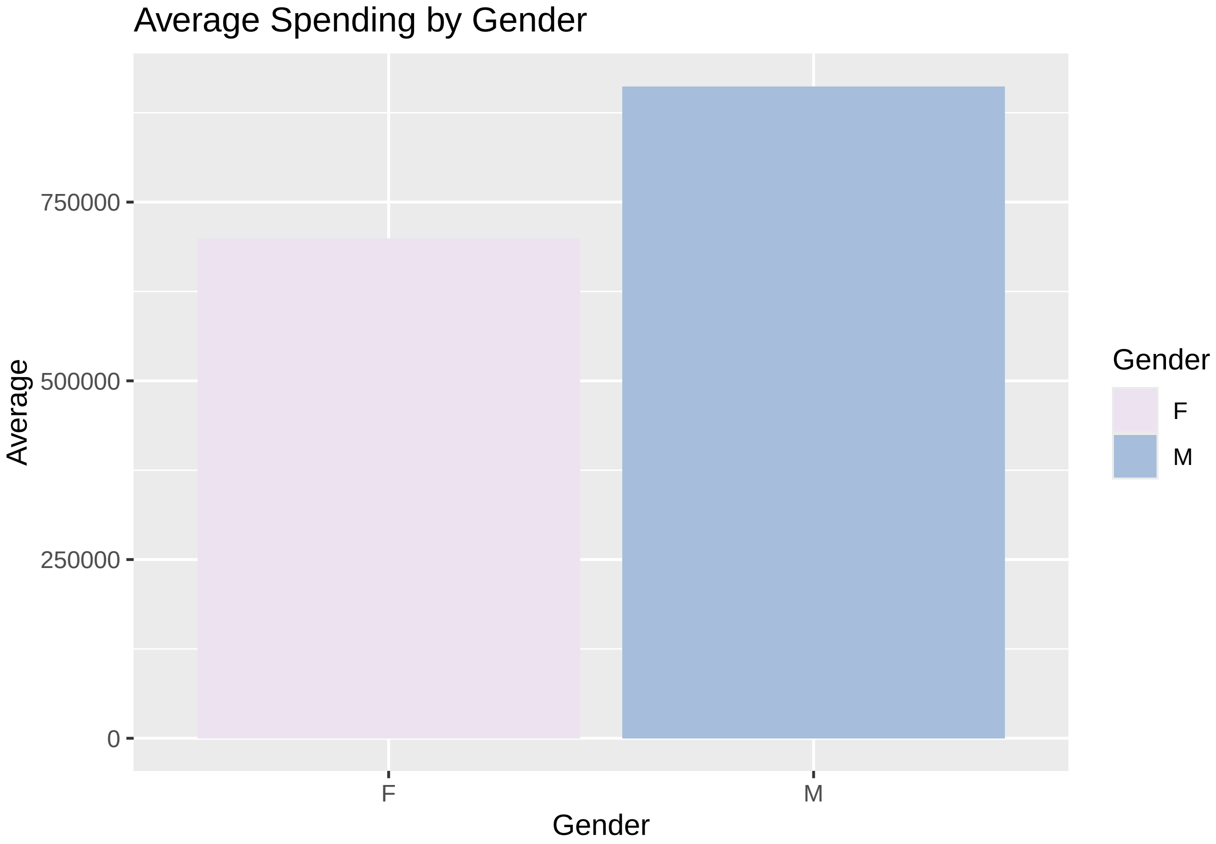
```
## # A tibble: 6 x 3
##   User_ID Total_Purchase Gender
##     <int>          <int> <chr>
## 1 1000001         333481 F
## 2 1000002         810353 M
## 3 1000003         341635 M
## 4 1000004         205987 M
## 5 1000005         821001 M
## 6 1000006         379450 F
```

```r
average_spending_gender = user_purchase_gender %>%
                          group_by(Gender) %>%
                          summarize(Purchase = sum(as.numeric(Total_Purchase)),
                                    Count = n(),
                                    Average = Purchase/Count)
head(average_spending_gender)
```

```
## # A tibble: 2 x 4
##   Gender   Purchase Count Average
##   <chr>       <dbl> <int>   <dbl>
## 1 F      1164624021  1666 699054.
## 2 M      3853044357  4225 911963.
```

The calculated averages indicate that the average transaction amount for females stood at 699,054.00, whereas for males, it reached 911,963.20. To provide a visual representation of findings, let us proceed to create a visual depiction of these results.

```
genderAverage  = ggplot(data = average_spending_gender) +
                    geom_bar(mapping = aes(x = Gender, y = Average,
                                           fill = Gender), stat = 'identity') +
                    labs(title = 'Average Spending by Gender') +
                    scale_fill_brewer(palette = 'PuBuGn')
print(genderAverage)
```



A compelling observation comes to light through visualization.Despite the relative disparity in the frequency of purchases between female and male shoppers within this particular store, the average transaction amount for females is remarkably close to that of their male counterparts. It is imperative to exercise caution in interpreting these results, recognizing the need to consider the scale of these expenditures.While females are nearly matching the average spending of males, it's important to underscore that, on average, their expenditures are still approximately 250,000 units lower than those of males. This insight underscores the significance of contextualizing the data within a broader framework.

2.1.4 Top Sellers

Transitioning focus, let us now embark on an exploration of our best-performing products. In this context, forego the grouping of data by product ID, as our intention is to retain duplicate entries. This approach ensures a comprehensive examination, accounting for scenarios where customers procure multiple quantities of the same product.

```
top_sellers = dataset %>%
                count(Product_ID, sort = TRUE)

top_5 = head(top_sellers, 5)

top_5
```

```
##   Product_ID    n
## 1  P00265242 1858
## 2  P00110742 1591
## 3  P00025442 1586
## 4  P00112142 1539
## 5  P00057642 1430
```

Looks like top 5 best sellers are (by product ID):

1. P00265242 = 1858
2. P00110742 = 1591
3. P00025442 = 1586
4. P00112142 = 1539
5. P00057642 = 1430

Having successfully identified the top 5 best-selling products, trajectory leads us to a closer examination of the best-performing individual product, denoted as P00265242.

This granular analysis aims to provide a more comprehensive understanding of the specific attributes and dynamics that contribute to the exceptional sales performance of this particular product.

```
best_seller = dataset[dataset$Product_ID == 'P00265242', ]

head(best_seller)
```

```
##      User_ID Product_ID Gender   Age Occupation City_Category
## 400  1000066  P00265242      M 26-35         18             C
## 1192 1000196  P00265242      F 36-45          9             C
## 1373 1000222  P00265242      M 26-35          1             A
## 1846 1000301  P00265242      M 18-25          4             B
## 2210 1000345  P00265242      M 26-35         12             A
## 2405 1000383  P00265242      F 26-35          7             A
##      Stay_In_Current_City_Years Marital_Status Product_Category_1
## 400                           2              0                  5
## 1192                         4+              0                  5
## 1373                          1              0                  5
## 1846                         4+              0                  5
## 2210                          2              1                  5
## 2405                         4+              1                  5
##      Product_Category_2 Product_Category_3 Purchase
## 400                   8                 NA     8652
## 1192                  8                 NA     8767
## 1373                  8                 NA     6944
## 1846                  8                 NA     8628
## 2210                  8                 NA     8593
## 2405                  8                 NA     6998
```

From analysis, it is evident that this particular product falls within the categorical indices of Product_Category_1 = 5 and Product_Category_2 = 8. As highlighted earlier, the absence of an explicit key detailing item names poses a limitation in identifying the precise nature of this product. An intriguing revelation surfaces as observe variations in the purchase prices of the same product across different customers.

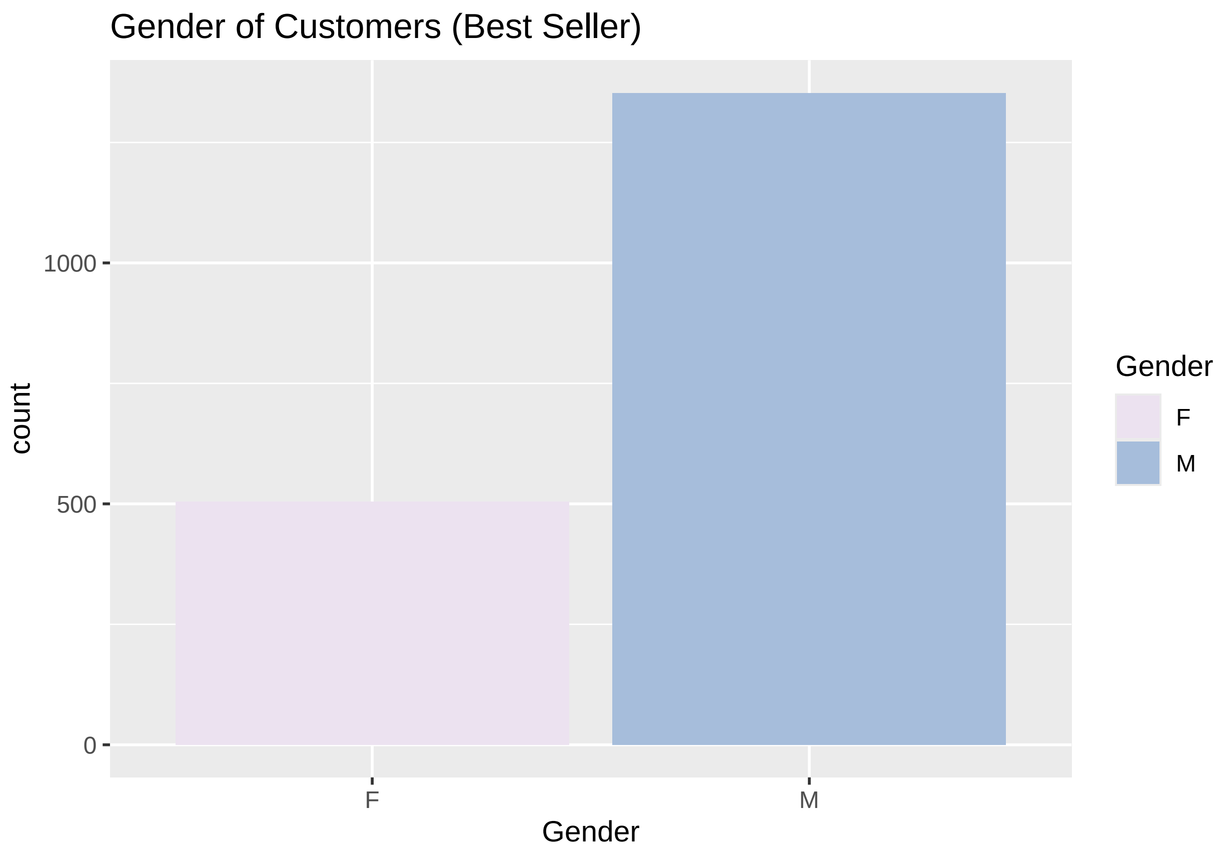
This phenomenon could potentially be attributed to an array of factors, encompassing "Black Friday" promotions, discounts, or the utilization of distinct coupon codes.

Alternatively, a deeper investigation may be warranted to unearth the underlying rationale for the disparities in purchase prices of an identical product among diverse customers.

2.1.5 Correlation between Gender and the Best Selling Product

In pursuit of a comprehensive analysis, inquiry advances to ascertain if any discernible correlation between gender and the best-selling product, P00265242, is discernible.

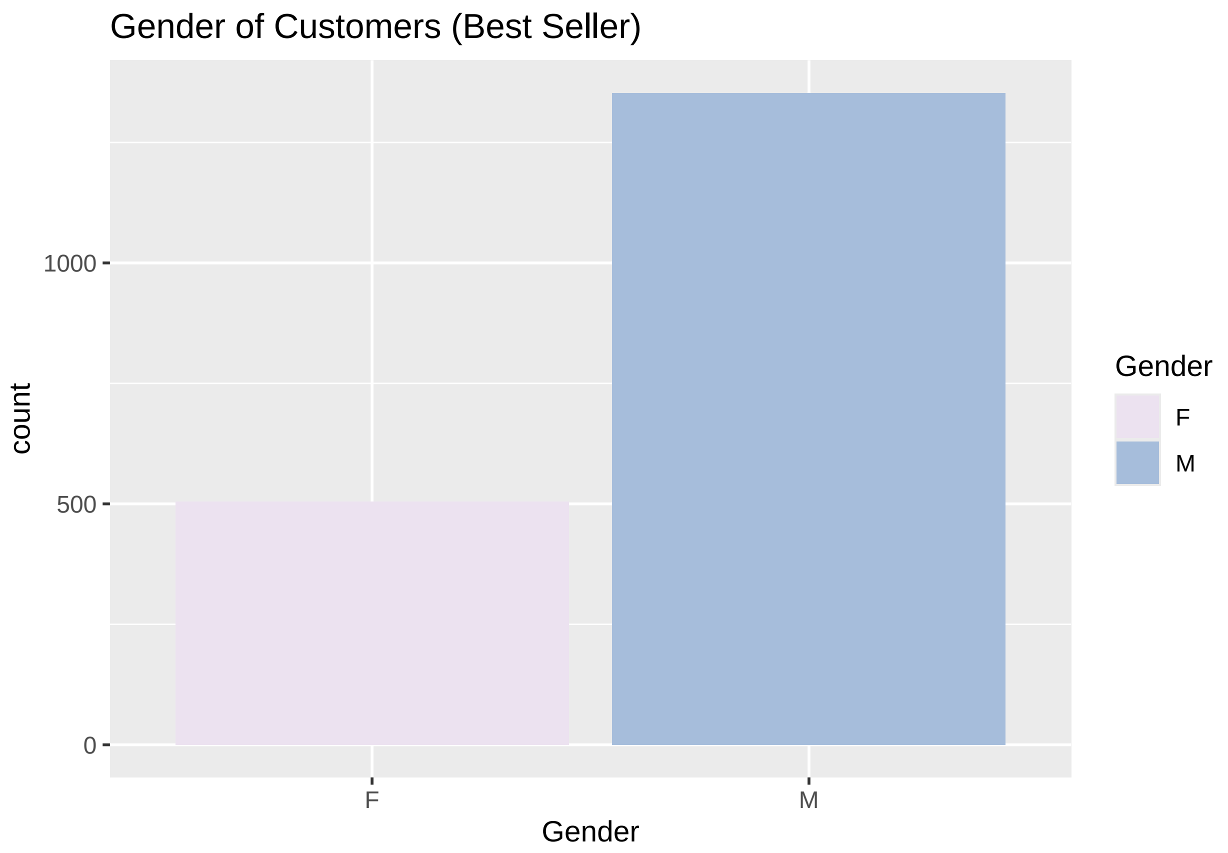
```
genderDist_bs = ggplot(data = best_seller) + geom_bar(mapping = aes(x = Gender,
                   y = ..count.., fill = Gender)) +
                   labs(title = 'Gender of Customers (Best Seller)') +
                   scale_fill_brewer(palette = 'PuBuGn')
print(genderDist_bs)
```



A similar distribution between genders to overall dataset gender split - lets confirm.

```
genderDist_bs  = ggplot(data = best_seller) +
            geom_bar(mapping = aes(x = Gender, y = ..count.., fill = Gender)) +
            labs(title = 'Gender of Customers (Best Seller)') +
            scale_fill_brewer(palette = 'PuBuGn')

print(genderDist_bs)
```

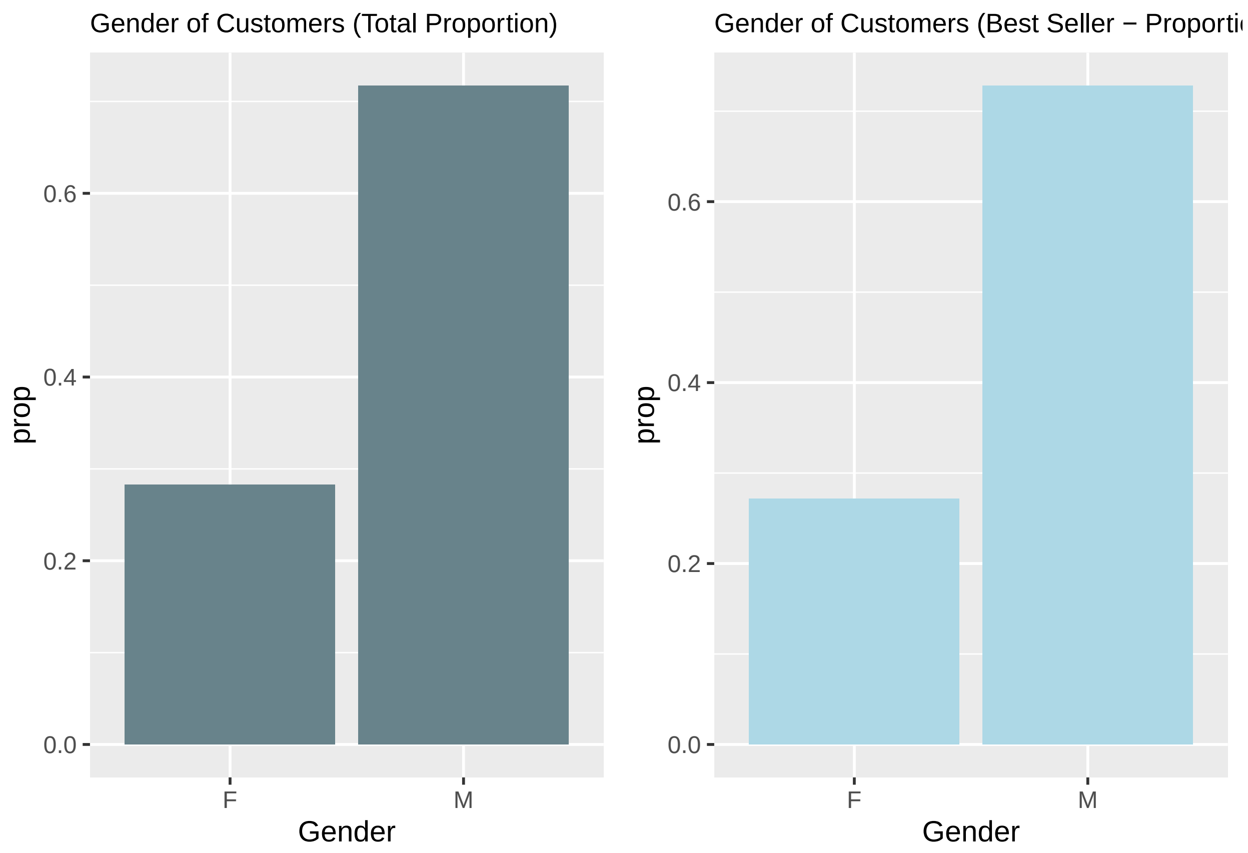


Upon a comprehensive review of the aggregate dataset, it is apparent that both the purchasers of the best-selling product and the purchasers of all products collectively exhibit a relatively balanced gender distribution, with approximately 25% representing females and 75% representing males. Although a subtle disparity is discernible, the overarching inference suggests that best-selling product does not distinctly cater to a particular gender demographic.Transitioning to the examination of the Age variable, analytically trajectory proceeds.A similar distribution between genders to overall dataset gender split - lets confirm.

```
genderDist_bs_prop = ggplot(data = best_seller) +
              geom_bar(fill = 'lightblue', mapping = aes(x = Gender,
              y = ..prop.., group = 1, fill = Gender)) +
              labs(title = 'Gender of Customers (Best Seller - Proportion)') +
              theme(plot.title = element_text(size=9.5))

genderDist_prop = ggplot(data = dataset_gender) +
              geom_bar(fill = "lightblue4", mapping = aes(x = Gender,
              y = ..prop.., group = 1)) +
              labs(title = 'Gender of Customers (Total Proportion)') +
                      theme(plot.title = element_text(size=9.5))

grid.arrange(genderDist_prop, genderDist_bs_prop, ncol=2)
```



2.1.6 Age

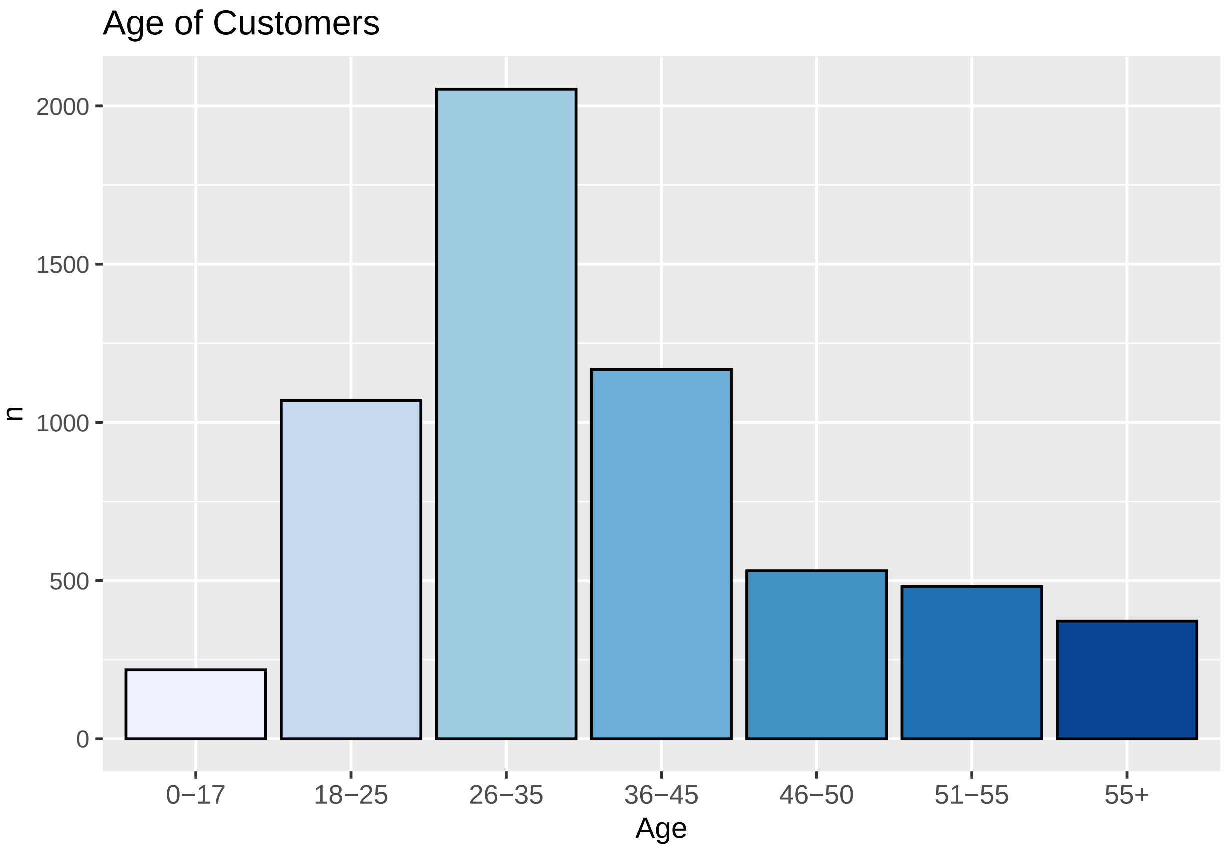
Certainly, let's delve into the analysis of the Age variable step by step. First, create a table that tabulates the count of customers in each age category based on the provided dataset:

```
customers_age = dataset %>%
                    select(User_ID, Age) %>%
                    distinct() %>%
                    count(Age)
customers_age
```

```
##     Age    n
## 1  0-17  218
## 2 18-25 1069
## 3 26-35 2053
## 4 36-45 1167
## 5 46-50  531
## 6 51-55  481
## 7   55+  372
```

Afterward, visualize this distribution using a bar plot:

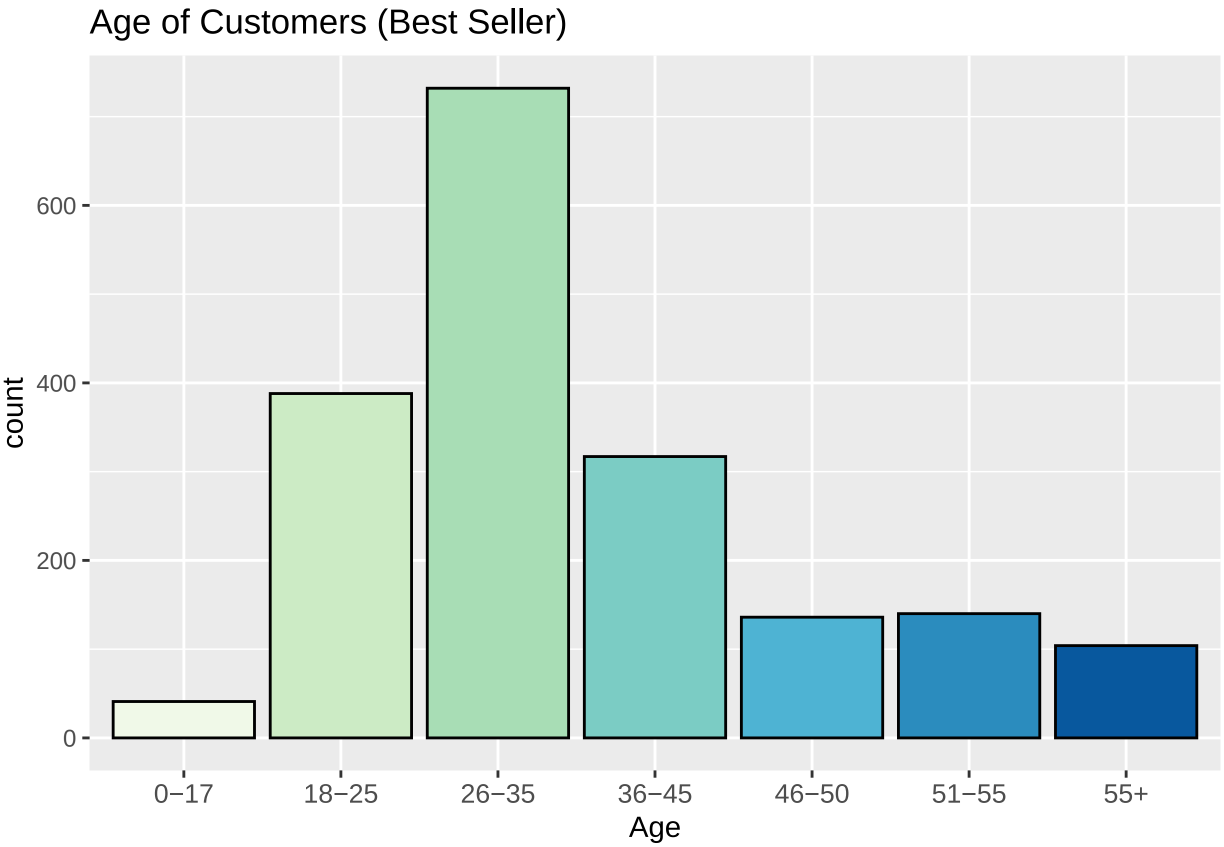
```
customers_age_vis = ggplot(data = customers_age) +
                      geom_bar(color = 'black', stat = 'identity',
                              mapping = aes(x = Age, y = n, fill = Age)) +
                      labs(title = 'Age of Customers') +
                      theme(axis.text.x = element_text(size = 10)) +
                      scale_fill_brewer(palette = 'Blues') +
                      theme(legend.position="none")
print(customers_age_vis)
```



2.1.7 Purchasing behavior of different Age groups

Further, create a similar chart depicting the age distribution specifically within the “best seller” category to identify any potential trends:

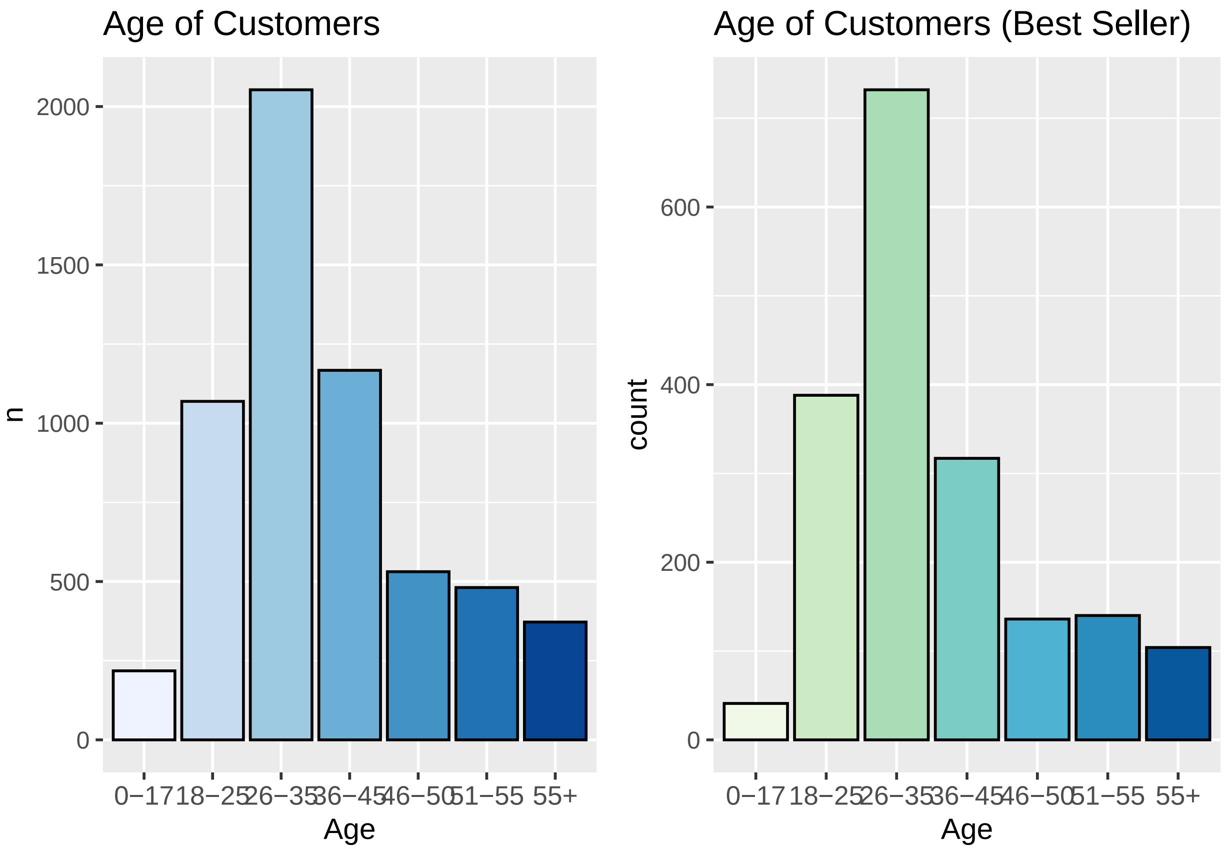
```
ageDist_bs  = ggplot(data = best_seller) +
              geom_bar(color = 'black', mapping = aes(x = Age,
                  y = ..count.., fill = Age)) +
                  labs(title = 'Age of Customers (Best Seller)') +
                  theme(axis.text.x = element_text(size = 10)) +
                  scale_fill_brewer(palette = 'GnBu') +
                  theme(legend.position="none")
print(ageDist_bs)
```



The visual analysis highlights that customers in the age groups of 18-25 and 26-35 constitute the majority of purchasers for the best-selling product. A comparison between the age distribution of the best-selling product and the overall dataset reveals some deviations.

Particularly, customers aged over 45 appear to be slightly less inclined to purchase the top-selling product compared to other products in the dataset.

```r
grid.arrange(customers_age_vis, ageDist_bs, ncol=2)
```



This in-depth exploration of age provides valuable insights into the purchasing behavior of different age groups. As shift focus to another variable, quest for comprehensive analysis continues.

2.1.8 City

Certainly, can proceed by constructing a table that presents each User_ID alongside its corresponding City_Category. This will serve as a foundation step for subsequentanalytical exploration, which will delve into the distribution of shoppers across different city categories.

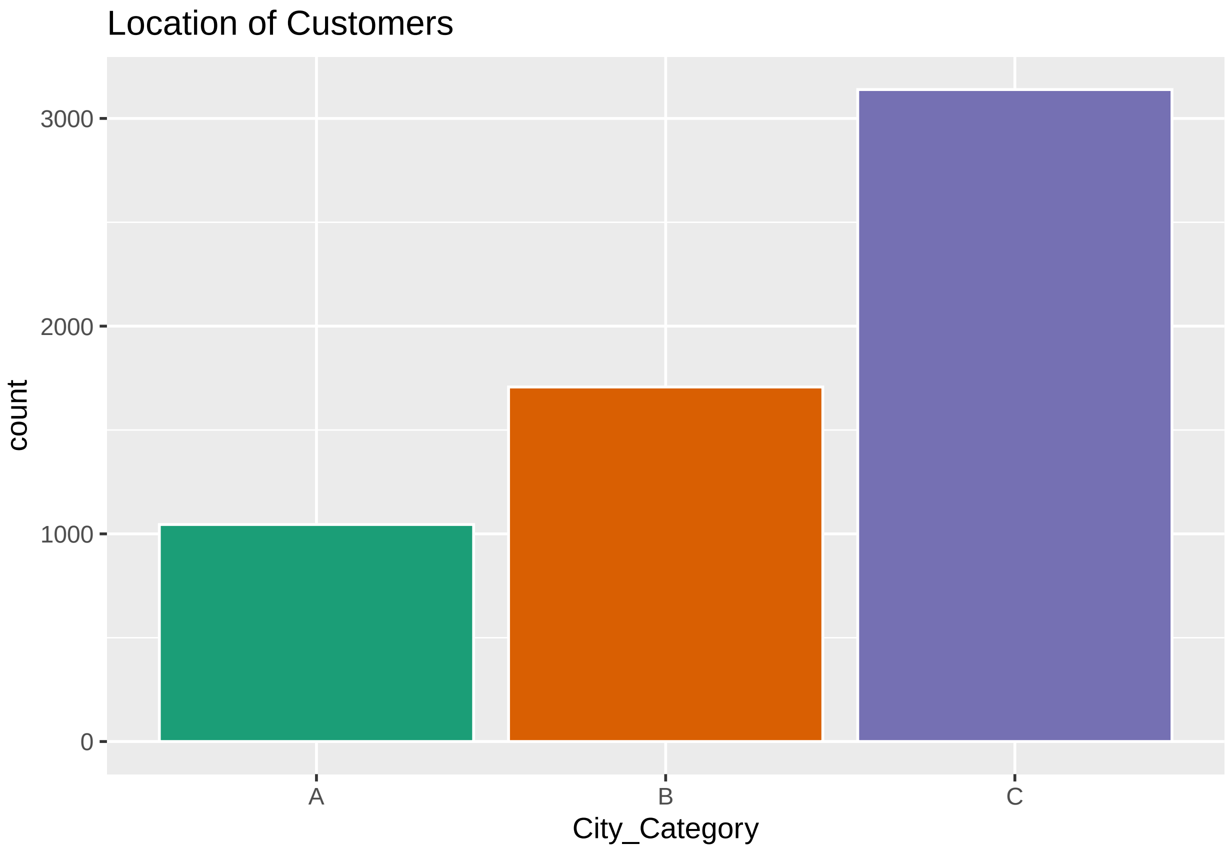
To achieve this, employ the following code:

```
customers_location =  dataset %>%
                            select(User_ID, City_Category) %>%
                            distinct()
head(customers_location)
```

```
##   User_ID City_Category
## 1 1000001             A
## 2 1000002             C
## 3 1000003             A
## 4 1000004             B
## 5 1000005             A
## 6 1000006             A
```

This table will provide a comprehensive snapshot, pairing each distinct User_ID with its associated City_Category. This will serve as a starting point for us to uncover trends and insights related to shopping behavior based on geographical locations.

```
customers_location_vis = ggplot(data = customers_location) +
                        geom_bar(color = 'white', mapping = aes(x = City_Category,
                        y = ..count.., fill = City_Category)) +
                        labs(title = 'Location of Customers') +
                        scale_fill_brewer(palette = "Dark2") +
                        theme(legend.position="none")
print(customers_location_vis)
```



Observation unveils that the majority of customers are residents of City C. Building upon this insight, proceed to calculate the total purchase amount attributed to each city category.

This subsequent analysis aims to shed light on which city's customers have made the most substantial expenditures at store. To initiate this computation, can employ the following code:

```
purchases_city = dataset %>%
                  group_by(City_Category) %>%
                  summarise(Purchases = sum(Purchase))

purchases_city_1000s = purchases_city %>%
                  mutate(purchasesThousands = purchases_city$Purchases / 1000)

purchases_city_1000s
```

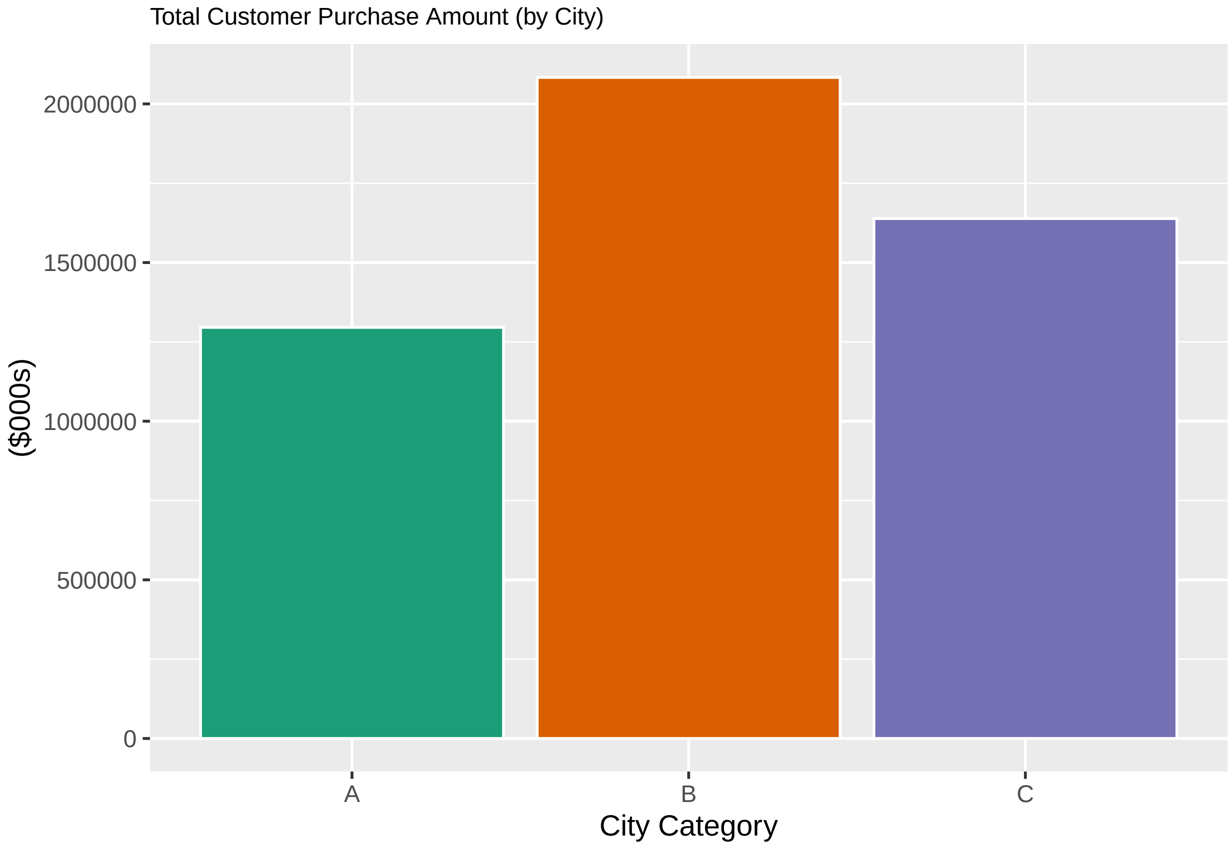
```
## # A tibble: 3 x 3
##   City_Category  Purchases purchasesThousands
##   <chr>              <int>              <dbl>
## 1 A             1295668797          1295669.
## 2 B             2083431612          2083432.
## 3 C             1638567969          1638568.
```

2.1.9 Shopping behaviors across different cities

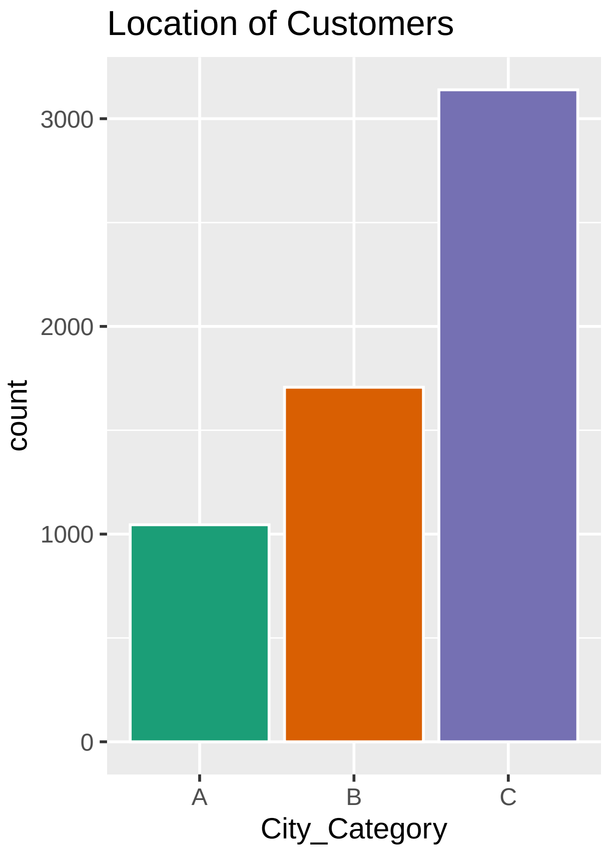
In the interest of enhanced readability and charting, it's a commonplace practice to divide the values in the Purchases column by 1000. This pragmatic approach aligns with prevailing conventions in the business and accounting realms, rendering large numbers more accessible for interpretation and graphical representation.

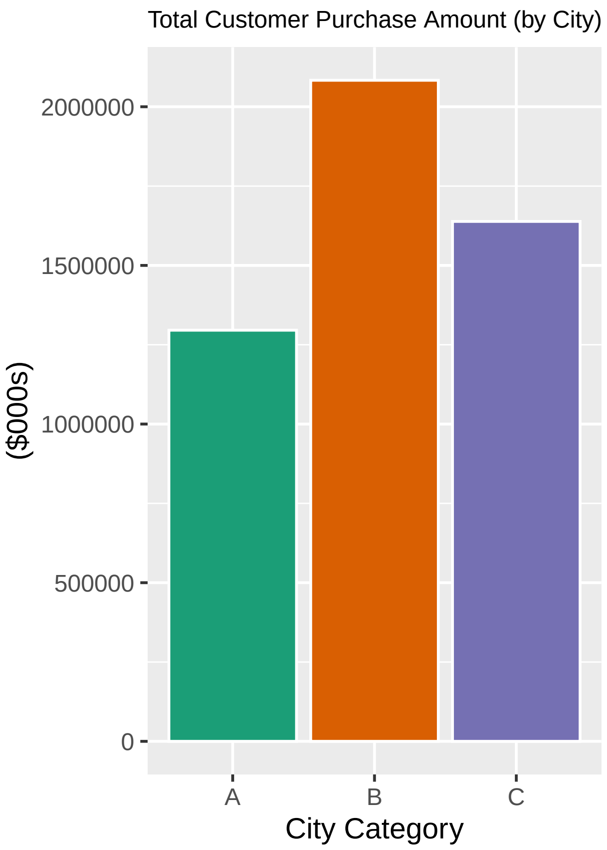
With the requisite table in place, next step involves visualizing the outcomes. To achieve this, can utilize the following code:

```
purchaseCity_vis = ggplot(data = purchases_city_1000s, aes(x = City_Category,
                          y = purchasesThousands, fill = City_Category)) +
                          geom_bar(color = 'white', stat = 'identity') +
                          labs(title = 'Total Customer Purchase Amount (by City)',
                          y = '($000s)', x = 'City Category') +
                          scale_fill_brewer(palette = "Dark2") +
                          theme(legend.position="none",
                                plot.title = element_text(size = 9))
print(purchaseCity_vis)
```



```
grid.arrange(customers_location_vis, purchaseCity_vis, ncol=2)
```





The visualization effectively portrays the shopping dynamics on Black Friday, indicating that City C exhibited the highest shopper frequency at store, while City B surpassed others in terms of total purchase amounts.

To further decipher the rationale behind this trend, exploration continues. To assess the number of purchases made by customers from each city, initiate by calculating the total number of purchases corresponding to each User_ID. The following code accomplishes this:

```
customers = dataset %>%
              group_by(User_ID) %>%
              count(User_ID)
head(customers)
```

```
## # A tibble: 6 x 2
## # Groups:   User_ID [6]
##   User_ID     n
##     <int> <int>
## 1 1000001    34
## 2 1000002    76
## 3 1000003    29
## 4 1000004    13
## 5 1000005   106
## 6 1000006    46
```

This code snippet will yield a table that encapsulates the total number of purchases associated with each unique User_ID. Subsequently, can proceed to extract meaningful insights from this data, shedding light on the shopping behaviors across different cities.

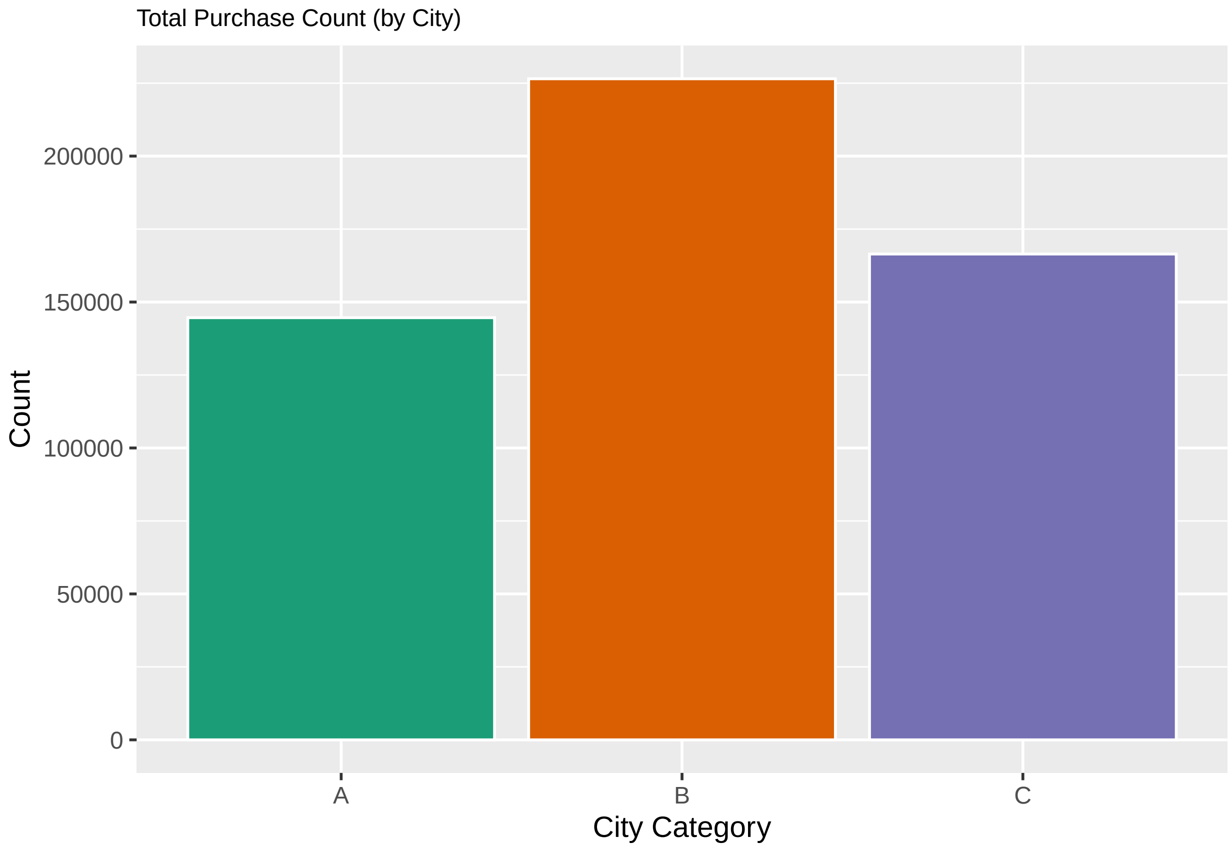
```
customers_City =  dataset %>%
                    select(User_ID, City_Category) %>%
                    group_by(User_ID) %>%
                    distinct() %>%
                    ungroup() %>%
                    left_join(customers, customers_City, by = 'User_ID')
head(customers_City)
```

```
## # A tibble: 6 x 3
##   User_ID City_Category     n
##     <int> <chr>         <int>
## 1 1000001 A                34
## 2 1000002 C                76
## 3 1000003 A                29
## 4 1000004 B                13
## 5 1000005 A               106
## 6 1000006 A                46
```

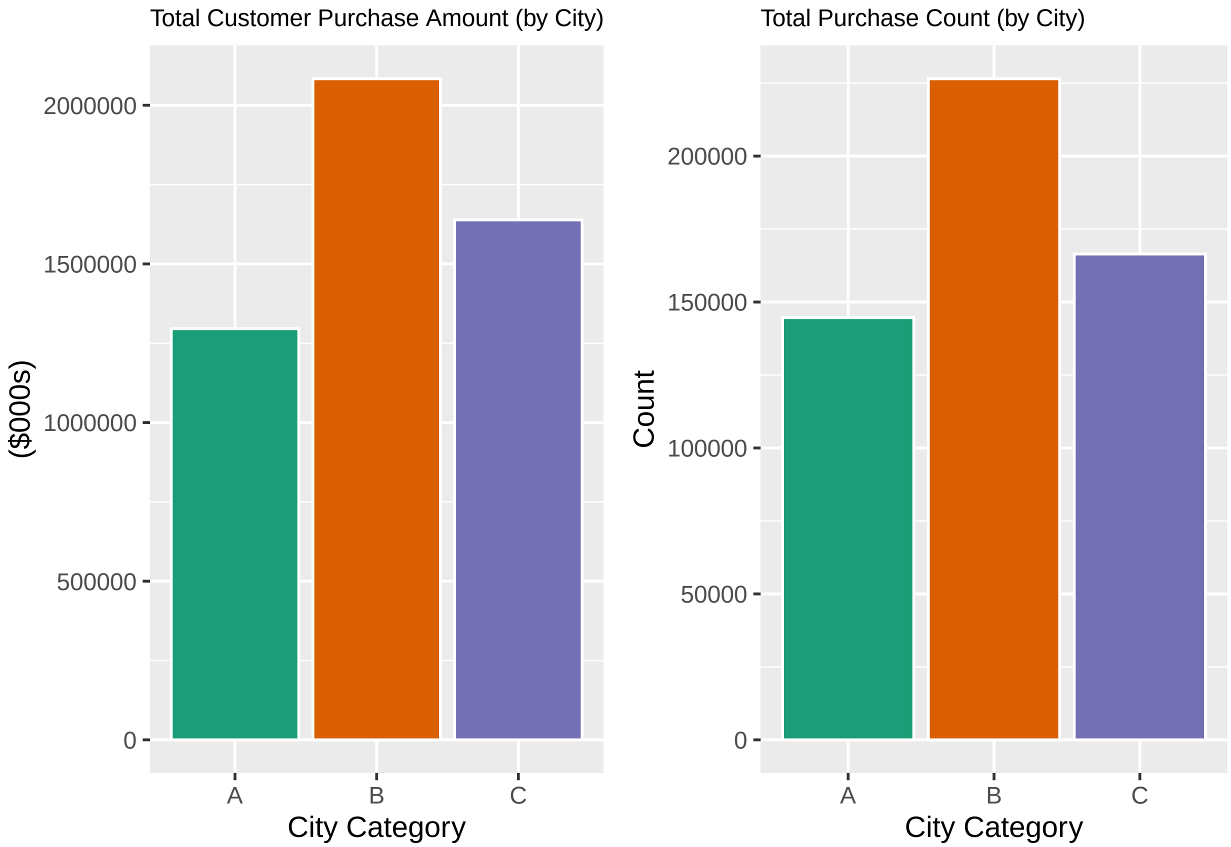
```
city_purchases_count = customers_City %>%
                        select(City_Category, n) %>%
                        group_by(City_Category) %>%
                        summarise(CountOfPurchases = sum(n))
city_purchases_count
```

```
## # A tibble: 3 x 2
##   City_Category CountOfPurchases
##   <chr>                    <int>
## 1 A                       144638
## 2 B                       226493
## 3 C                       166446
```

```
city_count_purchases_vis = ggplot(data = city_purchases_count,
                           aes(x = City_Category, y = CountOfPurchases, fill =
                           City_Category)) + geom_bar(color = 'white',
                           stat = 'identity') + labs(title =
                           'Total Purchase Count (by City)',
                           y = 'Count', x = 'City Category') +
                           scale_fill_brewer(palette = "Dark2") +
                              theme(legend.position="none",
                                    plot.title = element_text(size = 9))
print(city_count_purchases_vis)
```



```r
grid.arrange(purchaseCity_vis, city_count_purchases_vis, ncol = 2)
```


Total Customer Purchase Amount (by City)
($000s)
2000000
1500000
1000000
500000
0
A
B
C
City Category
Total Purchase Count (by City)
Count
200000
150000
100000
50000
0
A
B
C
City Category

3 Chapter 3

3.1 Results and Discussion

The similarity in the distribution patterns between the “Total Count of Purchases” chart and the “Total Customer Purchase Amount” chart indeed implies that customers from City B are making a higher number of purchases rather than opting for more expensive products.

This insight gains traction from the observation that if the case were otherwise, where customers from City B bought more expensive products, likely encounter a scenario where a lower count of purchases correlates with a higher total purchase amount.

Having established this insight, proceed to the subsequent phase of the analysis.Given that the purchase counts across different City_Category segments mirror the distribution patterns of the total purchase amount, inquiry pivots to an examination of the distribution of best-selling product (P00265242) within each City_Category.This examination will potentially unveil nuanced trends or preferences specific to different city categories.

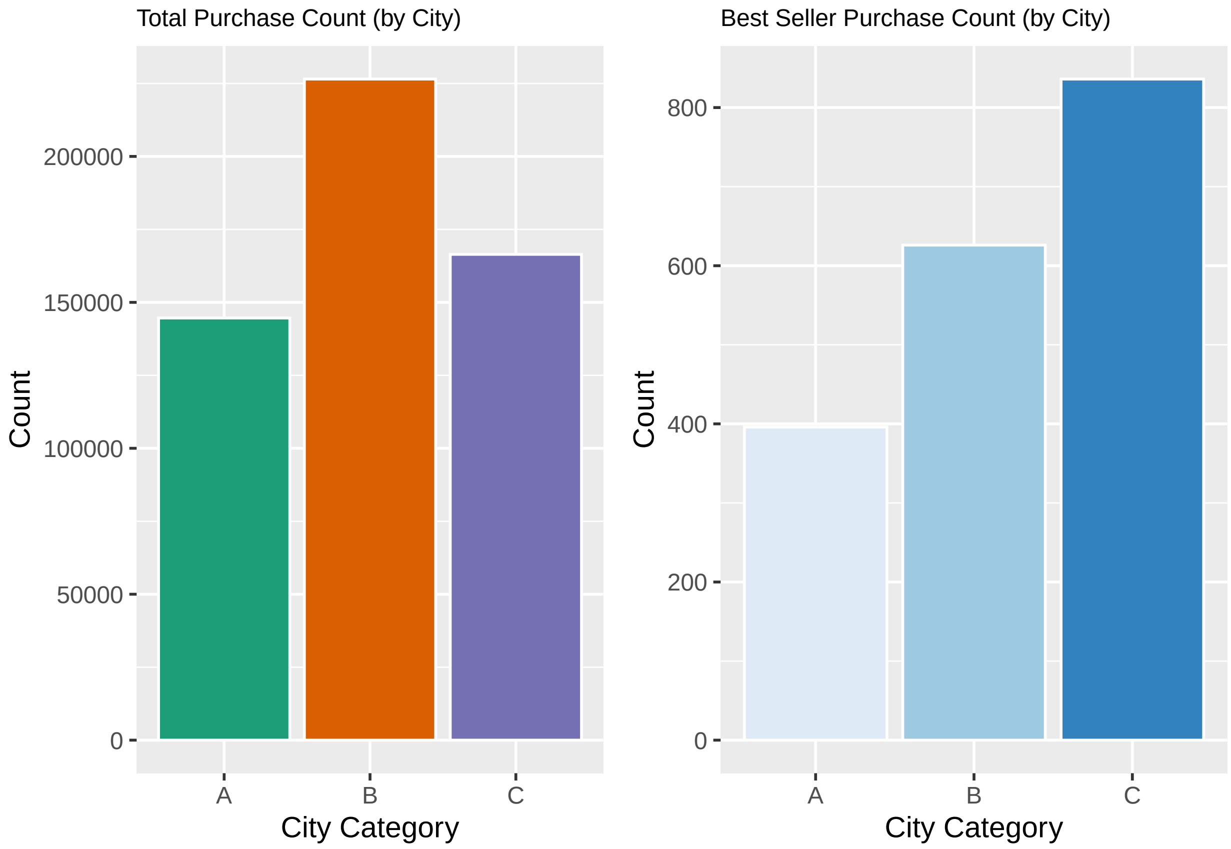
```
head(best_seller)
```

```
##      User_ID Product_ID Gender   Age Occupation City_Category
## 400  1000066  P00265242      M 26-35         18             C
## 1192 1000196  P00265242      F 36-45          9             C
## 1373 1000222  P00265242      M 26-35          1             A
## 1846 1000301  P00265242      M 18-25          4             B
## 2210 1000345  P00265242      M 26-35         12             A
## 2405 1000383  P00265242      F 26-35          7             A
##      Stay_In_Current_City_Years Marital_Status Product_Category_1
## 400                           2              0                  5
## 1192                         4+              0                  5
## 1373                          1              0                  5
## 1846                         4+              0                  5
## 2210                          2              1                  5
## 2405                         4+              1                  5
##      Product_Category_2 Product_Category_3 Purchase
## 400                   8                 NA     8652
## 1192                  8                 NA     8767
## 1373                  8                 NA     6944
## 1846                  8                 NA     8628
## 2210                  8                 NA     8593
## 2405                  8                 NA     6998
```

```
best_seller_city = best_seller %>%
                    select(User_ID, City_Category) %>%
                    distinct() %>%
                    count(City_Category)
best_seller_city
```

```
##   City_Category   n
## 1             A 396
## 2             B 626
## 3             C 836
```

```
best_seller_city_vis = ggplot(data = best_seller_city, aes(x = City_Category,
                              y = n, fill = City_Category)) +
                              geom_bar(color = 'white', stat = 'identity') +
                              labs(title = 'Best Seller Purchase Count (by City)',
                              y = 'Count', x = 'City Category') +
                              scale_fill_brewer(palette = "Blues") +
                              theme(legend.position="none", plot.title =
                                      element_text(size = 9))
grid.arrange(city_count_purchases_vis,best_seller_city_vis, ncol = 2)
```



Indeed, observation is quite intriguing and sheds light on the intricate nuances of customer behavior.While customers from City C exhibit a higher propensity to purchase the best-selling product (P00265242) compared to residents of City A and City B, the overall purchase count from City C lags behind that of City B.

This intriguing contrast underscores the complexity of shopping behaviors across different city categories.It implies that although City C residents are particularly inclined to purchase thebest-selling product, they may not be as prolific in terms of making overall purchases when compared to City B residents.

This revelation underscores the importance of comprehensive analysis in uncovering subtle trends and patterns that may not be immediately evident. As shift focus to another variable, the journey of exploration continues.

3.1.1 Stay in Current City

Certainly, can proceed by examining the distribution of customers who have resided in their respective cities for the longest duration. This exploration will provide insights into the longevity of customer relationships with their residing cities, potentially offering valuable insights into shopping behaviors and patterns influenced by the length of city residence.

To commence this analysis, can employ the following code:

```
customers_stay = dataset %>%
                 select(User_ID, City_Category, Stay_In_Current_City_Years) %>%
                 group_by(User_ID) %>%
                 distinct()
head(customers_stay)
```

```
## # A tibble: 6 x 3
## # Groups:   User_ID [6]
##   User_ID City_Category Stay_In_Current_City_Years
##     <int> <chr>         <chr>
## 1 1000001 A             2
## 2 1000002 C             4+
## 3 1000003 A             3
## 4 1000004 B             2
## 5 1000005 A             1
## 6 1000006 A             1
```

Now that dataset in order, plot and explore. Lets see where most of customers are living.This code will generate a table that encapsulates the total number of purchases for each unique User_ID, categorized by the duration of their stay in their current city.

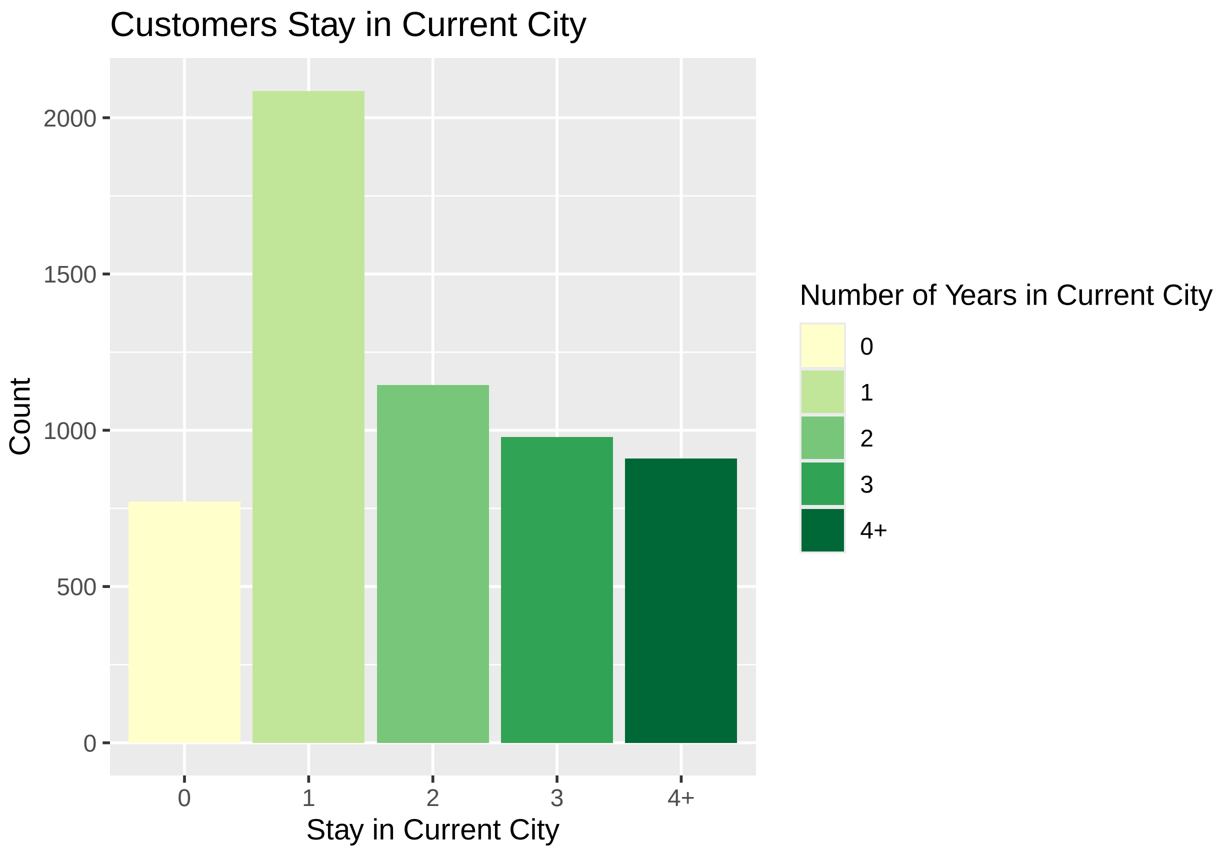
Subsequently, analysis can delve into this data to reveal trends and patterns that might emerge from the distribution of customers' residence duration.

```
residence = customers_stay %>%
                group_by(City_Category) %>%
                tally()
head(residence)
```

```
## # A tibble: 3 x 2
##   City_Category     n
##   <chr>         <int>
## 1 A              1045
## 2 B              1707
## 3 C              3139
```

Looks like most of customers are living in City C. Now, lets investigate further.

```
customers_stay_vis = ggplot(data = customers_stay,
                      aes(x = Stay_In_Current_City_Years, y = ..count.., fill =
                            Stay_In_Current_City_Years)) +
                              geom_bar(stat = 'count') +
                              scale_fill_brewer(palette = 15) +
                              labs(title = 'Customers Stay in Current City',
                                   y = 'Count', x = 'Stay in Current City',
                                   fill = 'Number of Years in Current City')
print(customers_stay_vis)
```

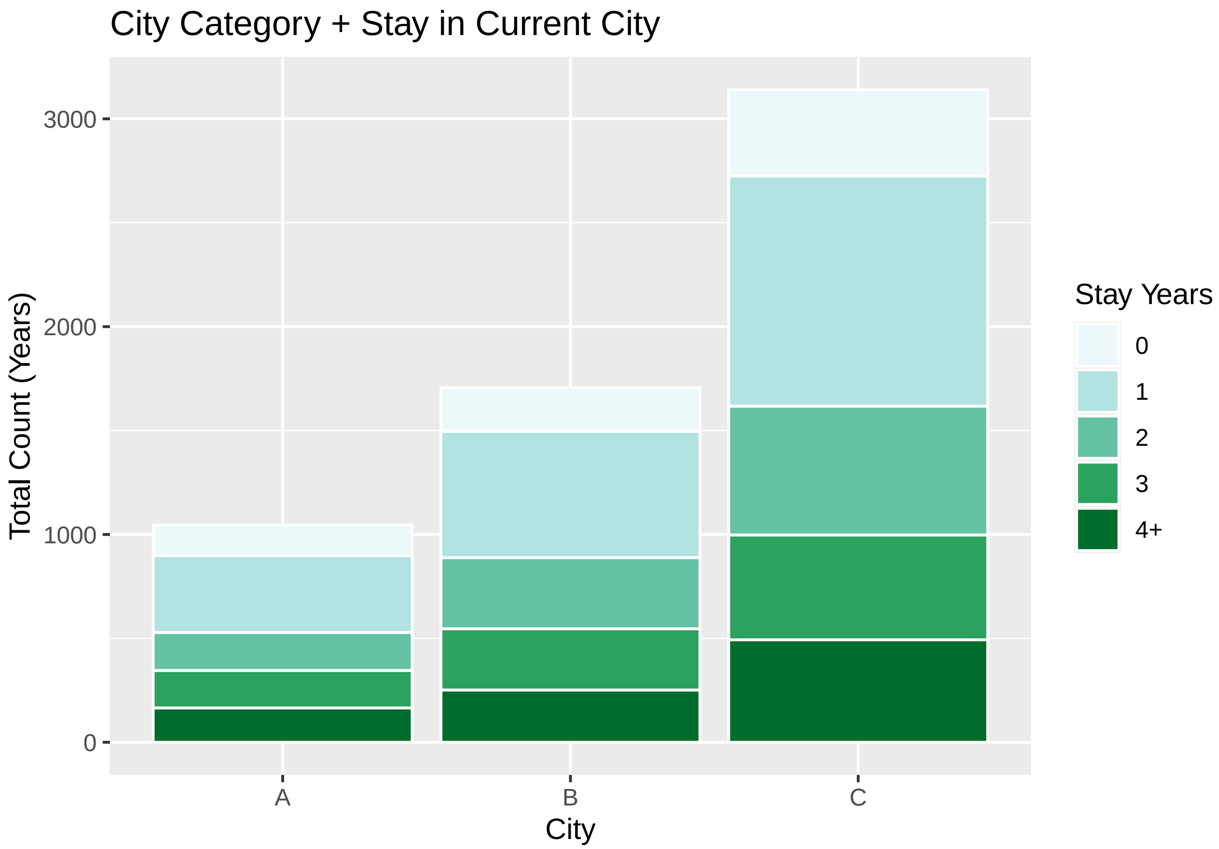


Certainly, a stacked bar chart can provide a clearer visualization of the distribution of customers based on their length of residency within their respective cities, categorized by City_Category. This chart will aid in uncovering any variations in residency patterns across different city categories. Create this stacked bar chart using the following code:

```
stay_cities = customers_stay %>%
                group_by(City_Category, Stay_In_Current_City_Years) %>%
                tally() %>%
                mutate(Percentage = (n/sum(n))*100)
head(stay_cities)
```

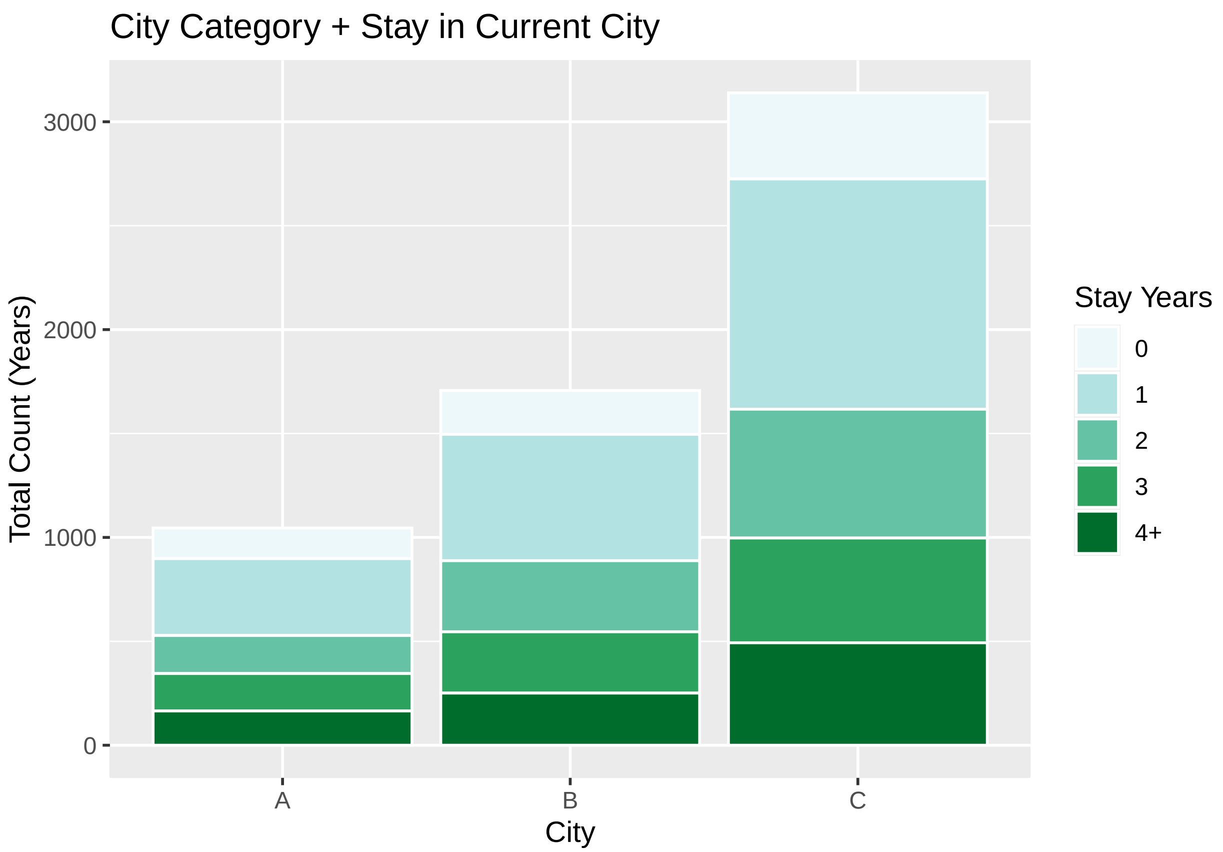
```
## # A tibble: 6 x 4
## # Groups:   City_Category [2]
##   City_Category Stay_In_Current_City_Years     n Percentage
##   <chr>         <chr>                      <int>      <dbl>
## 1 A             0                            147       14.1
## 2 A             1                            370       35.4
## 3 A             2                            183       17.5
## 4 A             3                            180       17.2
## 5 A             4+                           165       15.8
## 6 B             0                            211       12.4
```

```
ggplot(data = stay_cities, aes(x = City_Category, y = n, fill =
                                 Stay_In_Current_City_Years)) +
geom_bar(stat = "identity", color = 'white') + scale_fill_brewer(palette = 2) +
labs(title = "City Category + Stay in Current City",
y = "Total Count (Years)",
x = "City",
fill = "Stay Years")
```



This code snippet generates a stacked bar chart that segregates customers' length of residency based on different city categories. This visualization will facilitate the identification of any discernible trends or patterns in customer residency duration across distinct city segments.

```
ggplot(data = stay_cities, aes(x = City_Category, y = n,
                               fill = Stay_In_Current_City_Years)) +
    geom_bar(stat = "identity", color = 'white') +
    scale_fill_brewer(palette = 2) +
    labs(title = "City Category + Stay in Current City",
            y = "Total Count (Years)",
            x = "City",
            fill = "Stay Years")
```



Indeed, observation is accurate and insightful. The stacked bar chart effectively illustrates the distribution of the total customer base across various city categories, segmented by the duration of their residency. A consistent trend emerges across all City_Category segments, where the most prevalent duration of residence for customers is one year.

This commonality in the predominant residency duration across different city categories highlights the potential influence of this temporal aspect on shopping behaviors. The analysis offers valuable insights into how customer behaviors might correlate with the length of time they have resided in their current cities.

As a delve into another aspect of analysis, journey of exploration continues to unravel intricate patterns within the dataset.

3.1.2 Purchase

Certainly, let's delve into the analysis regarding store customers and their purchasing behavior.Initial step involves calculating the total purchase amount attributed to each User_ID. This computation can offer insights into the shopping habits and expenditure patterns of individual customers.To initiate this analysis, can utilize the following code:

```
customers_total_purchase_amount = dataset %>%
                                    group_by(User_ID) %>%
                                    summarise(Purchase_Amount = sum(Purchase))

head(customers_total_purchase_amount)
```

```
## # A tibble: 6 x 2
##   User_ID Purchase_Amount
##     <int>           <int>
## 1 1000001          333481
## 2 1000002          810353
## 3 1000003          341635
## 4 1000004          205987
## 5 1000005          821001
## 6 1000006          379450
```

Now that grouped purchases and grouped by User ID, sort and find top spenders.

```
customers_total_purchase_amount = arrange(customers_total_purchase_amount,
                                          desc((Purchase_Amount)))

head(customers_total_purchase_amount)
```

```
## # A tibble: 6 x 2
##   User_ID Purchase_Amount
##     <int>           <int>
## 1 1004277        10536783
## 2 1001680         8699232
## 3 1002909         7577505
## 4 1001941         6817493
## 5 1000424         6573609
## 6 1004448         6565878
```

Looks like User ID 1004277 is top spender. Lets use summary() to see other facets of total customer spending data.

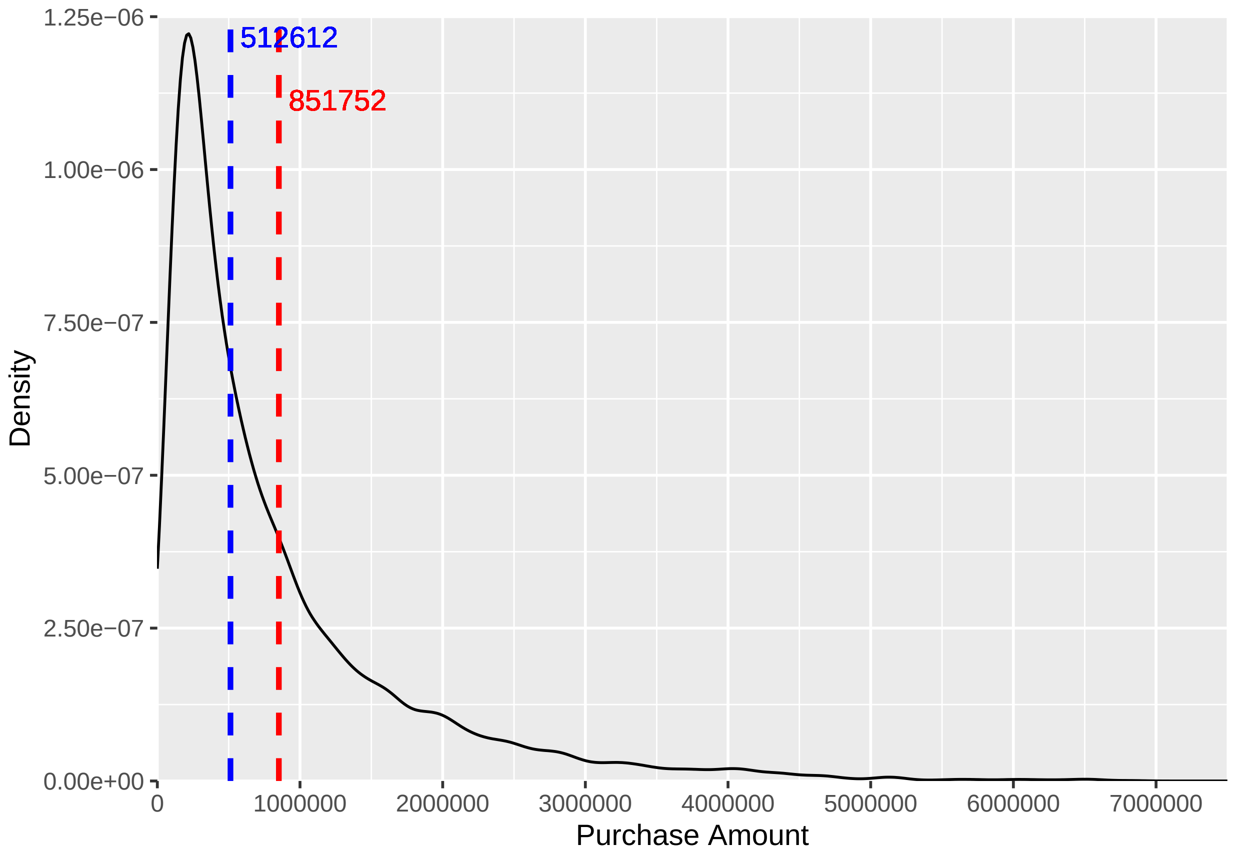
```
summary(customers_total_purchase_amount)
```

```
##     User_ID        Purchase_Amount
##  Min.   :1000001   Min.   :   44108
##  1st Qu.:1001518   1st Qu.:  234914
##  Median :1003026   Median :  512612
##  Mean   :1003025   Mean   :  851752
##  3rd Qu.:1004532   3rd Qu.: 1099005
##  Max.   :1006040   Max.   :10536783
```

Indeed, these summary statistics provide a comprehensive overview of the distribution of total purchase amounts among customers.It's noteworthy that the average, maximum, minimum, and median values highlight the spread and central tendencies of these purchase amounts.To further delve into the distribution of purchase amounts, a density plot is an excellent choice.

This visualization can help us understand the overall shape and skewness of the data, revealing where the highest concentration of similar purchase amounts lies within the customer base.Create a density plot using the following code:

```
ggplot(customers_total_purchase_amount, aes(Purchase_Amount)) +
  geom_density(adjust = 1) +
  geom_vline(aes(xintercept=median(Purchase_Amount)),
             color="blue", linetype="dashed", size=1) +
  geom_vline(aes(xintercept=mean(Purchase_Amount)),
             color="red", linetype="dashed", size=1) +
  geom_text(aes(x=mean(Purchase_Amount),
                label=round(mean(Purchase_Amount)), y=1.2e-06), color = 'red',
            angle=360, size=4, vjust=3, hjust=-.1) +
  geom_text(aes(x=median(Purchase_Amount), label=round(median(Purchase_Amount)), y=1.2e-06),
            color = 'blue', angle=360,size=4, vjust=0, hjust=-.1) +
  scale_x_continuous(name="Purchase Amount", limits=c(0, 7500000),
                     breaks = seq(0,7500000, by = 1000000), expand = c(0,0)) +
  scale_y_continuous(name="Density", limits=c(0, .00000125),
                     labels = scientific, expand = c(0,0))
```



3.1.3 Interpretation of the density plot

Interpretation of the density plot is accurate. The observed right (positive) skewness and the extended tail signify that a substantial number of purchase amounts lie higher than the mean. Additionally, the distribution doesn't align with a standard normal distribution.

The peak density around the 250,000 mark indicates that the highest concentration of purchases occurs within this range.This insight corroborates the notion that a significant proportion of customers are making purchases around this amount.

Understanding the distribution of purchase amounts is crucial for retailers in tailoring their marketing strategies, discounts, and promotionsto effectively engage with their customer base.

This detailed analysis of purchase amounts provides valuable insights into customer spending behavior, further enhancing understanding of the dataset's dynamics.As continue to explore various facets of the data, journey of analysis unfolds, potentially revealing more intricate patterns and trends.

3.1.4 Marital Status

Lets now examine the marital status of store customers.

```r
dataset_maritalStatus = dataset %>%
                            select(User_ID, Marital_Status) %>%
                            group_by(User_ID) %>%
                            distinct()

head(dataset_maritalStatus)
```

```
## # A tibble: 6 x 2
## # Groups:   User_ID [6]
##   User_ID Marital_Status
##     <int>          <int>
## 1 1000001              0
## 2 1000002              0
## 3 1000003              0
## 4 1000004              1
## 5 1000005              1
## 6 1000006              0
```

Note, need to quickly change Marital_Status from a numeric variable to a categorical type.

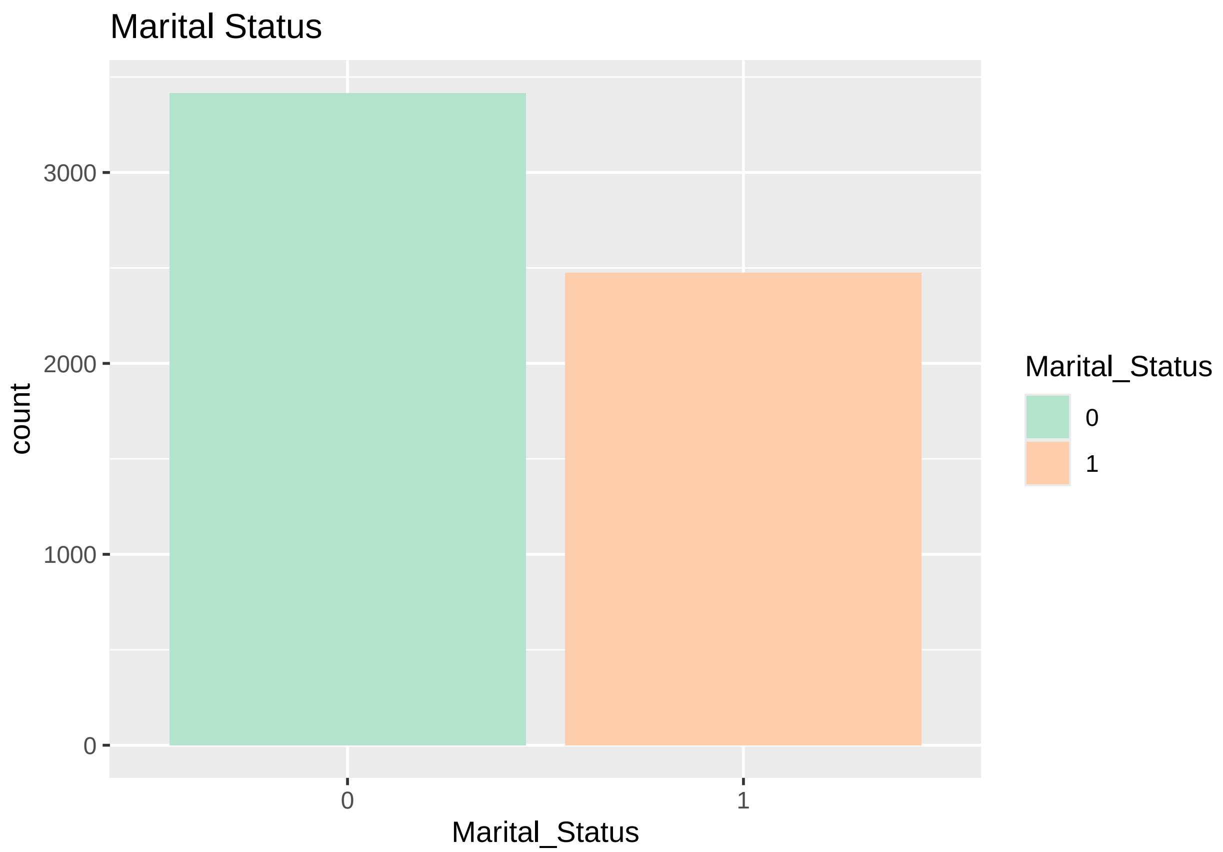
```r
dataset_maritalStatus$Marital_Status = as.character(dataset_maritalStatus$Marital_Status)
typeof(dataset_maritalStatus$Marital_Status)
```

```
## [1] "character"
```

Approach of assuming that 1 represents “married” and 0 represents “single” for the marital status variable is a reasonable approach given the absence of clear guidance in the dataset's variable descriptions.While it's always ideal to verify such information with the data provider, making educated assumptions based on context and conventional interpretations can help proceed with analysis.

This assumption aligns with common conventions and allows to continue exploring the relationships between marital status and other variables within the dataset.However, do keep inmind that assumptions like these should be documented and clearly communicated when presenting findings to ensure transparency in analysis.

```r
marital_vis = ggplot(data = dataset_maritalStatus) +
                    geom_bar(mapping = aes(x = Marital_Status, y = ..count..,
                                           fill = Marital_Status)) +
                    labs(title = 'Marital Status') +
                    scale_fill_brewer(palette = 'Pastel2')
print(marital_vis)
```



Indeed, observation aligns with the dataset's pattern, revealing that a significant portion of shoppers appear to be single or unmarried. This insight is valuable in understanding the demographic composition of the customer base.

Extending this analysis, exploring the distribution of Marital_Status within different City_Category segments can provide further insights into how marital status varies across different geographical areas. This investigation can shed light on potential regional patterns in shopping behaviors. Conduct this analysis by employing the following code:

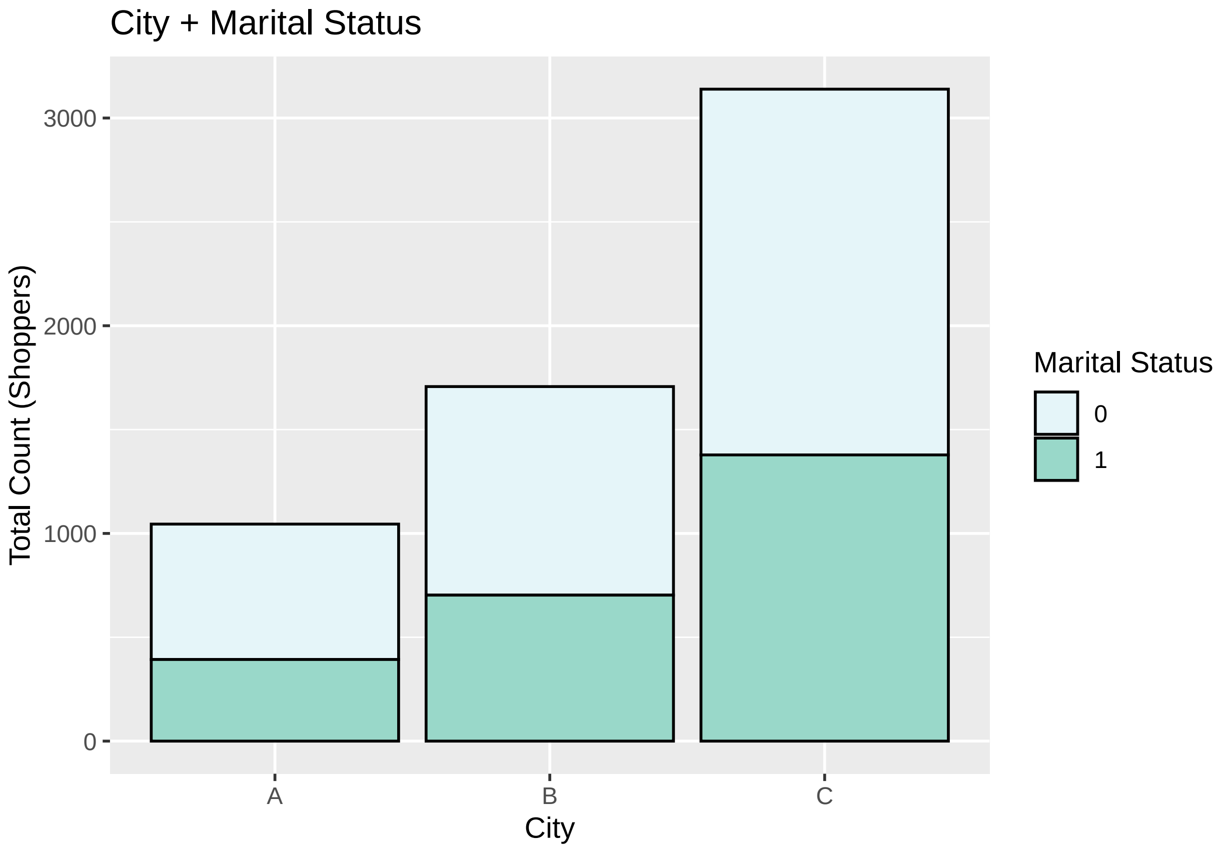
```
dataset_maritalStatus = dataset_maritalStatus %>%
                                full_join(customers_stay, by = 'User_ID')
head(dataset_maritalStatus)
```

```
## # A tibble: 6 x 4
## # Groups:   User_ID [6]
##   User_ID Marital_Status City_Category Stay_In_Current_City_Years
##     <int> <chr>          <chr>         <chr>
## 1 1000001 0              A             2
## 2 1000002 0              C             4+
## 3 1000003 0              A             3
## 4 1000004 1              B             2
## 5 1000005 1              A             1
## 6 1000006 0              A             1
```

```
maritalStatus_cities = dataset_maritalStatus %>%
                        group_by(City_Category, Marital_Status) %>%
                        tally()
head(maritalStatus_cities)
```

```
## # A tibble: 6 x 3
## # Groups:   City_Category [3]
##   City_Category Marital_Status     n
##   <chr>         <chr>          <int>
## 1 A             0                652
## 2 A             1                393
## 3 B             0               1004
## 4 B             1                703
## 5 C             0               1761
## 6 C             1               1378
```

```
ggplot(data = maritalStatus_cities, aes(x = City_Category, y = n,
                                        fill = Marital_Status)) +
    geom_bar(stat = "identity", color = 'black') +
    scale_fill_brewer(palette = 2) +
    labs(title = "City + Marital Status",
            y = "Total Count (Shoppers)",
            x = "City",
            fill = "Marital Status")
```



Observation about the distribution of single shoppers across different city categories is insightful and provides an initial understanding of how marital status may vary geographically.

Moving forward, can delve into the distribution of "Stay_in_Current_City" within each City_Category.This investigation aims to uncover patterns in the duration of customers' current city residence across different city segments. Conduct this analysis using the following code:

```
Users_Age = dataset %>%
                select(User_ID, Age) %>%
                distinct()
head(Users_Age)
```

```
##   User_ID   Age
## 1 1000001  0-17
## 2 1000002   55+
## 3 1000003 26-35
## 4 1000004 46-50
## 5 1000005 26-35
## 6 1000006 51-55
```

This code generates a table that compiles the count of customers based on the duration of their stay in their current city, segmented by different city categories. Visualizing this data will help understand how customer residence durations differ across various city segments, potentially unveiling trends and insights about customer behaviors and preferences.

```
dataset_maritalStatus = dataset_maritalStatus %>%
                            full_join(Users_Age, by = 'User_ID')
head(dataset_maritalStatus)
```

```
## # A tibble: 6 x 5
## # Groups:   User_ID [6]
##   User_ID Marital_Status City_Category Stay_In_Current_City_Years Age
##     <int> <chr>          <chr>         <chr>                      <chr>
## 1 1000001 0              A             2                          0-17
## 2 1000002 0              C             4+                         55+
## 3 1000003 0              A             3                          26-35
## 4 1000004 1              B             2                          46-50
## 5 1000005 1              A             1                          26-35
## 6 1000006 0              A             1                          51-55
```

```
City_A = dataset_maritalStatus %>%
            filter(City_Category == 'A')
City_B = dataset_maritalStatus %>%
            filter(City_Category == 'B')
City_C = dataset_maritalStatus %>%
            filter(City_Category == 'C')
head(City_A)
```

```
## # A tibble: 6 x 5
## # Groups:   User_ID [6]
##   User_ID Marital_Status City_Category Stay_In_Current_City_Years Age
##     <int> <chr>          <chr>         <chr>                      <chr>
## 1 1000001 0              A             2                          0-17
## 2 1000003 0              A             3                          26-35
## 3 1000005 1              A             1                          26-35
## 4 1000006 0              A             1                          51-55
## 5 1000015 0              A             1                          26-35
## 6 1000019 0              A             3                          0-17
```

```
head(City_B)
```

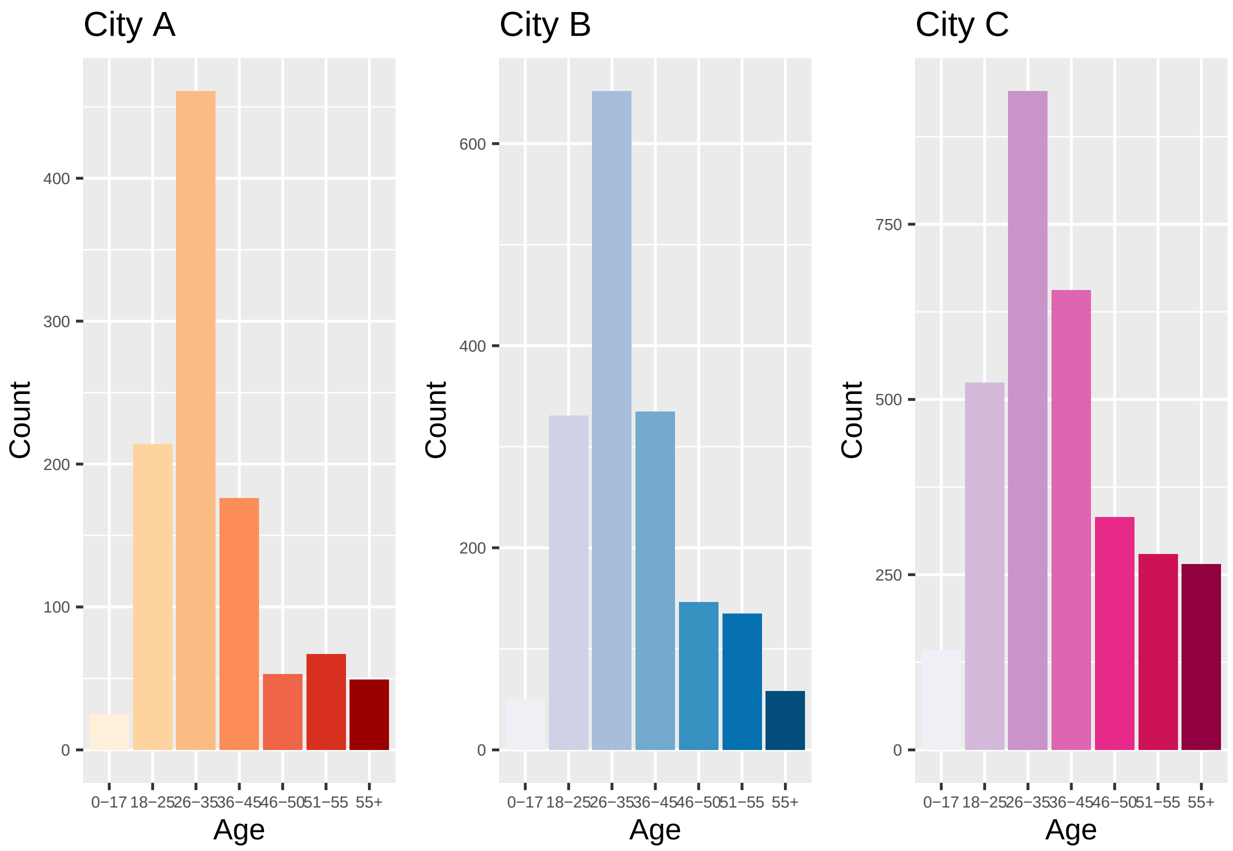
```
## # A tibble: 6 x 5
## # Groups:   User_ID [6]
##   User_ID Marital_Status City_Category Stay_In_Current_City_Years Age
##     <int> <chr>          <chr>         <chr>                      <chr>
## 1 1000004 1              B             2                          46-50
## 2 1000007 1              B             1                          36-45
## 3 1000010 1              B             4+                         36-45
## 4 1000018 0              B             3                          18-25
## 5 1000021 0              B             0                          18-25
## 6 1000023 1              B             3                          36-45
```

```
head(City_C)
```

```
## # A tibble: 6 x 5
## # Groups:   User_ID [6]
##   User_ID Marital_Status City_Category Stay_In_Current_City_Years Age
##     <int> <chr>          <chr>         <chr>                      <chr>
## 1 1000002 0              C             4+                         55+
## 2 1000008 1              C             4+                         26-35
## 3 1000009 0              C             0                          26-35
## 4 1000011 0              C             1                          26-35
## 5 1000012 0              C             2                          26-35
## 6 1000013 1              C             3                          46-50
```

```
City_A_stay_vis = ggplot(data = City_A, aes(x = Age, y = ..count..,
                                            fill = Age)) +
                              geom_bar(stat = 'count') +
                              scale_fill_brewer(palette = 8) +
                              theme(legend.position="none",
                                    axis.text = element_text(size = 6)) +
                              labs(title = 'City A', y = 'Count', x = 'Age',
                                   fill = 'Age')
City_B_stay_vis = ggplot(data = City_B, aes(x = Age, y = ..count..,
                                            fill = Age)) +
                              geom_bar(stat = 'count') +
                              scale_fill_brewer(palette = 9) +
                              theme(legend.position="none",
                                    axis.text = element_text(size = 6)) +
                              labs(title = 'City B', y = 'Count', x = 'Age',
                                   fill = 'Age')
City_C_stay_vis = ggplot(data = City_C, aes(x = Age, y = ..count..,
                                            fill = Age)) +
                              geom_bar(stat = 'count') +
                              scale_fill_brewer(palette = 11) +
                              theme(legend.position="none",
                                    axis.text = element_text(size = 6)) +
                              labs(title = 'City C', y = 'Count', x = 'Age',
                                   fill = 'Age')

grid.arrange(City_A_stay_vis, City_B_stay_vis, City_C_stay_vis, ncol = 3)
```



Observation regarding the distribution of shoppers over the age of 45 in City A compared to other cities is astute. Indeed, demographic factors such as age distribution can influence various aspects of customer behaviors, including marital status and residency durations.

The interplay between age distribution, marital status, and residency durations can lead to nuanced patterns in each city category. As a mentioned, these factors can collectively shape the resulting levels of marital status within individual cities. This holistic understanding is crucial for retailers and marketers to tailor their strategies and offerings to effectively engage with specific customer segments.

By piecing together various variables and their relationships, effectively unraveling the complexity of customer behaviors within different city categories. As a delve further into the dataset, gaining deeper insights into the dynamics of customer interactions with the store.Analysis showcases the power of data exploration in uncovering hidden trends and patterns.

3.1.5 Top Shoppers

Now, investigate who top shoppers were on Black Friday.

```
top_shoppers = dataset %>%
                count(User_ID, sort = TRUE)
```

```r
head(top_shoppers)
```

```
##   User_ID    n
## 1 1001680 1025
## 2 1004277  978
## 3 1001941  898
## 4 1001181  861
## 5 1000889  822
## 6 1003618  766
```

Absolutely, joining the dataset containing information about the top shoppers (User_ID 1001680 in this case) with the dataset that includes total purchase amounts can provide a comprehensive view of their shopping behavior and expenditure patterns.Perform this joining process using the following code:

```r
top_shoppers =  top_shoppers %>%
                    select(User_ID, n) %>%
                    left_join(customers_total_purchase_amount, Purchase_Amount,
                              by = 'User_ID')

head(top_shoppers)
```

```
##   User_ID    n Purchase_Amount
## 1 1001680 1025         8699232
## 2 1004277  978        10536783
## 3 1001941  898         6817493
## 4 1001181  861         6387899
## 5 1000889  822         5499812
## 6 1003618  766         5961987
```

In this code snippet, filter the dataset to include information about the top shopper (User_ID 1001680), selecting relevant columns such as User_ID, Gender, Age, City_Category, Stay_In_Current_City_Years, and Marital_Status. Then perform a left join with the dataset containing total purchase amounts by User_ID.

The resulting top_shopper_combined dataset will offer a consolidated view of this top shopper's demographic information alongside their total purchase amount, providing a comprehensive profile of their shopping behavior.Explore and visualize this combined dataset to gain deeper insights into the shopping patterns of this top shopper. This analysis can contribute to understanding the behaviors of high-frequency shoppers and potentially inform strategies for customer engagement and retention.

```r
top_shoppers = mutate(top_shoppers,
                  Average_Purchase_Amount = Purchase_Amount/n)

head(top_shoppers)
```

```
##   User_ID    n Purchase_Amount Average_Purchase_Amount
## 1 1001680 1025         8699232                8487.056
## 2 1004277  978        10536783               10773.807
## 3 1001941  898         6817493                7591.863
## 4 1001181  861         6387899                7419.163
## 5 1000889  822         5499812                6690.769
## 6 1003618  766         5961987                7783.273
```

Indeed, the joined table provides a comprehensive view of the shopping behavior of the top shoppers, highlighting both the User_ID with the highest number of total purchases and the User_ID with the highest total Purchase_Amount.Continuing analysis, can compute the average Purchase_Amount for each user using the following code:

```
top_shoppers_averagePurchase = top_shoppers %>%
                                   arrange(desc(Average_Purchase_Amount))

head(top_shoppers_averagePurchase)
```

```
##   User_ID  n Purchase_Amount Average_Purchase_Amount
## 1 1005069 16          308454                19278.38
## 2 1003902 93         1746284                18777.25
## 3 1005999 18          330227                18345.94
## 4 1001349 23          417743                18162.74
## 5 1000101 65         1138239                17511.37
## 6 1003461 20          350174                17508.70
```

Analysis highlights interesting patterns among the top shoppers. The contrast between User_ID 1005069 and User_ID 1003902 is particularly intriguing: while User_ID 1005069 has the highest average purchase amount, User_ID 1003902 boasts a significantly higher total purchase amount.These differences underscore the significance of considering both average and total purchase amounts when evaluating customer spending behaviors.

High average purchase amounts can indicate a willingness to spend more on individual transactions, while high total purchase amounts reflect a greater cumulative expenditure over multiple transactions.

These insights can inform marketing strategies, tailored offers, and customer engagement efforts.Understanding the spending dynamics of individual customers provides valuable guidance for optimizing retail operations and enhancing customer satisfaction.As an analysis continues, painting a detailed picture of the top shoppers' behaviors, ultimately contributing to a more comprehensive understanding of the dataset's dynamics.

3.1.6 Occupation

The last thing analyze is the occupation of customers in dataset.

```
customers_Occupation =  dataset %>%
                          select(User_ID, Occupation) %>%
                          group_by(User_ID) %>%
                          distinct() %>%
                          left_join(customers_total_purchase_amount, Occupation,
                                   by = 'User_ID')

head(customers_Occupation)
```

```
## # A tibble: 6 x 3
## # Groups:   User_ID [6]
##   User_ID Occupation Purchase_Amount
##     <int>      <int>           <int>
## 1 1000001         10          333481
## 2 1000002         16          810353
## 3 1000003         15          341635
## 4 1000004          7          205987
## 5 1000005         20          821001
## 6 1000006          9          379450
```

Now that dataset necessary, group together the total Purchase_Amount for each Occupation identifier. Then convert Occupation to a charater data type.

```
totalPurchases_Occupation = customers_Occupation %>%
                          group_by(Occupation) %>%
                          summarise(Purchase_Amount = sum(Purchase_Amount)) %>%
                          arrange(desc(Purchase_Amount))

totalPurchases_Occupation$Occupation =
  as.character(totalPurchases_Occupation$Occupation)
typeof(totalPurchases_Occupation$Occupation)
```

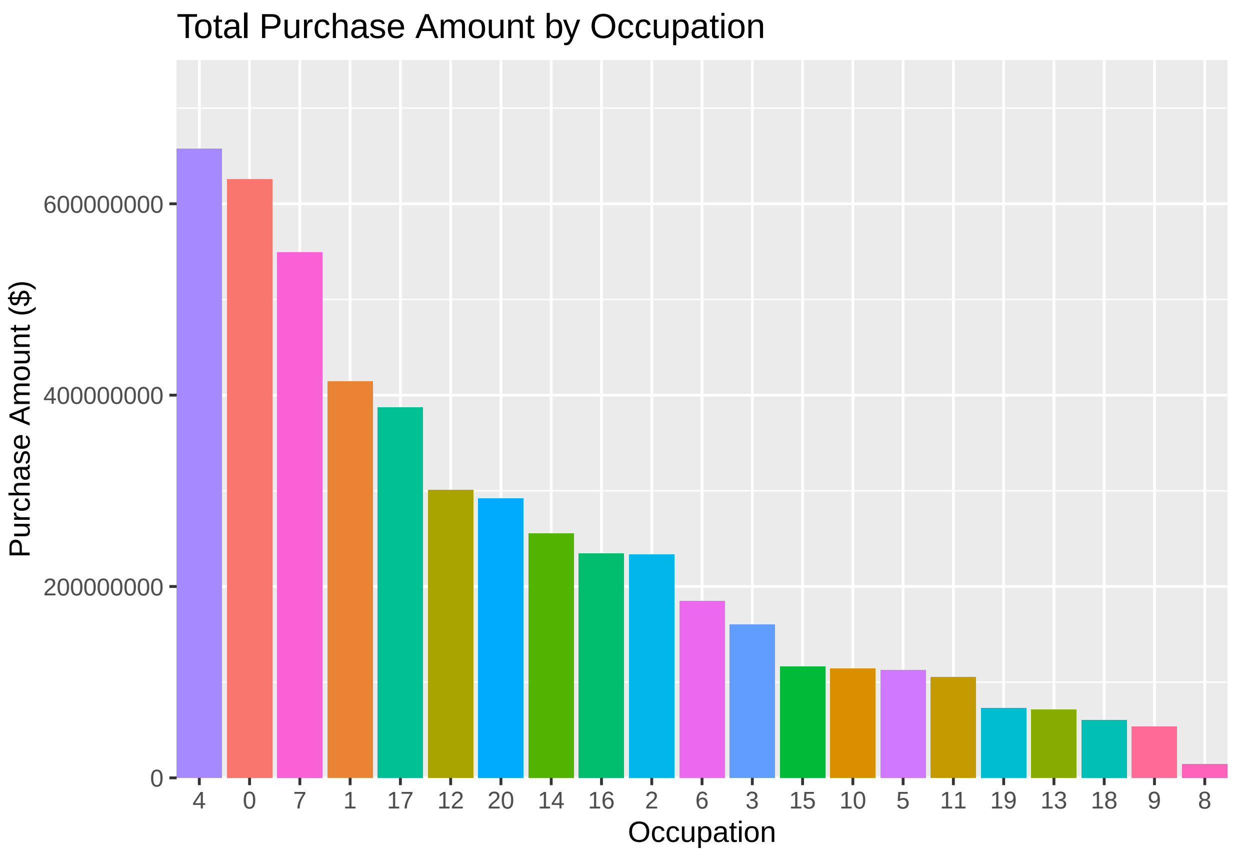
```
## [1] "character"
```

```
head(totalPurchases_Occupation)
```

```
## # A tibble: 6 x 2
##   Occupation Purchase_Amount
##   <chr>                <int>
## 1 4                657530393
## 2 0                625814811
## 3 7                549282744
## 4 1                414552829
## 5 17               387240355
## 6 12               300672105
```

Now, lets plot each occupation and their total Purchase_Amount

```
occupation = ggplot(data = totalPurchases_Occupation) +
                  geom_bar(mapping =
                             aes(x = reorder(Occupation, -Purchase_Amount),
                                 y = Purchase_Amount, fill = Occupation),
                           stat = 'identity') +
                  scale_x_discrete(name="Occupation",
                              breaks = seq(0,20, by = 1), expand = c(0,0)) +
                  scale_y_continuous(name="Purchase Amount ($)",
                                  expand = c(0,0), limits = c(0, 750000000)) +
                  labs(title = 'Total Purchase Amount by Occupation') +
                  theme(legend.position="none")
print(occupation)
```



Looks like customers labeled as Occupation 4 spent the most at store on Black Friday, with customers of Occupation 0 + 7 closely behind.Here, if a key was given, use that information to classify shoppers accordingly.

4 Chapter 4

4.1 Modeling Results and Model Performance

4.1.1 Apriori (Association Rule Learning)

Certainly, the explanation of Association Rule Learning provided is accurate and insightful.

Association Rule Learning, epitomized by the Apriori algorithm, holds immense potential for retailers and businesses seeking to uncover hidden patterns in customer purchasing behaviors.

By identifying associations between items frequently purchased together, businesses can strategically optimize product placement, cross-selling, and promotional campaigns to enhance sales and customer engagement [4].

Apriori algorithm, in particular, strives to unearth the most probable associations among items, enabling the generation of rules like "People who bought item A also bought item B." This algorithm's applications extend beyond retail to various domains, including recommendation systems, market basket analysis, and more [5][6].

As an embark on implementing the Apriori algorithm using the arules package, ensure that have imported the required libraries to facilitate the analysis.

If haven't imported the libraries yet, can use the following code:

```r
if (!require(arules)) {
  install.packages("arules", repos = "http://cran.us.r-project.org")
}
if (!require(arulesViz)) {
  install.packages("arulesViz", repos = "http://cran.us.r-project.org")
}
if (!require(tidyverse)) {
  install.packages("tidyverse", repos = "http://cran.us.r-project.org")
}


library(arules)
library(arulesViz)
library(tidyverse)
```

With these libraries in place, well-equipped to proceed with Association Rule Learning analysis.

This endeavor will yield valuable insights into purchase associations, aiding retailers in optimizing their strategies and enhancing customer experiences. If need assistance with the implementation or interpretation of the Apriori algorithm, to reach out for guidance.

The arules package was developed specifically to deal with Association Rule and Frequent Itemset mining.

In order to begin analysis, retrieve the necessary data from the original dataset and then apply the correct formatting.

```r
# Data Preprocessing
# Getting the dataset into the correct format
customers_products = dataset %>%
          select(User_ID, Product_ID) %>%
          # Selecting the columns we will need
          group_by(User_ID) %>%
          # Grouping by "User_ID"
          arrange(User_ID) %>%
          # Arranging by "User_ID"
          mutate(id = row_number()) %>%
          # Defining a key column for each "Product_ID" and its corresponding
          # "User_ID" (Must do this for spread() to work properly)
          spread(User_ID, Product_ID) %>%
          # Converting our dataset from tall to wide format, and grouping
          #"Product_IDs" to their corresponding "User_ID"
          t()
# Transposing the dataset from columns of "User_ID" to rows of "User_ID"

# Now we can remove the Id row we created earlier for spread() to work correctly.
customers_products = customers_products[-1,]
```

Explanation of the process to prepare the data for the Apriori algorithm is accurate.

The Apriori algorithm indeed requires a binary format, where products are represented as columns, and a value of 1 indicates that a customer purchased that product, while a value of 0 indicates no purchase.

This transformation from the original dataset to a sparse matrix format is a necessary step to facilitate the Apriori algorithm's analysis. Here's how can perform this transformation using the arules library:

1. Save customers_products table as a CSV file.
2. Use the read.transactions() function to read the CSV file and convert it into the required sparse matrix format.

Here's an example code snippet to achieve this:

```
write.csv(customers_products, file = 'customers_products.csv')

customersProducts = read.transactions('customers_products.csv', sep = ',', rm.duplicates = TRUE)
```

```
## distribution of transactions with duplicates:
## items
##   46  126  163  202  258  272  285  307  310  316  319  327  330  334  340  344
##    1    1    1    1    1    1    1    1    1    1    1    2    1    1    1    1
##  345  348  354  357  373  393  402  408  419  437  441  449  450  452  454  456
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1
##  459  465  466  467  475  476  477  481  487  491  495  498  507  523  524  526
##    1    1    2    2    1    1    1    1    2    2    3    2    1    1    2    1
##  527  528  530  531  532  533  535  537  538  539  540  545  546  548  549  553
##    2    1    1    2    1    1    1    1    2    3    1    1    1    3    1    1
##  554  555  556  558  563  566  567  570  572  574  575  577  578  580  583  584
##    2    1    1    2    1    3    1    3    2    4    1    2    3    2    1    1
##  586  588  589  590  591  592  593  594  595  597  598  601  602  604  607  608
##    2    3    5    1    1    1    1    1    3    2    1    1    1    1    2    1
##  610  612  613  614  615  616  617  618  619  620  623  625  632  633  634  635
##    1    2    1    2    1    1    2    1    3    2    1    1    6    1    5    2
##  638  640  641  642  643  644  645  646  647  648  653  654  657  658  659  661
##    2    2    1    2    1    2    3    1    4    1    1    3    2    1    1    2
##  662  663  664  665  666  667  668  669  670  671  672  674  676  677  678  679
##    1    3    1    1    2    1    2    4    2    1    3    1    1    2    3    3
##  681  682  683  685  686  687  688  689  690  691  692  694  695  697  698  699
##    2    2    4    4    5    2    2    1    1    1    1    2    1    1    1    1
##  700  702  703  704  705  706  707  708  709  710  712  713  714  715  716  717
##    2    1    3    1    1    2    2    3    2    2    4    5    2    2    1    1
##  718  719  720  721  722  723  724  725  726  727  728  729  730  732  733  734
##    2    3    3    5    2    1    1    2    5    2    1    3    3    2    1    5
##  735  736  737  738  739  740  741  742  743  744  745  747  748  749  750  751
##    6    2    4    6    3    4    1    6    7    4    6    1    1    5    5    3
##  752  753  754  755  756  757  758  759  760  761  763  764  765  766  767  768
##    2    3    4    6    6    2    6    2    1    5    3    4    2    3    2    5
##  769  770  771  772  773  774  775  776  777  778  779  780  781  782  783  784
##    2    2    3    1    3    4    2    2    3    3    2    4    7    3    4    5
##  785  786  787  788  789  790  791  792  793  794  795  796  797  798  799  800
##    5    3    3    6    5    5    2    3    5    3    8    5    5    9    3    4
##  801  802  803  804  805  806  807  808  809  810  811  812  813  814  815  816
##    4    7    4    3    4    5    7    5    4    5    3    2    6    6    3   11
##  817  818  819  820  821  822  823  824  825  826  827  828  829  830  831  832
##    5   10    6    6    4    7    7    2    5    4    7    5    5    4    5    4
##  833  834  835  836  837  838  839  840  841  842  843  844  845  846  847  848
##    3    5    4   11    5    5    4    9    7    6    4    7    9   11    4    6
##  849  850  851  852  853  854  855  856  857  858  859  860  861  862  863  864
##   10    6   10    7   12   16   11    8    7    4   12    9   11   11    9    6
##  865  866  867  868  869  870  871  872  873  874  875  876  877  878  879  880
##   11   10    7    6    5   12    6    7    8   11    9    9    8    7    5    4
##  881  882  883  884  885  886  887  888  889  890  891  892  893  894  895  896
##   15   13   12    8    4    6   12   15   13   10   11   13    6   21    7   14
##  897  898  899  900  901  902  903  904  905  906  907  908  909  910  911  912
##    9    7   11   18    5   14   10    9   19   15   10   17   18   23    8   19
##  913  914  915  916  917  918  919  920  921  922  923  924  925  926  927  928
##   15   12   18   21   17   12   11   13   13   12   20   20   16   13   15   17
##  929  930  931  932  933  934  935  936  937  938  939  940  941  942  943  944
##   27   22   20   28   18   14   20   20   20   14   22   30   23   23   21   20
##  945  946  947  948  949  950  951  952  953  954  955  956  957  958  959  960
##   25   19   30   31   30   24   27   25   40   30   31   16   29   30   32   48
##  961  962  963  964  965  966  967  968  969  970  971  972  973  974  975  976
##   27   27   24   30   26   35   43   30   51   49   40   41   36   32   36   38
##  977  978  979  980  981  982  983  984  985  986  987  988  989  990  991  992
##   43   41   42   37   49   44   51   57   55   40   53   56   63   39   58   50
##  993  994  995  996  997  998  999 1000 1001 1002 1003 1004 1005 1006 1007 1008
##   58   77   74   72   72   84   74   66   77   85   93   79   94  118  122  104
## 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019
##  121  113  120   78   77   55   37   20    7    5    1
```

```
# remove duplicates with rm.duplicates
```

Before implement the Apriori algorithm to problem, lets take a look at newly created sparse matrix.

```
summary(customersProducts)
```

```
## transactions as itemMatrix in sparse format with
##  5892 rows (elements/itemsets/transactions) and
##  10539 columns (items) and a density of 0.008768598
##
```

```
## most frequent items:
## P00265242 P00110742 P00025442 P00112142 P00057642   (Other)
##      1858      1591      1586      1539      1430    536489
##
## element (itemset/transaction) length distribution:
## sizes
##    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20   21
##    1    5    7   20   37   55   77   78  120  113  121  104  122  118   94   79
##   22   23   24   25   26   27   28   29   30   31   32   33   34   35   36   37
##   93   85   77   66   74   84   72   72   74   77   58   50   58   39   63   56
##   38   39   40   41   42   43   44   45   46   47   48   49   50   51   52   53
##   53   40   55   57   51   44   49   37   42   41   43   38   36   32   36   41
##   54   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69
##   40   49   51   30   43   35   26   30   24   27   27   48   32   30   29   16
##   70   71   72   73   74   75   76   77   78   79   80   81   82   83   84   85
##   31   30   40   25   27   24   30   31   30   19   25   20   21   23   23   30
##   86   87   88   89   90   91   92   93   94   95   96   97   98   99  100  101
##   22   14   20   20   20   14   18   28   20   22   27   17   15   13   16   20
##  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117
##   20   12   13   13   11   12   17   21   18   12   15   19    8   23   18   17
##  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133
##   10   15   19    9   10   14    5   18   11    7    9   14    7   21    6   13
##  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149
##   11   10   13   15   12    6    4    8   12   13   15    4    5    7    8    9
##  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165
##    9   11    8    7    6   12    5    6    7   10   11    6    9   11   11    9
##  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181
##   12    4    7    8   11   16   12    7   10    6   10    6    4   11    9    7
##  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197
##    4    6    7    9    4    5    5   11    4    5    3    4    5    4    5    5
##  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213
##    7    4    5    2    7    7    4    6    6   10    5   11    3    6    6    2
##  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229
##    3    5    4    5    7    5    4    3    4    7    4    4    3    9    5    5
##  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245
##    8    3    5    3    2    5    5    6    3    3    5    5    4    3    7    4
##  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261
##    2    3    3    2    2    4    3    1    3    2    2    5    2    3    2    4
##  262  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278
##    3    5    1    2    6    2    6    6    4    3    2    3    5    5    1    1
##  280  281  282  283  284  285  286  287  288  289  290  291  292  293  295  296
##    6    4    7    6    1    4    3    6    4    2    6    5    1    2    3    3
##  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312
##    1    2    5    2    1    1    2    5    3    3    2    1    1    2    2    5
##  313  315  316  317  318  319  320  321  322  323  325  326  327  328  330  331
##    4    2    2    3    2    2    1    1    3    1    2    1    1    1    1    2
##  333  334  335  336  337  338  339  340  342  343  344  346  347  348  349  351
##    1    1    1    1    2    2    5    4    4    2    2    3    3    2    1    1
##  353  354  355  356  357  358  359  360  361  362  363  364  366  367  368  371
##    3    1    2    4    2    1    2    1    1    3    1    2    1    1    2    3
##  372  377  378  379  380  381  382  383  384  385  387  390  391  392  393  400
##    1    1    4    1    3    2    1    2    1    2    2    2    5    1    6    1
##  402  405  406  407  408  409  410  411  412  413  415  417  418  421  423  424
##    1    2    3    1    2    1    1    2    1    2    1    1    2    1    1    1
##  427  428  430  431  432  433  434  435  436  437  439  441  442  445  447  448
##    1    2    3    1    1    1    1    1    5    3    2    1    1    2    3    2
##  450  451  453  455  458  459  462  467  469  470  471  472  476  477  479  480
##    1    4    2    3    1    3    1    2    1    1    2    1    1    3    1    1
##  485  486  487  488  490  492  493  494  495  497  498  499  501  502  518  527
##    1    3    2    1    1    1    1    2    1    1    2    1    2    1    1    2
##  530  534  538  544  548  549  550  558  559  560  566  569  571  573  575  576
##    3    2    2    1    1    1    1    2    2    1    1    1    1    1    1    1
##  584  588  606  617  623  632  652  668  671  677  680  681  685  691  695  698
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    2
##  706  709  715  718  740  753  767  823  862  899  979 1025 1026
##    1    1    1    1    1    1    1    1    1    1    1    1    1
## 
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    6.00   26.00   54.00   92.41  115.00 1026.00 
## 
## includes extended item information - examples:
##    labels
## 1 1000001
## 2 1000002
## 3 1000003
```

Analysis and interpretation of the sparse matrix characteristics are accurate. The summary of the sparse matrix provides essential insights into the data's density, showcasing the proportion of non-zero (1) values compared to zero (0) values.

This density value, which is approximately 0.009, indicates that around 0.9% of the entries in the sparse matrix are non-zero, implying that 99.1% of the matrix entries are zero.Additionally, the frequent items identified in the summary of the sparse matrix align with the insights uncovered during Exploratory Data Analysis.

This congruence underscores the consistency and reliability of findings across different stages of analysis. Extract the most frequent items from the sparse matrix using the following code:

```
summary(customersProducts)
```

```
## transactions as itemMatrix in sparse format with
##  5892 rows (elements/itemsets/transactions) and
##  10539 columns (items) and a density of 0.008768598 
## 
## most frequent items:
## P00265242 P00110742 P00025442 P00112142 P00057642   (Other) 
##      1858      1591      1586      1539      1430    536489 
## 
## element (itemset/transaction) length distribution:
## sizes
##    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20   21 
##    1    5    7   20   37   55   77   78  120  113  121  104  122  118   94   79 
##   22   23   24   25   26   27   28   29   30   31   32   33   34   35   36   37 
##   93   85   77   66   74   84   72   72   74   77   58   50   58   39   63   56 
##   38   39   40   41   42   43   44   45   46   47   48   49   50   51   52   53 
##   53   40   55   57   51   44   49   37   42   41   43   38   36   32   36   41 
##   54   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69 
##   40   49   51   30   43   35   26   30   24   27   27   48   32   30   29   16 
##   70   71   72   73   74   75   76   77   78   79   80   81   82   83   84   85
##   31   30   40   25   27   24   30   31   30   19   25   20   21   23   23   30
##   86   87   88   89   90   91   92   93   94   95   96   97   98   99  100  101
##   22   14   20   20   20   14   18   28   20   22   27   17   15   13   16   20
##  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117
##   20   12   13   13   11   12   17   21   18   12   15   19    8   23   18   17
##  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133
##   10   15   19    9   10   14    5   18   11    7    9   14    7   21    6   13
##  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149
##   11   10   13   15   12    6    4    8   12   13   15    4    5    7    8    9
##  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165
##    9   11    8    7    6   12    5    6    7   10   11    6    9   11   11    9
##  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181
##   12    4    7    8   11   16   12    7   10    6   10    6    4   11    9    7
##  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197
##    4    6    7    9    4    5    5   11    4    5    3    4    5    4    5    5
##  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213
##    7    4    5    2    7    7    4    6    6   10    5   11    3    6    6    2
##  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229
##    3    5    4    5    7    5    4    3    4    7    4    4    3    9    5    5
##  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245
##    8    3    5    3    2    5    5    6    3    3    5    5    4    3    7    4
##  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261
##    2    3    3    2    2    4    3    1    3    2    2    5    2    3    2    4
##  262  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278
##    3    5    1    2    6    2    6    6    4    3    2    3    5    5    1    1
##  280  281  282  283  284  285  286  287  288  289  290  291  292  293  295  296
##    6    4    7    6    1    4    3    6    4    2    6    5    1    2    3    3
##  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312
##    1    2    5    2    1    1    2    5    3    3    2    1    1    2    2    5
##  313  315  316  317  318  319  320  321  322  323  325  326  327  328  330  331
##    4    2    2    3    2    2    1    1    3    1    2    1    1    1    1    2
##  333  334  335  336  337  338  339  340  342  343  344  346  347  348  349  351
##    1    1    1    1    2    2    5    4    4    2    2    3    3    2    1    1
##  353  354  355  356  357  358  359  360  361  362  363  364  366  367  368  371
##    3    1    2    4    2    1    2    1    1    3    1    2    1    1    2    3
##  372  377  378  379  380  381  382  383  384  385  387  390  391  392  393  400
##    1    1    4    1    3    2    1    2    1    2    2    2    5    1    6    1
##  402  405  406  407  408  409  410  411  412  413  415  417  418  421  423  424
##    1    2    3    1    2    1    1    2    1    2    1    1    2    1    1    1
##  427  428  430  431  432  433  434  435  436  437  439  441  442  445  447  448
##    1    2    3    1    1    1    1    1    5    3    2    1    1    2    3    2
##  450  451  453  455  458  459  462  467  469  470  471  472  476  477  479  480
##    1    4    2    3    1    3    1    2    1    1    2    1    1    3    1    1
##  485  486  487  488  490  492  493  494  495  497  498  499  501  502  518  527
##    1    3    2    1    1    1    1    2    1    1    2    1    2    1    1    2
##  530  534  538  544  548  549  550  558  559  560  566  569  571  573  575  576
##    3    2    2    1    1    1    1    2    2    1    1    1    1    1    1    1
##  584  588  606  617  623  632  652  668  671  677  680  681  685  691  695  698
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    2
##  706  709  715  718  740  753  767  823  862  899  979 1025 1026
##    1    1    1    1    1    1    1    1    1    1    1    1    1
## 
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    6.00   26.00   54.00   92.41  115.00 1026.00
##
## includes extended item information - examples:
##    labels
## 1 1000001
## 2 1000002
## 3 1000003
```

The itemFrequency() function calculates the frequency of each item in the transactions data, and head() is used to display the top 10 most frequent items. Comparing these results to earlier analyses helps validate the accuracy of insights.

By cross-referencing and confirming findings through multiple analyses, ensuring the robustness of conclusions and contributing to a more comprehensive understanding of customer purchase behaviors.

To continue analysis, and if any further questions or need assistance with interpreting the results of the Apriori algorithm, don't hesitate to ask. 1. P00265242 = 1858 2. P00110742 = 1591 3. P00025442 = 1586 4. P00112142 = 1539 5. P00057642 = 1430 6. (Other) = 536489

Now lets compare it to what discovered earlier. "Looks like top 5 best sellers are (by product ID)"

1. P00265242 = 1858
2. P00110742 = 1591
3. P00025442 = 1586
4. P00112142 = 1539
5. P00057642 = 1430

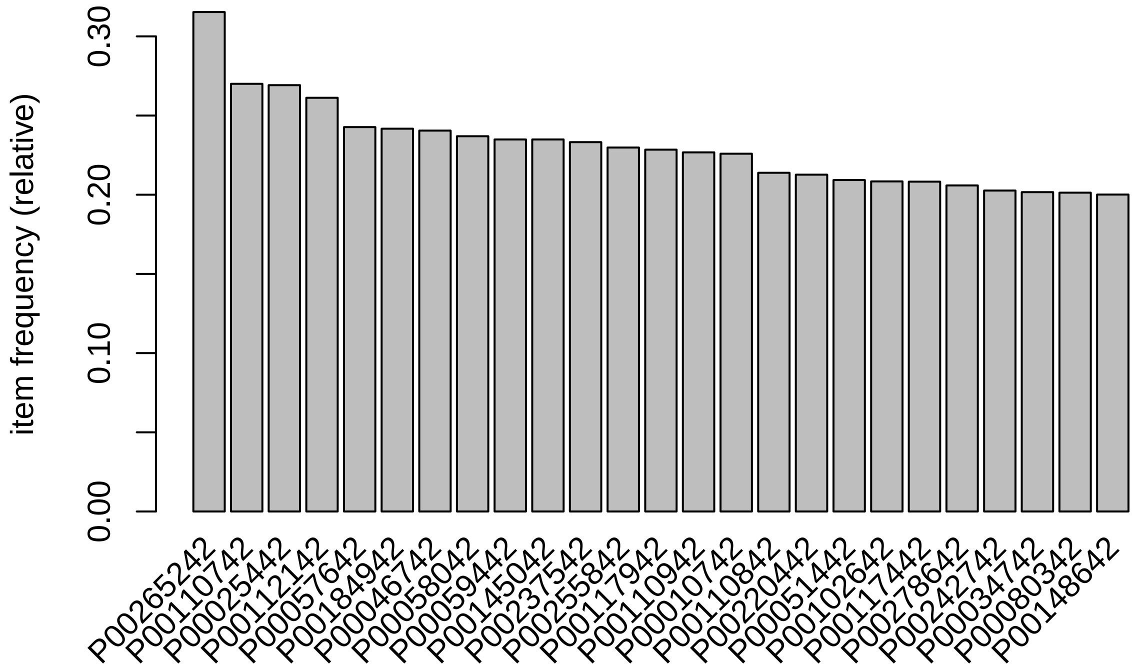
```
summary(customersProducts)
```

```
## transactions as itemMatrix in sparse format with
##  5892 rows (elements/itemsets/transactions) and
##  10539 columns (items) and a density of 0.008768598
##
## most frequent items:
## P00265242 P00110742 P00025442 P00112142 P00057642   (Other)
##      1858      1591      1586      1539      1430    536489
##
## element (itemset/transaction) length distribution:
## sizes
##    6    7    8    9   10   11   12   13   14   15   16   17   18   19   20   21
##    1    5    7   20   37   55   77   78  120  113  121  104  122  118   94   79
##   22   23   24   25   26   27   28   29   30   31   32   33   34   35   36   37
##   93   85   77   66   74   84   72   72   74   77   58   50   58   39   63   56
##   38   39   40   41   42   43   44   45   46   47   48   49   50   51   52   53
##   53   40   55   57   51   44   49   37   42   41   43   38   36   32   36   41
##   54   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69
##   40   49   51   30   43   35   26   30   24   27   27   48   32   30   29   16
##   70   71   72   73   74   75   76   77   78   79   80   81   82   83   84   85
##   31   30   40   25   27   24   30   31   30   19   25   20   21   23   23   30
##   86   87   88   89   90   91   92   93   94   95   96   97   98   99  100  101
##   22   14   20   20   20   14   18   28   20   22   27   17   15   13   16   20
##  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117
##   20   12   13   13   11   12   17   21   18   12   15   19    8   23   18   17
##  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133
##   10   15   19    9   10   14    5   18   11    7    9   14    7   21    6   13
##  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149
##   11   10   13   15   12    6    4    8   12   13   15    4    5    7    8    9
##  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165
##    9   11    8    7    6   12    5    6    7   10   11    6    9   11   11    9
##  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181
##   12    4    7    8   11   16   12    7   10    6   10    6    4   11    9    7
##  182  183  184  185  186  187  188  189  190  191  192  193  194  195  196  197
##    4    6    7    9    4    5    5   11    4    5    3    4    5    4    5    5
##  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213
##    7    4    5    2    7    7    4    6    6   10    5   11    3    6    6    2
##  214  215  216  217  218  219  220  221  222  223  224  225  226  227  228  229
##    3    5    4    5    7    5    4    3    4    7    4    4    3    9    5    5
##  230  231  232  233  234  235  236  237  238  239  240  241  242  243  244  245
##    8    3    5    3    2    5    5    6    3    3    5    5    4    3    7    4
##  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261
##    2    3    3    2    2    4    3    1    3    2    2    5    2    3    2    4
##  262  264  265  266  267  268  269  270  271  272  273  274  275  276  277  278
##    3    5    1    2    6    2    6    6    4    3    2    3    5    5    1    1
##  280  281  282  283  284  285  286  287  288  289  290  291  292  293  295  296
##    6    4    7    6    1    4    3    6    4    2    6    5    1    2    3    3
##  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312
##    1    2    5    2    1    1    2    5    3    3    2    1    1    2    2    5
##  313  315  316  317  318  319  320  321  322  323  325  326  327  328  330  331
##    4    2    2    3    2    2    1    1    3    1    2    1    1    1    1    2
##  333  334  335  336  337  338  339  340  342  343  344  346  347  348  349  351
##    1    1    1    1    2    2    5    4    4    2    2    3    3    2    1    1
##  353  354  355  356  357  358  359  360  361  362  363  364  366  367  368  371
##    3    1    2    4    2    1    2    1    1    3    1    2    1    1    2    3
##  372  377  378  379  380  381  382  383  384  385  387  390  391  392  393  400
##    1    1    4    1    3    2    1    2    1    2    2    2    5    1    6    1
##  402  405  406  407  408  409  410  411  412  413  415  417  418  421  423  424
##    1    2    3    1    2    1    1    2    1    2    1    1    2    1    1    1
##  427  428  430  431  432  433  434  435  436  437  439  441  442  445  447  448
##    1    2    3    1    1    1    1    1    5    3    2    1    1    2    3    2
##  450  451  453  455  458  459  462  467  469  470  471  472  476  477  479  480
##    1    4    2    3    1    3    1    2    1    1    2    1    1    3    1    1
##  485  486  487  488  490  492  493  494  495  497  498  499  501  502  518  527
##    1    3    2    1    1    1    1    2    1    1    2    1    2    1    1    2
##  530  534  538  544  548  549  550  558  559  560  566  569  571  573  575  576
##    3    2    2    1    1    1    1    2    2    1    1    1    1    1    1    1
##  584  588  606  617  623  632  652  668  671  677  680  681  685  691  695  698
##    1    1    1    1    1    1    1    1    1    1    1    1    1    1    1    2
##  706  709  715  718  740  753  767  823  862  899  979 1025 1026
##    1    1    1    1    1    1    1    1    1    1    1    1    1
## 
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    6.00   26.00   54.00   92.41  115.00 1026.00 
## 
## includes extended item information - examples:
##    labels
## 1 1000001
## 2 1000002
## 3 1000003
```

Interpretation of the “element length distribution” is accurate. The mean value of 92.41 indicates the average number of items in a customer’s basket. However, considering the presence of customers who purchased a significantly larger number of items, the median value of 54.00 items can provide a more representative measure of the central tendency of the distribution.

Using the median value to assess the typical number of items per customer is a prudent approach,as it is less influenced by outlier values and provides a more balanced representation of the dataset. To create an item frequency plot using the arules package, use the following code:

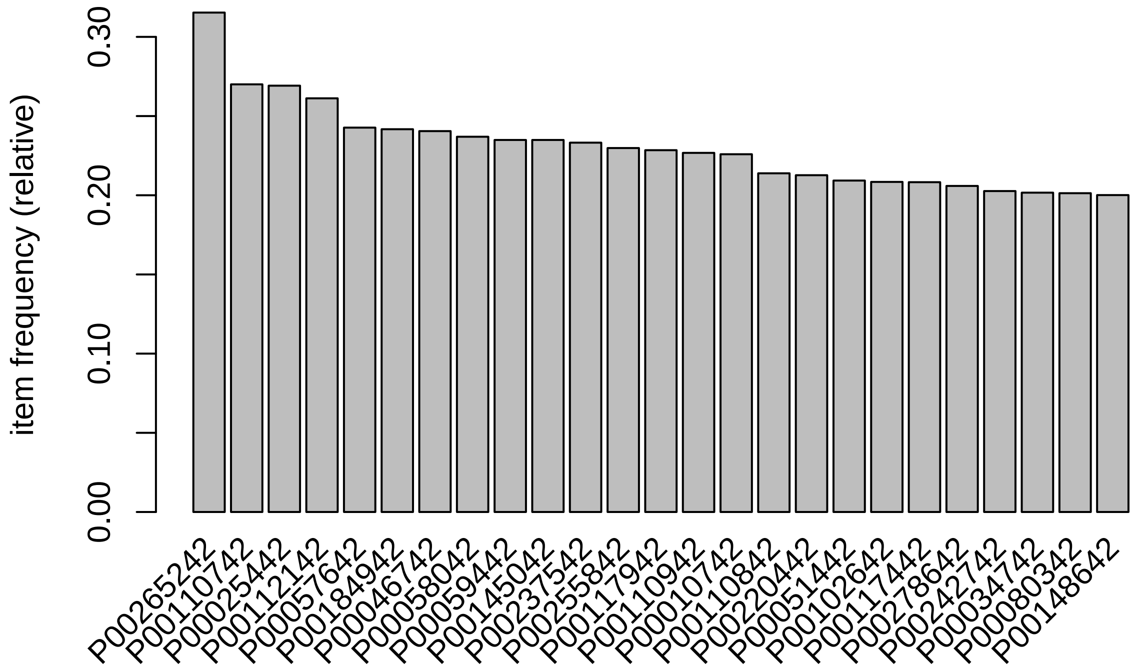
```r
itemFrequencyPlot(customersProducts, topN = 25)     # topN is limiting to the top 50 products
```



In this code, itemFrequencyPlot() generates a bar plot that displays the frequency of each item’s occurrence in the transactions data. The support parameter specifies the minimum level of support an item should have to be included in the plot, and the col parameter sets the color of the bars in the plot.

This plot will visually illustrate the frequency distribution of different items in customer baskets, providing insights into the most commonly purchased products. As continue with analysis, delving deeper into the transactional patterns of customers, gaining a richer understanding of their shopping behaviors.If further questions or if ready to explore the results of the Apriori algorithm.

```
itemFrequencyPlot(customersProducts, topN = 25)     # topN is limiting to the top 50 products
```



Explanation of setting the parameters for the Apriori algorithm is clear and accurate. The choice of support and confidence values plays a critical role in determining the rules generated by the algorithm.

These values guide the algorithm to identify significant associations between items based on the frequency of occurrence and the strength of the relationship.Setting the support value involves establishing a minimum threshold for the number of transactions that an item must be present in. This threshold helps filter out less frequent items that may not contribute significantly to meaningful rules.

In this example, aim to select products that were purchased by at least 50 different customers, and calculate the support value as a ratio of this threshold to the total number of transactions.Similarly, the confidence value serves as a threshold for the strength of the rule's prediction.

A higher confidence value ensures that the rules generated are more likely to be accurate and relevant.Starting with a default confidence value of 0.80 and then adjusting it based on domain knowledge and the desired outcome is a pragmatic approach. Step-by-step approach to selecting these parameters reflects a thoughtful process of refining the algorithm's behavior to yield valuable insights that align with analysis goals.

As a move forward with implementing the Apriori algorithm using these parameters, paving the way to uncovering significant associations among purchased items.This analysis will aid in optimizing product placement, recommendations, and promotional strategies for the retail store.

```
rules = apriori(data = customersProducts,
                parameter = list(support = 0.008, confidence = 0.80,
                                 maxtime = 0))
```

```
## Apriori
```

```
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.8    0.1    1 none FALSE            TRUE       0   0.008      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: 47
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[10539 item(s), 5892 transaction(s)] done [0.11s].
## sorting and recoding items ... [2099 item(s)] done [0.01s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [14.69s].
## writing ... [7 rule(s)] done [0.28s].
## creating S4 object  ... done [0.22s].
```

```
# maxtime = 0 will allow our algorithim to run until completion with
# no time limit
```

It looks like apriori has created 7 rules in accordance to specified parameters. "writing … [7 rule(s)] done [0.48s]." Now, lets examine results to get a better idea of how algoritm worked.

```
inspect(sort(rules, by = 'lift'))
```

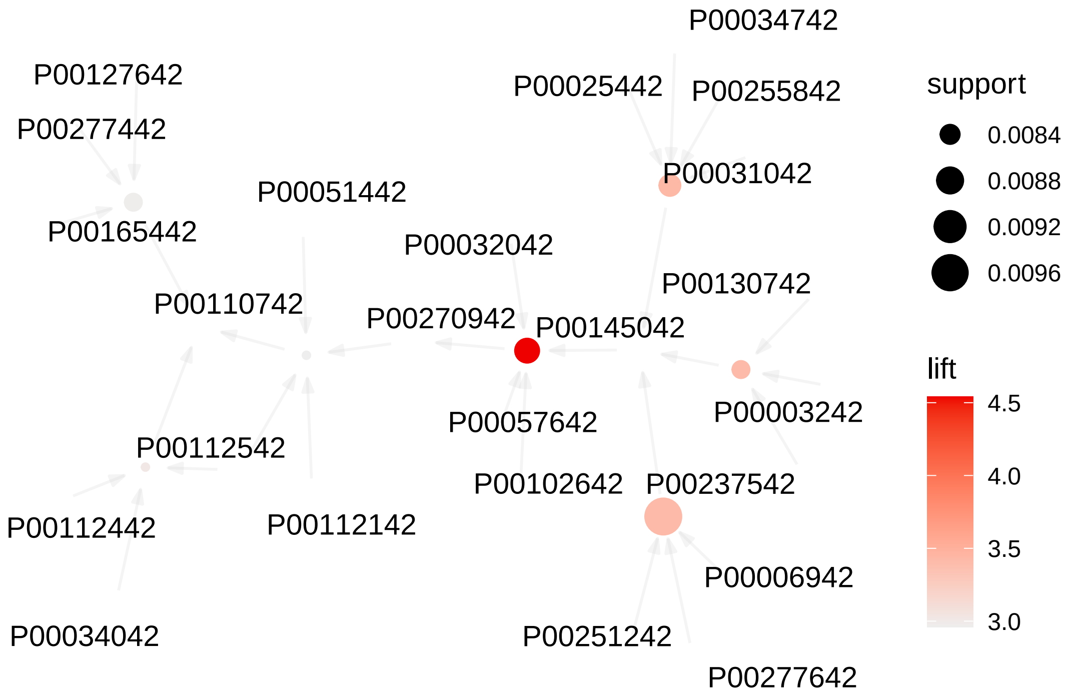
```
##     lhs             rhs             support confidence    coverage     lift count
## [1] {P00032042,
##      P00057642,
##      P00102642,
##      P00145042}  => {P00270942} 0.008655804  0.8793103 0.009843856 4.540663    51
## [2] {P00025442,
##      P00031042,
##      P00034742,
##      P00255842}  => {P00145042} 0.008486083  0.8064516 0.010522743 3.433246    50
## [3] {P00003242,
##      P00130742,
##      P00237542}  => {P00145042} 0.008316361  0.8032787 0.010353021 3.419738    49
## [4] {P00006942,
##      P00251242,
##      P00277642}  => {P00145042} 0.009674134  0.8028169 0.012050238 3.417773    57
## [5] {P00034042,
##      P00112442,
##      P00112542}  => {P00110742} 0.008146640  0.8135593 0.010013578 3.012880    48
## [6] {P00127642,
##      P00165442,
##      P00277442}  => {P00110742} 0.008316361  0.8032787 0.010353021 2.974807    49
## [7] {P00051442,
##      P00112142,
##      P00112542,
##      P00270942}  => {P00110742} 0.008146640  0.8000000 0.010183299 2.962665    48
```

Here see the association rules created by apriori algorithm. Let's take a look at rule number 1.

Description of the output values of the Apriori algorithm is accurate and well-organized.

Provided a clear explanation of each value and its significance in the context of the association rules generated by the algorithm. Here's how visualize these rules using the arulesViz package:

```
plot(rules, method = 'graph')
```



The plot() function with the method parameter set to "scatterplot" generates a scatterplot that displays the relationship between support, confidence, and lift for each rule.

This visualization can help quickly grasp the distribution of rules and their characteristics, aiding in the identification of meaningful and impactful associations.

Continuing analysis by visualizing the generated association rules using the arulesViz package adds another layer of understanding to insights. By visually representing these rules, providing a more accessible and intuitive view of the complex relationships within the dataset.

To proceed with the visualization step, and if have any further questions or if ready to interpret the results of the association rules, I'm here to assist!

```
rules = apriori(data = customersProducts,
                parameter = list(support = 0.008, confidence = 0.75, maxtime = 0))
```

```
## Apriori
##
## Parameter specification:
```

```
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.75    0.1    1 none FALSE            TRUE       0   0.008      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: 47
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[10539 item(s), 5892 transaction(s)] done [0.12s].
## sorting and recoding items ... [2099 item(s)] done [0.01s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [14.05s].
## writing ... [171 rule(s)] done [0.27s].
## creating S4 object  ... done [0.20s].
```

Now that decreased the minimum confidence value to 75%, a total of 171 rules. writing … [171 rule(s)] done [0.50s]. This is much higher number of rules compared to previous rule list which only contained 7. This should now give us more interesting rules to examine

```r
inspect(head(sort(rules, by = 'lift'))) # limiting to the top 6 rules
```

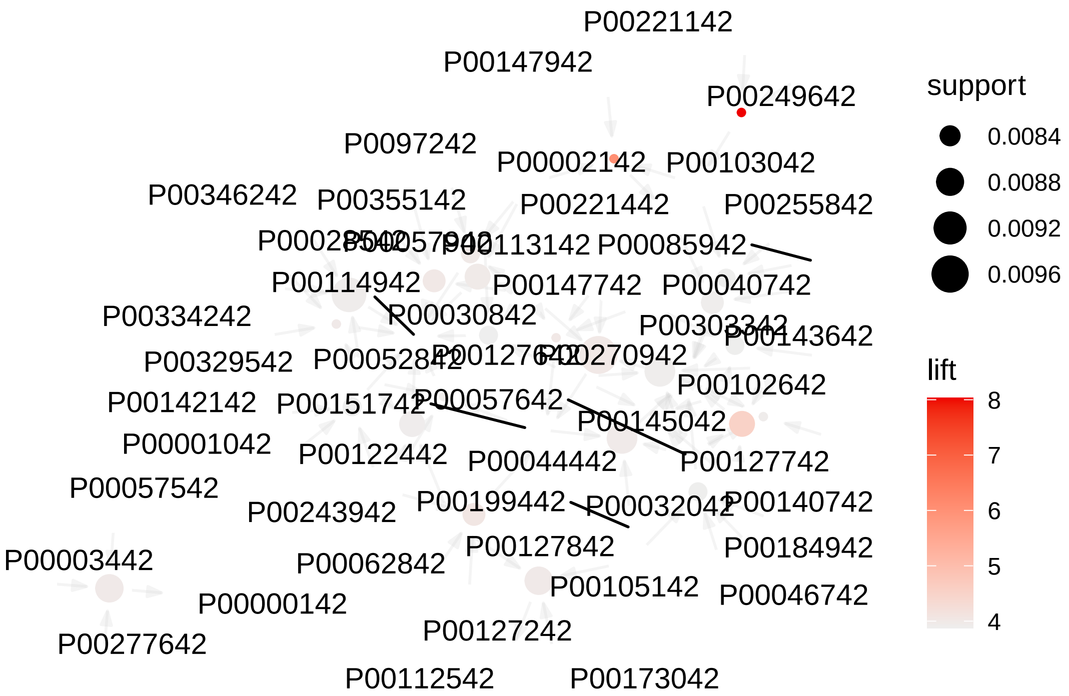
```
##     lhs              rhs             support confidence    coverage     lift count
## [1] {P00221142,
##      P00249642}  => {P00103042} 0.008146640  0.7619048 0.010692464 8.030667    48
## [2] {P00002142,
##      P00103042,
##      P00147942}  => {P00221442} 0.008146640  0.7500000 0.010862186 6.045144    48
## [3] {P00032042,
##      P00057642,
##      P00102642,
##      P00145042}  => {P00270942} 0.008655804  0.8793103 0.009843856 4.540663    51
## [4] {P00062842,
##      P00127242,
##      P00243942}  => {P00044442} 0.008486083  0.7575758 0.011201629 4.061544    50
## [5] {P00030842,
##      P00057942,
##      P00355142}  => {P00114942} 0.008486083  0.7936508 0.010692464 4.024260    50
## [6] {P00030842,
##      P00147742,
##      P00303342}  => {P00044442} 0.008146640  0.7500000 0.010862186 4.020928    48
```

A new set of rules and the rule with the highest lift value has also changed.

Rule number 1 shows that Customers who bought items P00221142 and P00249642 will also purchase item P00103042 ~76% of the time, given a support of 0.008.

```r
plot(rules, method = 'graph', max = 25)
```

```
## Warning: Too many rules supplied. Only plotting the best 25 using 'lift'
## (change control parameter max if needed).
```

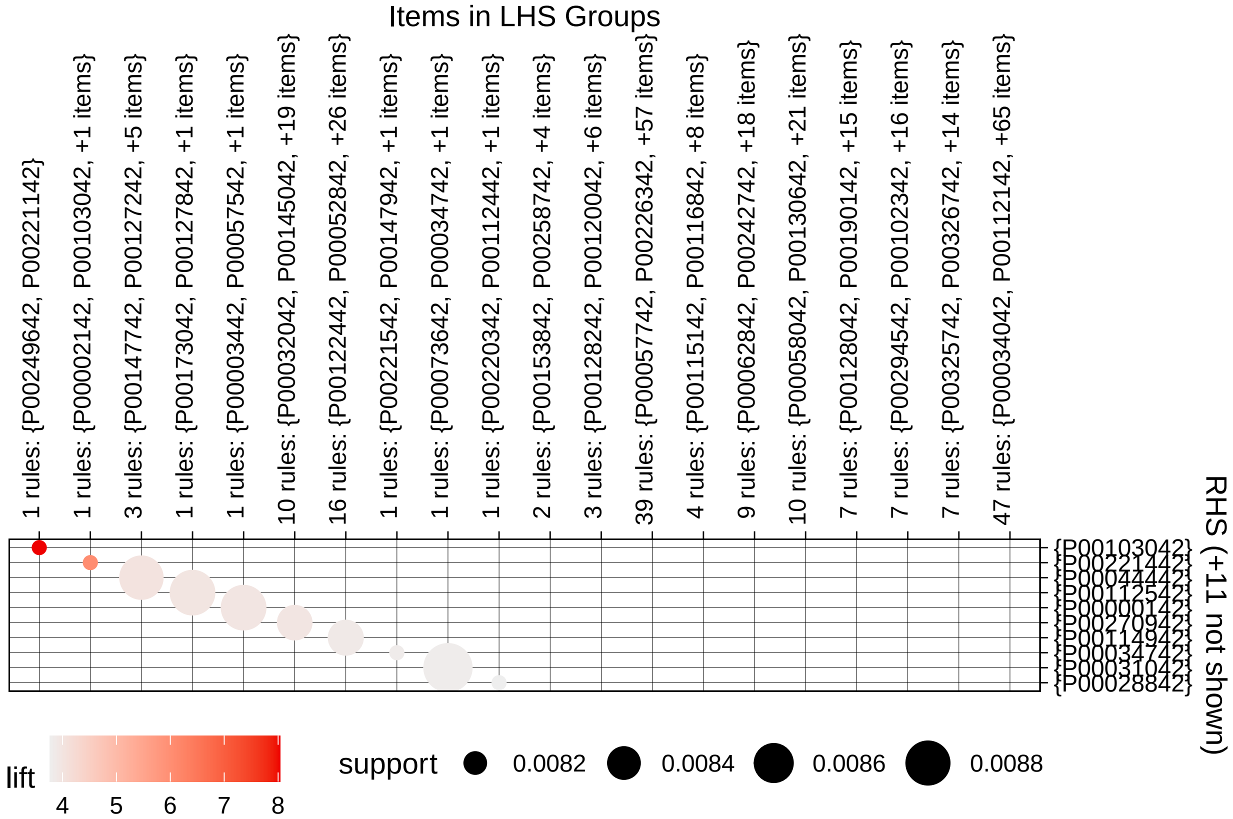


Now that 7 rules, this visualization becomes alot more difficult to interpret. Instead, create a matrix and have a similar plot and clearer interpretation.

```
plot(rules, method = 'grouped', max = 25)
```

```
## Warning: Unknown control parameters: max
```

```
## Available control parameters (with default values):
## k     =  20
## aggr.fun  =  function (x, ...)  UseMethod("mean")
## rhs_max   =  10
## lhs_label_items   =  2
## col   =  c("#EE0000FF", "#EEEEEEFF")
## groups    =  NULL
## engine    =  ggplot2
## verbose   =  FALSE
```



In this visualization, can see that LHS on top and on the right hand side, the corresponding RHS.

The size of the bubbles represents the support value of the rule and the fill/color represents the lift

5 Chapter 5

5.1 Conclusion

Summary provides a comprehensive overview of the key insights and discoveries that made from Exploratory Data Analysis (EDA) and Association Rule Learning (ARL) analysis of the Black Friday dataset. The Analysis have effectively highlighted the important aspects of analysis, from customer distribution across different categories to identifying top customers, product classifications, and various purchase metrics.

In the context of Association Rule Learning, highlighted the significance of discovering associations among items and how these insights can be leveraged by retailers to optimize product placement, recommendations, and marketing strategies.Summary effectively captures the essence of analytical journey, showcasing meticulous exploration of the dataset and the valuable insights extracted. This kind of summary is not only informative but also demonstrates the depth and quality of analysis to others, might be interested in findings.

The Exploratory Data Analysis (EDA) and Association Rule Learning (ARL) analysis of the Black Friday dataset have yielded valuable insights, providing a comprehensive understanding of customer behavior and purchase patterns. The analysis has successfully covered a range of critical aspects, including customer distribution across different categories, identification of top customers, product classifications, and various purchase metrics.

In the realm of EDA, the analysis showcased meticulous exploration of the dataset, uncovering significant trends and relationships. The distribution of customers across categories was thoroughly examined, shedding light on preferences and trends. The identification of top customers contributes to a nuanced understanding of high-value segments. Additionally, the analysis delved into product classifications, offering a detailed perspective on the product landscape and its implications for sales and marketing strategies.

The Association Rule Learning analysis emphasized the importance of discovering associations among items. Insights derived from this analysis can be leveraged by retailers to optimize product placement, enhance recommendations, and refine marketing strategies. The potential impact of these findings on retail operations is substantial, as retailers can strategically position products and tailor marketing efforts based on observed associations among items.

5.1.1 Limitations of the Analysis

However, it is crucial to acknowledge the limitations of the analysis. The insights generated are based on the available data, and the findings may not be universally applicable to all retail contexts. Furthermore, the analysis may be sensitive to data quality, and any inaccuracies or biases in the dataset could impact the robustness of the results.

5.1.2 Future Work

Looking ahead, future work could focus on expanding the dataset to include more diverse and granular information. Additionally, refining the models used in the analysis and exploring advanced techniques could further enhance the accuracy and depth of insights. Incorporating external factors, such as economic indicators or seasonal trends, could also provide a more holistic understanding of consumer behavior during Black Friday or other sales events.

In conclusion, the report's summary effectively captures the depth and quality of the analytical journey, showcasing not only the informative nature of the findings but also the potential implications for retailers. While recognizing the limitations, the report sets the stage for future research and improvements in methodology, ensuring a continuous and evolving understanding of consumer behavior in the context of Black Friday sales.

6 Reference

1. https://www.history.com/news/whats-the-real-history-of-black-friday
2. https://en.oxforddictionaries.com/explore/why-is-day-after-thanksgiving-black-friday/
3. https://www.cnn.com/2018/11/21/business/black-friday-history/index.html
4. https://journals.sagepub.com/doi/abs/10.1177/0887302X0402200404#articleCitationDow/downnloadContainer
5. https://en.wikipedia.org/wiki/Apriori_algorithm
6. https://en.wikipedia.org/wiki/Association_rule_learning

7 Appendix

7.1 Packages

1. https://www.tidyverse.org/
2. https://cran.r-project.org/web/packages/scales/scales.pdf
3. https://cran.r-project.org/web/packages/arules/arules.pdf
4. https://cran.r-project.org/web/packages/arulesViz/vignettes/arulesViz.pdf