Customer prototypes

Anca Macovei

2024-05-07

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.2.3

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.2.3

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(tidyr)

## Warning: package 'tidyr' was built under R version 4.2.3

data <- read.csv("D:/Facultate/Anul 2/Period 5/eLab II/supermarket\_enhanced.csv")

# heating map departments

# most expensive customers to departments  
  
# Step 1: Calculate total spending in each department  
department\_spending <- aggregate(price ~ departmentnumber, data = data, FUN = sum)  
  
# Step 2: Identify the departments with the highest total spending  
# Check the structure of the 'department\_spending' dataframe  
# Check the structure of the 'department\_spending' dataframe  
str(department\_spending)

## 'data.frame': 18 obs. of 2 variables:  
## $ departmentnumber: int 1 2 3 4 5 6 7 8 9 10 ...  
## $ price : num 200252 932718 216447 277200 205879 ...

# Check if the 'price' column exists and its class  
class(department\_spending$price)

## [1] "numeric"

# Convert 'price' to numeric if it's not already numeric  
department\_spending$price <- as.numeric(as.character(department\_spending$price))  
  
# Now, try reassigning the 'Price' column  
department\_spending$Price <- department\_spending$price  
  
# Check the structure of 'department\_spending' again  
str(department\_spending)

## 'data.frame': 18 obs. of 3 variables:  
## $ departmentnumber: int 1 2 3 4 5 6 7 8 9 10 ...  
## $ price : num 200252 932718 216447 277200 205879 ...  
## $ Price : num 200252 932718 216447 277200 205879 ...

# Now, try ordering again  
top\_departments <- head(department\_spending[order(department\_spending$Price, decreasing = TRUE), ], n = 18)  
  
  
# Step 3: Count the number of unique customers who made purchases in each of the identified departments  
customer\_count <- aggregate(tripnumber ~ departmentnumber, data = data, FUN = function(x) length(unique(x)))  
# If you want to count the number of purchases rather than the number of unique customers, you can use: FUN = length  
  
# Filter the customer count for the top departments  
customer\_count\_top <- customer\_count[customer\_count$departmentnumber %in% top\_departments$departmentnumber, ]  
  
# Print the results  
print("Top Departments by Total Spending:")

## [1] "Top Departments by Total Spending:"

print(top\_departments)

## departmentnumber price Price  
## 14 14 1002262.5 1002262.5  
## 2 2 932718.2 932718.2  
## 17 17 713893.5 713893.5  
## 12 12 685064.2 685064.2  
## 10 10 645333.4 645333.4  
## 16 16 421137.9 421137.9  
## 18 18 405813.7 405813.7  
## 15 15 390411.2 390411.2  
## 7 7 332560.3 332560.3  
## 11 11 326939.2 326939.2  
## 8 8 321927.8 321927.8  
## 9 9 299791.7 299791.7  
## 4 4 277200.2 277200.2  
## 3 3 216446.7 216446.7  
## 13 13 209633.0 209633.0  
## 5 5 205879.0 205879.0  
## 6 6 205867.6 205867.6  
## 1 1 200252.4 200252.4

print("Number of Customers per Top Department:")

## [1] "Number of Customers per Top Department:"

print(customer\_count\_top)

## departmentnumber tripnumber  
## 1 1 64175  
## 2 2 47398  
## 3 3 31776  
## 4 4 43280  
## 5 5 38494  
## 6 6 60582  
## 7 7 51007  
## 8 8 44831  
## 9 9 41971  
## 10 10 67698  
## 11 11 41225  
## 12 12 58287  
## 13 13 65334  
## 14 14 67044  
## 15 15 59165  
## 16 16 56363  
## 17 17 57917  
## 18 18 46302

# departments with longest + shortest time  
  
  
# Initialize vectors to store department-wise average timebetween  
department\_avg\_time <- numeric(length = 18)  
  
# Calculate average timebetween for each department  
for (dept in 1:18) {  
 department\_avg\_time[dept] <- mean(data$timebetween[data$departmentnumber == dept])  
}  
  
# Create a data frame to store department-wise average timebetween  
department\_avg\_time\_df <- data.frame(Department = 1:18, AvgTimeBetween = department\_avg\_time)  
  
# Identify the department with the highest average timebetween  
highest\_avg\_department <- department\_avg\_time\_df[which.max(department\_avg\_time\_df$AvgTimeBetween), ]  
  
# Identify the department with the lowest average timebetween  
lowest\_avg\_department <- department\_avg\_time\_df[which.min(department\_avg\_time\_df$AvgTimeBetween), ]  
  
# Print the results  
print("Department with Highest Average Timebetween:")

## [1] "Department with Highest Average Timebetween:"

print(highest\_avg\_department)

## Department AvgTimeBetween  
## 11 11 136.5869

print("Department with Lowest Average Timebetween:")

## [1] "Department with Lowest Average Timebetween:"

print(lowest\_avg\_department)

## Department AvgTimeBetween  
## 10 10 53.92871

# Combine highest and lowest average departments into a single data frame  
highest\_and\_lowest <- rbind(highest\_avg\_department, lowest\_avg\_department)  
  
# Add a column to specify whether it's the highest or lowest  
highest\_and\_lowest$Type <- c("Highest", "Lowest")  
  
# Reorder the columns with "Type" as the first one  
highest\_and\_lowest <- highest\_and\_lowest[, c("Type", "Department", "AvgTimeBetween")]  
  
# Print the combined results  
print("Departments with Highest and Lowest Average Timebetween:")

## [1] "Departments with Highest and Lowest Average Timebetween:"

print(highest\_and\_lowest)

## Type Department AvgTimeBetween  
## 11 Highest 11 136.58693  
## 10 Lowest 10 53.92871

# average number of items of customers  
  
  
  
# Calculate the number of items brought by each customer  
customer\_total\_items <- aggregate(purchasenumber ~ tripnumber, data = data, FUN = function(x) length(unique(x)))  
  
# Rename the column for clarity  
names(customer\_total\_items)[2] <- "numberofitems"  
  
# Print the first few rows to verify the results  
head(customer\_total\_items)

## tripnumber numberofitems  
## 1 1 13  
## 2 2 17  
## 3 3 29  
## 4 4 31  
## 5 5 22  
## 6 6 17

# Calculate the average number of items brought by customers  
average\_items\_per\_customer <- mean(customer\_total\_items$numberofitems)  
  
# Print the result  
print(paste("Average number of items brought by customers:", round(average\_items\_per\_customer, 0)))

## [1] "Average number of items brought by customers: 21"

# Average number of items per department  
  
# Calculate the total number of items brought by each customer in each department  
customer\_department\_total\_items <- aggregate(purchasenumber ~ tripnumber + departmentnumber, data = data, FUN = function(x) length(unique(x)))  
  
# Calculate the average number of items brought by customers per department and round to integer  
average\_items\_per\_department <- aggregate(purchasenumber ~ departmentnumber, data = customer\_department\_total\_items, FUN = function(x) round(mean(x)))  
  
# Rename the column for clarity  
names(average\_items\_per\_department)[2] <- "AvgItemsPerCustomer"  
  
# Print the result  
print("Average number of items brought by customers per department (rounded to nearest integer):")

## [1] "Average number of items brought by customers per department (rounded to nearest integer):"

print(average\_items\_per\_department)

## departmentnumber AvgItemsPerCustomer  
## 1 1 2  
## 2 2 3  
## 3 3 2  
## 4 4 2  
## 5 5 2  
## 6 6 3  
## 7 7 2  
## 8 8 2  
## 9 9 3  
## 10 10 2  
## 11 11 2  
## 12 12 2  
## 13 13 2  
## 14 14 2  
## 15 15 3  
## 16 16 2  
## 17 17 3  
## 18 18 2

# number of customer who brought more than average items per department  
  
# Merge the average items per department with the original data  
merged\_data <- merge(customer\_department\_total\_items, average\_items\_per\_department, by = "departmentnumber")  
  
# Filter customers who brought more than the average number of items per department  
customers\_above\_avg\_per\_department <- merged\_data[merged\_data$purchasenumber > merged\_data$AvgItemsPerCustomer, ]  
  
# Calculate the number of unique customers for each department  
num\_customers\_above\_avg\_per\_department <- aggregate(tripnumber ~ departmentnumber, data = customers\_above\_avg\_per\_department, FUN = function(x) length(unique(x)))  
  
# Rename the column for clarity  
names(num\_customers\_above\_avg\_per\_department)[2] <- "NumCustomersAboveAvg"  
  
# Print the result  
print("Number of customers per each department that brought more than the average items brought per department:")

## [1] "Number of customers per each department that brought more than the average items brought per department:"

print(num\_customers\_above\_avg\_per\_department)

## departmentnumber NumCustomersAboveAvg  
## 1 1 19120  
## 2 2 10592  
## 3 3 5376  
## 4 4 9639  
## 5 5 7608  
## 6 6 14431  
## 7 7 9436  
## 8 8 13481  
## 9 9 9115  
## 10 10 23188  
## 11 11 5617  
## 12 12 20152  
## 13 13 17272  
## 14 14 15438  
## 15 15 12977  
## 16 16 17660  
## 17 17 15424  
## 18 18 12792

# number of customers who brought more than 20 items per department  
  
# Filter customers who brought more than 20 items per department  
customers\_above\_20\_items\_per\_department <- customer\_department\_total\_items[customer\_department\_total\_items$purchasenumber > 20, ]  
  
# Calculate the number of unique customers for each department  
num\_customers\_above\_20\_items\_per\_department <- aggregate(tripnumber ~ departmentnumber, data = customers\_above\_20\_items\_per\_department, FUN = function(x) length(unique(x)))  
  
# Rename the column for clarity  
names(num\_customers\_above\_20\_items\_per\_department)[2] <- "NumCustomersAbove20Items"  
  
# Print the result  
print("Number of customers per each department that brought more than 20 items per department:")

## [1] "Number of customers per each department that brought more than 20 items per department:"

print(num\_customers\_above\_20\_items\_per\_department)

## departmentnumber NumCustomersAbove20Items  
## 1 1 3  
## 2 2 19  
## 3 4 3  
## 4 6 8  
## 5 9 42  
## 6 10 8  
## 7 12 1  
## 8 13 1  
## 9 14 1  
## 10 15 14  
## 11 16 18  
## 12 17 17  
## 13 18 19