Assignemnt2-online Retail Analytics

Avinash Ravipudi

2022-10-19

library(dplyr)

##   
## Attaching package: 'dplyr'

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

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

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(tidyverse)

## ── Attaching packages  
## ───────────────────────────────────────  
## tidyverse 1.3.2 ──

## ✔ tibble 3.1.8 ✔ purrr 0.3.4  
## ✔ tidyr 1.2.0 ✔ stringr 1.4.1  
## ✔ readr 2.1.2 ✔ forcats 0.5.2  
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ✖ purrr::lift() masks caret::lift()

#install.packages("tinytex")  
library(tinytex)

#Importing Data set

#importing and converting data sets   
getwd()

## [1] "/Users/avinashravipudi/Documents/BA/Assignament2-Online Retail Analytics"

OR<-read.csv("Online\_Retail.csv")  
head(OR,10)

## InvoiceNo StockCode Description Quantity  
## 1 536365 85123A WHITE HANGING HEART T-LIGHT HOLDER 6  
## 2 536365 71053 WHITE METAL LANTERN 6  
## 3 536365 84406B CREAM CUPID HEARTS COAT HANGER 8  
## 4 536365 84029G KNITTED UNION FLAG HOT WATER BOTTLE 6  
## 5 536365 84029E RED WOOLLY HOTTIE WHITE HEART. 6  
## 6 536365 22752 SET 7 BABUSHKA NESTING BOXES 2  
## 7 536365 21730 GLASS STAR FROSTED T-LIGHT HOLDER 6  
## 8 536366 22633 HAND WARMER UNION JACK 6  
## 9 536366 22632 HAND WARMER RED POLKA DOT 6  
## 10 536367 84879 ASSORTED COLOUR BIRD ORNAMENT 32  
## InvoiceDate UnitPrice CustomerID Country  
## 1 12/1/2010 8:26 2.55 17850 United Kingdom  
## 2 12/1/2010 8:26 3.39 17850 United Kingdom  
## 3 12/1/2010 8:26 2.75 17850 United Kingdom  
## 4 12/1/2010 8:26 3.39 17850 United Kingdom  
## 5 12/1/2010 8:26 3.39 17850 United Kingdom  
## 6 12/1/2010 8:26 7.65 17850 United Kingdom  
## 7 12/1/2010 8:26 4.25 17850 United Kingdom  
## 8 12/1/2010 8:28 1.85 17850 United Kingdom  
## 9 12/1/2010 8:28 1.85 17850 United Kingdom  
## 10 12/1/2010 8:34 1.69 13047 United Kingdom

# In this case, I'm computing the proportion of missing values for each variable in the data frame.  
colMeans(is.na(OR))

## InvoiceNo StockCode Description Quantity InvoiceDate UnitPrice   
## 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000   
## CustomerID Country   
## 0.2492669 0.0000000

#Total number of transactions in dataset by nation and display only those more than 1%  
OR %>%   
 group\_by(Country) %>%  
 summarise(Total\_Transcation = n(), Percent\_Total = 100\*(n()/nrow(OR))) %>%  
 filter(Percent\_Total > 1.0) %>%   
 arrange(desc(Percent\_Total))

## # A tibble: 4 × 3  
## Country Total\_Transcation Percent\_Total  
## <chr> <int> <dbl>  
## 1 United Kingdom 495478 91.4   
## 2 Germany 9495 1.75  
## 3 France 8557 1.58  
## 4 EIRE 8196 1.51

# added new variable to dataframe TransactionValue = Quality + Price  
OR$TransactionValue<-(OR$Quantity+OR$UnitPrice)  
head(OR,10)

## InvoiceNo StockCode Description Quantity  
## 1 536365 85123A WHITE HANGING HEART T-LIGHT HOLDER 6  
## 2 536365 71053 WHITE METAL LANTERN 6  
## 3 536365 84406B CREAM CUPID HEARTS COAT HANGER 8  
## 4 536365 84029G KNITTED UNION FLAG HOT WATER BOTTLE 6  
## 5 536365 84029E RED WOOLLY HOTTIE WHITE HEART. 6  
## 6 536365 22752 SET 7 BABUSHKA NESTING BOXES 2  
## 7 536365 21730 GLASS STAR FROSTED T-LIGHT HOLDER 6  
## 8 536366 22633 HAND WARMER UNION JACK 6  
## 9 536366 22632 HAND WARMER RED POLKA DOT 6  
## 10 536367 84879 ASSORTED COLOUR BIRD ORNAMENT 32  
## InvoiceDate UnitPrice CustomerID Country TransactionValue  
## 1 12/1/2010 8:26 2.55 17850 United Kingdom 8.55  
## 2 12/1/2010 8:26 3.39 17850 United Kingdom 9.39  
## 3 12/1/2010 8:26 2.75 17850 United Kingdom 10.75  
## 4 12/1/2010 8:26 3.39 17850 United Kingdom 9.39  
## 5 12/1/2010 8:26 3.39 17850 United Kingdom 9.39  
## 6 12/1/2010 8:26 7.65 17850 United Kingdom 9.65  
## 7 12/1/2010 8:26 4.25 17850 United Kingdom 10.25  
## 8 12/1/2010 8:28 1.85 17850 United Kingdom 7.85  
## 9 12/1/2010 8:28 1.85 17850 United Kingdom 7.85  
## 10 12/1/2010 8:34 1.69 13047 United Kingdom 33.69

#Transcation Value by countries  
OR %>%   
 group\_by(Country) %>%  
 summarise(Total\_Spend = sum(TransactionValue)) %>%  
 filter(Total\_Spend > 130000) %>%   
 arrange(desc(Total\_Spend))

## # A tibble: 5 × 2  
## Country Total\_Spend  
## <chr> <dbl>  
## 1 United Kingdom 6509544.  
## 2 Netherlands 206621.  
## 3 EIRE 191084.  
## 4 Germany 155114   
## 5 France 153512.

#I'm creating a temporary variable that will format the transaction date to mm/dd/yyyy, and I'm checking the format using the head command.  
Temp=strptime(OR$InvoiceDate,format='%m/%d/%Y %H:%M',tz='GMT')  
head(Temp)

## [1] "2010-12-01 08:26:00 GMT" "2010-12-01 08:26:00 GMT"  
## [3] "2010-12-01 08:26:00 GMT" "2010-12-01 08:26:00 GMT"  
## [5] "2010-12-01 08:26:00 GMT" "2010-12-01 08:26:00 GMT"

OR$New\_Invoice\_Date <- as.Date(Temp)# The Temp variable is used to format the New Invoice Date column into a date format.  
  
OR$New\_Invoice\_Date[20000]- OR$New\_Invoice\_Date[10]

## Time difference of 8 days

OR$Invoice\_Day\_Week= weekdays(OR$New\_Invoice\_Date) #converting dates to weekdays  
  
OR$New\_Invoice\_Hour = as.numeric(format(Temp, "%H"))# Adding a new column containing the transaction hour. New Invoice Hour has been allocated  
  
OR$New\_Invoice\_Month = as.numeric(format(Temp, "%m"))#assigning a new column with the transaction month to New Invoice Hour  
  
#grouping the data frame by weekday and computing the percentage of transactions (by number) each day and returning the numbers in decreasing order of percentages  
  
OR %>%  
 group\_by(Invoice\_Day\_Week) %>%  
 summarise(percent\_of\_transactions = 100\*(n()/nrow(OR))) %>%  
 arrange(desc(percent\_of\_transactions))

## # A tibble: 6 × 2  
## Invoice\_Day\_Week percent\_of\_transactions  
## <chr> <dbl>  
## 1 Thursday 19.2  
## 2 Tuesday 18.8  
## 3 Monday 17.6  
## 4 Wednesday 17.5  
## 5 Friday 15.2  
## 6 Sunday 11.9

#arranging the data frame by weekday, I'm calculating the proportion of transactions (by transaction values) every day and returning the percentages in decreasing order.   
  
OR %>%  
 group\_by(Invoice\_Day\_Week) %>%  
 summarise(percent\_of\_transactions\_by\_volume = 100\*(sum(TransactionValue)/sum(OR$TransactionValue))) %>%  
 arrange(desc(percent\_of\_transactions\_by\_volume))

## # A tibble: 6 × 2  
## Invoice\_Day\_Week percent\_of\_transactions\_by\_volume  
## <chr> <dbl>  
## 1 Thursday 20.8   
## 2 Tuesday 19.3   
## 3 Wednesday 18.3   
## 4 Monday 17.6   
## 5 Friday 15.5   
## 6 Sunday 8.52

#I'm grouping the data frame by year, calculating the % of transactions (by transaction values) per month, and providing the results in decreasing order of percentages.  
  
OR %>%  
 group\_by(New\_Invoice\_Month) %>%  
 summarise(percent\_of\_transactions\_by\_volume = 100\*(sum(TransactionValue)/sum(OR$TransactionValue))) %>%  
 arrange(desc(percent\_of\_transactions\_by\_volume))

## # A tibble: 12 × 2  
## New\_Invoice\_Month percent\_of\_transactions\_by\_volume  
## <dbl> <dbl>  
## 1 11 13.9   
## 2 12 12.5   
## 3 10 10.9   
## 4 9 9.76  
## 5 5 7.44  
## 6 7 7.34  
## 7 8 7.25  
## 8 6 7.07  
## 9 3 6.82  
## 10 1 6.28  
## 11 4 5.45  
## 12 2 5.28

# producing a subset of data for Australian transactions, grouping by invoice date, and returning the year's top values  
  
subset(OR, Country == "Australia") %>%  
 group\_by(New\_Invoice\_Date) %>%  
 summarise(n\_transactions = n()) %>%  
 top\_n(3)

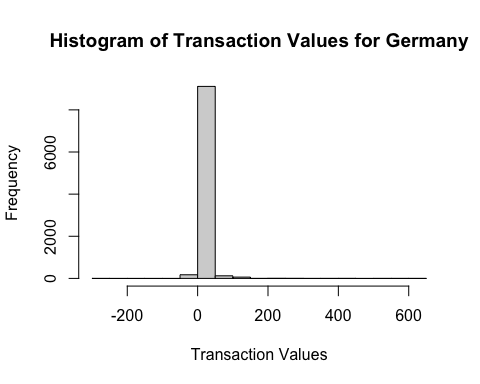
## Selecting by n\_transactions

## # A tibble: 3 × 2  
## New\_Invoice\_Date n\_transactions  
## <date> <int>  
## 1 2011-06-15 139  
## 2 2011-07-19 137  
## 3 2011-08-18 97

# I'm grouping the data frame for transactions by hours and summarizing the data to get the percentage of transactions by number, and then I'm returning the values in ascending order.  
  
OR %>%  
 group\_by(New\_Invoice\_Hour) %>%  
 summarise(percent\_of\_transactions = 100\*(n()/nrow(OR))) %>%  
 arrange(percent\_of\_transactions)

## # A tibble: 15 × 2  
## New\_Invoice\_Hour percent\_of\_transactions  
## <dbl> <dbl>  
## 1 6 0.00757  
## 2 7 0.0707   
## 3 20 0.161   
## 4 19 0.684   
## 5 18 1.47   
## 6 8 1.64   
## 7 17 5.26   
## 8 9 6.34   
## 9 10 9.05   
## 10 16 10.1   
## 11 11 10.6   
## 12 14 12.5   
## 13 13 13.3   
## 14 15 14.3   
## 15 12 14.5

# I'm now developing a new variable for Germany and visualizing transaction numbers on a histogram.  
Germany\_Transactions <- subset(OR, Country == "Germany")  
hist(Germany\_Transactions$TransactionValue, main = "Histogram of Transaction Values for Germany", xlab = "Transaction Values", ylab = "Frequency")



# I'm aggregating the data by customer, then summarizing it based on count and returning the top three numbers that are shown in decreasing value order.  
  
OR %>%  
 group\_by(CustomerID) %>%  
 summarise(n\_transactions = n()) %>%  
 top\_n(3) %>%  
 arrange(desc(n\_transactions))

## Selecting by n\_transactions

## # A tibble: 3 × 2  
## CustomerID n\_transactions  
## <int> <int>  
## 1 NA 135080  
## 2 17841 7983  
## 3 14911 5903

#I'm aggregating the data by customer, then summarizing it based on transaction values and returning the top three numbers that are shown in decreasing value order.  
  
OR %>%  
 group\_by(CustomerID) %>%  
 summarise(transaction\_sum = sum(TransactionValue)) %>%  
 top\_n(3) %>%  
 arrange(desc(transaction\_sum))

## Selecting by transaction\_sum

## # A tibble: 3 × 2  
## CustomerID transaction\_sum  
## <int> <dbl>  
## 1 NA 1360546.  
## 2 14646 202119.  
## 3 14911 108241.

# filtering out non-NA data, grouping by nation, and summarizing by total count  
  
OR %>%  
 filter(is.na(OR$CustomerID)) %>%  
 group\_by(Country) %>%  
 summarise(n\_missing\_ID = n()) %>%  
 arrange(desc(n\_missing\_ID))

## # A tibble: 9 × 2  
## Country n\_missing\_ID  
## <chr> <int>  
## 1 United Kingdom 133600  
## 2 EIRE 711  
## 3 Hong Kong 288  
## 4 Unspecified 202  
## 5 Switzerland 125  
## 6 France 66  
## 7 Israel 47  
## 8 Portugal 39  
## 9 Bahrain 2

# by deleting "NA" CustomerIDs from a data frame  
  
Online\_Retail\_NA\_Removed <- na.omit(OR)  
  
#constructing a data frame by eliminating cancelled transactions  
  
Online\_Retail\_NA\_Neg\_Removed <- subset(Online\_Retail\_NA\_Removed, Quantity > 0)  
  
#generating a data frame containing only the customerID and transaction date  
  
Online\_Retail\_Subset <- Online\_Retail\_NA\_Neg\_Removed[,c("CustomerID","New\_Invoice\_Date")]  
  
#constructing a data frame that deletes numerous invoices from the same client on the same day  
  
Online\_Retail\_Subset\_Distinct <- distinct(Online\_Retail\_Subset)  
  
#Organizing the data set by CustomerID and date, and calculating the average duration between successive transactions for each client. removing CustomerIDs that result in a NA value (i.e. having just one distinct transaction) and summarizing the data to get the average duration between shopping visits for all CustomerIDs  
  
Online\_Retail\_Subset\_Distinct %>%  
 group\_by(CustomerID) %>%  
 arrange(New\_Invoice\_Date) %>%  
 summarise(avg = mean(diff(New\_Invoice\_Date))) %>%  
 na.omit() %>%  
 summarise(avg\_days\_between\_shopping = mean(avg))

## # A tibble: 1 × 1  
## avg\_days\_between\_shopping  
## <drtn>   
## 1 78.42025 days

#Creating two new subgroups for France that compute the total number of returns and total number of transactions and are used to determine the return rate.  
  
France\_Transactions\_Cancelled <- subset(OR, Country == "France" & Quantity < 0)  
France\_Transactions <- subset(OR, Country == "France")  
France\_Return\_Rate <- 100\*(nrow(France\_Transactions\_Cancelled) / nrow(France\_Transactions))  
France\_Return\_Rate

## [1] 1.741264

# I'm categorizing data by StockCode and item description, then summarizing it based on transaction values. and then returning the data in descending order  
  
OR %>%  
 group\_by(StockCode, Description) %>%  
 summarise(transaction\_sum = sum(TransactionValue)) %>%  
 arrange(desc(transaction\_sum))

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

## # A tibble: 5,752 × 3  
## # Groups: StockCode [4,070]  
## StockCode Description transaction\_sum  
## <chr> <chr> <dbl>  
## 1 AMAZONFEE AMAZON FEE 249013.  
## 2 M Manual 217612.  
## 3 DOT DOTCOM POSTAGE 206959.  
## 4 84077 WORLD WAR 2 GLIDERS ASSTD DESIGNS 54020.  
## 5 85099B JUMBO BAG RED RETROSPOT 52696.  
## 6 POST POSTAGE 49391.  
## 7 22423 REGENCY CAKESTAND 3 TIER 43394.  
## 8 85123A WHITE HANGING HEART T-LIGHT HOLDER 42171.  
## 9 84879 ASSORTED COLOUR BIRD ORNAMENT 38966.  
## 10 22197 POPCORN HOLDER 37208.  
## # … with 5,742 more rows

#By deleting duplicate entries, the length of the CustomerID vector is returned.  
length(unique(OR$CustomerID))

## [1] 4373