BA OL Retail Analytics

Sai Supriya Vengala

10/23/2022

#Required libraries are to be loaded first  
library(readr)

## Warning: package 'readr' was built under R version 4.1.2

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.1.2

##   
## 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

Online\_Retail <- read\_csv("Online\_Retail.csv")

## Rows: 541909 Columns: 8

## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (5): InvoiceNo, StockCode, Description, InvoiceDate, Country  
## dbl (3): Quantity, UnitPrice, CustomerID  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

#task1: calculating the countries accounting for more than 1% of the total transactions.  
trans\_country <-table(Online\_Retail$Country)  
transaction\_percent<- round(100\*prop.table(trans\_country))  
percentage <- cbind(trans\_country, transaction\_percent)  
account <-subset(percentage, transaction\_percent >1)  
account

## trans\_country transaction\_percent  
## EIRE 8196 2  
## France 8557 2  
## Germany 9495 2  
## United Kingdom 495478 91

#task2:Creating a new variable ‘TransactionValue’ that is the product of the exising #‘Quantity’ and ‘UnitPrice’ variables.

TransactionValue <-Online\_Retail$Quantity \* Online\_Retail$UnitPrice  
Online\_Retail <- Online\_Retail %>% mutate(TransactionValue)  
summary(Online\_Retail$TransactionValue)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -168469.60 3.40 9.75 17.99 17.40 168469.60

#task3:Calculating the countries with total transaction exceeding 130,000 British Pound

Sum\_trans <- sum(TransactionValue)  
store<-summarise(group\_by(Online\_Retail, Country), Sum\_trans)  
Total <- filter(store, Sum\_trans >130000)  
Total

## # A tibble: 38 × 2  
## Country Sum\_trans  
## <chr> <dbl>  
## 1 Australia 9747748.  
## 2 Austria 9747748.  
## 3 Bahrain 9747748.  
## 4 Belgium 9747748.  
## 5 Brazil 9747748.  
## 6 Canada 9747748.  
## 7 Channel Islands 9747748.  
## 8 Cyprus 9747748.  
## 9 Czech Republic 9747748.  
## 10 Denmark 9747748.  
## # … with 28 more rows

#task4

Temp=strptime(Online\_Retail$InvoiceDate,format='%m/%d/%Y %H:%M',tz='GMT')  
Online\_Retail$New\_Invoice\_Date <- as.Date(Temp)  
Diff <- Online\_Retail$New\_Invoice\_Date[20000]- Online\_Retail$New\_Invoice\_Date[10]  
Diff

## Time difference of 8 days

Online\_Retail$Invoice\_Day\_Week= weekdays(Online\_Retail$New\_Invoice\_Date)   
Online\_Retail$New\_Invoice\_Hour = as.numeric(format(Temp, "%H"))  
Online\_Retail$New\_Invoice\_Month = as.numeric(format(Temp, "%m"))

#a)Showing the percentage of transactions (by numbers) by #days of the week

country\_totaltran1<-summarise(group\_by(Online\_Retail,Invoice\_Day\_Week) ,Trans\_val=n\_distinct(InvoiceNo))  
percentage1<-mutate(country\_totaltran1,  
 Trans\_perc=(Trans\_val/sum(Trans\_val))\*100)  
percentage1

## # A tibble: 6 × 3  
## Invoice\_Day\_Week Trans\_val Trans\_perc  
## <chr> <int> <dbl>  
## 1 Friday 4184 16.2   
## 2 Monday 4138 16.0   
## 3 Sunday 2381 9.19  
## 4 Thursday 5660 21.9   
## 5 Tuesday 4722 18.2   
## 6 Wednesday 4815 18.6

#b) Showing the percentage of transactions  
#(by transaction volume) by days of the week

country\_tran1<-summarise(group\_by(Online\_Retail,Invoice\_Day\_Week),Trans\_val1=sum(TransactionValue))  
percen1<-mutate(country\_tran1,Trans\_perc1=(Trans\_val1/sum(Trans\_val1))\*100)  
percen1

## # A tibble: 6 × 3  
## Invoice\_Day\_Week Trans\_val1 Trans\_perc1  
## <chr> <dbl> <dbl>  
## 1 Friday 1540611. 15.8   
## 2 Monday 1588609. 16.3   
## 3 Sunday 805679. 8.27  
## 4 Thursday 2112519 21.7   
## 5 Tuesday 1966183. 20.2   
## 6 Wednesday 1734147. 17.8

#c)Show the percentage of transactions (by transaction volume) #by month of the year

country\_totaltran2<-summarise(group\_by(Online\_Retail,New\_Invoice\_Month),Trans\_val2=sum(TransactionValue))  
percentage2<-mutate(country\_totaltran2,Trans\_perc2=(Trans\_val2/sum(Trans\_val2))\*100)  
percentage2

## # A tibble: 12 × 3  
## New\_Invoice\_Month Trans\_val2 Trans\_perc2  
## <dbl> <dbl> <dbl>  
## 1 1 560000. 5.74  
## 2 2 498063. 5.11  
## 3 3 683267. 7.01  
## 4 4 493207. 5.06  
## 5 5 723334. 7.42  
## 6 6 691123. 7.09  
## 7 7 681300. 6.99  
## 8 8 682681. 7.00  
## 9 9 1019688. 10.5   
## 10 10 1070705. 11.0   
## 11 11 1461756. 15.0   
## 12 12 1182625. 12.1

#d) The date with the highest number of transactions from Australia.

Online\_Retail %>% filter(Country == 'Australia') %>% group\_by(New\_Invoice\_Date) %>%   
 summarise(max=max(TransactionValue))

## # A tibble: 49 × 2  
## New\_Invoice\_Date max  
## <date> <dbl>  
## 1 2010-12-01 51   
## 2 2010-12-08 71.4   
## 3 2010-12-14 -6.25  
## 4 2010-12-17 148.   
## 5 2011-01-06 1020   
## 6 2011-01-10 81.6   
## 7 2011-01-11 35.4   
## 8 2011-01-14 142.   
## 9 2011-01-17 47.4   
## 10 2011-01-19 38.2   
## # … with 39 more rows

#e)Calculating the hour of the day to start this so that the distribution is at #minimum for the customers?

library(zoo)

## Warning: package 'zoo' was built under R version 4.1.2

##   
## Attaching package: 'zoo'

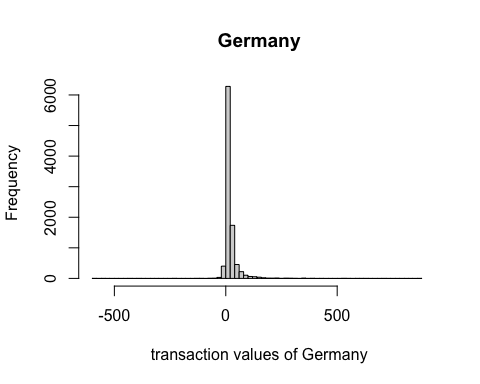
## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

start1<-summarise(group\_by(Online\_Retail,New\_Invoice\_Hour),  
 Tran\_mini=n\_distinct(InvoiceNo))  
start1<-filter(start1,New\_Invoice\_Hour>=7&New\_Invoice\_Hour<=20)  
start2<-rollapply(start1$Tran\_mini,3,sum)  
start3<-which.min(start2)  
start3

## [1] 12

#task5:Plotting the histogram of transaction values from Germany. Using the hist() function to plot.

German\_Trans <- subset(Online\_Retail$TransactionValue, Online\_Retail$Country =="Germany")  
hist(German\_Trans, xlim = c (-600, 900), breaks = 100, xlab = "transaction values of Germany",   
 main = "Germany")

 #task6:Calculating the customer with highest number of transactions most valuable customer. #(i.e. highest total sum of transactions)

High\_Trans <- na.omit(Online\_Retail)  
High\_Trans <- summarise(group\_by(Online\_Retail, CustomerID), sum2 = sum(TransactionValue))  
High\_Trans[which.max(High\_Trans$sum2),]

## # A tibble: 1 × 2  
## CustomerID sum2  
## <dbl> <dbl>  
## 1 NA 1447682.

store1 <- table(High\_Trans$CustomerID)  
store1 <- as.data.frame(store1)  
Val\_Cust <- store1[which.max(store1$Freq),]  
Val\_Cust

## Var1 Freq  
## 1 12346 1

#task7:Calculating the percentage of missing values for each variable in the dataset

Miss\_Val <-colMeans(is.na(Online\_Retail)\*100)  
Miss\_Val

## InvoiceNo StockCode Description Quantity   
## 0.0000000 0.0000000 0.2683107 0.0000000   
## InvoiceDate UnitPrice CustomerID Country   
## 0.0000000 0.0000000 24.9266943 0.0000000   
## TransactionValue New\_Invoice\_Date Invoice\_Day\_Week New\_Invoice\_Hour   
## 0.0000000 0.0000000 0.0000000 0.0000000   
## New\_Invoice\_Month   
## 0.0000000

#task8:Calculating the number of transactions with missing CustomerID records by countries?

Val\_Cust <- Online\_Retail %>% filter(is.na(CustomerID)) %>% group\_by(Country)  
summary(Val\_Cust$Country)

## Length Class Mode   
## 135080 character character

#task9:Calculating how often the costumers #comeback to the website for their next shopping

Freaq <- Online\_Retail %>%  
 group\_by(InvoiceNo, CustomerID, Country, New\_Invoice\_Date, New\_Invoice\_Month, New\_Invoice\_Hour,   
 Invoice\_Day\_Week) %>%  
 summarise(Trans6 = sum(TransactionValue)) %>%  
 mutate(Freq1 = Sys.Date() - New\_Invoice\_Date) %>%  
 ungroup()

## `summarise()` has grouped output by 'InvoiceNo', 'CustomerID', 'Country',  
## 'New\_Invoice\_Date', 'New\_Invoice\_Month', 'New\_Invoice\_Hour'. You can override  
## using the `.groups` argument.

Freaq$Freq1 <- as.character(Freaq$Freq1)  
Freaq$Freq2 <- sapply(Freaq$Freq1,   
 FUN = function(x) {strsplit(x, split = '[ ]')[[1]][1]})  
Freaq$Freq2 <- as.integer(Freaq$Freq2)  
head(Freaq, n = 5)

## # A tibble: 5 × 10  
## Invoic…¹ Custo…² Country New\_Invo…³ New\_I…⁴ New\_I…⁵ Invoi…⁶ Trans6 Freq1 Freq2  
## <chr> <dbl> <chr> <date> <dbl> <dbl> <chr> <dbl> <chr> <int>  
## 1 536365 17850 United… 2010-12-01 12 8 Wednes… 139. 4345 4345  
## 2 536366 17850 United… 2010-12-01 12 8 Wednes… 22.2 4345 4345  
## 3 536367 13047 United… 2010-12-01 12 8 Wednes… 279. 4345 4345  
## 4 536368 13047 United… 2010-12-01 12 8 Wednes… 70.1 4345 4345  
## 5 536369 13047 United… 2010-12-01 12 8 Wednes… 17.8 4345 4345  
## # … with abbreviated variable names ¹​InvoiceNo, ²​CustomerID, ³​New\_Invoice\_Date,  
## # ⁴​New\_Invoice\_Month, ⁵​New\_Invoice\_Hour, ⁶​Invoice\_Day\_Week

attach(Freaq)  
FreaqCust <- Online\_Retail %>%  
 group\_by(CustomerID, Country) %>%  
 summarise(Cust\_order = n\_distinct(InvoiceNo),  
 Trans7 = sum(TransactionValue),   
 PerDay = names(which.max(table(Invoice\_Day\_Week))),   
PerHour=names(which.max(table(New\_Invoice\_Hour))),  
 Frequency = min(Freaq$Freq2))%>%  
 ungroup()

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

head(FreaqCust)

## # A tibble: 6 × 7  
## CustomerID Country Cust\_order Trans7 PerDay PerHour Frequency  
## <dbl> <chr> <int> <dbl> <chr> <chr> <int>  
## 1 12346 United Kingdom 2 0 Tuesday 10 3972  
## 2 12347 Iceland 7 4310 Tuesday 14 3972  
## 3 12348 Finland 4 1797. Thursday 19 3972  
## 4 12349 Italy 1 1758. Monday 9 3972  
## 5 12350 Norway 1 334. Wednesday 16 3972  
## 6 12352 Norway 11 1545. Tuesday 14 3972

#task10:Calculating the return rate for the French customers.

France\_Trans <- filter(Online\_Retail, Country=="France")  
Trow <- nrow(France\_Trans)  
Cancel\_Trans <- nrow(subset(France\_Trans,TransactionValue<0))  
Cancel\_Trans

## [1] 149

No\_Cancel<- Trow-Cancel\_Trans  
No\_Cancel

## [1] 8408

Return=(Cancel\_Trans/8556)  
Return

## [1] 0.01741468

#task11:Calculating the product that has generated the highest revenue for the retailer. #(i.e. item with the highest total sum of ‘TransactionValue’).

TransactionValue <- tapply(Online\_Retail$TransactionValue, Online\_Retail$StockCode, sum)  
TransactionValue[which.max(TransactionValue)] # to find highest value

## DOT   
## 206245.5

#task12:Finding the unique customers that are represented in the dataset using unique() function

Unique\_Cust <- unique(Online\_Retail$CustomerID)  
length(Unique\_Cust )

## [1] 4373