BA\_64036\_Assignment\_2

Yeswanth Siripurapu

2023-10-15

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

#Reading the CSV File

OnlineRetail<- read.csv("D:/Online\_Retail.csv")

#1. Show the breakdown of the number of transactions by countries i.e., how many transactions are in the dataset for each country (consider all records including #cancelled transactions). Show this in total number and also in percentage. #Show only countries accounting for more than 1% of the total transactions.

Countries\_count <- OnlineRetail %>% group\_by(Country) %>% count(Country)  
Countries\_pct <- OnlineRetail %>% group\_by(Country) %>% summarise(percent = 100\* n()/nrow(OnlineRetail))  
Fltrd\_Cntry\_pct <- filter(Countries\_pct, percent>1)  
  
#Countries Count  
Countries\_count

## # A tibble: 38 × 2  
## # Groups: Country [38]  
## Country n  
## <chr> <int>  
## 1 Australia 1259  
## 2 Austria 401  
## 3 Bahrain 19  
## 4 Belgium 2069  
## 5 Brazil 32  
## 6 Canada 151  
## 7 Channel Islands 758  
## 8 Cyprus 622  
## 9 Czech Republic 30  
## 10 Denmark 389  
## # ℹ 28 more rows

#Percentage of transactions greater than 1

Fltrd\_Cntry\_pct

## # A tibble: 4 × 2  
## Country percent  
## <chr> <dbl>  
## 1 EIRE 1.51  
## 2 France 1.58  
## 3 Germany 1.75  
## 4 United Kingdom 91.4

#2 Create a new variable ‘TransactionValue’ that is the product of the exising #‘Quantity’ and ‘UnitPrice’ variables. Add this variable to the dataframe.

TransactionValue = (OnlineRetail$Quantity \* OnlineRetail$UnitPrice)  
  
#Adding the TransactionValue column to the OnlineRetail table  
Online\_Retail = cbind(OnlineRetail,TransactionValue)

#3 Using the newly created variable, TransactionValue, show the breakdown of #transaction values by countries i.e. how much money in total has been spent #each country. Show this in total sum of transaction values. Show only countries #with total transaction exceeding 130,000 British Pound.

Trans\_sum = Online\_Retail %>% group\_by(Country) %>%   
 summarise(sum=sum(TransactionValue))  
  
Fltrd\_Trans\_sum = filter(Trans\_sum,Trans\_sum$sum>130000)  
  
#Sum of TransactionValue for each countries  
Trans\_sum

## # A tibble: 38 × 2  
## Country sum  
## <chr> <dbl>  
## 1 Australia 137077.  
## 2 Austria 10154.  
## 3 Bahrain 548.  
## 4 Belgium 40911.  
## 5 Brazil 1144.  
## 6 Canada 3666.  
## 7 Channel Islands 20086.  
## 8 Cyprus 12946.  
## 9 Czech Republic 708.  
## 10 Denmark 18768.  
## # ℹ 28 more rows

#Filtering the transactions greater than 130000

Fltrd\_Trans\_sum

## # A tibble: 6 × 2  
## Country sum  
## <chr> <dbl>  
## 1 Australia 137077.  
## 2 EIRE 263277.  
## 3 France 197404.  
## 4 Germany 221698.  
## 5 Netherlands 284662.  
## 6 United Kingdom 8187806.

#4 This is an optional question which carries additional marks #(golden questions). In this question,we are dealing with the InvoiceDate #variable. The variable is read as a categorical when you read data from the #file. Now we need to explicitly instruct R to interpret this as a Date #variable. “POSIXlt” and “POSIXct” are two powerful object classes in R to #deal with date and time.

Temp=strptime(Online\_Retail$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"

Online\_Retail$New\_Invoice\_Date <- as.Date(Temp)  
Online\_Retail$Invoice\_Day\_Week= weekdays(Online\_Retail$New\_Invoice\_Date)  
Online\_Retail$New\_Invoice\_Hour = as.numeric(format(Temp, "%H"))  
Online\_Retail$New\_Invoice\_Month = as.numeric(format(Temp, "%m"))  
Online\_Retail$New\_Invoice\_Date[20000]- Online\_Retail$New\_Invoice\_Date[10]

## Time difference of 8 days

#4(a)

#Percentage of number of transactions based on week days  
Week\_days\_count = Online\_Retail %>% group\_by(Invoice\_Day\_Week) %>%   
 summarise(percent = 100\* n()/nrow(Online\_Retail))  
Week\_days\_count

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

#4(b)

#percentage of TransactionsValue  
Week\_days\_sum = Online\_Retail %>% group\_by(Invoice\_Day\_Week) %>% summarise(sum=sum(TransactionValue))  
#Calculating the percentage for TransactionValue by week days  
Week\_quan\_pct = 100\*(Week\_days\_sum$sum)/sum(Week\_days\_sum$sum)  
#replacing the sum with the percentage value  
Week\_days\_sum$sum = Week\_quan\_pct  
Week\_days\_sum

## # A tibble: 6 × 2  
## Invoice\_Day\_Week sum  
## <chr> <dbl>  
## 1 Friday 15.8   
## 2 Monday 16.3   
## 3 Sunday 8.27  
## 4 Thursday 21.7   
## 5 Tuesday 20.2   
## 6 Wednesday 17.8

#4(c)

#Percentage of TransactionsValue by month of the year  
Invoice\_month\_sum = Online\_Retail %>% group\_by(New\_Invoice\_Month) %>% summarise(sum=sum(TransactionValue))  
Month\_quan\_pct = 100\*(Invoice\_month\_sum$sum)/sum(Invoice\_month\_sum$sum)  
Invoice\_month\_sum$sum = Month\_quan\_pct  
Invoice\_month\_sum

## # A tibble: 12 × 2  
## New\_Invoice\_Month sum  
## <dbl> <dbl>  
## 1 1 5.74  
## 2 2 5.11  
## 3 3 7.01  
## 4 4 5.06  
## 5 5 7.42  
## 6 6 7.09  
## 7 7 6.99  
## 8 8 7.00  
## 9 9 10.5   
## 10 10 11.0   
## 11 11 15.0   
## 12 12 12.1

#4(d)

#Filtering the Australia's transactions based on New\_Invoice\_date  
Australia\_trans = Online\_Retail %>% filter(Country == "Australia") %>% group\_by(New\_Invoice\_Date) %>% summarise(total=n())  
#Finding the date which has maximum number of transactions  
Max\_trans\_date = Australia\_trans[which.max(Australia\_trans$total),]  
Max\_trans\_date

## # A tibble: 1 × 2  
## New\_Invoice\_Date total  
## <date> <int>  
## 1 2011-06-15 139

#4(e)

#Filtering the transactions for the hours between 7:00 to 20:00  
Sum\_quan = Online\_Retail %>% filter( New\_Invoice\_Hour >=7) %>%   
 group\_by(New\_Invoice\_Hour) %>% summarise(sum\_val= sum(Quantity))  
#install.packages("zoo")  
library(zoo)

##   
## Attaching package: 'zoo'

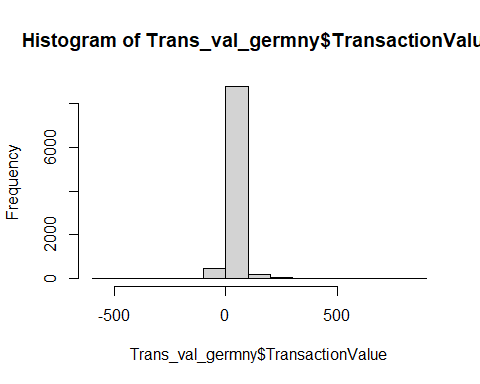
## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

#Adding the two consecutive rows  
Consec\_sum=rollapply(Sum\_quan$sum\_val,2,sum)  
#Creating the maintainance column  
maintainance=c(7:19)  
#creating the dataframe for the maintainance and Consec\_sum  
Main\_tab=data.frame(maintainance,Consec\_sum)  
#checking the minimum value of Consec\_sum and the hour where they can start  
#maintainance  
maintainance\_hour=Main\_tab[which.min(Main\_tab$Consec\_sum),]  
maintainance\_hour

## maintainance Consec\_sum  
## 13 19 40298

#5 Plot the histogram of transaction values from Germany. Use the hist() #function to plot.

Trans\_val\_germny = filter(Online\_Retail, Online\_Retail$Country == "Germany")  
#Plotting graph between transaction value with the frequency for Germany country  
hist(Trans\_val\_germny$TransactionValue)



#6 Which customer had the highest number of transactions? Which customer is most #valuable (i.e.highest total sum of transactions)?

#Removing the NA values of CustomerID Column  
NA\_OnlineRetail=Online\_Retail[!is.na(Online\_Retail$CustomerID),]  
#Number of transactions with respect to CustomerID  
Count\_transactions = NA\_OnlineRetail %>% group\_by(CustomerID) %>%  
 summarise(count=n())  
#printing the row which has max count of transactions  
Max\_Count\_transactions= Count\_transactions[which.max(Count\_transactions$count),]  
# Adding the transaction value with respect to Customer ID  
Sum\_transactions = NA\_OnlineRetail %>% group\_by(CustomerID) %>% summarise(Numoftransactions=(sum(TransactionValue,na.rm = T)))  
#printing the row which has max sum of transaction value  
Max\_Sum\_transactions= Sum\_transactions[which.max  
 (Sum\_transactions$Numoftransactions),]  
Max\_Count\_transactions

## # A tibble: 1 × 2  
## CustomerID count  
## <int> <int>  
## 1 17841 7983

Max\_Sum\_transactions

## # A tibble: 1 × 2  
## CustomerID Numoftransactions  
## <int> <dbl>  
## 1 14646 279489.

#7Calculate the percentage of missing values for each variable in the dataset.

#Percentage of NA's for each column  
NA\_per = colMeans(is.na(Online\_Retail))\*100  
NA\_per

## InvoiceNo StockCode Description Quantity   
## 0.00000 0.00000 0.00000 0.00000   
## InvoiceDate UnitPrice CustomerID Country   
## 0.00000 0.00000 24.92669 0.00000   
## TransactionValue New\_Invoice\_Date Invoice\_Day\_Week New\_Invoice\_Hour   
## 0.00000 0.00000 0.00000 0.00000   
## New\_Invoice\_Month   
## 0.00000

#8 What are the number of transactions with missing CustomerID records by #countries?

#Number of Transactions with missing customer ID  
null\_Customer = Online\_Retail[is.na(Online\_Retail$CustomerID),]  
# Segregating the missing CustomerID based on countries  
table(null\_Customer$Country)

##   
## Bahrain EIRE France Hong Kong Israel   
## 2 711 66 288 47   
## Portugal Switzerland United Kingdom Unspecified   
## 39 125 133600 202

#9 On average, how often the costumers comeback to the website for their #next shopping? (i.e. what is the average number of days between #consecutive shopping)

# Check for missing values  
if (any(is.na(OnlineRetail$InvoiceDate))) {  
 # Handle missing values (e.g., remove or impute)  
 OnlineRetail <- OnlineRetail[!is.na(OnlineRetail$InvoiceDate), ]  
}  
  
# Ensure the correct data type and format  
OnlineRetail$InvoiceDate <- as.POSIXct(OnlineRetail$InvoiceDate,  
 format = "%Y-%m-%d %H:%M:%S")  
  
# Calculate the difference in days between consecutive purchases  
days\_between\_purchases <- diff(OnlineRetail$InvoiceDate)  
  
# Calculate the average of the differences in days  
average\_days\_between\_purchases <- mean(days\_between\_purchases, na.rm = TRUE)  
  
# Print the result  
cat("Average days between consecutive purchases:",  
 average\_days\_between\_purchases, "days\n")

## Average days between consecutive purchases: NaN days

#10 In the retail sector, it is very important to understand the return rate of #the goods purchased by customers. In this example, we can define this quantity, #simply, as the ratio of the number of transactions cancelled (regardless of the #transaction value) over the total number of transactions.With this definition, #what is the return rate for the French customers?

# Filtering the dataset for french customers  
French\_cstmrs = filter(Online\_Retail,Country=="France" )  
#Returnrate for the french customers  
Return\_rate = nrow(filter(French\_cstmrs,Quantity<1))/nrow(French\_cstmrs)  
Return\_rate

## [1] 0.01741264

#11 What is the product that has generated the highest revenue for the retailer?

#revenue of each product  
Prd\_revenue= Online\_Retail %>% group\_by(StockCode) %>% summarise(Sum\_trnsvalue = sum(TransactionValue))  
#Selecting the product with highest revenue  
Prd\_revenue[which.max(Prd\_revenue$Sum\_trnsvalue),]

## # A tibble: 1 × 2  
## StockCode Sum\_trnsvalue  
## <chr> <dbl>  
## 1 DOT 206245.

#12 How many unique customers are represented in the dataset? You can use #unique() and length() functions.

#Number of unique customers  
length(unique(Online\_Retail$CustomerID))

## [1] 4373

#Summary: \* A dataset is loaded from “Online\_Retail.csv” file in the beginning of the script.

* By grouping the data by ‘Country’, the script calculates the total number of transactions for each country and calculating its percentage of transactions out of the total. It also filters out countries with less than 1% of total transactions.
* The Transaction Value variable is created by multiplying the Quantity and the UnitPrice for each transaction. Each entry in this variable represents the total value of the transaction.
* An analysis of transaction value by country is performed by grouping the data by ‘Country’ and calculating the total transaction value per country. It filters and displays countries with transaction values exceeding $130000.
* An analysis of the invoiced date is performed by converting the variable ‘InvoiceDate’ to a date format for time and date analysis. The script considers various factors, such as weekdays, months, specific dates with high transaction numbers, and maintenance hours.
* An analysis of missing values is performed by the script which calculates the percentages of missing values for each variable.
* This report analyzes transactions with missing ‘CustomerID’ records by country, and determines the number of transactions with missing records.
* By calculating the average number of days between consecutive shopping visits, the script can provide insight into the frequency with which customers return to the website during their next shopping trip.