BUSINESS ANALYTICS ASSIGNMENT_2

ALLEN RICHARDS

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
```

```
library(magrittr)
library(zoo)
```

```
##
## Attaching package: 'zoo'
```

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

```
library(readr)
```

```
# Load the dataset
retail_data <- read.csv("D:/KSU SEM-1/BUSINESS ANALYTICS/Assignment_2/OnlineRetail.csv", head
er = TRUE, stringsAsFactors = FALSE)
```

Question 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

```
OnlineRetail <- read.csv("D:/KSU SEM-1/BUSINESS ANALYTICS/Assignment_2/OnlineRetail.csv")

Countries_count = OnlineRetail %>% group_by(Country) %>% count(Country)
Countries_pct = OnlineRetail %>% group_by(Country) %>% summarise(percent = 100* n()/nrow(Onli
neRetail))
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
```

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

```
# Create the new variable 'TransactionValue' by multiplying 'Quantity' and 'UnitPrice'
OnlineRetail$TransactionValue <- OnlineRetail$Quantity * OnlineRetail$UnitPrice

# Display the first few rows of the dataframe to verify the new variable
head(OnlineRetail)
```

```
##   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
##      InvoiceDate UnitPrice CustomerID        Country TransactionValue
## 1 12/1/2010 8:26      2.55      17850 United Kingdom            15.30
## 2 12/1/2010 8:26      3.39      17850 United Kingdom            20.34
## 3 12/1/2010 8:26      2.75      17850 United Kingdom            22.00
## 4 12/1/2010 8:26      3.39      17850 United Kingdom            20.34
## 5 12/1/2010 8:26      3.39      17850 United Kingdom            20.34
## 6 12/1/2010 8:26      7.65      17850 United Kingdom            15.30
```

Question 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

```
# Load the required libraries
library(dplyr)

# Breakdown of transaction values by countries
Trans_sum = OnlineRetail %>% 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
```

```
# Group and summarize the data by country, calculating the total transaction value
country_transaction_values <- OnlineRetail %>%
  group_by(Country) %>%
  summarise(Total_TransactionValue = sum(TransactionValue))

# Filter countries with a total transaction value exceeding £130,000
country_transaction_values_filtered <- country_transaction_values %>%
  filter(Total_TransactionValue > 130000)

# Print the result
print(country_transaction_values_filtered)
```

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

Question 4 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.

```
# First let’s convert ‘InvoiceDate’ into a POSIXlt object
Temp <- strptime(OnlineRetail$InvoiceDate, format = '%m/%d/%Y %H:%M', tz = 'GMT')

# Check the First few entries of the 'TEMP' variable
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"
```

```
# Let Separate date, day of the week, and hour components
OnlineRetail$New_Invoice_Date <- as.Date(Temp)
OnlineRetail$Invoice_Day_Week <- weekdays(OnlineRetail$New_Invoice_Date)
OnlineRetail$New_Invoice_Hour <- as.numeric(format(Temp, "%H"))
OnlineRetail$New_Invoice_Month <- as.numeric(format(Temp, "%m"))

# The Date objects have a lot of flexible functions. For example knowing two date values, the
object allows you to know the difference between the two dates in terms of the number days.

OnlineRetail$New_Invoice_Date[20000] - OnlineRetail$New_Invoice_Date[10]
```

```
## Time difference of 8 days
```

```
# Let Convert Dates to Days of the Week
OnlineRetail$Invoice_Day_Week <- weekdays(OnlineRetail$New_Invoice_Date)

# Let Extract hour component
OnlineRetail$New_Invoice_Hour <- as.numeric(format(Temp, "%H"))

# Let define the month as a seperate numeric variable
OnlineRetail$New_Invoice_Month <- as.numeric(format(Temp, "%m"))
```

(a) Let show the percentage of transaction (by number) by days of the week

```
# Assuming you have the 'Online_Retail' dataset with the 'Invoice_Day_Week' variable
# You can use the table() function to count the number of transactions for each day of the we
ek
day_of_week_counts <- table(OnlineRetail$Invoice_Day_Week)

# Calculate the total number of transactions
total_transactions <- sum(day_of_week_counts)

# Calculate the percentage for each day of the week
percentages <- (day_of_week_counts / total_transactions) * 100

# Create a data frame to display the results
result_df <- data.frame(DayOfWeek = names(day_of_week_counts),
                        Count = as.vector(day_of_week_counts),
                        Percentage = percentages)

# Display the result
print(result_df)
```

```
##   DayOfWeek  Count Percentage.Var1 Percentage.Freq
## 1    Friday  82193          Friday        15.16731
## 2    Monday  95111          Monday        17.55110
## 3    Sunday  64375          Sunday        11.87930
## 4  Thursday 103857        Thursday        19.16503
## 5   Tuesday 101808         Tuesday        18.78692
## 6 Wednesday  94565       Wednesday        17.45035
```

(b) Let Show the percentage of transactions (by transaction volume) by days of the week

```
# Assuming you have the 'OnlineRetail' dataset with the 'Invoice_Day_Week' variable
# You can use the aggregate() function to calculate the total transaction volume (sum of Quan
tity) for each day of the week

OnlineRetail %>% group_by(Invoice_Day_Week)%>% summarise(Volume.of.transaction=(sum(Transacti
onValue)))%>% mutate(Volume.of.transaction,'percent'=(Volume.of.transaction*100)/sum(Volume.o
f.transaction))
```

```
## # A tibble: 6 × 3
##   Invoice_Day_Week Volume.of.transaction percent
##   <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) Let Show the percentage of transactions (by transaction volume) by month of the year

```
OnlineRetail %>% group_by(New_Invoice_Month)%>%
summarise(Volume.By.Month=sum(TransactionValue))%>% mutate(Volume.By.Month,'Percent'=(Volume.
By.Month*100)/sum(Volume.By.Month))
```

```
## # A tibble: 12 × 3
##    New_Invoice_Month Volume.By.Month Percent
##                <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
```

Question : What was the date with the highest number of transactions from Australia?

```
# Assuming you have the 'Online_Retail' dataset with 'InvoiceDate' and 'Country' columns

# Filter the dataset to include only transactions from Australia
australia_data <- OnlineRetail[OnlineRetail$Country == "Australia", ]

# Create a table of counts for each unique InvoiceDate within Australia
date_counts <- table(australia_data$InvoiceDate)

# Find the date with the highest number of transactions
max_date <- names(date_counts[date_counts == max(date_counts)])

# Display the result
print(max_date)
```

```
## [1] "6/15/2011 13:37"
```

Question E : The company needs to shut down the website for two consecutive hours

for maintenance. What would be the hour of the day to start this so that the distribution is at minimum for the customers? The responsible IT team is available from 7:00 to 20:00 every day.

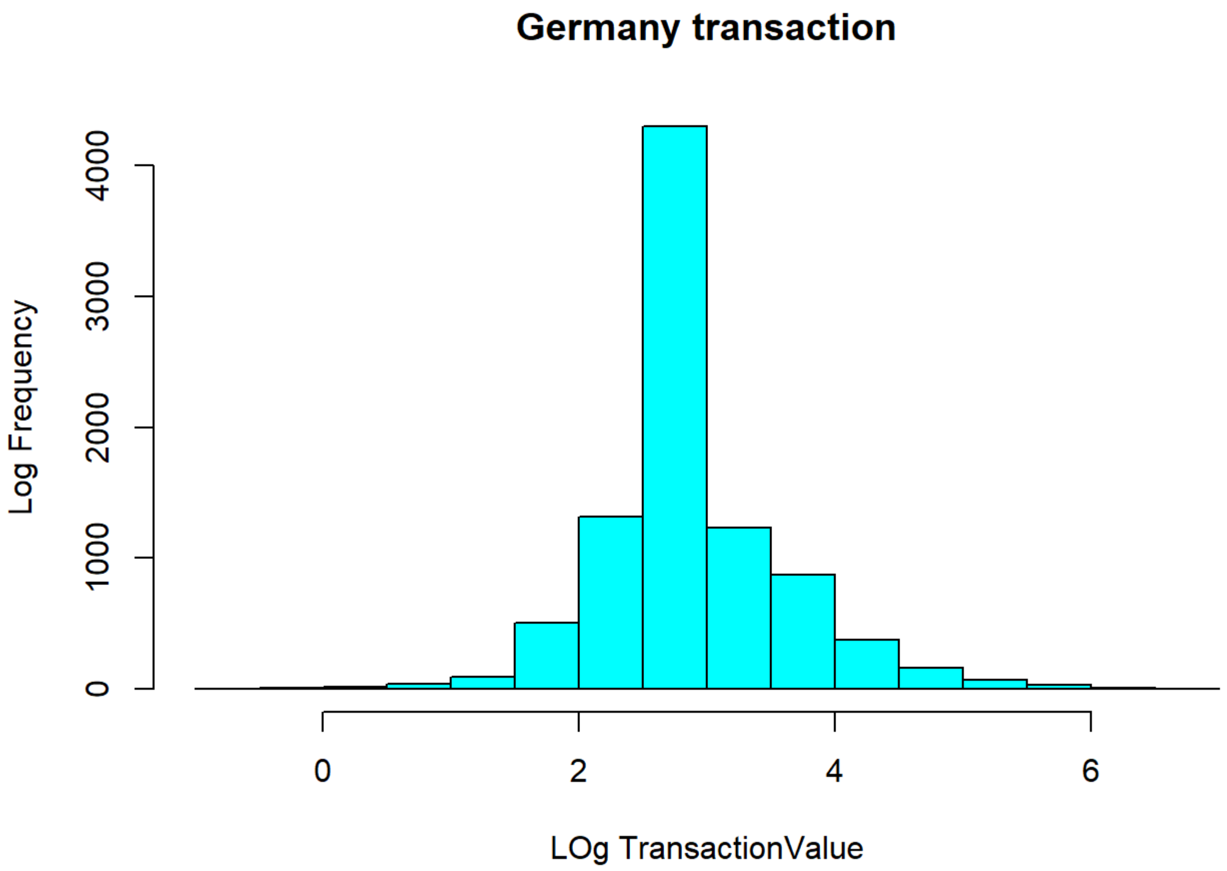
```
# Assuming you have the 'Online_Retail' dataset with 'New_Invoice_Hour' indicating the hour o
f transactions

# Filter transactions within the available time frame (7:00 to 20:00)
available_hours_data <- OnlineRetail[OnlineRetail$New_Invoice_Hour >= 7 & OnlineRetail$New_In
voice_Hour <= 20, ]

# Create a table of counts for each hour of the day
hourly_counts <- table(available_hours_data$New_Invoice_Hour)

# Calculate the rolling sum of transaction volumes for two consecutive hours
rolling_sum <- sapply(1:(length(hourly_counts) - 1), function(i) {
  sum(hourly_counts[i:(i + 1)])
})

# Find the starting hour with the minimum rolling sum
optimal_start_hour <- which(rolling_sum == min(rolling_sum))

# Display the result
print(paste("Optimal Start Hour for Maintenance:", optimal_start_hour + 6))  # Add 6 to get t
he actual hour (e.g., 7:00, 8:00, etc.)
```

```
## [1] "Optimal Start Hour for Maintenance: 19"
```

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

```
hist(x=log(OnlineRetail$TransactionValue[OnlineRetail$Country=="Germany"]),xlab = "LOg Transa
ctionValue",col = 'cyan' ,main = 'Germany transaction',ylab = 'Log Frequency')
```

```
## Warning in log(OnlineRetail$TransactionValue[OnlineRetail$Country ==
## "Germany"]): NaNs produced
```



#

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

```
# Let the data summarise the customer transaction
Data123<- OnlineRetail %>% group_by(CustomerID)%>%
summarise(CustomerTransaction = n())%>% filter(CustomerID != "NA")%>% filter(CustomerTransaction ==max(CustomerTransaction) )

print(paste('The customerID had the highest number of transactions is',Data123$CustomerID,'with max transaction of ',Data123$CustomerTransaction))
```

```
## [1] "The customerID had the highest number of transactions is 17841 with max transaction of  7983"
```

```
Data234<- OnlineRetail %>% group_by(CustomerID)%>%
summarise(total.transaction.by.each.customer = sum(TransactionValue))%>%
arrange(desc(total.transaction.by.each.customer))%>%
filter(CustomerID != "NA")%>% filter(total.transaction.by.each.customer ==max(total.transaction.by.each.customer) )

# Calculate the total transaction amount with customer ID
print(paste('Most valuable customerID is',Data234$CustomerID,'with total transaction Amount $',Data234$total.transaction.by.each.customer))
```

```
## [1] "Most valuable customerID is 14646 with total transaction Amount $ 279489.02"
```

Question 7 - Calculate the percentage of missing values for each variable in the dataset

```
# Assuming you have the 'OnlineRetail' dataset

# Calculate the percentage of missing values for each variable
missing_percentages <- sapply(OnlineRetail, function(col) {
  missing_count <- sum(is.na(col))
  missing_percentage <- (missing_count / length(col)) * 100
  return(missing_percentage)
})

# Create a data frame to display the missing value percentages
missing_df <- data.frame(Variable = names(missing_percentages), Percentage = missing_percenta
ges)

# Display the result
print(missing_df)
```

```
##                             Variable Percentage
## InvoiceNo                  InvoiceNo    0.00000
## StockCode                  StockCode    0.00000
## Description              Description    0.00000
## Quantity                    Quantity    0.00000
## InvoiceDate              InvoiceDate    0.00000
## UnitPrice                  UnitPrice    0.00000
## CustomerID                CustomerID   24.92669
## Country                      Country    0.00000
## TransactionValue    TransactionValue    0.00000
## New_Invoice_Date    New_Invoice_Date    0.00000
## Invoice_Day_Week    Invoice_Day_Week    0.00000
## New_Invoice_Hour    New_Invoice_Hour    0.00000
## New_Invoice_Month New_Invoice_Month    0.00000
```

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

```
# Assuming you have the 'Online_Retail' dataset

# Filter the dataset to include only transactions with missing CustomerID
missing_customerID_data <- OnlineRetail[is.na(OnlineRetail$CustomerID), ]

# Create a table of counts for missing CustomerID transactions by country
missing_customerID_counts <- table(missing_customerID_data$Country)

# Display the result
print(missing_customerID_counts)
```

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

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

```
# Calculate the time difference between consecutive shopping visits for each customer
Average<- OnlineRetail %>% group_by(CustomerID)%>%
summarise(difference.in.consecutivedays= diff(New_Invoice_Date))%>%

# Convert 'CONSECUTIVEDAYS' to a DIFFERENCE
filter(difference.in.consecutivedays>0)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## ℹ Please use `reframe()` instead.
## ℹ When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

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

```
# Display the result
print(paste('The average  number  of  days  between  consecutive  shopping is',mean(Average$d
ifference.in.consecutivedays)))
```

```
## [1] "The average  number  of  days  between  consecutive  shopping is 38.4875"
```

Question 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. Page 4 With this definition, what is the return rate for the French customers?

```
# Assuming you have the 'Online_Retail' dataset

# Filter the dataset to include only transactions from French customers
french_data <- OnlineRetail[OnlineRetail$Country == "France", ]

# Calculate the total number of transactions for French customers
total_transactions_french <- nrow(french_data)

# Calculate the number of canceled transactions for French customers
canceled_transactions_french <- sum(substr(french_data$InvoiceNo, 1, 1) == "C")

# Calculate the return rate for French customers
return_rate_french <- canceled_transactions_french / total_transactions_french

# Display the result as a percentage
return_rate_percentage <- return_rate_french * 100
print(paste("Return Rate for French Customers:", round(return_rate_percentage, 2), "%"))
```

```
## [1] "Return Rate for French Customers: 1.74 %"
```

Question - 11 What is the product that has generated the highest revenue for the retailer? (i.e. item with the highest total sum of ‘TransactionValue’).

```
# Assuming you have the 'Online_Retail' dataset with 'StockCode', 'Quantity', and 'UnitPrice' columns

# Calculate the 'TransactionValue' for each transaction
OnlineRetail$TransactionValue <- OnlineRetail$Quantity * OnlineRetail$UnitPrice

# Aggregate the total revenue generated by each product
product_revenue <- aggregate(TransactionValue ~ StockCode, data = OnlineRetail, FUN = sum)
highest_revenue_product <- product_revenue[which.max(product_revenue$TransactionValue), "StockCode"]
print(paste("Product with the Highest Revenue:", highest_revenue_product))
```

```
## [1] "Product with the Highest Revenue: DOT"
```

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

```
# Assuming you have the 'Online_Retail' dataset with a 'CustomerID' column

# Extract unique customer IDs
unique_customers <- unique(OnlineRetail$CustomerID)

# Calculate the number of unique customers
num_unique_customers <- length(unique_customers)

# Display the result
print(paste("Number of Unique Customers:", num_unique_customers))
```

```
## [1] "Number of Unique Customers: 4373"
```