BA64036_Assignment2

Ruthvick Bulagakula

2023-10-12

```
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(lubridate)
```

```
##
## Attaching package: 'lubridate'

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

```
data = read.csv("C:/Users/ruthvick/Desktop/Rhistory/Online_Retail.csv")

head(data)
```

```
##   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
## 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
```

```
tail(data)
```

```
##        InvoiceNo StockCode                      Description Quantity
## 541904    581587     23256      CHILDRENS CUTLERY SPACEBOY         4
## 541905    581587     22613      PACK OF 20 SPACEBOY NAPKINS        12
## 541906    581587     22899     CHILDREN'S APRON DOLLY GIRL         6
## 541907    581587     23254    CHILDRENS CUTLERY DOLLY GIRL         4
## 541908    581587     23255 CHILDRENS CUTLERY CIRCUS PARADE         4
## 541909    581587     22138    BAKING SET 9 PIECE RETROSPOT         3
##             InvoiceDate UnitPrice CustomerID Country
## 541904 12/9/2011 12:50      4.15      12680  France
## 541905 12/9/2011 12:50      0.85      12680  France
## 541906 12/9/2011 12:50      2.10      12680  France
## 541907 12/9/2011 12:50      4.15      12680  France
## 541908 12/9/2011 12:50      4.15      12680  France
## 541909 12/9/2011 12:50      4.95      12680  France
```

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. (10% of total points)

```
country_count = data %>% group_by(Country) %>% count(Country)

country_count
```

```
## # A tibble: 38 x 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
```

```
country_percentage = data %>% group_by(Country) %>% summarise(Percentage = 100* n()/nrow(data))

country_percentage
```

```
## # A tibble: 38 x 2
##    Country         Percentage
##    <chr>                <dbl>
##  1 Australia          0.232
##  2 Austria            0.0740
```

```
##  3 Bahrain            0.00351
##  4 Belgium            0.382
##  5 Brazil             0.00591
##  6 Canada             0.0279
##  7 Channel Islands    0.140
##  8 Cyprus             0.115
##  9 Czech Republic     0.00554
## 10 Denmark            0.0718
## # i 28 more rows
```

```
filter = filter(country_percentage, Percentage>1)

filter
```

```
## # A tibble: 4 x 2
##   Country        Percentage
##   <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. (10% of total points)

```
data$TransactionValue = data$Quantity * data$UnitPrice

head(data$TransactionValue)
```

```
## [1] 15.30 20.34 22.00 20.34 20.34 15.30
```

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. (15% of total points)

```
country_transaction = data %>% group_by(Country) %>% summarise(Sum_Transaction=sum(TransactionValue))


filtered_country_transaction = filter(country_transaction, country_transaction$Sum_Transaction>13000)

filtered_country_transaction
```

```
## # A tibble: 17 x 2
##    Country         Sum_Transaction
##    <chr>                     <dbl>
##  1 Australia               137077.
##  2 Belgium                  40911.
##  3 Channel Islands          20086.
##  4 Denmark                  18768.
##  5 EIRE                    263277.
```

```
##  6 Finland                  22327.
##  7 France                  197404.
##  8 Germany                 221698.
##  9 Italy                    16891.
## 10 Japan                    35341.
## 11 Netherlands             284662.
## 12 Norway                   35163.
## 13 Portugal                 29367.
## 14 Spain                    54775.
## 15 Sweden                   36596.
## 16 Switzerland              56385.
## 17 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. Click here for more information. First let's convert 'InvoiceDate' into a POSIXlt object: Temp=strptime(Online_Retail$InvoiceDate, format ='$ New_Invoice_Date <- as.Date(Temp) 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. Try this: Online_Retail$New_Invoice_Date[20000] - Online_Retail$New_Invoice_Date[10] Also we can convert dates to days of the week. Let's define a new variable for that Online_Retail$Invoice_Day_Week = weekdays(Online_Retail$New_Invoice_Date) For the Hour, let's just take the hour (ignore the minute) and convert into a normal numerical value: Online_Retail$New_Invoice_Hour = as.numeric(format(Temp, "New_Invoice_Month = as.numeric(format(Temp, "%m"))

```
Temp = strptime(data$InvoiceDate, format = '%m/%d/%Y %H:%M', tz = 'GMT')
data$New_Invoice_Date = as.Date(Temp)
data$Invoice_Day_Week = weekdays(data$New_Invoice_Date)
data$New_Invoice_Hour = as.numeric(format(Temp, "%H"))
data$New_Invoice_Month = as.numeric(format(Temp, "%m"))
```

```
head(data)
```

```
##   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
##   New_Invoice_Date Invoice_Day_Week New_Invoice_Hour New_Invoice_Month
## 1       2010-12-01        Wednesday               8                12
## 2       2010-12-01        Wednesday               8                12
## 3       2010-12-01        Wednesday               8                12
```

```
## 4       2010-12-01        Wednesday                 8                 12
## 5       2010-12-01        Wednesday                 8                 12
## 6       2010-12-01        Wednesday                 8                 12
```

a. Show the percentage of transactions (by numbers) by days of the week (extra 1% of total points)

```
day_per = data %>% group_by(Invoice_Day_Week) %>% summarise(Percentage = 100* n()/nrow(data))

day_per
```

```
## # A tibble: 6 x 2
##   Invoice_Day_Week Percentage
##   <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
```

b. Show the percentage of transactions (by transaction volume) by days of the week (extra 1% of total points)

```
day_transaction = data %>% group_by(Invoice_Day_Week) %>% summarise(Percentage= sum(TransactionValue))

day_transaction_per = 100*(day_transaction$Percentage)/sum(day_transaction$Percentage)

day_transaction$Percentage = day_transaction_per

day_transaction
```

```
## # A tibble: 6 x 2
##   Invoice_Day_Week Percentage
##   <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
```

c. Show the percentage of transactions (by transaction volume) by month of the year (extra 2% of total points)

```
month_transaction = data %>%
  group_by(New_Invoice_Month) %>%
  summarise(Monthly_Transaction = sum(TransactionValue))

month_transaction$New_Invoice_Month = month(month_transaction$New_Invoice_Month, label = TRUE)

month_transaction_per = 100 * (month_transaction$Monthly_Transaction) / sum(month_transaction$Monthly_T
```

```
month_transaction$Monthly_Transaction = month_transaction_per

month_transaction
```

```
## # A tibble: 12 x 2
##    New_Invoice_Month Monthly_Transaction
##    <ord>                           <dbl>
##  1 Jan                              5.74
##  2 Feb                              5.11
##  3 Mar                              7.01
##  4 Apr                              5.06
##  5 May                              7.42
##  6 Jun                              7.09
##  7 Jul                              6.99
##  8 Aug                              7.00
##  9 Sep                             10.5
## 10 Oct                             11.0
## 11 Nov                             15.0
## 12 Dec                             12.1
```

d. What was the date with the highest number of transactions from Australia? (extra 2% of total points)

```
australia = data %>%
filter(Country=='Australia')%>% group_by(New_Invoice_Date)%>%
summarise(Number=sum(Quantity),amount=sum(TransactionValue))%>%
arrange(desc(Number))

date_highest_transactions = australia$New_Invoice_Date[which.max(table(australia$New_Invoice_Date))]

print(paste("Date which has highest number of transations from australia is", date_highest_transactions)
```

```
## [1] "Date which has highest number of transations from australia is 2011-06-15"
```

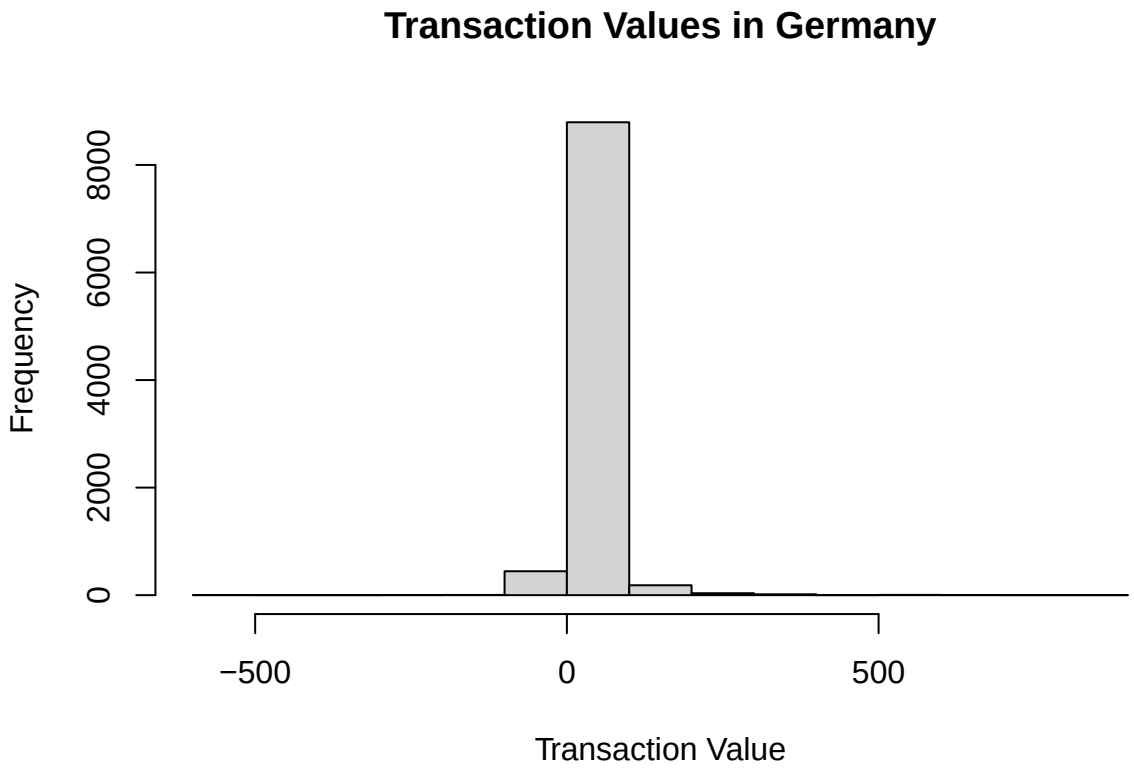
e. The company needs to shut down the website for two consecutive hours for maintenance. What would be the hour of the day to start this so that the distribution is at minimum for the customers? The responsible IT team is available from 7:00 to 20:00 every day. (extra 4% of total points)

```
hour_transaction_data = data %>% group_by(New_Invoice_Hour) %>% summarize(Count = n()) %>% filter(New_In

optimal_start_hour = hour_transaction_data %>% slice(1) %>% pull(New_Invoice_Hour)

optimal_start_hour2 = ifelse(optimal_start_hour == 20, optimal_start_hour - 1, optimal_start_hour + 1)

start_of_maintenance = paste(optimal_start_hour, ":00", sep = "")
end_of_maintenance = paste(optimal_start_hour2, ":00", sep = "")

cat("Start of maintaince should be",start_of_maintenance,"P.M and end of maintance should be",end_of_mai
```

```
## Start of maintaince should be 7:00 P.M and end of maintance should be 8:00 P.M
```

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

```r
germany = data[data$Country == "Germany", ]

hist(germany$TransactionValue, main = "Transaction Values in Germany", xlab = "Transaction Value")
```



6. Which customer had the highest number of transactions? Which customer is most valuable (i.e. highest total sum of transactions)? (15% of total points)

```r
customer_highest_transaction =data  %>% group_by(CustomerID)%>%
summarise(CustomerTransaction = n())%>% filter(CustomerID != "NA")%>% filter(CustomerTransaction ==max(

cat("Customer who had highest number of transactions is",customer_highest_transaction$CustomerID)
```

```
## Customer who had highest number of transactions is 17841
```

```r
customer_transaction_total = data %>% group_by(CustomerID)%>%
summarise(total.transaction.by.each.customer = sum(TransactionValue))%>% arrange(desc(total.transaction
filter(CustomerID != "NA")%>% filter(total.transaction.by.each.customer ==max(total.transaction.by.each

cat("\nCustomer who is most valuable is", customer_transaction_total$CustomerID)
```

```
##
## Customer who is most valuable is 14646
```

7. Calculate the percentage of missing values for each variable in the dataset (5% of total points). Hint colMeans():

```
missing_percentage = colMeans(is.na(data)) * 100

missing_percentage
```

```
##         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? (10 % of total points)

```
missing_customer_transaction = data[is.na(data$CustomerID), ]
missing_customer_counts_country = table(missing_customer_transaction$Country)

missing_customer_counts_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) (5% of total points!) Hint: 1. A close approximation is also acceptable and you may find diff() function useful.

```
time_diff = data %>%
  group_by(CustomerID) %>%
  mutate(difference.in.consecutivedays = c(0, diff(New_Invoice_Date))) %>%
  filter(difference.in.consecutivedays > 0) %>%
  ungroup()

average_time_diff = mean(time_diff$difference.in.consecutivedays)

cat("On an average customer customers comeback to the websire for their next shopping is after",average_
```

```
## On an average customer customers comeback to the websire for their next shopping is after 38.4875 day
```

10. n 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? (10% of total points). Consider the cancelled transactions as those where the 'Quantity' variable has a negative value.

```r
french_data = filter(data, Country == "France" )

return_rate = nrow(filter(french_data, Quantity<1)) / nrow(french_data)

cat("Return rate of french customers is",return_rate)
```

```
## Return rate of french customers is 0.01741264
```

11. What is the product that has generated the highest revenue for the retailer? (i.e. item with the highest total sum of 'TransactionValue'). (10% of total points)

```r
product_revenue = data %>% group_by(StockCode) %>% summarise(sum = sum(TransactionValue))

highest_stock = product_revenue[which.max(product_revenue$sum), ]

cat(highest_stock$StockCode, "has highest revenue for the retailer which is around", highest_stock$sum)
```

```
## DOT has highest revenue for the retailer which is around 206245.5
```

12. How many unique customers are represented in the dataset? You can use unique() and length()functions. (10% of total points)

```r
unique_customers = length(unique(data$CustomerID))

cat("Unique customer in dataset is",unique_customers)
```

```
## Unique customer in dataset is 4373
```