# Customer Segmentation. Application of Unsupervised Learning Methods for Trend Exploration

*by Ketao Li, Kush Halani, Josue Romain, Juan Peña*

**Abstract** Customer segmentation is the process of dividing customers into groups based on common characteristics so companies can market to each group effectively and appropriately.

### Background

Without a deep understanding of how a company's best current customers are segmented, a business often lacks the market focus needed to allocate and spend its precious human and capital resources efficiently. Furthermore, a lack of best current customer segment focus can cause diffused go-to-market and product development strategies that hamper a company's ability to fully engage with its target segments. Together, all of those factors can ultimately impede a company's growth.
RFM (recency, frequency, monetary) analysis is a marketing technique used to determine quantitatively which customers are the best ones by examining how recently a customer has purchased (recency), how often they purchase (frequency), and how much the customer spends (monetary).

### Objective

The objective of customers segment according to their purchase history, is to turn them into loyal customers by recommending products of their choice.

## Data Analysis

Typically e-commerce datasets are proprietary and consequently hard to find among publicly available data. However, The UCI Machine Learning Repository has made this dataset containing actual transactions from 2010 and 2011.The data set used for this research contains 540k of transaction from UK retailer. The data has been sourced from Kaggle.

### Data Dictionary

| Column Name | Column Description |
|---|---|
| InvoiceNo | Unique ID to identify each Invoice |
| StockCode | Unique ID for each item in stock |
| Description | A short description for each item |
| Quantity | Number of items bought |
| InvoiceDate | Invoice Date |
| UnitPrice | The price of each item |
| CustomerID | Unique ID for each custumer |
| Country | The country were the custumer lives |

### Data Exploration

Firstly we are going to load and examine content and statistics of the data set

```
data = read.csv("../data/data.csv", header = T,
              na.strings = c("NA","","#NA"),sep=",")
```

**Table 2:** Online Retail Dataset Summary

| No | Variable | Stats / Values | Freqs (% of Valid) | Missing |
|---|---|---|---|---|
| 1 | InvoiceNo [factor] | 1. 536365<br>2. 536366<br>3. 536367<br>[ 25897 others ] | 7 ( 0.0%)<br>2 ( 0.0%)<br>12 ( 0.0%)<br>541888 (100.0%) | 0 (0%) |
| 2 | StockCode [factor] | 1. 10002<br>2. 10080<br>3. 10120<br>[ 4067 others ] | 73 ( 0.0%)<br>24 ( 0.0%)<br>30 ( 0.0%)<br>541782 (100.0%) | 0 (0%) |
| 3 | Description [factor] | 1. ·4 PURPLE FLOCK DINNER CA<br>2. ·50'S CHRISTMAS GIFT BAG<br>3. ·DOLLY GIRL BEAKER<br>[ 4220 others ] | 41 ( 0.0%)<br>130 ( 0.0%)<br>181 ( 0.0%)<br>540103 (99.9%) | 1454 (0.27%) |
| 4 | Quantity [integer] | Mean (sd) : 9.6 (218.1)<br>min < med < max:<br>-80995 < 3 < 80995<br>IQR (CV) : 9 (22.8) | 722 distinct values | 0 (0%) |
| 5 | InvoiceDate [factor] | 1. 1/10/2011 10:04<br>2. 1/10/2011 10:07<br>3. 1/10/2011 10:08<br>[ 23257 others ] | 1 ( 0.0%)<br>1 ( 0.0%)<br>1 ( 0.0%)<br>541906 (100.0%) | 0 (0%) |
| 6 | UnitPrice [numeric] | Mean (sd) : 4.6 (96.8)<br>min < med < max:<br>-11062.1 < 2.1 < 38970<br>IQR (CV) : 2.9 (21) | 1630 distinct values | 0 (0%) |
| 7 | CustomerID [integer] | Mean (sd) : 15287.7 (1713.6)<br>min < med < max:<br>12346 < 15152 < 18287<br>IQR (CV) : 2838 (0.1) | 4372 distinct values | 135080 (24.93%) |
| 8 | Country [factor] | 1. Australia<br>2. Austria<br>3. Bahrain<br>[ 35 others ] | 1259 ( 0.2%)<br>401 ( 0.1%)<br>19 ( 0.0%)<br>540230 (99.7%) | 0 (0%) |

From the above summary, we can find that there are some negative values for Quantity and UnitPrice.These values don't make sense, so we'll delete them directly.

```
customerData <- data %>%
  mutate(Quantity = replace(Quantity, Quantity<=0, NA),
         UnitPrice = replace(UnitPrice, UnitPrice<=0, NA))


customerData = customerData %>%filter(complete.cases(.))
```

##Missing data

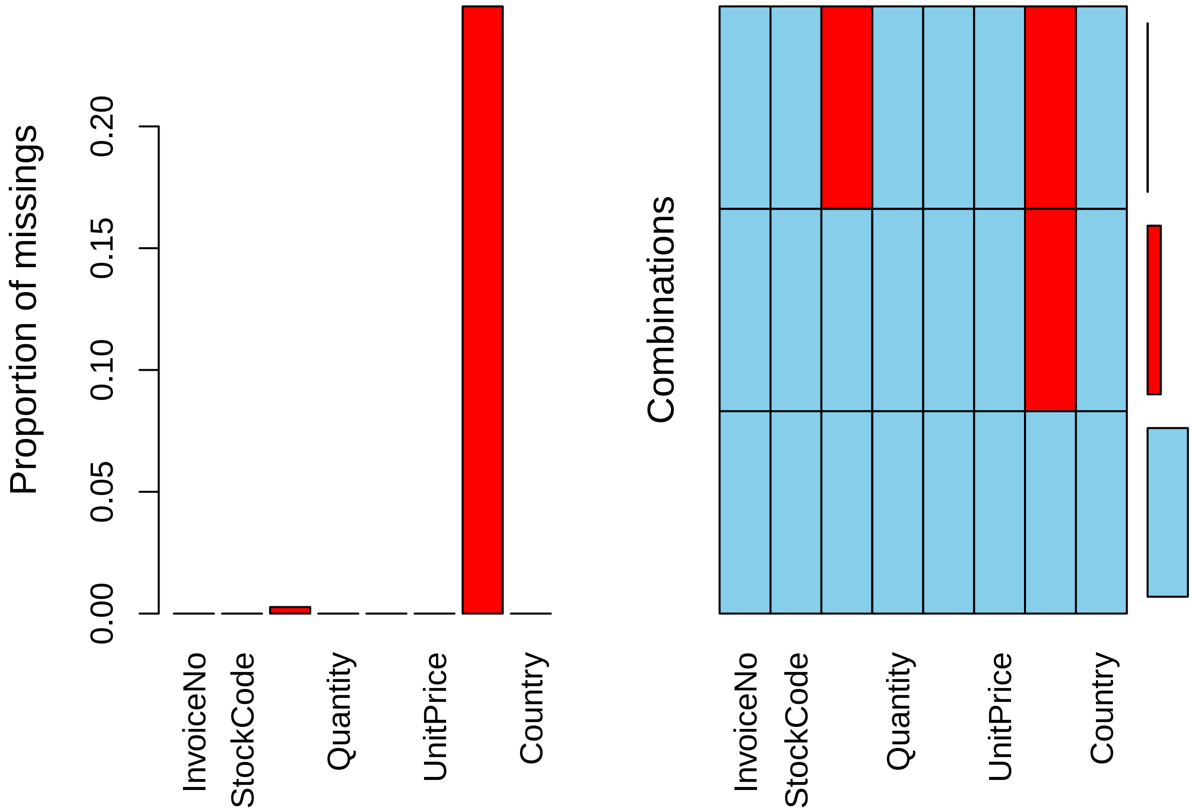



**Figure 1:** Missing data

```
summary(a)
```

```
 Missings per variable:
    Variable  Count
   InvoiceNo      0
   StockCode      0
 Description   1454
    Quantity      0
 InvoiceDate      0
   UnitPrice      0
  CustomerID 135080
     Country      0

 Missings in combinations of variables:
    Combinations  Count    Percent
 0:0:0:0:0:0:0:0 406829 75.0733057
 0:0:0:0:0:0:1:0 133626 24.6583836
 0:0:1:0:0:0:1:0   1454  0.2683107
```

There are some missing data for CustomerID and Desciption, we just remove them directly considering we have enough data.

**Table 3:** Online Retail Dataset Summary

| No | Variable | Stats / Values | Freqs (% of Valid) | Missing |
|---|---|---|---|---|
| 1 | InvoiceNo<br>[factor] | 1. 536365<br>2. 536366<br>3. 536367<br>[ 25897 others ] | 7 ( 0.0%)<br>2 ( 0.0%)<br>12 ( 0.0%)<br>397863 (100.0%) | 0<br>(0%) |

| No | Variable | Stats / Values | Freqs (% of Valid) | Missing |
|---|---|---|---|---|
| 2 | StockCode [factor] | 1. 10002<br>2. 10080<br>3. 10120<br>[ 4067 others ] | 49 ( 0.0%)<br>21 ( 0.0%)<br>30 ( 0.0%)<br>397784 (100.0%) | 0 (0%) |
| 3 | Description [factor] | 1. ·4 PURPLE FLOCK DINNER CA<br>2. ·50'S CHRISTMAS GIFT BAG<br>3. ·DOLLY GIRL BEAKER<br>[ 4220 others ] | 39 ( 0.0%)<br>109 ( 0.0%)<br>138 ( 0.0%)<br>397598 (99.9%) | 0 (0%) |
| 4 | Quantity [integer] | Mean (sd) : 13 (179.3)<br>min < med < max:<br>1 < 6 < 80995<br>IQR (CV) : 10 (13.8) | 301 distinct values | 0 (0%) |
| 5 | InvoiceDate [factor] | 1. 1/10/2011 10:04<br>2. 1/10/2011 10:07<br>3. 1/10/2011 10:08<br>[ 23257 others ] | 0 ( 0.0%)<br>0 ( 0.0%)<br>0 ( 0.0%)<br>397884 (100.0%) | 0 (0%) |
| 6 | UnitPrice [numeric] | Mean (sd) : 3.1 (22.1)<br>min < med < max:<br>0 < 2 < 8142.8<br>IQR (CV) : 2.5 (7.1) | 440 distinct values | 0 (0%) |
| 7 | CustomerID [integer] | Mean (sd) : 15294.4 (1713.1)<br>min < med < max:<br>12346 < 15159 < 18287<br>IQR (CV) : 2826 (0.1) | 4338 distinct values | 0 (0%) |
| 8 | Country [factor] | 1. Australia<br>2. Austria<br>3. Bahrain<br>[ 35 others ] | 1182 ( 0.3%)<br>398 ( 0.1%)<br>17 ( 0.0%)<br>396287 (99.6%) | 0 (0%) |

## Data Preparation

We need do some some data transformation and add one new variant total.

```
customerData <- customerData %>%
  mutate( InvoiceDate=as.Date(InvoiceDate, '%m/%d/%Y %H:%M'),
          CustomerID=as.factor(CustomerID),
          Country = as.character(Country))

customerData <- customerData %>%
  mutate(total = Quantity*UnitPrice)

glimpse(customerData)

Observations: 397,884
Variables: 9
$ InvoiceNo   <fct> 536365, 536365, 536365, 536365, 536365, 536365, 536365,...
$ StockCode   <fct> 85123A, 71053, 84406B, 84029G, 84029E, 22752, 21730, 22...
$ Description <fct> WHITE HANGING HEART T-LIGHT HOLDER, WHITE METAL LANTERN...
$ Quantity    <int> 6, 6, 8, 6, 6, 2, 6, 6, 6, 32, 6, 6, 8, 6, 6, 3, 2, 3, ...
$ InvoiceDate <date> 2010-12-01, 2010-12-01, 2010-12-01, 2010-12-01, 2010-1...
$ UnitPrice   <dbl> 2.55, 3.39, 2.75, 3.39, 3.39, 7.65, 4.25, 1.85, 1.85, 1...
$ CustomerID  <fct> 17850, 17850, 17850, 17850, 17850, 17850, 17850, 17850,...
$ Country     <chr> "United Kingdom", "United Kingdom", "United Kingdom", "...
$ total       <dbl> 15.30, 20.34, 22.00, 20.34, 20.34, 15.30, 25.50, 11.10,...
```

**Descriptive Features.** *Description* is free-text features that might provide additional insights about the customer shopping. We are going to take a close look at this feature and decide if we could utilize it.

Lets' begin with the *Description*


**Figure 2:** Most Common Words in Description

From the word cloud, we can get some highly frequently used words such as set,red,box,lunch,blue,box,paper,glass.

Unfortunately *Description* feature does not provide more knowledge to what the others features already supply. Thus it will be dropped.

Country gives information about the country were the customer lives.

```
length(unique(data$Country))
```

```
[1] 38
```

The custumers are from 38 different countries. Lets visualize this.

```
33 codes from your data successfully matched countries in the map
5 codes from your data failed to match with a country code in the map
210 codes from the map weren't represented in your data
```

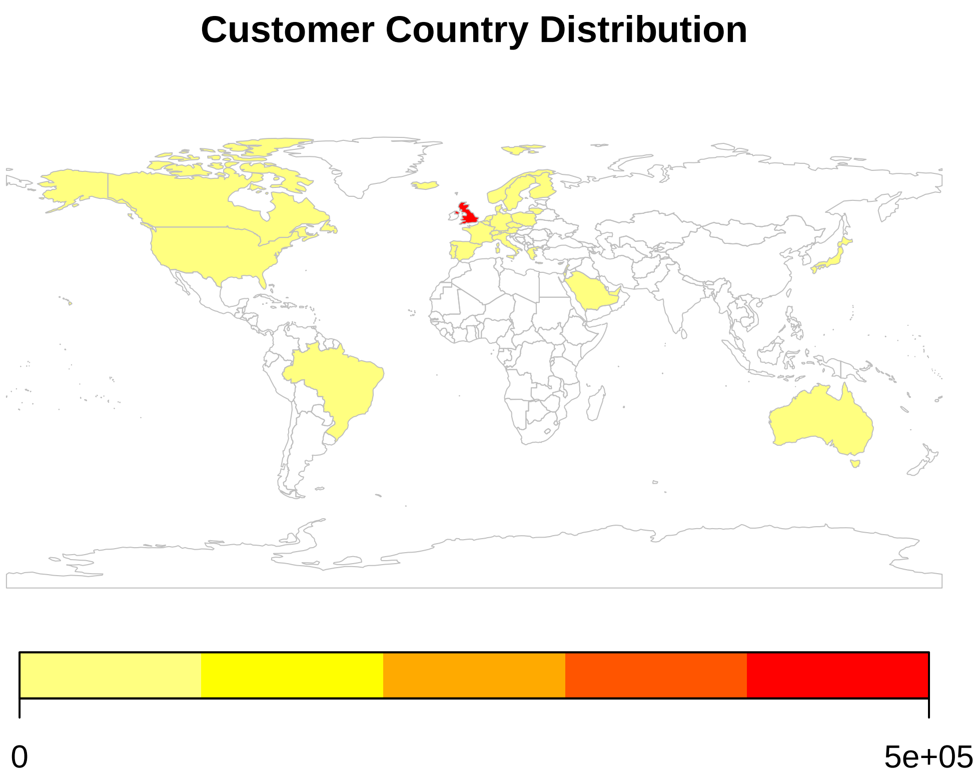



**Figure 3:** customer country distribution

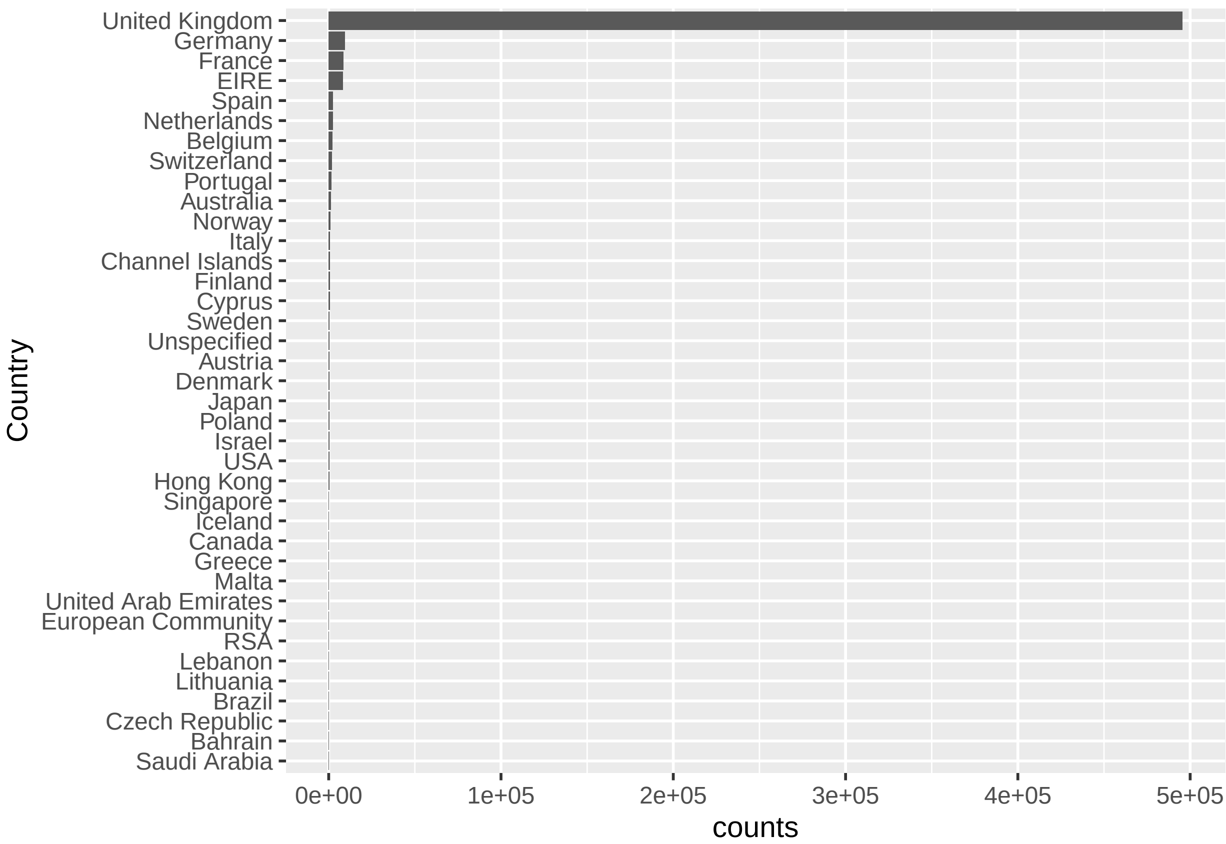



**Figure 4:** Countries Description

```
data1 <- data
data1$InvoiceDate <- mdy_hm(data$InvoiceDate)
```

```
head(data1)
```

```
  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 2010-12-01 08:26:00      2.55      17850 United Kingdom
2 2010-12-01 08:26:00      3.39      17850 United Kingdom
3 2010-12-01 08:26:00      2.75      17850 United Kingdom
4 2010-12-01 08:26:00      3.39      17850 United Kingdom
5 2010-12-01 08:26:00      3.39      17850 United Kingdom
6 2010-12-01 08:26:00      7.65      17850 United Kingdom
```

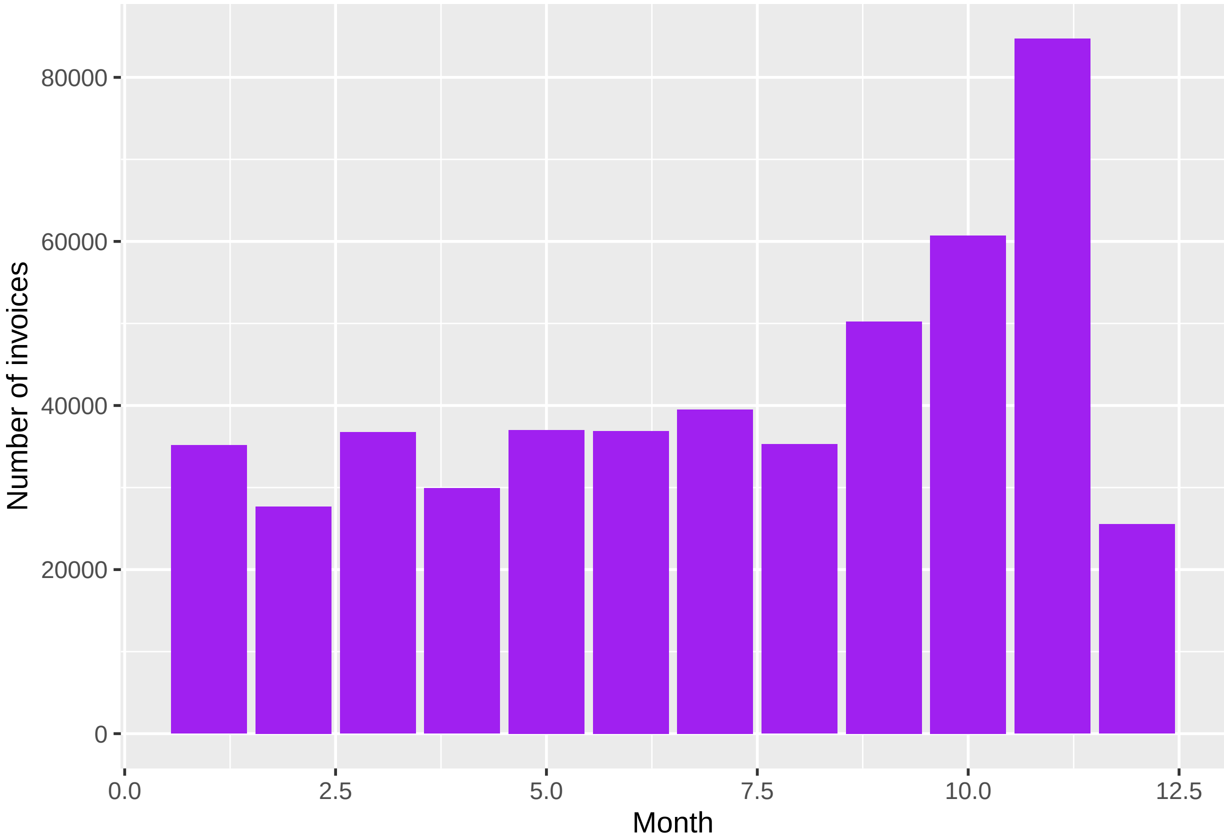
We now have the data transformed into datetime data. From the variable InvoiceDate we can extract the year, month, day and time.

```
data1$InvoiceYear <- year(data1$InvoiceDate)
data1$InvoiceMonth <- month(data1$InvoiceDate)
data1$InvoiceWeekday <- wday(data1$InvoiceDate)
data1$InvoiceHour <- hour(data1$InvoiceDate)
```

Here we have the number of transactions per month for 2011.

```
timedata <- data1 %>%
  filter(InvoiceYear==2011) %>%
```

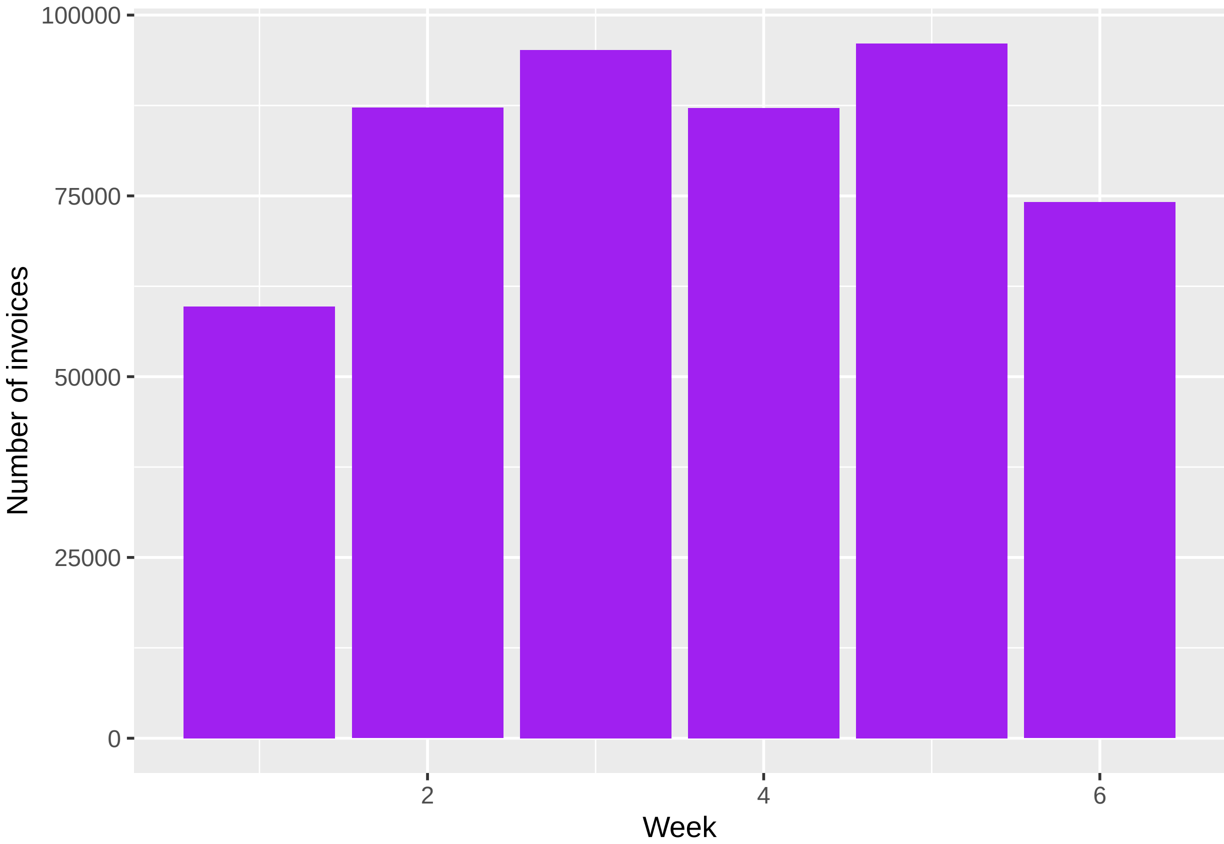
```
  count(InvoiceMonth)  #count the number of invoices per month for 2011

ggplot(timedata, aes(InvoiceMonth, n)) +  #plot the number of invoices per day
  geom_col(fill = "purple") +
  labs(x="Month", y="Number of invoices")
```




It seems that the number of transactions is rising from September and the highest in November. In december the lowest number of transactions is performed.

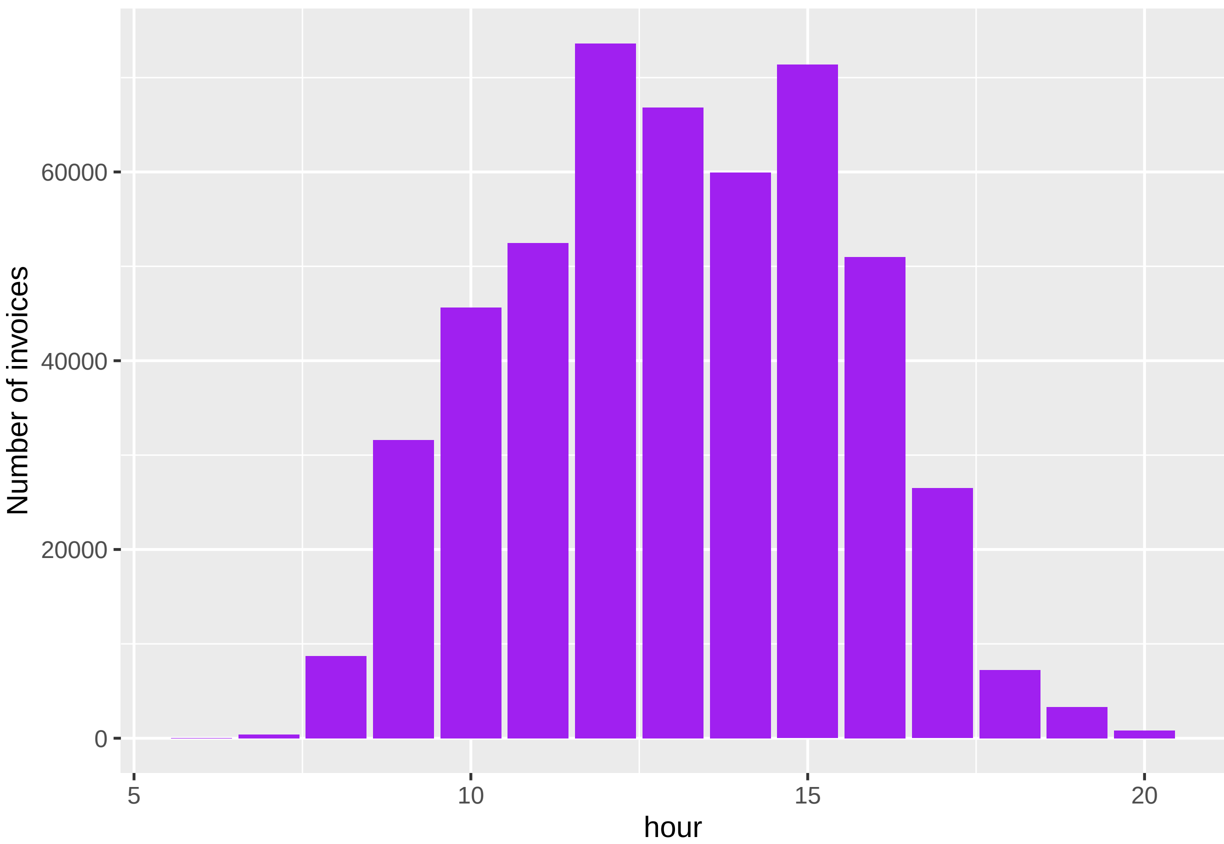
Lets explore which days are the most busy ones

```
timedata <- data1 %>%
  filter(InvoiceYear==2011) %>%
  count(InvoiceWeekday)

ggplot(timedata, aes(InvoiceWeekday, n)) +  #plot the number of invoices per day
    geom_col(fill = "purple") +
  labs(x="Week", y="Number of invoices")
```




Most transactions are placed on monday, tuesday, wednesday and thursday.

```
timedata <- data1 %>%
 filter(InvoiceYear==2011) %>%
  count(InvoiceHour)

ggplot(timedata, aes(InvoiceHour, n)) +  #plot the number of invoices per day
    geom_col(fill = "purple") +
  labs(x="hour", y="Number of invoices")
```




### Calculate RFM

To implement the RFM analysis, we need to take steps to get the rfm values:

1. Find the most recent date for each customer ID and calculate the days to the 2012-01-01, to get the recency data.
2. Calculate the quantity of transactions of a customer, to get the frequency data
3. Sum the amount of money a customer spent and divide it by frequency, to get the amount per transaction on average, that is the monetary data.

```
cd_RFM <- customerData %>%
  group_by(CustomerID) %>%
  summarise(recency=as.numeric(as.Date("2012-01-01")-max(InvoiceDate)),
            frequenci=n_distinct(InvoiceNo), monitery= sum(total)/n_distinct(InvoiceNo),
            country = max(Country)

            )

summary(cd_RFM)

head(cd_RFM)
```

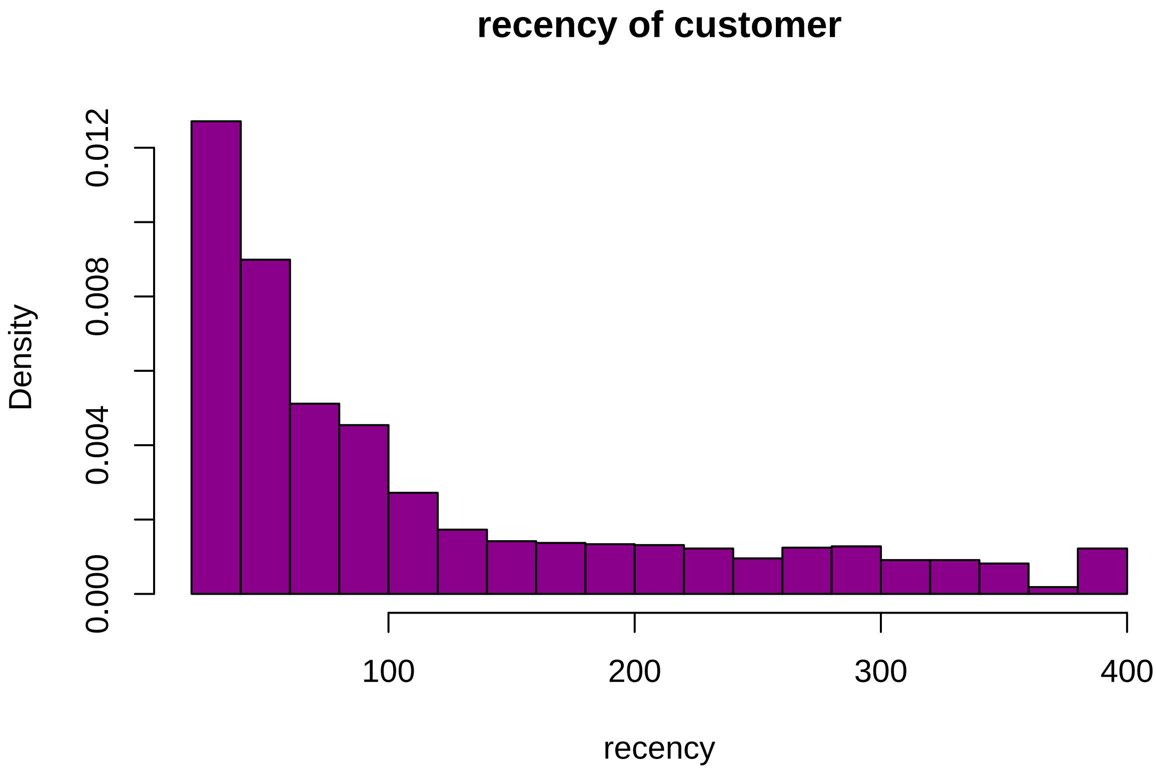
```
   CustomerID      recency         frequenci          monitery
 12346   :    1   Min.   : 23.0   Min.   :  1.000   Min.   :    3.45
 12347   :    1   1st Qu.: 40.0   1st Qu.:  1.000   1st Qu.:  178.62
 12348   :    1   Median : 73.0   Median :  2.000   Median :  293.90
 12349   :    1   Mean   :115.1   Mean   :  4.272   Mean   :  419.17
 12350   :    1   3rd Qu.:164.8   3rd Qu.:  5.000   3rd Qu.:  430.11
 12352   :    1   Max.   :396.0   Max.   :209.000   Max.   :84236.25
 (Other):4332
   country
 Length:4338
 Class :character
 Mode  :character
```

```
# A tibble: 6 x 5
  CustomerID recency frequenci monitery country
  <fct>        <dbl>     <int>    <dbl> <chr>
1 12346          348         1   77184. United Kingdom
2 12347           25         7     616. Iceland
3 12348           98         4     449. Finland
4 12349           41         1    1758. Italy
5 12350          333         1     334. Norway
6 12352           59         8     313. Norway
```

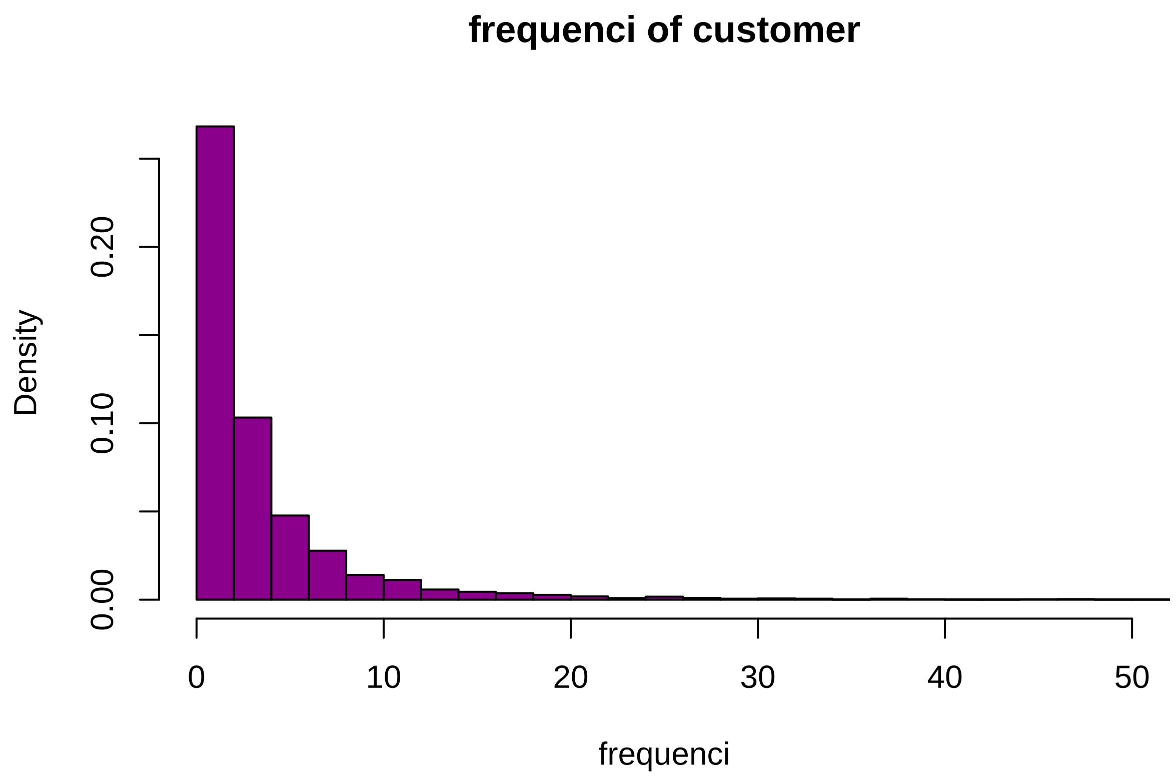
**Table 4:** Online Retail Dataset Summary

| No | Variable | Stats / Values | Freqs (% of Valid) | Missing |
|---|---|---|---|---|
| 1 | CustomerID<br>[factor] | 1. 12346<br>2. 12347<br>3. 12348<br>[ 4335 others ] | 1 ( 0.0%)<br>1 ( 0.0%)<br>1 ( 0.0%)<br>4335 (99.9%) | 0<br>(0%) |
| 2 | recency<br>[numeric] | Mean (sd) : 115.1 (100)<br>min < med < max:<br>23 < 73 < 396<br>IQR (CV) : 124.8 (0.9) | 304 distinct values | 0<br>(0%) |
| 3 | frequenci<br>[integer] | Mean (sd) : 4.3 (7.7)<br>min < med < max:<br>1 < 2 < 209<br>IQR (CV) : 4 (1.8) | 59 distinct values | 0<br>(0%) |
| 4 | monitery<br>[numeric] | Mean (sd) : 419.2 (1796.5)<br>min < med < max:<br>3.5 < 293.9 < 84236.2<br>IQR (CV) : 251.5 (4.3) | 4249 distinct values | 0<br>(0%) |
| 5 | country<br>[character] | 1. United Kingdom<br>2. Germany<br>3. France<br>[ 34 others ] | 3920 (90.4%)<br>94 ( 2.2%)<br>87 ( 2.0%)<br>237 ( 5.5%) | 0<br>(0%) |

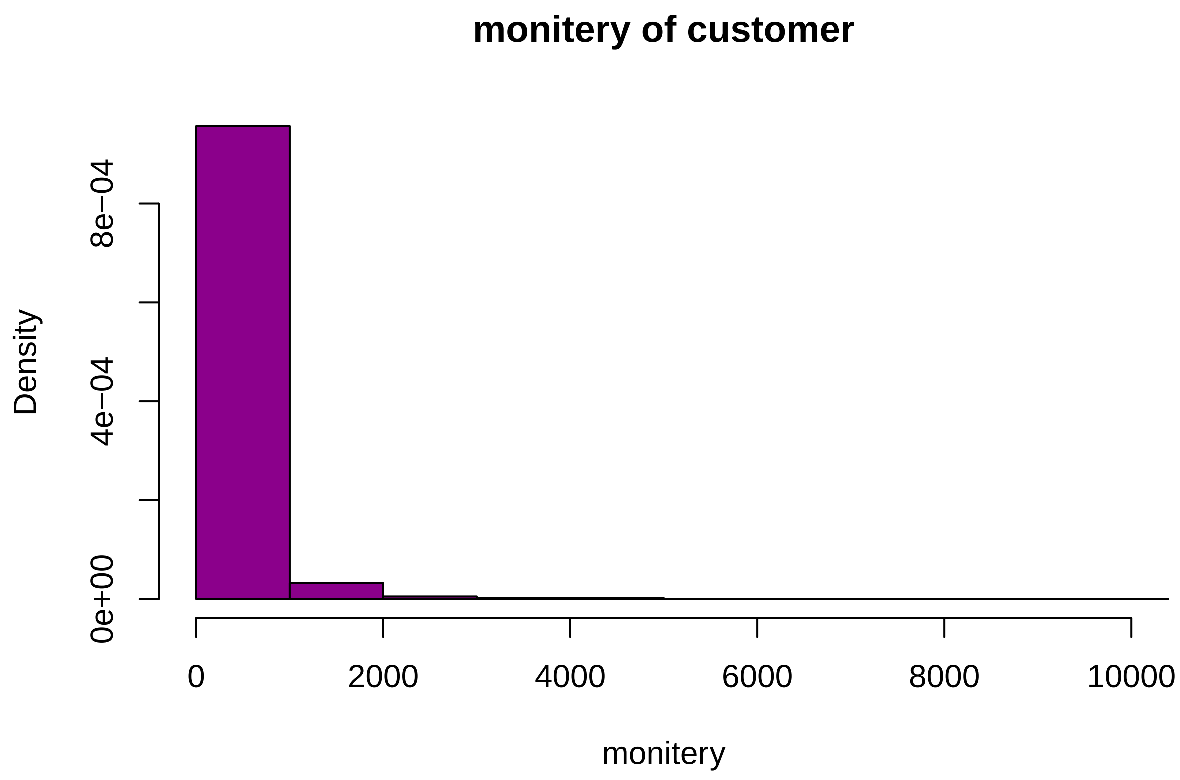
```
# histogram with added parameters
hist(cd_RFM$recency,
main="recency of customer",
xlab="recency",
xlim=c(20,400),
col="darkmagenta",
freq=FALSE
)
```




```
# histogram with added parameters
hist(cd_RFM$frequenci,
main="frequenci of customer",
xlab="frequenci",
breaks=100,
xlim=c(0,50),
col="darkmagenta",
freq=FALSE
)
```

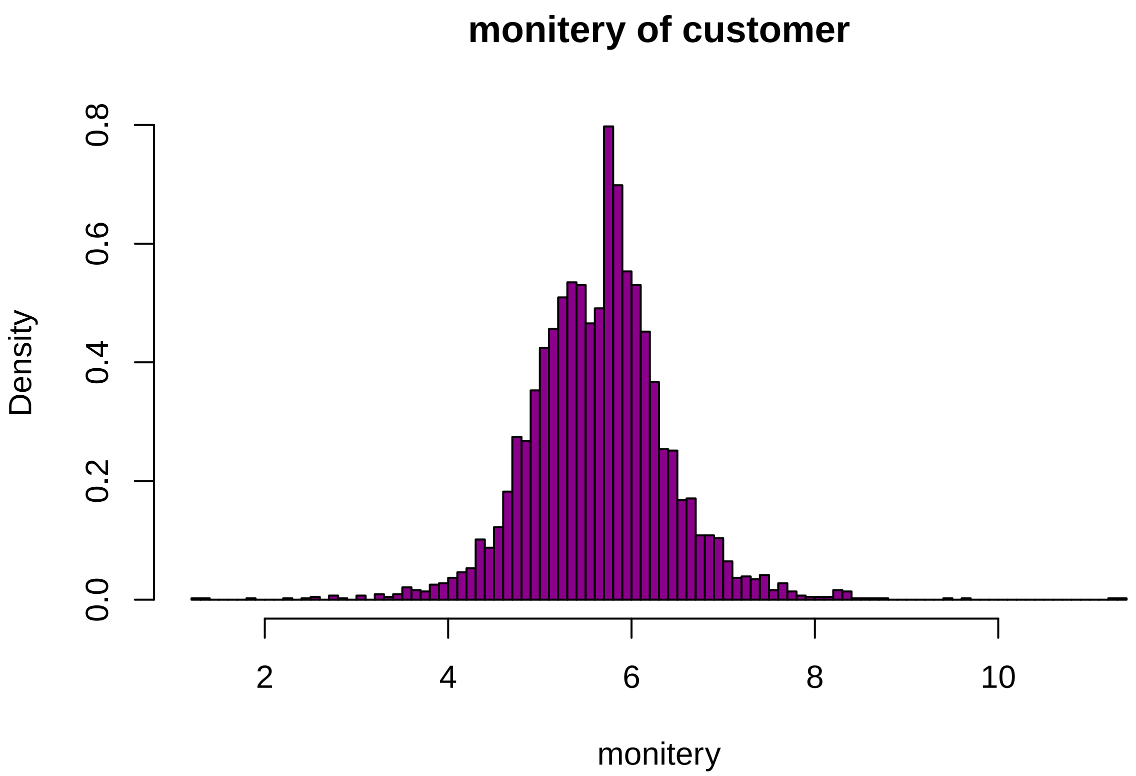



```
# histogram with added parameters
hist(cd_RFM$monitery,
main="monitery of customer",
xlab="monitery",
breaks=100,
xlim=c(0,10000),
col="darkmagenta",
freq=FALSE
)
```



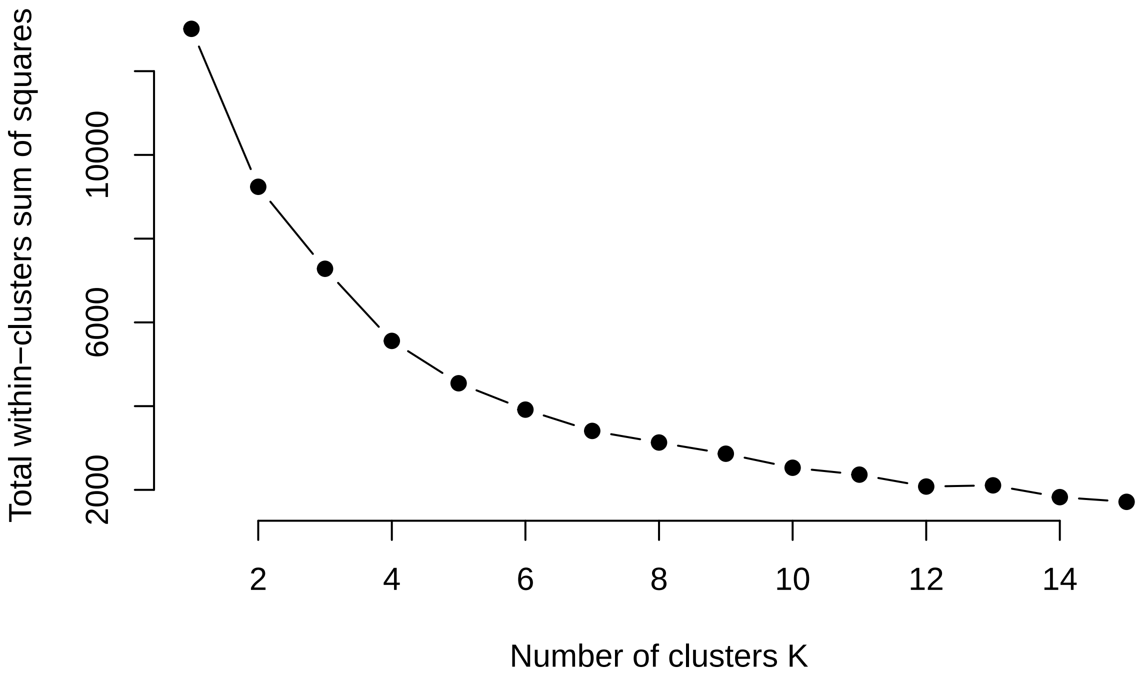

Because the data is realy skewed, we use log scale to normalize

```
cd_RFM$monitery <- log(cd_RFM$monitery)
hist(cd_RFM$monitery,
main="monitery of customer",
xlab="monitery",
breaks=100,
col="darkmagenta",
freq=FALSE
)
```

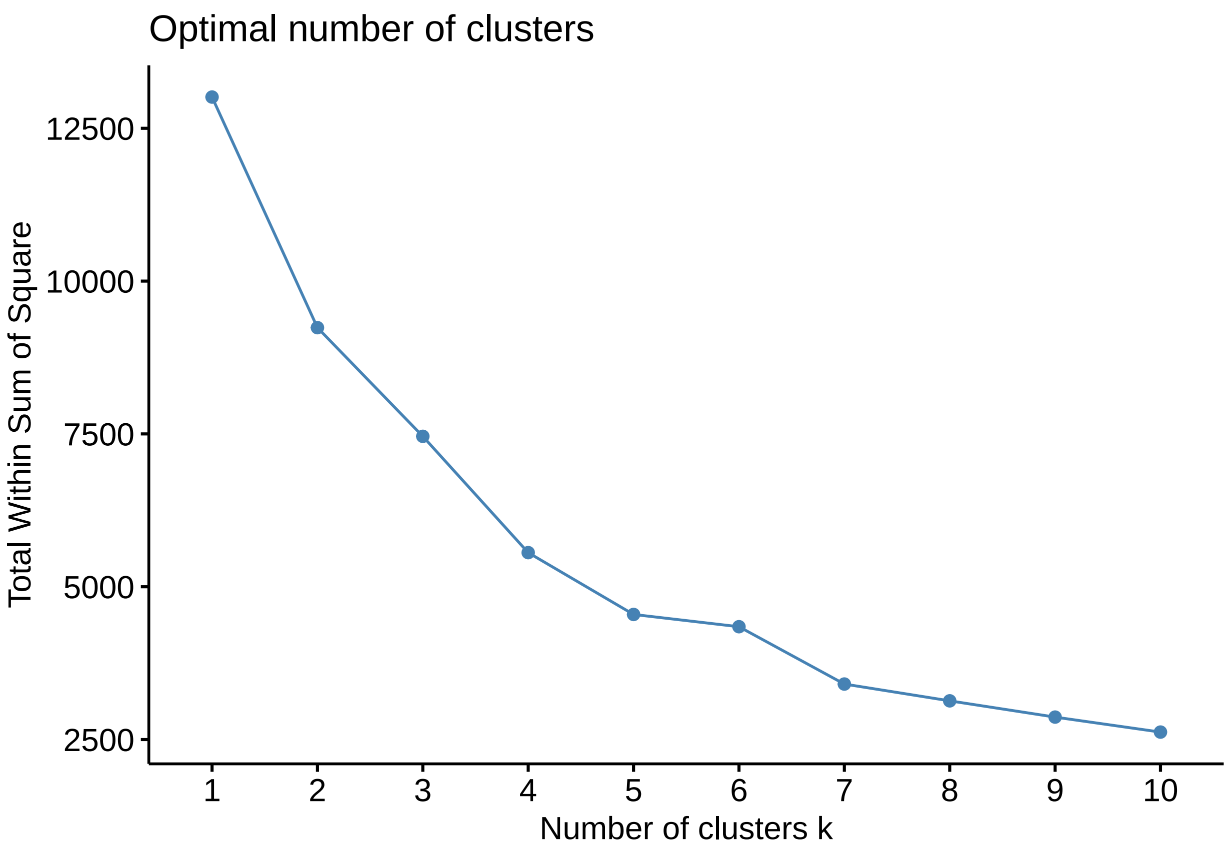



```
cd_RFM1 = cd_RFM%>%
dplyr::select(-CustomerID,-country)

summary(cd_RFM1)

    recency        frequenci          monitery
 Min.   : 23.0   Min.   :  1.000   Min.   : 1.238
 1st Qu.: 40.0   1st Qu.:  1.000   1st Qu.: 5.185
 Median : 73.0   Median :  2.000   Median : 5.683
 Mean   :115.1   Mean   :  4.272   Mean   : 5.646
 3rd Qu.:164.8   3rd Qu.:  5.000   3rd Qu.: 6.064
 Max.   :396.0   Max.   :209.000   Max.   :11.341

cd_RFM2 <- cd_RFM1 %>%
mutate(recency = scale(recency),
       frequenci = scale(frequenci),
       monitery = scale(monitery)

)

summary(cd_RFM2)

      recency.V1         frequenci.V1          monitery.V1
 Min.   :-0.9204819   Min.   :-0.425048   Min.   :-5.883231
 1st Qu.:-0.7505027   1st Qu.:-0.425048   1st Qu.:-0.615310
 Median :-0.4205432   Median :-0.295144   Median : 0.049302
 Mean   : 0.0000000   Mean   : 0.000000   Mean   : 0.000000
 3rd Qu.: 0.4968443   3rd Qu.: 0.094568   3rd Qu.: 0.557567
 Max.   : 2.8090607   Max.   :26.594965   Max.   : 7.601186

set.seed(123)

# function to compute total within-cluster sum of square
wss <- function(k) {
  kmeans(cd_RFM2, k, nstart = 10 )$tot.withinss
```

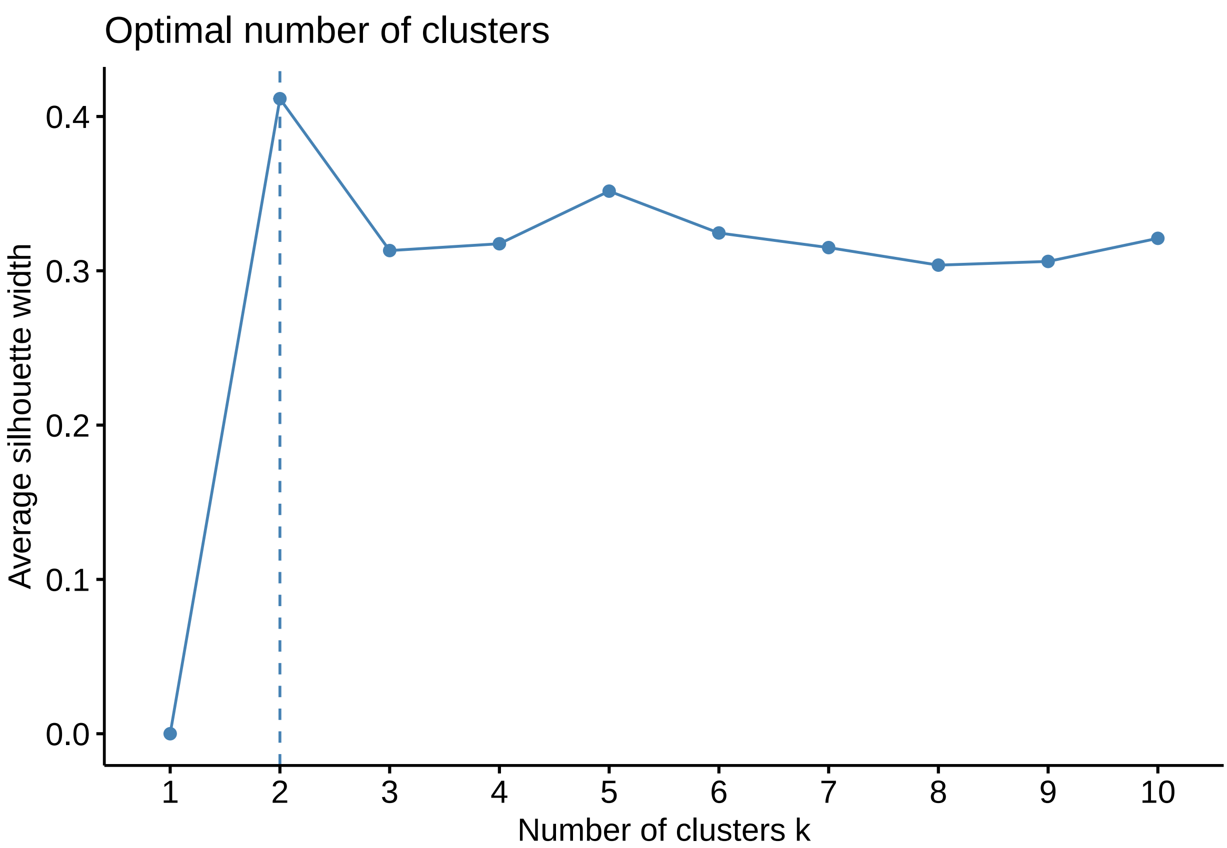
```
}

# Compute and plot wss for k = 1 to k = 15
k.values <- 1:15

# extract wss for 2-15 clusters
wss_values <- map_dbl(k.values, wss)

plot(k.values, wss_values,
       type="b", pch = 19, frame = FALSE,
       xlab="Number of clusters K",
       ylab="Total within-clusters sum of squares")
```



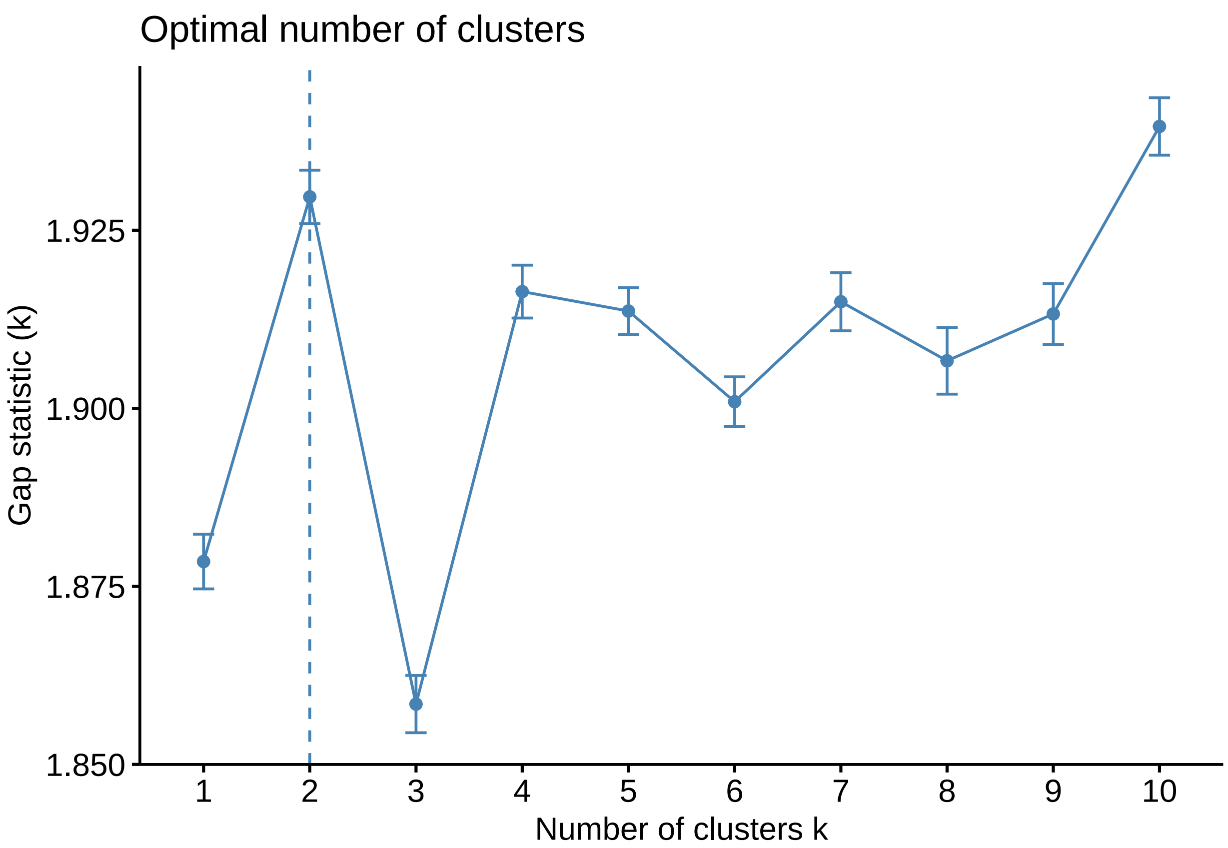

```
set.seed(123)

fviz_nbclust(cd_RFM2, kmeans, method = "wss")
```




```
set.seed(123)
fviz_nbclust(cd_RFM2, kmeans, method = "silhouette")
```

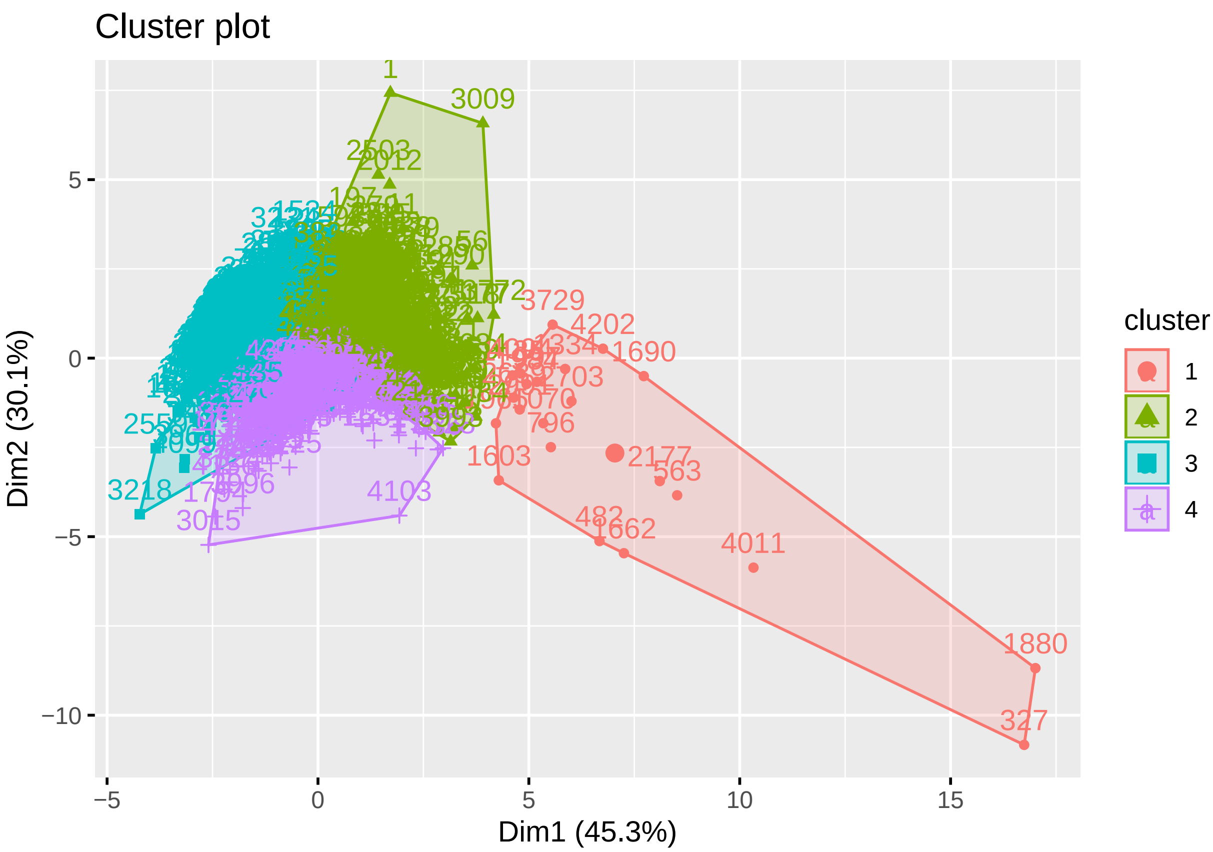


```
set.seed(123)
gap_stat <- clusGap(cd_RFM2, FUN = kmeans, nstart = 25,
                    K.max = 10, B = 50)
# Print the result
print(gap_stat, method = "firstmax")
fviz_gap_stat(gap_stat)
```




```
Clustering Gap statistic ["clusGap"] from call:
clusGap(x = cd_RFM2, FUNcluster = kmeans, K.max = 10, B = 50,     nstart = 25)
B=50 simulated reference sets, k = 1..10; spaceH0="scaledPCA"
 --> Number of clusters (method 'firstmax'): 2
          logW   E.logW      gap      SE.sim
 [1,] 7.665127 9.543614 1.878487 0.003841697
 [2,] 7.428966 9.358659 1.929692 0.003743205
 [3,] 7.385286 9.243754 1.858468 0.004020079
 [4,] 7.221823 9.138214 1.916391 0.003710861
 [5,] 7.142412 9.056076 1.913664 0.003289008
 [6,] 7.070778 8.971714 1.900935 0.003488891
 [7,] 6.992752 8.907722 1.914971 0.004083335
 [8,] 6.940304 8.846979 1.906675 0.004682498
 [9,] 6.896881 8.810133 1.913252 0.004272970
[10,] 6.836043 8.775627 1.939584 0.004036459
```

```
k2 <- kmeans(cd_RFM2, centers = 4, nstart = 25)
k2
fviz_cluster(k2, data = cd_RFM2)
```




```
K-means clustering with 4 clusters of sizes 22, 1840, 972, 1504

Cluster means:
     recency   frequenci    monitery
1 -0.8695790  9.50669672  1.2750038
2 -0.4715478  0.13311570  0.7398225
3  1.6396157 -0.35448242 -0.3008688
4 -0.4700318 -0.07282135 -0.7293078

Clustering vector:
   [1] 2 2 2 2 3 2 3 3 3 2 2 2 2 2 3 2 4 2 3 4 2 2 2 3 2 4 3 2 2 2 2 3 4 3 2 2 2
  [38] 2 2 4 2 2 4 3 3 2 2 2 2 2 2 3 2 4 3 2 2 2 2 4 4 4 2 2 3 4 2 2 4 2 2 2 2 2
  [75] 2 2 2 3 4 2 4 2 3 2 2 4 2 4 2 2 2 2 2 2 2 2 2 4 2 3 2 2 2 2 2 2 2 2 2 2 2
 [112] 2 2 2 3 2 2 4 3 4 2 4 2 3 2 4 3 2 4 3 2 4 2 3 3 2 2 2 2 2 2 4 4 2 2 4 2 2
 [149] 4 2 2 2 2 2 2 2 2 2 2 2 3 4 2 3 4 2 2 2 4 3 2 3 2 3 3 2 2 4 4 2 3 3 4 4 2
 [186] 2 3 4 3 2 4 3 4 4 4 2 2 3 4 4 2 2 2 2 4 4 3 4 2 4 2 2 2 2 2 2 3 2 2 2 4 2
 [223] 2 2 3 3 2 2 2 2 4 2 2 2 2 2 2 3 2 4 2 2 2 2 2 2 3 4 2 3 3 2 4 2 2 2 4 4 2
 [260] 2 3 2 2 2 2 3 2 2 4 2 4 4 2 2 2 2 2 3 2 2 3 4 4 2 4 2 2 2 4 4 2 2 3 2 2 2
 [297] 2 2 2 2 2 2 4 2 2 2 4 4 4 3 2 2 4 2 2 3 3 3 3 3 3 2 2 4 2 2 1 2 4 2 2 3 3
 [334] 4 2 2 2 2 4 3 2 3 2 2 4 4 2 2 2 2 4 3 4 4 3 3 3 3 2 2 3 2 3 4 3 3 4 4 2 4
 [371] 3 3 4 3 2 2 2 4 4 4 3 2 3 4 2 3 2 3 2 2 2 4 2 4 4 3 2 2 4 3 2 2 3 2 3 2 4
 [408] 2 3 2 3 4 3 3 4 2 4 3 4 3 2 4 2 2 2 3 2 3 4 4 3 2 3 4 2 3 2 2 2 2 4 2 2 4
 [445] 4 2 3 4 4 2 2 3 4 2 4 2 4 2 4 2 2 4 2 3 4 2 2 2 4 2 2 2 3 2 4 4 2 4 3 4 4
 [482] 1 3 3 4 2 4 3 4 2 4 3 2 4 4 4 2 4 2 3 4 2 3 4 2 3 4 3 3 4 2 2 2 4 4 2 2 2
 [519] 4 2 2 2 2 2 3 2 2 2 2 2 3 3 2 2 2 2 2 2 3 2 4 3 3 4 3 4 3 4 4 4 4 3 3 2 4
 [556] 2 4 4 2 2 4 3 1 2 2 4 3 4 4 2 2 4 3 2 4 4 4 2 3 2 4 2 2 2 4 4 3 3 2 2 4 4
 [593] 4 2 3 3 2 2 2 2 4 2 4 2 2 3 4 4 4 2 2 3 2 4 2 2 2 4 3 2 2 3 4 2 2 4 4 4 2
 [630] 2 4 2 4 2 4 3 2 3 4 4 2 4 4 2 2 2 4 4 4 4 2 2 2 4 2 3 4 4 4 3 3 2 2 3 3 2
 [667] 3 4 4 2 2 2 4 3 4 4 3 4 2 2 4 4 4 2 4 4 4 2 3 3 2 2 2 2 2 2 3 4 4 4 2 4 2
 [704] 4 4 2 4 3 2 3 4 3 3 4 2 2 3 3 4 4 3 4 4 2 2 2 2 2 4 2 2 4 2 2 4 4 4 2 2 2
 [741] 3 2 2 2 2 2 2 2 2 3 3 2 3 2 4 4 2 2 2 2 2 2 3 3 2 2 2 4 2 4 3 3 3 2 2 2 2
 [778] 4 2 3 2 3 2 4 2 3 2 3 2 2 3 2 4 4 4 1 4 4 2 2 2 2 4 4 2 2 2 2 4 4 4 2 2 4
 [815] 2 4 2 3 4 2 2 4 2 2 3 3 2 2 2 4 2 3 2 3 2 4 4 2 4 3 3 2 4 3 2 3 4 3 3 2 4
 [852] 2 4 4 2 3 3 2 2 3 4 2 3 4 4 4 2 3 2 3 4 2 3 4 4 4 2 2 2 4 2 2 4 2 4 2 2 4
```

```
  [889] 3 4 3 2 2 4 4 4 4 2 4 3 2 2 4 4 4 3 2 2 2 4 2 3 4 3 4 2 4 4 2 2 4 3 2 3 2
  [926] 2 4 2 2 4 2 2 2 4 2 3 2 2 4 4 2 4 2 2 3 4 3 4 4 3 2 2 2 2 2 2 2 4 4 4 2 3
  [963] 4 2 4 2 3 2 4 2 2 2 3 4 2 3 2 3 4 3 2 4 2 3 4 3 3 2 4 3 3 2 3 2 4 2 1 4 4
 [1000] 3 4 3 4 4 3 3 3 2 4 3 3 3 3 4 3 4 4 2 2 3 4 2 4 4 2 3 2 2 2 3 4 3 4 2 4 2
 [1037] 3 3 4 2 3 4 3 4 4 2 2 2 4 2 3 2 2 2 2 2 2 2 2 4 4 3 4 4 3 4 2 2 4 1 2 3 2
 [1074] 2 3 2 2 4 3 2 3 2 3 4 2 2 2 4 4 3 4 2 2 2 3 4 4 2 4 3 4 3 4 3 2 2 2 4 2 2
 [1111] 2 2 2 2 3 4 3 4 2 3 4 4 2 2 2 2 2 4 3 2 4 4 2 4 2 4 2 4 3 4 3 2 2 4 4 3 4
 [1148] 2 2 3 4 3 2 4 2 2 2 4 2 4 3 2 2 4 4 2 4 3 4 2 3 2 3 3 3 4 4 4 2 4 4 3 2 4
 [1185] 3 2 2 4 4 4 4 4 4 3 4 2 2 4 3 2 2 3 4 2 4 2 4 4 2 2 2 2 3 4 4 2 2 4 2 3 2
 [1222] 4 2 2 2 3 3 4 2 2 2 2 2 3 3 4 4 4 4 2 2 2 4 2 3 3 2 4 2 3 4 2 3 4 3 2 2 2
 [1259] 4 4 4 2 2 2 3 2 2 4 3 4 3 4 4 2 4 4 4 2 4 3 3 4 2 4 2 4 3 2 2 2 4 2 4 2 4
 [1296] 4 3 2 2 4 2 4 2 2 3 4 4 2 2 2 4 4 2 4 3 2 4 2 2 2 2 4 3 4 2 2 4 3 3 2 2 3
 [1333] 3 1 4 2 2 3 3 2 2 3 4 2 4 4 2 3 4 4 2 4 2 3 2 2 2 2 2 3 2 4 3 4 4 4 3 4 2
 [1370] 2 4 2 4 2 2 4 4 4 3 2 2 2 4 2 2 2 3 2 2 4 2 4 3 3 4 2 3 3 4 3 2 3 3 2 2 3
 [1407] 2 2 4 2 2 3 4 2 3 3 4 4 2 2 3 2 2 2 2 4 3 4 2 2 2 4 4 2 1 2 4 2 3 2 2 2 4
 [1444] 3 4 4 2 2 3 2 2 3 3 2 4 4 2 2 4 2 3 3 3 2 4 4 4 2 3 4 3 4 3 3 4 3 2 4 2 2
 [1481] 4 2 2 3 3 3 3 3 4 4 2 2 4 3 2 2 2 2 3 2 4 4 2 2 2 4 4 2 2 2 4 3 2 3 4 2 4
 [1518] 2 4 2 2 2 3 2 2 4 4 3 2 4 4 3 3 3 2 2 3 2 4 4 4 4 3 2 4 3 2 3 2 4 2 4 4 3
 [1555] 2 4 4 4 4 2 2 3 2 3 4 3 2 3 2 4 4 2 3 3 2 4 4 3 3 2 4 4 3 4 2 4 2 2 4 4 4
 [1592] 3 4 2 2 4 4 4 4 4 4 2 1 2 4 2 3 2 2 2 3 2 4 4 2 4 3 4 2 2 2 3 3 2 4 4 4 2
 [1629] 4 2 4 2 2 2 2 2 4 3 2 3 3 4 4 4 4 4 3 4 3 4 3 2 4 4 4 2 2 4 3 4 3 1 2 2 4
 [1666] 3 4 3 3 2 3 4 4 3 2 2 2 2 3 3 2 2 2 2 4 4 3 2 3 1 3 2 2 4 2 2 3 4 4 2 4 4
 [1703] 4 2 2 4 2 3 3 3 2 2 2 2 4 4 4 2 2 3 4 2 4 3 2 3 4 3 2 2 4 2 4 3 2 2 4 2 2
 [1740] 4 4 4 4 2 4 4 4 2 3 3 4 4 2 2 4 2 4 2 3 4 4 4 3 2 2 2 2 2 2 2 2 2 3 4 3 4
 [1777] 4 2 4 2 3 4 2 4 2 3 4 3 4 4 2 4 4 4 4 4 3 2 4 4 2 4 3 2 4 4 2 4 3 2 3 3 4
 [1814] 4 4 3 3 4 4 2 4 4 2 4 2 3 2 2 2 3 2 3 4 2 4 2 2 2 2 2 4 4 2 4 4 2 3 2 2 3
 [1851] 4 3 2 4 4 4 2 4 2 3 2 3 3 3 3 2 2 2 3 2 4 4 2 2 2 3 2 4 4 1 2 2 4 2 2 3 2
 [1888] 2 3 4 2 4 3 2 3 2 2 2 4 4 4 2 2 2 4 2 3 4 4 4 2 3 2 3 2 4 2 4 2 2 4 2 4 4
 [1925] 4 2 3 4 3 4 3 4 4 3 2 2 2 2 2 3 2 2 4 4 3 4 4 3 4 4 2 4 2 4 4 4 4 4 3 4 2
 [1962] 4 4 4 1 3 3 2 2 2 2 3 4 3 2 2 3 4 4 3 4 4 4 1 4 3 2 2 2 4 2 3 2 2 2 2 2 2
 [1999] 3 4 3 3 2 2 3 2 3 2 2 4 4 2 2 3 4 2 3 2 2 3 2 3 4 2 2 2 2 2 2 4 3 2 4 2 2
 [2036] 4 2 2 2 2 3 4 2 4 3 2 3 2 2 3 2 2 2 3 2 2 2 2 2 4 2 2 2 2 3 4 4 3 3 4 3 2
 [2073] 3 4 3 2 3 4 3 4 2 4 2 2 2 2 2 2 2 4 2 4 2 3 2 4 4 3 2 3 3 4 4 4 2 3 2 3 3
 [2110] 3 3 3 3 4 2 3 2 3 3 3 2 2 2 2 4 2 2 2 3 2 2 2 2 4 2 2 4 3 3 4 4 3 4 2 2 3
 [2147] 2 2 2 4 2 2 2 4 4 3 3 4 4 4 2 2 2 2 3 2 4 2 2 2 2 3 4 2 4 3 1 4 4 3 4 3 2
 [2184] 3 4 2 2 4 3 4 2 2 3 2 4 4 4 2 2 2 4 2 3 3 4 3 3 2 3 4 2 2 2 3 3 2 2 2 2 2
 [2221] 2 2 4 4 2 2 2 3 2 3 2 3 3 3 2 4 2 3 4 3 2 4 4 3 3 4 4 3 2 2 3 4 3 2 2 4 4
 [2258] 3 4 3 2 2 4 4 4 3 4 4 2 3 4 4 3 2 2 3 3 3 2 2 4 4 3 3 3 4 3 4 4 3 2 2 4 4
 [2295] 4 2 2 4 2 2 2 2 4 2 2 4 2 2 2 2 2 2 4 2 3 4 2 2 3 3 3 4 4 2 2 4 3 4 2 2 2
 [2332] 4 2 2 4 4 2 3 2 2 2 2 4 3 3 2 4 4 3 2 4 2 2 4 2 3 2 3 2 4 2 4 3 2 4 4 4 4
 [2369] 2 4 2 2 3 2 4 3 4 2 4 4 3 4 3 2 4 2 4 2 4 2 2 4 4 4 4 2 3 4 3 4 4 4 2 4 2
 [2406] 2 2 2 3 4 2 2 2 2 2 4 2 4 2 2 2 3 3 2 4 3 3 4 2 2 2 3 4 4 3 2 4 3 4 2 2 2
 [2443] 3 4 3 2 2 2 4 2 2 2 4 2 2 4 2 4 4 3 4 2 3 2 2 3 2 3 2 3 4 4 2 3 4 4 4 3 3
 [2480] 2 2 2 2 3 3 2 2 3 2 3 3 2 4 2 2 4 4 4 4 4 2 4 2 2 2 3 4 2 4 2 4 3 4 2 2 2
 [2517] 4 2 2 4 3 4 3 3 4 2 4 2 2 2 3 4 2 2 2 2 2 2 2 2 2 2 2 2 4 2 3 4 2 2 2 3 2
 [2554] 2 2 3 4 2 3 2 2 4 2 2 4 3 2 4 4 2 2 2 4 2 3 4 2 4 4 3 4 4 4 2 4 2 4 2 4 4
 [2591] 2 2 4 2 3 2 4 2 4 3 3 3 4 4 4 4 3 3 3 4 3 4 3 2 4 2 4 4 2 3 4 3 4 3 3 2 4
 [2628] 3 3 3 3 2 2 2 3 4 4 2 3 2 3 3 2 2 4 2 2 4 4 2 2 4 4 4 4 4 3 2 3 2 2 3 2 2
 [2665] 2 2 2 2 2 4 4 4 2 4 2 2 4 2 2 2 4 3 2 4 4 3 4 2 1 4 4 4 4 4 2 4 3 4 4 2 4
 [2702] 4 1 3 4 2 3 2 3 4 2 2 4 4 2 3 2 3 2 2 2 3 2 2 4 2 3 3 2 2 2 3 3 4 3 2 4 2
 [2739] 2 3 4 4 4 4 4 4 3 3 2 4 2 2 4 2 3 4 2 2 3 2 4 3 3 4 3 3 4 2 2 4 2 3 3 2 2
 [2776] 2 2 2 2 3 4 4 2 2 3 2 3 2 3 2 3 4 2 3 2 2 4 3 2 2 3 3 4 4 2 2 4 4 4 2 4 2
 [2813] 4 4 2 2 2 2 4 4 2 2 4 4 4 2 3 2 2 2 3 4 2 4 4 2 2 2 2 2 2 4 3 4 3 3 2 4 3
 [2850] 2 4 3 3 2 3 4 4 4 2 4 2 4 2 4 3 2 3 4 2 3 2 2 3 4 2 2 4 2 4 2 3 2 2 3 3 2
 [2887] 2 2 2 3 2 2 3 2 4 2 3 4 4 2 2 3 2 2 2 4 3 2 2 2 3 2 4 4 3 4 4 4 2 4 4 2 4
 [2924] 3 3 2 2 4 4 4 2 2 4 3 4 3 4 2 4 3 4 2 4 2 4 4 4 2 4 2 2 4 2 2 4 4 2 3 3 2
 [2961] 2 2 4 4 4 3 2 3 4 2 2 2 4 2 2 3 3 2 3 4 4 2 4 4 4 4 4 2 4 4 1 2 3 3 4 3 4
 [2998] 2 3 4 4 4 4 2 4 4 4 4 2 4 3 3 3 4 4 4 2 3 4 4 3 4 2 2 4 4 4 4 3 4 4 4 2 4
 [3035] 4 2 4 3 3 2 4 2 2 3 4 2 2 4 3 4 2 2 2 4 3 3 2 4 2 4 3 4 4 3 2 2 3 4 4 3 4
 [3072] 2 2 2 2 2 4 3 3 2 2 2 3 2 4 4 2 4 2 2 4 2 3 4 3 4 2 2 2 2 2 3 2 3 2 3 3 4
 [3109] 4 2 4 4 2 4 4 3 4 4 2 4 4 2 3 3 2 2 3 4 3 2 3 4 4 3 2 3 2 2 2 4 2 3 2 4 2
 [3146] 4 2 3 3 3 4 2 2 2 4 2 2 4 3 4 2 3 4 4 4 2 2 2 2 4 2 2 3 4 2 4 2 4 2 2 4 4
 [3183] 3 4 4 3 3 2 2 2 4 2 2 4 2 2 2 3 2 3 2 3 4 4 3 2 2 3 4 3 4 2 4 2 4 2 2 3 3
 [3220] 2 2 4 2 2 2 2 4 2 3 4 3 2 3 3 2 4 2 2 3 4 3 3 2 4 3 4 2 4 4 4 2 4 4 2 4 3
```

```
[3257] 4 4 2 4 4 4 3 4 4 3 2 2 2 2 4 3 4 2 4 4 3 4 3 4 2 4 4 2 2 3 4 2 2 4 3 2 3
[3294] 4 3 4 4 2 3 2 2 2 2 4 4 4 3 3 4 4 4 4 3 2 4 4 4 4 2 4 2 2 4 4 4 2 4 3 4 4
[3331] 4 3 4 4 3 4 3 4 4 2 2 2 4 2 4 4 2 4 4 3 4 4 4 3 4 4 4 2 4 3 4 4 4 4 4 4 4
[3368] 2 4 2 2 4 3 4 3 4 4 2 2 4 4 4 3 2 3 4 4 4 3 3 2 3 4 4 2 3 2 2 3 2 2 2 2 2
[3405] 4 4 4 3 2 4 3 4 4 2 2 2 4 4 4 4 4 2 2 2 2 4 4 4 3 4 2 4 4 2 4 3 2 4 2 2 4
[3442] 2 2 4 2 2 2 2 2 3 2 2 4 2 2 3 2 3 4 4 4 2 2 2 2 4 4 4 4 4 4 2 2 2 2 2 2 2
[3479] 2 2 3 2 4 4 2 2 3 3 2 4 3 4 4 4 3 3 2 3 4 3 3 4 4 3 2 2 4 4 3 2 3 3 2 4 2
[3516] 4 3 4 2 4 4 2 2 2 4 4 2 3 4 2 3 2 2 3 2 3 3 4 4 2 4 3 2 4 4 2 4 3 2 4 2 2
[3553] 3 2 4 2 3 2 4 3 4 4 4 4 2 3 3 4 4 4 3 2 4 4 3 4 4 4 4 3 2 2 2 3 4 2 4 3 4
[3590] 4 4 4 4 3 4 3 4 4 4 4 2 4 3 3 4 2 4 4 4 4 2 4 3 4 4 4 4 4 4 4 3 2 4 2 4 2
[3627] 2 4 2 2 2 4 4 3 4 4 4 2 3 2 2 4 2 3 3 4 4 4 4 4 3 2 4 2 2 3 4 3 4 4 4 3 2
[3664] 2 3 4 4 4 4 4 3 3 4 4 2 2 4 4 4 4 2 2 2 2 3 3 3 2 2 4 2 4 4 2 2 3 4 3 4 4
[3701] 4 2 2 2 4 2 4 2 2 2 2 4 2 4 4 3 4 2 2 4 3 4 2 3 2 4 4 2 1 4 4 3 3 3 2 2 2
[3738] 4 4 2 2 3 2 4 2 4 3 4 2 2 4 3 4 2 2 3 4 3 4 3 2 3 4 4 3 2 2 3 3 2 2 2 2 2
[3775] 4 4 4 3 2 2 4 4 4 3 3 2 4 2 4 3 4 3 3 2 4 4 3 4 3 4 3 3 4 2 2 4 3 4 4 4 4
[3812] 4 2 2 3 2 4 4 3 4 3 4 3 2 3 2 4 2 2 2 2 2 2 4 2 2 4 3 2 2 3 4 4 2 2 2 2 4
[3849] 3 2 3 4 2 2 2 2 4 4 3 4 2 4 4 4 3 2 2 3 2 3 4 2 2 4 2 2 2 3 2 2 2 4 4 4 2
[3886] 2 2 4 2 3 2 2 4 4 2 4 2 3 2 4 2 4 2 2 2 3 4 4 3 2 2 3 3 2 3 2 4 3 4 2 3 3
[3923] 3 4 2 3 3 2 3 2 2 2 2 2 2 2 2 2 2 3 2 4 2 4 4 2 2 4 2 3 4 3 4 3 2 4 2 4 4
[3960] 2 3 2 4 4 2 2 3 4 4 2 3 4 4 4 4 4 4 2 2 3 4 4 2 3 4 4 2 4 2 2 2 2 2 4 2 4
[3997] 4 2 2 2 2 3 4 4 4 4 4 4 2 2 1 3 4 2 4 3 3 4 3 3 2 2 2 3 3 4 3 2 4 2 2 4 4
[4034] 4 3 2 3 3 2 3 3 3 4 4 4 2 4 4 3 3 3 4 3 4 4 4 3 4 3 3 4 4 4 3 3 2 3 4 2 4
[4071] 2 2 4 3 2 3 4 4 4 2 4 2 2 4 2 2 4 2 2 4 3 4 2 1 4 2 4 3 3 4 2 4 4 4 4 4 4
[4108] 3 3 3 3 3 2 4 2 3 4 3 4 3 4 2 4 3 4 3 3 4 3 4 4 4 3 4 2 4 3 3 3 4 4 4 2 4
[4145] 4 4 2 3 4 4 3 4 4 4 4 4 4 4 4 4 4 4 3 3 2 2 3 4 4 4 3 2 4 2 2 3 2 3 3 2 3
[4182] 4 4 3 2 2 4 2 4 3 3 2 4 2 2 2 3 2 2 4 4 1 2 4 3 3 2 4 4 3 3 4 2 2 3 3 3 2
[4219] 2 4 4 4 3 2 3 4 4 3 2 2 2 4 2 4 4 4 4 2 4 4 2 4 3 2 2 2 3 2 3 4 2 4 2 2 4
[4256] 3 4 2 2 2 2 3 4 3 2 2 3 3 2 3 2 2 2 4 3 3 2 4 4 2 4 3 3 2 2 2 3 4 3 4 2 2
[4293] 3 2 2 3 2 2 2 3 4 3 2 4 4 3 2 4 2 2 2 2 4 3 2 4 4 2 2 2 4 4 2 2 4 3 4 2 4
[4330] 4 2 4 4 3 3 4 4 2

Within cluster sum of squares by cluster:
[1]  803.7674 2239.1390 1288.0246 1227.1171
 (between_SS / total_SS =  57.3 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

group 1: Champions
Bought recently, buy often and spend the most!
Reward them. Can be early adopters for new products. Will promote your brand.

group 2: Recent Customers
Bought most recently, but not often.
Provide on-boarding support, give them early success, start building relationship.

group 3: Hibernating Last purchase was long back, low spenders and low number of orders. Offer other relevant products and special discounts. Recreate brand value.

group 4: Promising
Recent shoppers, but haven't spent much.
Create brand awareness, offer free trials

```
group <- k2$cluster
cd_RFM3 <- cbind(cd_RFM, group)
#write.csv(cd_RFM3, "../data/mydata1.csv")
```

## Note from the Authors

This file was generated using *The R Journal* style article template, additional information on how to prepare articles for submission is here - Instructions for Authors. The article itself is an executable R Markdown file that could be downloaded from Github with all the necessary artifacts.

*Ketao Li*

*York University*

liketao@yahoo.com

*Kush Halani*
*York University*

kush.halani@ontariotechu.net

*Josue Romain*
*York University*

josue.rolland.romain@gmail.com

*Juan Peña*
*York University*

jppena62@my.yorku.ca