Shopify_Ds_Winter_Challenge

sandeep kumar yedla

5/13/2022

```
#Libraries
# install.packages('googlesheets4')
library("readxl")
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
##
## 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(plotly)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(googlesheets4)
library(skimr)

library(Hmisc)
```

```
## Loading required package: lattice
```

```
## Loading required package: survival
```

```
## Warning: package 'survival' was built under R version 4.0.5
```

```
## Loading required package: Formula
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following object is masked from 'package:plotly':
##
##     subplot
```

```
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
Shopify_Df <-read_xlsx('2019_Winter_Data_Science_Intern_Challenge_DataSet.xlsx')
#Shopify_Df <- read_sheet('https://docs.google.com/spreadsheets/d/16i38oonuX1y1g7C_UAmiK9GkY7cS-64DfiDMNiR41LM/edi#t#gid=0')
```

EDA

```
glimpse(Shopify_Df)
```

```
## Rows: 5,000
## Columns: 7
## $ order_id       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, …
## $ shop_id        <dbl> 53, 92, 44, 18, 18, 58, 87, 22, 64, 52, 66, 40, 54, 100…
## $ user_id        <dbl> 746, 925, 861, 935, 883, 882, 915, 761, 914, 788, 848, …
## $ order_amount   <dbl> 224, 90, 144, 156, 156, 138, 149, 292, 266, 146, 322, 3…
## $ total_items    <dbl> 2, 1, 1, 1, 1, 1, 1, 2, 2, 1, 2, 2, 2, 1, 3, 2000, 1, 1…
## $ payment_method <chr> "cash", "cash", "cash", "credit_card", "credit_card", "…
## $ created_at     <dttm> 2017-03-13 12:36:56, 2017-03-03 17:38:51, 2017-03-14 0…
```

```
glimpse(Shopify_Df)
```

```
## Rows: 5,000
## Columns: 7
## $ order_id       <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, …
## $ shop_id        <dbl> 53, 92, 44, 18, 18, 58, 87, 22, 64, 52, 66, 40, 54, 100…
## $ user_id        <dbl> 746, 925, 861, 935, 883, 882, 915, 761, 914, 788, 848, …
## $ order_amount   <dbl> 224, 90, 144, 156, 156, 138, 149, 292, 266, 146, 322, 3…
## $ total_items    <dbl> 2, 1, 1, 1, 1, 1, 1, 2, 2, 1, 2, 2, 2, 1, 3, 2000, 1, 1…
## $ payment_method <chr> "cash", "cash", "cash", "credit_card", "credit_card", "…
## $ created_at     <dttm> 2017-03-13 12:36:56, 2017-03-03 17:38:51, 2017-03-14 0…
```

```
Sub_df<-select(Shopify_Df,shop_id, order_amount, total_items, payment_method,created_at)
skim(Sub_df) # Direct Average is showing  an average of 3145 for Order_amount
```

Data summary

Name	Sub_df
Number of rows	5000
Number of columns	5
_____	
Column type frequency:	
character	1
numeric	3
POSIXct	1
_____	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
payment_method	0	1	4	11	0	3	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
shop_id	0	1	50.08	29.01	1	24	50	75	100	▇▇▇▇▇
order_amount	0	1	3145.13	41282.54	90	163	284	390	704000	▇▁▁▁▁
total_items	0	1	8.79	116.32	1	1	2	3	2000	▇▁▁▁▁

Variable type: POSIXct

skim_variable	n_missing	complete_rate	min	max	median	n_unique
created_at	0	1	2017-03-01 00:08:09	2017-03-30 23:55:35	2017-03-16 00:21:20	4995

```
#skim(mtcars) %>% skimr::kable(format = "latex", booktabs = T)
```

```
summary(Shopify_Df$order_amount)
```

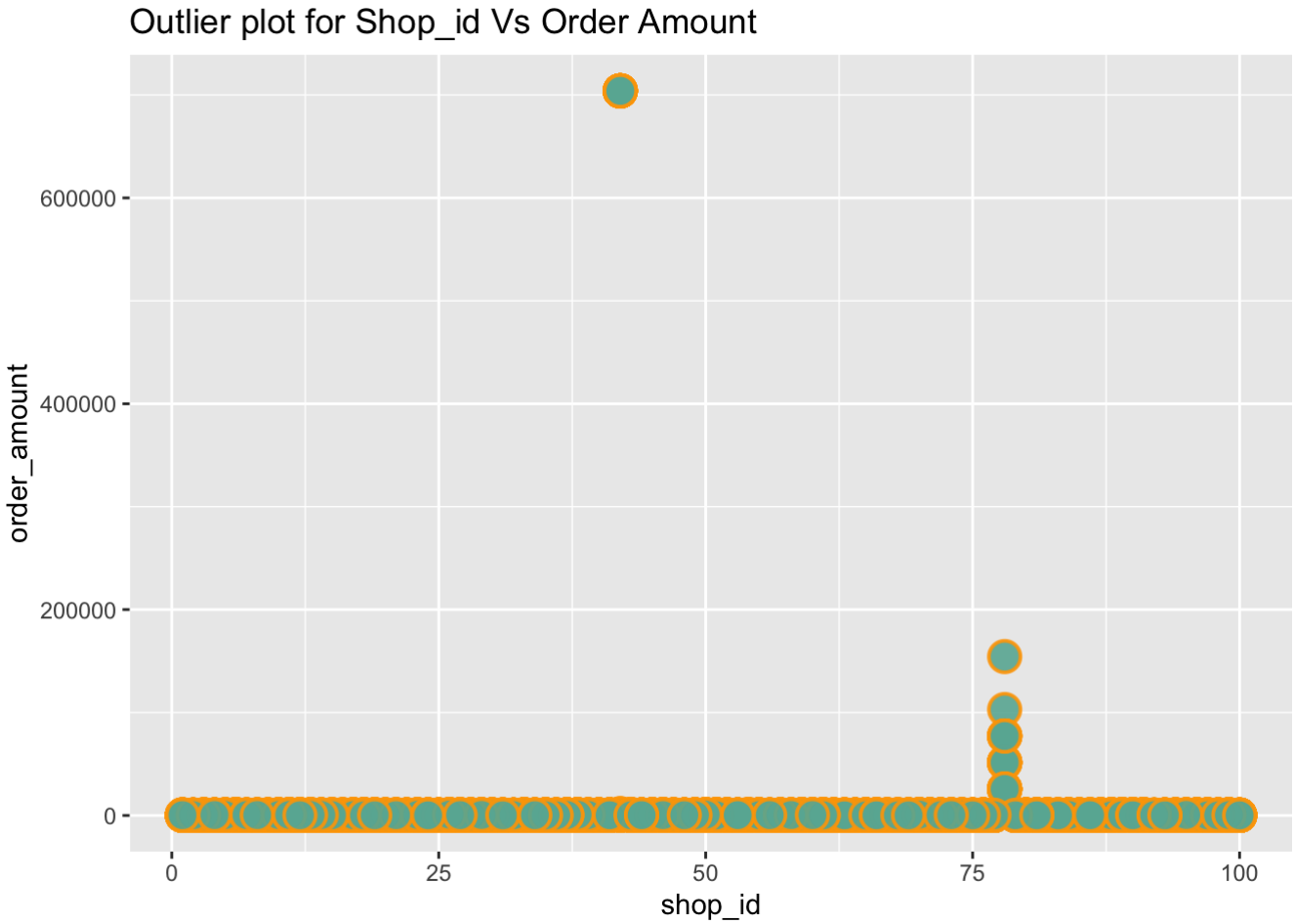
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      90     163     284    3145     390  704000
```

```
mean(Shopify_Df$order_amount)
```

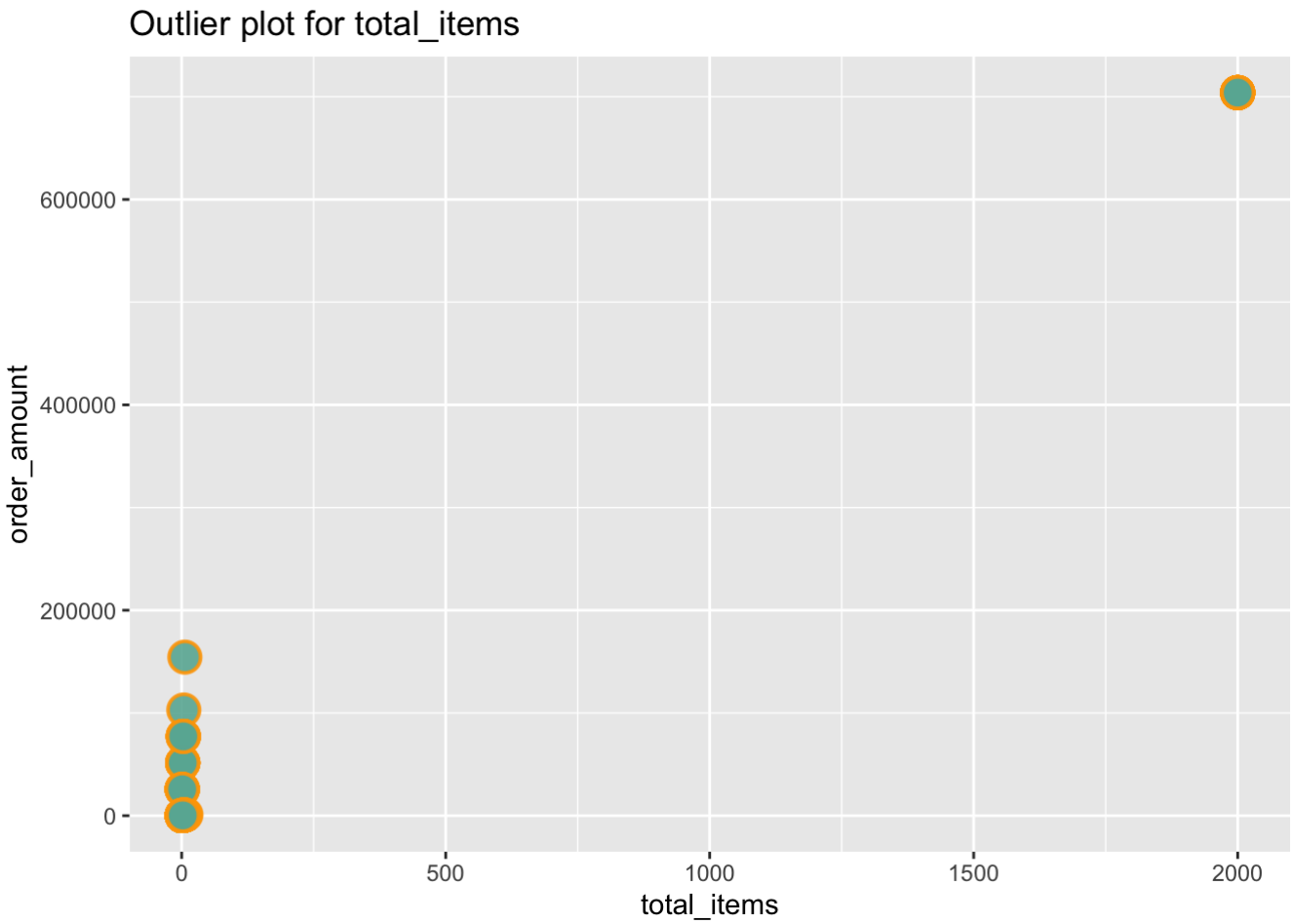
```
## [1] 3145.128
```

Visualize the Data

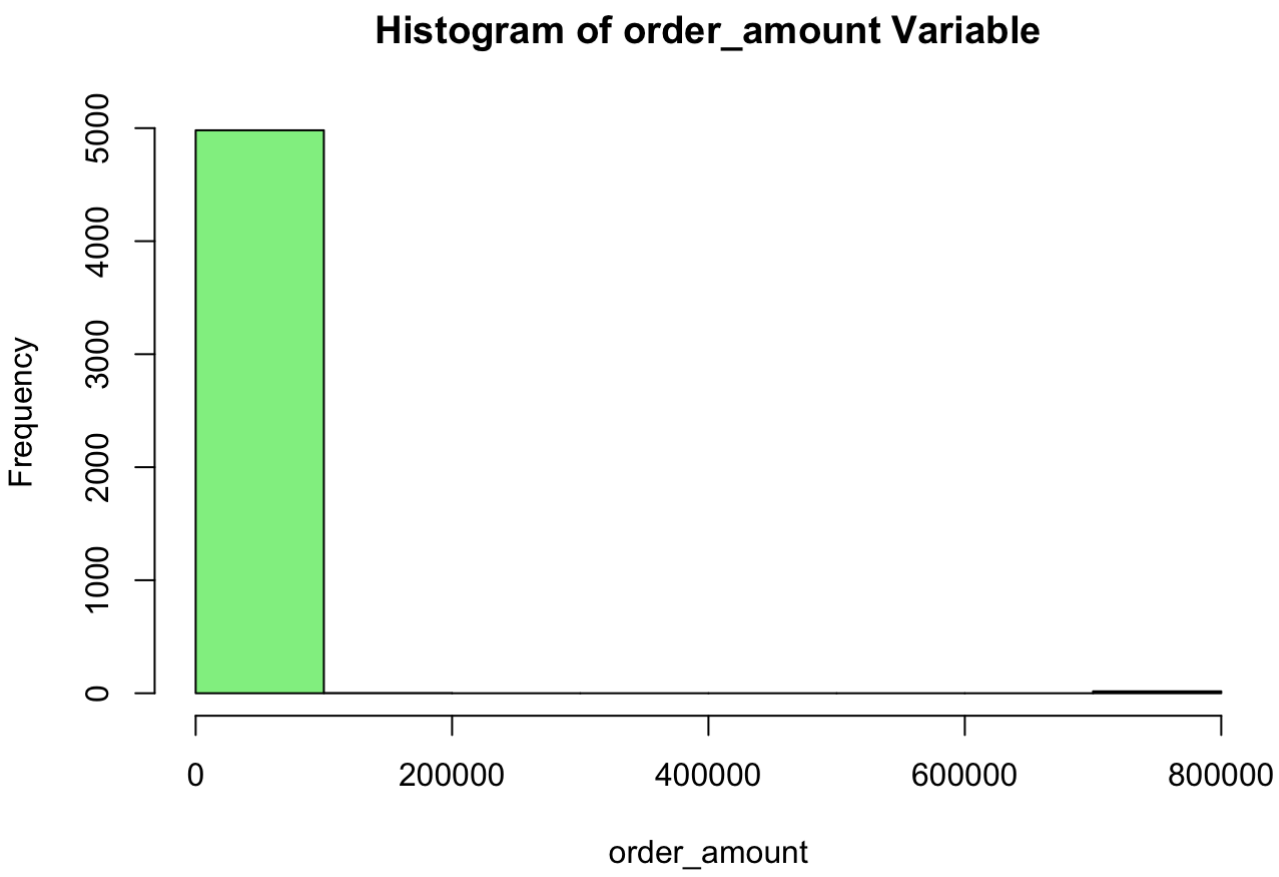
```
# library
library(ggplot2)

options(scipen=999) # removing scientfic notation
# use options!
ggplot(Shopify_Df, aes(x=shop_id, y=order_amount)) +
    geom_point(
        color="orange",
        fill="#69b3a2",
        shape=21,
        alpha=0.9,
        size=5,
        stroke = 1
        )+ggtitle("Outlier plot for Shop_id Vs Order Amount")
```



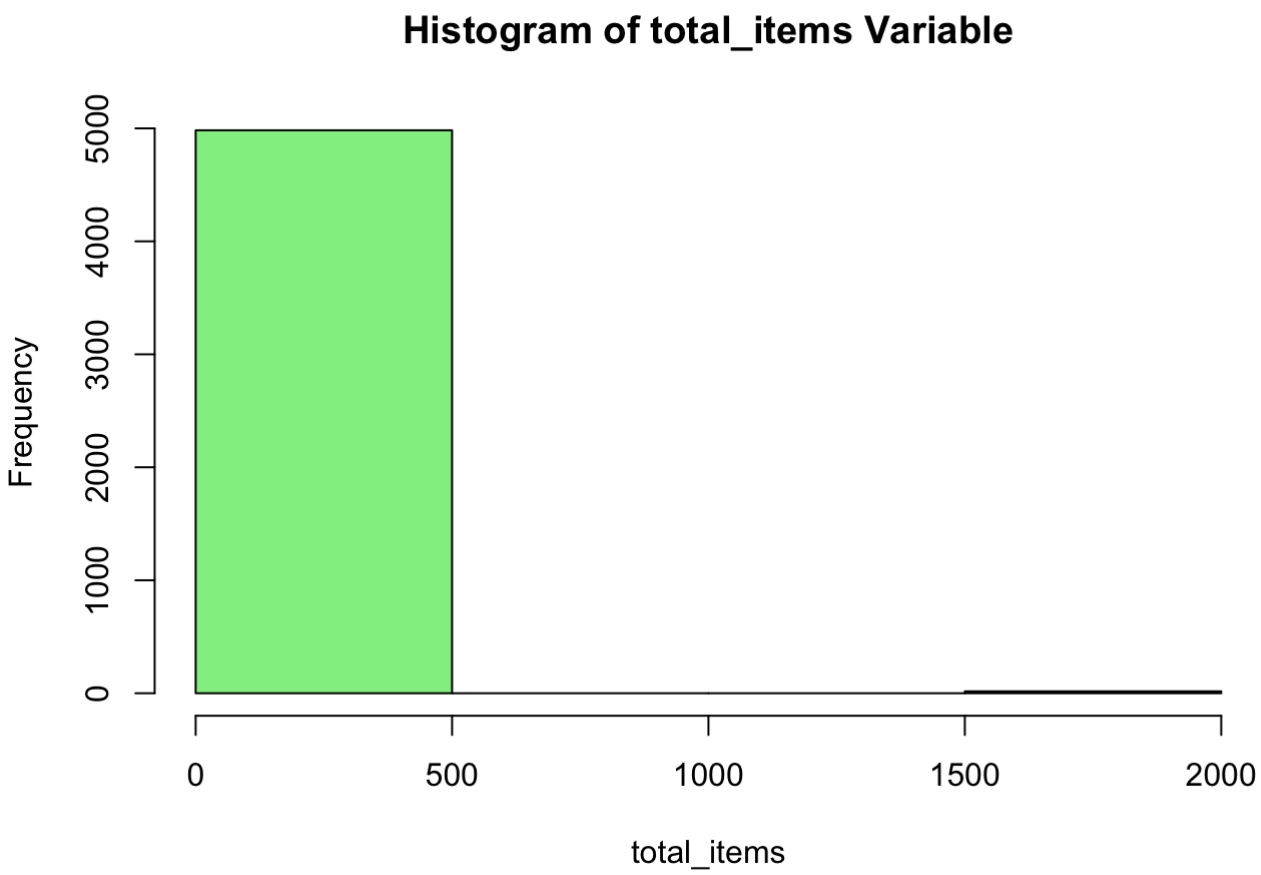

```
ggplot(Shopify_Df, aes(y=order_amount, x=total_items)) +
    geom_point(
        color="orange",
        fill="#69b3a2",
        shape=21,
        alpha=0.9,
        size=5,
        stroke = 1
        )+ ggtitle("Outlier plot for total_items ")
```




```
# Building a histogram (Second Visualization for order amount)
hist(Shopify_Df$order_amount,
     breaks = 10,
     col = "lightgreen",
     main = "Histogram of order_amount Variable",
     xlab = "order_amount")
```




```
# Building a histogram (Second Visualization for )
hist(Shopify_Df$total_items,
     breaks = 5,
     col = "lightgreen",
     main = "Histogram of total_items Variable",
     xlab = "total_items")
```




```
##Calculating thr Average by shop_id and arranging in descending order to look at the ou
tlier shop_id that
## affecting the total avaerage.
AOV_By_Shop <- Shopify_Df %>%
       group_by(shop_id) %>%
     summarise(averagebyshop = sum(order_amount)/sum(total_items)) %>%
        arrange(desc(averagebyshop))
#AOV_By_Shop
```

Ans) We can see that the Shopid 78 has 25725 as its average which is manipulating the overal avaerage.

Idea to check on Order amount and total_items

```
## Grouby operation on total-items and arranging in descending order to check the outlie
r
##  and arranging it in descending order to find out the number of occurances.

Iterm_Count <-Shopify_Df %>% group_by(total_items) %>% summarise(Number_of_occurances_In
_DF=n())
Iterm_Count %>%arrange(desc(total_items))
```

```
## # A tibble: 8 × 2
##   total_items Number_of_occurances_In_DF
##         <dbl>                      <int>
## 1        2000                         17
## 2           8                          1
## 3           6                          9
## 4           5                         77
## 5           4                        293
## 6           3                        941
## 7           2                       1832
## 8           1                       1830
```

ANS) Among the total items ordered there is 2000 an outlier in the data which is affecting the order_amount dractically due to more items wrongly ordered comparitively.

```
## Finding out the shop_id that has the order amount of 2000
view_ShopID_2000<- subset(Shopify_Df, select = c(shop_id,total_items), Shopify_Df$total_
items==2000)
view_ShopID_2000
```

```
## # A tibble: 17 × 2
##    shop_id total_items
##      <dbl>       <dbl>
##  1      42        2000
##  2      42        2000
##  3      42        2000
##  4      42        2000
##  5      42        2000
##  6      42        2000
##  7      42        2000
##  8      42        2000
##  9      42        2000
## 10      42        2000
## 11      42        2000
## 12      42        2000
## 13      42        2000
## 14      42        2000
## 15      42        2000
## 16      42        2000
## 17      42        2000
```

#Ans) After summerizing it can be seen that the 2000 items are ordered from the shop_id 42. SHop_id 42 is effecting the Average Cost.

```
# Removing the orders with 2000

cleaned_df <-subset(Shopify_Df, Shopify_Df$total_items!=2000)
cleaned_df
```

```
## # A tibble: 4,983 × 7
##    order_id shop_id user_id order_amount total_items payment_method
##       <dbl>   <dbl>   <dbl>        <dbl>       <dbl> <chr>
##  1        1      53     746          224           2 cash
##  2        2      92     925           90           1 cash
##  3        3      44     861          144           1 cash
##  4        4      18     935          156           1 credit_card
##  5        5      18     883          156           1 credit_card
##  6        6      58     882          138           1 credit_card
##  7        7      87     915          149           1 cash
##  8        8      22     761          292           2 cash
##  9        9      64     914          266           2 debit
## 10       10      52     788          146           1 credit_card
## # … with 4,973 more rows, and 1 more variable: created_at <dttm>
```

```
Num_Orders <-Shopify_Df %>% group_by(shop_id) %>% summarise(Num_Orders_on_ShopID=n(), It
ems_ordered=total_items) %>% arrange(desc(Num_Orders_on_ShopID))
```

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

```
Num_Orders
```

```
## # A tibble: 5,000 × 3
## # Groups:   shop_id [100]
##    shop_id Num_Orders_on_ShopID Items_ordered
##      <dbl>                <int>         <dbl>
##  1      53                   68             2
##  2      53                   68             1
##  3      53                   68             5
##  4      53                   68             2
##  5      53                   68             2
##  6      53                   68             2
##  7      53                   68             1
##  8      53                   68             2
##  9      53                   68             3
## 10      53                   68             1
## # … with 4,990 more rows
```

Ans) After cleaning the order_items and total_items with for equal distribution of data.

```
Orders_outliers <-cleaned_df %>% filter( order_amount>20000)

Orders_outliers%>%group_by(shop_id) %>% summarise(Num_Orders_on_ShopID=n())
```

```
## # A tibble: 1 × 2
##   shop_id Num_Orders_on_ShopID
##     <dbl>                <int>
## 1      78                   46
```

```
cleaned_shopify_df <-subset(cleaned_df, cleaned_df$shop_id!=78)
cleaned_shopify_df
```

```
## # A tibble: 4,937 × 7
##    order_id shop_id user_id order_amount total_items payment_method
##       <dbl>   <dbl>   <dbl>        <dbl>       <dbl> <chr>
##  1        1      53     746          224           2 cash
##  2        2      92     925           90           1 cash
##  3        3      44     861          144           1 cash
##  4        4      18     935          156           1 credit_card
##  5        5      18     883          156           1 credit_card
##  6        6      58     882          138           1 credit_card
##  7        7      87     915          149           1 cash
##  8        8      22     761          292           2 cash
##  9        9      64     914          266           2 debit
## 10       10      52     788          146           1 credit_card
## # … with 4,927 more rows, and 1 more variable: created_at <dttm>
```

```
cleaned_shopify_df['Amount_per_Order']= (cleaned_shopify_df$order_amount /cleaned_shopif
y_df$total_items)
cleaned_shopify_df
```

```
## # A tibble: 4,937 × 8
##    order_id shop_id user_id order_amount total_items payment_method
##       <dbl>   <dbl>   <dbl>        <dbl>       <dbl> <chr>
##  1        1      53     746          224           2 cash
##  2        2      92     925           90           1 cash
##  3        3      44     861          144           1 cash
##  4        4      18     935          156           1 credit_card
##  5        5      18     883          156           1 credit_card
##  6        6      58     882          138           1 credit_card
##  7        7      87     915          149           1 cash
##  8        8      22     761          292           2 cash
##  9        9      64     914          266           2 debit
## 10       10      52     788          146           1 credit_card
## # … with 4,927 more rows, and 2 more variables: created_at <dttm>,
## #   Amount_per_Order <dbl>
```

```
describe(cleaned_shopify_df$order_amount)
```

```
## cleaned_shopify_df$order_amount
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     4937        0      252        1    302.6    173.4      122      133
##      .25      .50      .75      .90      .95
##      163      284      387      520      592
##
## lowest :   90   94  101  111  112, highest: 1056 1064 1086 1408 1760
```

```
summary(cleaned_shopify_df$order_amount)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    90.0   163.0   284.0   302.6   387.0  1760.0
```

```
mean(cleaned_shopify_df$order_amount)
```

```
## [1] 302.5805
```

```
describe(cleaned_shopify_df$Amount_per_Order)
```

```
## cleaned_shopify_df$Amount_per_Order
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##     4937        0       57    0.999    151.8    29.53      112      117
##      .25      .50      .75      .90      .95
##      132      153      166      181      190
##
## lowest :  90  94 101 111 112, highest: 193 195 196 201 352
```

```
summary(cleaned_shopify_df$Amount_per_Order)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    90.0   132.0   153.0   151.8   166.0   352.0
```

```
#Mean

MeanOfAPO<-mean(cleaned_shopify_df$Amount_per_Order)
meadianAPO<-median(cleaned_shopify_df$Amount_per_Order)

# Create the function.
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Create the vector with numbers.
result <- cleaned_shopify_df$Amount_per_Order

# Calculate the mode using the user function.
modeAPO <- getmode(result)
print("Mode Value")
```

```
## [1] "Mode Value"
```

```
print(modeAPO)
```

```
## [1] 153
```

```
print("meadian Value")
```

```
## [1] "meadian Value"
```

```
print(meadianAPO)
```

```
## [1] 153
```

```
print("Mean Value")
```

```
## [1] "Mean Value"
```

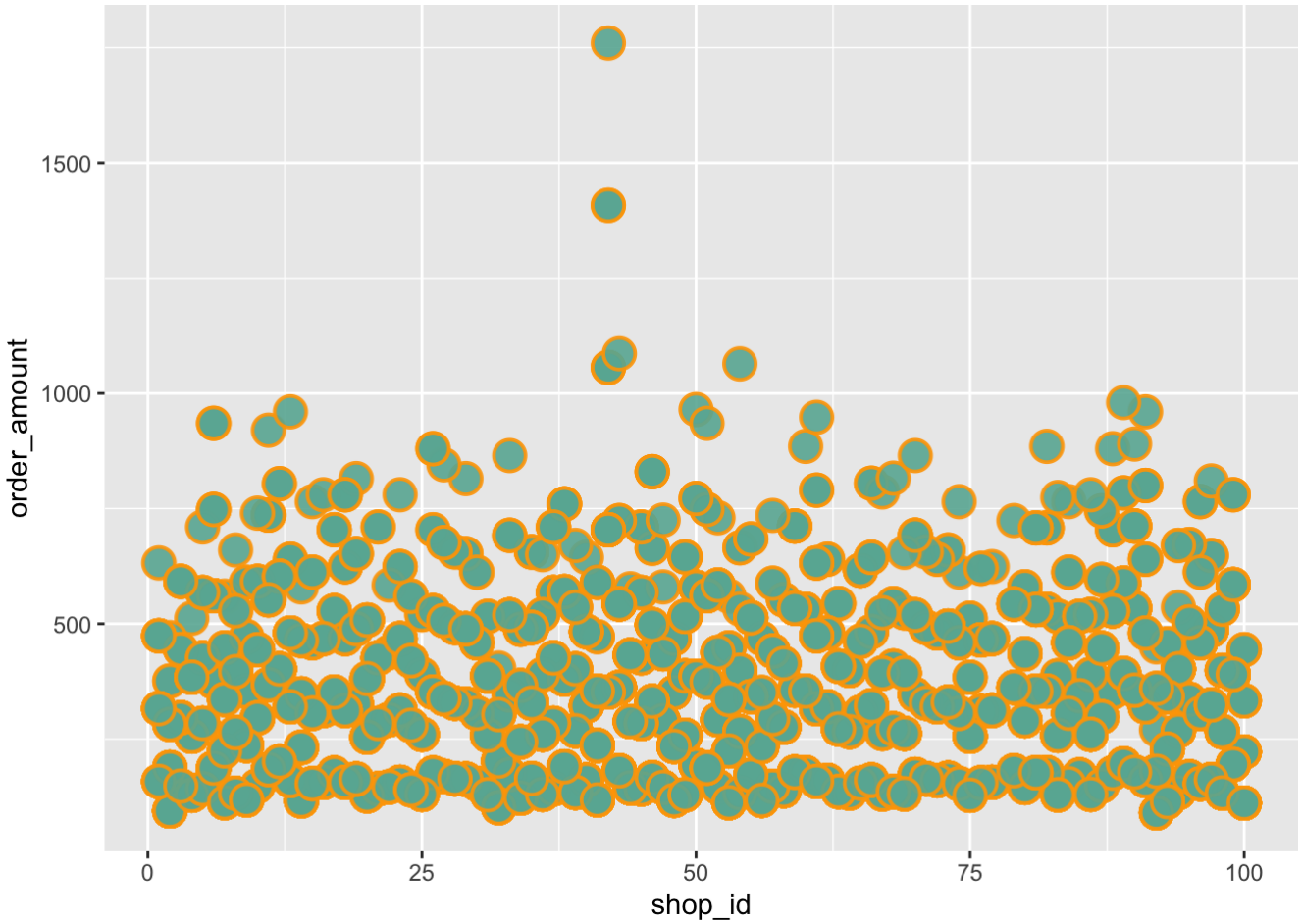
```
print( MeanOfAPO)
```

```
## [1] 151.7885
```

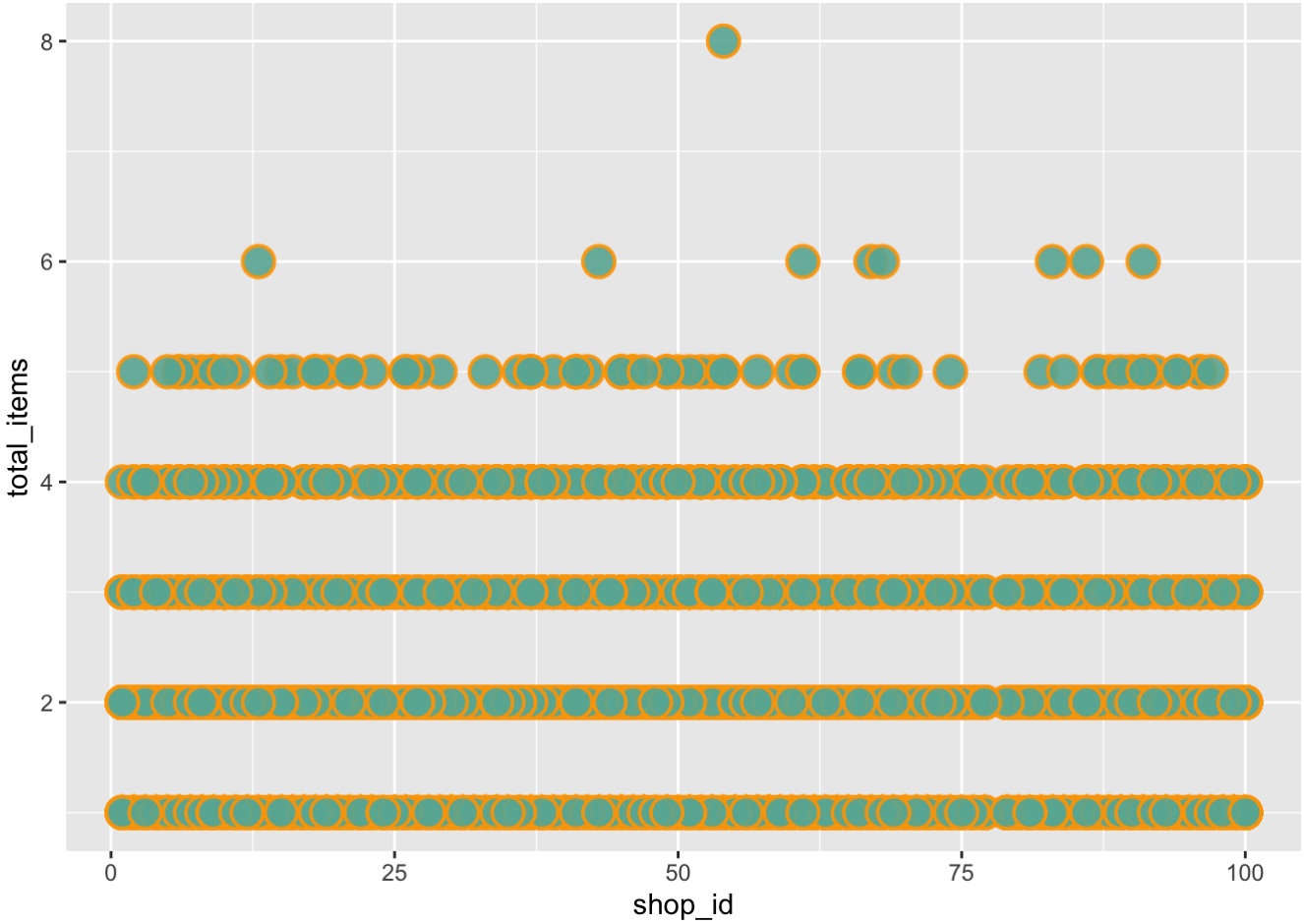
Ans) It can be Concluded that the mean average can be around $152

Visualizing the data after removing the potential outliers

```
ggplot(cleaned_shopify_df, aes(x=shop_id, y=order_amount)) +
    geom_point(
        color="orange",
        fill="#69b3a2",
        shape=21,
        alpha=0.9,
        size=5,
        stroke = 1
        )
```




```
ggplot(cleaned_shopify_df, aes(x=shop_id, y=total_items)) +
    geom_point(
        color="orange",
        fill="#69b3a2",
        shape=21,
        alpha=0.9,
        size=5,
        stroke = 1
        )
```



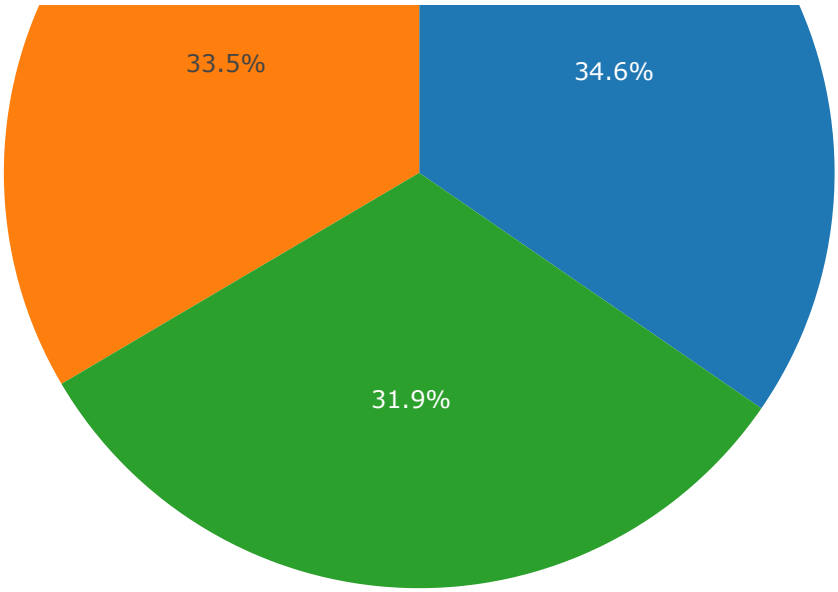
Q. Which type of payment is done more

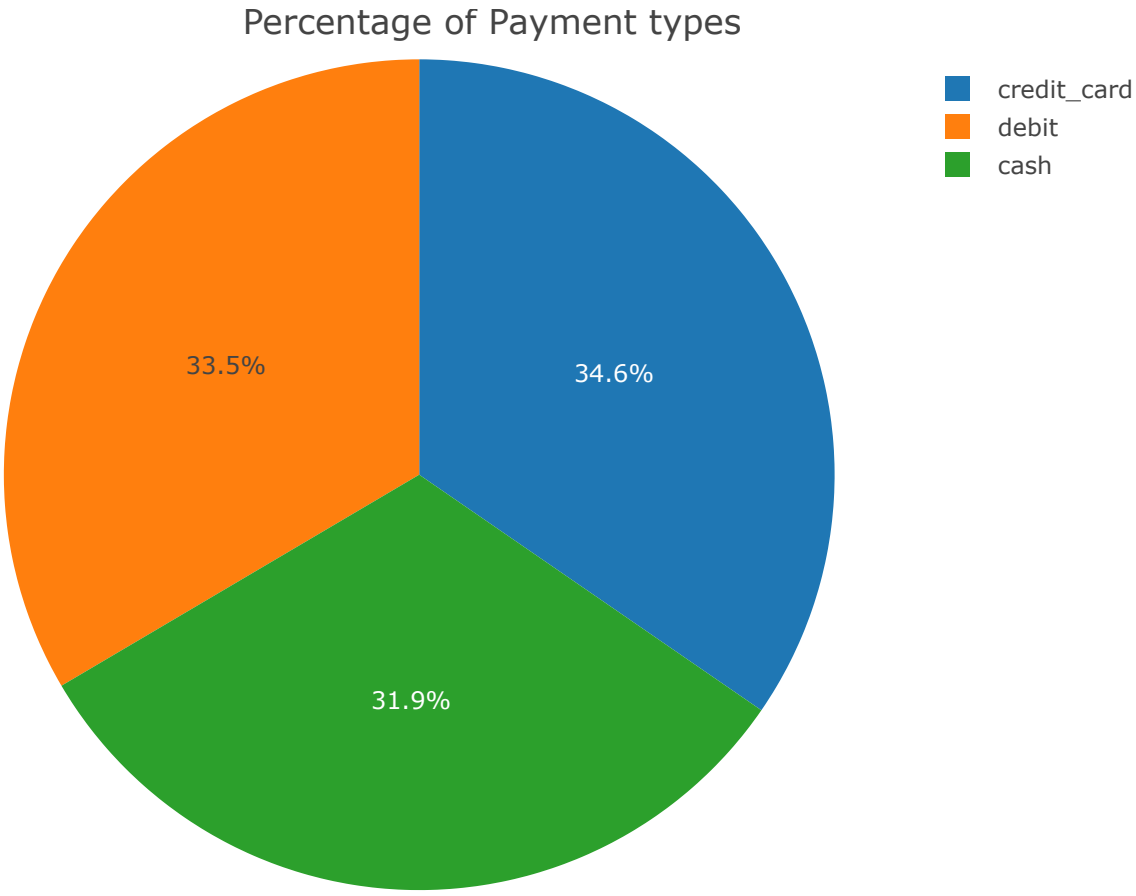
```
Payment_Menthod_count<- cleaned_shopify_df %>% count(payment_method, name = 'Num_Of_Paym
ents',sort = TRUE)
Payment_Menthod_count
```

```
## # A tibble: 3 × 2
##   payment_method Num_Of_Payments
##   <chr>                    <int>
## 1 credit_card               1708
## 2 debit                     1653
## 3 cash                      1576
```

Including Plots

You can also embed plots, for example:





Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.