Data Science Challenge Solution

QiHan Zhao

1/14/2022

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Question 1

On Shopify, we have exactly 100 sneaker shops, and each of these shops sells only one model of shoe. We want to do some analysis of the average order value (AOV). When we look at orders data over a 30 day window, we naively calculate an AOV of $3145.13. Given that we know these shops are selling sneakers, a relatively affordable item, something seems wrong with our analysis.

```
## Rows: 5000 Columns: 7

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): payment_method, created_at
## dbl (5): order_id, shop_id, user_id, order_amount, total_items

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

A. Think about what could be going wrong with our calculation. Think about a better way to evaluate this data.

First, I want to check if there is any missing value in the columns 'order_amount' and 'total_items', since the two directly tie to the calculation of AOV:

```
shopData%>%
  select(order_amount, total_items)%>%
  summarize(NA.in.Order.Amount = sum(is.na(order_amount)),
            NA.in.Total.items = sum(is.na(total_items)))
```

```
## # A tibble: 1 x 2
##   NA.in.Order.Amount NA.in.Total.items
##                <int>             <int>
## 1                  0                 0
```

Since there are no missing values, I would proceed to check for the range of values the variable 'order_amount' take:

```
summary(shopData$order_amount)
```

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

I also checked the range of values for 'total_items':

```
summary(shopData$total_items)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    1.000    1.000    2.000    8.787    3.000 2000.000
```

Based on the two outputs, I would propose that one possible cause to the unrealistically high AOV is that the value is calculated through averaging total earnings over the numbers of entries in the dataset, instead of the actual number of orders. Because we have some extreme values in the trade deals, calculating AOV in the former way would pull the value to a much higher number than it should be.

Alternative 1:

The simplest alternative is to average the total amount over the total number of orders in the 30-days window:

```
sum(shopData$order_amount)/sum(shopData$total_items)
```

```
## [1] 357.9215
```

This value is much more realistic than the given $3145.13, but it treats all 100 shops as one entity and fails to take into account how "well" each store is doing. Therefore, the following alternatives are all based on averaging each store's own AOV, which I believe reflects the market better for the shop owners.

```
aov <- shopData%>%
  group_by(shop_id)%>%
  summarize(totalOrders = sum(total_items),
            aov = sum(order_amount)/totalOrders)
```

Alternative 2:

In this alternative, I will calculate the metric using the average of the shops' individual AOV. But first, I will check to see if any shop is selling very expensive sneakers. These shops are likely to be the minority in the market, but their AOV may pull the metric up depending on how high the former is:

```
aov%>%
  arrange(desc(aov))
```

```
## # A tibble: 100 x 3
##    shop_id totalOrders   aov
##      <dbl>       <dbl> <dbl>
##  1      78          88 25725
##  2      42       34063   352
##  3      12          93   201
##  4      89         118   196
##  5      99          94   195
##  6      50          92   193
##  7      38          72   190
##  8       6         121   187
##  9      51          89   187
## 10      11          95   184
## # ... with 90 more rows
```

As we can see from the output, shop 78's AOV is $25725, which is much higher than the rest of the shops. Since it is just one of the shops, I will calculate the average after dropping it from the dataset.

```
aov%>%
  filter(aov < 1000)%>%
  summarize(Average.AOV.of.All = trunc(mean(aov)))
```

```
## # A tibble: 1 x 1
##   Average.AOV.of.All
##                <dbl>
## 1                152
```

Alternative 2:

Assuming the shop owners would also like

```
df <- aov%>%
  group_by(aov)%>%
  summarize(n=n())
```