Shopify Data Science Challenge

We start by loading the dataset to begin our analysis. An convert the column for date to the correct data type. We also clean the dataset by removing any rows that contain null values.

```
shopper_data <-
  read_csv("2019 Winter Data Science Intern Challenge Data Set - Sheet1.csv")
```

```
## 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.
```

```
shopper_data$created_at <-
  as.POSIXct(shopper_data$created_at, format = "%Y-%m-%d %H:%M:%S")
shopper_data <- na.omit(shopper_data)
head(shopper_data)
```

```
## # A tibble: 6 x 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
## # ... with 1 more variable: created_at <dttm>
```

To gain additional insight, we print certain key values that will give us addional insight on the dataset.

```
# Print the shape of the datset
dim(shopper_data)
```

```
## [1] 4994    7
```

```
# Determine the range of dates for the dataset
shopper_data %>%
  mutate(created_date = as.Date(as.character(created_at))) %>%
  summarise(min = min(created_date), max = max(created_date))
```

```
## # A tibble: 1 x 2
##   min        max
##   <date>     <date>
## 1 2017-03-01 2017-03-30
```

```
# Get the AOV and average item cost
round(mean(shopper_data$order_amount), 2)
```

```
## [1] 3148.55
```

```
round(mean(shopper_data$order_amount/shopper_data$total_items), 2)
```

```
## [1] 387.98
```

```
# Orders over 1000$
big_orders <- shopper_data[shopper_data$order_amount > 1000,]
dim(big_orders)
```

```
## [1] 71  7
```

```
big_orders
```

```
## # A tibble: 71 x 7
##    order_id shop_id user_id order_amount total_items payment_method
##       <dbl>   <dbl>   <dbl>        <dbl>       <dbl> <chr>
##  1       16      42     607       704000        2000 credit_card
##  2       61      42     607       704000        2000 credit_card
##  3      161      78     990        25725           1 credit_card
##  4      491      78     936        51450           2 debit
##  5      494      78     983        51450           2 cash
##  6      512      78     967        51450           2 cash
##  7      521      42     607       704000        2000 credit_card
##  8      618      78     760        51450           2 cash
##  9      692      78     878       154350           6 debit
## 10      939      42     808         1056           3 credit_card
## # ... with 61 more rows, and 1 more variable: created_at <dttm>
```

From the printed values, we see that the dataset indeed spans a month of time. Furthermore, we obtain similar AOV as presented in the problem description. Upon closer investigation of the dataset, we see that there are a few very large orders that skew the dataset. In fact, most orders are under 1000$, so we decide to remove all orders above 1000$ to get a more accurate representation of the order value.

```
# Orders under 1000$
small_orders <- shopper_data[!shopper_data$order_amount > 1000,]
dim(small_orders)
```

```
## [1] 4923    7
```

```r
head(small_orders)
```

```
## # A tibble: 6 x 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
## # ... with 1 more variable: created_at <dttm>
```

```r
# Get the AOV and average item cost
round(mean(small_orders$order_amount), 2)
```

```
## [1] 301.07
```

```r
round(mean(small_orders$order_amount/small_orders$total_items), 2)
```

```
## [1] 151.5
```

After cleaning the dataset, we obtain an AOV of 301.07$ which is much more reasonable. Finally, a better metric one might want to look at in this scenario is the median of the order amounts. Since large orders skew the data, choosing the median might yield a more accurate representation of what a typical order looks like.

```r
# Get median order value
round(median(shopper_data$order_amount), 2)
```

```
## [1] 284
```

We obtain a median of 284$ which is close to the AOV of 301.07$ obtained previously.