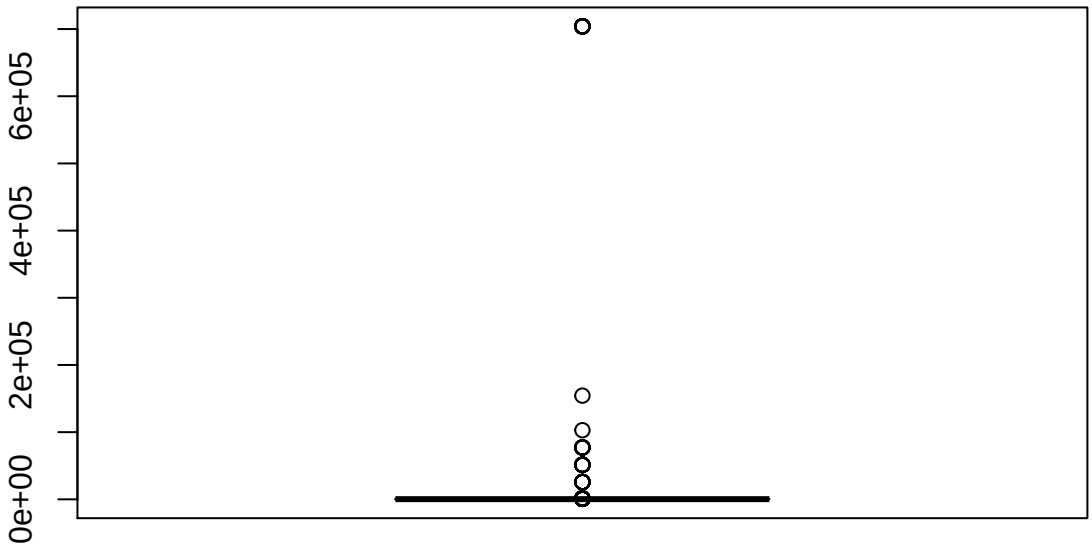
shopify_interview_challenge

Thomas Xu

5/10/2022

Question 1

```
# Exploration
boxplot(shoes$order_amount)
```




```
summary(shoes$order_amount)
```

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

```
mean(shoes$order_amount) # this is the reported amount
```

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

```
summary(shoes$total_items)
```

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

All of these exploratory outputs suggest there is something wrong with the data.

a

On the surface level, there is nothing wrong with the answer provided. Under the assumption that the data provided is perfect, the actual average order value is 3145.128. However, it is correct to be suspicious of such a high average. And, a simple glance at the plots and tables of the orders make it immediately obvious something is awry. Nobody should be spending 6 figures on shoes. Since the variable in question is order

price, the analysis could be broken down into looking at the input two variables of order price: price per pair of shoes and number of pairs ordered.

```
store_sale <- shoes %>%
  mutate(price_per_shoe = order_amount / total_items) %>%
  group_by(price_per_shoe,shop_id) %>%
  summarise(orders = n())
```

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

```
head(store_sale[order(store_sale$price_per_shoe, decreasing = TRUE),])
```

```
## # A tibble: 6 x 3
## # Groups:   price_per_shoe [6]
##   price_per_shoe shop_id orders
##            <dbl>   <int>  <int>
## 1          25725      78     46
## 2            352      42     51
## 3            201      12     53
## 4            196      89     61
## 5            195      99     54
## 6            193      50     44
```

First, I was curious about the price of the shoes that each store was selling. I noticed that in the explanation of the data every store only sells one shoe so I created a table that shows the price per shoe of each store. This was partially motivated to check the validity of the data set as well; more than one price per shoe for a shop would indicate some fault in the data. Thankfully, every shop had only one price, but this is where I first found something suspicious. Shop 78, the shop with corresponding shop_ID 78, was selling shoes for 25725 units of money (from here on out I will just assume dollars). I am a sneakerhead. There is absolutely no reason for the price of a pair of shoes to be this high, especially if the price is fixed. It is possible there is some custom sneaker that can make you run faster and walk on water, but the price would most likely be variable. So if given the chance, confirming that this shop actually sells shoes would be essential.

```
head(shoes[order(shoes$total_items, decreasing = TRUE), -1], keepnums = FALSE)
```

```
##      shop_id user_id order_amount total_items payment_method         created_at
## 16        42     607       704000        2000    credit_card 2017-03-07 4:00:00
## 61        42     607       704000        2000    credit_card 2017-03-04 4:00:00
## 521       42     607       704000        2000    credit_card 2017-03-02 4:00:00
## 1105      42     607       704000        2000    credit_card 2017-03-24 4:00:00
## 1363      42     607       704000        2000    credit_card 2017-03-15 4:00:00
## 1437      42     607       704000        2000    credit_card 2017-03-11 4:00:00
```

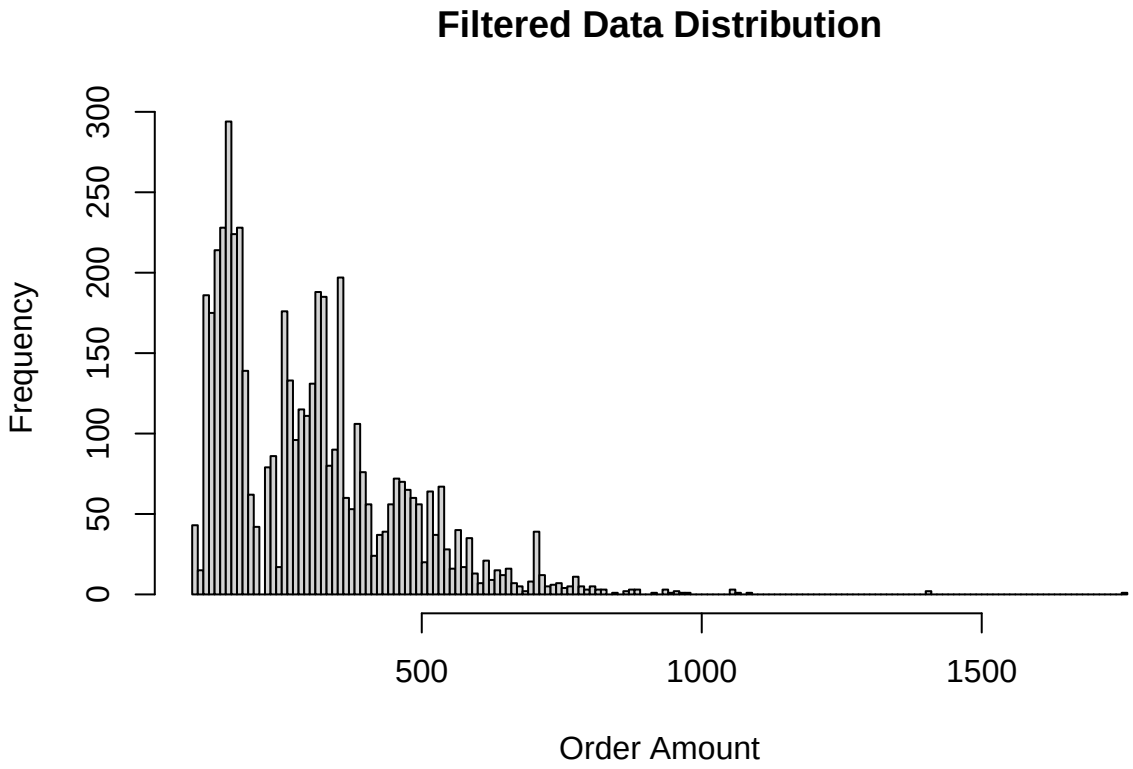
```
cat("Number of suspicous orders: ", sum(shoes$order_amount == 704000))
```

```
## Number of suspicous orders:  17
```

When looking at the rows of the orders with the largest order amount. We see some pretty concerning things. There are many things wrong with the top 17 orders. All of them are ordered by the same user (607), same store (42), same total items, using a credit card, and all of the orders were placed at exactly 4 o'clock. All of these orders indicate that these orders are not authentic. Since we do not have data on canceled orders, we can not know for sure that these orders never go through. However, both the amount of money spent on each order and the usage of credit cards suggest that none of these orders went through. At the very least, these orders were not made by an actual human being. Other orders at this store seem relatively normal, so it may just be the user that is suspicious.

```
suggest_data <- shoes %>% filter(shop_id != 78, user_id != 607)
removed_data <- shoes %>% filter(shop_id == 78 | user_id == 607)
```

```
hist(suggest_data$order_amount, xlab = "Order Amount",
     main = "Filtered Data Distribution",
     breaks = seq(min(suggest_data$order_amount),
                  max(suggest_data$order_amount), by = 10))
```




```
summary_order <- summary(shoes$order_amount)
upperIQR <- summary_order["3rd Qu."] +
  1.5*(summary_order["3rd Qu."] - summary_order["1st Qu."])
sum(upperIQR < removed_data$order_amount)/count(removed_data)
```

```
##   n
## 1 1
```

After I removed the orders from the suspicious shop and suspicious user, we get a much more reasonable distribution of order prices. This means I removed 63 rows of data which is 1.26% of the original data set. Without further information, using this filtered data set may be the safest bet to getting a representative population. Also note that all the data points filtered are outliers, with the outliers selected using the IQR method.

b

```
## Summary Statistics of Cleaned Data:
```

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

```
## Summary Statistics of Original Data:
```

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

As we can see, the majority of the summary statistics are the same between the cleaned and uncleaned data

sets. The major difference between the two is the mean or Average Order Value. The clean data gives us an AOV of 302.6, a much more reasonable AOV.

Another metric that can be considered is the median, a different measure for the middle of the data. The median is much less susceptible to large variations due to outliers, a huge benefit compared to the mean. As we can see here, we have the same median in both data sets.

c

The value most representative of the data is **302.60**, the average order value after cleaning the data. The cleaned data is the original data without orders from shop 78 and without orders made by user 607. I believe that the average is still the best metric to get a fundamental understanding of the data. However, I believe the best way to understand the data is to use both the mean and median to get a more comprehensive view of the shape of the data, but the question just asks for one metric.

My main choices were between the median and different variations of cleaning the data and taking its mean. There are benefits and drawbacks to both metrics. The median is robust. We don't need that much prior knowledge of the data to get a good sense of what a typical order looks like. Here, the median is 284, but believe that number is too low. Data is very right-skewed, which causes the median to be lower than the mean. The median is very powerful in the situation where we don't need precise representations of the data and we need a figure fast.

On the other hand, an accurate representation of the data using the mean requires more time. In this data set, I believe there were two mistakes: a fake user, and an incorrectly classified store. If I am correct, then my metric is a better representation of the data, but, if wrong, then we are better off using the mean. Thus, highlighting the faults of using mean in this situation. I lack domain knowledge and am susceptible to making wrong assumptions.

2

a

```
SELECT COUNT( * ) as "Number of orders shipped by Speedy Express"
FROM Orders
WHERE ShipperID = 1
```

Speedy Express had 54 orders.

b

```
SELECT LastName,
       COUNT(OrderID) AS OrderCount
FROM Employees
LEFT JOIN Orders ON Employees.EmployeeID = Orders.EmployeeID
GROUP BY Employees.EmployeeID
ORDER BY OrderCount DESC
```

Employee Peacock had the most orders, with 40 orders total.

c

```
SELECT ProductName,
       SUM(Quantity) AS TotalQuantity
FROM OrderDetails
LEFT JOIN Orders ON OrderDetails.OrderID = Orders.OrderID
LEFT JOIN Customers ON Orders.CustomerID = Customers.CustomerID
LEFT JOIN Products ON OrderDetails.ProductID = Products.ProductID
WHERE Country = "Germany"
```

```
GROUP BY Products.ProductID
ORDER BY TotalQuantity DESC
```

Boston Crab Meat was ordered the most by customers in Germany.