Shopify Data Science Intern Challenge

Melissa Van Bussel

Question 1

a) Think about what could be going wrong with our calculation. Think about a better way to evaluate this data.
b) What metric would you report for this dataset?
c) What is its value?

Solution 1a)

We begin by loading in the dataset.

```
# Load package to read Excel files
library(readxl)

# Load file
shopify <- read_excel("2019 Winter Data Science Intern Challenge Data Set.xlsx")
```

Next, we compute summary statistics on the `order_amount` variable:

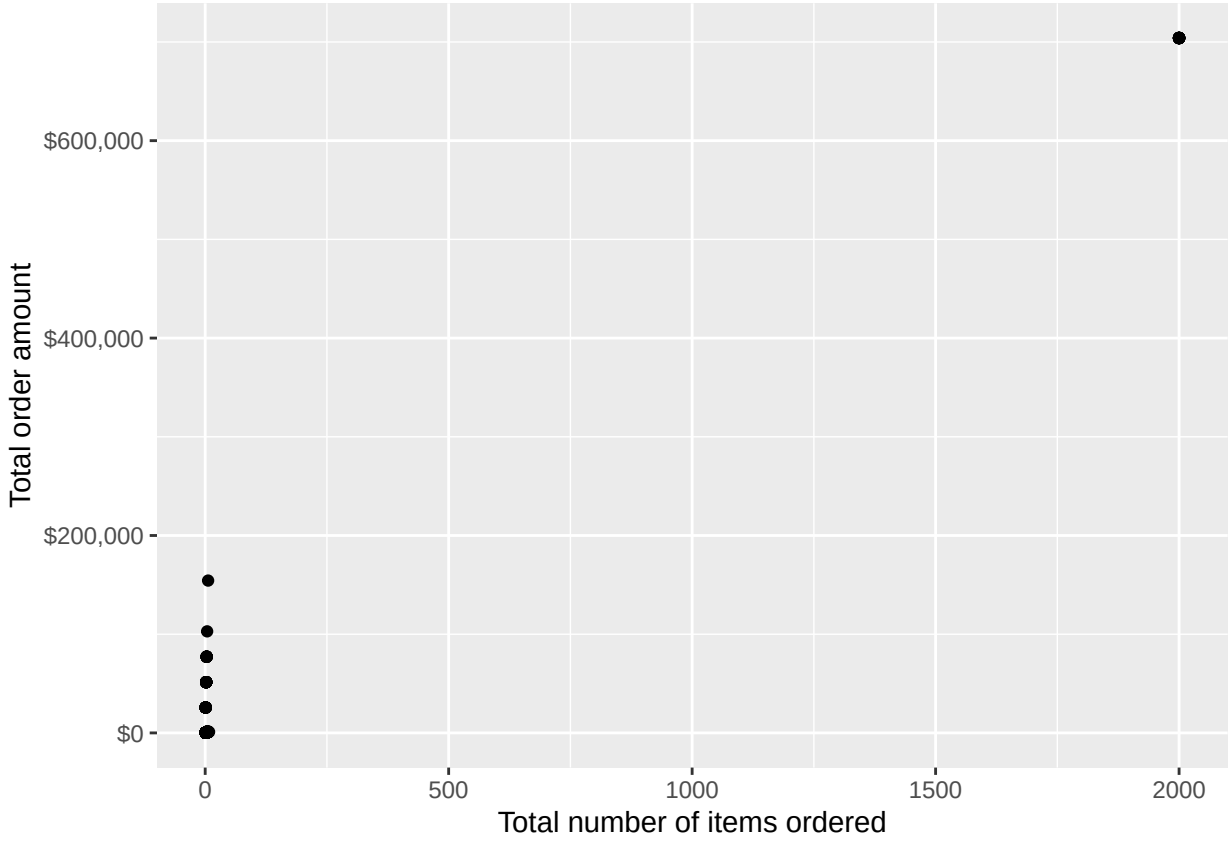
```
summary(shopify$order_amount)
```

```
##        Min.     1st Qu.      Median        Mean     3rd Qu.        Max.
##      90.000     163.000     284.000    3145.128     390.000 704000.000
```

We can see that the arithmetic average is indeed $3145.13, but we also see that the maximum value is achieved at $704,000. To get a better sense of what's going on, we can plot the order amounts as a function of the total number of items ordered.

```
# Load package for creating visually appealing plots
library(ggplot2)

# Plot order amount as a function of total items ordered
ggplot(aes(x = total_items, y = order_amount), data = shopify) +
  geom_point() +
  labs(x = "Total number of items ordered", y = "Total order amount") +
  scale_y_continuous(labels = scales::dollar_format())
```



We can see some obvious outliers. There are 17 orders which contain 2000 pairs of shoes and came out to $704,000 (see below). These orders could be to a wholesale retailer who only sells shoes in bulk, which is different from the other stores in the dataset. Additionally, there are some stores which sell very expensive shoes – even as high as $25,725 (see below).

```
# Calculate how many orders contain 2000 items and cost $704,000
length(which(shopify$order_amount == 704000 & shopify$total_items == 2000))
```

```
## [1] 17
```

```
# Calculate the price of the most expensive pair of shoes
max(shopify$order_amount / shopify$total_items)
```

```
## [1] 25725
```

Since we know that each observation is weighted equally in the computation of the arithmetic average, these outliers dramatically pull the mean upwards (particularly the 17 orders of $704,000). Therefore, the lack of consideration for the outliers is what is going wrong with this calculation. Rather than looking at the average order value alone, a better way to evaluate this data is to explore the distribution of the variables of interest, both numerically and visually. Specifically, we looked at the order amounts as a function of the number of items ordered in order to gain more valuable insight.

Solution 1b)

The metric I would report for this dataset is the median order value. Just like the average order value, the median order value is a single summary value that gives an idea of the typical order being made. Unlike the arithmetic average, though, the median is more robust to outliers. In our case, we know that there are many outliers and thus the better choice is the median.

Solution 1c)

The value of the median order value is $284, as shown below.

```
median(shopify$order_amount)
```

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

Question 2

a) How many orders were shipped by Speedy Express in total?
b) What is the last name of the employee with the most orders?
c) What product was ordered the most by customers in Germany?

Solution 2a)

We begin by performing an inner join on the `Order` and `Shippers` tables, joining on `ShipperID`. We then filter down to only the observations that come from Speedy Express. Finally, we count the number of orders that meet these requirements. The query is below:

```
SELECT COUNT (orderid)
FROM   (SELECT *
        FROM   orders
               INNER JOIN shippers
                       ON orders.shipperid = shippers.shipperid)
WHERE  shippername = "Speedy Express";
```

The final numerical answer returned by this query is 54.

Solution 2b)

We begin by performing an inner join on the `Employees` and `Orders` tables, joining on `EmployeeID`. Then, we group by `EmployeeID` and count the number of observations in each group, place in descending order, retain only the top result, and select the employee's last name. The query is below:

```
SELECT lastname
FROM   (SELECT lastname,
               COUNT(*) AS counts
        FROM   (SELECT *
                FROM   employees
                       INNER JOIN orders
                               ON employees.employeeid = orders.employeeid)
        GROUP  BY employeeid
        ORDER  BY counts DESC
        LIMIT  1);
```

The final answer returned by this query is Peacock.

Solution 2c)

First, we perform a series of inner joins (on the `Customers`, `Orders`, `OrderDetails`, and `Products` tables), using aliases along the way for efficiency. Next, we filter down to only the items that were ordered by German customers. Then, we add up the

quantities of each product that were ordered, sort by this value (in descending order), retain only the top result, and then select the name of the product. The query is below:

```
SELECT productname
FROM   (SELECT productname,
               SUM(quantity) AS sums
        FROM   (SELECT productname,
                       quantity
                FROM   (SELECT T1.customerid,
                               T2.orderid,
                               T3.productid,
                               T3.quantity,
                               T4.country,
                               T5.productname
                        FROM   customers T1
                               INNER JOIN orders T2
                                       ON T1.customerid = T2.customerid
                               INNER JOIN orderdetails T3
                                       ON T2.orderid = T3.orderid
                               INNER JOIN customers T4
                                       ON T1.customerid = T4.customerid
                               INNER JOIN products T5
                                       ON T3.productid = T5.productid)
                WHERE  country = "Germany")
        GROUP  BY productname
        ORDER  BY sums DESC
        LIMIT  1);
```

The final answer returned by this query is Boston Crab Meat.