Summer 2022 Data Science Intern Challenge

January 16, 2022

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.

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

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

[54]:
```
df = pd.read_csv("2019 Winter Data Science Intern Challenge Data Set - Sheet1.
↪csv")
df.head()
```

[54]:

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
0	1	53	746	224	2	cash	2017-03-13 12:36:56
1	2	92	925	90	1	cash	2017-03-03 17:38:52
2	3	44	861	144	1	cash	2017-03-14 4:23:56
3	4	18	935	156	1	credit_card	2017-03-26 12:43:37
4	5	18	883	156	1	credit_card	2017-03-01 4:35:11

[5]:
```
df['order_amount'].mean()
```

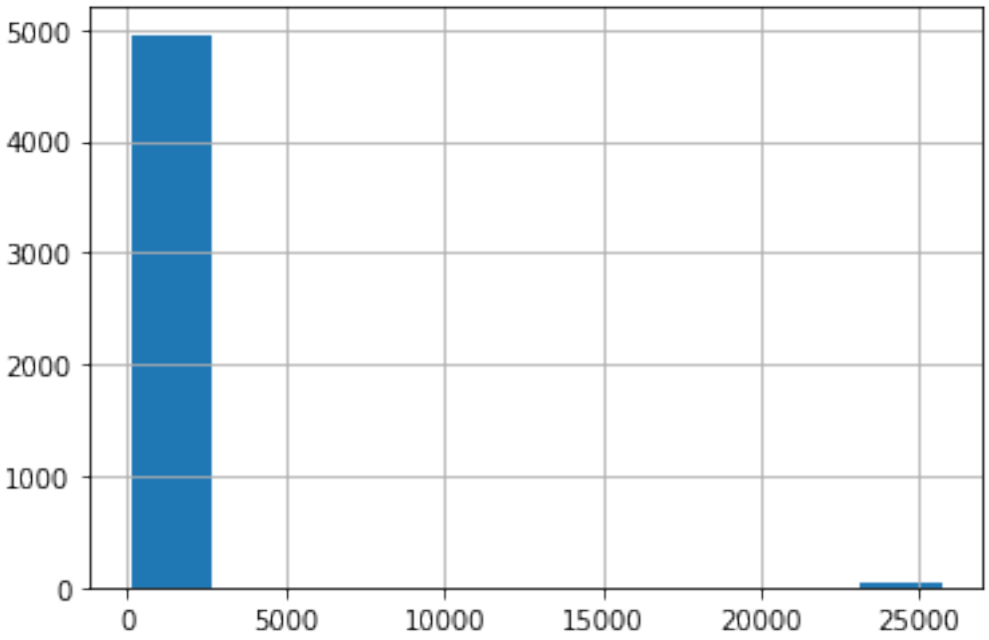
[5]: 3145.128

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

Here introduces the problem! In this case, the AOV is actually calculating the average value for the `order_amount`, but disregarding the item number within the order. AOV value could be largely influenced by the value of `total_items`. For instance, multiple orders with 100 or more items will result in a higher AOV, but that does not necessarily imply that sneakers are not affordable.

[56]:
```
df['new_aov']=df['order_amount']/df['total_items']
mean_value = df['new_aov'].mean()
median_value = df['new_aov'].median()
max_value = df['new_aov'].max()
min_value = df['new_aov'].min()
print("mean_value = ", mean_value)
print("median_value = ", median_value)
print("max_value = ", max_value)
print("min_value = ", min_value)
df['new_aov'].hist()
outlier = df.query('new_aov >= 1000')
print('number of extreme case = ', len(outlier), 'and percentage of extreme␣
↪case = ', (len(outlier)/len(df['new_aov'])*100.0))
```

```
mean_value =  387.7428
median_value =  153.0
max_value =  25725.0
min_value =  90.0
number of extreme case =  46 and percentage of extreme case =
0.9199999999999999
```



For a better understanding of the price of sneakers, we can calculate the average value of "the average amount per order", or we can refer to it as the average price per pair. As a result, the average value seems more appropriate for a sneakers shop, however, the maximum amount for each order is 25725.0, which is very high for a pair of sneakers. Here are two possible explanations:

- There may have been an error when collecting the data, which means we should remove those extremely large values.
- It is possible to find some extreme rare designer collections or collaborations with luxury brands with prices exceeding 20000 dollars. However, these examples should still be regarded as outliers.

There are 46 cases with an average price exceeding 1000 dollars, which is equal to 0.92% among all the orders. We should remove those outlier and recalculate the average price for a pair of sneakers.

[60]:
```
df_filtered = df.query('new_aov <= 1000')
print("mean_value = ", df_filtered['new_aov'].mean())
print("median_value = ", df_filtered['new_aov'].median())
```

```
mean_value =  152.47557529269278
median_value =  153.0
```

It seems like we have a reasonable mean value equal to 152.4 and it is close to the median value. Yet, we also notice that we indeed remove those high price outlier

[62]:
```
df_filtered.sort_values('order_amount', ascending=False)[:25]
```

[62]:

	order_id	shop_id	user_id	order_amount	total_items	payment_method	\
60	61	42	607	704000	2000	credit_card	
3332	3333	42	607	704000	2000	credit_card	
4056	4057	42	607	704000	2000	credit_card	
1362	1363	42	607	704000	2000	credit_card	
1436	1437	42	607	704000	2000	credit_card	
2153	2154	42	607	704000	2000	credit_card	
15	16	42	607	704000	2000	credit_card	
2297	2298	42	607	704000	2000	credit_card	
4646	4647	42	607	704000	2000	credit_card	
1562	1563	42	607	704000	2000	credit_card	
1104	1105	42	607	704000	2000	credit_card	
1602	1603	42	607	704000	2000	credit_card	
2835	2836	42	607	704000	2000	credit_card	
520	521	42	607	704000	2000	credit_card	
4868	4869	42	607	704000	2000	credit_card	
4882	4883	42	607	704000	2000	credit_card	
2969	2970	42	607	704000	2000	credit_card	
1364	1365	42	797	1760	5	cash	
1471	1472	42	907	1408	4	debit	
1367	1368	42	926	1408	4	cash	
3538	3539	43	830	1086	6	debit	

```
4141      4142       54      733          1064            8          debit
3513      3514       42      726          1056            3          debit
2987      2988       42      819          1056            3           cash
938        939       42      808          1056            3    credit_card

                created_at  new_aov
60      2017-03-04 4:00:00    352.0
3332    2017-03-24 4:00:00    352.0
4056    2017-03-28 4:00:00    352.0
1362    2017-03-15 4:00:00    352.0
1436    2017-03-11 4:00:00    352.0
2153    2017-03-12 4:00:00    352.0
15      2017-03-07 4:00:00    352.0
2297    2017-03-07 4:00:00    352.0
4646    2017-03-02 4:00:00    352.0
1562    2017-03-19 4:00:00    352.0
1104    2017-03-24 4:00:00    352.0
1602    2017-03-17 4:00:00    352.0
2835    2017-03-28 4:00:00    352.0
520     2017-03-02 4:00:00    352.0
4868    2017-03-22 4:00:00    352.0
4882    2017-03-25 4:00:00    352.0
2969    2017-03-28 4:00:00    352.0
1364    2017-03-10 6:28:21    352.0
1471   2017-03-12 23:00:22    352.0
1367    2017-03-13 2:38:34    352.0
3538   2017-03-17 19:56:29    181.0
4141   2017-03-07 17:05:18    133.0
3513   2017-03-24 17:51:05    352.0
2987    2017-03-03 9:09:25    352.0
938    2017-03-13 23:43:45    352.0
```

There appears to be a reasonable mean value of 152.4 and it is close to the median value. Nevertheless, we also notice that we have indeed removed those outliers with extremely high prices, but some orders contain large item numbers that could be due to duplication introduced by system errors. All of these orders are from the same shop and it is rare for a sneaker shop to offer more than ten thousand pairs of the same model of shoe. Therefore, we also decided to remove these cases.

[59]:
```
df_filtered = df_filtered[df_filtered['total_items'] != 2000]
print("mean_value = ", df_filtered['new_aov'].mean())
print("median_value = ", df_filtered['new_aov'].median())
```

```
mean_value =  151.7885355479036
median_value =  153.0
```

0.2 2. What metric would you report for this dataset?

The `new_aov` with outliers removed will be reported as the key metric along with the `order_id`, `order_amount`, and `total_items` to assist managers or other analysts who wish to continue tracking or evaluating the results.

0.3 3 What is its value?

The final results show a mean value of `new_aov` is 151.78 and median value of `new_aov` is 153. Both are reasonable price for sneakers.

[ ]:

Question 2: For this question you'll need to use SQL. Follow this link to access the data set required for the challenge. Please use queries to answer the following questions. Paste your queries along with your final numerical answers below.

a. How many orders were shipped by Speedy Express in total?

```
SELECT COUNT(*)
FROM Orders
WHERE ShipperID == 1;
```

Ans: 54

b. What is the last name of the employee with the most orders?

```
SELECT Employees.FirstName || " " || Employees.LastName Employee,
COUNT(Orders.OrderID) NumberOfOrder
FROM Orders
INNER JOIN Employees ON Employees.EmployeeID =  Orders.EmployeeID
GROUP BY Employee
ORDER BY NumberOfOrder DESC
```

Ans: Peacock

c. What product was ordered the most by customers in Germany?

```
SELECT SUM(OrderDetails.Quantity) NumberOfOrder, Customers.Country Country,
Products.ProductName ProductName
FROM Orders
INNER JOIN Customers ON Customers.CustomerID = Orders.CustomerID
INNER JOIN OrderDetails ON OrderDetails.OrderID = Orders.OrderID
INNER JOIN Products ON Products.ProductID = OrderDetails.ProductID
WHERE Country == 'Germany'
GROUP BY ProductName
ORDER BY NumberOfOrder DESC
```

Ans: Boston Crab Meat