Data Science Shopify Challenge

Ricardo Diaz

Importing Data

In [154…

```
import pandas as pd
import numpy as np
df = pd.read_csv('2019 Winter Data Science Intern Challenge Data Set.csv')
```

Question 1

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

Exploratory Data Analysis

In [155…

```
df.describe()
```

Out[155…

	order_id	shop_id	user_id	order_amount	total_items
count	5000.000000	5000.000000	5000.000000	5000.000000	5000.00000
mean	2500.500000	50.078800	849.092400	3145.128000	8.78720
std	1443.520003	29.006118	87.798982	41282.539349	116.32032
min	1.000000	1.000000	607.000000	90.000000	1.00000
25%	1250.750000	24.000000	775.000000	163.000000	1.00000
50%	2500.500000	50.000000	849.000000	284.000000	2.00000
75%	3750.250000	75.000000	925.000000	390.000000	3.00000
max	5000.000000	100.000000	999.000000	704000.000000	2000.00000

In the statistic description of the dataset, I can identify the max of order amount 704000$ exceed the interquartile range, this may be an outlier, and also the total items max of 2000 is too high comparing to its mean and it exceeds interquantile range, this may be an outlier too.

Outlier Detection

In [156…

```
df[df['order_amount']> 1000].head()
```

Out[156…

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
15	16	42	607	704000	2000	credit_card	3/7/2017 4:00
60	61	42	607	704000	2000	credit_card	3/4/2017 4:00
160	161	78	990	25725	1	credit_card	3/12/2017 5:56
490	491	78	936	51450	2	debit	3/26/2017 17:08
493	494	78	983	51450	2	cash	3/16/2017 21:39

We can see shop id 42 and shop id 78 are potential outliers, let dive in

In [157…

```
df[(df['shop_id']== 42) | (df['shop_id'] == 78)].head()
```

Out[157…

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
15	16	42	607	704000	2000	credit_card	3/7/2017 4:00
40	41	42	793	352	1	credit_card	3/24/2017 14:15
60	61	42	607	704000	2000	credit_card	3/4/2017 4:00
160	161	78	990	25725	1	credit_card	3/12/2017 5:56
308	309	42	770	352	1	credit_card	3/11/2017 18:14

After taking a deeper look Im going to drop shop_id 42 my two reasons are the order amount is absurly high and all this high order_amount come from the same user_id... who needs that many shoes. Im also dropping shop_id 78 because there is no way a shoes cost 25000$.

Naively Calculate AOV

In [158…

```
sum(df['order_amount'])/len(df['order_id']) # here is the problem with the calculation, explation later on
```

Out[158…

```
3145.128
```

Total Stores AOV

In [159…

```
df = df[(df['shop_id'] != 42) & (df['shop_id'] != 78)]
sum(df['order_amount'])/sum(df['total_items'])
```

Out[159…

```
150.4
```

Answer

The problem with our AOV of $3145.13 calculation is each order can have multiple order items and we are making the mistake of not taking this in consideration in number of orders for the AOV formula. We could do the total stores AOV like we did above but there is a better way to evaluate this data and the approach will be to take AOV per store which will give you a more detail insight per store. AOV is not particulary useful to examine your data because each store only sell one type of shoes, so I will need to create another feature to report this dataset.

AOV per Store

In [160…

```
shop_aov = df.groupby('shop_id',as_index=False).apply(lambda x: x['order_amount'].sum()/x['total_items'].sum())
shop_aov.columns = ['shop_id','AOV_per_store']
shop_aov.head(5)
```

Out[160…

	shop_id	AOV_per_store
0	1	158.0
1	2	94.0
2	3	148.0
3	4	128.0
4	5	142.0

b.What metric would you report for this dataset?

An important aspect of the business is to identify stores which are the best performers stores so the company can have a deeper look on what are they doing right. I will create a total order per store metric to add on the AOV per store feature to identify the best perfoming store.

In [161…

```
order_t = df[['shop_id','total_items']].groupby('shop_id',as_index=False).agg('sum','total_items').sort_values
order_t.columns = ['shop_id','total_order_store']
order_t.head(5)
```

Out[161…

	shop_id	total_order_store
12	13	136
81	84	132
51	53	130
69	71	130
78	81	128

c.What is its value?

In [162…

```
value = pd.merge(shop_aov, order_t, left_on='shop_id', right_on='shop_id', how='inner')
value.sort_values(by = 'total_order_store', ascending = False).head(3)
```

Out[162…

	shop_id	AOV_per_store	total_order_store
12	13	160.0	136
81	84	153.0	132
51	53	112.0	130

No one should use just AOV, thats why I created two metrics AOV per store and total_order_store , AOV tells you how much money in average a customer spends on your store and total_order_store which will show you which stores sell more items to customers. Here you can identify the best shop_id is 13, now you have more evidence to conclude this is your best performing store and now the company should take a deeper look at this particular store to see what are they doing right. In this scenario case since each store only sells 1 item the AOV will tell you the price of the shoes.

Question 2

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

```
SELECT *
FROM Shippers s JOIN Orders o
ON s.ShipperID = o.ShipperID
WHERE ShipperName = 'Speedy Express';
```

Answer: 54

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

```
SELECT o.EmployeeID , COUNT(OrderID), LastName
FROM Orders o JOIN Employees e
ON o.EmployeeID = e.EmployeeID
GROUP BY 1
ORDER BY COUNT(OrderID) DESC
```

Answer: Peacock is the last name of the employee with the most orders

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

```
SELECT ProductName, SUM(Quantity)
FROM Customers c JOIN Orders o ON c.CustomerID = o.CustomerID
JOIN OrderDetails od ON o.OrderID = od.OrderID
JOIN Products p ON od.ProductID = p.ProductID
WHERE Country = 'Germany'
GROUP BY 1
ORDER BY SUM(QUANTITY) DESC
```

Answer: Boston Crab Meat