Name – Nisarg Patel
Email – npatel@dal.ca
Phone – 902-880-5936

Data Science Intern Challenge Summer 2022 (Shopify)

Question 1

I have included my code with detailed analysis on the next page. Please have a look.

1. The AOV calculated here is incorrect because it uses wrong formula: total sum of order values/ total count of orders and misses out the fact that each order has multiple items inside it. After looking into the output from two approaches truncating the data set, removing outlier, and finding the median value of the new dataset is a better approach if removing data is not costly for other data analysis in the future with this dataset.

2. As you can observe in my detailed analysis, the use the median value of truncated data gives a better insight into data as the dataset follows skewed data distribution.

3. The last cell of my detailed analysis shows the output for median of the truncated data which is 276.

```
import pandas as pd
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Saving 2019 Winter Data Science Intern Challenge Data Set - Sheet1.csv to 2019 W

```
import io
df = pd.read_csv(io.BytesIO(uploaded['2019 Winter Data Science Intern Challenge Data S
```

```
df.head()
```

	order_id	shop_id	user_id	order_amount	total_items	payment_method	created_
0	1	53	746	224	2	cash	2017-03 12:36
1	2	92	925	90	1	cash	2017-03 17:38
2	3	44	861	144	1	cash	2017-03 4:23
							2017-03

Data analysi

given average order value = $3145.13;

1. first, looking into data analysis data below, it looks like there are two types of orders in the dataset. big amount of order (more than 1000 items) small amount of order (around between 1 - 5)
2. standard deviation value in the Amount column is very high which indicates, the amount values are spread out over a wider range from the mean(AOV) value. so using mean as average order value is not useful .
3. Amounts against no of orders list show some of the amounts is repeatable in many orders which shows some of the orders with the same amount of item is transferred between shops and user multiple time.

```
print("1. Data details for amount column")
print(df.order_amount.describe())
```

```
print("2. Data details for item column")
print(df.total_items.describe())
```

```
1. Data details for amount column
count      5000.000000
mean       3145.128000
std       41282.539349
min          90.000000
25%         163.000000
50%         284.000000
75%         390.000000
max      704000.000000
Name: order_amount, dtype: float64
2. Data details for item column
count    5000.00000
mean        8.78720
std       116.32032
min         1.00000
25%         1.00000
50%         2.00000
75%         3.00000
max      2000.00000
Name: total_items, dtype: float64
```

```
amounts_list = df.groupby(['order_amount']).size().reset_index(name='count').sort_valu
print(amounts_list);
```

```
     order_amount  count
257        704000     17
256        154350      1
255        102900      1
254         77175      9
253         51450     16
252         25725     19
251          1760      1
250          1408      2
249          1086      1
248          1064      1
247          1056      3
246           980      1
245           965      1
244           960      2
243           948      1
```

Way to evaluate data in a better approach.

1. Divide dataset into two datasets based on no of items ordered. After data analysis, we found that there are two types of orders. so we can divide the dataset into two parts based on their amount and no of items in one order. make two datasets (data with high amount orders and small amount order). calculate the median value for both dataset individually

2. Truncte data which are outliers. Trucnte data from dataset, which have a smaller amount compare to our lowest threshold value or higher than the highest threshold value. After truncating outliers we can use the median of the dataset as a metric for the dataset.

```
# Approach 1 (Divide dataset into two dataset based on no of items ordered.)
df_copy = df.copy()

small_amount_order_df = df_copy[(df_copy.total_items < 1000)];
print(small_amount_order_df.order_amount.describe())
print("******************")
big_amount_order_df = df_copy[(df_copy.total_items > 1000)];
print(big_amount_order_df.order_amount.describe())
```

```
count      4983.000000
mean        754.091913
std        5314.092293
min          90.000000
25%         163.000000
50%         284.000000
75%         390.000000
max      154350.000000
Name: order_amount, dtype: float64
******************
count        17.0
mean     704000.0
std           0.0
min      704000.0
25%      704000.0
50%      704000.0
75%      704000.0
max      704000.0
Name: order_amount, dtype: float64
```

```
def remove_outlier_function(df, column_name):
  q1 = df[column_name].quantile(0.25)
  q2 = df[column_name].quantile(0.50)
  q3 = df[column_name].quantile(0.75)
  interquartile_range = q3-q1
  low = q2-1.7*interquartile_range
  high = q2+1.7*interquartile_range
  df = df.loc[(df[column_name] > low) & (df[column_name] < high)]
  return df

df = remove_outlier_function(df, 'order_amount');
print(df.order_amount.describe())
```

```
count    4785.000000
mean      287.390178
std       136.232218
min        90.000000
25%       161.000000
50%       276.000000
```

```
75%       368.000000
max       665.000000
Name: order_amount, dtype: float64
```

Compare Two approach output.

1. In the first approach we can see that std is lower compare to the old std value, which shows that the given median and mean value is more accurate for both new datasets.
2. In the second approach we truncated data that are far from the median value and then find the median of that truncated data which gives 130 std value, which is very low compare to the old value which shows more accuracy in the new AOV.

```
median_value = df['order_amount'].median();
print(median_value)
```

```
276.0
```

✓ 0s completed at 20:36

Name – Nisarg Patel
Email – npatel@dal.ca
Phone – 902-880-5936

Question 2

1. 54

```
SELECT COUNT(*) AS NoOfOrders
FROM Orders
JOIN Shippers ON Shippers.ShipperID = Orders.ShipperID
WHERE Shippers.ShipperName = 'Speedy Express'
```

2. Peacock

```
SELECT MAX(total), lastname
FROM (SELECT COUNT(orderID) AS total, lastname, Orders.employeeID
      FROM Orders
      JOIN Employees ON Employees.employeeID = Orders.employeeID
      GROUP BY Employees.employeeID
      )
```

3. Boston Crab Meat

```
SELECT MAX(Total), productName
FROM (SELECT productID, SUM(Quantity) AS Total, productName
      FROM (SELECT * FROM Customers
            JOIN ( SELECT orderID,productID,Quantity,CustomerID
                  FROM Orders JOIN OrderDetails USING (orderID))
            USING (customerID)
            WHERE Country = "Germany")
      JOIN products USING (productID)
      GROUP BY productID
      )
```