Shopify Intern Challenge

January 18, 2022

0.1 Shopify 2022 Data Science Intern Challenge

0.1.1 *By Anjali Chauhan*

Please complete the following questions, and provide your thought process/work. You can attach your work in a text file, link, etc. on the application page. Please ensure answers are easily visible for reviewers!

0.1.2 Question 1:

Given some sample data, write a program to answer the following: click here to access the required data set

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.

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

(ii) What metric would you report for this dataset?

(iii) What is its value?

0.1.3 Import libraries

[2]:
```
import pandas as pd
import numpy as np
```

0.1.4 Reading the data

[3]:
```
df = pd.read_csv('2019 Winter Data Science Intern Challenge Data Set - Sheet1.
 ↪csv')
df.head()
```

[3]:
```
   order_id  shop_id  user_id  order_amount  total_items payment_method  \
0         1       53      746           224            2           cash
1         2       92      925            90            1           cash
2         3       44      861           144            1           cash
3         4       18      935           156            1    credit_card
```

```
4          5       18      883           156             1    credit_card

            created_at
0  2017-03-13 12:36:56
1  2017-03-03 17:38:52
2   2017-03-14 4:23:56
3  2017-03-26 12:43:37
4   2017-03-01 4:35:11
```

0.1.5 Exploratory Data Analysis

[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   order_id        5000 non-null   int64
 1   shop_id         5000 non-null   int64
 2   user_id         5000 non-null   int64
 3   order_amount    5000 non-null   int64
 4   total_items     5000 non-null   int64
 5   payment_method  5000 non-null   object
 6   created_at      5000 non-null   object
dtypes: int64(5), object(2)
memory usage: 273.6+ KB
```

[5]:
```
df.describe()
```

[5]:
```
          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
```

[6]:
```
# Check for null values in the dataframe
null_values = df.isnull().sum().to_frame()
null_values.columns = ['null_values']
null_values
```

[6]:
```
                null_values
order_id                  0
```

```
shop_id                0
user_id                0
order_amount           0
total_items            0
payment_method         0
created_at             0
```

[7]:
```
# AOV calculated in the question
df.order_amount.sum()/len(df.order_amount) # mean
```

[7]:
```
3145.128
```

[8]:
```
df.order_amount.median()
```

[8]:
```
284.0
```

[9]:
```
# Correct Mean
round(df.order_amount.sum()/sum(df.total_items),2) # mean
```

[9]:
```
357.92
```

[13]:
```
# Mode
df.order_amount.mode()
```

[13]:
```
0    153
dtype: int64
```

[10]:
```
# Converting id's from integer type to string
df.order_id = df.order_id.astype(str)
df.shop_id = df.shop_id.astype(str)
df.user_id = df.user_id.astype(str)
```

[12]:
```
df['item_price'] = (df.order_amount/df.total_items).astype(int)
df
```

[12]:
```
     order_id shop_id user_id  order_amount  total_items payment_method  \
0           1      53     746           224            2           cash
1           2      92     925            90            1           cash
2           3      44     861           144            1           cash
3           4      18     935           156            1    credit_card
4           5      18     883           156            1    credit_card
…         …       …       …             …            …              …
4995     4996      73     993           330            2          debit
4996     4997      48     789           234            2           cash
4997     4998      56     867           351            3           cash
4998     4999      60     825           354            2    credit_card
4999     5000      44     734           288            2          debit
```

```
               created_at  item_price
0     2017-03-13 12:36:56         112
1     2017-03-03 17:38:52          90
2      2017-03-14 4:23:56         144
3     2017-03-26 12:43:37         156
4      2017-03-01 4:35:11         156
…                    …          …
4995  2017-03-30 13:47:17         165
4996  2017-03-16 20:36:16         117
4997   2017-03-19 5:42:42         117
4998  2017-03-16 14:51:18         177
4999  2017-03-18 15:48:18         144

[5000 rows x 8 columns]
```

[19]:
```
# Average order amount for each shop (lowest 5)
df.groupby('shop_id')[['order_amount']].mean().sort_values('order_amount').
 ↪head()
```

[19]:
```
         order_amount
shop_id
92         162.857143
2          174.327273
32         189.976190
100        213.675000
53         214.117647
```

[20]:
```
# Average order amount for each shop (highest 5)
df.groupby('shop_id')[['order_amount']].mean().sort_values('order_amount').
 ↪tail()
```

[20]:
```
          order_amount
shop_id
38          390.857143
90          403.224490
50          403.545455
78        49213.043478
42       235101.490196
```

[24]:
```
# Shop 42
df[df.shop_id == '42'][['order_amount', 'total_items', 'item_price']].
 ↪value_counts()
```

[24]:
```
order_amount  total_items  item_price
704000        2000         352           17
352           1            352           15
```

```
704           2            352            13
1056          3            352             3
1408          4            352             2
1760          5            352             1
dtype: int64
```

[25]:
```
# Shop 78
df[df.shop_id == '78'][['order_amount', 'total_items', 'item_price']].
 ↪value_counts()
```

[25]:
```
order_amount  total_items  item_price
25725         1            25725          19
51450         2            25725          16
77175         3            25725           9
102900        4            25725           1
154350        6            25725           1
dtype: int64
```

0.1.6 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.

(i) How many orders were shipped by Speedy Express in total?

Answer: 54 orders were shipped by Speedy Express in total

```
SELECT o.OrderID, o.ShipperID, s.ShipperID, s.ShipperName, COUNT(*) FROM [Orders] o
LEFT JOIN [Shippers] s ON o.ShipperID = s.ShipperID
WHERE s.ShipperName = 'Speedy Express'
```

(ii) What is the last name of the employee with the most orders?

Answer: Peacock is the last name of the employee with the most orders, about 40

```
SELECT e.LastName, COUNT(DISTINCT(o.OrderID)) AS total_orders FROM  [Employees] e
LEFT JOIN [Orders] o
ON e.EmployeeID =  o.EmployeeID
GROUP BY e.LastName
ORDER BY  total_orders DESC
LIMIT 1
```

(iii) What product was ordered the most by customers in Germany?

Answer: Boston Crab Meat was the product ordered the most by customers in Germany, about 160. In this problem, product most ordered is defined as total quantity of the product ordered and not how many times it was ordered.

```
SELECT d.OrderID, p.ProductID, p.ProductName, SUM(d.Quantity) AS TotalQuantity FROM [Customers
LEFT JOIN [Orders] o
ON c.CustomerID = o.CustomerID
```

```
LEFT JOIN [OrderDetails] d
ON o.OrderID = d.OrderID
LEFT JOIN [PRODUCTS] p
ON d.ProductID = p.ProductID
WHERE c.Country = 'Germany'
GROUP BY p.ProductName
ORDER BY TotalQuantity DESC
LIMIT 1
```