

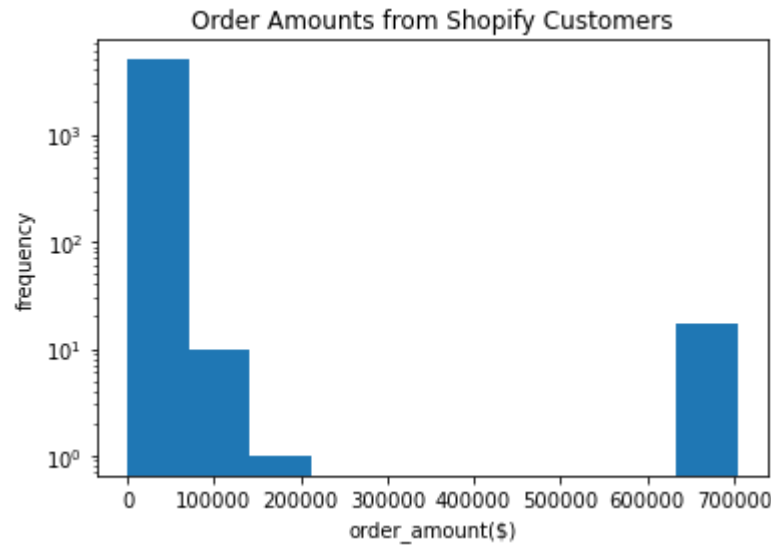
```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
In [27]: df = pd.read_csv('shopifyData.csv')
```

```
In [28]: # EDA
df.head(10)
```

	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	6	58	882	138	1	credit_card	2017-03-14 15:25:01
6	7	87	915	149	1	cash	2017-03-01 21:37:57
7	8	22	761	292	2	cash	2017-03-08 2:05:38
8	9	64	914	266	2	debit	2017-03-17 20:56:50
9	10	52	788	146	1	credit_card	2017-03-30 21:08:26

```
In [29]: # EDA
new = df['order_amount'].value_counts(ascending=True).reset_index()
# print(list(new['index'])) #list of ascending most popular purchase values
# print(df[df['order_amount'] == 704000].head())
revenue = sum(df['order_amount'])
num_orders = df.shape[0]
lst = list(new['index'])
plt.title('Order Amounts from Shopify Customers');
plt.hist(df['order_amount'], log=True);
plt.xlabel('order_amount($)');
plt.ylabel('frequency');
```



we can see how AOV is skewed based on this histogram and that's why we should consider another metric

```
In [30]: # a
# AOV - revenue / # of orders

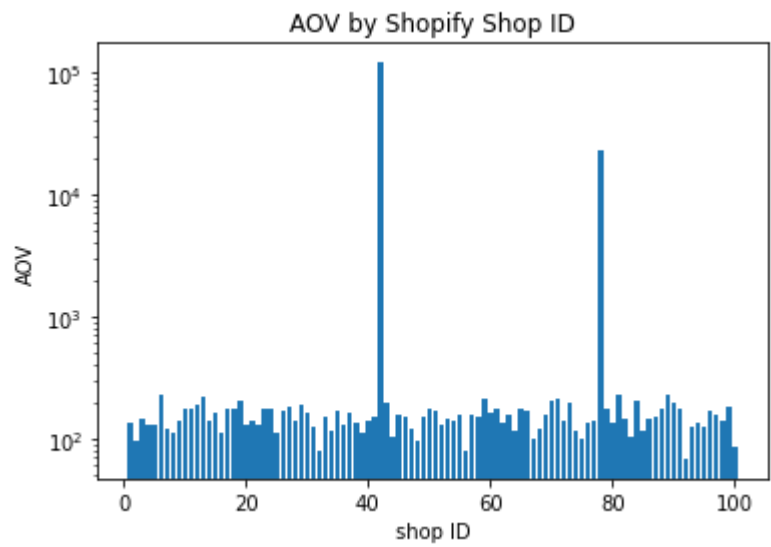
print("our AOV: "+str(revenue / num_orders))

our AOV: 3145.128
```

There's outliers in the (order_amount) field that's causing the AOV to be a huge overestimate. Specifically, the number of orders and the order amount is not the same across each of the 100 stores in our data.

```
In [31]: aovGraph = df.groupby(['shop_id'])['order_amount'].agg('sum') / len(df.groupby(['shop_id']))
newGraph = aovGraph.to_frame(name = 'AOV')
```

```
In [35]: plt.bar(newGraph.index, newGraph['AOV'], log = True);
plt.xlabel('shop ID');
plt.ylabel('AOV');
plt.title('AOV by Shopify Shop ID');
```



Looks like shops with ID 78 and 42 are outliers with the highest AOVs.

Also I think AOV isn't as usefull unless your store has a more compact price range. With the given data, we can see our order total prices are everywhere and as a result our AOV is being skewed. Instead, let's use median since it takes outliers into account

MOV (Median Order Value)

```
In [36]: # b
median = np.median(newGraph['AOV'])
```

```
In [37]: # c
print("our median: " + str(median))

our median: 148.875

the median calculated should be a better indicator of the price of one purchase
```

Question 2

a

select count(*) as count from Shippers s join Orders o on s.ShipperID = o.ShipperID where s.ShipperName = "Speedy Express";

In total, 54 Orders were shipped by Speedy Express

b

select e.LastName from Employees e where e.EmployeeID = (select o.EmployeeID from Orders o group by o.EmployeeID order by count(OrderID) desc limit 1);

The last name of the employee with the most orders is Peacock

c

select p.productName from products p where p.productID = (select productID from (select * from customers c join orders o on c.CustomerID == o.CustomerID where Country == "Germany") g join orderDetails d on d.OrderID = g.OrderID group by productID order by SUM(d.QUANTITY) desc limit 1)

Boston Crab Meat was ordered most by customers in Germany