Shopify Data Science Internship Challenge 2022

Joshua Boehm

1/17/2022

Our examination of sneaker shops hosted with Shopify begins with simply calculating the mean order amount as things stand.

```
prices <- shop$order_amount
mean(prices)
```

```
## [1] 3145.128
```

As mentioned in the original prompt, this is an oddly high average considering the context. Consequently, this leads me to think that there could be outliers in our data. This does appear to be the case:

```
# Sort order amounts descending
head <- prices[order(-prices)]

# Examining the 30 highest unique order amounts
unique(head)[1:30]
```

```
##  [1] 704000 154350 102900  77175  51450  25725   1760   1408   1086   1064
## [11]   1056    980    965    960    948    935    920    890    885    880
## [21]    865    845    830    816    815    810    805    804    800    790
```

Quickly looking at the first thirty unique order amounts, most on the lower end are around $1,000. After this, however, order numbers increase suddenly and sharply, going as high as $50,000, $100,000, and even $700,000. When we have extreme outliers in this case, it's a sign that we should be considering a different statistic.

Let's see what's happening with these $700,000 orders.

```
outliers <- subset(shop, shop$order_amount > 700000)
kable(outliers)
```

order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
16	42	607	704000	2000	credit_card	2017-03-07 4:00:00
61	42	607	704000	2000	credit_card	2017-03-04 4:00:00

order_id	shop_id	user_id	order_amount	total_items	payment_method	created_at
521	42	607	704000	2000	credit_card	2017-03-02 4:00:00
1105	42	607	704000	2000	credit_card	2017-03-24 4:00:00
1363	42	607	704000	2000	credit_card	2017-03-15 4:00:00
1437	42	607	704000	2000	credit_card	2017-03-11 4:00:00
1563	42	607	704000	2000	credit_card	2017-03-19 4:00:00
1603	42	607	704000	2000	credit_card	2017-03-17 4:00:00
2154	42	607	704000	2000	credit_card	2017-03-12 4:00:00
2298	42	607	704000	2000	credit_card	2017-03-07 4:00:00
2836	42	607	704000	2000	credit_card	2017-03-28 4:00:00
2970	42	607	704000	2000	credit_card	2017-03-28 4:00:00
3333	42	607	704000	2000	credit_card	2017-03-24 4:00:00
4057	42	607	704000	2000	credit_card	2017-03-28 4:00:00
4647	42	607	704000	2000	credit_card	2017-03-02 4:00:00
4869	42	607	704000	2000	credit_card	2017-03-22 4:00:00
4883	42	607	704000	2000	credit_card	2017-03-25 4:00:00

These outlandishly-expensive orders have multiple things in common: all were placed at shop number 42, ordered by user_id 607, consist of 2000 items, and placed at exactly 4:00:00. It should be undoubtedly irrational for one person to make orders like this over multiple days at 4AM. With all of this considered, we may have enough evidence to drop these data points *if needed*. Perhaps these are bot orders, or possibly maintenance/testing being done–in other words, not normal consumer behavior.

The next unique order numbers still seem to be a little high based on orders from shoe shops, so we should look into these as well.

```
outliers <- subset(shop, shop$order_amount > 20000 & shop$order_amount < 700000)
```

Some facets of this subset differ from the previous subset. For example, there's nothing too suspicious about the order dates and user IDs associated with orders. One thing that stands out is that all of these expensive orders were all placed at shop number 78. We know that each shop only sells one type of shoe, and the order amounts are all multiples of $25,725. It's possible that this one shop just happens to be selling exorbitantly expensive shoes (resale prices on some shoes can get pretty crazy). In comparison to the other shops/order amounts, this shop is an outlier, but we can't necessarily consider it to be irrelevant like we might have done with the $700,000 orders.

The average after removing orders over $700,000 is significantly lower than before (~$754), but still somewhat high.

```
prices_adj <- subset(shop, shop$order_amount < 700000)

mean(prices_adj$order_amount)
```

```
## [1] 754.0919
```

Let's switch gears and think about any new metrics that can be used to get more insight into what's being purchased. The simplest possible alternative would be to calculate the median, which would give the most typical total order amount.

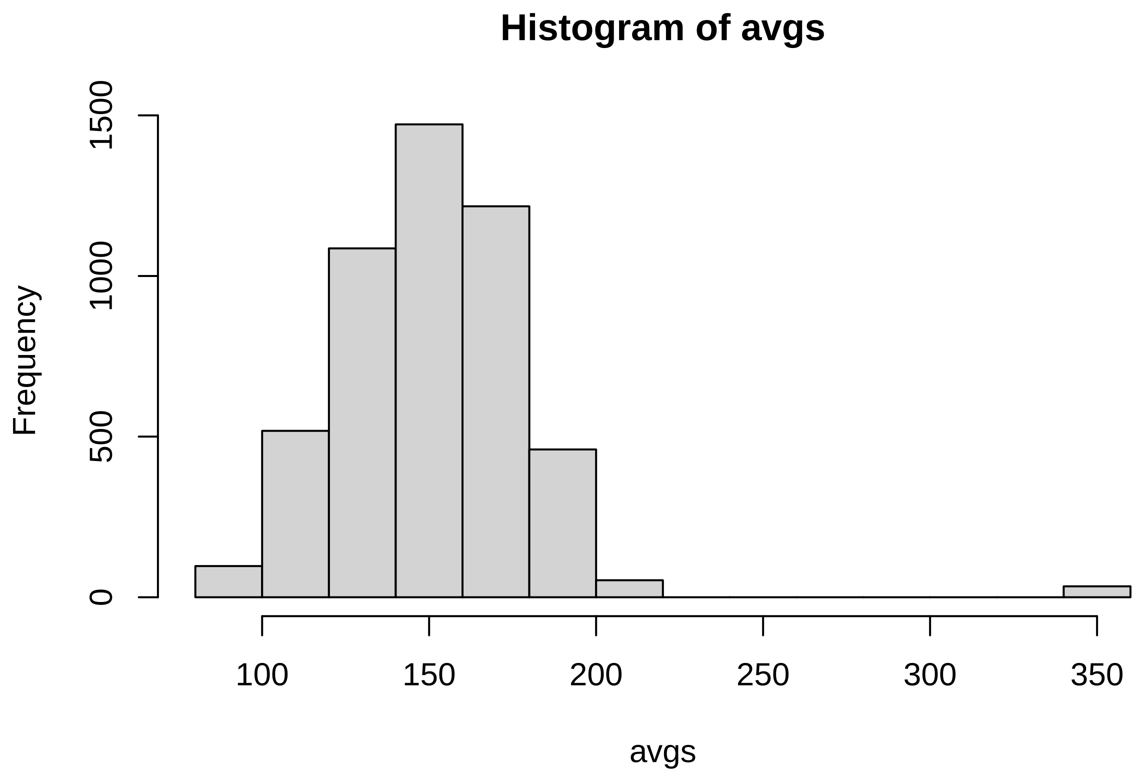
```
median(prices_adj$order_amount)
```

```
## [1] 284
```

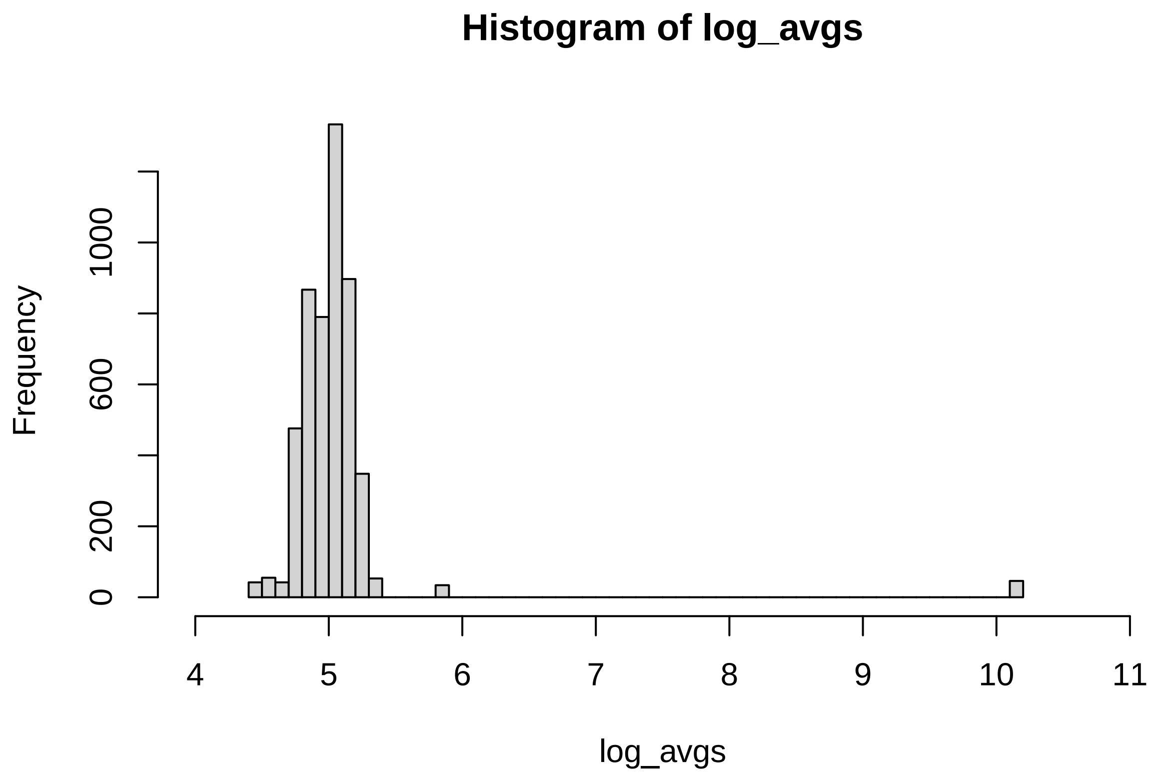
While looking at the median is more realistic than the mean, since it's not affected as heavily by outliers, it's still not as insightful as we might want to be. We have access to the number of total items in an order, so it would make sense to consider the average price per item in an order. For instance, an order of three shoes for a total of $900 means that the average price per item is $300.

For visualization purposes only, I've omitted all orders with amounts greater than $2000 from the general histogram. To get a sense of the overall shape, a histogram of the log of average price per item is provided as well.

```
prices_adj <- subset(shop, shop$order_amount < 2000)

n <- length(prices_adj$order_id)

avgs <- numeric(n)

for (i in 1:n) {

    avgs[i] <- prices_adj$order_amount[i] / prices_adj$total_items[i]
}

hist(avgs)
```




```
# ----

prices_adj <- subset(shop, shop$order_amount < 700000)

n <- length(prices_adj$order_id)

log_avgs <- numeric(n)

for (i in 1:n) {

    log_avgs[i] <- log(prices_adj$order_amount[i] / prices_adj$total_items[i])
}

hist(log_avgs, breaks=50, xlim=c(4,11))
```



One final metric that we might want to consider is the average total order price per shop. This is in the same vein as the original proposal of average order value, but provides a bit more specific insight when filtering by individual shops. This could also account for overall outliers–for example, we saw previously that shop 78 has items worth $25,000.

```
avgs <- numeric(100)

for (i in 1:100) {

  # Subsets into the current shop in the loop
  curr_shop <- subset(shop, shop$shop_id == i)

  # From the current shop, sum all order amounts and divide by the number of orders (length)
  avgs[i] <- sum(curr_shop$order_amount) / nrow(curr_shop)

}

# Bar plot of untransformed values; ylim=(0,500)
barplot(avgs, names.arg = c(1:length(avgs)), ylim=c(0,5000), xlab="Shop ID",
        ylab="Average order amount ($)", main="Avg. order amt. by shop")
```


```
# Log transformed bar plot
barplot(log(avgs), names.arg = c(1:length(avgs)), xlab="Shop ID",
        ylab="log(Average order amount)", main="Avg. log order amt. by shop")
```


```
df <- data.frame("shop_id" = c(1:100), "avg_order_amt" = avgs)

kable(df)
```

shop_id	avg_order_amt
1	308.8182
2	174.3273
3	305.2500
4	258.5098
5	290.3111
6	383.5085
7	218.0000
8	241.0435
9	234.0000
10	332.3019
11	356.7347
12	352.6981
13	345.3968
14	242.0000
15	308.9423
16	270.1463
17	332.0755
18	342.5882
19	320.9062
20	251.5577

shop_id	avg_order_amt
21	308.6957
22	273.7500
23	317.6727
24	320.7273
25	232.9167
26	341.2245
27	334.8704
28	320.3721
29	331.6207
30	295.0714
31	268.9787
32	189.9762
33	376.2750
34	234.2400
35	328.0000
36	254.8000
37	340.2083
38	390.8571
39	268.0000
40	295.1667
41	254.0000
42	235101.4902
43	333.9138
44	262.1538
45	269.3103
46	347.4419
47	259.1489
48	242.7750
49	279.9057
50	403.5455
51	361.8043
52	316.9268
53	214.1176
54	276.6400
55	327.7500
56	218.1892
57	296.7736
58	254.9492
59	358.9667
60	350.2340
61	344.4400
62	308.8372
63	264.9655
64	272.1860
65	330.8148
66	312.8868
67	272.6216
68	254.6383
69	264.1833
70	343.0678
71	323.0303
72	309.5652

shop_id	avg_order_amt
73	335.6897
74	306.0000
75	240.7619
76	321.0714
77	280.8000
78	49213.0435
79	328.4815
80	299.6667
81	384.0000
82	349.7857
83	248.7857
84	342.3051
85	329.2571
86	277.5000
87	292.2692
88	355.5200
89	379.1475
90	403.2245
91	325.9259
92	162.8571
93	214.4746
94	297.7778
95	318.7692
96	330.0000
97	324.0000
98	245.3621
99	339.4444
100	213.6750