Case Study Report

Analysis of E-commerce Data for Target Corporation

Submitted by: Aruna Panwar

Date of Submission: 01-10-2023

1. Import the dataset and do the usual exploratory analysis steps like checking the structure & and characteristics of the dataset:

QUERY(1a) DATA TYPE OF ALL COLUMN?

```
SELECT
column_name,
data_type
from `my-project-16925-396802.target_market`.INFORMATION_SCHEMA.COLUMNS
WHERE    table_name = 'customer';
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	column_name	data_type
1	customer_id	STRING
2	customer_unique_id	STRING
3	customer_zip_code_prefix	INT64
4	customer_city	STRING
5	customer_state	STRING

QUERY (1b) Get the time range between which the order were placed.

```
SELECT
MIN(order_purchase_timestamp) AS start_time, MAX(order_purchase_timestamp) AS end_time
FROM `target_market.orders`
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	start_time	end_time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

QUERY(1c) Count the Cities and states of customers who ordered during the given period.

```
SELECT
COUNT(*) AS count,c.customer_city, c.customer_state
from
(
SELECT DISTINCT c.customer_city, c.customer_state
FROM `target_market.customer` as c
JOIN `target_market.orders`  as o
ON c.customer_id = o.customer_id
WHERE order_purchase_timestamp BETWEEN '2016-09-04' AND '2018-10-17'
) as c
group by c.customer_city, c.customer_state
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART

Row	customer_city	customer_state	count
1	acu	RN	1
2	ico	CE	1
3	ipe	RS	1
4	ipu	CE	1
5	ita	SC	1
6	itu	SP	1
7	jau	SP	1
8	luz	MG	1
9	poa	SP	1
10	uba	MG	1

2. In-depth Exploration:

QUERY (2a) Is there a growing trend in the number of orders placed over the past years?

```
SELECT
EXTRACT(YEAR FROM order_purchase_timestamp) AS order_year,
    COUNT(*) AS num_orders
FROM`target_market.orders`
WHERE  order_purchase_timestamp IS NOT NULL
GROUP BY order_year
ORDER BY order_year;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	order_year	num_orders
1	2016	329
2	2017	45101
3	2018	54011

Query (2b) Can we see some kind of monthly seasonality in terms of the no. of orders being placed?

```
SELECT
    EXTRACT(MONTH FROM order_purchase_timestamp) AS order_month,
    COUNT(*) AS num_orders
FROM
    `target_market.orders`
WHERE
    order_purchase_timestamp IS NOT NULL
GROUP BY   order_month
ORDER BY   order_month;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	order_year	num_orders
1	2016	329
2	2017	45101
3	2018	54011

QUERY (2c) During what time of the day, do the Brazilian customers mostly place their orders? (Dawn, Morning, Afternoon or Night)

- **0-6 hrs: Dawn**
- **7-12 hrs: Mornings**
- **13-18 hrs: Afternoon**
- **19-23 hrs: Night**

```
SELECT
CASE
when extract(hour FROM order_purchase_timestamp) between 0 AND 6 then 'Dawn'
when extract(hour FROM order_purchase_timestamp) between 7 AND 12 then 'Morning'
when extract(hour FROM order_purchase_timestamp) between 13 AND 18 then 'Afternoon'
when extract(hour FROM order_purchase_timestamp) between 19 AND 23 then 'Night'
end AS time_of_day,
COUNT(*) AS order_count
FROM `target_market.orders`
GROUP BY time_of_day;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	time_of_day	order_count
1	Morning	27733
2	Dawn	5242
3	Afternoon	38135
4	Night	28331

3.

3.Evolution of E-commerce orders in the Brazil region:

QUERY(3a) Get the month-on-month number of orders placed in each state.

```
select
extract(YEAR FROM o.order_purchase_timestamp) AS order_year,
extract(MONTH FROM o.order_purchase_timestamp) AS order_month,
c.customer_state,
COUNT(*) AS order_count
FROM `target_market.orders` as o
JOIN `target_market.customer` as c
ON o.customer_id = c.customer_id
GROUP BY order_year, order_month,c.customer_state
ORDER By order_year,order_month,c.customer_state;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW

Row	order_year	order_month	customer_state	order_count
1	2016	9	RR	1
2	2016	9	RS	1
3	2016	9	SP	2
4	2016	10	AL	2
5	2016	10	BA	4
6	2016	10	CE	8
7	2016	10	DF	6
8	2016	10	ES	4
9	2016	10	GO	9
10	2016	10	MA	4

QUERY(3b) How are the customers distributed across all the states?

```
SELECT
    customer_state,
    COUNT(*) AS customer_count
FROM
    `target_market.customer`
GROUP BY
    customer_state
ORDER BY
    customer_count DESC;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION D

Row	customer_state	customer_count
1	SP	41746
2	RJ	12852
3	MG	11635
4	RS	5466
5	PR	5045
6	SC	3637
7	BA	3380
8	DF	2140
9	ES	2033
10	GO	2020

4. Impact on the Economy: Analyse the money movement by e-commerce by looking at order prices, freight, and others.

QUERY(4a) Get the % increase in the cost of orders from year 2017 to 2018 (include months between Jan to Aug only).

```
select
(sum(case
when
extract(year  from order_purchase_timestamp) = 2018 AND
extract(month from order_purchase_timestamp) between 1 AND 8 then payment_value end) - sum(case
when
extract(year  from order_purchase_timestamp) = 2017 AND
extract(month from order_purchase_timestamp) between 1 AND 8 then payment_value end)) /
sum(case
when
extract(year from order_purchase_timestamp) = 2017 AND
extract(month from order_purchase_timestamp) between 1 AND 8 then payment_value end) * 100 AS cost_increase_percentage
FROM `target_market.payments` as p
JOIN `target_market.orders` as o
ON p.order_id = o.order_id;
'
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	cost_increase_perce
1	136.9768716466...

QUERY (4b) Calculate the Total and average value of the order price for each state

```
SELECT customer_state, sum(payment_value) as total_order_price,
avg(payment_value) AS avg_order_price
FROM `target_market.payments` as p
JOIN `target_market.orders`  as o
ON p.order_id = o.order_id
JOIN `target_market.customer` as c
on o.customer_id = c.customer_id
GROUP BY customer_state;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	total_order_price	avg_order_price
1	RJ	2144379.689999...	158.5258882235...
2	RS	890898.5399999...	157.1804057868...
3	SP	5998226.959999...	137.5046297739...
4	DF	355141.0800000...	161.1347912885...
5	PR	811156.3799999...	154.1536259977...
6	MT	187029.2900000...	195.2289039665...
7	MA	152523.0200000...	198.8566101694...
8	AL	96962.05999999...	227.0774238875...
9	MG	1872257.260000...	154.7064336473...
10	PE	324850.4400000...	187.9921527777...

QUERY(4c) Calculate the Total and average value of order freight for each state.

```
select
customer_state,
sum(freight_value) as total_freight,
avg(freight_value) as avg_freight
FROM `target_market.order_item` as ot
JOIN `target_market.orders` as o
on ot.order_id = o.order_id
JOIN `target_market.customer` as c
on o.customer_id = c.customer_id
GROUP BY customer_state;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	total_freight	avg_freight
1	SP	718723.0699999...	15.14727539041...
2	RJ	305589.3100000...	20.96092393168...
3	PR	117851.6800000...	20.53165156794...
4	SC	89660.26000000...	21.47036877394...
5	DF	50625.49999999...	21.04135494596...
6	MG	270853.4600000...	20.63016680630...
7	PA	38699.30000000...	35.83268518518...
8	BA	100156.6799999...	26.36395893656...
9	GO	53114.97999999...	22.76681525932...
10	RS	135522.7400000...	21.73580433039...

5. Analysis based on sales, freight, and delivery time

QUERY (5a) Find the number of days taken to deliver each order from the order's purchase date as delivery time.

```sql
SELECT
order_id,
order_purchase_timestamp,
order_delivered_customer_date,
(order_delivered_customer_date - order_purchase_timestamp) AS delivery_time,
(order_estimated_delivery_date - order_delivered_customer_date) AS diff_estimated_delivery
FROM `target_market.orders`
where order_delivered_customer_date is not null
```

Query results

SAVE RESULTS | EXPLORE DAT

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS | CHART PREVIEW | EXECUTION GRAPH

Row	order_id	order_purchase_timestamp	order_delivered_customer_date	delivery_time	diff_estimated_delivery
1	770d331c84e5b214bd9dc70a...	2016-10-07 14:52:30 UTC	2016-10-14 15:07:11 UTC	0-0 0 168:14:41	0-0 0 1088:52:49
2	1950d777989f6a877539f5379...	2018-02-19 19:48:52 UTC	2018-03-21 22:03:51 UTC	0-0 0 722:14:59	0-0 0 -310:3:51
3	2c45c33d2f9cb8ff8b1c86cc28...	2016-10-09 15:39:56 UTC	2016-11-09 14:53:50 UTC	0-0 0 743:13:54	0-0 0 681:6:10
4	dabf2b0e35b423f94618bf965f...	2016-10-09 00:56:52 UTC	2016-10-16 14:36:59 UTC	0-0 0 181:40:7	0-0 0 1065:23:1
5	8beb59392e21af5eb9547ae1a...	2016-10-08 20:17:50 UTC	2016-10-19 18:47:43 UTC	0-0 0 262:29:53	0-0 0 989:12:17
6	65d1e226dfaeb8cdc42f66542...	2016-10-03 21:01:41 UTC	2016-11-08 10:58:34 UTC	0-0 0 853:56:53	0-0 0 397:1:26
7	c158e9806f85a33877bdfd4f60...	2017-04-14 22:06:32 UTC	2017-05-08 11:10:26 UTC	0-0 0 565:3:54	0-0 0 228:49:34
8	b60b53ad0bb7dacacf2989fe2...	2017-05-10 14:03:27 UTC	2017-05-23 13:12:27 UTC	0-0 0 311:9:0	0-0 0 -133:12:27
9	c830f223aae08493ebecb52f2...	2017-04-22 15:50:30 UTC	2017-05-05 13:27:50 UTC	0-0 0 309:37:20	0-0 0 298:32:10
10	a8aa2cd070eeac7e4368cae3d...	2017-05-09 17:42:45 UTC	2017-05-16 23:22:20 UTC	0-0 0 173:39:35	0-0 0 24:37:40

QUERY(5b) Find out the top 5 states with the highest & lowest average freight value.

```sql
SELECT customer_state, AVG(freight_value) AS avg_freight
FROM `target_market.order_item` as ot
JOIN `target_market.orders`  as o
ON ot.order_id = o.order_id
JOIN `target_market.customer` as c
 ON o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY avg_freight DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	avg_freight
1	RR	42.98442307692...
2	PB	42.72380398671...
3	RO	41.06971223021...
4	AC	40.07336956521...
5	PI	39.14797047970...

QUERY (5c) Find out the top 5 states with the highest & and lowest average delivery time.

```
SELECT customer_state, AVG(order_delivered_customer_date - order_purchase_timestamp) AS avg_delivery_time
FROM `target_market.orders` as o
JOIN `target_market.customer` as c
ON o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY avg_delivery_time DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	avg_delivery_time
1	RR	0-0 0 705:18:3.975609756
2	AP	0-0 0 652:26:29.850746268
3	AM	0-0 0 634:13:25.613793103
4	AL	0-0 0 589:3:9.103274559
5	PA	0-0 0 570:33:0.021141649

QUERY(5d) Find out the top 5 states where the order delivery is really fast as compared to the estimated date of delivery.

```sql
SELECT customer_state, AVG(order_delivered_customer_date - order_purchase_timestamp) - AVG(order_estimated_delivery_date -
order_delivered_customer_date) AS delivery_speed
FROM `target_market.orders` as o
JOIN `target_market.customer` as c on o.customer_id = c.customer_id
GROUP BY customer_state
ORDER BY delivery_speed DESC
LIMIT 5;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	customer_state	delivery_speed
1	AL	0-0 0 396:16:59.216624685
2	RR	0-0 0 307:1:50.731707317
3	MA	0-0 0 304:26:35.404463040
4	SE	0-0 0 292:35:15.083582089
5	CE	0-0 0 267:45:27.810789679

6. Analysis based on the payments

QUERY(6a) Find the month-on-month number of orders placed using different payment types.

```sql
SELECT

    EXTRACT(month FROM order_purchase_timestamp) AS year_month,
    payment_type,
    COUNT(*) AS order_count
FROM `target_market.payments` as p
JOIN `target_market.orders` as o
ON p.order_id = o.order_id
GROUP BY year_month, payment_type
ORDER BY year_month, payment_type;
```

Query results

JOB INFORMATION | RESULTS | JSON | EXECUTION DETAILS

Row	year_month	payment_type	order_count
1	1	UPI	1715
2	1	credit_card	6103
3	1	debit_card	118
4	1	voucher	477
5	2	UPI	1723
6	2	credit_card	6609
7	2	debit_card	82
8	2	voucher	424
9	3	UPI	1942
10	3	credit_card	7707

QUERY(6b) Find the no. of orders placed on the basis of the payment instalments that have been paid.

```
SELECT payment_installments,
count(*) AS order_count
FROM `target_market.payments`
GROUP BY payment_installments;
```

Query results

JOB INFORMATION | RESULTS | JSON

Row	payment_installment	order_count
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626

Actionable Insights & Recommendations:

1. Monitoring and Adapting to Trends:

- **Action Item**: Continuously monitor the trend of growing orders over the years, and adapt marketing and inventory strategies accordingly.
- **Recommendation:** Use historical data to forecast demand for different seasons and plan promotions and stock accordingly.

2. Peak Shopping Time Optimization:

- **Action Item:** Schedule promotions, advertising, and customer engagement efforts during the afternoon and morning hours when Brazilian customers are most active.
- **Recommendation**: Run targeted marketing campaigns during these peak times to maximize reach and impact.

3. Regional Expansion and Focus:

- **Action Item:** Consider expanding services and marketing efforts in regions with high customer concentration, such as São Paulo and Rio de Janeiro.
- **Recommendation:** Tailor product offerings and promotions to suit the preferences and needs of customers in these key regions.

4. Cost Management:

- **Action Item:** Continuously monitor the cost of orders and identify factors contributing to cost increases.
- **Recommendation**: Conduct cost analysis to understand the drivers of cost changes and implement cost-saving measures without compromising service quality.

5. Logistics and Delivery Efficiency:

- **Action Item**: Investigate regions with higher average freight costs and explore more cost-effective shipping solutions.
- **Recommendation**: Optimize logistics routes and partner with reliable local carriers to reduce freight costs and improve delivery efficiency.

6.Payment Option Optimization:

- **Action Item**: Based on payment type popularity and installment preferences, optimize the payment options offered to customers.
- **Recommendation:** Implement a variety of payment methods and consider flexible installment plans to cater to diverse customer preferences.

7.Delivery Speed Showcase:

- **Action Item**: Highlight states where orders are consistently delivered faster than estimated.
- **Recommendation**: Promote the company's efficiency and reliability in these states to attract more customers and build trust.

Thank you