Project Overview

Name of the project	Ecommerce store – analysis
Author	Piotr Milner piotr.milner@outlook.com
Project description	The project concerns the analysis of the customer behaviour data, with over 66M records, from a large multi-category online store. The goal was to obtain results showing the performance indicators allowing to take appropriate steps to increase the efficiency of purchases made by customers and thus increase revenue.
Dataset source	Kaggle.com - Contains a customer behaviour data from a large multi-category online store. Data collected by Open CDP project.
GitHub	https://github.com/piotr-milner/Data-Projects/tree/main/eCommerce%20Project

Roadmap

Table of Contents

1. Defining Questions

First step in our analysis is to define a goal including answering to the following questions:
1. Show the breakdown of the revenue generated by the largest brands.
2. Top 10 products generating the most revenue along with the number of units sold.
3. What hours do most customers make purchases + comparison to the target?
4. What days do most customers make purchases + comparison to the target?
5. Show the division of events in the store.

2. Collecting data

Dataset: 2020-Apr.csv (9,27 GB)
Source on the first page.

3. Exploratory Data Analysis (EDA)

Dataset contains a customer behaviour data from a large multi-category online store.
Each row in the file represents an event. All events are related to products and users. Each event is like many-to-many relation between products and users.

1. View data in a CSV file
2. Import file to PostgreSQL

Property	Description
event_time	Time when event happened at.
event_type	Kind of event: purchase (user purchased a product) / view (user viewed a product) / cart (user added a product to shopping cart)
product_id	ID of a product
category_id	Product's category ID
category_code	Product's category taxonomy (code name) if it was possible to make it. Usually present for meaningful categories and skipped for different kinds of accessories.
brand	Downcased string of brand name.

Property	Description
price	Float price of a product. Present.
user_id	Permanent user ID.
user_session	Temporary user's session ID. Same for each user's session. Is changed every time user come back to online store from a long pause.

```
-- Creating a table for dataset

CREATE TABLE eCommerce(
    event_time TIMESTAMP,
    event_type VARCHAR(10),
    product_id INT,
    category_id BIGINT,
    category_code VARCHAR(50),
    brand VARCHAR(50),
    price FLOAT,
    user_id INT,
    user_session VARCHAR(50)
)


-- Copying dataset from file

COPY eCommerce FROM '/Users/piotrmilner/Desktop/Project B/2020-Apr.csv' DELIMITER ',' CSV HEADER;
```

Checking the parameters of each column, taking into account the type of data contained in them below:

```
-- EDA

SELECT
  'category_id' as "Column",
  count(category_id) as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL as "75th percentile",
```

```
  (MIN(category_id)::VARchar(20)) as "Min",
  (MAX(category_id)::VARchar(20)) as "Max",
  NULL AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(category_id) as "Null" from ecommerce where category_id is nULL) as
null_searching, (select distinct category_id as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

Union

SELECT
  'price' as "Column",
  count(price) as "Not null",
  null_searching."Null",
  PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY price) as "25th percentile",
  PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY price) as "Median",
  PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY price) as "75th percentile",
  (MIN(price)::VARchar(20)) as "Min",
  (MAX(price)::VARchar(20)) as "Max",
  AVG(price) AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(price) as "Null" from ecommerce where price is nULL) as null_searching,
(select distinct price as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

Union

SELECT
  'user_id' as "Column",
  count(user_id) as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL as "75th percentile",
  (MIN(user_id)::VARchar(20)) as "Min",
  (MAX(user_id)::VARchar(20)) as "Max",
  NULL AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(user_id) as "Null" from ecommerce where user_id is nULL) as null_searching,
(select distinct  user_id as dist from ecommerce) as search_distinct
```

```
GROUP BY "Column", null_searching."Null"

UNION

SELECT
  'event_time' as "Column",
  count(event_time) as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL AS "75th percentile",
  (MIN(event_time)::VARchar(20)) as "Min",
  (MAX(event_time)::VARchar(20)) as "Max",
  NULL AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(*) as "Null" from ecommerce where event_time is nULL) as null_searching,
(select distinct event_time as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

Union

SELECT
  'product_id' as "Column",
  count(user_id) as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL as "75th percentile",
  (MIN(product_id)::VARchar(20)) as "Min",
  (MAX(product_id)::VARchar(20)) as "Max",
  NULL AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(user_id) as "Null" from ecommerce where product_id is nULL) as
null_searching, (select distinct product_id as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

union

SELECT
  'user_sessison' as "Column",
```

```
  count('user_sessison') as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL as "75th percentile",
  (MIN(length('user_sessison'))::VARchar(20)) as "Min",
  (MAX(length('user_sessison'))::VARchar(20)) as "Max",
  NULL AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count('user_sessison') as "Null" from ecommerce where 'user_sessison' is nULL) as
null_searching, (select distinct user_session as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

union

SELECT
  'event_type' as "Column",
  count(event_type) as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL as "75th percentile",
  (MIN(length(event_type))::VARchar(20)) as "Min",
  (MAX(length(event_type))::VARchar(20)) as "Max",
  NULL AS "Avg",
  COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(event_type) as "Null" from ecommerce where event_type is nULL) as
null_searching, (select distinct event_type as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

UNION

SELECT
  'category_code' as "Column",
  count(category_code) as "Not null",
  null_searching."Null",
  NULL as "25th percentile",
  NULL as "Median",
  NULL AS "75th percentile",
  (MIN(length(category_code))::VARchar(20)) as "Min",
```

```
 (MAX(length(category_code))::VARchar(20)) as "Max",
 AVG(length(category_code)) AS "Avg",
 COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(*) as "Null" from ecommerce where category_code is nULL) as null_searching ,
(select distinct 'category_code' as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"

union

SELECT
 'brand' as "Column",
 count(brand) as "Not null",
 null_searching."Null",
 NULL as "25th percentile",
 NULL as "Median",
 NULL AS "75th percentile",
 (MIN(length(brand))::VARchar(20)) as "Min",
 (MAX(length(brand))::VARchar(20)) as "Max",
 AVG(length(brand)) AS "Avg",
 COUNT(dist) as "Unique Values"
FROM ecommerce, (select count(*) as "Null" from ecommerce where brand is nULL) as null_searching, (select
distinct brand as dist from ecommerce) as search_distinct
GROUP BY "Column", null_searching."Null"
```

OUTPUT:

Column	Not null	Null	25th percentile	Median	75th percentile	Min	Max	Avg	Unique Values
user_sessison	66589268	0				13	13		11652261
user_id	66589268	0				27396220	649775983		4509623
event_type	66589268	0				4	8		3
category_id	66589268	0				2053013551857008829	2298840883472106102		1201
brand	57604211	8985057				2	25	5,896190992	4886
category_code	59833395	6755873				9	38	22,90316346	4886
event_time	66589268	0				2020-04-01 00:00:00	2020-04-30 23:59:59		2564826
product_id	66589268	0				1000365	100234250		263503
price	66589268	0	51,46	148,91	342,35	0	2574.07	273,3337723	54692

Results

The 'brand' and 'category_code' columns contain NULLs, but this is irrelevant for the current analysis. Other than that, the data looks clean and contains no outliners.
No data cleaning needed.

4. Data Analysis

```
-- Shows traffic by day of week + target

WITH t1 AS (SELECT EXTRACT (dow FROM event_time) AS h, count(user_id) AS count_cart
FROM ecommerce
WHERE event_type = 'cart'
GROUP BY h),
t2 AS (SELECT EXTRACT (dow FROM event_time) AS h, count(user_id) AS count_purchase
FROM ecommerce
WHERE event_type = 'purchase'
GROUP BY h)

SELECT t2.h, t2.count_purchase, 0.3*t1.count_cart AS target
FROM t1
JOIN t2 USING(h)
ORDER BY t2.h
```

h	count_purchase	target
0	127556	122649.0
1	100526	128779.2
2	98848	127458.9
3	175656	165040.2
4	182844	168740.7
5	143956	137033.1
6	137373	130878.9

```
-- Shows traffic by hour + target

WITH t1 AS (SELECT EXTRACT (HOUR FROM event_time) AS h, count(user_id) AS count_cart
FROM ecommerce
WHERE event_type = 'cart'
GROUP BY h),
t2 AS (SELECT EXTRACT (HOUR FROM event_time) AS h, count(user_id) AS count_purchase
```

```
FROM ecommerce
WHERE event_type = 'purchase'
GROUP BY h)

SELECT t2.h, t2.count_purchase, 0.3*t1.count_cart AS target
FROM t1
JOIN t2 USING(h)
ORDER BY t2.h
```

h	count_purchase	target
0	4404	4961.7
1	7140	7987.2
2	14078	15243.9
3	28820	29028.9
4	43950	43438.2
5	56285	55395.6
6	64296	64160.1
7	70190	69227.1
8	72777	71660.7
9	71866	71002.8
10	69294	68301.0
11	64788	64044.0
12	59876	60092.1
13	56449	56285.4
14	52517	54209.1
15	51827	55473.0
16	48015	52313.1
17	42124	45783.6
18	32499	34384.5
19	21613	22287.0
20	13625	14398.8
21	8919	8902.8
22	6634	6801.0
23	4773	5198.4

```
-- Shows the breakdown of event types

SELECT ecommerce.event_type, (count(event_type)/t1.events_count_total::FLOAT)*100 AS
percentage_of_events_count
FROM ecommerce, (SELECT count(event_type) AS events_count_total FROM ecommerce) AS t1
GROUP BY ecommerce.event_type, t1.events_count_total
```

event_type	percentage_of_events_count

view	93,63957718
purchase	1,451824039
cart	4,908598785

```sql
-- Shows the breakdown of brands and revenue from their products

WITH
t1 AS (
    SELECT sum(price) AS total_income
    FROM ecommerce
    WHERE event_type = 'purchase'
    ),
t2 AS
    (
    SELECT initcap(ecommerce.brand) AS brand, (sum(ecommerce.price)/t1.total_income)*100 AS
percentage_of_total_income,
    sum(ecommerce.price) AS income_per_brand
    FROM ecommerce, t1
    WHERE brand IS NOT NULL AND event_type = 'purchase'
    GROUP BY ecommerce.brand, t1.total_income
    ORDER BY percentage_of_total_income DESC
    LIMIT 10
    )

SELECT *
FROM t2

UNION

SELECT 'Others', 100-sum(t2.percentage_of_total_revenue)AS percentage_of_total_revenue,
t1.total_revenue - sum(t2.revenue_per_brand) AS revenue_per_brand
FROM t1, t2
GROUP BY t1.total_revenue
ORDER BY percentage_of_total_revenue DESC
```

brand	percentage_of_total_revenue	revenue_per_brand
Apple	30,38	74288067,38
Sum Of Others	25,448	62226710,64
Samsung	23,225	56791598,49

Xiaomi	4,9503	12104816,44
Lg	3,0833	7539392,84
Acer	2,8439	6954180,02
Huawei	2,4493	5989082,48
Lenovo	2,087	5103321,68
Oppo	2,0807	5087836,84
Sony	1,7301	4230660,64
Asus	1,7218	4210124,66

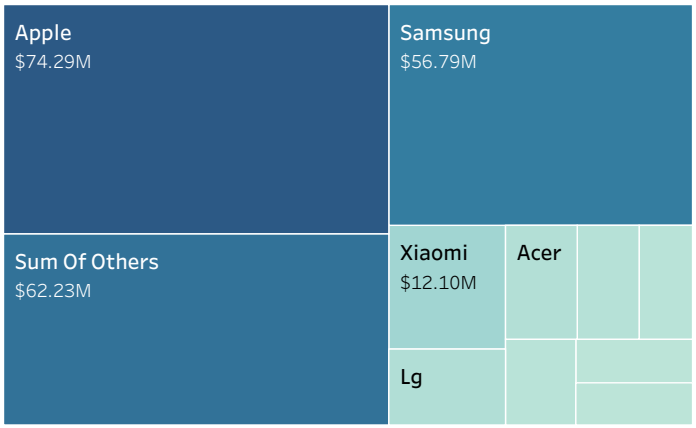
```
-- Loking for top 10 products with the highest revenue and pcs sold

SELECT product_id, sum(price) AS revenue, count(product_id) AS pcs_sold
FROM ecommerce
WHERE event_type = 'purchase'
GROUP BY product_id
ORDER BY sum(price) DESC
LIMIT 10
```

product_id	revenue	pcs_sold
1005115	8294713,67	8873
1005105	4819198,2	3608
1002544	4488925,27	10849
100068493	4319998,93	13932
100068488	4210075,31	14915
1005116	3927949,24	3912
1005100	3849294,64	26230
1005212	3804571,93	22227
100154083	3688287,7	2124
1004249	3456536,47	4523

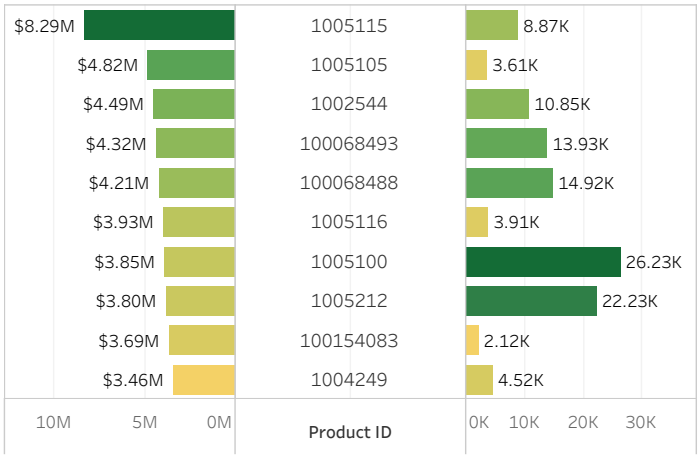
All the outputs were exported to MS Excel and imported to Tableau.

5. Data Visualisation

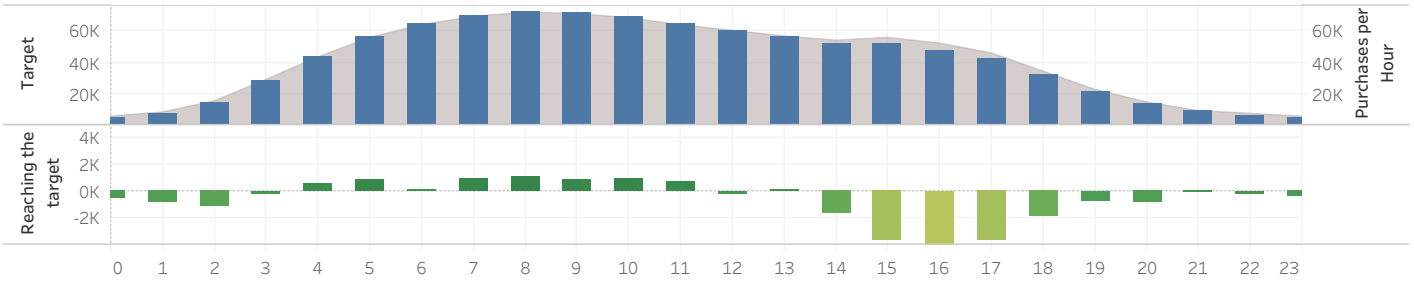
Top Revenue Brands

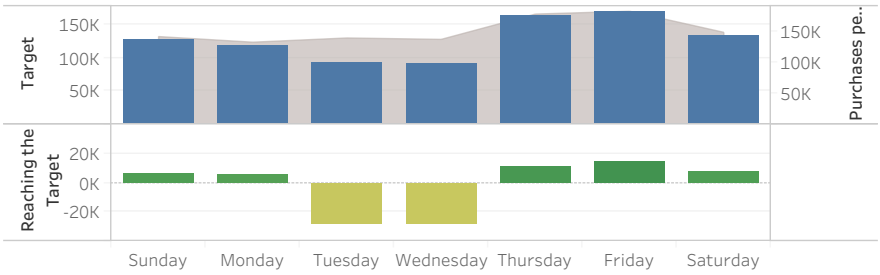


Top Revenue Products

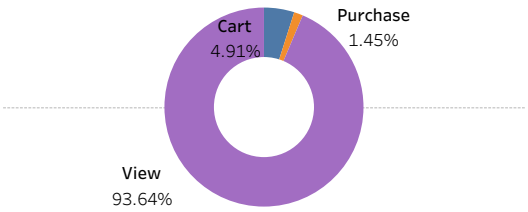


Traffic per hour / weekday





Events Comparison



6. Summary

Answering the initial questions asked:

1. Show the breakdown of the revenue generated by the largest brands
   The biggest brand that generated the highest revenue is Apple with $74,29M, followed by Samsung with $56.79M. Then, the sum of the highest value of revenue is distinguished by the smallest brands. What is worth noting that the three items mentioned above have a significantly larger market share than the next ones
2. Top 10 products generating the most revenue along with the number of units sold
   The list of products generating the highest revenue for the store can be found on the chart. Product ID 1005115 - $8.29M provided exceptionally high value. At the same time, there is no unambiguous correlation between the number of units sold and the revenue obtained.
3. What hours do most customers make purchases? + comparison to the target i.e. 30%
   Most customers make purchases between 8 AM and 9 AM. On the other hand, the fewest purchases were recorded between 11 PM and midnight. The conversion of 30% between adding to the cart and purchasing varies within the limits of the assumed goal. At the same time, the hours from 3 PM to 5 PM have the worst performance.
4. What days do most customers make purchases + comparison to the target i.e. 30%
   Most customers make purchases between Thursday and Friday. On the other hand, the fewest purchases were recorded between Tuesday and Wednesday. The conversion of 30% between adding to the cart and purchasing varies within the limits of the assumed goal. At the same time, Monday and Wednesday have the worst performance.
5. Show the division of events in the store
   Product's view: 93,64$ of total events
   Adding to cart: 4,91% of total events
   Purchase: 1,45% of total events