Taobao User Behavior Analysis

Course: FE-512-A Database Engineering

Instructor: Olorundamilola 'Dami' Kazeem

Group: 12

Members: Yating Liu, Xing Fang ¶

University: Stevens Institute of Technology

Semester: Spring 2019

In [1]:
```
%load_ext sql
```

In [2]:
```
%sql mysql+pymysql://root:@fe512_mysql/fe512db
```

Out[2]: 'Connected: root@fe512db'

In [3]:
```
%sql USE fe512db;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[3]: []

In [4]:
```
%sql SHOW DATABASES;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[4]:

Database
fe512db
information_schema
mysql
performance_schema
sys

In [5]:
```
%sql SELECT DATABASE();
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[5]:

DATABASE()
fe512db

1.Introduction

Background: With the development of science and technology, the pace of life has accelerated, and online shopping has gradually entered thousands of households. The surge in user volume has brought about the emergence of online shopping platforms. At the same time, some small platforms often cannot be operated for a long time due to lack of reasonable user-oriented measures.

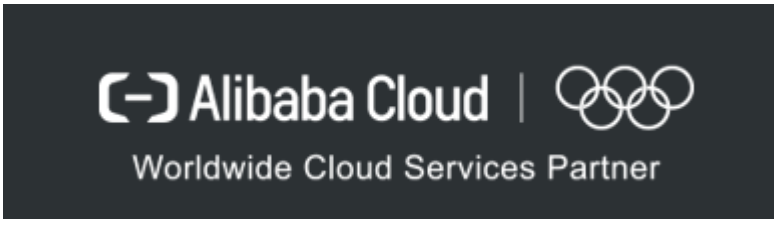
Through the analysis of user behavior of Taobao, the largest e-commerce platform in China, we try to find practical methods and provide suggestions to help e-commerce platforms increase sales, consolidate loyal users, and ultimately achieve long-term, profitable operating models.

Questions:

This analysis wants to solve the following business problems by analyzing Taobao user behavior data.

- ***Flow of quantity – number of each behavior***
- ***Conversion rate – PV/UV, CART/UV, FAV/UV, BUY/UV***
- ***User activity analysis***
- ***User consumption trend analysis***
- ***Retention rate***
- ***Repurchase number***
- ***RMF model***
- ***Item sales analysis***

2. Data



Source:

The dataset is provided by Alibaba Cloud, which is a platform where businesses meet top data scientists globally to solve the toughest industry problems. Alibaba Cloud also provide lots of standard datasets for academic use, and it is the only open data sharing platform of Ali. So, we select the dataset here about Taobao, which is a large online retail and business circle in the Asia-Pacific region, which was founded by Alibaba group in May 2003.

The data set contains all the behaviors (click, buy, add an item to shopping cart, favor) of about one million random users who had behaviors between November 25, 2017 and December 3, 2017. The size of data set is as follows: the number of users is about 1 million (987,994), the number of commodities is about 4.1 million (4162,024), the number of commodity categories is 9,439, and the total number of taobao user behavior records is 100 million (100,150,807).

User Behavior Data from Taobao for Recommendation
(https://tianchi.aliyun.com/dataset/dataDetail?dataId=649&userId=1 (https://tianchi.aliyun.com/dataset/dataDetail?dataId=649&userId=1))

Introduction:

Our data is mainly about user purchase history(the specific time when each user bought which product) , user behavior history(the specific time when each user had specific action) and item category (which category each item belongs to). Based on these information, we want to analyze customer's shopping behavior, and provide useful business suggestions to Taobao.

Size:

There are 100 million data records in the original data set, which is a huge amount of data. This analysis selected about 1 million records for analysis:

- User_purchase_history: 21258 rows
- Item_category: 1048575 rows
- User_behavior_history: 1048575 rows

Period:

November 25, 2017 - December 03, 2017

Data Dictionary

Field Name	Data Type	Description	Example
Timestamp	Integer	Timestamp of the behavior	1761333505
User ID	Integer	Serialized ID that represents a user	310413
Item ID	Integer	Serialized ID that represents an item	1203012
Category ID	Integer	Serialized ID that represents the category which the corresponding item belongs to	4163659
Behavior Type	String	Four behavior types	'pv', 'buy', 'cart', 'fav'

Datentime	**Timestamp**	**Timestamp of the behavior**	**2017-11-25 00:00:00**
Date	**CHAR**	**Date**	**2017-12-01**
Times	**CHAR**	**Time**	**10:01:16**

Samples of Data Records in Each Table

- User Purchase History

UserID	ItemID	Timestamp
739263	7385	1511705098

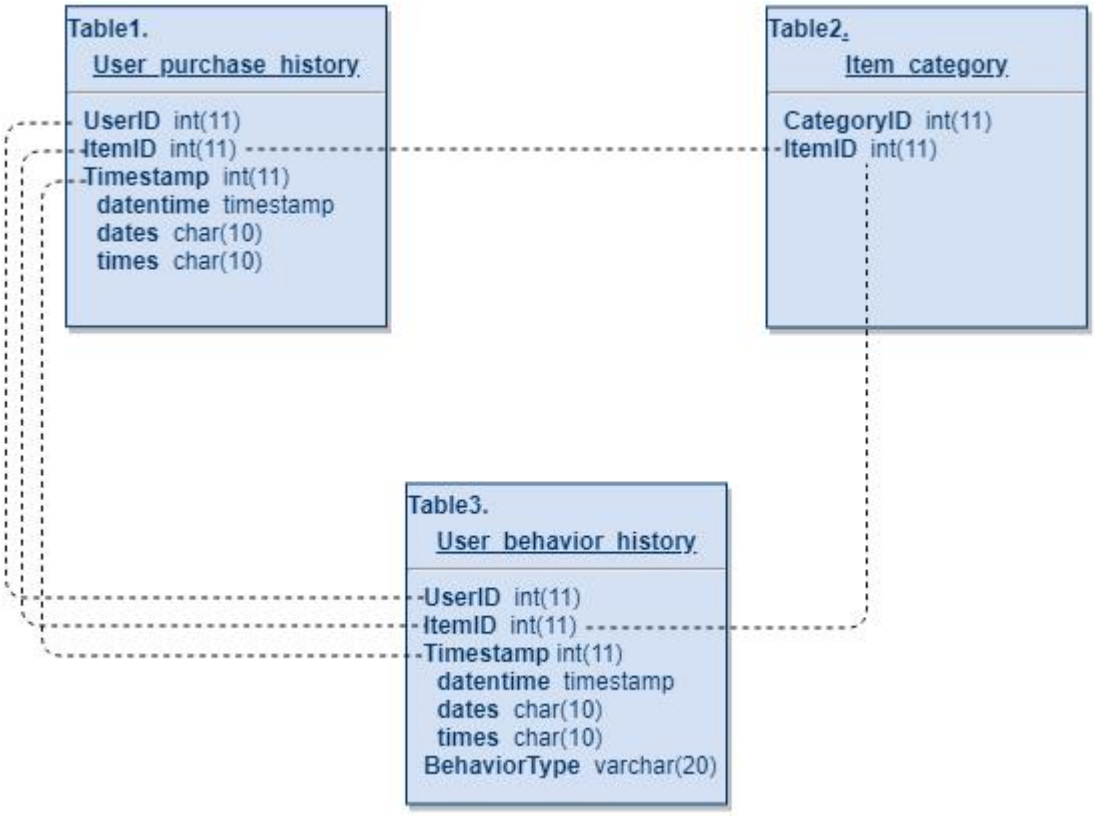
- User Behavior History

UserID	Behavior_type	Timestamp
411686	pv	1512122476

- Item Category

ItemID	CategoryID
4907788	4151801

3.Data Model



Design Reasons

- Analyze the loss of users from browsing to final purchase, and propose suggestions for improving conversion rate.
- Find the most active date of the user and the daily active time period during the study period to understand the user's behavior time mode.
- Determine which products and product categories have the highest purchase rate, find the most popular products, and optimize product sales.
- Study which users are the most important, find out the most core paid user groups, and push personalized product sales solutions based on their purchase preferences.

4.Preprocessing

- **Creating tables**
- **Changing the form of timestamp**
- **Time outlier processing**

Creating table - User Purchase History

In [7]:
```
%%sql
CREATE TABLE IF NOT EXISTS User_purchase_history(
    UserID INTEGER,
    ItemID INTEGER,
    Timestamp INTEGER);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[7]: []

In [8]:
```
%%sql
  LOAD DATA INFILE '/home/data/user_purchase_history.csv' INTO TABLE User_purchase_hi
  FIELDS
       TERMINATED BY ','
    LINES
       TERMINATED BY '\n'
        STARTING BY ''
        IGNORE 1 LINES
    (@UserID, @ItemID,@Timestamp)
    SET
     UserID = NULLIF(@UserID,''),
     ItemID = NULLIF(@ItemID,''),
     Timestamp = NULLIF(@Timestamp,'');
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
21258 rows affected.
```

Out[8]: []

Creating table - Item Category

In [9]:
```
%%sql
CREATE TABLE IF NOT EXISTS Item_category(
    ItemID INTEGER,
    CategoryID INTEGER);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[9]: []

In [10]:
```
%%sql
  LOAD DATA INFILE '/home/data/item_category.csv' INTO TABLE Item_category
  FIELDS
       TERMINATED BY ','
    LINES
       TERMINATED BY '\n'
        STARTING BY ''
        IGNORE 1 LINES
    (@ItemID, @CategoryID)
    SET
     ItemID = NULLIF(@ItemID,''),
     CategoryID = NULLIF(@CategoryID,'');
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1048575 rows affected.
```

Out[10]: []

Creating table - User Behavior History

In [11]:
```
%%sql
CREATE TABLE IF NOT EXISTS User_behavior_history(
    Timestamp INTEGER,
    UserID INTEGER,
    Behavior_type varchar(20));
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[11]: []

In [12]:
```
%%sql
  LOAD DATA INFILE '/home/data/user_behavior_history.csv' INTO TABLE User_behavior_hi
  FIELDS
       TERMINATED BY ','
    LINES
       TERMINATED BY '\n'
        STARTING BY ''
        IGNORE 1 LINES
    (@UserID, @Behavior_type, @Timestamp)
    SET
     UserID = NULLIF(@UserID,''),
     Behavior_type = NULLIF(@Behavior_type,''),
     Timestamp = NULLIF(@Timestamp,'');
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1048575 rows affected.
```

Out[12]: []

In [13]:
```
%%sql
SHOW TABLES;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
3 rows affected.
```

Out[13]:

Tables_in_fe512db
Item_category
User_behavior_history
User_purchase_history

Changing the form of timestamp - User_purchase_history

In [14]:
```
%sql ALTER TABLE User_purchase_history ADD COLUMN datentime TIMESTAMP(0) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[14]: []

In [15]:
```
%%sql
UPDATE User_purchase_history
SET datentime = FROM_UNIXTIME(Timestamp);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
21258 rows affected.
```

Out[15]: []

In [16]:
```
%%sql
ALTER TABLE User_purchase_history ADD COLUMN dates CHAR(10) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[16]: []

In [17]:
```
%%sql
UPDATE User_purchase_history
SET dates = SUBSTRING(datentime FROM 1 FOR 10);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
21258 rows affected.
```

Out[17]: []

In [18]:
```
%%sql
ALTER TABLE User_purchase_history ADD COLUMN times CHAR(10) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[18]: []

In [19]:
```
%%sql
UPDATE User_purchase_history
SET times = SUBSTRING(datentime FROM 12 FOR 8);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
21258 rows affected.
```

Out[19]: []

In [20]:
```
%%sql
SELECT datentime, dates, times
from User_purchase_history
LIMIT 5;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[20]:

datentime	dates	times
2017-11-26 14:04:58	2017-11-26	14:04:58
2017-11-28 23:40:22	2017-11-28	23:40:22
2017-11-26 08:25:09	2017-11-26	08:25:09
2017-12-03 14:55:05	2017-12-03	14:55:05
2017-11-27 15:08:56	2017-11-27	15:08:56

In [21]:
```
%sql DESCRIBE User_purchase_history;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
6 rows affected.
```

Out[21]:

Field	Type	Null	Key	Default	Extra
UserID	int(11)	YES		None	
ItemID	int(11)	YES		None	
Timestamp	int(11)	YES		None	
datentime	timestamp	YES		None	
dates	char(10)	YES		None	
times	char(10)	YES		None	

Changing the form of timestamp - User_behavior_history

In [22]:
```
%%sql
ALTER TABLE User_behavior_history ADD COLUMN datentime TIMESTAMP(0) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[22]: []

In [23]:
```
%%sql
UPDATE User_behavior_history
SET datentime = FROM_UNIXTIME(Timestamp);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1048575 rows affected.
```

Out[23]: []

In [24]:
```
%%sql
ALTER TABLE User_behavior_history ADD COLUMN dates CHAR(10) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[24]: []

In [25]:
```
%%sql
UPDATE User_behavior_history
SET dates = SUBSTRING(datentime FROM 1 FOR 10);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1048575 rows affected.
```

Out[25]: []

In [26]:
```
%%sql
ALTER TABLE User_behavior_history ADD COLUMN times CHAR(10)NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[26]: []

In [27]:
```
%%sql
UPDATE User_behavior_history
SET times = SUBSTRING(datentime FROM 12 FOR 8);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1048575 rows affected.
```

Out[27]: []

In [28]:
```
%%sql
SELECT times, dates, datentime
from User_behavior_history
LIMIT 5;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[28]:

times	dates	datentime
10:01:16	2017-12-01	2017-12-01 10:01:16
11:57:01	2017-12-03	2017-12-03 11:57:01
10:57:07	2017-11-27	2017-11-27 10:57:07
10:36:00	2017-12-01	2017-12-01 10:36:00
04:45:53	2017-11-28	2017-11-28 04:45:53

In [41]:
```
%%sql
DESCRIBE User_behavior_history;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
6 rows affected.
```

Out[41]:

Field	Type	Null	Key	Default	Extra
Timestamp	int(11)	YES		None	
UserID	int(11)	YES		None	
Behavior_type	varchar(20)	YES		None	
datentime	timestamp	YES		None	
dates	char(10)	YES		None	
times	char(10)	YES		None	

Time Outlier Processing - Only Saving Time Between November 25 to December 03, 2017

User_purchase_history

In [31]:
```
%%sql
SELECT MAX(Timestamp),
       MIN(Timestamp),
       MAX(datentime),
       MIN(datentime)
FROM User_purchase_history;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[31]:

MAX(Timestamp)	MIN(Timestamp)	MAX(datentime)	MIN(datentime)
1512316789	1511539214	2017-12-03 15:59:49	2017-11-24 16:00:14

In [32]:
```
%%sql
DELETE FROM User_purchase_history
WHERE datentime < '2017-11-25 00:00:00'
OR datentime > '2017-12-04 00:00:00';
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
179 rows affected.
```

Out[32]: []

In [33]:
```
%%sql
SELECT MAX(Timestamp),
       MIN(Timestamp),
       MAX(datentime),
       MIN(datentime)
FROM User_purchase_history;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[33]:

MAX(Timestamp)	MIN(Timestamp)	MAX(datentime)	MIN(datentime)
1512316789	1511568132	2017-12-03 15:59:49	2017-11-25 00:02:12

User_behavior_history

In [34]:
```
%%sql
SELECT MAX(Timestamp),
       MIN(Timestamp),
       MAX(datentime),
       MIN(datentime)
FROM User_behavior_history;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[34]:

MAX(Timestamp)	MIN(Timestamp)	MAX(datentime)	MIN(datentime)
1761333505	-1553400454	2025-10-24 19:18:25	1970-01-01 10:34:57

In [35]:
```
%%sql
DELETE FROM User_behavior_history
WHERE datentime < '2017-11-25 00:00:00'
OR datentime > '2017-12-04 00:00:00';
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
13049 rows affected.
```

Out[35]: []

In [36]:
```
%%sql
SELECT MAX(Timestamp),
       MIN(Timestamp),
       MAX(datentime),
       MIN(datentime)
FROM User_behavior_history;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[36]:

MAX(Timestamp)	MIN(Timestamp)	MAX(datentime)	MIN(datentime)
1512321645	-1553400454	2017-12-03 17:20:45	2017-11-25 00:00:00

5. Questions & Answers

5.1 Flow - Quantity - Number of Each Behavior

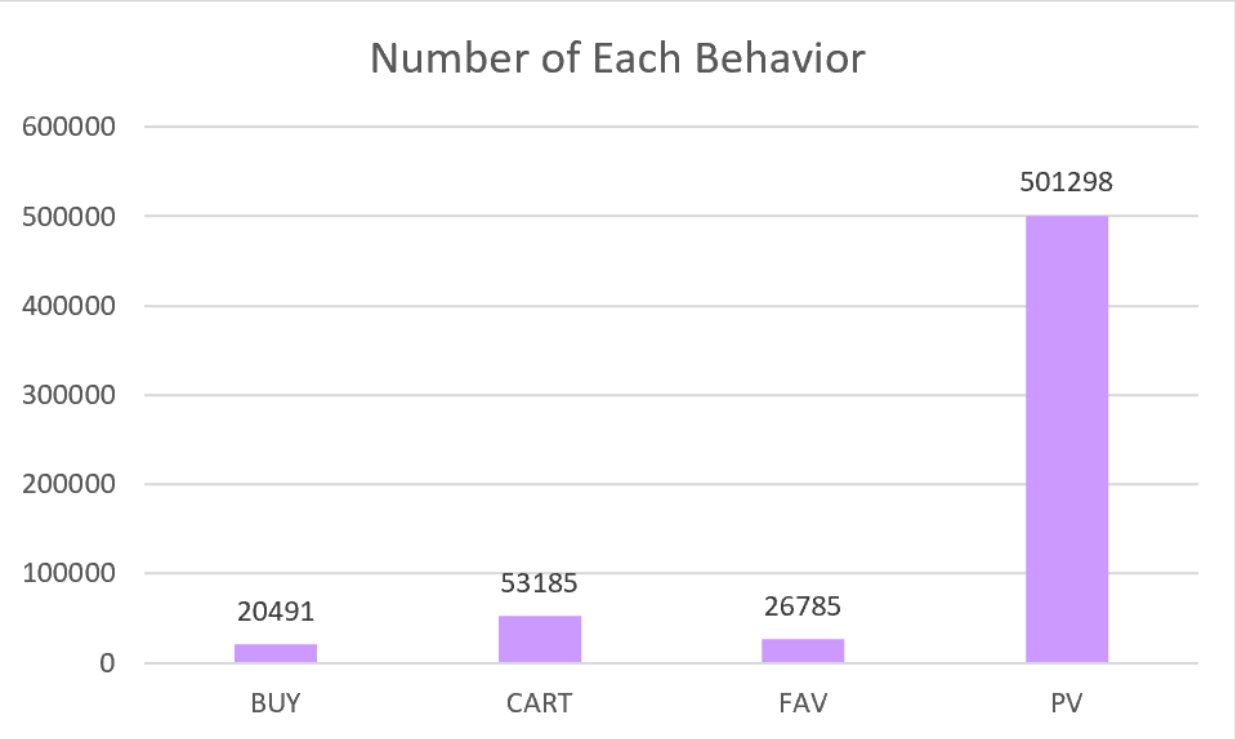
how many users have each behavior

In [43]:
```
%%sql
SELECT Behavior_type,
       COUNT(DISTINCT UserID) AS Number
FROM User_behavior_history
GROUP BY Behavior_type;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
4 rows affected.
```

Out[43]:

Behavior_type	Number
buy	20491
cart	53185
fav	26785
pv	501298



- ***Page viewing takes the most part of user behavior.***

5.2. Conversion Rate

- PV/UV
- CART/UV
- FAV/UV
- BUY/UV

Conversion Rate - Bounce Rate:PV/UV

- ***Percentage of people who just view but dont't favorite, cart or buy.***

Number of people who just viewed

In [44]:
```
%%sql
SELECT COUNT(DISTINCT UserID)
FROM User_behavior_history
WHERE UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_
AND UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_ty
AND UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_ty
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[44]:

COUNT(DISTINCT UserID)
439601

PV/UV

In [45]:
```
%%sql
SELECT(SELECT COUNT(DISTINCT UserID)
FROM User_behavior_history
WHERE UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_
AND UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_ty
AND UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_ty
FROM User_behavior_history) AS 'Bounce Rate';
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[45]:

Bounce Rate
0.8194

Conversion Rate - CART/UV

- ***Percentage of people who put items in cart but don't buy.***

In [46]:
```
%%sql
SELECT(SELECT COUNT(DISTINCT UserID)
FROM User_behavior_history
WHERE Behavior_type = 'cart'
AND UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_ty
/(SELECT COUNT(DISTINCT UserID) AS 'UV'
FROM User_behavior_history) AS 'CART/UV';
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[46]:

CART/UV
0.0962

Conversion Rate - FAV/UV

- ***Percentage of people who take the action of "favorite" but don't buy.***

In [47]:
```
%%sql
SELECT(SELECT COUNT(DISTINCT UserID)
FROM User_behavior_history
WHERE Behavior_type = 'fav'
AND UserID NOT IN(SELECT DISTINCT UserID FROM User_behavior_history WHERE Behavior_ty
/(SELECT COUNT(DISTINCT UserID) AS 'UV'
FROM User_behavior_history) AS 'FAV/UV';
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[47]:

FAV/UV
0.0487

Conversion Rate - BUY/UV

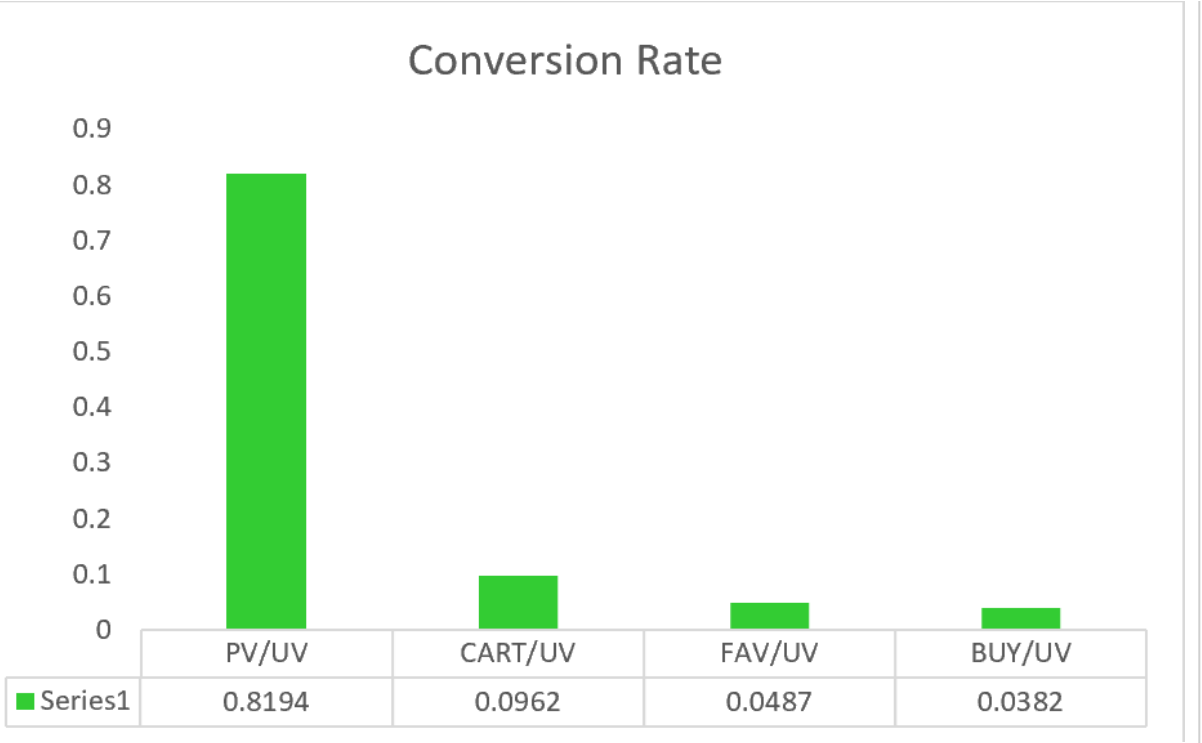
- ***Percentage of people who finally buy.***

In [48]:
```
%%sql
SELECT(SELECT COUNT(DISTINCT UserID)
FROM User_behavior_history
WHERE Behavior_type = 'buy')
/(SELECT COUNT(DISTINCT UserID) AS 'UV'
FROM User_behavior_history) AS 'BUY/UV';
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1 rows affected.
```

Out[48]:

BUY/UV
0.0382



- ***82%of all users just view pages but have no other actions.***
- ***Compared with "favorite", people more like to take items into cart.***
- ***Only 3% of users will finally buy.***

5.3. User Activity Analysis

- UAA-hour
- UAA-day

UAA-hour

The sum number of each behavior during each hour

Add column hour to table User_behavior_history

In [49]:
```
%%sql
ALTER TABLE User_behavior_history ADD COLUMN hour CHAR(10) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[49]: []

In [50]:
```
%%sql
UPDATE User_behavior_history
SET hour = SUBSTRING(datentime FROM 12 FOR 2);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
1035526 rows affected.
```

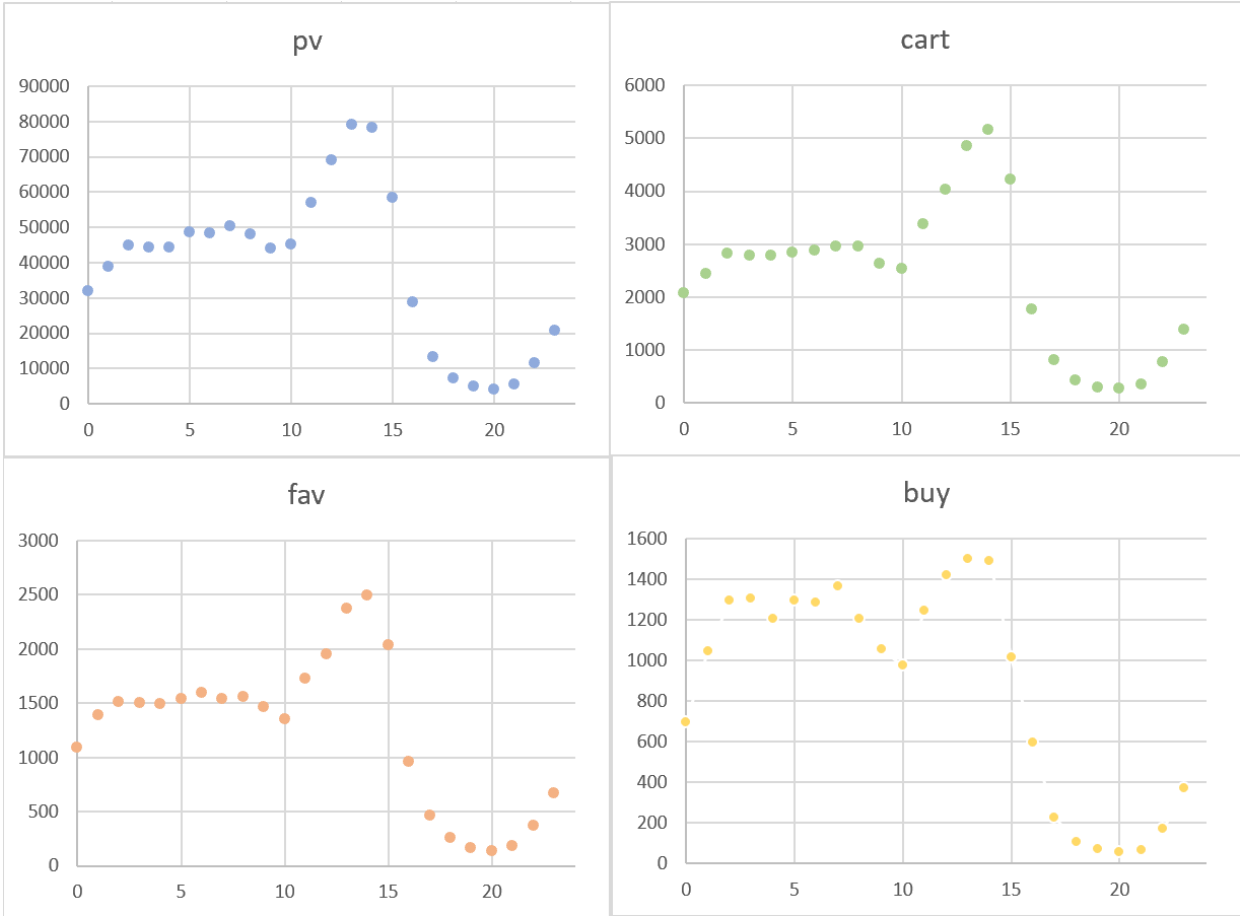
Out[50]: []

In [52]:
```
%%sql
SELECT hour,
SUM(CASE WHEN Behavior_type='pv' THEN 1 ELSE 0 END) AS 'Num_PV',
SUM(CASE WHEN Behavior_type='fav' THEN 1 ELSE 0 END) AS 'Num_FAV',
SUM(CASE WHEN Behavior_type='cart' THEN 1 ELSE 0 END) AS 'Num_CART',
SUM(CASE WHEN Behavior_type='buy' THEN 1 ELSE 0 END) AS 'Num_BUY'
FROM User_behavior_history
GROUP BY hour
ORDER BY hour;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
25 rows affected.
```

Out[52]:

hour	Num_PV	Num_FAV	Num_CART	Num_BUY
None	1	0	0	0
00	31942	1093	2077	698
01	38806	1395	2439	1044
02	44938	1516	2829	1295
03	44336	1507	2786	1308
04	44279	1491	2786	1206
05	48613	1541	2840	1295
06	48457	1598	2880	1288
07	50249	1541	2952	1368
08	48181	1562	2962	1205
09	43975	1462	2632	1057
10	45149	1357	2544	976
11	57068	1729	3370	1244
12	69139	1948	4035	1421
13	78971	2369	4863	1503
14	78141	2493	5161	1492
15	58318	2039	4231	1018
16	28835	962	1764	597
17	13431	469	810	228
18	7312	256	438	106
19	4988	169	293	71
20	4212	141	270	54
21	5547	188	355	64
22	11493	372	777	171
23	20717	668	1389	370



- ***During the period from 12 to 15, it is the active peak period for users.***
- ***8：00 pm is the least active time for users. However after 8pm, it is gradually getting active again.***

UAA-day

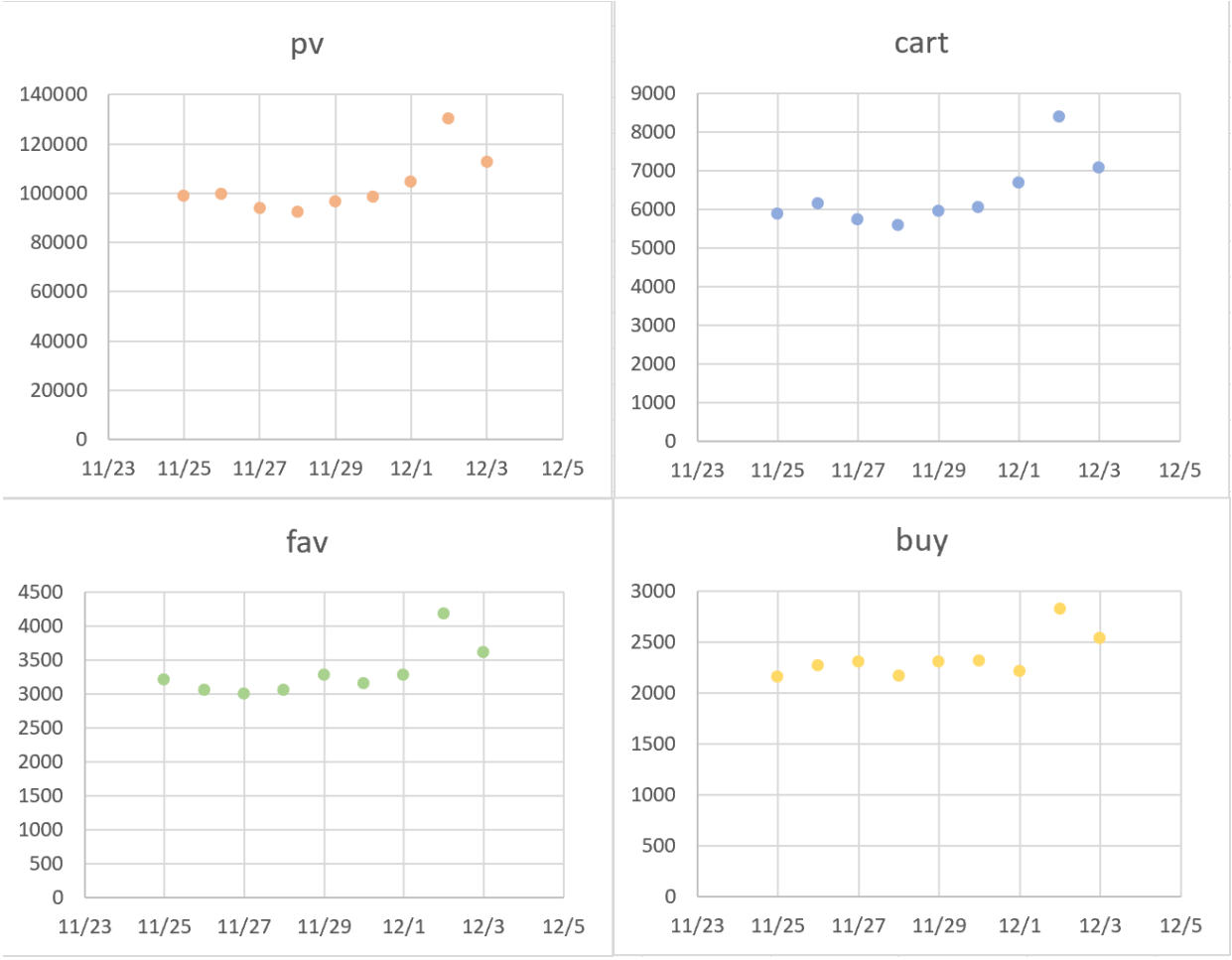
The sum number of each behavior during each day

In [54]:
```
%%sql
SELECT dates,
SUM(CASE WHEN Behavior_type='pv' THEN 1 ELSE 0 END) AS 'Num_PV',
SUM(CASE WHEN Behavior_type='fav' THEN 1 ELSE 0 END) AS 'Num_FAV',
SUM(CASE WHEN Behavior_type='cart' THEN 1 ELSE 0 END) AS 'Num_CART',
SUM(CASE WHEN Behavior_type='buy' THEN 1 ELSE 0 END) AS 'Num_BUY'
FROM User_behavior_history
GROUP BY dates
ORDER BY dates ASC;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

Out[54]:

dates	Num_PV	Num_FAV	Num_CART	Num_BUY
None	1	0	0	0
2017-11-25	98971	3211	5873	2153
2017-11-26	99593	3054	6145	2264
2017-11-27	93793	3011	5721	2302
2017-11-28	92457	3060	5591	2165
2017-11-29	96684	3279	5941	2306
2017-11-30	98262	3161	6059	2318
2017-12-01	104507	3282	6694	2213
2017-12-02	130353	4187	8389	2820
2017-12-03	112477	3621	7070	2538



- ***The sudden increase in 12/2 and 12/3 may be related to DOUBLE 12 which is similar to Black Friday in US; also the two days are weekends so the increase may have a connection with weekends.***

5.4.User Consumption Trend Analysis(Based on Hour)

- Number of Users
- Number of Orders
- Number of Products
- Regression Model

Add column hour to table User_purchase_history

In [71]:
```
%%sql
ALTER TABLE User_purchase_history ADD COLUMN hour CHAR(10) NULL;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
0 rows affected.
```

Out[71]: []

In [72]:
```
%%sql
UPDATE User_purchase_history
SET hour = SUBSTRING(datentime FROM 12 FOR 2);
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
21079 rows affected.
```

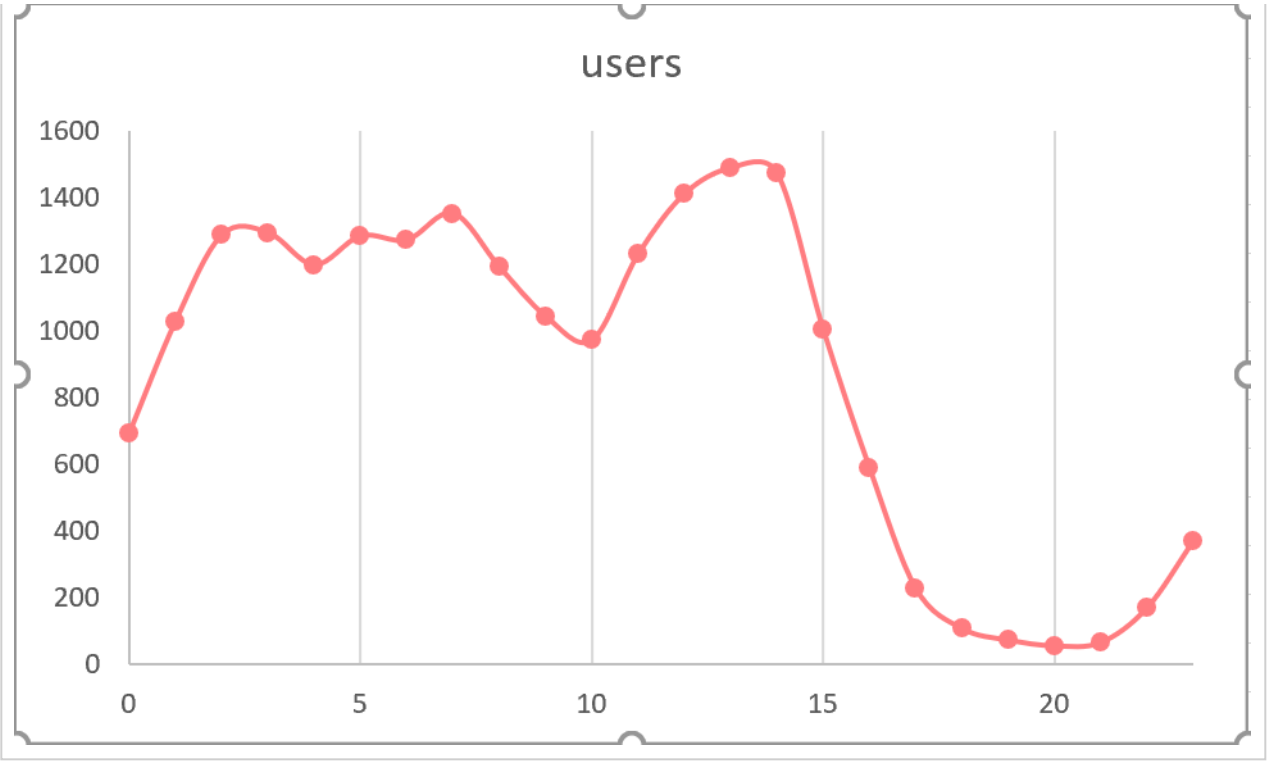
Out[72]: []

Number of Users

- ***During each hour, how many people purchased on Taoba.***

In [73]:
```
%%sql
SELECT hour,COUNT(DISTINCT UserID)
FROM User_purchase_history
GROUP BY hour
ORDER BY hour;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
24 rows affected.
```



Number of Orders

- ***During each hour, how many orders are generated on Taobao.***

In [75]:
```
%%sql
SELECT hour, count(UserID)
FROM User_purchase_history
GROUP BY hour
ORDER BY hour;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
24 rows affected.
```

Out[75]:

hour	count(UserID)
00	698
01	1044
02	1295
03	1308
04	1206
05	1295
06	1288
07	1368
08	1205
09	1057
10	976
11	1244
12	1421
13	1503
14	1492
15	1018
16	597
17	228
18	106
19	71
20	54
21	64
22	171
23	370

Number of Products

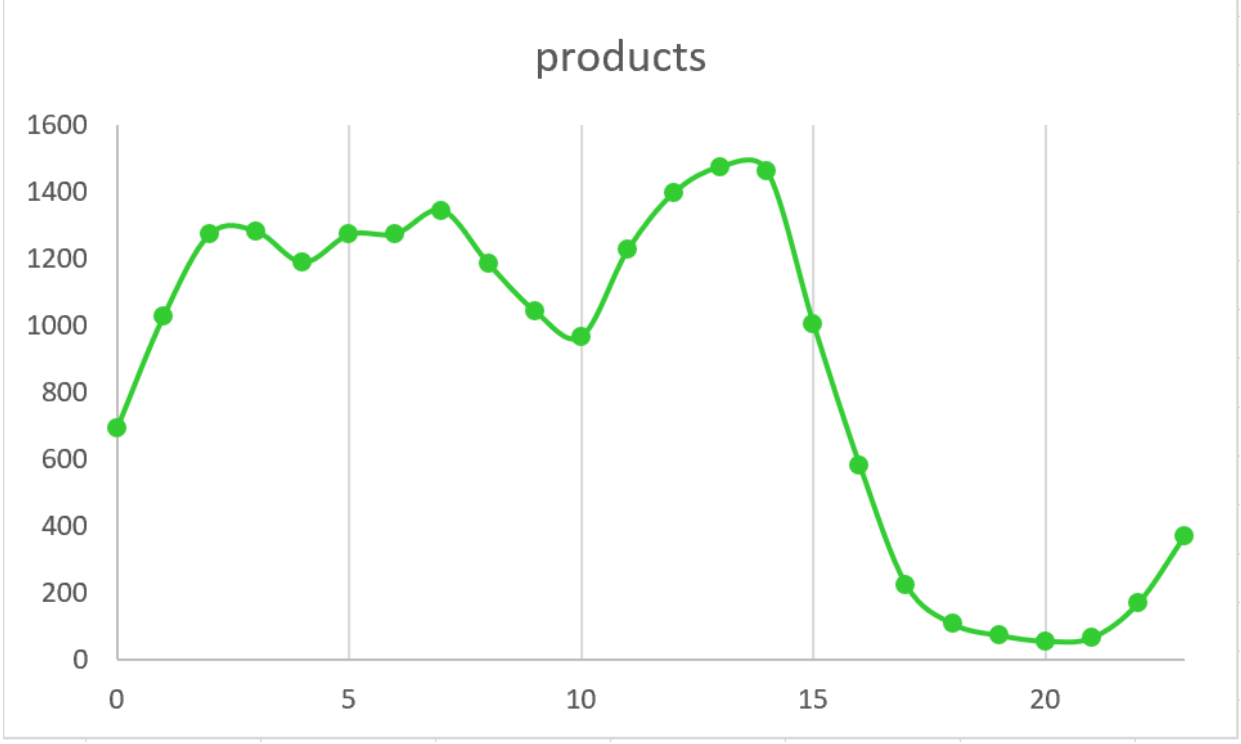
- ***During each hour, how many products are sold.***

In [78]:
```
%%sql
SELECT hour,COUNT(DISTINCT ItemID)
FROM User_purchase_history
GROUP BY hour
ORDER BY hour;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
24 rows affected.
```

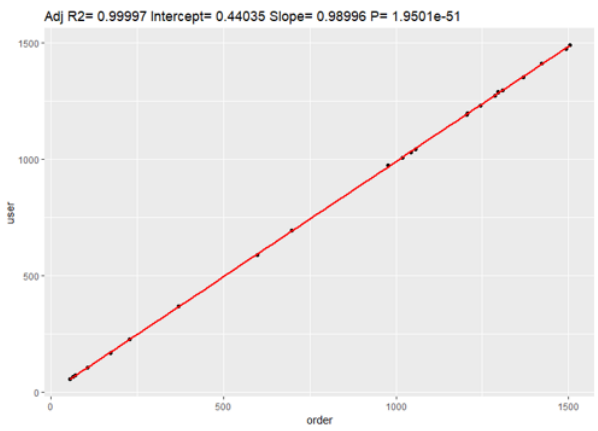
Out[78]:

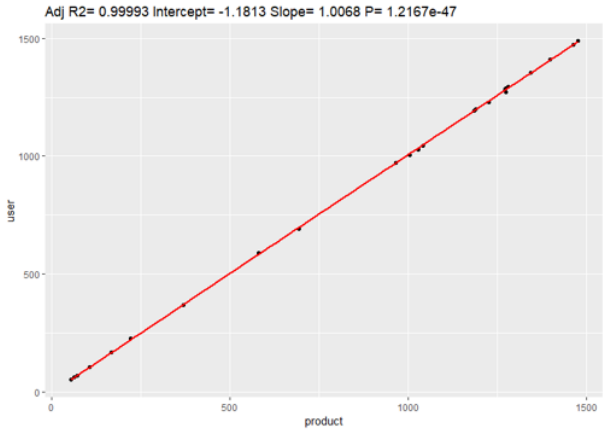
hour	COUNT(DISTINCT ItemID)
00	694
01	1029
02	1275
03	1282
04	1189
05	1273
06	1275
07	1345
08	1185
09	1043
10	967
11	1227
12	1399
13	1476
14	1464
15	1006
16	581
17	223
18	106
19	71
20	54
21	64
22	168
23	369

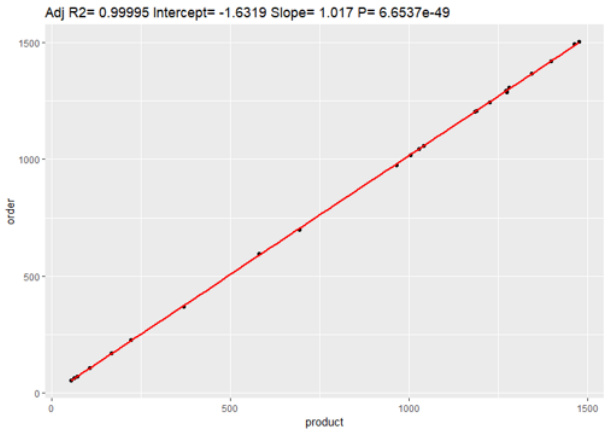


Regression Models

- ***To further test the relationship among 3 indicators，we used linear regression models between any 2 of them.***







- ***From the output of 3 regression models, any two indicators are positively correlated. we can conclude that the more users there are, the more orders there are, and the more types of goods are sold.***

5.5.Retention Rate

- ***Wiki： Retention rate is the ratio of the number of retained customers to the number at risk***
- ***In our project, we calculated the ratio of people who still have any actions on Taobao after the first day they logged in.***

First step: Select UserID, dates

In [43]:
```
%%sql
SELECT UserID,dates
FROM User_behavior_history
GROUP BY 1,2
LIMIT 5;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[43]:

UserID	dates
411686	2017-12-01
322567	2017-12-03
211779	2017-11-27
421743	2017-12-01
689964	2017-11-28

Second step: Calculate the first day of each user

- ***first_day: The day users had the first action during the period of our raw data.***

In [44]:
```
%%sql
SELECT b.UserID, b.dates,c.first_day
FROM
    (SELECT UserID,dates
      FROM User_behavior_history
      GROUP BY 1,2) b
LEFT JOIN
    (SELECT UserID,min(dates) first_day
      FROM
          (SELECT UserID,dates
            FROM User_behavior_history
            GROUP BY 1,2) a
      GROUP BY 1) c
ON b.UserID = c.UserID
ORDER BY 1,2
LIMIT 5;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[44]:

UserID	dates	first_day
1	2017-11-29	2017-11-29
1	2017-12-02	2017-11-29
3	2017-12-01	2017-12-01
4	2017-11-28	2017-11-28
4	2017-12-03	2017-11-28

Third step: Calculate the time difference between every login time and first time for each user

In [46]:
```
%%sql
SELECT UserID,dates,first_day,DATEDIFF(dates,first_day) AS by_day
FROM
    (SELECT b.UserID,  b.dates,c.first_day
FROM
    (SELECT UserID,dates
      FROM User_behavior_history
      GROUP BY 1,2) b
LEFT JOIN
    (SELECT UserID,min(dates) first_day
      FROM
          (SELECT UserID,dates
            FROM User_behavior_history
            GROUP BY 1,2) a
      GROUP BY 1) c
ON b.UserID = c.UserID
ORDER BY 1,2) e
ORDER BY 1,2
LIMIT 5;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
5 rows affected.
```

Out[46]:

UserID	dates	first_day	by_day
1	2017-11-29	2017-11-29	0
1	2017-12-02	2017-11-29	3
3	2017-12-01	2017-12-01	0
4	2017-11-28	2017-11-28	0
4	2017-12-03	2017-11-28	5

Fourth step: Calculate 1st-7th day retention rate

In [48]:
```
%%sql
SELECT first_day,
       SUM(CASE WHEN by_day=1 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
       SUM(CASE WHEN by_day=2 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
       SUM(CASE WHEN by_day=3 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
       SUM(CASE WHEN by_day=4 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
       SUM(CASE WHEN by_day=5 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
       SUM(CASE WHEN by_day=6 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
       SUM(CASE WHEN by_day=7 THEN 1 ELSE 0 END)/SUM(CASE WHEN by_day=0 THEN 1 ELSE (
FROM
    (SELECT UserID,dates,first_day,DATEDIFF(dates,first_day) AS by_day
FROM
    (SELECT b.UserID,  b.dates,c.first_day
FROM
    (SELECT UserID,dates
      FROM User_behavior_history
      GROUP BY 1,2) b
LEFT JOIN
    (SELECT UserID,min(dates) first_day
      FROM
          (SELECT UserID,dates
            FROM User_behavior_history
            GROUP BY 1,2) a
      GROUP BY 1) c
ON b.UserID = c.UserID
ORDER BY 1,2) e
ORDER BY 1,2) f
GROUP BY 1
ORDER BY 1;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
9 rows affected.
```

Out[48]:

first_day	day_1	day_2	day_3	day_4	day_5	day_6	day_7
2017-11-25	0.1604	0.1373	0.1289	0.1309	0.1290	0.1355	0.1617
2017-11-26	0.1425	0.1293	0.1290	0.1253	0.1293	0.1559	0.1374
2017-11-27	0.1395	0.1343	0.1297	0.1275	0.1479	0.1294	0.0000
2017-11-28	0.1384	0.1290	0.1264	0.1456	0.1287	0.0000	0.0000
2017-11-29	0.1334	0.1277	0.1469	0.1268	0.0000	0.0000	0.0000
2017-11-30	0.1316	0.1449	0.1234	0.0000	0.0000	0.0000	0.0000
2017-12-01	0.1490	0.1253	0.0000	0.0000	0.0000	0.0000	0.0000
2017-12-02	0.1219	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2017-12-03	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

- ***From the output, we can see retention rate of Taobao is quite low, no matter the first_day retention rate or the seventh-day retention rate.***
- ***Because of the limitation of time span of raw data, the number of retention rate is decreasing gradually from11/27/2017.***

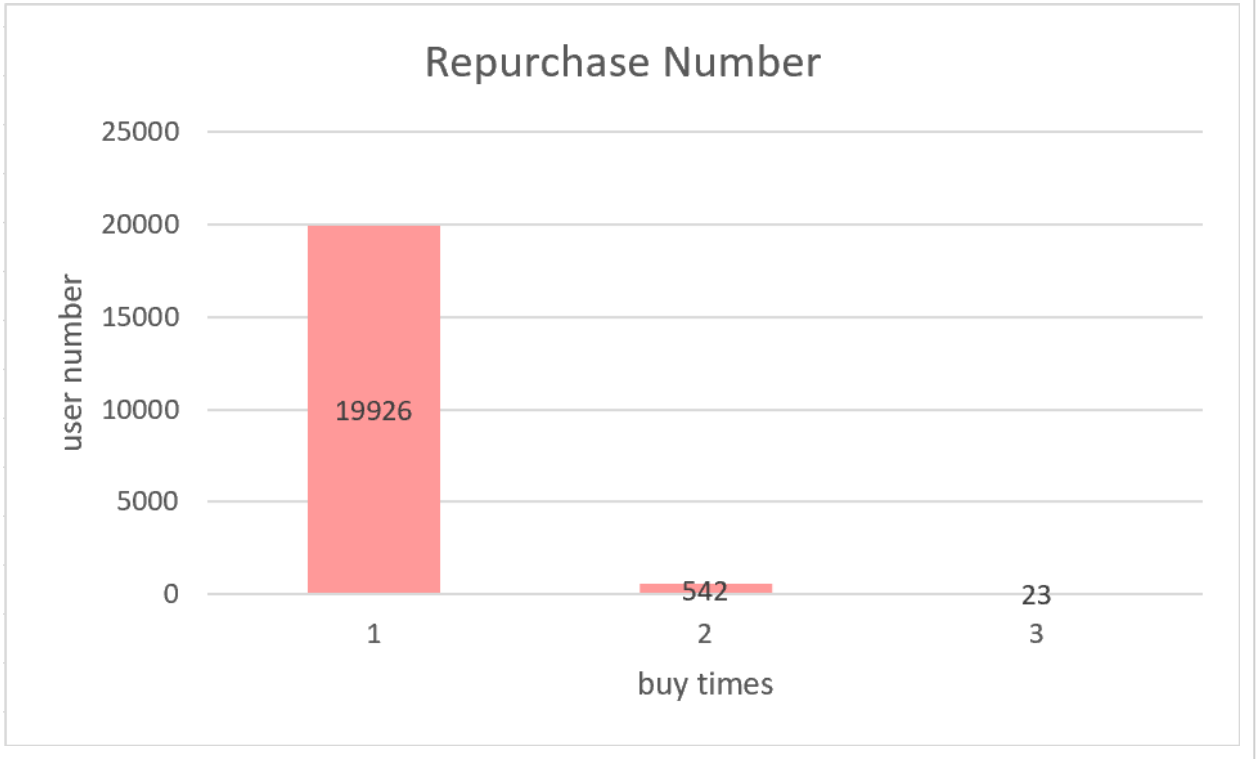
5.6.Repurchase Number

In [57]:
```
%%sql
SELECT buy_times,COUNT(buy_times) User_number
FROM
   (SELECT UserID,COUNT(UserID) AS buy_times
    FROM User_purchase_history
    GROUP BY UserID) a
GROUP BY buy_times
ORDER BY buy_times
LIMIT 10;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
3 rows affected.
```

Out[57]:

buy_times	User_number
1	19926
2	542
3	23



- ***Almost all users just bought once on Taobao, and users only shop at Taobao for up to 3 times during this period.***

5.7. RFM Model

Using RFM model to rate users based on their purchase behavior and divide users into different groups.

RFM is a method used for analyzing customer value.

RFM stands for the three dimensions:

- **Recency – How recently did the customer purchase?**
- **Frequency – How often do they purchase?**
- **Monetary Value – How much do they spend?**

Because the data source does not contain monetary value, we score customer value based on the R and F.

First step: Creating table RFM, which contains UserID, the rank of recent purchasing, the rank of frequency of purchasing and UserValue.

Here, the last day of the period is December 3, and customers shopping that day means they purchased recently. The later they purchased, the higher recency. And we set the number of a user purchasing as the frequency. The more they purchased, the higher frequency. So, when the rank of recency or frequency is greater than a half of 20491, it returns 0; if not, 1. By the way, 20491 is the number of rows, that is, the number of users.

Then we use 'concat' function to combine the two values, finally we get the uservalue.

In [46]:
```
%%sql
CREATE TABLE RFM(
SELECT R.UserID,F.Frequency,R.RecentRank,F.FreqRank,
CONCAT(CASE WHEN RecentRank<=(20491)/2 THEN '0'
            ELSE '1' END ,
       CASE WHEN FreqRank<=(20491)/2 THEN '0'
            ELSE '1' END)
            AS UserValue
FROM
(SELECT a.*,(@rank:=@rank+1) as RecentRank
FROM
((SELECT UserID,DATEDIFF('2017-12-04',MAX(datentime)) AS Recent
FROM User_purchase_history
GROUP BY UserID
ORDER BY Recent) AS a ,(SELECT @rank:=0) AS b )) AS R,
(SELECT a.*,@rank1:=@rank1+1 AS FreqRank
FROM
((SELECT UserID,COUNT(*) AS Frequency
FROM User_purchase_history
GROUP BY UserID
ORDER BY Frequency DESC) AS a ,(SELECT @rank1:=0) AS b)) AS F
WHERE R.UserID=F.UserID)
;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
20491 rows affected.
```

Out[46]: []

Second step: Adding the corresponding label to each UserValue

Valuable customers have purchased multiple times but have not purchased items recently. Important customers have purchased multiple times and have recently purchased items. Retained customers have fewer purchases but have not purchased items recently. Potential customers have purchased fewer times but have recently purchased items.

In [38]:
```
%%sql
SELECT *,
(CASE
WHEN UserValue='00' THEN 'Valued customer'
WHEN UserValue='10' THEN 'Important customers'
WHEN UserValue='01' THEN 'Retained customers'
WHEN UserValue='11' THEN 'Potential customers'
END) AS Label
FROM RFM
LIMIT 10
;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

Out[38]:

UserID	Frequency	RecentRank	FreqRank	UserValue	Label
310413	1	1.0	17641.0	01	Retained customers
729803	1	2.0	4295.0	00	Valued customer
671474	1	3.0	20266.0	01	Retained customers
665313	1	4.0	19598.0	01	Retained customers
546471	1	5.0	8859.0	00	Valued customer
790596	1	6.0	1189.0	00	Valued customer
979745	1	7.0	4908.0	00	Valued customer
500640	1	8.0	18639.0	01	Retained customers
837917	1	9.0	19889.0	01	Retained customers
705129	1	10.0	8878.0	00	Valued customer

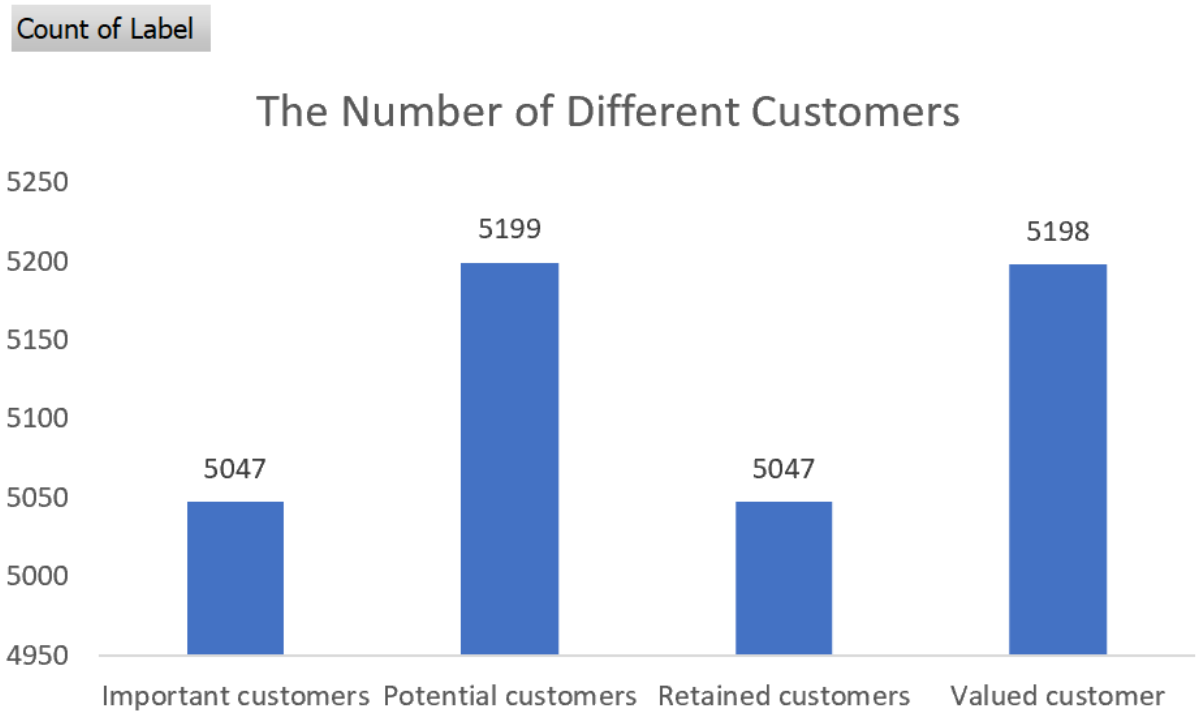
So, we can divide customers into 4 groups.

- **Valued customers purchased many times, but not recently.**
- **Important customers purchased many times, including recently.**
- **Retained customers purchased few times and not recently.**
- **Potential customers purchased few times, but recently.**

1. Users with high R and F scores are the most important users in the system, and they need to be focused on the recommendation activities.

2. Users with low R and low F are not sticky and have a short consumption time. The operation needs to focus on these users.

3. Users with low R and low F can be called back through discounts, promotions, redemption and other activities.



- **The number of important customers is the same as the number of retained customers,while they both less than the number of potential customers and the number of valued customers.**
- **This may because we sample the dataset into a smaller one and we do not have data about monetary value, so we could not divide them more accurately.**

5.8. Item Sales Analysis

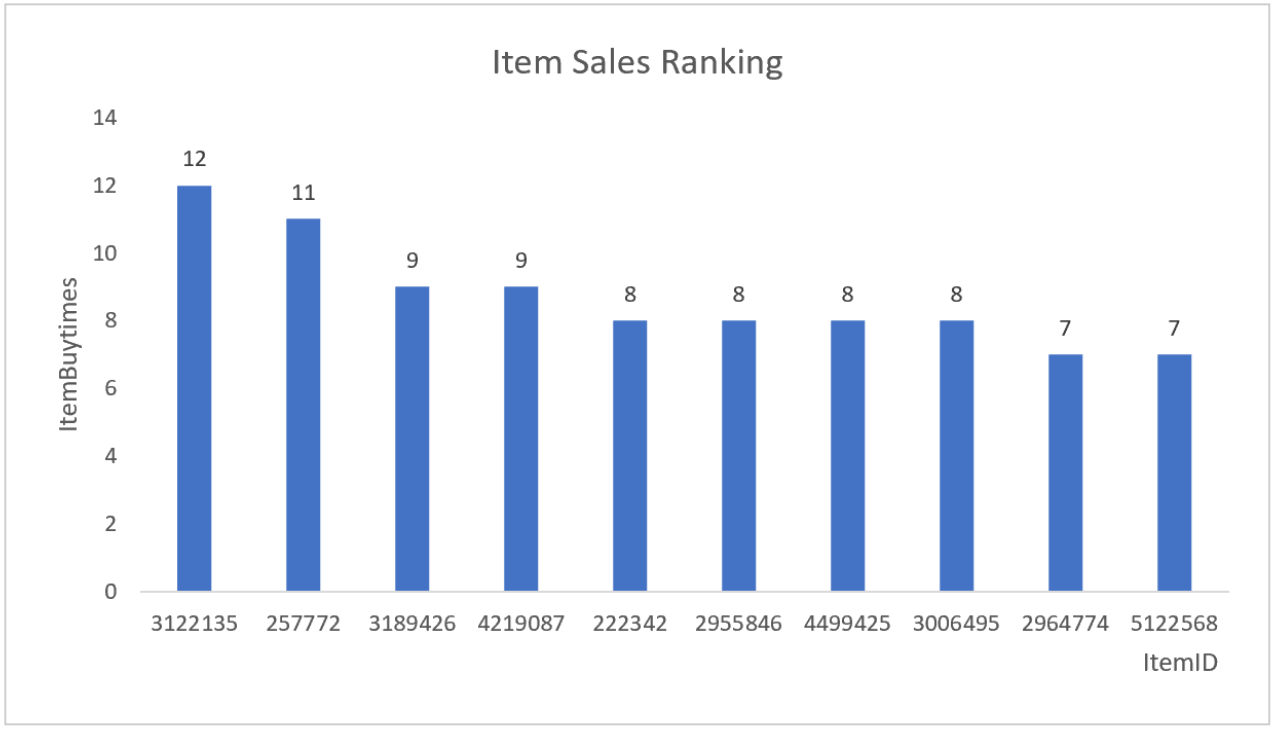
In this section, we try to optimize item sales by finding the item and item category with the highest purchase rate, that is, the most popular item and item category.

In [49]:
```
%%sql
SELECT ItemID, COUNT(*) AS ItemBuytimes
FROM
User_purchase_history
GROUP BY ItemID
ORDER BY ItemBuytimes DESC
LIMIT 10;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

Out[49]:

ItemID	ItemBuytimes
3122135	12
257772	11
3189426	9
4219087	9
222342	8
2955846	8
4499425	8
3006495	8
2964774	7
5122568	7

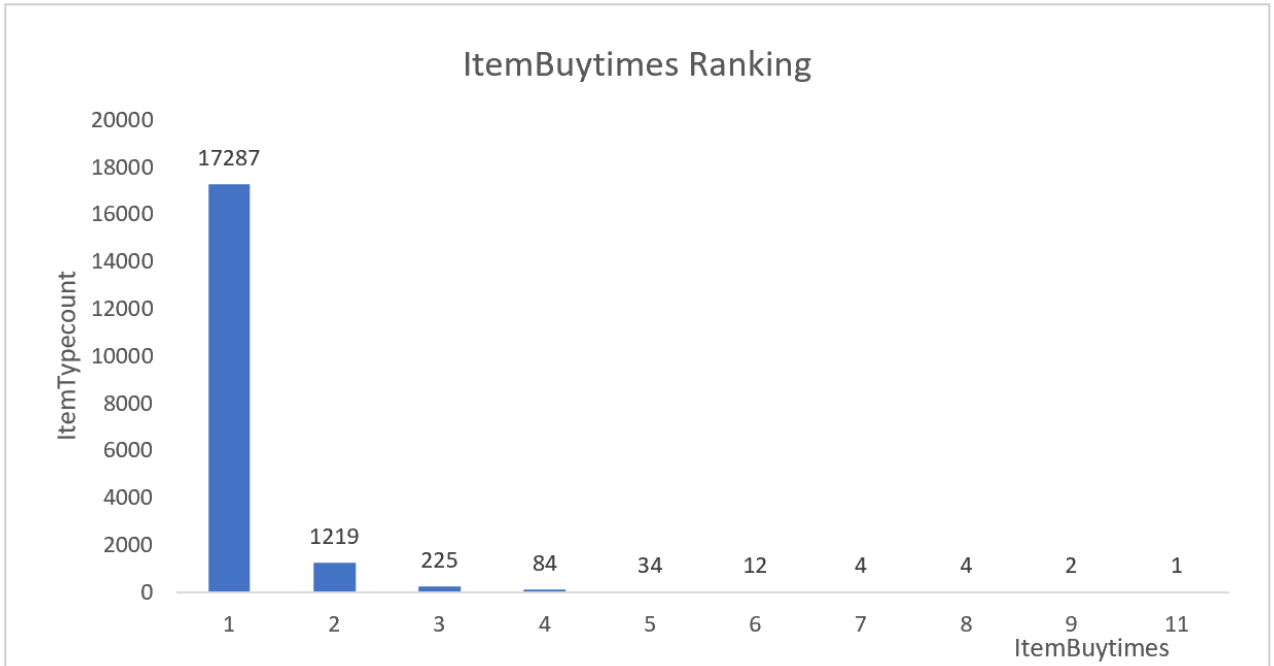


In [37]:
```
%%sql
SELECT ItemBuytimes, COUNT(*) AS ItemTypecount
FROM
(SELECT COUNT(UserID) AS ItemBuytimes
FROM User_purchase_history
GROUP BY ItemID) AS ItemBuypool
GROUP BY ItemBuytimes
ORDER BY ItemBuytimes ASC
;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
11 rows affected.
```

Out[37]:

ItemBuytimes	ItemTypecount
1	17287
2	1219
3	225
4	84
5	34
6	12
7	4
8	4
9	2
11	1



- ***There are 17287 types of items purchased only once, and there are 1219 items purchased two times.***

- ***Most items are purchased only once during this period, indicating that the "explosion" is still not formed and the item sales are relatively low.***

So what about the sales rankings of item categories?

To analyze the problem, we left join table User_purchase_history and Item_category, and then group by CategoryID.

In [38]:
```
%%sql
SELECT UserID,User_purchase_history.ItemID,CategoryID
FROM User_purchase_history
LEFT JOIN Item_category
ON User_purchase_history.ItemID=Item_category.ItemID
LIMIT 10;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

Out[38]:

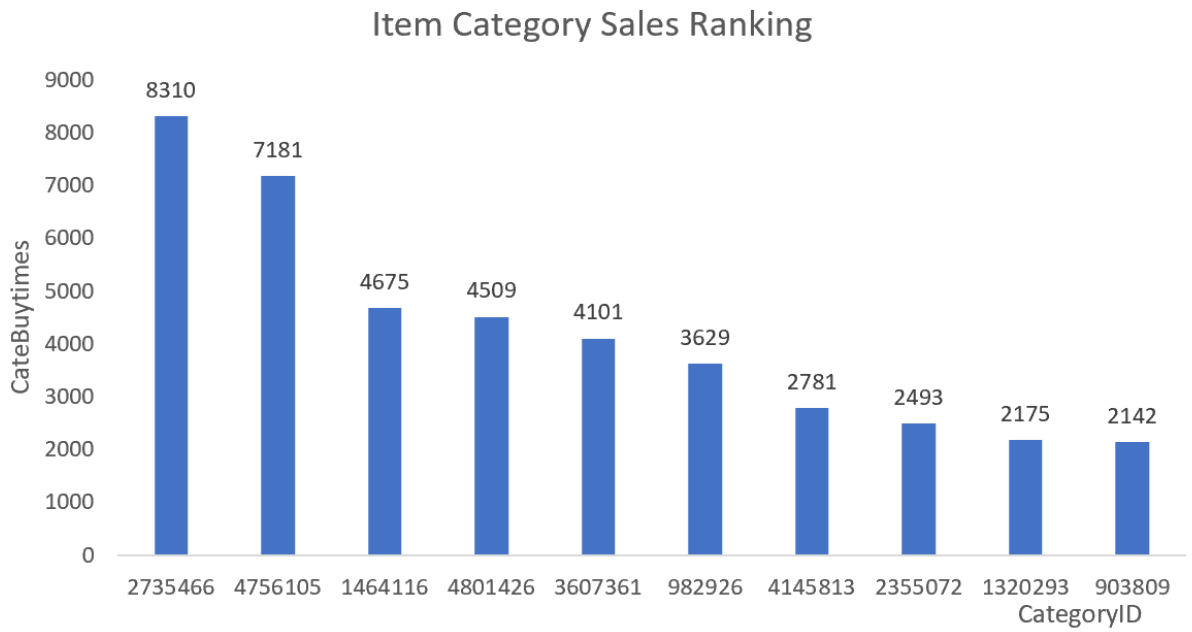
UserID	ItemID	CategoryID
879265	1203012	4163659
518295	1493764	4159072
815988	3330337	4756105
317490	3330337	4756105
877663	3330337	4756105
932142	3330337	4756105
331107	837113	245312
905497	580562	2355072
618133	2871705	2735466
4456	3000506	4643350

In [40]:
```
%%sql
SELECT CategoryID,COUNT(UserID) AS CateBuytimes
FROM (SELECT UserID,User_purchase_history.ItemID,CategoryID
FROM User_purchase_history
LEFT JOIN Item_category
ON User_purchase_history.ItemID=Item_category.ItemID) a
GROUP BY CategoryID
ORDER BY CateBuytimes DESC
LIMIT 10;
```

```
 * mysql+pymysql://root:***@fe512_mysql/fe512db
10 rows affected.
```

Out[40]:

CategoryID	CateBuytimes
2735466	8310
4756105	7181
1464116	4675
4801426	4509
3607361	4101
982926	3629
4145813	2781
2355072	2493
1320293	2175
903809	2142



- ***The item category whose ID is '2735466' is the most popular.***

6.Conclusion and Recommendation

1) Acquisition

December 2 and 3, 2017 is weekend, and the number of clicks increased steeply, it is possible that Taobao held a promotional event at that time. At different times of the day, clicks rose steadily from 10 o'clock to reach their peak by 13 o'clock, then gradually decreased, and at 20 o'clock began to rise, with 24 o'clock to reach the second peak.

Interestingly, people are keen on shopping before bed. So, if a shopping platform plans to carry out activities, the best period is noon or late at night, such as holding a special discount at lunch time or holding a snap at midnight.

2) Activation

User behavior includes clicking, adding item in the shopping cart, favoring, and buying. While 'pv' for 81.9% of total behavior, 'cart' for 9.6%, 'fav' but not buying 4.9%, and finally actually 'buy' down to 3%.

So, we could conclude that some items have been successful to arouse the interest of users, but for some reason the user hesitated on shopping, so that potential 'buy' users diverted to the 'fav'. According to the data analysis results, the suggestions to improve the conversion rate are that:

- **optimizing the screening function of the e-commerce platform, increasing the accuracy of keywords, making it easier for users to find the right item;**
- **providing customers with similar item comparison functions, so that users do not need to return multiple times search results;**
- **streamline the next single step and provide a one-click order service, such as including only clicks-buy-pay three link, shorten the purchase process, improve the user experience.**

3) Retention

Keeping users in the habit of using specified e-commerce platforms is the key to increasing retention rates, and the options available are:

- **Daily Online check-in points, daily "tasks", including adding items to shopping cart, adding favorite items and shopping, continuous check-in or completion of tasks a week, a month can automatically collect points, to the middle or end of the year can be exchanged for shopping vouchers;**
- **introduce VIP service to customers whose annual purchase quantity and amount reach the specified number. Get a 95% discount when they buy, and a higher-level discount one year after they buy. These methods can improve the retention rate of high-value users and cultivate their loyalty to the platform.**

4) Revenue

We can determine the valued users through the repurchase rate; through analysis to find out the valued users ' purchase preferences, items and item categories to develop personalized item recommendations ("Guess you Like"), so as to improve the user experience and e-commerce platform sales.

Possible appropriate improvement options are:

- **for the previously identified valued users to provide personalized product recommendations, such as the most concerned about the product categories and types, after the new regular push to the user;**
- **for the repurchase rate, can be launched within 3 months of the repurchase preferential activities, so that customers maintain the frequency of purchase.**

7.Shortage and Future Work

- **Since the time span is too short, just 1 month, we could not analyze the user behavior in a long term.**
- **Additionally, there is no data about monetary value, so we cannot divide users in groups accurately.**
- **As we mentioned, the dataset is quite large, so we cannot analyze based on the initial whole dataset.**

  So, the future work is to analyze user behavior in several months even 1 year, to find order amount to improve RFM model and to analyze the whole dataset on AWS or other big data platforms.

8. References

* https://www.jianshu.com/p/072e5b981040
* https://blog.treasuredata.com/blog/2016/07/22/rolling-retention-done-right-in-sql/
* https://zhuanlan.zhihu.com/p/59091803