

In [2]: # This Python 3 environment comes with many helpful analytics libraries installed  
 # It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker  
 # For example, here's several helpful packages to load

```
import numpy as np # Linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets pr
# You can also write temporary files to /kaggle/temp/, but they won't be saved outs
```

/kaggle/input/e-commerce-events-history-in-cosmetics-shop/2019-Nov.csv  
 /kaggle/input/e-commerce-events-history-in-cosmetics-shop/2020-Feb.csv  
 /kaggle/input/e-commerce-events-history-in-cosmetics-shop/2019-Oct.csv  
 /kaggle/input/e-commerce-events-history-in-cosmetics-shop/2019-Dec.csv  
 /kaggle/input/e-commerce-events-history-in-cosmetics-shop/2020-Jan.csv

目标是观察2019.10-2020.02之间，消费者在这个电商平台上的消费数据和用户消费行为数据。根据这个数据，如果想要提高商户的收益，应该指定怎样的营销策略？

关于商户的收益是我们会比较关心——对应的是GMV的变化  
 对于GMV的变化我们可以将其大体分为两个部分：

- 外部因素
- 内部因素（通过这里的数据，主要分析的是内部因素）
  - 可以根据流量，转化率，客单价将GMV进行分解：
 
$$\text{GMV} = \text{UV} \times \text{转化率} \times \text{客单价} \times \text{每个客户的平均订单数量}$$
  - PV指的是页面浏览的数量
  - UV指的是用户访问数量，就是一个用户最多能够计入一次UV
  - 转化率 = 购买的人数/总用户人数
  - 客单价 = 总的销售额/订单总数
  - 每个客户的平均订单数量 = 订单数量/购买用户
- 用户行为分析
  - 漏斗分析，看用户在哪个阶段流失了，转化率不高
- 用户分群分析
  - 将用户按照RFM进行分类，看各个群的用户有什么区别

我们需要计算的数据

- pv&uv的每日数据
- 每日的客单价
- 用户的行为数据（漏斗） view -> cart -> purchase 中间有一个移除购物车不知道怎么分析
- 复购率 同一个人重复购买的用户所占比率
- RFM模型进行用户分群？ Recency 最近一次消费 Frequency 消费频率 Money 消费金额
- 每天的销售额之类的
- 还有什么其他的分析想法？



```
In [6]: df2.shape
```

```
Out[6]: (4156682, 9)
```

```
In [7]: df3 = pd.read_csv("/kaggle/input/ecommerce-events-history-in-cosmetics-shop/2019-Oct.csv")
df3.shape
```

```
Out[7]: (4102283, 9)
```

```
In [8]: df4 = pd.read_csv("/kaggle/input/ecommerce-events-history-in-cosmetics-shop/2019-Dec.csv")
df4.shape
```

```
Out[8]: (3533286, 9)
```

```
In [9]: df5 = pd.read_csv("/kaggle/input/ecommerce-events-history-in-cosmetics-shop/2020-Jan.csv")
df5.shape
```

```
Out[9]: (4264752, 9)
```

```
In [10]: df = pd.concat([df1, df2, df3, df4, df5], axis = 0)
df.shape
```

```
Out[10]: (20692840, 9)
```

```
In [11]: df.dtypes
```

```
Out[11]: event_time      object
event_type       object
product_id      int64
category_id     int64
category_code   object
brand           object
price           float64
user_id         int64
user_session    object
dtype: object
```

```
In [12]: df['product_id'] = df['product_id'].astype(str)
df['category_id'] = df['category_id'].astype(str)
df['user_id'] = df['user_id'].astype(str)
df.dtypes
```

```
Out[12]: event_time      object
event_type       object
product_id      object
category_id     object
category_code   object
brand           object
price           float64
user_id         object
user_session    object
dtype: object
```

```
In [13]: df.describe()
```

Out[13]:

	price
<b>count</b>	2.069284e+07
<b>mean</b>	8.534735e+00
<b>std</b>	1.938142e+01
<b>min</b>	-7.937000e+01
<b>25%</b>	2.060000e+00
<b>50%</b>	4.050000e+00
<b>75%</b>	7.060000e+00
<b>max</b>	3.277800e+02

In [14]:

`df.isnull().sum()`

Out[14]:

event_time	0
event_type	0
product_id	0
category_id	0
category_code	20339246
brand	8757117
price	0
user_id	0
user_session	4598
dtype:	int64

In [15]:

```
# 价格最小值为负数明显不对，将其转化为0
df['price'] = df['price'].apply(lambda x: 0 if x < 0 else x)
df['price'].min()
```

Out[15]:

0.0

In [16]:

```
# 将发生时间的UTC字符串给去掉
def get_datetime(data):
    datetime = ' '.join((data.split(' ')[0], data.split(' ')[1]))
    return datetime
df['event_time'] = df.event_time.map(get_datetime)
df.head()
```



In [18]:

```
# 对PV按照天和月进行分析
total_pv = df.shape[0]
# 其实好像也不用对year_month进行分组
daily_pv = df[['user_id', 'date']].groupby(['date']).count()
```

In [19]:

```
# 对UV按照天和月进行分析
df_user = df[['user_id', 'date']].drop_duplicates(subset=['user_id', 'date'])
total_uv = df_user.shape[0]
daily_uv = df_user.groupby(['date']).count()
df_user2 = df[['user_id', 'year_month']].drop_duplicates(subset=['user_id', 'year_month'])
month_uv = df_user2.groupby('year_month').count()
```

In [20]:

```
month_uv.rename(columns={'user_id': 'month_uv'}, inplace=True)
month_uv.head()
```

Out[20]:

year_month	month_uv
2019-10	399664
2019-11	368232
2019-12	370154
2020-01	410073
2020-02	391055

In [21]:

```
pv_uv = daily_pv.merge(daily_uv, how='inner', on = 'date').reset_index()
pv_uv.rename(columns={'user_id_x':'pv', 'user_id_y':'uv'}, inplace=True)
# x 标识pv y标识uv
pv_uv.head()
```

Out[21]:

	date	pv	uv
0	2019-10-01	142414	19230
1	2019-10-02	201068	33859
2	2019-10-03	124847	16323
3	2019-10-04	115612	14732
4	2019-10-05	106343	14990

## 客单价 = 总销售额/订单数量

In [22]:

```
# 计算总销售额
total_price = df[df['event_type'] == 'purchase'][['date', 'price']].groupby('date')
# 计算订单数量
order_acount = df[df['event_type'] == 'purchase'][['date', 'price']].groupby('date')
```

In [23]:

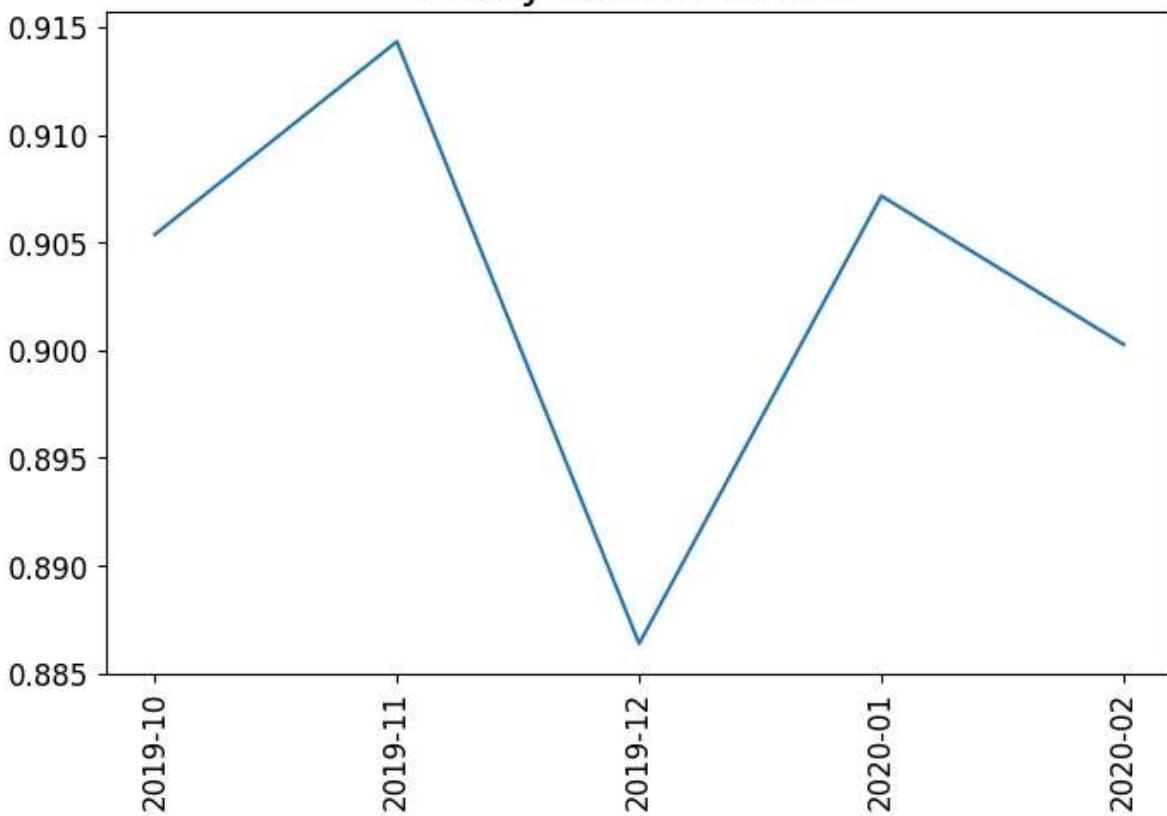
```
data = pv_uv.merge(total_price, how='inner', on='date')
data = data.merge(order_acount, how='inner', on='date')
data.rename(columns={'price_x': 'total_price', 'price_y': 'order_acount'}, inplace=True)
data.head()
```







## rebuy rate of month



从首次购买和复购的趋势图，可以看出

- 购买人数最高的时间出现在11月29日附近，可以预测那个时候是黑色星期五，电商网站上营销活动较多，但是用户忠诚度不高，很多客户都在购买一次之后流失。
- 在12月左右的时候，开始过圣诞节，消费金额可能更多的倾向于其他产品，比如食品、礼物等，从而导致客户的化妆品上的消费能力下降。
- 且观察购买行为呈现出一定的周期性变化趋势，可能是因为化妆品是消耗品，在一段时间用完之后才会有购买的可能性

转化率部分（结合漏斗分析）

用户分群（RFM模型）

```
In [51]: # 构建RFM模型
data_purchase = df[df.event_type == 'purchase']
# 获取R/F/M
rfm = data_purchase.pivot_table(index = 'user_id',
                                 values = {'date', 'user_session', 'price'},
                                 aggfunc = {'date':'max',
                                            'user_session':'count',
                                            'price':'sum'})
rfm.head()#最近一次消费时间, 消费次数, 消费总金额
```





```
month12_data = buy_change_data[buy_change_data['year_month'] == '2019-12'].iloc[:10]
month1_data = buy_change_data[buy_change_data['year_month'] == '2020-01'].iloc[:10]
month2_data = buy_change_data[buy_change_data['year_month'] == '2020-02'].iloc[:10]
month2_data.head()
```

Out[61]:

	<b>year_month</b>	<b>product_id</b>	<b>sales</b>
<b>109435</b>	2020-02	5810480	3719.10
<b>109434</b>	2020-02	5792800	4365.36
<b>109433</b>	2020-02	5877453	5048.97
<b>109432</b>	2020-02	5751383	5052.02
<b>109431</b>	2020-02	5849033	5273.52

这里每个月销量前十的表要不要输出，画一个图？

tableau仪表盘建议分为两个部分

用户部分：

- pv uv随时间变化的趋势
- 客户消费金额时间变化

可以粗略判断消费金额可能是由营销活动黑五产生的

同时对比其他历史同期数据（但这里并没有其他历史的同期数据）

此外，可以对比uv pv在那段时间是否上涨，客单价是否上升来佐证营销活动的效果

- 首次购买和复购的时间，计算一下月的复购率（python直接计算）

产品部分：

- 销售额
- 每月销售前十榜单

In [ ]: