Аналитика продукта

Домашнее задание 2.

Выполнил: Абраменко Александр

Оглавление:

Задание 0. Обработка данных

Я работаю в Jupyter NoteBook на языке программирования Python.

Подключим необходимые библиотеки для работы с базой данных:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Далее считаем датафрейм и изучим структуры базы:

```
df = pd.read_csv("online_retail.xlsx - online_retail.csv")
```

```
df
```

	Invoice	Description	Quantity	InvoiceDate	Price	Customer ID	Amount
0	493410	This is a test product.	5	2010-01-04 9:24:00	4,5	12346	22,5
1	493412	This is a test product.	5	2010-01-04 9:53:00	4,5	12346	22,5
2	493414	RETRO SPOT MUG	36	2010-01-04 10:28:00	2,55	14590	91,8
3	493414	RETRO SPOT LARGE MILK JUG	12	2010-01-04 10:28:00	4,25	14590	51
4	493414	NEW ENGLAND CERAMIC CAKE SERVER	2	2010-01-04 10:28:00	2,55	14590	5,1
...	...	...	...	...	...	...	...
164517	514212	BAG 250g SWIRLY MARBLES	2	2010-06-30 17:04:00	0,85	14882	1,7
164518	514212	BLUE/CREAM STRIPE FRINGE HAMMOCK	1	2010-06-30 17:04:00	7,95	14882	7,95
164519	514212	OCEAN STRIPE HAMMOCK	1	2010-06-30 17:04:00	7,95	14882	7,95
164520	514212	RED/CREAM STRIPE FRINGE HAMMOCK	1	2010-06-30 17:04:00	7,95	14882	7,95
164521	514212	UNION STRIPE WITH FRINGE HAMMOCK	1	2010-06-30 17:04:00	7,95	14882	7,95

164522 rows × 7 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 164522 entries, 0 to 164521
Data columns (total 7 columns):
 #   Column       Non-Null Count   Dtype 
---  ------       --------------   ----- 
 0   Invoice      164522 non-null  int64 
 1   Description  164522 non-null  object
 2   Quantity     164522 non-null  int64 
 3   InvoiceDate  164522 non-null  object
 4   Price        164522 non-null  object
 5   Customer ID  164522 non-null  int64 
 6   Amount       164522 non-null  object
dtypes: int64(3), object(4)
memory usage: 8.8+ MB
```

```
df.shape
```

```
(164522, 7)
```

Из недочётов сразу видно, что необходимо изменить названия колонок, а также в колонках “price” и “amount” изменить тип данных на вещественный, также в колонке “invoice_date” сменить на тип дата. Конечно же избавимся от дубликатов и проверим датафрейм на пропуски.

```
df = df.drop_duplicates()
df.shape
```

```
(161990, 7)
```

```
df.isnull().sum()
```

```
Invoice        0
Description    0
Quantity       0
InvoiceDate    0
Price          0
Customer ID    0
Amount         0
dtype: int64
```

```
df.rename(columns={'Invoice': 'invoice',
                   'Description': 'description',
                   'Quantity':'quantity',
                   'InvoiceDate':'invoice_date',
                   'Price':'price',
                   'Customer ID':'customer_id',
                   'Amount':'amount'}, inplace=True)
```

```
df['price'] = df['price'].str.replace(',', '.').astype(float)
df['amount'] = df['amount'].str.replace(',', '.').astype(float)
```

```
df['invoice_date'] = pd.to_datetime(df['invoice_date'])
```

```
df.head(5)
```

	invoice	description	quantity	invoice_date	price	customer_id	amount
0	493410	This is a test product.	5	2010-01-04 09:24:00	4.50	12346	22.5
1	493412	This is a test product.	5	2010-01-04 09:53:00	4.50	12346	22.5
2	493414	RETRO SPOT MUG	36	2010-01-04 10:28:00	2.55	14590	91.8
3	493414	RETRO SPOT LARGE MILK JUG	12	2010-01-04 10:28:00	4.25	14590	51.0
4	493414	NEW ENGLAND CERAMIC CAKE SERVER	2	2010-01-04 10:28:00	2.55	14590	5.1

Задание 1. Как изменилось количество уникальных пользователей за полгода?

Добавим вспомогательные колонки с месяцем и неделей заказа:

```
df['week'] = df['invoice_date'].dt.to_period('W')
df['month'] = df['invoice_date'].dt.to_period('M')
```

```
df.head(5)
```

	invoice	description	quantity	invoice_date	price	customer_id	amount	week	month
0	493410	This is a test product.	5	2010-01-04 09:24:00	4.50	12346	22.5	2010-01-04/2010-01-10	2010-01
1	493412	This is a test product.	5	2010-01-04 09:53:00	4.50	12346	22.5	2010-01-04/2010-01-10	2010-01
2	493414	RETRO SPOT MUG	36	2010-01-04 10:28:00	2.55	14590	91.8	2010-01-04/2010-01-10	2010-01
3	493414	RETRO SPOT LARGE MILK JUG	12	2010-01-04 10:28:00	4.25	14590	51.0	2010-01-04/2010-01-10	2010-01
4	493414	NEW ENGLAND CERAMIC CAKE SERVER	2	2010-01-04 10:28:00	2.55	14590	5.1	2010-01-04/2010-01-10	2010-01

Вынесем в отдельные датафреймы количество уникальных пользователей по неделям и месяцам:

```
wau_data = df.groupby('week')['customer_id'].nunique()
mau_data = df.groupby('month')['customer_id'].nunique()
```

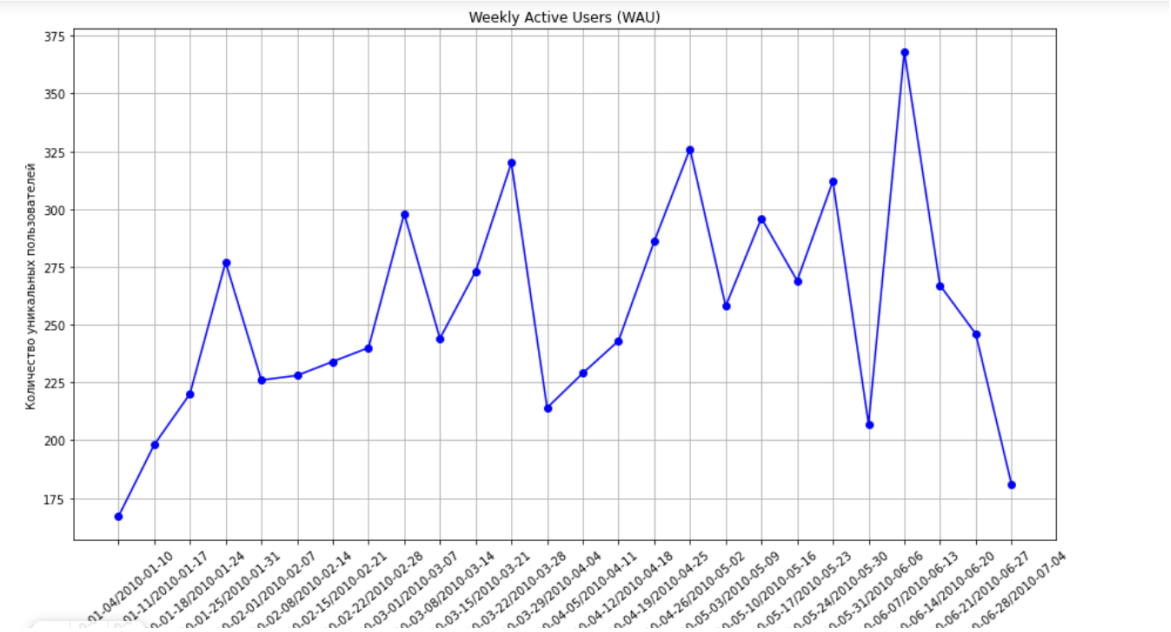
```
wau_data
```

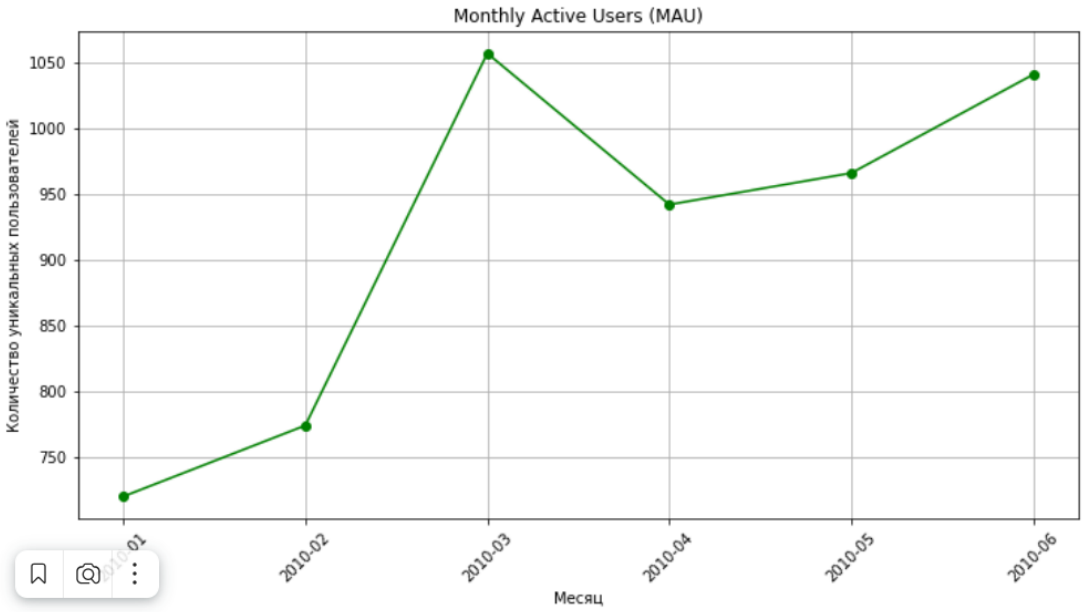
```
week
2010-01-04/2010-01-10    167
2010-01-11/2010-01-17    198
2010-01-18/2010-01-24    220
2010-01-25/2010-01-31    277
2010-02-01/2010-02-07    226
2010-02-08/2010-02-14    228
2010-02-15/2010-02-21    234
2010-02-22/2010-02-28    240
2010-03-01/2010-03-07    298
2010-03-08/2010-03-14    244
2010-03-15/2010-03-21    273
2010-03-22/2010-03-28    320
2010-03-29/2010-04-04    214
2010-04-05/2010-04-11    229
2010-04-12/2010-04-18    243
2010-04-19/2010-04-25    286
2010-04-26/2010-05-02    326
2010-05-03/2010-05-09    258
2010-05-10/2010-05-16    296
2010-05-17/2010-05-23    269
2010-05-24/2010-05-30    312
2010-05-31/2010-06-06    207
2010-06-07/2010-06-13    368
2010-06-14/2010-06-20    267
2010-06-21/2010-06-27    246
2010-06-28/2010-07-04    181
Freq: W-SUN, Name: customer_id, dtype: int64
```

```
mau_data
```

```
month
2010-01     720
2010-02     774
2010-03    1057
2010-04     942
2010-05     966
2010-06    1041
Freq: M, Name: customer_id, dtype: int64
```

А также взглянем на графики изменения количества уникальных пользователей:





Как можно заметить, исходя из статистики по месяцам, количество пользователей растёт, основной скачок и максимум за полгода пришёлся на март месяц, а к июню количество уникальных пользователей выросло на 27,7% по сравнению с январём.

Задание 2. На сколько процентов изменился средний чек во втором квартале по сравнению с первым?

В датафрейме result_df посчитаем общие стоимости заказов (стоимости чеков):

```
total_receipts = df.groupby('invoice')['amount'].sum().reset_index()
df = pd.merge(df, total_receipts, on='invoice', how='left', suffixes=('', '_total'))
```

```
result_df = df[['invoice', 'month', 'amount_total']].drop_duplicates()
result_df
```

	invoice	month	amount_total
0	493410	2010-01	22.50
1	493412	2010-01	22.50
2	493414	2010-01	290.20
8	493427	2010-01	264.38
26	493428	2010-01	230.90
...	...	...	...
161925	514208	2010-06	310.41
161945	514209	2010-06	192.30
161950	514210	2010-06	287.98
161969	514211	2010-06	224.04
161979	514212	2010-06	135.90

7844 rows × 3 columns

После чего разделим получившийся датафрейм на кварталы, для каждого посчитаем средний чек и выведем разницу в процентах:

```
first_quarter = result_df[result_df['month'].dt.quarter == 1]
second_quarter = result_df[result_df['month'].dt.quarter == 2]
average_first_quarter = first_quarter['amount_total'].mean()
average_second_quarter = second_quarter['amount_total'].mean()
print(f'{average_second_quarter/average_first_quarter*100-100}%')
```

```
-9.968133168828501%
```

Как итог, мы выяснили, что во втором квартале средний чек уменьшился на 9,97%. Что конечно же печально:(

Задание 3. Объясните с чем может быть связано падение ARPU?

Для данного задания я решил считать ARPU по месяцам, следовательно вычислим по месяцам значения дохода и количества покупателей, а также показатель ARPU:

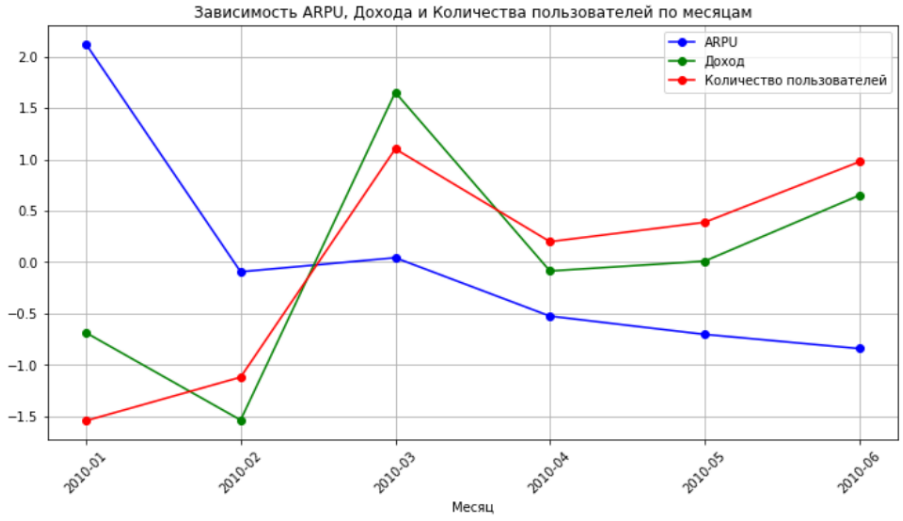
```
monthly_data = df.groupby('month').agg({'amount': 'sum', 'customer_id': 'nunique'}).reset_index()
monthly_data['ARPU'] = monthly_data['amount'] / monthly_data['customer_id']
monthly_data
```

	month	amount	customer_id	ARPU
0	2010-01	555751.192	720	771.876656
1	2010-02	504479.656	774	651.782501
2	2010-03	696770.571	1057	659.196377
3	2010-04	591948.252	942	628.395172
4	2010-05	597732.130	966	618.770321
5	2010-06	636346.130	1041	611.283506

Как мы видим получившиеся числовые признаки не коррелируют, поэтому сперва отнормируем данный датасет и построим ломанные зависимости по месяцам:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
monthly_data[['amount','customer_id','ARPU']] = scaler.fit_transform(monthly_data[['amount','customer_id','ARPU']])
monthly_data
```

	month	amount	customer_id	ARPU
0	2010-01	-0.687325	-1.544688	2.123290
1	2010-02	-1.538124	-1.120553	-0.094199
2	2010-03	1.652748	1.102226	0.042695
3	2010-04	-0.086672	0.198977	-0.526036
4	2010-05	0.009306	0.387481	-0.703755
5	2010-06	0.650066	0.976557	-0.841996



Выводы: как видно из графиков ARPU плавно уменьшается, сильный выброс случился в феврале из-за того, что доход уменьшился, а количество покупателей увеличилось, следующий месяц наблюдается незначительный рост в связи с тем, что оба параметра увеличилось и доход был больше количества пользователей, но далее картина печальная, так как доход перманентно меньше количества людей, следовательно для решения данной проблемы необходимо увеличить доход, как одно из решений гипотезы, увеличение цен на товары.

Задание 4. Как изменилось количество заказов на покупателя?

Посчитаем в первую очередь количество заказов на покупателя по месяцам, затем среднее значение заказов на покупателя в каждом из месяец, а также отобразим в виде графика зависимости:

```
In [25]:
orders_per_customer = df.groupby(['month', 'customer_id'])['invoice'].nunique().reset_index()
average_orders_per_customer = orders_per_customer.groupby('month')['invoice'].mean().reset_index()
average_orders_per_customer
```

Out[25]:

	month	invoice
0	2010-01	1.404167
1	2010-02	1.428941
2	2010-03	1.441816
3	2010-04	1.410828
4	2010-05	1.425466
5	2010-06	1.438040

Как видно из графика количество заказов на пользователя растёт, за исключением сильной просадки в апреле и мае, так как мы не обладаем всей базой данных за прошлые года, можно выдвинуть слабые гипотезы насчёт того, что это возможно сезонная проблема или были проблемы с поставкой товаров или с системой магазина в целом, например если это проблема заключается в сезоне, то достаточно было сравнить данные показатели за прошлые года.

Задание 5. На сколько процентов изменилось количество товаров в заказе в июне по сравнению с мартом?

Сперва посчитаем общее количество товаров в каждом заказе:

```
total_number_of_products = df.groupby('invoice')['quantity'].sum()
df = pd.merge(df, total_number_of_products, on='invoice', how='left', suffixes=('', '_total'))
df
```

	invoice	description	quantity	invoice_date	price	customer_id	amount	week	month	amount_total	quantity_total
0	493410	This is a test product.	5	2010-01-04 09:24:00	4.50	12346	22.50	2010-01-04/2010-01-10	2010-01	22.5	5
1	493412	This is a test product.	5	2010-01-04 09:53:00	4.50	12346	22.50	2010-01-04/2010-01-10	2010-01	22.5	5
2	493414	RETRO SPOT MUG	36	2010-01-04 10:28:00	2.55	14590	91.80	2010-01-04/2010-01-10	2010-01	290.2	88
3	493414	RETRO SPOT LARGE MILK JUG	12	2010-01-04 10:28:00	4.25	14590	51.00	2010-01-04/2010-01-10	2010-01	290.2	88
4	493414	NEW ENGLAND CERAMIC CAKE SERVER	2	2010-01-04 10:28:00	2.55	14590	5.10	2010-01-04/2010-01-10	2010-01	290.2	88
...	...	...	...	...	...	...	...	...	...	...	...
161985	514212	BAG 250g SWIRLY MARBLES	2	2010-06-30 17:04:00	0.85	14882	1.70	2010-06-28/2010-07-04	2010-06	135.9	62
161986	514212	BLUE/CREAM STRIPE FRINGE HAMMOCK	1	2010-06-30 17:04:00	7.95	14882	7.95	2010-06-28/2010-07-04	2010-06	135.9	62
161987	514212	OCEAN STRIPE HAMMOCK	1	2010-06-30 17:04:00	7.95	14882	7.95	2010-06-28/2010-07-04	2010-06	135.9	62
161988	514212	RED/CREAM STRIPE FRINGE HAMMOCK	1	2010-06-30 17:04:00	7.95	14882	7.95	2010-06-28/2010-07-04	2010-06	135.9	62
161989	514212	UNION STRIPE WITH FRINGE HAMMOCK	1	2010-06-30 17:04:00	7.95	14882	7.95	2010-06-28/2010-07-04	2010-06	135.9	62

Далее разделим датасет на два периода: май и июнь. После чего посчитаем в каждом среднее количество товаров в заказе и выведем разницу в процентах:

```
selected_months = df[df['month'].isin([pd.Period('2010-03'), pd.Period('2010-06')])]
selected_months1 = selected_months[['invoice','month','quantity_total']].drop_duplicates()
selected_months1
average_quantity_per_order = selected_months1.groupby('month')['quantity_total'].mean()
average_quantity_per_order
```

```
month
2010-03    329.373360
2010-06    260.427522
Freq: M, Name: quantity_total, dtype: float64
```

```
print(f"{average_quantity_per_order['2010-06']/average_quantity_per_order['2010-03']*100-100}%")
```

```
-20.932426944872176%
```

Как мы видим, среднее количество товаров в заказе в июне уменьшилось на 20,93% по сравнению с маем, из чего можно сделать пару гипотез, что возможно в мае покупали больше расходных товаров, из-за чего количество было выше, нежели в июне или также может быть связано с тем, что в июне больше людей уходят в отпуск и не нуждались в приобретении конкретных товаров, так что можно сказать, что это не так плохо, играет роль сам бизнес и товары, что предоставляет.

Задание 6. Предложите 1 - 2 идеи, как увеличить GMV в 3 квартале в 1,5 раза.

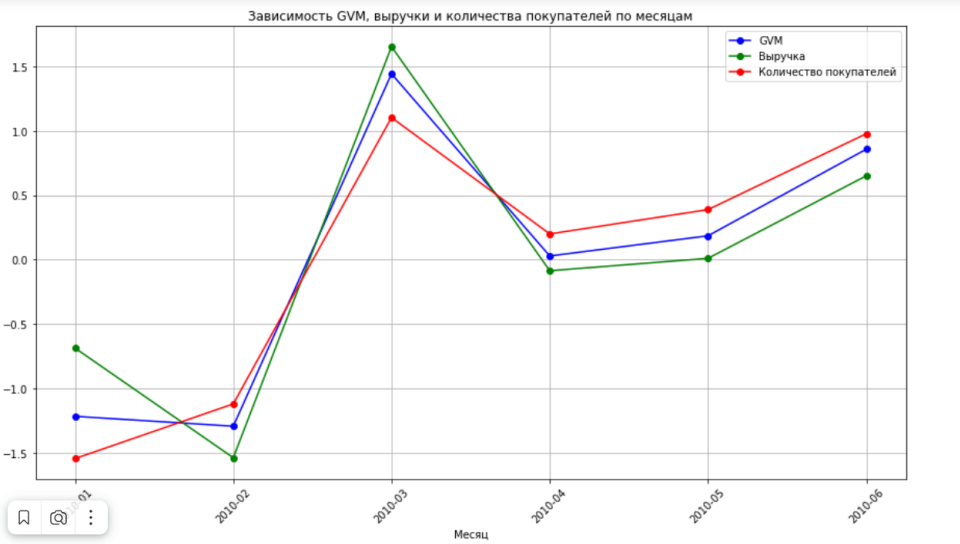
Метрика GVM - демонстрирует характеристику прибыльности бизнеса, соответственно $GVM = N \times (ARPU - CAC)$, так как мы не располагаем информацией о стоимости привлечения покупателей, будем считать её за единицу. Соответственно посчитаем выручку и количество покупателей, а также показатель GVM. После отнормируем данные для построения графиков зависимостей.

```
GVM_stat = df.groupby('month').agg({'amount': 'sum', 'customer_id': 'nunique'}).reset_index()
GVM_stat['GVM'] = GVM_stat['amount']*GVM_stat['customer_id']
GVM_stat
```

	month	amount	customer_id	GVM
0	2010-01	555751.192	720	4.001409e+08
1	2010-02	504479.656	774	3.904673e+08
2	2010-03	696770.571	1057	7.364865e+08
3	2010-04	591948.252	942	5.576153e+08
4	2010-05	597732.130	966	5.774092e+08
5	2010-06	636346.130	1041	6.624363e+08

```
GVM_stat[['amount','customer_id','GVM']] = scaler.fit_transform(GVM_stat[['amount','customer_id','GVM']])
GVM_stat
```

	month	amount	customer_id	GVM
0	2010-01	-0.687325	-1.544688	-1.216865
1	2010-02	-1.538124	-1.120553	-1.293327
2	2010-03	1.652748	1.102226	1.441678
3	2010-04	-0.086672	0.198977	0.027844
4	2010-05	0.009306	0.387481	0.184300
5	2010-06	0.650066	0.976557	0.856370



Как видно наблюдается тенденция роста GVM и видим приятный выброс в марте месяце, чтобы наверняка выяснить хороший ли результат в марте, нужно убедиться, что в этот период все данные корректно “*трэкались*” и не было никаких сбоев системы, чтобы убедиться в правдивости данных показателей.

Теперь обсудим, как можно увеличить в 1,5 раза исследуемый показатель, так как это линейная формула у нас два варианта: увеличить количество покупателей или выручку. Если с первым мало, что известно, так как мы не знаем из данных способы привлечения и какие суммы туда затрачены, сперва обсудим выручку.

Выручку можно увеличить двумя способами, увеличением цены товаров, но что может повлечь за собой упадок спроса на товары или наоборот ввести скидки на товары, чтобы увеличить спрос (то есть покупателей) с целью увеличения общей выручки, но в таком случае надо следить за прибылью.

Лично я вижу две стратегии, если товары которые покупают чаще всего являются товарами первой необходимости(неэластичными), то можно смело увеличивать на них цену, ведь они будут также актуальны среди покупателей и они будут вынуждены покупать его за эту цену, только важный момент, это оценить своих конкурентов в данной сфере, чтобы наши покупатели не стали приобретать товар у конкурента за меньшую цену. Взглянем на самые популярные товары:

```
In [39]: popularity_products = df.groupby(['month','description'])['quantity'].sum().sort_values(ascending=False)
         popularity_products.head(10)

Out[39]: month    description
         2010-02  BLACK AND WHITE PAISLEY FLOWER MUG    19248
         2010-03  SET/6 WOODLAND PAPER PLATES           13099
                  SET/6 WOODLAND PAPER CUPS             13062
                  SET/6 STRAWBERRY PAPER CUPS           13009
                  SET/6 STRAWBERRY PAPER PLATES         12504
         2010-02  SMALL FAIRY CAKE FRIDGE MAGNETS       11960
         2010-03  PACK OF 12 SUKI TISSUES               11800
                  PACK OF 12 WOODLAND TISSUES           11562
                  PACK OF 12 PINK PAISLEY TISSUES       11334
                  PACK OF 12 RED SPOTTY TISSUES         10792
         Name: quantity, dtype: int64
```

Вторая стратегия будет заключаться в том, чтобы увеличить цену на товары, которые меньше всего приносят вклад в общую выручку, так как эти товары явно покупают редко, следовательно будет актуальным увеличить на данные позиции цену. Взглянем на данные товары:

```
profitability_of_goods = df.groupby(['month','description'])['amount'].sum().sort_values()
profitability_of_goods.head(10)
```

```
month    description
2010-02  This is a test product.             0.000
2010-03  PADS TO MATCH ALL CUSHIONS           0.001
2010-04  PADS TO MATCH ALL CUSHIONS           0.002
2010-01  PADS TO MATCH ALL CUSHIONS           0.002
2010-02  PADS TO MATCH ALL CUSHIONS           0.006
2010-01  DISCO BALL CHRISTMAS DECORATION      0.120
2010-02  PINK FLUFFY CHRISTMAS DECORATION     0.190
2010-05  CHAMPAGNE TRAY BLANK CARD            0.190
         GLOW IN DARK DOLPHINS                0.210
2010-04  SCULPTED ROUND IVORY CANDLE          0.210
Name: amount, dtype: float64
```

Как итог, хочется сказать, что это всего лишь гипотезы, которые также нужно обсуждать с продукт-менеджером в своей команде и обязательно проверять AB-тестами, ведь если товары эластичные, то увеличения их цены могу понести сильные потери в величине спроса, если же иначе, то смело можно увеличивать цену, тем самым улучшая GVM. Или же уменьшать цены с целью увеличения спроса на товары, которые продаются в очень малых количествах.