RuoxinWang-EcommerceProject

August 10, 2023

[1]:
```
# AIPin E-Commerce Project
# Eidtor: Ruoxin Wang
# 08/06/2023
```

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

import datetime as dt
import seaborn as sns
sns.set(color_codes=True)
pd.set_option('display.max_columns', None)

import warnings
warnings.filterwarnings("ignore")

import statsmodels.api as sm
from scipy import stats
from scipy.stats.mstats import zscore

import plotly.express as px
from sklearn.datasets import make_swiss_roll
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
```

1 Part 1: Import Dataset

[3]:
```
# Customers Dataset
customers = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/customers.
↪csv')
customers.info()
customers['id']=customers['id'].astype(object)
customers.info()
customers.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44661 entries, 0 to 44660
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   id          44661 non-null  int64
 1   full_name   33699 non-null  object
 2   created_at  44661 non-null  object
dtypes: int64(1), object(2)
memory usage: 1.0+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44661 entries, 0 to 44660
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   id          44661 non-null  object
 1   full_name   33699 non-null  object
 2   created_at  44661 non-null  object
dtypes: object(3)
memory usage: 1.0+ MB
```

[3]:

	id	full_name	created_at
0	8652230815	NaN	2016-08-16
1	8686141151	Warren Perez	2016-08-22
2	8686909727	Micheal Robles	2016-08-22
3	8686915935	Michael Ellis	2016-08-22
4	8686918303	Robert Stewart	2016-08-22

full_name has null value

[4]:
```
# Orders_items Dataset
orders_items = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/
↪orders_items.csv')
orders_items.info()
orders_items['id']=orders_items['id'].astype(object)
orders_items['order_id']=orders_items['order_id'].astype(object)
orders_items['product_id']=orders_items['product_id'].astype(object)
orders_items['variant_id']=orders_items['variant_id'].astype(object)
orders_items.info()
orders_items.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36826 entries, 0 to 36825
Data columns (total 10 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   id                   36826 non-null  int64
 1   order_id             36826 non-null  int64
```

```
 2   product_id         36802 non-null  float64
 3   product_style      36826 non-null  object
 4   variant_id         36826 non-null  int64
 5   sku                36826 non-null  object
 6   product_title      36826 non-null  object
 7   fulfillment_status  35257 non-null  object
 8   price              36826 non-null  float64
 9   quantity           36826 non-null  int64
dtypes: float64(2), int64(4), object(4)
memory usage: 2.8+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36826 entries, 0 to 36825
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   id                  36826 non-null  object
 1   order_id            36826 non-null  object
 2   product_id          36802 non-null  object
 3   product_style       36826 non-null  object
 4   variant_id          36826 non-null  object
 5   sku                 36826 non-null  object
 6   product_title       36826 non-null  object
 7   fulfillment_status  35257 non-null  object
 8   price               36826 non-null  float64
 9   quantity            36826 non-null  int64
dtypes: float64(1), int64(1), object(8)
memory usage: 2.8+ MB
```

[4]:

```
            id    order_id     product_id                     product_style  \
0  13325125855  7675398239  12927629215.0  2c259a42d38f5f097274beff811168e2
1  13327045983  7676331935  12927632095.0  dd804c4025d230467823200aa82e9219
2  13327109727  7676363167  12928055775.0  f4e2e3c5433e4120889e2a7e0e0180a8
3  13327495903  7676539359  12927625695.0  08ba660ec5643520a73108bef6f3ddd6
4  13327518751  7676549855  12927690655.0  68ac90e5df73ae9b662174b21dc1586f

    variant_id                               sku  \
0  50547057311  000d96b3b77b33af530eec77689bd210
1  50547118303  e26c77e84b91c9939c23c3e3ef66475a
2  50553858975  0be0c8bf78ecf36416a40c9012acd19e
3  50547001887  0503dec809a8a2600d9acc5249900ecb
4  50548035807  38de0d087208588510907b5c2d149e4b

                      product_title fulfillment_status  price  quantity
0  5cfd6c4e00b25e6dec5538928206b7b8                NaN   35.0         1
1  0e6e45ad42707e9732119f4b98aec7ce                NaN   79.0         1
2  bede8c8f4e3c9c9d9a061d9a8d086cdc                NaN   58.0         1
3  27d598cb953eff3667f7d051fe795284          fulfilled   25.0         1
```

```
4  07dd8ba2ccadf3f3766750f10f6d05b5          fulfilled    25.0          1
```

[5]:
```python
orders_items = orders_items.rename(columns={'id':'orders_items_id'})
```

product_id, fulfillment_status have null value

[6]:
```python
# Orders Dataset
orders = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/orders.csv')
orders.info()
orders['id']=orders['id'].astype(object)
orders['customer_id']=orders['customer_id'].astype(object)
orders.info()
orders.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21358 entries, 0 to 21357
Data columns (total 13 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   id                     21358 non-null  int64
 1   created_at             21358 non-null  object
 2   closed_at              20195 non-null  object
 3   cancelled_at           410 non-null    object
 4   customer_id            21358 non-null  int64
 5   financial_status       21358 non-null  object
 6   fulfillment_status     20680 non-null  object
 7   processed_at           21358 non-null  object
 8   total_price            21358 non-null  float64
 9   shipping_rate          21358 non-null  float64
 10  subtotal_price         21358 non-null  float64
 11  total_discounts        21358 non-null  float64
 12  total_line_items_price 21358 non-null  float64
dtypes: float64(5), int64(2), object(6)
memory usage: 2.1+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21358 entries, 0 to 21357
Data columns (total 13 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   id                     21358 non-null  object
 1   created_at             21358 non-null  object
 2   closed_at              20195 non-null  object
 3   cancelled_at           410 non-null    object
 4   customer_id            21358 non-null  object
 5   financial_status       21358 non-null  object
 6   fulfillment_status     20680 non-null  object
 7   processed_at           21358 non-null  object
 8   total_price            21358 non-null  float64
```

```
 9   shipping_rate           21358 non-null  float64
 10  subtotal_price          21358 non-null  float64
 11  total_discounts         21358 non-null  float64
 12  total_line_items_price  21358 non-null  float64
dtypes: float64(5), object(8)
memory usage: 2.1+ MB
```

[6]:
```
           id  created_at   closed_at cancelled_at customer_id  \
0  7675398239  2016-08-21  2016-08-25   2016-08-22  8683754719
1  7676331935  2016-08-22  2016-08-22          NaN  8686224991
2  7676363167  2016-08-22         NaN   2016-08-22  8686224991
3  7676539359  2016-08-22  2016-08-22          NaN  8686915935
4  7676549855  2016-08-22  2016-08-22          NaN  8686924319

  financial_status fulfillment_status processed_at  total_price  \
0           voided                NaN   2016-08-21        44.57
1         refunded                NaN   2016-08-22       124.55
2           voided                NaN   2016-08-22        97.68
3             paid          fulfilled   2016-08-22       131.10
4             paid          fulfilled   2016-08-22        91.12

   shipping_rate  subtotal_price  total_discounts  total_line_items_price
0           6.33            35.0              0.0                    35.0
1           0.00           114.0              0.0                   114.0
2           7.00            83.0              0.0                    83.0
3           0.00           120.0              0.0                   120.0
4           7.00            77.0              0.0                    77.0
```

[7]:
```
# Fulfillment has 3 status and fulfilled should be the one that be focused on
orders.fulfillment_status.value_counts()
```

[7]:
```
fulfilled    20369
partial        309
restocked        2
Name: fulfillment_status, dtype: int64
```

[8]:
```
orders = orders.rename(columns={"id": "order_id", 'created_at':
 ↪'order_created_at','closed_at':'order_closed_at'})
```

closed_at, cancelled_at, fulfillment_status have null value

[9]:
```
# Products_skus Dataset
products_skus = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/
 ↪products_skus.csv')
products_skus.info()
products_skus['id']=products_skus['id'].astype(object)
products_skus['product_id']=products_skus['product_id'].astype(object)
products_skus.info()
```

```
products_skus.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1356 entries, 0 to 1355
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             1356 non-null   int64
 1   product_id     1356 non-null   int64
 2   product_style  1356 non-null   object
 3   sku            1356 non-null   object
 4   created_at     1356 non-null   object
 5   price          1356 non-null   float64
dtypes: float64(1), int64(2), object(3)
memory usage: 63.7+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1356 entries, 0 to 1355
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   id             1356 non-null   object
 1   product_id     1356 non-null   object
 2   product_style  1356 non-null   object
 3   sku            1356 non-null   object
 4   created_at     1356 non-null   object
 5   price          1356 non-null   float64
dtypes: float64(1), object(5)
memory usage: 63.7+ KB
```

[9]:

```
               id     product_id                     product_style  \
0     50547147871    12927633311  d510a563d66df17daf05e72a6af123b7
1     50547117727    12927632095  dd804c4025d230467823200aa82e9219
2   4886503364093   375446050301  8f61ed9720d09c9303fbc0b3184d478d
3     50547000415    12927625695  08ba660ec5643520a73108bef6f3ddd6
4     50547135135    12927632799  6056dc7fb0e6987bfb6d08a8a707446f

                                sku  created_at  price
0  0ecbe4277237cb1207b31815166d37b9  2016-08-18   29.0
1  f8e9bf1495c45676e8822e7ad4c97a93  2016-08-18   39.5
2  ccf2a80ad99d9dc449fbd5a904210d2c  2016-11-14   24.0
3  db1ea83c6299a2df5e39e420223fbd81  2016-08-18   25.0
4  c4f9cdb1df7a9add57df53e34290cbeb  2016-08-18   31.5
```

No null value

[10]:

```
# Products Dataset
products = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/products.csv')
products.info()
```

```
products['id']=products['id'].astype(object)
products.info()
products.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 247 entries, 0 to 246
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            247 non-null    int64
 1   title         247 non-null    object
 2   product_type  242 non-null    object
 3   created_at    247 non-null    object
 4   published_at  223 non-null    object
dtypes: int64(1), object(4)
memory usage: 9.8+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 247 entries, 0 to 246
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            247 non-null    object
 1   title         247 non-null    object
 2   product_type  242 non-null    object
 3   created_at    247 non-null    object
 4   published_at  223 non-null    object
dtypes: object(5)
memory usage: 9.8+ KB
```

[10]:

```
            id                             title product_type  created_at  \
0  12927633311  6d1eeb39ae340f8d01d93779f80595ed        Dress  2016-08-18
1  12927632095  0e6e45ad42707e9732119f4b98aec7ce       Bomber  2016-08-18
2  12927625695  27d598cb953eff3667f7d051fe795284       Shirts  2016-08-18
3  12928059103  fb337868ffefe5e008e8dc6d6a4f283a       Blazer  2016-08-18
4  12927632799  d57bc87aca919b4758da6974cdf607fa       Hooide  2016-08-18

  published_at
0   2016-08-18
1   2016-08-18
2   2018-02-05
3   2016-08-18
4          NaN
```

[11]:

```
products = products.rename(columns={'id': 'product_id', 'created_at':
 ↪'product_create_at','published_at':'product_published_at'})
```

product_type, published_at have null value

[12]:
```python
# Traffic Dataset
traffic = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/traffic.csv')
traffic.info()
traffic.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 579 entries, 0 to 578
Data columns (total 8 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   index                  579 non-null    int64
 1   date_day               579 non-null    object
 2   page_views             579 non-null    int64
 3   sessions               579 non-null    int64
 4   product_detail_views   579 non-null    int64
 5   product_checkouts      579 non-null    int64
 6   product_adds_to_carts  579 non-null    int64
 7   avg_session_in_s       579 non-null    float64
dtypes: float64(1), int64(6), object(1)
memory usage: 36.3+ KB
```

[12]:
```
   index    date_day  page_views  sessions  product_detail_views  \
0      0  2016-08-17         204         6                     0
1      1  2016-08-18         661        27                     0
2      2  2016-08-19         241        12                     0
3      3  2016-08-20         534        23                     0
4      4  2016-08-21       10276      4946                     0

   product_checkouts  product_adds_to_carts  avg_session_in_s
0                  0                      0       2374.166667
1                  0                      0       1632.111111
2                  0                      0       1891.250000
3                  0                      0       1557.956522
4                  0                      0         73.470481
```

No null value

[13]:
```python
# Transactions Dataset
transactions = pd.read_csv('/Users/rwang0104/Desktop/AIPin/ecommerce/
 ↪transactions.csv')
transactions.info()
transactions['order_id']=transactions['order_id'].astype(object)
transactions['id']=transactions['id'].astype(object)
transactions['parent_id']=transactions['parent_id'].astype(object)
transactions.info()
transactions.head()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 27563 entries, 0 to 27562
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   order_id    27563 non-null  int64
 1   id          27563 non-null  int64
 2   parent_id   4877 non-null   float64
 3   amount      27563 non-null  float64
 4   error_code  1643 non-null   object
 5   kind        27563 non-null  object
 6   status      27563 non-null  object
 7   created_at  27563 non-null  object
dtypes: float64(2), int64(2), object(4)
memory usage: 1.7+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27563 entries, 0 to 27562
Data columns (total 8 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   order_id    27563 non-null  object
 1   id          27563 non-null  object
 2   parent_id   4877 non-null   object
 3   amount      27563 non-null  float64
 4   error_code  1643 non-null   object
 5   kind        27563 non-null  object
 6   status      27563 non-null  object
 7   created_at  27563 non-null  object
dtypes: float64(1), object(7)
memory usage: 1.7+ MB
```

[13]:

```
     order_id          id     parent_id  amount error_code           kind  \
0  7675398239  8330669343           NaN   44.57        NaN  authorization
1  7675398239  8331258783  8330669343.0    0.00        NaN           void
2  7676331935  8331688479           NaN  124.55        NaN  authorization
3  7676363167  8331722975           NaN   97.68        NaN  authorization
4  7676539359  8331919391           NaN  131.10        NaN  authorization

    status  created_at
0  success  2016-08-21
1  success  2016-08-21
2  success  2016-08-21
3  success  2016-08-21
4  success  2016-08-21
```

parent_id, error_code have null value

2 Part 2: Data Exploratory Analysis

[14]:
```python
orders['order_created_at'] = pd.to_datetime(orders['order_created_at'])
orders.head()
```

[14]:
```
     order_id order_created_at order_closed_at cancelled_at customer_id  \
0  7675398239       2016-08-21      2016-08-25   2016-08-22  8683754719
1  7676331935       2016-08-22      2016-08-22          NaN  8686224991
2  7676363167       2016-08-22             NaN   2016-08-22  8686224991
3  7676539359       2016-08-22      2016-08-22          NaN  8686915935
4  7676549855       2016-08-22      2016-08-22          NaN  8686924319

  financial_status fulfillment_status processed_at  total_price  \
0           voided                NaN   2016-08-21        44.57
1         refunded                NaN   2016-08-22       124.55
2           voided                NaN   2016-08-22        97.68
3             paid          fulfilled   2016-08-22       131.10
4             paid          fulfilled   2016-08-22        91.12

   shipping_rate  subtotal_price  total_discounts  total_line_items_price
0           6.33            35.0              0.0                    35.0
1           0.00           114.0              0.0                   114.0
2           7.00            83.0              0.0                    83.0
3           0.00           120.0              0.0                   120.0
4           7.00            77.0              0.0                    77.0
```

[15]:
```python
# Merge orders_items and products table
df2_1 = pd.merge(left=orders_items, right=products, how='left', on='product_id')
df2_1.head()
```

[15]:
```
  orders_items_id    order_id      product_id  \
0     13325125855  7675398239  12927629215.0
1     13327045983  7676331935  12927632095.0
2     13327109727  7676363167  12928055775.0
3     13327495903  7676539359  12927625695.0
4     13327518751  7676549855  12927690655.0

                      product_style   variant_id  \
0  2c259a42d38f5f097274beff811168e2  50547057311
1  dd804c4025d230467823200aa82e9219  50547118303
2  f4e2e3c5433e4120889e2a7e0e0180a8  50553858975
3  08ba660ec5643520a73108bef6f3ddd6  50547001887
4  68ac90e5df73ae9b662174b21dc1586f  50548035807

                                sku                     product_title  \
0  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
1  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
```

```
2  0be0c8bf78ecf36416a40c9012acd19e  bede8c8f4e3c9c9d9a061d9a8d086cdc
3  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
4  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5

  fulfillment_status  price  quantity                             title  \
0                NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
1                NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
2                NaN   58.0         1  bede8c8f4e3c9c9d9a061d9a8d086cdc
3          fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
4          fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5

  product_type product_create_at product_published_at
0        Tunic        2016-08-18                  NaN
1       Bomber        2016-08-18           2016-08-18
2     Trousers        2016-08-18           2016-08-18
3       Shirts        2016-08-18           2018-02-05
4       Shirts        2016-08-18                  NaN
```

[16]:
```python
# Merge df2_1 and orders table
df2_2 = pd.merge(left=df2_1, right=orders, how='left', on='order_id')
df2_2.head()
```

[16]:
```
  orders_items_id    order_id     product_id  \
0     13325125855  7675398239  12927629215.0
1     13327045983  7676331935  12927632095.0
2     13327109727  7676363167  12928055775.0
3     13327495903  7676539359  12927625695.0
4     13327518751  7676549855  12927690655.0

                     product_style   variant_id  \
0  2c259a42d38f5f097274beff811168e2  50547057311
1  dd804c4025d230467823200aa82e9219  50547118303
2  f4e2e3c5433e4120889e2a7e0e0180a8  50553858975
3  08ba660ec5643520a73108bef6f3ddd6  50547001887
4  68ac90e5df73ae9b662174b21dc1586f  50548035807

                               sku                     product_title  \
0  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
1  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
2  0be0c8bf78ecf36416a40c9012acd19e  bede8c8f4e3c9c9d9a061d9a8d086cdc
3  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
4  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5

  fulfillment_status_x  price  quantity                             title  \
0                  NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
1                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
2                  NaN   58.0         1  bede8c8f4e3c9c9d9a061d9a8d086cdc
```

```
3           fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
4           fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5

  product_type product_create_at product_published_at order_created_at  \
0        Tunic        2016-08-18                  NaN       2016-08-21
1       Bomber        2016-08-18           2016-08-18       2016-08-22
2     Trousers        2016-08-18           2016-08-18       2016-08-22
3       Shirts        2016-08-18           2018-02-05       2016-08-22
4       Shirts        2016-08-18                  NaN       2016-08-22

  order_closed_at cancelled_at customer_id financial_status  \
0      2016-08-25   2016-08-22  8683754719           voided
1      2016-08-22          NaN  8686224991         refunded
2             NaN   2016-08-22  8686224991           voided
3      2016-08-22          NaN  8686915935             paid
4      2016-08-22          NaN  8686924319             paid

  fulfillment_status_y processed_at  total_price  shipping_rate  \
0                  NaN   2016-08-21        44.57           6.33
1                  NaN   2016-08-22       124.55           0.00
2                  NaN   2016-08-22        97.68           7.00
3            fulfilled   2016-08-22       131.10           0.00
4            fulfilled   2016-08-22        91.12           7.00

   subtotal_price  total_discounts  total_line_items_price
0            35.0              0.0                    35.0
1           114.0              0.0                   114.0
2            83.0              0.0                    83.0
3           120.0              0.0                   120.0
4            77.0              0.0                    77.0
```

[17]:
```
# Calculate order item sale
df2_2['order_item_sale'] = df2_2['price']*df2_2['quantity']
```

[18]:
```
# Merge df2_2 and transactions table
df2_3 = pd.merge(left=df2_2, right=transactions, how='left', on='order_id')
df2_3.head()
```

[18]:
```
  orders_items_id    order_id     product_id  \
0     13325125855  7675398239  12927629215.0
1     13325125855  7675398239  12927629215.0
2     13327045983  7676331935  12927632095.0
3     13327045983  7676331935  12927632095.0
4     13327045983  7676331935  12927632095.0

                     product_style   variant_id  \
0  2c259a42d38f5f097274beff811168e2  50547057311
```

```
1  2c259a42d38f5f097274beff811168e2  50547057311
2  dd804c4025d230467823200aa82e9219  50547118303
3  dd804c4025d230467823200aa82e9219  50547118303
4  dd804c4025d230467823200aa82e9219  50547118303

                                sku                     product_title  \
0  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
1  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
2  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
3  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
4  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce

  fulfillment_status_x  price  quantity                             title  \
0                  NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
1                  NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
2                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
3                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
4                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce

  product_type product_create_at product_published_at order_created_at  \
0        Tunic        2016-08-18                  NaN       2016-08-21
1        Tunic        2016-08-18                  NaN       2016-08-21
2       Bomber        2016-08-18           2016-08-18       2016-08-22
3       Bomber        2016-08-18           2016-08-18       2016-08-22
4       Bomber        2016-08-18           2016-08-18       2016-08-22

  order_closed_at cancelled_at customer_id financial_status  \
0      2016-08-25   2016-08-22  8683754719           voided
1      2016-08-25   2016-08-22  8683754719           voided
2      2016-08-22          NaN  8686224991         refunded
3      2016-08-22          NaN  8686224991         refunded
4      2016-08-22          NaN  8686224991         refunded

  fulfillment_status_y processed_at  total_price  shipping_rate  \
0                  NaN   2016-08-21        44.57           6.33
1                  NaN   2016-08-21        44.57           6.33
2                  NaN   2016-08-22       124.55           0.00
3                  NaN   2016-08-22       124.55           0.00
4                  NaN   2016-08-22       124.55           0.00

   subtotal_price  total_discounts  total_line_items_price  order_item_sale  \
0            35.0              0.0                    35.0             35.0
1            35.0              0.0                    35.0             35.0
2           114.0              0.0                   114.0             79.0
3           114.0              0.0                   114.0             79.0
4           114.0              0.0                   114.0             79.0
```

```
           id      parent_id  amount error_code           kind   status  \
0  8330669343            NaN   44.57        NaN  authorization  success
1  8331258783  8330669343.0    0.00        NaN           void  success
2  8331688479            NaN  124.55        NaN  authorization  success
3  8333317599  8331688479.0  124.55        NaN        capture  success
4  8333318239  8333317599.0  124.55        NaN         refund  success

   created_at
0  2016-08-21
1  2016-08-21
2  2016-08-21
3  2016-08-22
4  2016-08-22
```

[19]:
```
# Select order_item status is fulfilled & transaction status is success
df2_4 = df2_3[(df2_3['fulfillment_status_x'] == 'fulfilled') & (df2_3['status']␣
↪== 'success')]
```

[20]:
```
# Add a column YYYY-MM
df2_4['month_year'] = df2_4['order_created_at'].dt.to_period('M')
df2_4.head()
```

[20]:
```
   orders_items_id    order_id     product_id  \
7      13327495903  7676539359  12927625695.0
8      13327495903  7676539359  12927625695.0
9      13327518751  7676549855  12927690655.0
10     13327518751  7676549855  12927690655.0
11     13327526495  7676553055  12950530079.0

                       product_style   variant_id  \
7   08ba660ec5643520a73108bef6f3ddd6  50547001887
8   08ba660ec5643520a73108bef6f3ddd6  50547001887
9   68ac90e5df73ae9b662174b21dc1586f  50548035807
10  68ac90e5df73ae9b662174b21dc1586f  50548035807
11  8945e6be376ffa754e06840e4865cc24  50766799839

                                 sku                     product_title  \
7   0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
8   0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
9   38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5
10  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5
11  2931fc65c83f771a597527925ff97131  08bbf9d4710e8bdbfd07c763ecb2f9e3

   fulfillment_status_x  price  quantity                             title  \
7             fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
8             fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
9             fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5
```

```
10           fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5
11           fulfilled   68.0         1  08bbf9d4710e8bdbfd07c763ecb2f9e3

   product_type product_create_at product_published_at order_created_at  \
7        Shirts        2016-08-18           2018-02-05       2016-08-22
8        Shirts        2016-08-18           2018-02-05       2016-08-22
9        Shirts        2016-08-18                  NaN       2016-08-22
10       Shirts        2016-08-18                  NaN       2016-08-22
11     Jumpsuit        2016-08-21                  NaN       2016-08-22

   order_closed_at cancelled_at customer_id financial_status  \
7       2016-08-22          NaN  8686915935             paid
8       2016-08-22          NaN  8686915935             paid
9       2016-08-22          NaN  8686924319             paid
10      2016-08-22          NaN  8686924319             paid
11      2016-08-22          NaN  8687041311             paid

   fulfillment_status_y processed_at  total_price  shipping_rate  \
7             fulfilled   2016-08-22       131.10            0.0
8             fulfilled   2016-08-22       131.10            0.0
9             fulfilled   2016-08-22        91.12            7.0
10            fulfilled   2016-08-22        91.12            7.0
11            fulfilled   2016-08-22        75.00            7.0

    subtotal_price  total_discounts  total_line_items_price  order_item_sale  \
7            120.0              0.0                   120.0             25.0
8            120.0              0.0                   120.0             25.0
9             77.0              0.0                    77.0             25.0
10            77.0              0.0                    77.0             25.0
11            68.0              0.0                    68.0             68.0

            id     parent_id  amount error_code           kind   status  \
7   8331919391           NaN  131.10        NaN  authorization  success
8   8333205471  8331919391.0  131.10        NaN        capture  success
9   8331930399           NaN   91.12        NaN  authorization  success
10  8333205599  8331930399.0   91.12        NaN        capture  success
11  8331934431           NaN   75.00        NaN  authorization  success

    created_at month_year
7   2016-08-21    2016-08
8   2016-08-22    2016-08
9   2016-08-21    2016-08
10  2016-08-22    2016-08
11  2016-08-21    2016-08
```

2.1 1) Trend of sales over the months

[21]:
```
df2_5 = df2_4.groupby(['month_year', 'product_type']).
 ↪agg('sum')['order_item_sale']
df2_5.head()
```

[21]:
```
month_year  product_type
2016-08     Blazer          18644.0
            Blouse          13184.0
            Bodysuit         3936.0
            Bomber          15484.0
            Cardigan         4140.0
Name: order_item_sale, dtype: float64
```

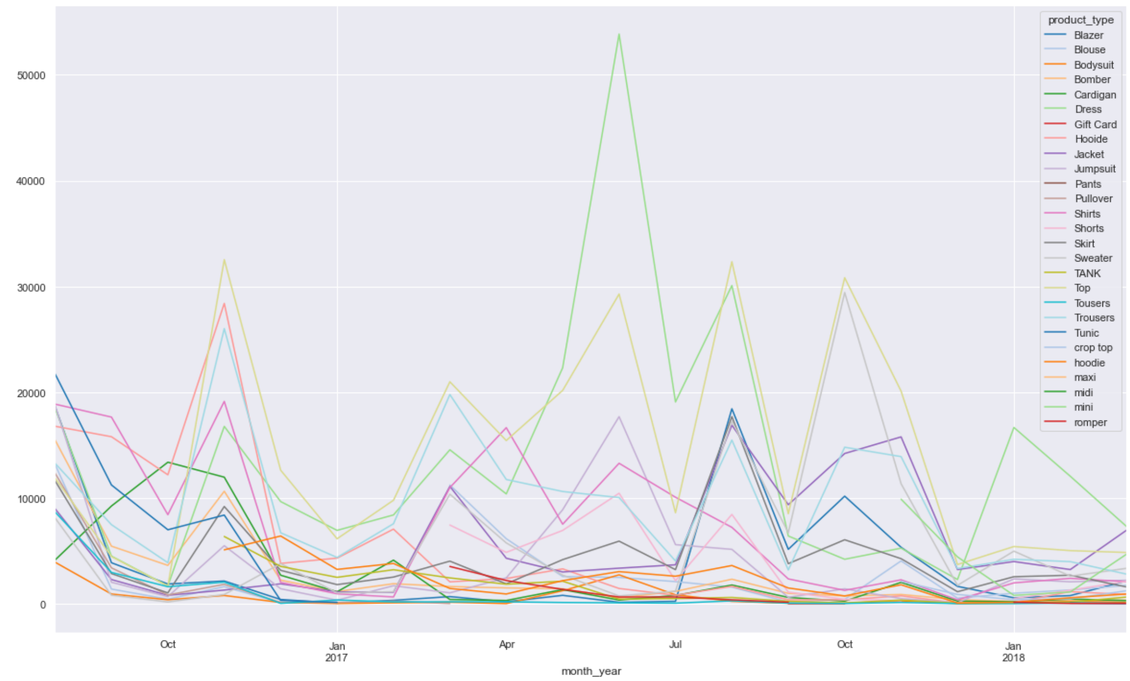
[22]:
```
df2_6 = df2_5.unstack('month_year').transpose()
df2_6.head()
```

[22]:
```
product_type   Blazer   Blouse  Bodysuit   Bomber  Cardigan    Dress  \
month_year
2016-08       18644.0  13184.0    3936.0  15484.0    4140.0  18500.0
2016-09        3884.0   1396.0     928.0   5451.0    9246.0   4514.0
2016-10        1874.0    440.0     384.0   3634.0   13386.0   1692.0
2016-11        2156.4   1629.6     790.4  10639.4   11971.5  16770.6
2016-12         408.0    104.0     128.0   2237.0    2703.0   9666.0

product_type  Gift Card   Hooide  Jacket  Jumpsuit  Pants  Pullover   Shirts  \
month_year
2016-08             NaN  16785.0  8990.0   18768.0    NaN   12348.0  18875.0
2016-09             NaN  15795.0  2294.0    2040.0    NaN    3402.0  17650.0
2016-10             NaN  12195.0   806.0     680.0    NaN     840.0   8425.0
2016-11             NaN  28395.5  1302.0    5494.4    NaN    1843.8  19132.5
2016-12             NaN   3821.0  1892.0    1428.0    NaN     126.0   2025.0

product_type  Shorts    Skirt  Sweater    TANK      Top  Tousers  Trousers  \
month_year
2016-08          NaN  11636.0   8120.0     NaN  11886.0   8685.0   13282.0
2016-09          NaN   2857.0    812.0     NaN   4464.0   2970.0    7482.0
2016-10          NaN   1035.0    174.0     NaN   1678.0   1620.0    3886.0
2016-11          NaN   9199.1    884.0  6365.0  32530.2   2052.0   26013.0
2016-12          NaN   3166.0   3802.0  3541.0  12630.0     45.0    6718.0

product_type    Tunic  crop top  hoodie  maxi  midi  mini  romper
month_year
2016-08       21735.0       NaN     NaN   NaN   NaN   NaN     NaN
2016-09       11235.0       NaN     NaN   NaN   NaN   NaN     NaN
2016-10        7000.0       NaN     NaN   NaN   NaN   NaN     NaN
2016-11        8389.5       NaN  5104.0   NaN   NaN   NaN     NaN
```

```
2016-12          315.0       NaN  6413.0    NaN    NaN    NaN      NaN
```

[23]:
```
camp = sns.color_palette('tab20')
df2_6.plot(figsize=(20,12), color=camp)
```

[23]:
```
<Axes: xlabel='month_year'>
```



Some products show the seasonal feature. We can sale them together

2.2 2) Total order and total cost for each customer

[24]:
```
# Total number of order for each customer
df2_7 = orders[['customer_id','order_id']].groupby('customer_id').agg('count').
 ↪sort_values(by=['order_id'],ascending=False)
df2_7.head()
```

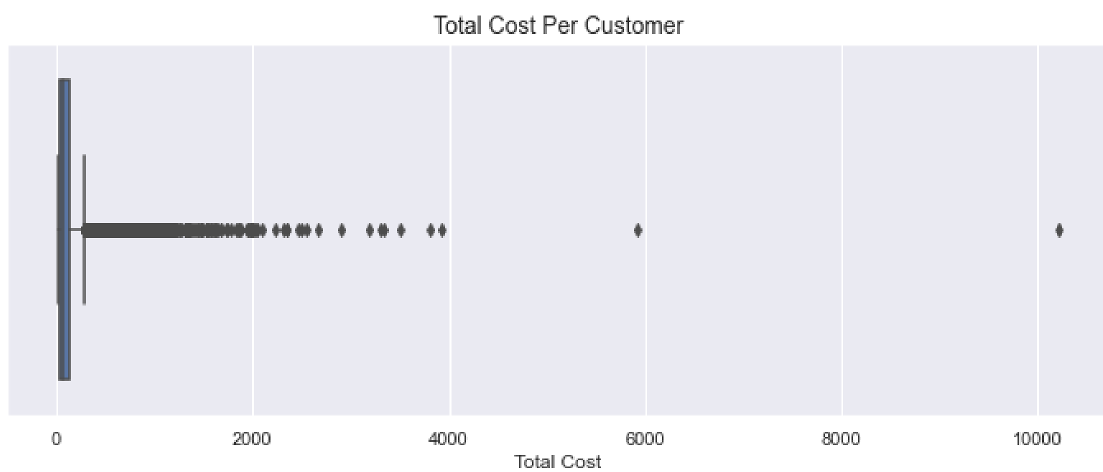
[24]:

	order_id
customer_id	
280479208957	355
413798176253	27
8689371999	25
8688688863	24
8689196063	20

[25]:
```python
# Total cost for each customer
df2_8 = df2_4[['customer_id','order_item_sale']].groupby(['customer_id']).
 ↪agg('sum')
df2_8.head()
```

[25]:
```
             order_item_sale
customer_id
8683754719             933.0
8686224991              87.0
8686913503             115.0
8686915935             240.0
8686924319             154.0
```

[26]:
```python
f, ax = plt.subplots(figsize=(12, 10))
ax1 = plt.subplot(211)
sns.boxplot(df2_7['order_id'])
ax1.set_title('Total Order Per Customer',fontsize=14)
ax1.set_xlabel('Number of Order',fontsize=12)

ax2 = plt.subplot(212)
sns.boxplot(df2_8['order_item_sale'])
ax2.set_title('Total Cost Per Customer', fontsize=14)
ax2.set_xlabel('Total Cost', fontsize=12)

plt.subplots_adjust(hspace=0.4)

plt.show()
```



There is one customer has more than 350 orders when the common number is less than 50. The common number for total cost per customer is less than 2000, and the outlier is 10000+

3 Part 3: Data Cleaning

3.1 1) Check outlier

[27]:
```
# Outlier of customer order
orders[['customer_id','order_id']].groupby('customer_id').agg('count').
↪sort_values(by=['order_id'],ascending=False).head(1)
```

[27]:
```
              order_id
customer_id
280479208957       355
```

[28]:
```python
# Check custoemr_id 280479208957 cost in each order
orders[orders['customer_id']==280479208957].groupby('total_price').agg('count')
# Customer_id 280479208957 has 343 orders with no cost, so this customer_is␣
↪should be defined as a fake account
```

[28]:
```
             order_id  order_created_at  order_closed_at  cancelled_at  \
total_price
0.00              343               343              324             1
18.62               2                 2                0             0
28.00               1                 1                0             0
28.47               1                 1                1             0
43.25               1                 1                0             0
122.64              1                 1                1             0
127.15              1                 1                0             0
209.00              1                 1                0             0
235.97              1                 1                0             0
241.00              1                 1                0             0
296.00              1                 1                0             0
308.17              1                 1                0             0

             customer_id  financial_status  fulfillment_status  processed_at  \
total_price
0.00                 343               343                 326           343
18.62                  2                 2                   0             2
28.00                  1                 1                   0             1
28.47                  1                 1                   1             1
43.25                  1                 1                   0             1
122.64                 1                 1                   1             1
127.15                 1                 1                   1             1
209.00                 1                 1                   1             1
235.97                 1                 1                   1             1
241.00                 1                 1                   1             1
296.00                 1                 1                   1             1
308.17                 1                 1                   1             1

             shipping_rate  subtotal_price  total_discounts  \
total_price
0.00                   343             343              343
18.62                    2               2                2
28.00                    1               1                1
28.47                    1               1                1
43.25                    1               1                1
122.64                   1               1                1
127.15                   1               1                1
209.00                   1               1                1
235.97                   1               1                1
241.00                   1               1                1
```

```
296.00                    1               1                1
308.17                    1               1                1

             total_line_items_price
total_price
0.00                            343
18.62                             2
28.00                             1
28.47                             1
43.25                             1
122.64                            1
127.15                            1
209.00                            1
235.97                            1
241.00                            1
296.00                            1
308.17                            1
```

[29]:
```python
# Delete custoemr_id 280479208957
fake_orders = orders[orders['customer_id']==280479208957]['order_id'].tolist()
customers.drop(customers.loc[customers['id']==280479208957].index, inplace=True)
orders.drop(orders.loc[orders['customer_id']==280479208957].index, inplace=True)
orders_items.drop(orders_items.loc[orders_items['order_id'].isin(fake_orders)].
 ↪index, inplace=True)
```

[30]:
```python
# Double check customer_id 280479208957
orders[orders['customer_id']==280479208957][['order_created_at','total_price']]
```

[30]:
```
Empty DataFrame
Columns: [order_created_at, total_price]
Index: []
```

[31]:
```python
# Outlier of customer order
df2_8[['order_item_sale']].groupby('customer_id').agg('sum').
 ↪sort_values(by=['order_item_sale'],ascending=False).head(1)
# Customer_id 8689196063 has the most total cost
```

[31]:
```
             order_item_sale
customer_id
8689196063           10218.0
```

[32]:
```python
# Check cost of each order of customer_id 8689196063
orders[orders['customer_id']==8689196063].groupby('total_price').agg('count')
# This cusomer_id looks not a fake account
```

[32]:
```
             order_id  order_created_at  order_closed_at  cancelled_at  \
total_price
```

15.76	1	1	1	0
101.17	2	2	2	0
110.98	1	1	1	0
113.88	1	1	1	0
127.02	1	1	1	0
134.69	1	1	1	0
147.83	1	1	1	0
187.91	1	1	1	0
226.67	1	1	1	0
231.05	1	1	1	0
259.68	1	1	0	0
315.71	1	1	1	0
358.07	1	1	1	0
362.71	1	1	1	0
376.05	1	1	1	0
440.72	1	1	1	0
459.94	1	1	1	0
464.28	1	1	0	0
660.29	1	1	1	0

	customer_id	financial_status	fulfillment_status	processed_at	\
total_price					
15.76	1	1	1	1	
101.17	2	2	2	2	
110.98	1	1	1	1	
113.88	1	1	1	1	
127.02	1	1	1	1	
134.69	1	1	1	1	
147.83	1	1	1	1	
187.91	1	1	1	1	
226.67	1	1	1	1	
231.05	1	1	1	1	
259.68	1	1	1	1	
315.71	1	1	1	1	
358.07	1	1	1	1	
362.71	1	1	1	1	
376.05	1	1	1	1	
440.72	1	1	1	1	
459.94	1	1	1	1	
464.28	1	1	1	1	
660.29	1	1	1	1	

	shipping_rate	subtotal_price	total_discounts	\
total_price				
15.76	1	1	1	
101.17	2	2	2	
110.98	1	1	1	

```
113.88                  1              1                1
127.02                  1              1                1
134.69                  1              1                1
147.83                  1              1                1
187.91                  1              1                1
226.67                  1              1                1
231.05                  1              1                1
259.68                  1              1                1
315.71                  1              1                1
358.07                  1              1                1
362.71                  1              1                1
376.05                  1              1                1
440.72                  1              1                1
459.94                  1              1                1
464.28                  1              1                1
660.29                  1              1                1

             total_line_items_price
total_price
15.76                             1
101.17                            2
110.98                            1
113.88                            1
127.02                            1
134.69                            1
147.83                            1
187.91                            1
226.67                            1
231.05                            1
259.68                            1
315.71                            1
358.07                            1
362.71                            1
376.05                            1
440.72                            1
459.94                            1
464.28                            1
660.29                            1
```

3.2 2) Check duplicate rows

[33]:
```python
# Customers Dataset
row_count = customers['id'].count()
unique_row_count = customers['id'].nunique()
print(row_count)
print(unique_row_count)
```

```
44660
44660
```

[34]:
```python
# Orders_items Dataset
row_count = orders_items['orders_items_id'].count()
unique_row_count = orders_items['orders_items_id'].nunique()
print(row_count)
print(unique_row_count)
```

```
35495
35495
```

[35]:
```python
# Orders Dataset
row_count = orders['order_id'].count()
unique_row_count = orders['order_id'].nunique()
print(row_count)
print(unique_row_count)
```

```
21003
21003
```

[36]:
```python
# Products_skus Dataset
row_count = products_skus['id'].count()
unique_row_count = products_skus['id'].nunique()
print(row_count)
print(unique_row_count)
```

```
1356
1356
```

[37]:
```python
# Products Dataset
row_count = products['product_id'].count()
unique_row_count = products['product_id'].nunique()
print(row_count)
print(unique_row_count)
```

```
247
247
```

[38]:
```python
# Traffic Dataset
row_count = traffic['index'].count()
unique_row_count = traffic['index'].nunique()
print(row_count)
print(unique_row_count)
```

```
579
579
```

[39]:
```
# Transactions Dataset
row_count = transactions['id'].count()
unique_row_count = transactions['id'].nunique()
print(row_count)
print(unique_row_count)
```

```
27563
27563
```

No Duplicate in Dataset

[ ]:
```

```

3.3 3) Check Missing Values

[40]:
```
# Customers Dataset
print('total rows: ({} rows)'.format(customers.shape[0]))
print(customers.isnull().sum())
```

```
total rows: (44660 rows)
id                 0
full_name      10962
created_at         0
dtype: int64
```

[41]:
```
# Orders_items Dataset
print('total rows: ({} rows)'.format(orders_items.shape[0]))
print(orders_items.isnull().sum())
```

```
total rows: (35495 rows)
orders_items_id          0
order_id                 0
product_id              24
product_style            0
variant_id               0
sku                      0
product_title            0
fulfillment_status    1387
price                    0
quantity                 0
dtype: int64
```

[42]:
```
# Orders Dataset
print('total rows: ({} rows)'.format(orders.shape[0]))
print(orders.isnull().sum())
```

```
total rows: (21003 rows)
order_id                     0
```

```
order_created_at               0
order_closed_at             1134
cancelled_at               20594
customer_id                    0
financial_status               0
fulfillment_status           657
processed_at                   0
total_price                    0
shipping_rate                  0
subtotal_price                 0
total_discounts                0
total_line_items_price         0
dtype: int64
```

[43]:

```
# Products_skus Dataset
print('total rows: ({} rows)'.format(products_skus.shape[0]))
print(products_skus.isnull().sum())
```

```
total rows: (1356 rows)
id               0
product_id       0
product_style    0
sku              0
created_at       0
price            0
dtype: int64
```

[44]:

```
# Products Dataset
print('total rows: ({} rows)'.format(products.shape[0]))
print(products.isnull().sum())
```

```
total rows: (247 rows)
product_id              0
title                   0
product_type            5
product_create_at       0
product_published_at   24
dtype: int64
```

[45]:

```
# Traffic Dataset
print('total rows: ({} rows)'.format(traffic.shape[0]))
print(traffic.isnull().sum())
```

```
total rows: (579 rows)
index                   0
date_day                0
page_views              0
sessions                0
```

```
product_detail_views     0
product_checkouts        0
product_adds_to_carts    0
avg_session_in_s         0
dtype: int64
```

[46]:
```
# Transactions Dataset
print('total rows: ({} rows)'.format(transactions.shape[0]))
print(transactions.isnull().sum())
```

```
total rows: (27563 rows)
order_id          0
id                0
parent_id     22686
amount            0
error_code    25920
kind              0
status            0
created_at        0
dtype: int64
```

3.4 4) Check Tepo

[47]:
```
products['product_type'].unique()
```

[47]:
```
array(['Dress', 'Bomber', 'Shirts', 'Blazer', 'Hooide', 'Tunic', 'Blouse',
       'Skirt', 'Top', 'TANK', 'Tousers', 'Sweater', 'Cardigan',
       'Trousers', 'Jumpsuit', 'Gift Card', 'hoodie', 'Jacket', 'romper',
       'Shorts', 'mini', 'Bodysuit', nan, 'crop top', 'Pullover', 'Pants',
       'maxi', 'midi', 'Accessory'], dtype=object)
```

[48]:
```
spelling = {'Hooide': 'Hoodie', 'TANK': 'Tank', 'Tousers': 'Trousers',
            'hoodie': 'Hoodie', 'romper': 'Romper', 'mini': 'Mini',
            'crop top': 'Crop Top', 'maxi': 'Maxi', 'midi': 'Midi'}
products['product_type'].replace(spelling, inplace=True)
products['product_type'].unique()
```

[48]:
```
array(['Dress', 'Bomber', 'Shirts', 'Blazer', 'Hoodie', 'Tunic', 'Blouse',
       'Skirt', 'Top', 'Tank', 'Trousers', 'Sweater', 'Cardigan',
       'Jumpsuit', 'Gift Card', 'Jacket', 'Romper', 'Shorts', 'Mini',
       'Bodysuit', nan, 'Crop Top', 'Pullover', 'Pants', 'Maxi', 'Midi',
       'Accessory'], dtype=object)
```

4 Part4: Website traffic and correlation

4.1 1) How's the trend of website traffic and the number of orders over time?

[49]:
```
orders['order_created_at'] = pd.to_datetime(orders['order_created_at'])
num_of_orders = orders.groupby('order_created_at').count()
num_of_orders.head()
```

[49]:
```
                  order_id  order_closed_at  cancelled_at  customer_id  \
order_created_at
2016-08-21               1                1             1            1
2016-08-22             794              780            16          794
2016-08-23             183              179             4          183
2016-08-24              44               43             0           44
2016-08-25              62               61             3           62

                  financial_status  fulfillment_status  processed_at  \
order_created_at
2016-08-21                       1                   0             1
2016-08-22                     794                 775           794
2016-08-23                     183                 180           183
2016-08-24                      44                  44            44
2016-08-25                      62                  54            62

                  total_price  shipping_rate  subtotal_price  total_discounts  \
order_created_at
2016-08-21                  1              1               1                1
2016-08-22                794            794             794              794
2016-08-23                183            183             183              183
2016-08-24                 44             44              44               44
2016-08-25                 62             62              62               62

                  total_line_items_price
order_created_at
2016-08-21                             1
2016-08-22                           794
2016-08-23                           183
2016-08-24                            44
2016-08-25                            62
```

[50]:
```
traffic['date_day'] = pd.to_datetime(traffic['date_day'])
```

[51]:
```
# Merge traffic and num_of_orders table
df_q1 = pd.merge(traffic, num_of_orders, how='inner', left_on = 'date_day',␣
↪right_on='order_created_at')
df_q1 = df_q1.rename(columns = {'order_id': 'order_num'})
df_q1.head()
```

[51]:

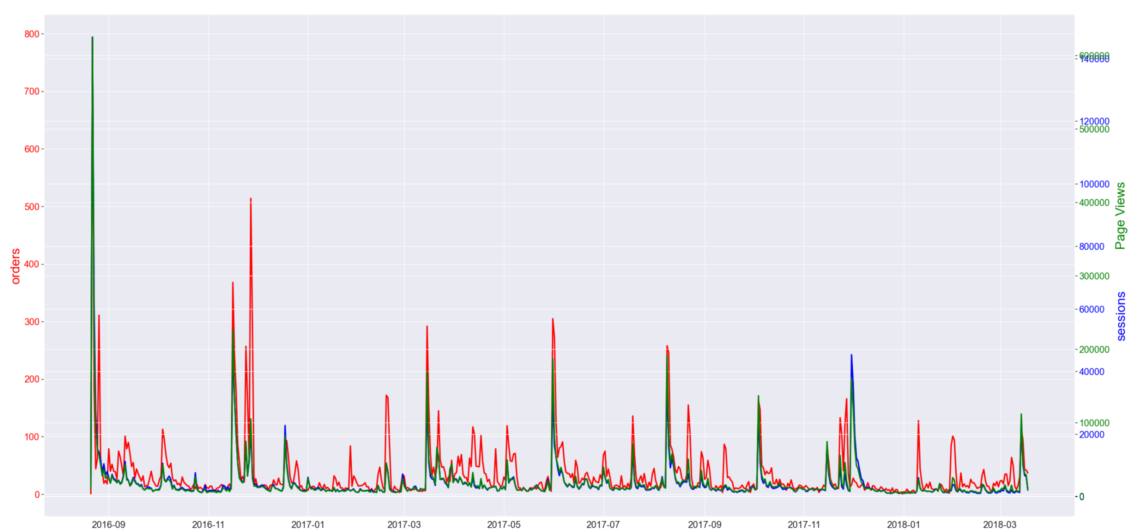
```
   index    date_day  page_views  sessions  product_detail_views  \
0      4  2016-08-21       10276      4946                     0
1      5  2016-08-22      625003    146860                175257
2      6  2016-08-23      220707     61654                 58940
3      7  2016-08-24       93694     27182                 24935
4      8  2016-08-25       63927     15239                 19167

   product_checkouts  product_adds_to_carts  avg_session_in_s  order_num  \
0                  0                      0         73.470481          1
1               5639                  10851        142.407837        794
2                761                   1817        106.161449        183
3                256                    638         98.999669         44
4                901                   1826        130.410854         62

   order_closed_at  cancelled_at  customer_id  financial_status  \
0                1             1            1                 1
1              780            16          794               794
2              179             4          183               183
3               43             0           44                44
4               61             3           62                62

   fulfillment_status  processed_at  total_price  shipping_rate  \
0                   0             1            1              1
1                 775           794          794            794
2                 180           183          183            183
3                  44            44           44             44
4                  54            62           62             62

   subtotal_price  total_discounts  total_line_items_price
0               1                1                       1
1             794              794                     794
2             183              183                     183
3              44               44                      44
4              62               62                      62
```

[52]:

```
fig, ax1 = plt.subplots(figsize=(40,20))
fig.suptitle('Trend of website traffic and the number of orders over time',␣
 ↪fontsize=60)
color = 'tab:cyan'
color = 'red'
ax1.set_ylabel('orders', color=color, fontsize=28)
ax1.plot(df_q1['date_day'], df_q1['order_num'], color=color, linewidth=3)
ax1.tick_params(axis='y', labelcolor=color)
plt.xticks(size=20)
plt.yticks(size=20)
```

```
ax2 = ax1.twinx()
color = 'blue'
ax2.set_xlabel('Time Period', fontsize=28)
ax2.set_ylabel('sessions', color = color, fontsize=28)
ax2.plot(df_q1['date_day'], df_q1['sessions'], color = color, linewidth=3)
ax2.tick_params(axis='y', labelcolor=color)
plt.xticks(size=20)
plt.yticks(size=20)
ax2.yaxis.set_label_coords(1.04,.4)

ax3 = ax1.twinx()
color = 'green'
ax3.set_xlabel('Time Period', fontsize=28)
ax3.set_ylabel('Page Views', color=color, fontsize=28)
ax3.plot(df_q1['date_day'], df_q1['page_views'], color=color, linewidth=3)
ax3.tick_params(axis='y', labelcolor=color)
plt.xticks(size=20)
plt.yticks(size=20)
ax3.yaxis.set_label_coords(1.04,.6)


plt.show()
```

Trend of website traffic and the number of orders over time



Basically, the trend of the number of orders has a strong relationship with the number of page views and sessions. Overall, there is a decreasing trend year by year.

4.2 2) Is there any correlation between the orders and the website traffic?

[53]:
```
df_q1.head()
```

[53]:
```
   index    date_day  page_views  sessions  product_detail_views  \
0      4  2016-08-21       10276      4946                     0
1      5  2016-08-22      625003    146860                175257
2      6  2016-08-23      220707     61654                 58940
3      7  2016-08-24       93694     27182                 24935
4      8  2016-08-25       63927     15239                 19167

   product_checkouts  product_adds_to_carts  avg_session_in_s  order_num  \
0                  0                      0         73.470481          1
1               5639                  10851        142.407837        794
2                761                   1817        106.161449        183
3                256                    638         98.999669         44
4                901                   1826        130.410854         62

   order_closed_at  cancelled_at  customer_id  financial_status  \
0                1             1            1                 1
1              780            16          794               794
2              179             4          183               183
3               43             0           44                44
4               61             3           62                62

   fulfillment_status  processed_at  total_price  shipping_rate  \
0                   0             1            1              1
1                 775           794          794            794
2                 180           183          183            183
3                  44            44           44             44
4                  54            62           62             62

   subtotal_price  total_discounts  total_line_items_price
0               1                1                       1
1             794              794                     794
2             183              183                     183
3              44               44                      44
4              62               62                      62
```

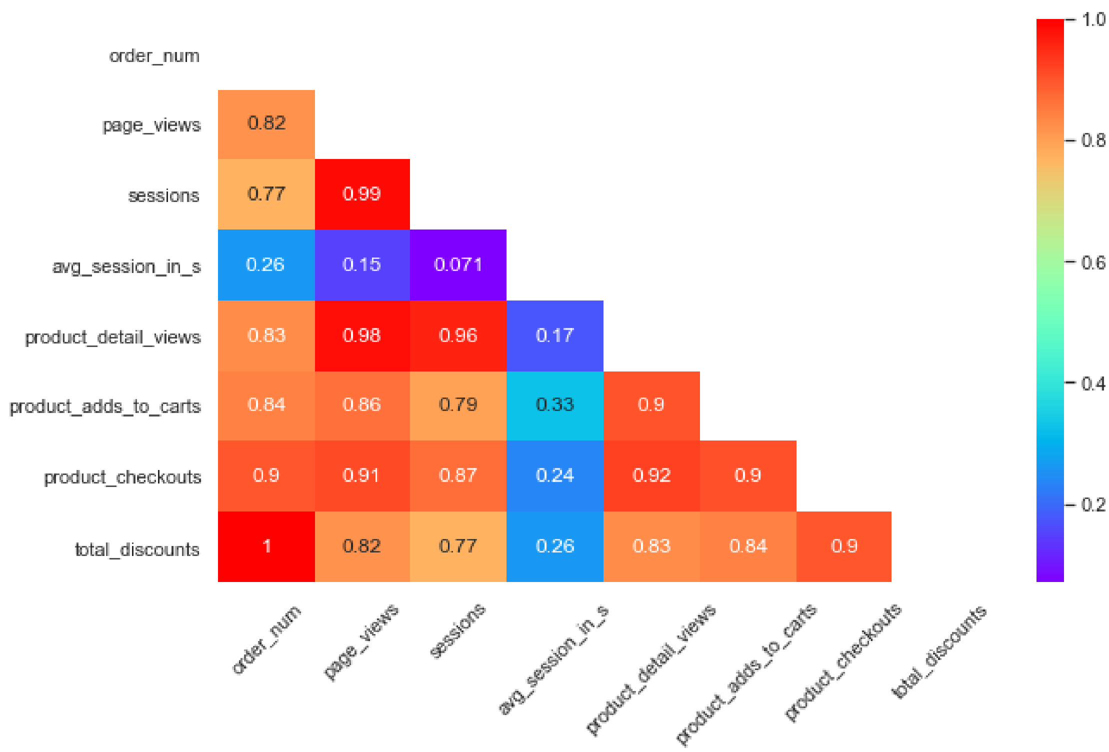
[54]:
```
df_corr = df_q1[['order_num', 'page_views', 'sessions', 'avg_session_in_s',␣
↪'product_detail_views', 'product_adds_to_carts', 'product_checkouts',␣
↪'total_discounts']].corr()
df_corr
```

[54]:
```
                       order_num  page_views  sessions  avg_session_in_s  \
order_num               1.000000    0.815809  0.770344          0.261292
page_views              0.815809    1.000000  0.989081          0.150182
```

```
sessions                0.770344    0.989081  1.000000           0.070872
avg_session_in_s        0.261292    0.150182  0.070872           1.000000
product_detail_views    0.828847    0.984876  0.959438           0.173966
product_adds_to_carts   0.841659    0.863625  0.793949           0.327379
product_checkouts       0.897855    0.910663  0.867374           0.235011
total_discounts         1.000000    0.815809  0.770344           0.261292

                       product_detail_views  product_adds_to_carts  \
order_num                          0.828847               0.841659
page_views                         0.984876               0.863625
sessions                           0.959438               0.793949
avg_session_in_s                   0.173966               0.327379
product_detail_views               1.000000               0.898455
product_adds_to_carts              0.898455               1.000000
product_checkouts                  0.921408               0.903816
total_discounts                    0.828847               0.841659

                       product_checkouts  total_discounts
order_num                       0.897855         1.000000
page_views                      0.910663         0.815809
sessions                        0.867374         0.770344
avg_session_in_s                0.235011         0.261292
product_detail_views            0.921408         0.828847
product_adds_to_carts           0.903816         0.841659
product_checkouts               1.000000         0.897855
total_discounts                 0.897855         1.000000
```

[55]:

```python
# Correlation visualization
plt.figure(figsize=(10,6))
mask = np.zeros_like(df_corr)
mask[np.triu_indices_from(mask)] = True
with sns.axes_style('white'):
    sns.heatmap(df_corr, annot=True, cmap='rainbow', mask=mask)
plt.xticks(rotation=45)
plt.show()
```



Based on the heatmap, the result shows a positive correlation between page views, sessions, and the number of orders. The coefficients are 0.82 and 0.77. Although both coefficients are positive, they are not the features that have the most effect on the total order number

5 Part5: Sales and Products

5.1 1) How's the sales from the different products over the seasons or months?

[56]:
```
# Merge orders_item and products table
df1_q2 = pd.merge(left=orders_items, right=products, how='left',␣
↪on='product_id')
df1_q2.head()
```

[56]:
```
   orders_items_id    order_id     product_id  \
0      13325125855  7675398239  12927629215.0
1      13327045983  7676331935  12927632095.0
2      13327109727  7676363167  12928055775.0
3      13327495903  7676539359  12927625695.0
4      13327518751  7676549855  12927690655.0

                      product_style   variant_id  \
0  2c259a42d38f5f097274beff811168e2  50547057311
1  dd804c4025d230467823200aa82e9219  50547118303
```

```
2  f4e2e3c5433e4120889e2a7e0e0180a8  50553858975
3  08ba660ec5643520a73108bef6f3ddd6  50547001887
4  68ac90e5df73ae9b662174b21dc1586f  50548035807

                                sku                      product_title  \
0  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
1  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
2  0be0c8bf78ecf36416a40c9012acd19e  bede8c8f4e3c9c9d9a061d9a8d086cdc
3  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
4  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5

  fulfillment_status  price  quantity                             title  \
0                NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
1                NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
2                NaN   58.0         1  bede8c8f4e3c9c9d9a061d9a8d086cdc
3          fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
4          fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5

  product_type product_create_at product_published_at
0        Tunic        2016-08-18                  NaN
1       Bomber        2016-08-18           2016-08-18
2     Trousers        2016-08-18           2016-08-18
3       Shirts        2016-08-18           2018-02-05
4       Shirts        2016-08-18                  NaN
```

[57]:

```python
# Merge df_q2 and orders table
df2_q2 = pd.merge(left=df1_q2, right=orders, how='left', on='order_id')

# Calculate order_item_sale
df2_q2['order_item_sale'] = df2_q2['price']*df2_q2['quantity']
df2_q2.head()
```

[57]:

```
   orders_items_id    order_id     product_id  \
0      13325125855  7675398239  12927629215.0
1      13327045983  7676331935  12927632095.0
2      13327109727  7676363167  12928055775.0
3      13327495903  7676539359  12927625695.0
4      13327518751  7676549855  12927690655.0

                      product_style   variant_id  \
0  2c259a42d38f5f097274beff811168e2  50547057311
1  dd804c4025d230467823200aa82e9219  50547118303
2  f4e2e3c5433e4120889e2a7e0e0180a8  50553858975
3  08ba660ec5643520a73108bef6f3ddd6  50547001887
4  68ac90e5df73ae9b662174b21dc1586f  50548035807

                                sku                      product_title  \
```

```
0  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
1  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
2  0be0c8bf78ecf36416a40c9012acd19e  bede8c8f4e3c9c9d9a061d9a8d086cdc
3  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
4  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5

  fulfillment_status_x  price  quantity                             title  \
0                  NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
1                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
2                  NaN   58.0         1  bede8c8f4e3c9c9d9a061d9a8d086cdc
3            fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
4            fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5

  product_type product_create_at product_published_at order_created_at  \
0        Tunic        2016-08-18                  NaN       2016-08-21
1       Bomber        2016-08-18           2016-08-18       2016-08-22
2     Trousers        2016-08-18           2016-08-18       2016-08-22
3       Shirts        2016-08-18           2018-02-05       2016-08-22
4       Shirts        2016-08-18                  NaN       2016-08-22

  order_closed_at cancelled_at customer_id financial_status  \
0      2016-08-25   2016-08-22  8683754719           voided
1      2016-08-22          NaN  8686224991         refunded
2             NaN   2016-08-22  8686224991           voided
3      2016-08-22          NaN  8686915935             paid
4      2016-08-22          NaN  8686924319             paid

  fulfillment_status_y processed_at  total_price  shipping_rate  \
0                  NaN   2016-08-21        44.57           6.33
1                  NaN   2016-08-22       124.55           0.00
2                  NaN   2016-08-22        97.68           7.00
3            fulfilled   2016-08-22       131.10           0.00
4            fulfilled   2016-08-22        91.12           7.00

   subtotal_price  total_discounts  total_line_items_price  order_item_sale
0            35.0              0.0                    35.0             35.0
1           114.0              0.0                   114.0             79.0
2            83.0              0.0                    83.0             58.0
3           120.0              0.0                   120.0             25.0
4            77.0              0.0                    77.0             25.0
```

[58]:
```python
# Merge df2_q2 and transactions table
df3_q2 = pd.merge(left=df2_q2, right=transactions, how='left', on='order_id')
df3_q2.head()
```

[58]:
```
  orders_items_id    order_id     product_id  \
0     13325125855  7675398239  12927629215.0
```

```
1     13325125855  7675398239  12927629215.0
2     13327045983  7676331935  12927632095.0
3     13327045983  7676331935  12927632095.0
4     13327045983  7676331935  12927632095.0

                      product_style   variant_id  \
0  2c259a42d38f5f097274beff811168e2  50547057311
1  2c259a42d38f5f097274beff811168e2  50547057311
2  dd804c4025d230467823200aa82e9219  50547118303
3  dd804c4025d230467823200aa82e9219  50547118303
4  dd804c4025d230467823200aa82e9219  50547118303

                                sku                     product_title  \
0  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
1  000d96b3b77b33af530eec77689bd210  5cfd6c4e00b25e6dec5538928206b7b8
2  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
3  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce
4  e26c77e84b91c9939c23c3e3ef66475a  0e6e45ad42707e9732119f4b98aec7ce

  fulfillment_status_x  price  quantity                             title  \
0                  NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
1                  NaN   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
2                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
3                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce
4                  NaN   79.0         1  0e6e45ad42707e9732119f4b98aec7ce

  product_type product_create_at product_published_at order_created_at  \
0        Tunic        2016-08-18                  NaN       2016-08-21
1        Tunic        2016-08-18                  NaN       2016-08-21
2       Bomber        2016-08-18           2016-08-18       2016-08-22
3       Bomber        2016-08-18           2016-08-18       2016-08-22
4       Bomber        2016-08-18           2016-08-18       2016-08-22

  order_closed_at cancelled_at customer_id financial_status  \
0      2016-08-25   2016-08-22  8683754719           voided
1      2016-08-25   2016-08-22  8683754719           voided
2      2016-08-22          NaN  8686224991         refunded
3      2016-08-22          NaN  8686224991         refunded
4      2016-08-22          NaN  8686224991         refunded

  fulfillment_status_y processed_at  total_price  shipping_rate  \
0                  NaN   2016-08-21        44.57           6.33
1                  NaN   2016-08-21        44.57           6.33
2                  NaN   2016-08-22       124.55           0.00
3                  NaN   2016-08-22       124.55           0.00
4                  NaN   2016-08-22       124.55           0.00
```

```
   subtotal_price  total_discounts  total_line_items_price  order_item_sale  \
0            35.0              0.0                    35.0             35.0
1            35.0              0.0                    35.0             35.0
2           114.0              0.0                   114.0             79.0
3           114.0              0.0                   114.0             79.0
4           114.0              0.0                   114.0             79.0

           id     parent_id  amount error_code           kind   status  \
0  8330669343           NaN   44.57        NaN  authorization  success
1  8331258783  8330669343.0    0.00        NaN           void  success
2  8331688479           NaN  124.55        NaN  authorization  success
3  8333317599  8331688479.0  124.55        NaN        capture  success
4  8333318239  8333317599.0  124.55        NaN         refund  success

   created_at
0  2016-08-21
1  2016-08-21
2  2016-08-21
3  2016-08-22
4  2016-08-22
```

[59]:
```
# Select order_item status is fulfilled & transaction status is success
df_q2 = df3_q2[(df3_q2['fulfillment_status_x'] == 'fulfilled') &␣
↪(df3_q2['status'] == 'success')]
# Add computing column YYYY-MM
df_q2['month_year'] = df_q2['order_created_at'].dt.to_period('M')
df_q2.head()
# Drop duplicates order_items_id and reset index
df_q2.drop_duplicates('orders_items_id', inplace=True)
df_q2.reset_index(drop=True, inplace=True)
df_q2.head()
```

[59]:
```
  orders_items_id    order_id     product_id  \
0     13327495903  7676539359  12927625695.0
1     13327518751  7676549855  12927690655.0
2     13327526495  7676553055  12950530079.0
3     13327549343  7676564127  12927629215.0
4     13327555615  7676566815  12927625695.0

                      product_style   variant_id  \
0  08ba660ec5643520a73108bef6f3ddd6  50547001887
1  68ac90e5df73ae9b662174b21dc1586f  50548035807
2  8945e6be376ffa754e06840e4865cc24  50766799839
3  2c259a42d38f5f097274beff811168e2  50547057823
4  08ba660ec5643520a73108bef6f3ddd6  50547000799

                               sku                     product_title  \
```

```
0  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
1  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5
2  2931fc65c83f771a597527925ff97131  08bbf9d4710e8bdbfd07c763ecb2f9e3
3  549684602f6a1e779751c80445d819fc  5cfd6c4e00b25e6dec5538928206b7b8
4  273bbc291163d41f2458c6694dd40fa1  27d598cb953eff3667f7d051fe795284

  fulfillment_status_x  price  quantity                             title  \
0            fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
1            fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5
2            fulfilled   68.0         1  08bbf9d4710e8bdbfd07c763ecb2f9e3
3            fulfilled   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
4            fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284

  product_type product_create_at product_published_at order_created_at  \
0       Shirts        2016-08-18           2018-02-05       2016-08-22
1       Shirts        2016-08-18                  NaN       2016-08-22
2     Jumpsuit        2016-08-21                  NaN       2016-08-22
3        Tunic        2016-08-18                  NaN       2016-08-22
4       Shirts        2016-08-18           2018-02-05       2016-08-22

  order_closed_at cancelled_at customer_id financial_status  \
0      2016-08-22          NaN  8686915935             paid
1      2016-08-22          NaN  8686924319             paid
2      2016-08-22          NaN  8687041311             paid
3      2016-08-22          NaN  8687317279             paid
4      2016-08-22          NaN  8687317407             paid

  fulfillment_status_y processed_at  total_price  shipping_rate  \
0            fulfilled   2016-08-22       131.10            0.0
1            fulfilled   2016-08-22        91.12            7.0
2            fulfilled   2016-08-22        75.00            7.0
3            fulfilled   2016-08-22        94.40            7.0
4            fulfilled   2016-08-22        34.31            7.0

   subtotal_price  total_discounts  total_line_items_price  order_item_sale  \
0           120.0              0.0                   120.0             25.0
1            77.0              0.0                    77.0             25.0
2            68.0              0.0                    68.0             68.0
3            80.0              0.0                    80.0             35.0
4            25.0              0.0                    25.0             25.0

           id parent_id  amount error_code           kind   status  \
0  8331919391       NaN  131.10        NaN  authorization  success
1  8331930399       NaN   91.12        NaN  authorization  success
2  8331934431       NaN   75.00        NaN  authorization  success
3  8331948191       NaN   94.40        NaN  authorization  success
4  8331950623       NaN   34.31        NaN  authorization  success
```

```
   created_at month_year
0  2016-08-21    2016-08
1  2016-08-21    2016-08
2  2016-08-21    2016-08
3  2016-08-22    2016-08
4  2016-08-22    2016-08
```

[60]:
```python
# Group by YYYY-MM and calculate the sum of order_item_sale
df_q2_1= df_q2[['month_year','order_item_sale']].groupby('month_year').
 ↪agg('sum')
df_q2_1.head()
```

[60]:
```
            order_item_sale
month_year
2016-08            114898.0
2016-09             83748.0
2016-10             54266.0
2016-11            154067.4
2016-12             51654.0
```

[61]:
```python
# Create overall line chart
fig, ax = plt.subplots(figsize=(30,15))
df_q2_1['order_item_sale'].plot(linewidth=5)
fig.suptitle('Trend of overall sales over months', fontsize=30)
ax.set_xlabel('MM-YYYY', fontsize=20)
ax.set_ylabel('Overall sales',fontsize=20)

plt.show()
```

[62]:
```
# Trend of the sales from the different products over the months
```

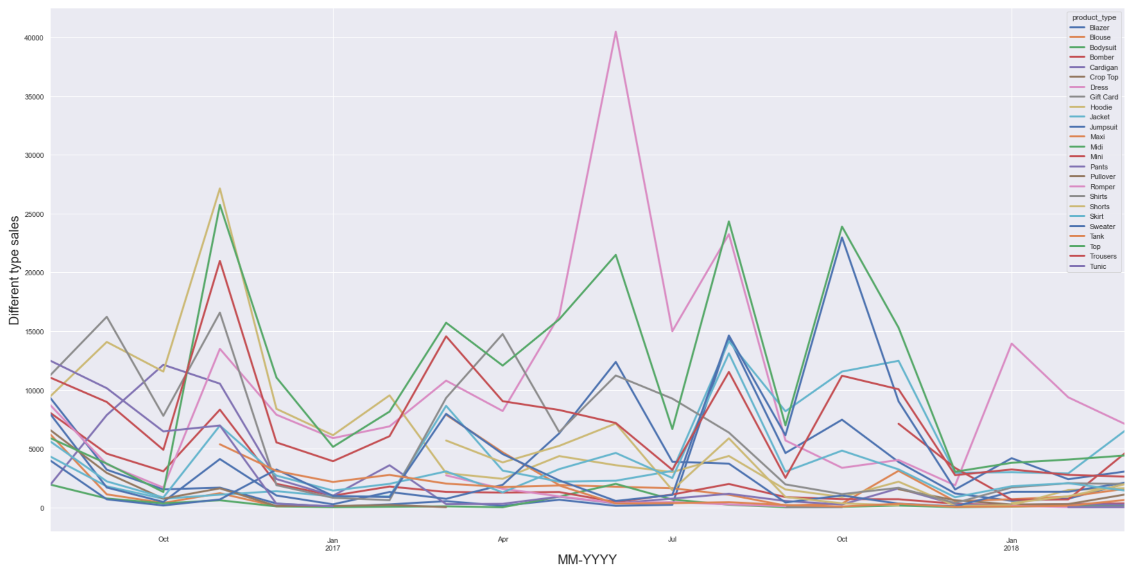
[63]:
```
# Group by YYYY-MM and product type, calculate the sum of order_item_sale
df_q2_2=df_q2.groupby(['month_year', 'product_type']).
↪agg('sum')['order_item_sale']
df_q2_3=df_q2_2.unstack('month_year').transpose()
df_q2_3.head()
```

[63]:
```
product_type  Blazer  Blouse  Bodysuit  Bomber  Cardigan  Crop Top    Dress  \
month_year
2016-08       9331.0  6230.0    1952.0  8058.0    1932.0       NaN   8716.0
2016-09       3216.0  1118.0     768.0  4582.0    7866.0       NaN   3670.0
2016-10       1543.0   388.0     352.0  3081.0   12144.0       NaN   1692.0
2016-11       1689.3  1216.6     614.4  8326.3   10536.3       NaN  13500.2
2016-12        340.0   104.0      96.0  2014.0    2418.0       NaN   7894.0

product_type  Gift Card   Hoodie  Jacket  Jumpsuit  Maxi  Midi  Mini  Pants  \
month_year
2016-08             NaN   9450.0  4340.0    7956.0   NaN   NaN   NaN    NaN
2016-09             NaN  14085.0  1798.0    1700.0   NaN   NaN   NaN    NaN
2016-10             NaN  11565.0   744.0     476.0   NaN   NaN   NaN    NaN
2016-11             NaN  27141.5  1091.2    4120.8   NaN   NaN   NaN    NaN
2016-12             NaN   8392.0  1388.0    1020.0   NaN   NaN   NaN    NaN

product_type  Pullover  Romper   Shirts  Shorts   Skirt  Sweater    Tank  \
month_year
2016-08         6594.0     NaN  11250.0     NaN  5628.0   4002.0     NaN
2016-09         2940.0     NaN  16225.0     NaN  2223.0    696.0     NaN
2016-10          756.0     NaN   7800.0     NaN   851.0    174.0     NaN
2016-11         1629.6     NaN  16582.5     NaN  6969.8    686.0  5377.0
2016-12          126.0     NaN   1900.0     NaN  2713.0   3230.0  3109.0

product_type      Top  Trousers    Tunic
month_year
2016-08        5910.0   11054.0  12495.0
2016-09        3734.0    8977.0  10150.0
2016-10        1318.0    4907.0   6475.0
2016-11       25747.2   20979.2   6975.5
2016-12       11058.0    5537.0    315.0
```

[64]:
```
# Create line chart for different product type
fig, ax = plt.subplots(figsize=(30,15))

df_q2_3.plot(ax=ax, linewidth=3)
fig.suptitle('Trend of different type sales over months', fontsize=30)
```

```python
ax.set_xlabel('MM-YYYY', fontsize=20)
ax.set_ylabel('Different type sales',fontsize=20)

plt.show()
```

Trend of different type sales over months



5.2 2)What are the popular products?

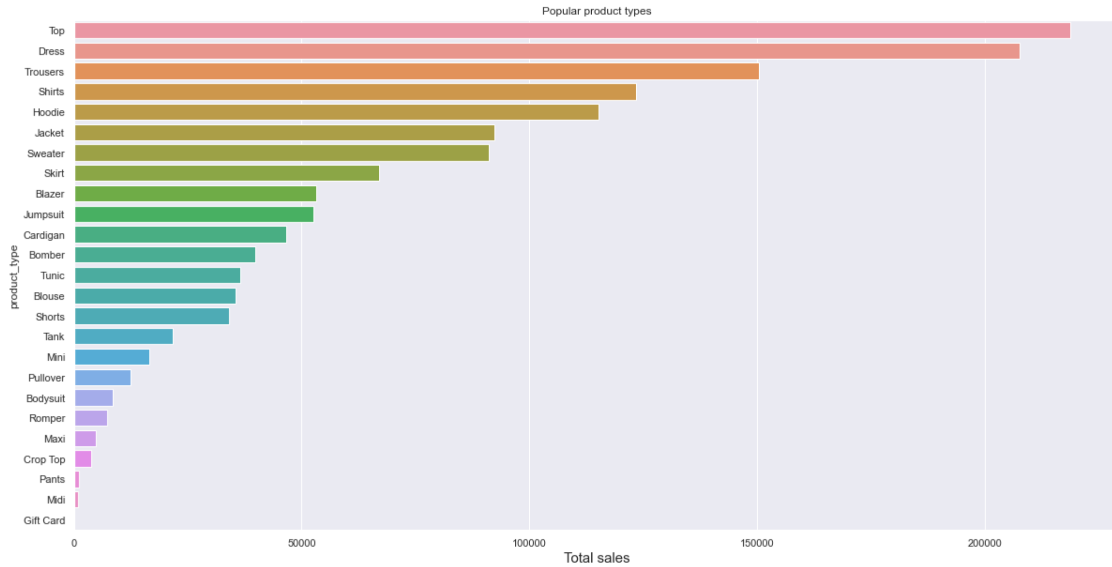
[65]:
```python
# Group by product_type and calculate sum of order_item_sale, and sort
df_q2_4=df_q2[['product_type', 'order_item_sale']].groupby(['product_type']).
↪agg('sum')
df_q2_4=df_q2_4.sort_values(by='order_item_sale', ascending=False)
df_q2_4.head()
```

[65]:

	order_item_sale
product_type	
Top	218929.05
Dress	207772.86
Trousers	150484.70
Shirts	123520.20
Hoodie	115149.63

[66]:
```python
# Create bar chart
fig, ax = plt.subplots(figsize=(20, 10))
ax = sns.barplot(x=df_q2_4['order_item_sale'],y=df_q2_4.index)
ax.set_xlabel('Total sales', fontsize=15)
ax.set_title('Popular product types')
```

```
plt.show()
```



5.3 3)Is there any correlation between different products?

[67]:
```
df_q2.head()
```

[67]:
```
   orders_items_id    order_id     product_id  \
0      13327495903  7676539359  12927625695.0
1      13327518751  7676549855  12927690655.0
2      13327526495  7676553055  12950530079.0
3      13327549343  7676564127  12927629215.0
4      13327555615  7676566815  12927625695.0

                      product_style   variant_id  \
0  08ba660ec5643520a73108bef6f3ddd6  50547001887
1  68ac90e5df73ae9b662174b21dc1586f  50548035807
2  8945e6be376ffa754e06840e4865cc24  50766799839
3  2c259a42d38f5f097274beff811168e2  50547057823
4  08ba660ec5643520a73108bef6f3ddd6  50547000799

                                sku                     product_title  \
0  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
1  38de0d087208588510907b5c2d149e4b  07dd8ba2ccadf3f3766750f10f6d05b5
2  2931fc65c83f771a597527925ff97131  08bbf9d4710e8bdbfd07c763ecb2f9e3
3  549684602f6a1e779751c80445d819fc  5cfd6c4e00b25e6dec5538928206b7b8
4  273bbc291163d41f2458c6694dd40fa1  27d598cb953eff3667f7d051fe795284

  fulfillment_status_x  price  quantity                             title  \
```

```
0           fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284
1           fulfilled   25.0         1  07dd8ba2ccadf3f3766750f10f6d05b5
2           fulfilled   68.0         1  08bbf9d4710e8bdbfd07c763ecb2f9e3
3           fulfilled   35.0         1  5cfd6c4e00b25e6dec5538928206b7b8
4           fulfilled   25.0         1  27d598cb953eff3667f7d051fe795284

  product_type product_create_at product_published_at order_created_at  \
0       Shirts        2016-08-18           2018-02-05       2016-08-22
1       Shirts        2016-08-18                  NaN       2016-08-22
2     Jumpsuit        2016-08-21                  NaN       2016-08-22
3        Tunic        2016-08-18                  NaN       2016-08-22
4       Shirts        2016-08-18           2018-02-05       2016-08-22

  order_closed_at cancelled_at customer_id financial_status  \
0      2016-08-22          NaN  8686915935             paid
1      2016-08-22          NaN  8686924319             paid
2      2016-08-22          NaN  8687041311             paid
3      2016-08-22          NaN  8687317279             paid
4      2016-08-22          NaN  8687317407             paid

  fulfillment_status_y processed_at  total_price  shipping_rate  \
0           fulfilled   2016-08-22       131.10            0.0
1           fulfilled   2016-08-22        91.12            7.0
2           fulfilled   2016-08-22        75.00            7.0
3           fulfilled   2016-08-22        94.40            7.0
4           fulfilled   2016-08-22        34.31            7.0

   subtotal_price  total_discounts  total_line_items_price  order_item_sale  \
0           120.0              0.0                   120.0             25.0
1            77.0              0.0                    77.0             25.0
2            68.0              0.0                    68.0             68.0
3            80.0              0.0                    80.0             35.0
4            25.0              0.0                    25.0             25.0

           id parent_id  amount error_code           kind   status  \
0  8331919391       NaN  131.10        NaN  authorization  success
1  8331930399       NaN   91.12        NaN  authorization  success
2  8331934431       NaN   75.00        NaN  authorization  success
3  8331948191       NaN   94.40        NaN  authorization  success
4  8331950623       NaN   34.31        NaN  authorization  success

   created_at month_year
0  2016-08-21    2016-08
1  2016-08-21    2016-08
2  2016-08-21    2016-08
3  2016-08-22    2016-08
4  2016-08-22    2016-08
```

[68]:
```
order_item1 = df_q2[df_q2.fulfillment_status_y == 'fulfilled']
order_item2 = df_q2[df_q2.fulfillment_status_y == 'fulfilled']
```

[69]:
```
df_q2_5 = pd.merge(left=order_item1, right=order_item2, how='inner',␣
 ↪on='order_id')
df_q2_6 = df_q2_5[df_q2_5.product_id_x != df_q2_5.product_id_y]
df_q2_6.head()
```

[69]:
```
  orders_items_id_x    order_id   product_id_x  \
1       13327495903  7676539359  12927625695.0
2       13327495903  7676539359  12927625695.0
4       13327495967  7676539359  12927690655.0
6       13327495967  7676539359  12927690655.0
7       13327495967  7676539359  12927690655.0

                    product_style_x variant_id_x  \
1  08ba660ec5643520a73108bef6f3ddd6  50547001887
2  08ba660ec5643520a73108bef6f3ddd6  50547001887
4  68ac90e5df73ae9b662174b21dc1586f  50548035935
6  68ac90e5df73ae9b662174b21dc1586f  50548035935
7  68ac90e5df73ae9b662174b21dc1586f  50548035935

                              sku_x                   product_title_x  \
1  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
2  0503dec809a8a2600d9acc5249900ecb  27d598cb953eff3667f7d051fe795284
4  0871fce3fac653a5b430adf1eeb66242  07dd8ba2ccadf3f3766750f10f6d05b5
6  0871fce3fac653a5b430adf1eeb66242  07dd8ba2ccadf3f3766750f10f6d05b5
7  0871fce3fac653a5b430adf1eeb66242  07dd8ba2ccadf3f3766750f10f6d05b5

  fulfillment_status_x_x  price_x  quantity_x  \
1              fulfilled     25.0           1
2              fulfilled     25.0           1
4              fulfilled     25.0           1
6              fulfilled     25.0           1
7              fulfilled     25.0           1

                            title_x product_type_x product_create_at_x  \
1  27d598cb953eff3667f7d051fe795284         Shirts          2016-08-18
2  27d598cb953eff3667f7d051fe795284         Shirts          2016-08-18
4  07dd8ba2ccadf3f3766750f10f6d05b5         Shirts          2016-08-18
6  07dd8ba2ccadf3f3766750f10f6d05b5         Shirts          2016-08-18
7  07dd8ba2ccadf3f3766750f10f6d05b5         Shirts          2016-08-18

  product_published_at_x order_created_at_x order_closed_at_x cancelled_at_x  \
1             2018-02-05         2016-08-22        2016-08-22            NaN
2             2018-02-05         2016-08-22        2016-08-22            NaN
4                    NaN         2016-08-22        2016-08-22            NaN
```

```
6                   NaN        2016-08-22        2016-08-22            NaN
7                   NaN        2016-08-22        2016-08-22            NaN

  customer_id_x financial_status_x fulfillment_status_y_x processed_at_x  \
1    8686915935              paid             fulfilled     2016-08-22
2    8686915935              paid             fulfilled     2016-08-22
4    8686915935              paid             fulfilled     2016-08-22
6    8686915935              paid             fulfilled     2016-08-22
7    8686915935              paid             fulfilled     2016-08-22

   total_price_x  shipping_rate_x  subtotal_price_x  total_discounts_x  \
1          131.1              0.0             120.0                0.0
2          131.1              0.0             120.0                0.0
4          131.1              0.0             120.0                0.0
6          131.1              0.0             120.0                0.0
7          131.1              0.0             120.0                0.0

   total_line_items_price_x  order_item_sale_x        id_x parent_id_x  \
1                     120.0               25.0  8331919391         NaN
2                     120.0               25.0  8331919391         NaN
4                     120.0               25.0  8331919391         NaN
6                     120.0               25.0  8331919391         NaN
7                     120.0               25.0  8331919391         NaN

   amount_x error_code_x         kind_x status_x created_at_x month_year_x  \
1     131.1          NaN  authorization  success   2016-08-21      2016-08
2     131.1          NaN  authorization  success   2016-08-21      2016-08
4     131.1          NaN  authorization  success   2016-08-21      2016-08
6     131.1          NaN  authorization  success   2016-08-21      2016-08
7     131.1          NaN  authorization  success   2016-08-21      2016-08

  orders_items_id_y   product_id_y                   product_style_y  \
1       13327495967  12927690655.0  68ac90e5df73ae9b662174b21dc1586f
2       13327496031  12927632799.0  6056dc7fb0e6987bfb6d08a8a707446f
4       13327495903  12927625695.0  08ba660ec5643520a73108bef6f3ddd6
6       13327496031  12927632799.0  6056dc7fb0e6987bfb6d08a8a707446f
7       13327496095  12927625695.0  08ba660ec5643520a73108bef6f3ddd6

  variant_id_y                             sku_y  \
1  50548035935  0871fce3fac653a5b430adf1eeb66242
2  50547135583  2d72be39ffac72ed072005b2a546c4ea
4  50547001887  0503dec809a8a2600d9acc5249900ecb
6  50547135583  2d72be39ffac72ed072005b2a546c4ea
7  50547004383  878cddb2f377e2787dea8075d3f56954

                    product_title_y fulfillment_status_x_y  price_y  \
1  07dd8ba2ccadf3f3766750f10f6d05b5              fulfilled     25.0
```

```
2  d57bc87aca919b4758da6974cdf607fa            fulfilled     45.0
4  27d598cb953eff3667f7d051fe795284            fulfilled     25.0
6  d57bc87aca919b4758da6974cdf607fa            fulfilled     45.0
7  27d598cb953eff3667f7d051fe795284            fulfilled     25.0

   quantity_y                           title_y product_type_y  \
1           1  07dd8ba2ccadf3f3766750f10f6d05b5         Shirts
2           1  d57bc87aca919b4758da6974cdf607fa         Hoodie
4           1  27d598cb953eff3667f7d051fe795284         Shirts
6           1  d57bc87aca919b4758da6974cdf607fa         Hoodie
7           1  27d598cb953eff3667f7d051fe795284         Shirts

  product_create_at_y product_published_at_y order_created_at_y  \
1          2016-08-18                    NaN         2016-08-22
2          2016-08-18                    NaN         2016-08-22
4          2016-08-18             2018-02-05         2016-08-22
6          2016-08-18                    NaN         2016-08-22
7          2016-08-18             2018-02-05         2016-08-22

  order_closed_at_y cancelled_at_y customer_id_y financial_status_y  \
1        2016-08-22            NaN    8686915935               paid
2        2016-08-22            NaN    8686915935               paid
4        2016-08-22            NaN    8686915935               paid
6        2016-08-22            NaN    8686915935               paid
7        2016-08-22            NaN    8686915935               paid

  fulfillment_status_y_y processed_at_y  total_price_y  shipping_rate_y  \
1              fulfilled     2016-08-22          131.1              0.0
2              fulfilled     2016-08-22          131.1              0.0
4              fulfilled     2016-08-22          131.1              0.0
6              fulfilled     2016-08-22          131.1              0.0
7              fulfilled     2016-08-22          131.1              0.0

   subtotal_price_y  total_discounts_y  total_line_items_price_y  \
1             120.0                0.0                     120.0
2             120.0                0.0                     120.0
4             120.0                0.0                     120.0
6             120.0                0.0                     120.0
7             120.0                0.0                     120.0

   order_item_sale_y        id_y parent_id_y  amount_y error_code_y  \
1               25.0  8331919391         NaN     131.1          NaN
2               45.0  8331919391         NaN     131.1          NaN
4               25.0  8331919391         NaN     131.1          NaN
6               45.0  8331919391         NaN     131.1          NaN
7               25.0  8331919391         NaN     131.1          NaN
```

```
         kind_y status_y created_at_y month_year_y
1  authorization  success   2016-08-21      2016-08
2  authorization  success   2016-08-21      2016-08
4  authorization  success   2016-08-21      2016-08
6  authorization  success   2016-08-21      2016-08
7  authorization  success   2016-08-21      2016-08
```

[70]:
```
df_purchased_together = df_q2_6[['product_type_x', 'product_type_y',␣
↪'orders_items_id_x']].groupby(['product_type_x', 'product_type_y']).
↪agg('count')
df_purchased_together = df_purchased_together.
↪rename(columns={'orders_items_id_x':'purchased_counts'})
df_purchased_together.
↪reset_index(level=['product_type_x','product_type_y'],inplace=True)
df_purchased_together = df_purchased_together.rename(columns={'product_type_x':
↪'product_type', 'product_type_y':'product_type_purchased_together'})
df_purchased_together.head()
```

[70]:
```
  product_type product_type_purchased_together  purchased_counts
0       Blazer                          Blazer                70
1       Blazer                          Blouse                17
2       Blazer                        Bodysuit                17
3       Blazer                          Bomber                14
4       Blazer                        Cardigan                17
```

[71]:
```
df_orders = order_item1[['product_type', 'orders_items_id']].
↪groupby(['product_type']).agg('count')
df_orders = df_orders.rename(columns={'orders_items_id':'num_orders'})
df_orders.reset_index(level='product_type',inplace=True)
df_orders.head()
```

[71]:
```
  product_type  num_orders
0       Blazer         763
1       Blouse         814
2     Bodysuit         251
3       Bomber         618
4     Cardigan         794
```

[72]:
```
purchased = pd.merge(left=df_purchased_together, right=df_orders, how='left',␣
↪on='product_type')
purchased.head()
```

[72]:
```
  product_type product_type_purchased_together  purchased_counts  num_orders
0       Blazer                          Blazer                70         763
1       Blazer                          Blouse                17         763
2       Blazer                        Bodysuit                17         763
3       Blazer                          Bomber                14         763
```

```
4       Blazer                            Cardigan                17          763
```

[73]:
```python
purchased['percentage_purchased_together'] = purchased.purchased_counts/
 ↪purchased.num_orders
pd.set_option('display.precision',10)
purchased.sort_values(by='percentage_purchased_together', ascending=False).
 ↪head()
```

[73]:
```
    product_type product_type_purchased_together  purchased_counts  \
121     Crop Top                        Trousers                63
333       Shorts                             Top               536
354        Skirt                             Top               918
289       Romper                             Top               107
443     Trousers                             Top              1345

     num_orders  percentage_purchased_together
121          93                   0.6774193548
333         827                   0.6481257557
354        1504                   0.6103723404
289         183                   0.5846994536
443        2612                   0.5149310873
```

[74]:
```python
hm_df=purchased.
 ↪pivot('product_type','product_type_purchased_together','percentage_purchased_together')
hm_df
```

[74]:
```
product_type_purchased_together        Blazer        Blouse      Bodysuit  \
product_type
Blazer                           0.0917431193  0.0222804718  0.0222804718
Blouse                           0.0208845209  0.1154791155  0.0282555283
Bodysuit                         0.0677290837  0.0916334661  0.0159362550
Bomber                           0.0226537217  0.0323624595  0.0080906149
Cardigan                         0.0214105793  0.0125944584  0.0088161209
Crop Top                         0.0537634409  0.0860215054           NaN
Dress                            0.0228685061  0.0489684315  0.0216256525
Hoodie                           0.0185039370  0.0196850394  0.0055118110
Jacket                           0.0528949249  0.0557541101  0.0100071480
Jumpsuit                         0.0353773585  0.0341981132  0.0330188679
Maxi                             0.0204081633  0.0408163265           NaN
Midi                                      NaN  0.1000000000           NaN
Mini                             0.0120481928  0.0963855422  0.0040160643
Pants                            0.2352941176           NaN           NaN
Pullover                         0.0398671096  0.0365448505  0.0265780731
Romper                                    NaN  0.2459016393  0.0054644809
Shirts                           0.0164175963  0.0223110924  0.0145232583
Shorts                           0.0858524788  0.1366384522  0.0302297461
```

Skirt	0.1050531915	0.0651595745	0.0159574468
Sweater	0.0397932817	0.0403100775	0.0051679587
Tank	0.0028222013	0.0216368768	0.0094073377
Top	0.0353016688	0.0606546855	0.0154043646
Trousers	0.0535987749	0.0585758040	0.0183767228
Tunic	0.0302743614	0.0170293283	0.0141911069

product_type_purchased_together	Bomber	Cardigan	Crop Top \
product_type			
Blazer	0.0183486239	0.0222804718	0.0065530799
Blouse	0.0245700246	0.0122850123	0.0098280098
Bodysuit	0.0199203187	0.0278884462	NaN
Bomber	0.0097087379	0.0631067961	NaN
Cardigan	0.0491183879	NaN	0.0012594458
Crop Top	NaN	0.0107526882	0.0430107527
Dress	0.0129256774	0.0243599304	0.0032314193
Hoodie	0.0362204724	0.0409448819	0.0007874016
Jacket	0.0250178699	0.0150107219	0.0114367405
Jumpsuit	0.0141509434	0.0235849057	0.0035377358
Maxi	NaN	NaN	NaN
Midi	NaN	NaN	0.1000000000
Mini	NaN	0.0040160643	0.0803212851
Pants	NaN	NaN	0.1176470588
Pullover	0.0465116279	0.0265780731	NaN
Romper	0.0054644809	0.0054644809	NaN
Shirts	0.0183119343	0.0246263944	0.0004209640
Shorts	0.0169286578	0.0048367594	0.0060459492
Skirt	0.0239361702	0.0332446809	0.0039893617
Sweater	0.0129198966	0.0165374677	0.0036175711
Tank	0.0150517404	0.0188146754	0.0018814675
Top	0.0157252888	0.0197368421	0.0033697047
Trousers	0.0206738132	0.0245022971	0.0241194487
Tunic	0.0700094607	0.0473036897	NaN

product_type_purchased_together	Dress	Hoodie	Jacket \
product_type			
Blazer	0.1205766710	0.0615989515	0.0969855832
Blouse	0.2420147420	0.0614250614	0.0958230958
Bodysuit	0.3466135458	0.0557768924	0.0557768924
Bomber	0.0841423948	0.1488673139	0.0566343042
Cardigan	0.1234256927	0.1309823678	0.0264483627
Crop Top	0.1397849462	0.0215053763	0.1720430108
Dress	0.2754163560	0.0748197862	0.0487198608
Hoodie	0.1185039370	0.0842519685	0.0362204724
Jacket	0.1401000715	0.0657612580	0.1015010722
Jumpsuit	0.2747641509	0.0495283019	0.0365566038
Maxi	0.1020408163	NaN	0.0204081633

Midi	NaN	NaN	0.1000000000
Mini	0.1124497992	0.0120481928	0.0963855422
Pants	NaN	NaN	0.1176470588
Pullover	0.0697674419	0.1162790698	0.0332225914
Romper	0.1967213115	0.0710382514	0.1530054645
Shirts	0.0829299095	0.0808250895	0.0231530204
Shorts	0.3482466747	0.0592503023	0.0725513906
Skirt	0.2652925532	0.1083776596	0.1097074468
Sweater	0.0976744186	0.0459948320	0.1038759690
Tank	0.1552210724	0.1326434619	0.0263405456
Top	0.2293003851	0.0909820282	0.0629011553
Trousers	0.1519908116	0.0807810107	0.0781010720
Tunic	0.0463576159	0.2336802271	0.0104068117

product_type_purchased_together	Jumpsuit	Maxi	Midi \
product_type			
Blazer	0.0393184797	0.0013106160	NaN
Blouse	0.0356265356	0.0024570025	0.0012285012
Bodysuit	0.1115537849	NaN	NaN
Bomber	0.0194174757	NaN	NaN
Cardigan	0.0251889169	NaN	NaN
Crop Top	0.0322580645	NaN	0.0107526882
Dress	0.0579169774	0.0012428536	NaN
Hoodie	0.0165354331	NaN	NaN
Jacket	0.0221586848	0.0007147963	0.0007147963
Jumpsuit	0.0471698113	NaN	NaN
Maxi	NaN	NaN	0.0204081633
Midi	NaN	0.1000000000	NaN
Mini	0.0080321285	0.0120481928	0.0120481928
Pants	0.0588235294	NaN	NaN
Pullover	0.0299003322	NaN	NaN
Romper	0.0218579235	NaN	NaN
Shirts	0.0212586824	0.0002104820	NaN
Shorts	0.0773881499	0.0012091898	NaN
Skirt	0.0485372340	0.0019946809	NaN
Sweater	0.0175710594	NaN	NaN
Tank	0.0150517404	0.0028222013	NaN
Top	0.0409178434	0.0001604621	0.0001604621
Trousers	0.0386676876	0.0003828484	NaN
Tunic	0.0122989593	NaN	NaN

product_type_purchased_together	Mini	Pants	Pullover \
product_type			
Blazer	0.0039318480	0.0052424640	0.0157273919
Blouse	0.0294840295	NaN	0.0135135135
Bodysuit	0.0039840637	NaN	0.0318725100
Bomber	NaN	NaN	0.0226537217

Cardigan	0.0012594458	NaN	0.0100755668
Crop Top	0.2150537634	0.0215053763	NaN
Dress	0.0069599801	NaN	0.0052199851
Hoodie	0.0011811024	NaN	0.0137795276
Jacket	0.0171551108	0.0014295926	0.0071479628
Jumpsuit	0.0023584906	0.0011792453	0.0106132075
Maxi	0.0612244898	NaN	NaN
Midi	0.3000000000	NaN	NaN
Mini	0.2248995984	0.0040160643	NaN
Pants	0.0588235294	NaN	NaN
Pullover	NaN	NaN	NaN
Romper	0.0054644809	NaN	NaN
Shirts	0.0004209640	0.0002104820	0.0126289202
Shorts	0.0036275695	0.0012091898	NaN
Skirt	0.0079787234	NaN	0.0099734043
Sweater	0.0025839793	NaN	0.0046511628
Tank	0.0018814675	0.0037629351	0.0047036689
Top	0.0048138639	NaN	0.0044929397
Trousers	0.0103369066	NaN	0.0566615620
Tunic	NaN	NaN	0.0520340587

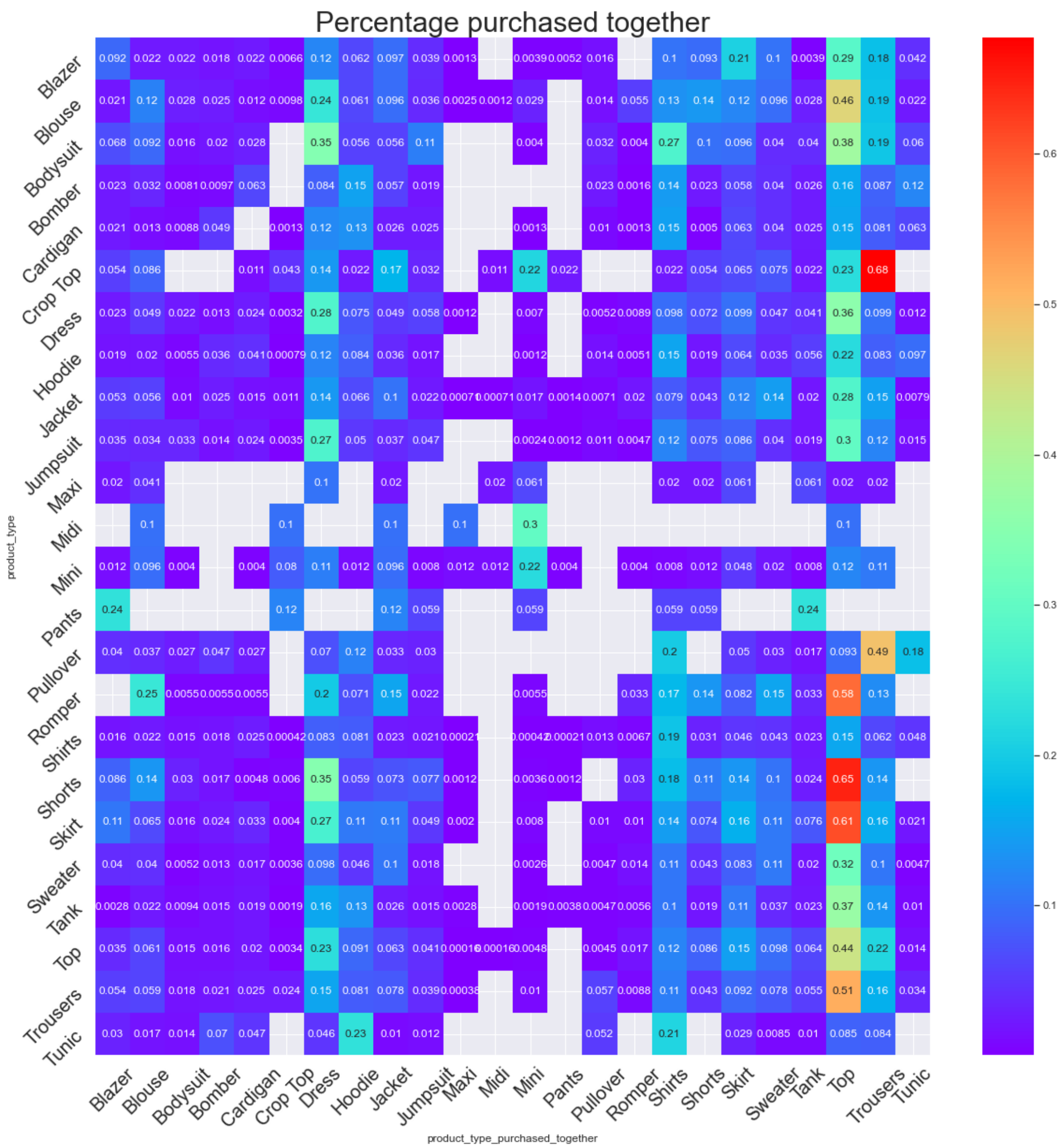
product_type_purchased_together	Romper	Shirts	Shorts \
product_type			
Blazer	NaN	0.1022280472	0.0930537353
Blouse	0.0552825553	0.1302211302	0.1388206388
Bodysuit	0.0039840637	0.2749003984	0.0996015936
Bomber	0.0016181230	0.1407766990	0.0226537217
Cardigan	0.0012594458	0.1473551637	0.0050377834
Crop Top	NaN	0.0215053763	0.0537634409
Dress	0.0089485459	0.0979368630	0.0715883669
Hoodie	0.0051181102	0.1511811024	0.0192913386
Jacket	0.0200142959	0.0786275911	0.0428877770
Jumpsuit	0.0047169811	0.1191037736	0.0754716981
Maxi	NaN	0.0204081633	0.0204081633
Midi	NaN	NaN	NaN
Mini	0.0040160643	0.0080321285	0.0120481928
Pants	NaN	0.0588235294	0.0588235294
Pullover	NaN	0.1993355482	NaN
Romper	0.0327868852	0.1748633880	0.1366120219
Shirts	0.0067354241	0.1894338034	0.0313618186
Shorts	0.0302297461	0.1801692866	0.1088270859
Skirt	0.0099734043	0.1442819149	0.0744680851
Sweater	0.0144702842	0.1054263566	0.0434108527
Tank	0.0056444026	0.1044214487	0.0188146754
Top	0.0171694480	0.1177792041	0.0860077022
Trousers	0.0088055130	0.1125574273	0.0432618683
Tunic	NaN	0.2138126774	NaN

```
product_type_purchased_together         Skirt      Sweater         Tank  \
product_type
Blazer                           0.2070773263 0.1009174312 0.0039318480
Blouse                           0.1203931204 0.0958230958 0.0282555283
Bodysuit                         0.0956175299 0.0398406375 0.0398406375
Bomber                           0.0582524272 0.0404530744 0.0258899676
Cardigan                         0.0629722922 0.0403022670 0.0251889169
Crop Top                         0.0645161290 0.0752688172 0.0215053763
Dress                            0.0991797166 0.0469798658 0.0410141685
Hoodie                           0.0641732283 0.0350393701 0.0555118110
Jacket                           0.1179413867 0.1436740529 0.0200142959
Jumpsuit                         0.0860849057 0.0400943396 0.0188679245
Maxi                             0.0612244898          NaN 0.0612244898
Midi                                      NaN          NaN          NaN
Mini                             0.0481927711 0.0200803213 0.0080321285
Pants                                     NaN          NaN 0.2352941176
Pullover                         0.0498338870 0.0299003322 0.0166112957
Romper                           0.0819672131 0.1530054645 0.0327868852
Shirts                           0.0456745948 0.0429383288 0.0233635024
Shorts                           0.1354292624 0.1015719468 0.0241837969
Skirt                            0.1648936170 0.1070478723 0.0764627660
Sweater                          0.0832041344 0.1085271318 0.0201550388
Tank                             0.1081843838 0.0366886171 0.0225776105
Top                              0.1473042362 0.0980423620 0.0635430039
Trousers                         0.0915007657 0.0777182236 0.0551301685
Tunic                            0.0293282876 0.0085146641 0.0104068117

product_type_purchased_together           Top     Trousers        Tunic
product_type
Blazer                           0.2883355177 0.1834862385 0.0419397117
Blouse                           0.4643734644 0.1879606880 0.0221130221
Bodysuit                         0.3824701195 0.1912350598 0.0597609562
Bomber                           0.1585760518 0.0873786408 0.1197411003
Cardigan                         0.1549118388 0.0806045340 0.0629722922
Crop Top                         0.2258064516 0.6774193548          NaN
Dress                            0.3552075565 0.0986825752 0.0121799652
Hoodie                           0.2232283465 0.0830708661 0.0972440945
Jacket                           0.2802001430 0.1458184417 0.0078627591
Jumpsuit                         0.3007075472 0.1191037736 0.0153301887
Maxi                             0.0204081633 0.0204081633          NaN
Midi                             0.1000000000          NaN          NaN
Mini                             0.1204819277 0.1084337349          NaN
Pants                                     NaN          NaN          NaN
Pullover                         0.0930232558 0.4916943522 0.1827242525
Romper                           0.5846994536 0.1256830601          NaN
Shirts                           0.1544937908 0.0618817091 0.0475689329
```

Shorts	0.6481257557	0.1366384522	NaN
Skirt	0.6103723404	0.1589095745	0.0206117021
Sweater	0.3157622739	0.1049095607	0.0046511628
Tank	0.3725305738	0.1354656632	0.0103480715
Top	0.4390243902	0.2158215661	0.0144415918
Trousers	0.5149310873	0.1646248086	0.0340735069
Tunic	0.0851466414	0.0842005676	NaN

[75]:

```
fig, ax = plt.subplots(figsize=(20,20))
ax = sns.heatmap(hm_df,annot=True, cmap='rainbow')
ax.set_title('Percentage purchased together',fontsize=30)
plt.xticks(size=20,rotation=45)
plt.yticks(size=20,rotation=45)
plt.show()
```



The value in the heatmap is percentage that purchased each product type together instand of correlation. The result shows from the heatmap, most of the percentages are ver small. But there are still few of them around or bigger than 50%

6 Part6: Sales and discount

6.1 1) How's the sales of different products with discount?

[76]:
```
df_q3_1 = df_q2[df_q2['total_discounts'] != 0][['product_type',␣
↪'order_item_sale', 'quantity']].groupby('product_type').agg('sum')
df_q3_1 = df_q3_1.rename(columns={'order_item_sale': 'sales_with_discount',␣
↪'quantity': 'quantity_with_discount'})
df_q3_1.head()
```

[76]:
```
              sales_with_discount  quantity_with_discount
product_type
Blazer                   18321.70                     266
Blouse                   12704.40                     326
Bodysuit                  2418.40                      70
Bomber                    9032.79                     179
Cardigan                  9140.36                     203
```

[77]:
```
df_q3_2 = df_q2[df_q2['total_discounts'] == 0][['product_type',␣
↪'order_item_sale', 'quantity']].groupby('product_type').agg('sum')
df_q3_2 = df_q3_2.rename(columns={'order_item_sale': 'sales_without_discount',␣
↪'quantity': 'quantity_without_discount'})
df_q3_2.head()
```

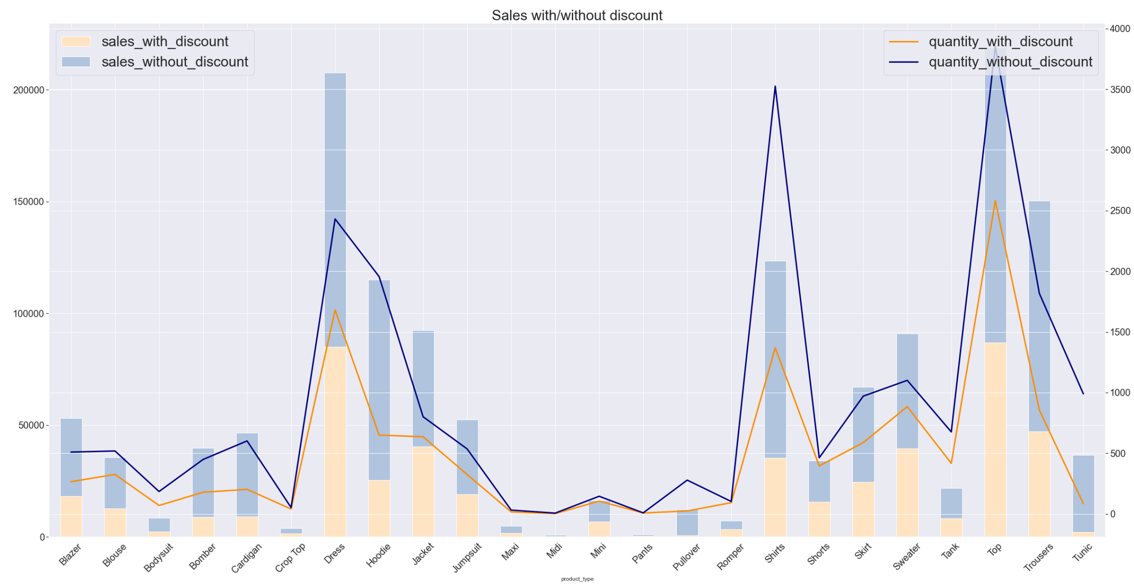
[77]:
```
              sales_without_discount  quantity_without_discount
product_type
Blazer                      34830.70                        510
Blouse                      22828.00                        519
Bodysuit                     6125.60                        185
Bomber                      30755.47                        449
Cardigan                    37532.40                        602
```

[78]:
```
df_q3_3 = pd.merge(df_q3_1, df_q3_2, on='product_type' )
df_q3_3.head()
```

[78]:
```
              sales_with_discount  quantity_with_discount  \
product_type
Blazer                   18321.70                     266
Blouse                   12704.40                     326
Bodysuit                  2418.40                      70
Bomber                    9032.79                     179
Cardigan                  9140.36                     203

              sales_without_discount  quantity_without_discount
product_type
Blazer                      34830.70                        510
Blouse                      22828.00                        519
```

```
Bodysuit                   6125.60                        185
Bomber                    30755.47                        449
Cardigan                  37532.40                        602
```

[79]:
```python
fig, ax1 = plt.subplots(figsize=(40,20))
ax1 = df_q3_3[['sales_with_discount','sales_without_discount']].plot(ax=ax1,␣
↪kind='bar',stacked=True,color=['bisque','lightsteelblue'])
ax1.set_title('Sales with/without discount',fontsize=30)
plt.xticks(size=20,rotation=45)
plt.yticks(size=20)
plt.legend(fontsize=30, loc ="upper left")

ax2 = ax1.twinx()
ax2 = df_q3_3[['quantity_with_discount','quantity_without_discount']].
↪plot(ax=ax2,linewidth=3,color=['darkorange','navy'])
plt.xticks(size=20)
plt.yticks(size=20)
plt.legend(fontsize=30, loc ="upper right")
plt.show()
```



6.2 2) Does the discount promote sales?

[80]:
```python
order_quantity = orders_items[['order_id','quantity']].groupby('order_id').
↪agg('sum').reset_index(level='order_id')
order_quantity.head()
```

[80]:

	order_id	quantity
0	7675398239	1
1	7676331935	2
2	7676363167	2
3	7676539359	4
4	7676549855	2

[81]:

```
df_q3_4=pd.merge(orders,order_quantity,how='left', on='order_id')
df_q3_4['discount_percentage']=df_q3_4['total_discounts']/
 ↪df_q3_4['total_line_items_price']
df_q3_4['discount_orders']=(orders.total_discounts!=0)*1
df_q3_4['no_discount_orders']=(orders.total_discounts==0)*1
df_q3_4['total_orders']=1
df_q3_4['discount_quantity']=(orders.total_discounts!=0)*df_q3_4['quantity']
df_q3_4['no_discount_quantity']=(orders.total_discounts==0)*df_q3_4['quantity']
df_q3_4.info()
df_q3_4.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 21003 entries, 0 to 21002
Data columns (total 20 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   order_id               21003 non-null  object
 1   order_created_at       21003 non-null  datetime64[ns]
 2   order_closed_at        19869 non-null  object
 3   cancelled_at           409 non-null    object
 4   customer_id            21003 non-null  object
 5   financial_status       21003 non-null  object
 6   fulfillment_status     20346 non-null  object
 7   processed_at           21003 non-null  object
 8   total_price            21003 non-null  float64
 9   shipping_rate          21003 non-null  float64
 10  subtotal_price         21003 non-null  float64
 11  total_discounts        21003 non-null  float64
 12  total_line_items_price 21003 non-null  float64
 13  quantity               21003 non-null  int64
 14  discount_percentage    21000 non-null  float64
 15  discount_orders        20669 non-null  float64
 16  no_discount_orders     20669 non-null  float64
 17  total_orders           21003 non-null  int64
 18  discount_quantity      20669 non-null  object
 19  no_discount_quantity   20669 non-null  object
dtypes: datetime64[ns](1), float64(8), int64(2), object(9)
memory usage: 3.9+ MB
```

[81]:
```
     order_id order_created_at order_closed_at cancelled_at customer_id  \
0  7675398239       2016-08-21      2016-08-25   2016-08-22  8683754719
1  7676331935       2016-08-22      2016-08-22          NaN  8686224991
2  7676363167       2016-08-22             NaN   2016-08-22  8686224991
3  7676539359       2016-08-22      2016-08-22          NaN  8686915935
4  7676549855       2016-08-22      2016-08-22          NaN  8686924319

  financial_status fulfillment_status processed_at  total_price  \
0           voided                NaN   2016-08-21        44.57
1         refunded                NaN   2016-08-22       124.55
2           voided                NaN   2016-08-22        97.68
3             paid          fulfilled   2016-08-22       131.10
4             paid          fulfilled   2016-08-22        91.12

   shipping_rate  subtotal_price  total_discounts  total_line_items_price  \
0           6.33            35.0              0.0                    35.0
1           0.00           114.0              0.0                   114.0
2           7.00            83.0              0.0                    83.0
3           0.00           120.0              0.0                   120.0
4           7.00            77.0              0.0                    77.0

   quantity  discount_percentage  discount_orders  no_discount_orders  \
0         1                  0.0              0.0                 1.0
1         2                  0.0              0.0                 1.0
2         2                  0.0              0.0                 1.0
3         4                  0.0              0.0                 1.0
4         2                  0.0              0.0                 1.0

   total_orders discount_quantity no_discount_quantity
0             1               0.0                  1.0
1             1               0.0                  2.0
2             1               0.0                  2.0
3             1               0.0                  4.0
4             1               0.0                  2.0
```

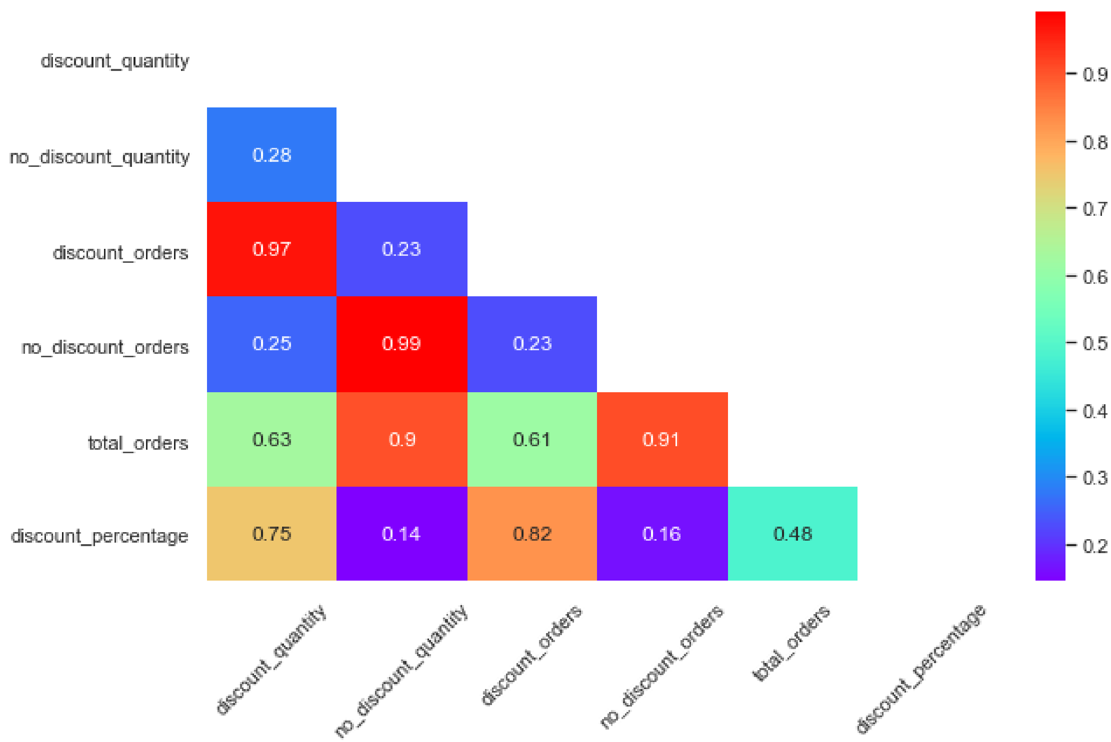
[82]:
```
df_q3_4.discount_quantity = df_q3_4.discount_quantity.astype('float')
df_q3_4.no_discount_quantity = df_q3_4.no_discount_quantity.astype('float')
```

[83]:
```
df_q3_5 = df_q3_4.
 ↪groupby('order_created_at')['discount_quantity','no_discount_quantity','discount_orders','n
 ↪sum()
df_q3_5.head()
```

[83]:
```
                  discount_quantity  no_discount_quantity  discount_orders  \
order_created_at
2016-08-21                      0.0                   1.0              0.0
2016-08-22                      4.0                1432.0              3.0
```

```
2016-08-23                       4.0                   279.0              2.0
2016-08-24                       0.0                    73.0              0.0
2016-08-25                       3.0                    80.0              3.0

                  no_discount_orders  total_orders  discount_percentage
order_created_at
2016-08-21                       1.0             1                  0.0
2016-08-22                     790.0           794                  1.0
2016-08-23                     181.0           183                  0.0
2016-08-24                      44.0            44                  0.0
2016-08-25                      59.0            62                  0.0
```

[84]:
```python
plt.figure(figsize=(10,6))
mask = np.zeros_like(df_q3_5.corr())
mask[np.triu_indices_from(mask)] = True
with sns.axes_style('white'):
    sns.heatmap(df_q3_5.corr(), annot=True, cmap='rainbow', mask=mask)
plt.xticks(rotation=45)
plt.show()
```



The correlation heatmap shows that discount promote sales. Because discount_percentage has positive correlation coefficient with discount_quantity(0.75) and discount_orders(0.82). The correlation of discount_orders and total_orders is only 0.61, and the correlation of no_discount_orders and total_orders is 0.91, which means no_discount_orders has more influential for the total orders because no_discount_orders is more common than discount_orders in reality.

7 Part7: More insights

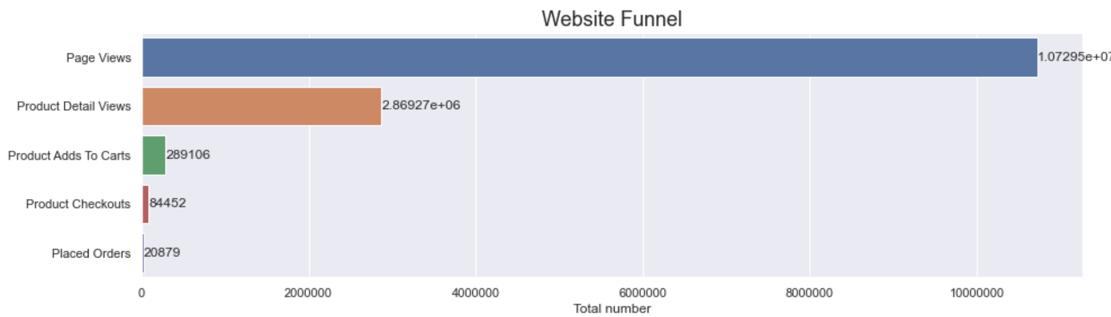
7.1 1) Website funnel

[85]:
```
df_q1.head()
```

[85]:
```
   index    date_day  page_views  sessions  product_detail_views  \
0      4  2016-08-21       10276      4946                     0
1      5  2016-08-22      625003    146860                175257
2      6  2016-08-23      220707     61654                 58940
3      7  2016-08-24       93694     27182                 24935
4      8  2016-08-25       63927     15239                 19167

   product_checkouts  product_adds_to_carts  avg_session_in_s  order_num  \
0                  0                      0     73.4704811969          1
1               5639                  10851    142.4078373962        794
2                761                   1817    106.1614493788        183
3                256                    638     98.9996688985         44
4                901                   1826    130.4108537306         62

   order_closed_at  cancelled_at  customer_id  financial_status  \
0                1             1            1                 1
1              780            16          794               794
2              179             4          183               183
3               43             0           44                44
4               61             3           62                62

   fulfillment_status  processed_at  total_price  shipping_rate  \
0                   0             1            1              1
1                 775           794          794            794
2                 180           183          183            183
3                  44            44           44             44
4                  54            62           62             62

   subtotal_price  total_discounts  total_line_items_price
0               1                1                       1
1             794              794                     794
2             183              183                     183
3              44               44                      44
4              62               62                      62
```

[86]:
```
f, ax= plt.subplots(figsize=(15,4))
ax = sns.barplot(y=['Page Views','Product Detail Views', 'Product Adds To␣
↪Carts', 'Product Checkouts', 'Placed Orders'],
```

```
            x=[df_q1['page_views'].sum() ,df_q1['product_detail_views'].sum(),␣
 ↪df_q1['product_adds_to_carts'].sum(), df_q1['product_checkouts'].sum(),␣
 ↪df_q1['order_num'].sum()]
            )
ax.bar_label(ax.containers[0])
ax.set_title('Website Funnel', fontsize=18)
plt.ticklabel_format(style='plain', axis='x')
plt.xlabel('Total number')
plt.show()
```



[ ]:

[87]:
```
funnel={'Page_Views': df_q1['page_views'].sum(),
        'Product_Detail_Views': df_q1['product_detail_views'].sum(),
        'Product_Add_To_Carts': df_q1['product_adds_to_carts'].sum(),
        'Product_Checkouts': df_q1['product_checkouts'].sum(),
        'Order_Placed': df_q1['order_num'].sum()
}
```

[88]:
```
df_funnel=pd.DataFrame(data=funnel,index=['total_number']).transpose()
df_funnel['lag_number'] = df_funnel['total_number'].shift(periods=1)
df_funnel['conversion_rate'] = df_funnel['total_number']/df_funnel['lag_number']
df_funnel.drop('lag_number', axis=1, inplace=True)
df_funnel['conversion_rate_page_views'] = df_funnel['total_number']/
 ↪df_q1['page_views'].sum()
df_funnel.head()
```

[88]:
```
                      total_number  conversion_rate  \
Page_Views                10729488              NaN
Product_Detail_Views       2869270     0.2674190977
Product_Add_To_Carts        289106     0.1007594266
Product_Checkouts            84452     0.2921143110
Order_Placed                 20879     0.2472291953

                      conversion_rate_page_views
```

```
Page_Views                        1.0000000000
Product_Detail_Views              0.2674190977
Product_Add_To_Carts              0.0269449950
Product_Checkouts                 0.0078710186
Order_Placed                      0.0019459456
```

Overall conversion rate from page views is 0.19%. - Step by step conversion rate: - From Page_Views to Product_Detail_Views views: 26.74% - From Product_Detail_Views to Product_Add_To_Carts: 10.08% - From Product_Add_To_Carts to Product_Checkouts: 29.21% - From Product_Checkouts to Order_Placed: 25.72% - Two steps that I think chould be improved: From Page_Views to Product_Detail_Views & From Product_Detail_Views to Product_Add_To_Carts

7.2 2) Churn Rate

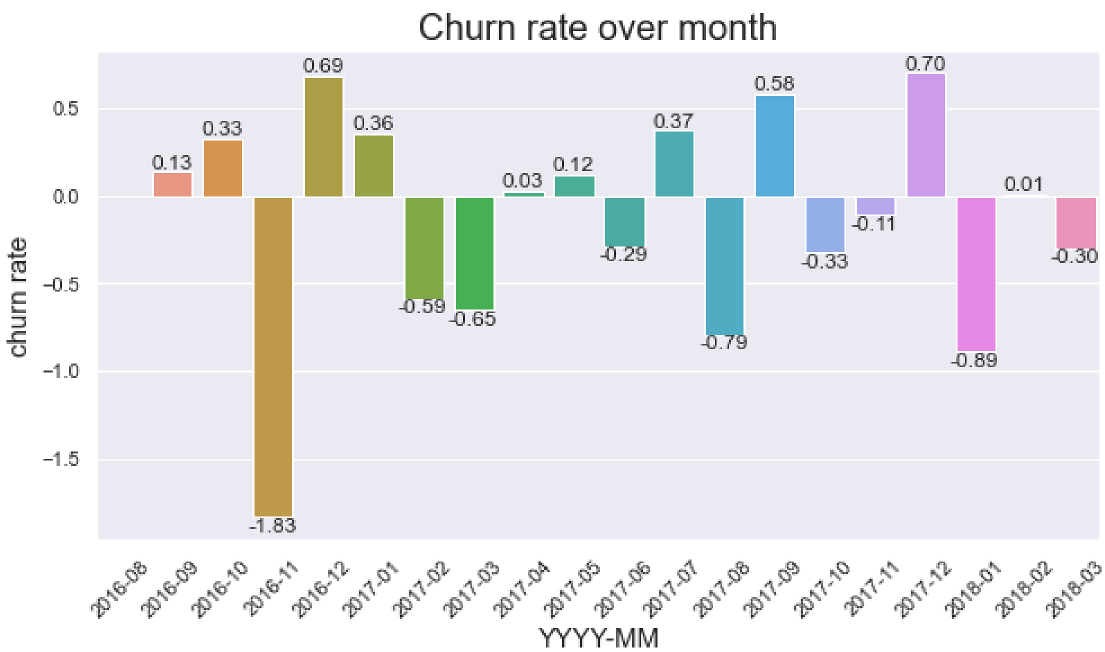
[89]:
```
df_churn = df_q2[['month_year', 'customer_id']].groupby('month_year').
 ↪agg({'customer_id': pd.Series.nunique})
df_churn = df_churn.rename(columns = {'customer_id':'num_customers'})
df_churn.reset_index('month_year',inplace=True)
```

[90]:
```
df_churn['last_num_customers'] = df_churn['num_customers'].shift(periods=1)
df_churn['num_churned'] = df_churn['last_num_customers'] -␣
 ↪df_churn['num_customers']
df_churn['churn_rate'] = df_churn['num_churned']/df_churn['last_num_customers']
df_churn.drop('last_num_customers',axis=1,inplace=True)
df_churn.head()
```

[90]:

	month_year	num_customers	num_churned	churn_rate
0	2016-08	1393	NaN	NaN
1	2016-09	1205	188.0	0.1349605169
2	2016-10	812	393.0	0.3261410788
3	2016-11	2302	-1490.0	-1.8349753695
4	2016-12	718	1584.0	0.6880973067

[91]:
```
fig, ax = plt.subplots(figsize=(10, 5))
ax = sns.barplot(df_churn.month_year, df_churn.churn_rate)
plt.title('Churn rate over month', size=20)
ax.tick_params(axis="x", rotation=45)
ax.bar_label(ax.containers[0], fmt='%.2f')
plt.ylabel('churn rate', size=15)
plt.xlabel('YYYY-MM', size=15)
plt.show()
```



7.3 3) Retention Rate

[92]:
```
df_retention=orders
df_retention['order_month'] = df_retention['order_created_at'].dt.to_period('M')
df_retention['cohort_month'] = df_retention.
↪groupby('customer_id')['order_month'].transform('min')
df_retention.head()
```

[92]:
```
     order_id order_created_at order_closed_at cancelled_at customer_id  \
0  7675398239       2016-08-21      2016-08-25   2016-08-22  8683754719
1  7676331935       2016-08-22      2016-08-22          NaN  8686224991
2  7676363167       2016-08-22             NaN   2016-08-22  8686224991
3  7676539359       2016-08-22      2016-08-22          NaN  8686915935
4  7676549855       2016-08-22      2016-08-22          NaN  8686924319

  financial_status fulfillment_status processed_at  total_price  \
0           voided                NaN   2016-08-21        44.57
1         refunded                NaN   2016-08-22       124.55
2           voided                NaN   2016-08-22        97.68
3             paid          fulfilled   2016-08-22       131.10
4             paid          fulfilled   2016-08-22        91.12

   shipping_rate  subtotal_price  total_discounts  total_line_items_price  \
0           6.33            35.0              0.0                    35.0
1           0.00           114.0              0.0                   114.0
```

```
2           7.00          83.0          0.0                   83.0
3           0.00         120.0          0.0                  120.0
4           7.00          77.0          0.0                   77.0

  order_month cohort_month
0     2016-08      2016-08
1     2016-08      2016-08
2     2016-08      2016-08
3     2016-08      2016-08
4     2016-08      2016-08
```

[93]:
```
df_grouped = df_retention.groupby(['cohort_month','order_month'])
```

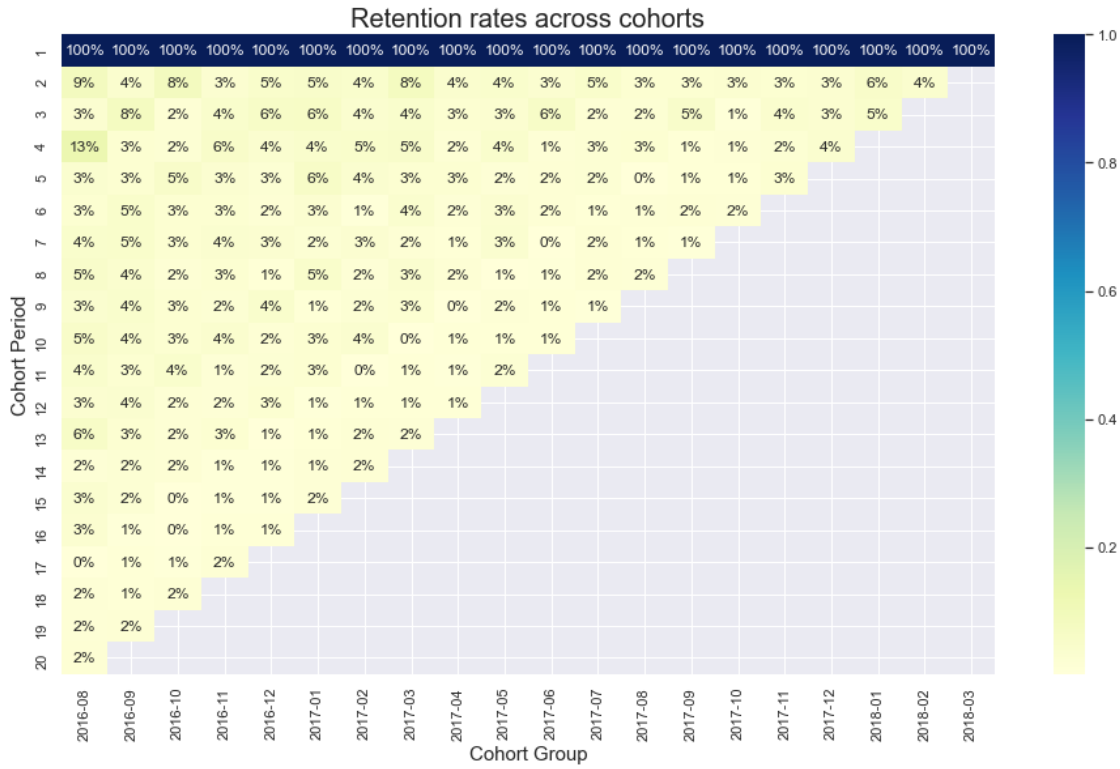
[94]:
```
df_cohorts = df_grouped.agg({'customer_id': pd.Series.nunique,
                      'order_id': pd.Series.nunique})
df_cohorts.rename(columns={'customer_id':'total_customers',
                       'order_id':'total_orders'}, inplace=True)
```

[95]:
```
def cohort_period(df):
    df['cohort_period'] = np.arange(len(df)) + 1
    return df

df_cohorts = df_cohorts.groupby(level=0).apply(cohort_period)
df_cohorts.head()
```

[95]:
```
                          total_customers  total_orders  cohort_period
cohort_month order_month
2016-08      2016-08                 1433          1554              1
             2016-09                  123           155              2
             2016-10                   45            48              3
             2016-11                  188           224              4
             2016-12                   50            52              5
```

[96]:
```
df_cohorts.reset_index(inplace=True)
df_cohorts.set_index(['cohort_month', 'cohort_period'], inplace=True)

cohort_sizes = df_cohorts.groupby(level=0)['total_customers'].first()

retention = df_cohorts['total_customers'].unstack(0).divide(cohort_sizes, axis␣
 ↪= 1)
plt.figure(figsize=(16,9))
ax = sns.heatmap(retention, annot=True,cmap="YlGnBu", fmt='.0%')

ax.set_ylabel('Cohort Period', fontsize = 15)
ax.set_xlabel('Cohort Group', fontsize = 15)

ax.set_title('Retention rates across cohorts', fontsize = 20)
```

```
plt.show()
```



7.4 4) RFM analysis

[97]:
```
last_date=orders['order_created_at'].max()
last_date
```

[97]: Timestamp('2018-03-22 00:00:00')

[98]:
```
rfm = orders.groupby('customer_id').agg({'order_created_at': lambda x:␣
↪(last_date - x.max()).days,
                                         'order_id': lambda x: len(x),
                                         'total_price': lambda x: x.sum()})
```

[99]:
```
rfm.rename(columns={'order_created_at': 'recency',
                    'order_id': 'frequency',
                    'total_price': 'monetary'}, inplace=True)
```

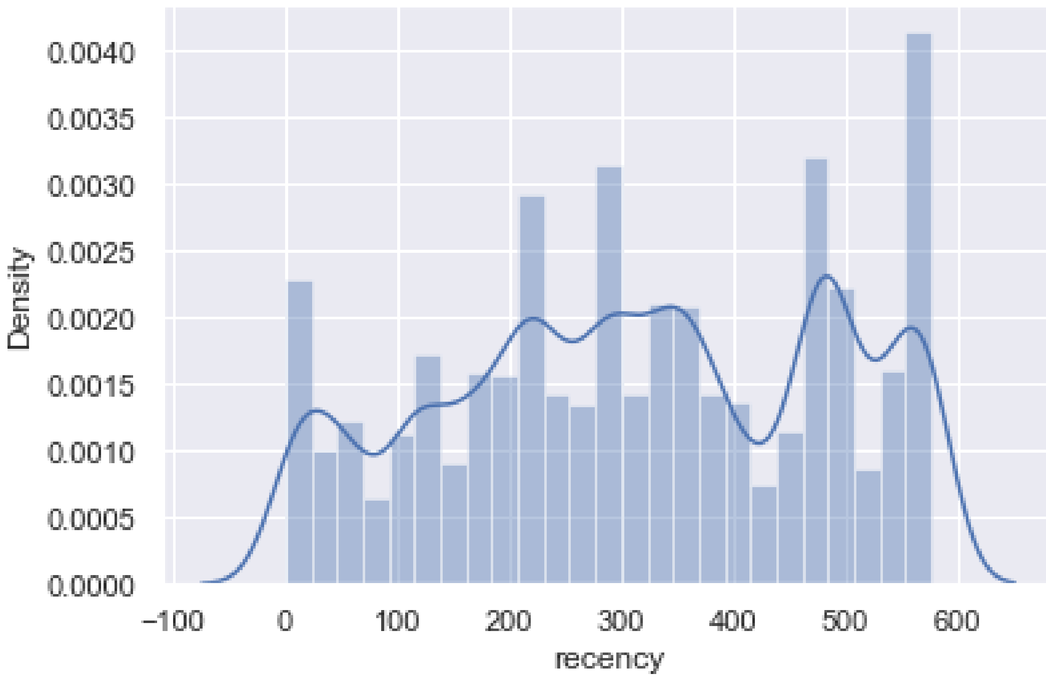
[100]:
```
rfm = rfm.reset_index()
rfm.head()
```

[100]:

	customer_id	recency	frequency	monetary
0	8683754719	357	10	875.80
1	8686224991	415	10	286.33
2	8686913503	293	3	140.28
3	8686915935	577	1	131.10
4	8686924319	577	1	91.12

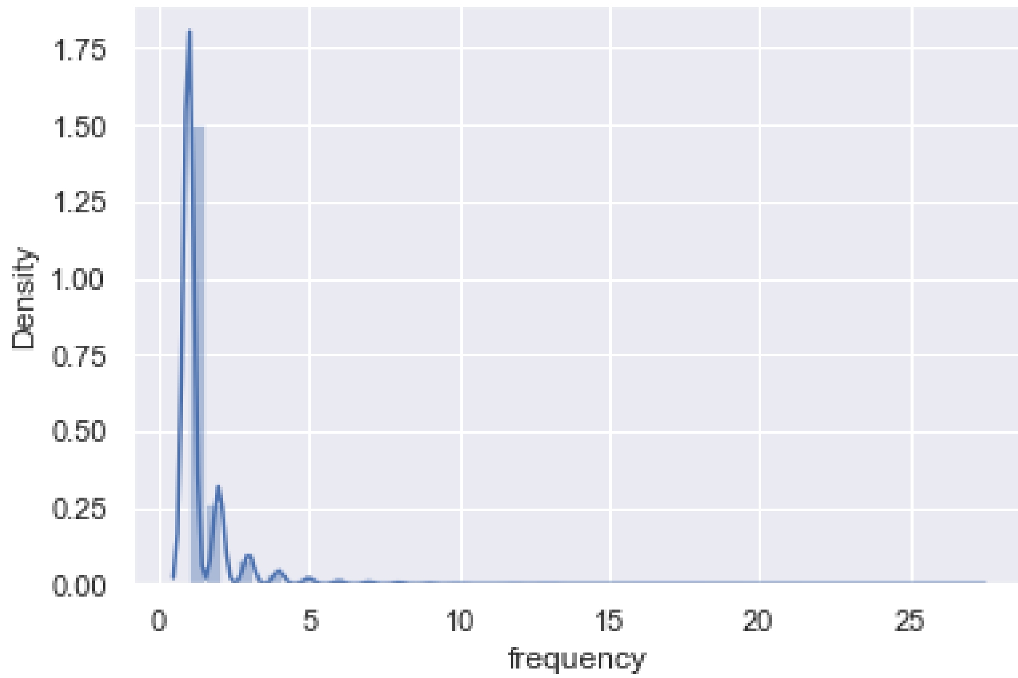
[101]:

```
# recency distribution plot
sns.distplot(rfm['recency'])
plt.show()
```

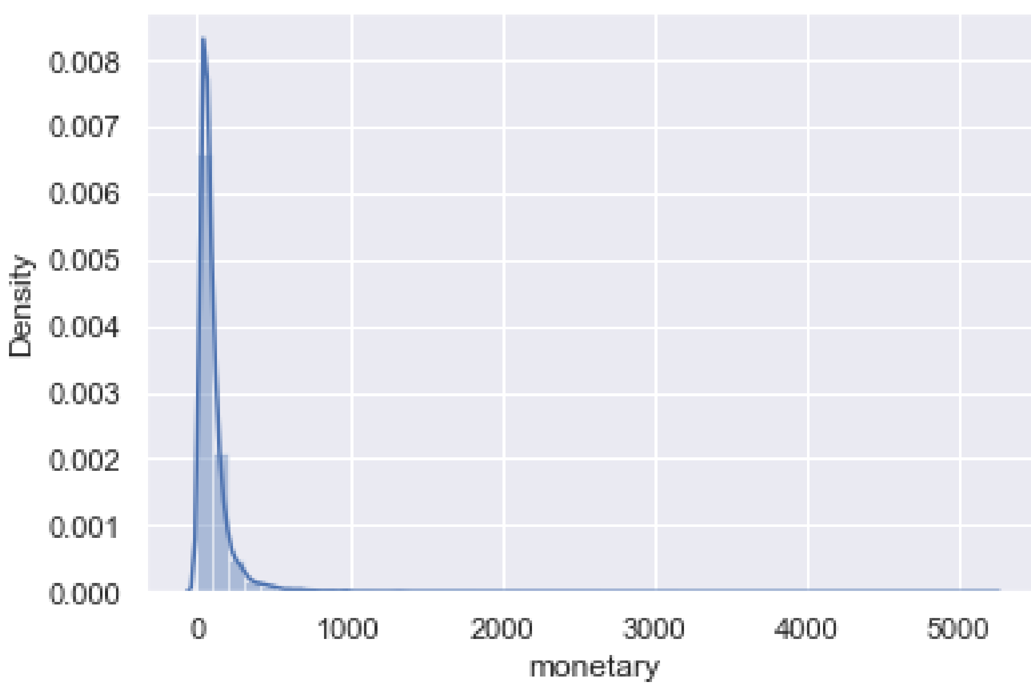


[102]:

```
# frequency distribution plot
sns.distplot(rfm['frequency'])
plt.show()
```



[103]:
```python
# monetary distribution plot
sns.distplot(rfm['monetary'])
plt.show()
```



[104]:

```python
# split into four segments using quantiles
quantiles = rfm.quantile(q=[0.25,0.5,0.75])
quantiles = quantiles.to_dict()
quantiles
```

[104]:

```
{'customer_id': {0.25: 8847997423.0,
  0.5: 394172863997.0,
  0.75: 611449583101.0},
 'recency': {0.25: 189.0, 0.5: 319.0, 0.75: 480.0},
 'frequency': {0.25: 1.0, 0.5: 1.0, 0.75: 1.0},
 'monetary': {0.25: 43.79, 0.5: 69.305, 0.75: 117.9}}
```

[131]:

```python
def Rscore(x, p, d):
    if x <= d[p][0.25]:
        return 4
    elif x <= d[p][0.50]:
        return 3
    elif x <= d[p][0.75]:
        return 2
    else:
        return 1

def Fscore(x, p, d):
    if x <= d[p][0.25]:
        return 1
    elif x <= d[p][0.50]:
        return 2
    elif x <= d[p][0.75]:
        return 3
    else:
        return 4

def Mscore(x, p, d):
    if x <= d[p][0.25]:
        return 1
    elif x <= d[p][0.50]:
        return 2
    elif x <= d[p][0.75]:
        return 3
    else:
        return 4
```

[133]:

```python
rfm['R'] = rfm['recency'].apply(Rscore, args=('recency', quantiles))
rfm['F'] = rfm['frequency'].apply(Fscore, args=('frequency', quantiles))
rfm['M'] = rfm['monetary'].apply(Mscore, args=('monetary', quantiles))
rfm.head()
```

[133]:

```
   customer_id  recency  frequency  monetary  R  F  M RFMGroup  RFMScore  \
0  8683754719      357         10    875.80  2  4  4      311         5
1  8686224991      415         10    286.33  2  4  4      311         5
2  8686913503      293          3    140.28  3  4  4      211         4
3  8686915935      577          1    131.10  1  1  4      441         9
4  8686924319      577          1     91.12  1  1  3      442        10

  RFM_Loyalty_Level  Cluster
0          Platinum        4
1          Platinum        4
2          Platinum        2
3              Gold        1
4            Silver        1
```

[134]:

```python
# calculate and Add RFMGroup value column showing combined concatenated score␣
↪of RFM
rfm['RFMGroup'] = rfm.R.map(str) + rfm.F.map(str) + rfm.M.map(str)

# calculate and Add RFMScore value column showing total sum of RFMGroup values
rfm['RFMScore'] = rfm[['R', 'F', 'M']].sum(axis = 1)
rfm.head()
```

[134]:

```
   customer_id  recency  frequency  monetary  R  F  M RFMGroup  RFMScore  \
0  8683754719      357         10    875.80  2  4  4      244        10
1  8686224991      415         10    286.33  2  4  4      244        10
2  8686913503      293          3    140.28  3  4  4      344        11
3  8686915935      577          1    131.10  1  1  4      114         6
4  8686924319      577          1     91.12  1  1  3      113         5

  RFM_Loyalty_Level  Cluster
0          Platinum        4
1          Platinum        4
2          Platinum        2
3              Gold        1
4            Silver        1
```

[146]:

```python
# assign Loyalty Level to each customer
Loyalty_Level = ['Bronze', 'Silver', 'Gold', 'Platinum']
Score_cuts = pd.qcut(rfm.RFMScore, q = 4, labels = Loyalty_Level)
rfm['RFM_Loyalty_Level'] = Score_cuts.values
rfm.reset_index().head()
```

[146]:

```
   index  customer_id  recency  frequency  monetary  R  F  M RFMGroup  \
0      0  8683754719      357         10    875.80  2  4  4      244
1      1  8686224991      415         10    286.33  2  4  4      244
2      2  8686913503      293          3    140.28  3  4  4      344
3      3  8686915935      577          1    131.10  1  1  4      114
```

```
4      4   8686924319      577          1      91.12  1   1   3       113

   RFMScore RFM_Loyalty_Level  Cluster
0        10          Platinum        4
1        10          Platinum        4
2        11          Platinum        3
3         6            Silver        0
4         5            Bronze        0
```

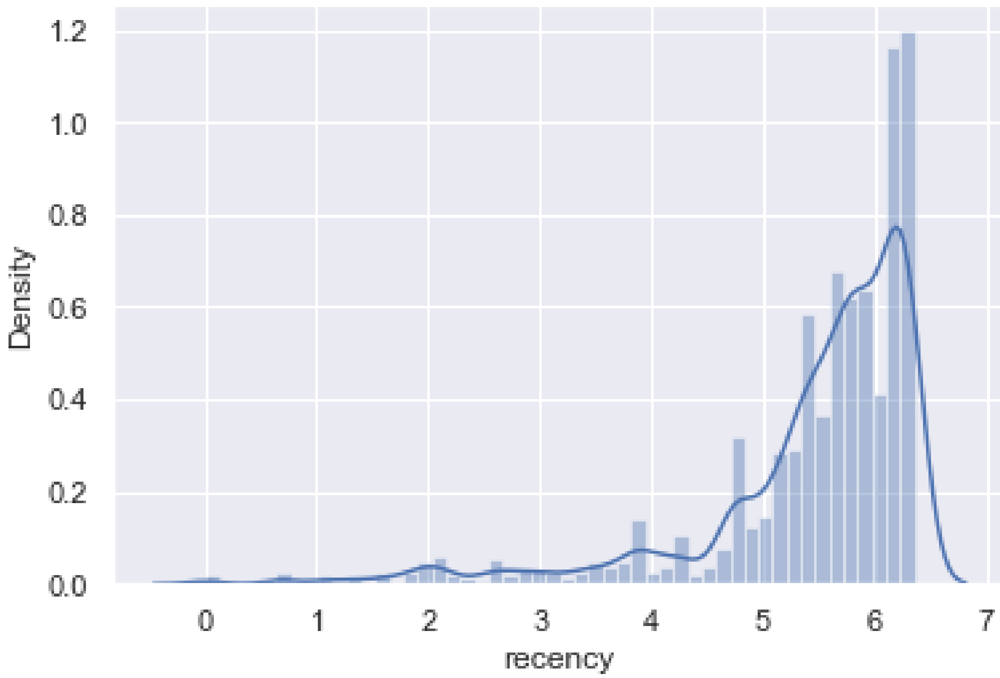
[147]:
```
# handle negative and zero values so as to handle infinite numbers during log␣
↪transformation
def handle_neg_n_zero(num):
    if num <= 0:
        return 1
    else:
        return num
# apply handle_neg_n_zero function to Recency and Monetary columns
rfm['recency'] = [handle_neg_n_zero(x) for x in rfm.recency]
rfm['monetary'] = [handle_neg_n_zero(x) for x in rfm.monetary]

# perform Log transformation to bring data into normal or near normal␣
↪distribution
Log_Tfd_Data = rfm[['recency', 'frequency', 'monetary']].apply(np.log, axis =␣
↪1).round(3)
Log_Tfd_Data
```

[147]:

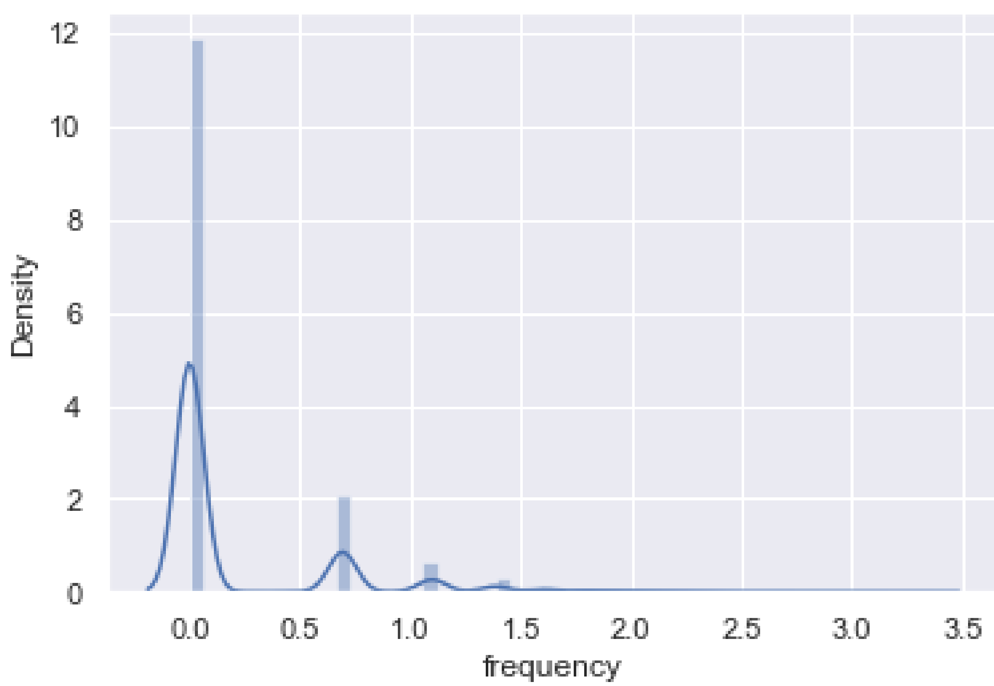
	recency	frequency	monetary
0	5.878	2.303	6.775
1	6.028	2.303	5.657
2	5.680	1.099	4.944
3	6.358	0.000	4.876
4	6.358	0.000	4.512
…	…	…	…
14929	0.000	0.000	5.076
14930	0.000	0.000	4.042
14931	0.000	0.000	4.591
14932	0.000	0.000	4.605
14933	0.000	0.000	5.244

[14934 rows x 3 columns]

[148]:
```
# Recency distribution plot after log
sns.distplot(Log_Tfd_Data.recency)
plt.show()
```

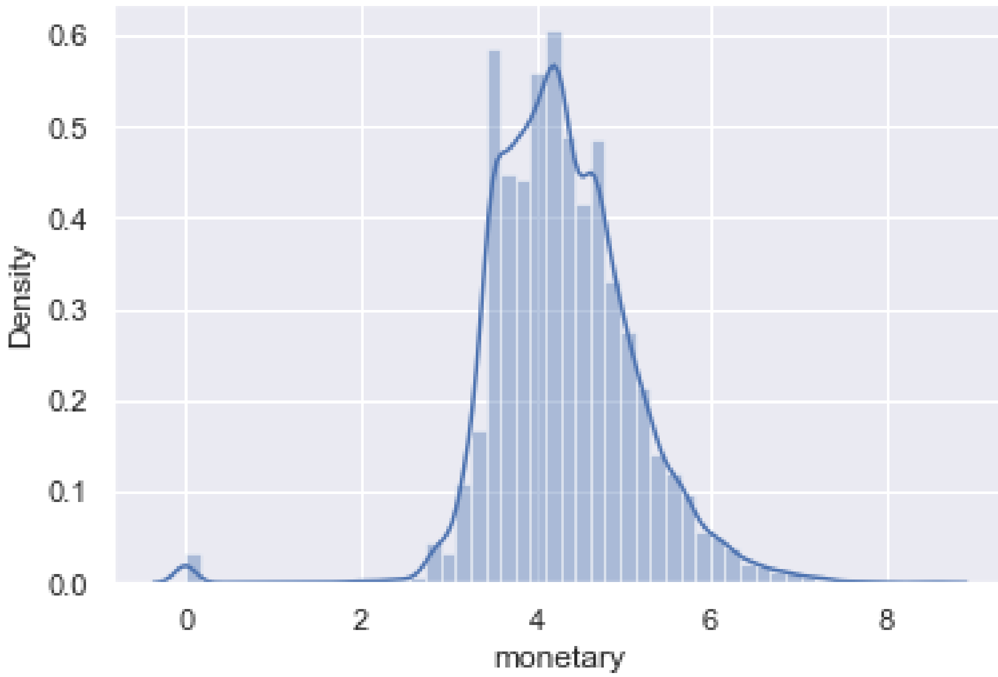


[149]:
```
# Frequency distribution plot after log
sns.distplot(Log_Tfd_Data.frequency)
plt.show()
```



[150]:

```
# Monetary distribution plot after log
sns.distplot(Log_Tfd_Data.monetary)
plt.show()
```



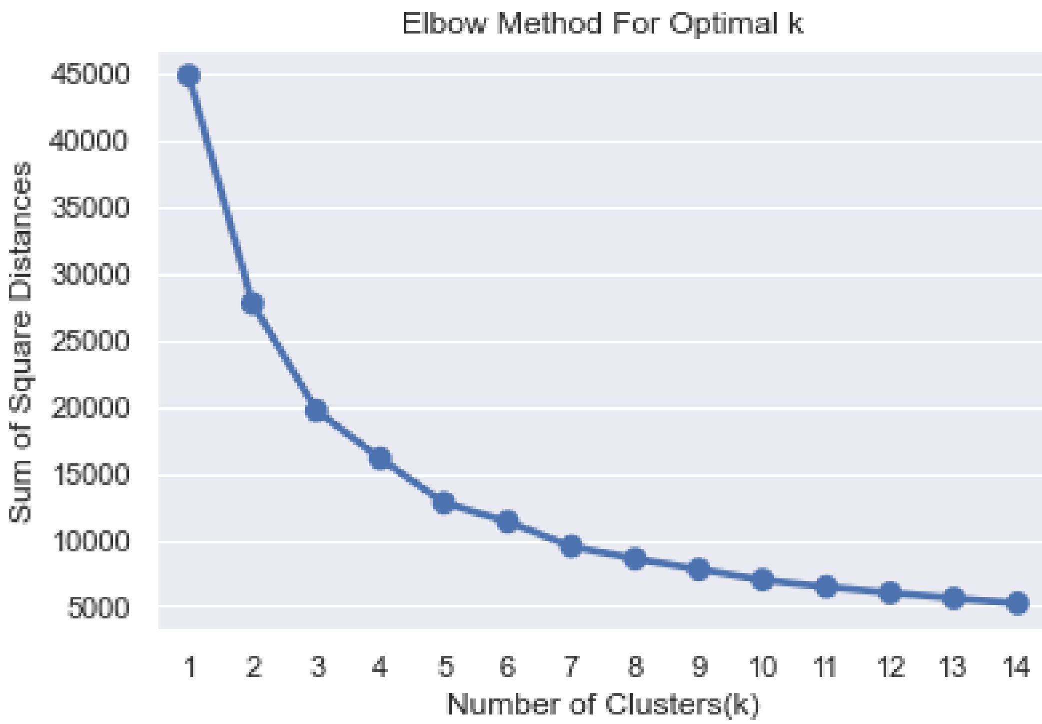
[151]:

```
# bring the data on same scale
scaleobj = StandardScaler()
Scaled_Data = scaleobj.fit_transform(Log_Tfd_Data)

# transform it back to dataframe
Scaled_Data = pd.DataFrame(Scaled_Data, index = rfm.index, columns =␣
↪Log_Tfd_Data.columns)
```

[152]:

```
# Elbow

sum_of_sq_dist = {}
for k in range(1,15):
    km = KMeans(n_clusters= k, init= 'k-means++', max_iter= 1000)
    km = km.fit(Scaled_Data)
    sum_of_sq_dist[k] = km.inertia_

#Plot the graph for the sum of square distance values and Number of Clusters
sns.pointplot(x = list(sum_of_sq_dist.keys()), y = list(sum_of_sq_dist.
↪values()))
```

```python
plt.xlabel('Number of Clusters(k)')
plt.ylabel('Sum of Square Distances')
plt.title('Elbow Method For Optimal k')
plt.show()
```

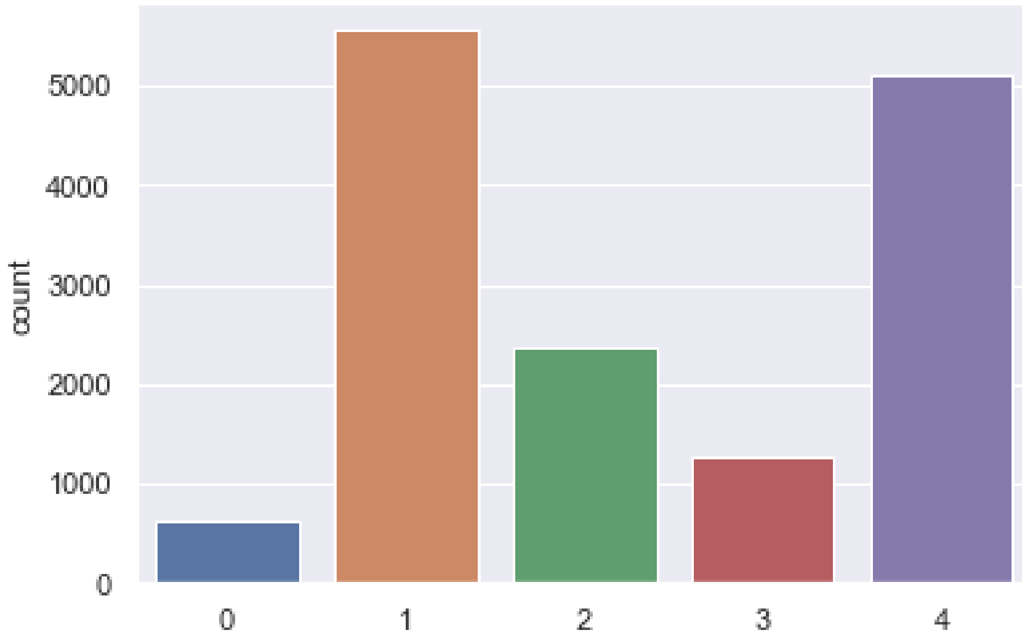


[153]:
```python
for key, val in sum_of_sq_dist.items():
    print(f'{key}: {val}')
```

```
1: 44802.000000000015
2: 27710.581270128132
3: 19670.599048050994
4: 16074.41905942356
5: 12751.585325634707
6: 11341.441634805527
7: 9482.964701950525
8: 8556.819396849332
9: 7765.321651173161
10: 6975.117309546858
11: 6456.253380518115
12: 6002.7849319401
13: 5594.351671833778
14: 5224.848408540871
```

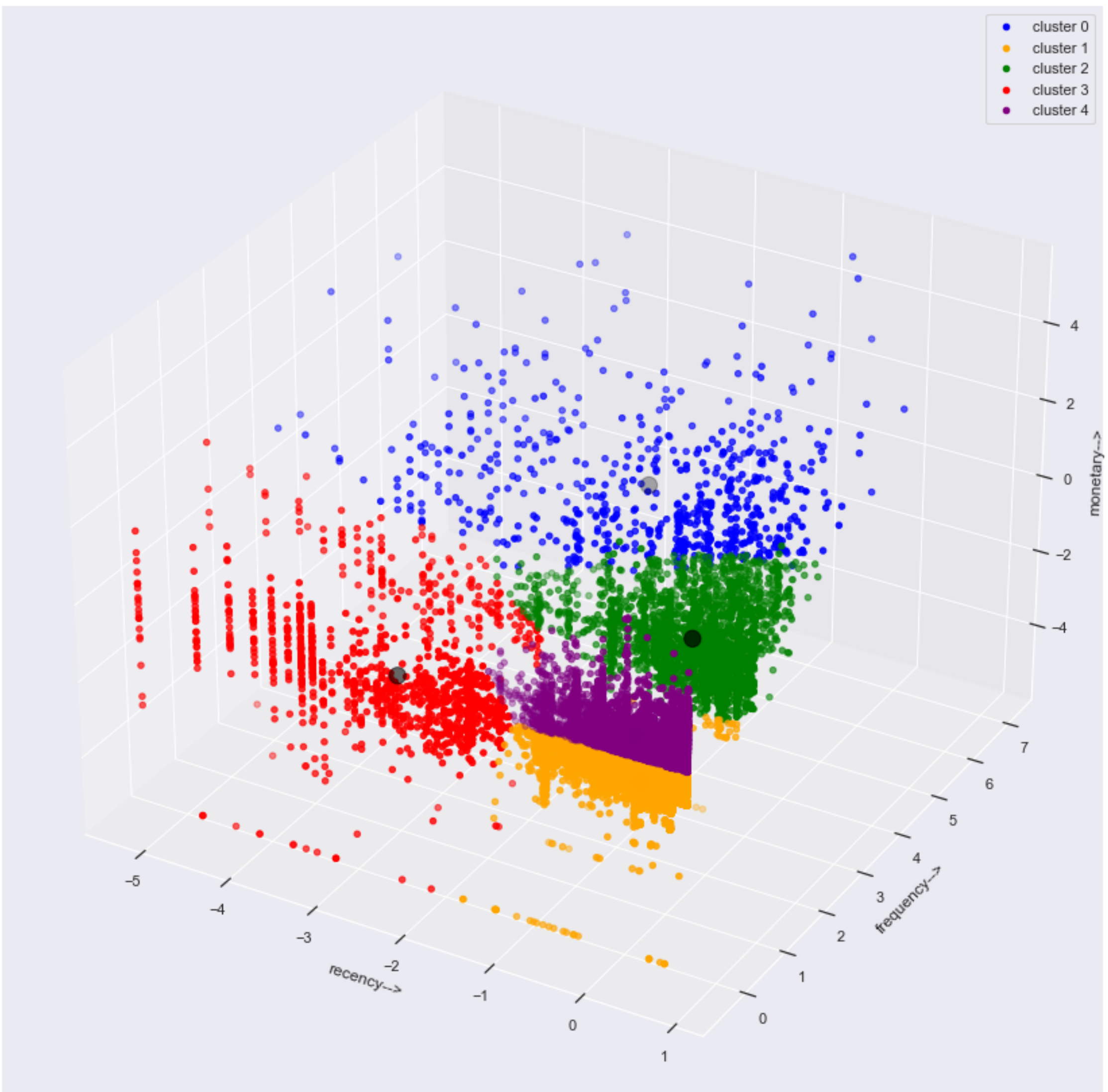
[154]:
```
# build the K-Means clustering model
kmeans = KMeans(n_clusters= 5, init= 'k-means++', max_iter= 1000)
kmeans.fit(Scaled_Data)
labels=kmeans.predict(Scaled_Data)
centroids=kmeans.cluster_centers_
sns.countplot(labels)
```

[154]: <Axes: ylabel='count'>



[155]:
```
fig=plt.figure(figsize=(15,15))
ax=fig.add_subplot(111, projection='3d')


ax.scatter(centroids[:,0],centroids[:,1],centroids[:,2],s=150,c='black')
ax.scatter(Scaled_Data.values[labels==0,0],Scaled_Data.
 ↪values[labels==0,1],Scaled_Data.values[labels==0,2],s=20,␣
 ↪color='blue',label='cluster 0')
ax.scatter(Scaled_Data.values[labels==1,0],Scaled_Data.
 ↪values[labels==1,1],Scaled_Data.values[labels==1,2],s=20,␣
 ↪color='orange',label='cluster 1')
ax.scatter(Scaled_Data.values[labels==2,0],Scaled_Data.
 ↪values[labels==2,1],Scaled_Data.values[labels==2,2],s=20,␣
 ↪color='green',label='cluster 2')
```

```
ax.scatter(Scaled_Data.values[labels==3,0],Scaled_Data.
 ↪values[labels==3,1],Scaled_Data.values[labels==3,2],s=20,␣
 ↪color='red',label='cluster 3')
ax.scatter(Scaled_Data.values[labels==4,0],Scaled_Data.
 ↪values[labels==4,1],Scaled_Data.values[labels==4,2],s=20,␣
 ↪color='purple',label='cluster 4')
ax.set_xlabel('recency-->')
ax.set_ylabel('frequency-->')
ax.set_zlabel('monetary-->')
ax.legend()
plt.show()
```



[156]:
```
rfm['Cluster'] = kmeans.labels_
rfm.head(20)
```

[156]:

	customer_id	recency	frequency	monetary	R	F	M	RFMGroup	RFMScore	\
0	8683754719	357	10	875.80	2	4	4	244	10	
1	8686224991	415	10	286.33	2	4	4	244	10	
2	8686913503	293	3	140.28	3	4	4	344	11	
3	8686915935	577	1	131.10	1	1	4	114	6	
4	8686924319	577	1	91.12	1	1	3	113	5	
5	8687041311	577	1	75.00	1	1	3	113	5	
6	8687102111	294	1	157.68	3	1	4	314	8	
7	8687175327	342	2	191.03	2	4	4	244	10	
8	8687301023	169	3	295.64	4	4	4	444	12	
9	8687317279	577	1	94.40	1	1	3	113	5	
10	8687317407	189	2	112.49	4	4	3	443	11	
11	8687323487	577	1	95.93	1	1	3	113	5	
12	8687329311	49	2	155.32	4	4	4	444	12	
13	8687334751	577	1	168.00	1	1	4	114	6	
14	8687338847	577	1	56.16	1	1	2	112	4	
15	8687346591	32	6	377.30	4	4	4	444	12	
16	8687351327	577	1	34.31	1	1	1	111	3	
17	8687357279	577	1	166.06	1	1	4	114	6	
18	8687362591	491	2	139.01	1	4	4	144	9	
19	8687377375	480	1	18.31	2	1	1	211	4	

	RFM_Loyalty_Level	Cluster
0	Platinum	0
1	Platinum	0
2	Platinum	2
3	Silver	4
4	Bronze	4
5	Bronze	4
6	Gold	4
7	Platinum	2
8	Platinum	2
9	Bronze	4
10	Platinum	2
11	Bronze	4
12	Platinum	2
13	Silver	4
14	Bronze	1
15	Platinum	0
16	Bronze	1
17	Silver	4
18	Platinum	2
19	Bronze	1

7.4.1 Summary of Clusters

Cluster 0 - Target Customers This group of customers has high frequency and monetary. We should reward these customers and try to keep the frequency of their purchases and purchase monetary.

Cluster 4 - Potential customers This group of customers has high monetary orders recently but not frequently. We should try to increase their order frequency, such as email them about new stuffs and rewards.

Cluster 2 - Stable Customer This group of customers has high frequency but low monetary. We should keep the frequency of their purchases and increase their purchase monetary.

Cluster 1 - Need Activation This group of customers has low frequency and monetary but still has orders rencently. We should eamil them about new deals to increase their frequency and moneary.

Cluster 3 - At Risk Customer This group of customers has high monetary but not purchase recently. We should do some action to avoid losing them.

[ ]:

```

```