Ecommerce Recommendation Engine

March 29, 2022

0.0.1 type0_recommend:

Function: Recommends most purchased items across users

Notes: If ratings variable available, this can be used in a weighted rating formula for recommendations

0.0.2 type1a_recommend:

Function: Takes in a user who has purchased one item and recommends the second most purchased item across users who have also purchased the item that the user has purchased.

Notes: This method can scale to more than one item and type2a_recommend is the scaled version to 2 purchases.

0.0.3 type1b_recommend:

Function: Takes in a user who has purchased one item and recommends other items in the same category (with the same brand).

Notes: This method isn't very strong as each brand and category have many products. This method is a watered down version of applying NLP and similarity metrics. This dataset is too big for calculating word vector distances - I believe I need to look into applying big data frameworks here such as pytorch or apache spark. Calculating word vector distances can scale to many purchases, I just need to figure out how to apply the appropriate big data frameworks to make the calculations when the dataset is this large.

0.0.4 type1c_recommend:

Function: Jaccard Score

Notes: Very similar to type3_recommendation, but uses jaccard score instead. Not ideal because it requires a large calculation which takes 17 seconds and I don't like how long it takes.

0.0.5 type2_recommend:

Function: Takes in a user who has purchased 2 or more items and recommends the next most popular purchased item across users.

Notes: This method is just a scaled version of type1a_recommend.

0.0.6 type3_recommend:

Function: Takes in a user who has purchased any # of items (the more the better) and recommends the items with descriptions that most closely match the descriptions of the user's previous purchases. The more purchases the better, and maybe a rolling window could be of use here.

Notes:

0.0.7 type4_recommend:

Function: KNN

Notes: With user demographic data, we can apply clustering as well. (This dataset doesn't have user demographic data)

[1]:

```
import pandas as pd
import numpy as np

# https://www.kaggle.com/hariwh0/userbehavior-ecommerce
df = pd.read_csv('preprocessed_data.csv')

df = df[['user_id', 'product_id', 'category_code','subcategory','brand',␣
 ↪'price', 'is_purchased', 'event_time']]

df['event_time'] = pd.to_datetime(df['event_time'],format="%Y-%m-%d %H:%M:%S␣
 ↪%Z")
df['category_code'] = df['subcategory'] + ' ' + df['brand']

df = df[df['is_purchased'] == 1]
```

[2]:

```
def type0_recommend(df,user_id):
    return df[df['is_purchased'] == 1][['user_id','product_id']].
 ↪drop_duplicates()['product_id'].value_counts(ascending=False)[:10].index()
```

[3]:

```
def type1a_recommend(df,user_id):
    df = df[df['is_purchased'] == 1]

    user_purchases = df[df['user_id'] == user_id]
    most_recent_purchase = user_purchases[user_purchases['event_time'] ==␣
 ↪user_purchases['event_time'].max()]['product_id'].values

    df_where_product_exists = df[df['product_id'].isin(most_recent_purchase)]
    df_of_users_who_also_purchased_product = df[df['user_id'].
 ↪isin(df_where_product_exists['user_id'])]

    return df_of_users_who_also_purchased_product[['user_id','product_id']].
 ↪drop_duplicates()['product_id'].value_counts(ascending=False)[:10].index
```

[4]:

```
def type1b_recommend(df,user_id):

    user_purchases = df[df['user_id'] == user_id]
    most_recent_purchase = user_purchases[user_purchases['event_time'] ==␣
 ↪user_purchases['event_time'].max()]['category_code'].values

    return df[df['category_code'].isin(most_recent_purchase)]['product_id']
```

[5]:

```
%%time

# 17 seconds with geforce rtx 2060 super

def type1c_recommend(df,user_id):
    from sklearn.metrics import jaccard_score
    from scipy.spatial.distance import pdist, squareform
    from sklearn.feature_extraction.text import CountVectorizer

    product_category = df[['product_id','category_code']].
 ↪drop_duplicates('product_id')

    count = CountVectorizer(stop_words='english')
    vectorized_data = count.fit_transform(product_category['category_code'])

    features = count.get_feature_names()
    word_vectors = vectorized_data.toarray()
    count_df = pd.DataFrame(word_vectors,columns=features)
    count_df.index = product_category['product_id']

    user_purchases = df[df['user_id'] == user_id]
    most_recent_purchase = user_purchases[user_purchases['event_time'] ==␣
 ↪user_purchases['event_time'].max()]['product_id'].values

    user_products = count_df.reindex(most_recent_purchase)
    user_profile = user_products.mean()

    jaccard_distances = 1 - squareform(pdist(count_df.values, metric='jaccard'))
    results = pd.DataFrame(jaccard_distances,
                           index=count_df.index,
                           columns=count_df.index)

    return results[most_recent_purchase][results[most_recent_purchase].values !
 ↪= 0].sort_values(most_recent_purchase[0],ascending=False)
```

```
Wall time: 0 ns
```

[6]:

```
def type2_recommend(df,user_id):
    df = df[df['is_purchased'] == 1]

    user_purchases = df[df['user_id'] == user_id]
    most_recent_purchases = user_purchases[(user_purchases['event_time'].
↪isin([user_purchases['event_time'].shift(1).
↪max(),user_purchases['event_time'].max()]))]['product_id'].values

    df_where_product_exists = df[df['product_id'].isin(most_recent_purchases)]
    df_of_users_who_also_purchased_product = df[df['user_id'].
↪isin(df_where_product_exists['user_id'])]

    return df_of_users_who_also_purchased_product[['user_id','product_id']].
↪drop_duplicates()['product_id'].value_counts(ascending=False)[:10].index
```

[7]:

```
def type3_recommend(df,user_id):

    from sklearn.feature_extraction.text import CountVectorizer
    from sklearn.metrics.pairwise import cosine_similarity

    product_category = df[['product_id','category_code']].
↪drop_duplicates('product_id')

    count = CountVectorizer(stop_words='english')
    vectorized_data = count.fit_transform(product_category['category_code'])

    features = count.get_feature_names()
    word_vectors = vectorized_data.toarray()
    count_df = pd.DataFrame(word_vectors,columns=features)
    count_df.index = product_category['product_id']

    user_purchases = df[df['user_id'] == user_id]
    most_recent_purchase = user_purchases['product_id']

    user_products = count_df.reindex(most_recent_purchase)
    user_profile = user_products.mean()

    cosine_sim = cosine_similarity(user_profile.values.reshape(1,-1), count_df).
↪T
    results = pd.DataFrame(cosine_sim, index=count_df.index, columns =␣
↪['similarity_score']).sort_values('similarity_score',ascending=False)

    return results[results['similarity_score'] != 0].index
```

[8]:

```
def get_user_type(user_id):
    return df[df['user_id'] == user_id]['is_purchased'].sum()
```

[9]:

```
def recommend(df,user_id):
    try:
        user_type = get_user_type(user_id)
    except:
        print('Please enter a pandas dataframe and valid user id.')
    else:
        if user_type == 0:
            recommendation = type0_recommend(df,user_id)
        elif user_type == 1:
            try:
                recommendation = type1a_recommend(df,user_id)
            except:
                try:
                    recommendation = type1b_recommend(df,user_id)
                except:
                    try:
                        recommendation = type1c_recommend(df,user_id)
                    except:
                        recommendation = type0_recommend(df)
        elif user_type == 2:
            recommendation = type2_recommend(df,user_id)

        elif user_type >= 3:
            recommendation = type3_recommend(df,user_id)
        else:
            recommendation = type0recommend(df,user_id)
    return user_type, recommendation
```

[12]:

```
%%time
recommendations = []
for user in pd.Series(df['user_id'].unique()).sample(100).values:
    recommendations.append(recommend(df,user))
recommendations
```

```
Wall time: 3.24 s
```

[12]:

```
[(3.0,
  Int64Index([3601084, 3601535, 3600661, 3600666, 3601525, 3601461, 3601527,
              3600835, 3601343, 3601505,
              …
              1200620, 1201505, 1201504, 1200617, 1201471, 1701343, 1004665,
              1700954, 1700872, 3601395],
             dtype='int64', name='product_id', length=501)),
 (1.0,
  Int64Index([1004858, 1004856, 1004833, 1004767, 1004857, 1005098, 1005100,
              1004836, 1004863, 1004209],
             dtype='int64')),
 (1.0,
 Int64Index([1004249, 1005115, 1002544, 1003306, 1002524, 1004767, 1004258,
              1004246, 1004856, 4804056],
             dtype='int64')),
(1.0,
 Int64Index([1004781, 1004785, 1004767, 1004870, 1004856, 1004565, 1004833,
              1002544, 1004873, 1004836],
             dtype='int64')),
(1.0,
 Int64Index([1004856, 1004833, 1004767, 1004870, 1004858, 1005100, 1004836,
              1004750, 1002544, 1004209],
             dtype='int64')),
(1.0,
 Int64Index([1004723, 1004856, 1004720, 1004833, 1004708, 1004748, 1004709,
              1005100, 1004750, 1004566],
             dtype='int64')),
(1.0,
 Int64Index([1005171, 1004856, 1004833, 1004767, 1004750, 1004870, 1004836,
              1005100, 1005098, 1005099],
             dtype='int64')),
(1.0,
 Int64Index([3700748, 1004856, 3700926, 1004767, 3701134, 3700766, 3701088,
              1004708, 1005115, 1002544],
             dtype='int64')),
(2.0,
 Int64Index([1004566, 1004565, 1004856, 1004833, 1004767, 1004708, 1004785,
              1004838, 1005100, 1004741],
             dtype='int64')),
(1.0,
 Int64Index([1801766, 1004856, 1801690, 1004833, 1801739, 1801805, 1004767,
              1801806, 1004870, 1002524],
             dtype='int64')),
(2.0,
 Int64Index([1002524, 1002633, 1002544, 1004767, 1004856, 1004249, 1003306,
              1005115, 1004833, 1004870],
             dtype='int64')),
(3.0,
 Int64Index([1004749, 1005182, 1004856, 1004659, 1004750, 1005099, 1004872,
              1005181, 1005174, 1003712,
              …
              1005123, 1005017, 1005113, 1004225, 1005141, 1005053, 1004174,
              1005054, 1004175, 1005097],
             dtype='int64', name='product_id', length=801)),
(1.0,
 Int64Index([1004870, 1004767, 1004856, 1004873, 1004833, 1004836, 1002544,
              1004750, 1004872, 1004768],
             dtype='int64')),
(1.0,
 Int64Index([1004767, 1004856, 1004870, 1004833, 1004836, 1004873, 1002544,
             1004750, 1004768, 1004249],
            dtype='int64')),
(1.0,
 Int64Index([1307310, 1004856, 1004767, 1004833, 1306650, 1307340, 1004838,
             1004836, 1002544, 1004870],
            dtype='int64')),
(1.0,
 Int64Index([4804056, 4804055, 1004249, 1005115, 1002544, 1004856, 1004767,
             1002524, 1005105, 1004833],
            dtype='int64')),
(1.0,
 Int64Index([1004227, 1005115, 1004237, 1004249, 1002544, 1003317, 1005105,
             1003306, 1004767, 1004856],
            dtype='int64')),
(1.0,
 Int64Index([1004836, 1004767, 1004856, 1004833, 1004870, 1004750, 1002544,
             1004873, 1004835, 1002633],
            dtype='int64')),
(1.0,
 Int64Index([6100130, 1004873, 1004870, 1004768, 1004785, 4700478, 4700419,
             3701088, 3700801, 3700926],
            dtype='int64')),
(1.0,
 Int64Index([3600182, 1004870, 3600263, 3600231, 1002544, 1004739, 1004856,
             1004767, 1801690, 1004833],
            dtype='int64')),
(1.0,
 Int64Index([1003312, 1002524, 1004249, 1005115, 1003310, 1002544, 1003306,
             1003317, 4804056, 1004767],
            dtype='int64')),
(4.0,
 Int64Index([1004749, 1005182, 1004856, 1004659, 1004750, 1005099, 1004872,
             1005181, 1005174, 1003712,
             …
             1005123, 1005017, 1005113, 1004225, 1005141, 1005053, 1004174,
             1005054, 1004175, 1005097],
            dtype='int64', name='product_id', length=801)),
(1.0,
 Int64Index([1004903, 1004856, 1004720, 1004565, 1004833, 1004209, 1005100,
             1004767, 1004904, 1005008],
            dtype='int64')),
(1.0,
 Int64Index([1004835, 1004836, 1004767, 1004856, 1004833, 1004750, 1004870,
             1005100, 1004768, 1002544],
            dtype='int64')),
(2.0,
 Int64Index([1005100, 1004856, 1004833, 1004767, 1005098, 1004870, 1004836,
             1004750, 1004857, 1004858],
            dtype='int64')),
(1.0,
 Int64Index([1005066, 1004741, 1005031, 1005160, 1004856, 1004767, 1004870,
             1005062, 1004785, 1004871],
            dtype='int64')),
(11.0,
 Int64Index([1002628, 1005230, 1004236, 1002634, 1005118, 1004362, 1005126,
             1003317, 1005121, 1005129,
             …
             1305570, 1306310, 5100567, 5100573, 5100346, 1004175, 1005054,
             1005053, 1005097, 1004174],
            dtype='int64', name='product_id', length=645)),
(2.0,
 Int64Index([1004961, 1004838, 1004839, 1004856, 1004767, 1004833, 1004836,
             1004858, 1004886, 1005205],
            dtype='int64')),
(1.0,
 Int64Index([1004813, 1004754, 1005160, 1004565, 1004767, 1002544, 1005161,
             1004856, 1004777, 3900002],
            dtype='int64')),
(1.0,
 Int64Index([1004767, 1004856, 1004870, 1004833, 1004836, 1004873, 1002544,
             1004750, 1004768, 1004249],
            dtype='int64')),
(7.0,
 Int64Index([1004361, 1003319, 1004229, 1003304, 1005132, 1004247, 1005230,
             1003318, 1005140, 1004362,
             …
             1307512, 1480527, 1306759, 1307162, 1201151, 2401198, 1306831,
             1306948, 1306607, 1306631],
            dtype='int64', name='product_id', length=1183)),
(4.0,
 Int64Index([1004225, 1002544, 1004258, 1005115, 1002528, 1002633, 1004250,
             1004245, 1004255, 1005104,
             …
             2701306, 4500859, 2401201, 4201416, 1201354, 2701289, 2701116,
             2401198, 2700984, 4201169],
            dtype='int64', name='product_id', length=996)),
(1.0,
 Int64Index([1003304, 1003306, 1002544, 1004249, 1003310, 1004856, 1004767,
             1005115, 1002524, 1004258],
            dtype='int64')),
(1.0,
 Int64Index([ 1004956,  1004957,  1004958,  1002544,  1005101,  1005102,
              1004767,  1005003,  1004739, 18500036],
            dtype='int64')),
(1.0,
 Int64Index([1004838, 1004856, 1004833, 1004839, 1004767, 1004961, 1004836,
             1004565, 1005100, 1004750],
            dtype='int64')),
(11.0,
 Int64Index([1003769, 1004545, 1004723, 1003879, 1005002, 1003982, 1004785,
             1003768, 1004321, 1005003,
             …
             5000530, 5000482, 5000568, 5000691, 5000565, 5000487, 5000418,
             5000477, 5000664, 5000015],
            dtype='int64', name='product_id', length=640)),
(2.0,
 Int64Index([1005098, 1004875, 1004856, 1004873, 1004767, 1005100, 1004870,
             1004833, 1004858, 1004836],
            dtype='int64')),
(9.0,
 Int64Index([1004655, 1005068, 1005179, 1005185, 1005186, 1005073, 1005071,
             1005173, 1004888, 1004663,
             …
             2702347, 1801762, 3601553, 1801853, 1801861, 1801847, 1801581,
             1800944, 1801350, 2702332],
            dtype='int64', name='product_id', length=1204)),
(2.0,
 Int64Index([1004856, 1004833, 1004767, 1004836, 1004870, 1005100, 1004750,
             1004858, 1002544, 1004209],
            dtype='int64')),
(3.0,
 Int64Index([1005168, 1004671, 1004944, 1004836, 1004653, 1004655, 1004666,
             1004209, 1005100, 1004529,
             …
             1306315, 1306310, 1304297, 1305570, 1307053, 1306312, 1306952,
             1307059, 1307052, 1305706],
            dtype='int64', name='product_id', length=916)),
(2.0,
 Int64Index([1004870, 1004768, 1004767, 1004856, 1004873, 1004833, 1004836,
             1002544, 1004750, 1004872],
            dtype='int64')),
(1.0,
 Int64Index([1004838, 1004856, 1004833, 1004839, 1004767, 1004961, 1004836,
             1004565, 1005100, 1004750],
            dtype='int64')),
(1.0,
 Int64Index([1201504, 1004856, 1004767, 1201465, 1200947, 1200957, 1004836,
             1201505, 1201466, 1004209],
            dtype='int64')),
(1.0,
 Int64Index([4600483, 4600551, 1004856, 1005203, 2500513, 3100640, 4600718,
             4600542, 4600559, 4600469],
            dtype='int64')),
(3.0,
 Int64Index([1004749, 1005182, 1004856, 1004659, 1004750, 1005099, 1004872,
             1005181, 1005174, 1003712,
             …
             1005123, 1005017, 1005113, 1004225, 1005141, 1005053, 1004174,
             1005054, 1004175, 1005097],
            dtype='int64', name='product_id', length=801)),
(1.0,
 Int64Index([1005004, 1004565, 1004785, 1005003, 1004833, 1004856, 1004566,
             1004767, 1004750, 1004741],
            dtype='int64')),
(1.0,
 Int64Index([1004026, 1004856, 1004720, 1004857, 1004767, 1004144, 1005160,
             1005064, 1004805, 1002544],
            dtype='int64')),
(5.0,
 Int64Index([1005113, 1002528, 1004241, 1004245, 1005139, 1004234, 1004237,
             1002542, 1005137, 1005133,
             …
             1004857, 1004767, 1004777, 1004768, 1005007, 1004174, 1005053,
             1005097, 1005054, 1004175],
            dtype='int64', name='product_id', length=645)),
(1.0,
 Int64Index([1004653, 1004659, 1004873, 1004856, 1004767, 1004249, 1004870,
             1002544, 1005186, 1005014],
            dtype='int64')),
(2.0,
 Int64Index([1004781, 1005161, 1004785, 1004767, 1004870, 1004856, 1005160,
             1004739, 1004565, 1004833],
            dtype='int64')),
(1.0,
 Int64Index([1801623, 1801690, 1004856, 1004767, 1004858, 1801881, 1004870,
             1004836, 1005100, 1004833],
            dtype='int64')),
(4.0,
 Int64Index([1005168, 1004671, 1004944, 1004836, 1004653, 1004655, 1004666,
             1004209, 1005100, 1004529,
             …
             1306315, 1306310, 1304297, 1305570, 1307053, 1306312, 1306952,
             1307059, 1307052, 1305706],
            dtype='int64', name='product_id', length=916)),
(1.0,
 Int64Index([1004856, 1004833, 1004767, 1004870, 1004858, 1005100, 1004836,
             1004750, 1002544, 1004209],
            dtype='int64')),
(1.0,
 Int64Index([1005100, 1004856, 1004833, 1004767, 1005098, 1004870, 1004836,
             1004750, 1004857, 1004858],
            dtype='int64')),
(1.0,
 Int64Index([8801085, 1004856, 1004838, 1004209, 1004833, 8800911, 8800313,
             1004767, 1004836, 1004788],
            dtype='int64')),
(1.0,
 Int64Index([4700589, 4802159, 1005105, 1005006, 3100491, 4700671, 4700590,
             4700387, 4700154, 4700419],
            dtype='int64')),
(1.0,
 Int64Index([1004739, 1004741, 1004856, 1004767, 1004833, 1005160, 1004565,
             1004836, 1004870, 1004777],
            dtype='int64')),
(1.0,
 Int64Index([1004856, 1004833, 1004767, 1004870, 1004858, 1005100, 1004836,
             1004750, 1002544, 1004209],
            dtype='int64')),
(1.0,
 Int64Index([1004836, 1004767, 1004856, 1004833, 1004870, 1004750, 1002544,
             1004873, 1004835, 1002633],
            dtype='int64')),
(1.0,
 Int64Index([1004210, 1004856, 1004209, 1004833, 1004767, 1005100, 1004858,
             1004857, 1004446, 1005008],
            dtype='int64')),
(1.0,
 Int64Index([1004836, 1004767, 1004856, 1004833, 1004870, 1004750, 1002544,
             1004873, 1004835, 1002633],
            dtype='int64')),
(1.0,
 Int64Index([1801691, 1801690, 1801881, 1801704, 1801623, 1801901, 1004856,
             1801849, 1004767, 1801882],
            dtype='int64')),
(3.0,
 Int64Index([1004856, 1004211, 1004650, 1005216, 1005073, 1004648, 1004659,
             1004749, 1004750, 1005015,
             …
             2701995, 6800906, 1801922, 2701707, 6800905, 1201512, 1201354,
             1201472, 2702028, 3700359],
            dtype='int64', name='product_id', length=801)),
(1.0, Int64Index([1307059, 1201466], dtype='int64')),
(2.0,
 Int64Index([1005112, 1005115, 1005104, 1004249, 1005105, 1005118, 1005121,
             1005109, 1004258, 1005113],
            dtype='int64')),
(1.0,
 Int64Index([1801881, 1801690, 1004856, 1004767, 1801882, 1801929, 1004833,
             1002544, 1005115, 1004870],
            dtype='int64')),
(1.0,
 Int64Index([1004957, 1004958, 1004767, 1004739, 1004741, 1005160, 1005102,
             1004836, 1005003, 1004856],
            dtype='int64')),
(2.0,
 Int64Index([3900815, 1801995, 1801873, 3900002, 3900749, 2800594, 2700071,
             3601537, 3601425, 3601423],
            dtype='int64')),
(1.0,
 Int64Index([5100562, 5100337, 4804056, 1004249, 4804055, 5100610, 1005115,
             1002544, 5100566, 1002633],
            dtype='int64')),
(2.0,
 Int64Index([1005100, 1004856, 1004833, 1004767, 1005098, 1004870, 1004836,
             1004750, 1004857, 1004858],
            dtype='int64')),
(1.0,
 Int64Index([1002544, 1002524, 1002633, 1004767, 1004856, 1004249, 1003306,
             1005115, 1004870, 1004833],
            dtype='int64')),
(1.0,
 Int64Index([1005105, 1005115, 1005135, 1004249, 1004856, 1005129, 1004767,
             1002544, 1005106, 1004237],
            dtype='int64')),
(1.0,
 Int64Index([1005238, 1005205, 1004839, 1004767, 1004856, 1004785, 1004838,
             1005252, 1005021, 1004873],
            dtype='int64')),
(2.0,
 Int64Index([1004767, 1004768, 1004856, 1004870, 1004833, 1004836, 1004873,
             1002544, 1004750, 1004249],
            dtype='int64')),
(3.0,
 Int64Index([1005190, 1005191, 1004940, 1004934, 1004937, 1004932, 1005069,
             1004931, 1004991, 1005070,
             …
             4804295, 5100674, 5100376, 5100551, 4803878, 5100328, 5100845,
             1306587, 5100377, 5100846],
            dtype='int64', name='product_id', length=607)),
(1.0,
 Int64Index([1004768, 1004767, 1004856, 1004870, 1004836, 1004833, 1004873,
             1004766, 1002544, 1004872],
            dtype='int64')),
(1.0,
 Int64Index([1801691, 1801690, 1801881, 1801704, 1801623, 1801901, 1004856,
             1801849, 1004767, 1801882],
            dtype='int64')),
(1.0,
 Int64Index([1004659, 1004873, 1004767, 1004856, 1004249, 1005115, 1004870,
             1002544, 1004665, 1005105],
            dtype='int64')),
(1.0,
 Int64Index([1004833, 1004856, 1004767, 1004836, 1004750, 1004870, 1005100,
             1004834, 1002544, 1004873],
            dtype='int64')),
(1.0,
 Int64Index([1004767, 1004856, 1004870, 1004833, 1004836, 1004873, 1002544,
             1004750, 1004768, 1004249],
            dtype='int64')),
(1.0,
 Int64Index([4000137, 1306659, 1004870, 4000169, 4000138, 3100970, 1200947,
             3200388, 3600661, 3601489],
            dtype='int64')),
(1.0,
 Int64Index([1002544, 1002524, 1002633, 1004767, 1004856, 1004249, 1003306,
             1005115, 1004870, 1004833],
            dtype='int64')),
(1.0,
 Int64Index([1005184, 1004665, 1004661, 1005014, 1005174, 1201044, 1201202,
             5100572, 1004670, 1005129],
            dtype='int64')),
(1.0,
 Int64Index([1201504, 1004856, 1004767, 1201465, 1200947, 1200957, 1004836,
             1201505, 1201466, 1004209],
            dtype='int64')),
(4.0,
 Int64Index([12300618, 12300231, 12300945, 12301394, 12300236, 12300510,
             12400024, 12300942, 12300928, 12300191,
             …
             19100217,  3700790,  3701088, 14500002, 53900007,  3700731,
              3700884,  3800157,  3701443,  3200513],
            dtype='int64', name='product_id', length=256)),
(3.0,
 Int64Index([1004749, 1005182, 1004856, 1004659, 1004750, 1005099, 1004872,
             1005181, 1005174, 1003712,
             …
             1005123, 1005017, 1005113, 1004225, 1005141, 1005053, 1004174,
             1005054, 1004175, 1005097],
            dtype='int64', name='product_id', length=801)),
(1.0,
 Int64Index([1004957, 1004958, 1004767, 1004739, 1004741, 1005160, 1005102,
             1004836, 1005003, 1004856],
            dtype='int64')),
(6.0,
 Int64Index([1005113, 1002528, 1004241, 1004245, 1005139, 1004234, 1004237,
             1002542, 1005137, 1005133,
             …
             1004857, 1004767, 1004777, 1004768, 1005007, 1004174, 1005053,
             1005097, 1005054, 1004175],
            dtype='int64', name='product_id', length=645)),
(1.0,
 Int64Index([1005006, 1004856, 1005008, 1004209, 1004833, 1004767, 1005009,
             1004210, 1005007, 1004903],
            dtype='int64')),
(1.0,
 Int64Index([1004856, 1004833, 1004767, 1004870, 1004858, 1005100, 1004836,
             1004750, 1002544, 1004209],
            dtype='int64')),
(9.0,
 Int64Index([1005123, 1005211, 1005215, 1004767, 1004650, 1005201, 1004670,
             1005209, 1004856, 1005135,
             …
             1304409, 1307058, 1307054, 1307051, 1306314, 1307059, 1307050,
             1306316, 1306315, 1306952],
            dtype='int64', name='product_id', length=916)),
(1.0,
 Int64Index([1004856, 1004833, 1004767, 1004870, 1004858, 1005100, 1004836,
             1004750, 1002544, 1004209],
            dtype='int64')),
(1.0,
 Int64Index([1801901, 1801849, 1801690, 1801881, 1801929, 1004767, 1801882,
             1801691, 1004870, 1002544],
            dtype='int64')),
(1.0,
 Int64Index([2701657, 1004856, 2701683, 2701639, 1004833, 1004767, 2701646,
             2700618, 1307310, 1004836],
            dtype='int64')),
(1.0,
 Int64Index([3600666, 3600661, 3600937, 3601244, 3600163, 3601250, 4804056,
             3601524, 1801690, 3601405],
            dtype='int64')),
(1.0,
 Int64Index([1004871, 1004870, 1004767, 1004856, 1004766, 1004873, 1004833,
             1004872, 1004768, 1004836],
            dtype='int64')),
(1.0,
 Int64Index([1004836, 1004767, 1004856, 1004833, 1004870, 1004750, 1002544,
             1004873, 1004835, 1002633],
            dtype='int64')),
(1.0,
 Int64Index([1004809, 1004856, 1004777, 1004833, 1004739, 1004720, 1004788,
             1004838, 1004741, 1004767],
            dtype='int64')),
(3.0,
 Int64Index([1005168, 1004671, 1004944, 1004836, 1004653, 1004655, 1004666,
             1004209, 1005100, 1004529,
             …
             1306315, 1306310, 1304297, 1305570, 1307053, 1306312, 1306952,
             1307059, 1307052, 1305706],
            dtype='int64', name='product_id', length=916)),
(1.0,
 Int64Index([4804056, 4804055, 1004249, 1005115, 1002544, 1004856, 1004767,
             1002524, 1005105, 1004833],
            dtype='int64'))]
```