# Joshua Kobuskie

In [1]:

```
#INSTALL JAX AND OTHER LIBRARIES
!pip install jax
import jax
import pandas as pd
import numpy as np
import gc
import sys
```

```
Requirement already satisfied: jax in /opt/conda/lib/python3.7/site-packages (0.3.1)
Requirement already satisfied: numpy>=1.19 in /opt/conda/lib/python3.7/site-packages (from jax) (1.20.3)
Requirement already satisfied: opt-einsum in /opt/conda/lib/python3.7/site-packages (from jax) (3.3.0)
Requirement already satisfied: scipy>=1.2.1 in /opt/conda/lib/python3.7/site-packages (from jax) (1.7.3)
Requirement already satisfied: typing-extensions in /opt/conda/lib/python3.7/site-packages (from jax) (4.1.1)
Requirement already satisfied: absl-py in /opt/conda/lib/python3.7/site-packages (from jax) (0.15.0)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages (from absl-py->jax) (1.16.0)
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package
manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

In [2]:

```
def obj_size_fmt(num):
    if num<10**3:
        return "{:.2f}{}".format(num,"B")
    elif ((num>=10**3)&(num<10**6)):
        return "{:.2f}{}".format(num/(1.024*10**3),"KB")
        return "{:.2f}{}".format(num/(1.024*10**3),"KB")
    elif ((num>=10**6)&(num<10**9)):
        return "{:.2f}{}".format(num/(1.024*10**6),"MB")
    else:
        return "{:.2f}{}".format(num/(1.024*10**9),"GB")



def memory_usage():
    memory_usage_by_variable=pd.DataFrame({k:sys.getsizeof(v)\
    for (k,v) in globals().items()},index=['Size'])
    memory_usage_by_variable=memory_usage_by_variable.T
    memory_usage_by_variable=memory_usage_by_variable.sort_values(by='Size',ascending=False).head(10)
    memory_usage_by_variable['Size']=memory_usage_by_variable['Size'].apply(lambda x: obj_size_fmt(x))
    return memory_usage_by_variable
```

In [3]:

```
#Read in customer data
customers = pd.read_csv('/kaggle/input/h-and-m-personalized-fashion-recommendations/customers.csv')

#Fill in columns with missing values and convert strings to weights
customers.loc[customers['FN'].isnull(),'FN'] = float(0.0)
customers.loc[customers['Active'].isnull(),'Active'] = float(0.0)



customers.loc[customers['fashion_news_frequency']=='NONE','fashion_news_frequency'] = float(0.0)
customers.loc[customers['fashion_news_frequency']=='None','fashion_news_frequency'] = float(0.0)
customers.loc[customers['fashion_news_frequency'].isnull(),'fashion_news_frequency'] = float(0.0)
customers.loc[customers['fashion_news_frequency']=='Regularly','fashion_news_frequency'] = float(0.5)
customers.loc[customers['fashion_news_frequency']=='Monthly','fashion_news_frequency'] = float(1.0)
customers['fashion_news_frequency'] = customers['fashion_news_frequency'].astype('float64')
```

```python
customers.loc[customers['club_member_status']=='LEFT CLUB','club_member_status'] = float(0.0)
customers.loc[customers['club_member_status'].isnull(),'club_member_status'] = float(0.0)
customers.loc[customers['club_member_status']=='PRE-CREATE','club_member_status'] = float(0.5)
customers.loc[customers['club_member_status']=='ACTIVE','club_member_status'] = float(1.0)
customers['club_member_status'] = customers['club_member_status'].astype('float64')

#Assume average age if none given
average_age = customers['age'].mean()
customers.loc[customers['age'].isnull(),'age'] = average_age

#Remove postal code
customers.drop(columns='postal_code', axis = 1, inplace = True)

customers['customer_id'] = customers['customer_id'].astype('string')

#Display results
customers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1371980 entries, 0 to 1371979
Data columns (total 6 columns):
 #   Column                  Non-Null Count    Dtype  
---  ------                  --------------    -----  
 0   customer_id             1371980 non-null  string 
 1   FN                      1371980 non-null  float64
 2   Active                  1371980 non-null  float64
 3   club_member_status      1371980 non-null  float64
 4   fashion_news_frequency  1371980 non-null  float64
 5   age                     1371980 non-null  float64
dtypes: float64(5), string(1)
memory usage: 62.8 MB
```

In [4]:

```python
#read in transactions and categorize years and months
transactions = pd.read_csv('/kaggle/input/h-and-m-personalized-fashion-recommendations/transactions_train.csv')
transactions['year'] = pd.DatetimeIndex(transactions['t_dat']).year
transactions['month'] = pd.DatetimeIndex(transactions['t_dat']).month
transactions = transactions[transactions['year'] >= 2020]
transactions['t_dat'] = transactions['t_dat'].astype('string')
transactions['customer_id'] = transactions['customer_id'].astype('string')
transactions.info()
transactions.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10980132 entries, 20808192 to 31788323
Data columns (total 7 columns):
 #   Column            Dtype  
---  ------            -----  
 0   t_dat             string 
 1   customer_id       string 
 2   article_id        int64  
 3   price             float64
 4   sales_channel_id  int64  
 5   year              int64  
 6   month             int64  
dtypes: float64(1), int64(4), string(2)
memory usage: 670.2 MB
```

Out[4]:

| | t_dat | customer_id | article_id | price | sales_channel_id | year | month |
|---|---|---|---|---|---|---|---|
| 20808192 | 2020-01-01 | 0034b3dced3e565a43438bdfb5447e7321fea65388b398 | 835247001 | 0.033881 | 2 | 2020 | 1 |

|  | t_dat | customer_id | article_id | price | sales_channel_id | year | month |
|---|---|---|---|---|---|---|---|
| 20808193 | 2020-01-01 | 00410b91d62eefa76958fa5cac12f5daa7cfc0556e417d... | 802930002 | 0.067780 | 2 | 2020 | 1 |
| 20808194 | 2020-01-01 | 00410b91d62eefa76958fa5cac12f5daa7cfc0556e417d... | 760084008 | 0.025407 | 2 | 2020 | 1 |
| 20808195 | 2020-01-01 | 004b0fb384bcab2f8e1059dd5ca68c17580365ab95c05a... | 804662002 | 0.033881 | 2 | 2020 | 1 |
| 20808196 | 2020-01-01 | 004b0fb384bcab2f8e1059dd5ca68c17580365ab95c05a... | 801554002 | 0.016932 | 2 | 2020 | 1 |

In [5]:

```python
#read in articles
from sklearn.preprocessing import LabelEncoder
articles = pd.read_csv('/kaggle/input/h-and-m-personalized-fashion-recommendations/articles.csv')
labelencoder = LabelEncoder()
#convert letters to numbers for groups
articles['product_group_name'] = labelencoder.fit_transform(articles['product_group_name'])
articles['index_code'] = labelencoder.fit_transform(articles['index_code'])
#select only numeric groups
articles = articles[['article_id', 'product_code', 'product_type_no', 'product_group_name', 'graphical_appearance_no', 'col
our_group_code', 'perceived_colour_value_id', 'perceived_colour_master_id', 'department_no', 'index_code', 'index_group_no'
, 'section_no', 'garment_group_no']]
articles.info()
articles.head(10)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 105542 entries, 0 to 105541
Data columns (total 13 columns):
 #   Column                        Non-Null Count   Dtype
---  ------                        --------------   -----
 0   article_id                    105542 non-null  int64
 1   product_code                  105542 non-null  int64
 2   product_type_no               105542 non-null  int64
 3   product_group_name          105542 non-null  int64
 4   graphical_appearance_no     105542 non-null  int64
 5   colour_group_code           105542 non-null  int64
 6   perceived_colour_value_id   105542 non-null  int64
 7   perceived_colour_master_id  105542 non-null  int64
 8   department_no               105542 non-null  int64
 9   index_code                  105542 non-null  int64
 10  index_group_no              105542 non-null  int64
 11  section_no                  105542 non-null  int64
 12  garment_group_no            105542 non-null  int64
dtypes: int64(13)
memory usage: 10.5 MB
```

Out[5]:

| | article_id | product_code | product_type_no | product_group_name | graphical_appearance_no | colour_group_code | perceived_colour_value_id | perceived_colour_master_ |
|---|---|---|---|---|---|---|---|---|
| 0 | 108775015 | 108775 | 253 | 7 | 1010016 | 9 | 4 | 5 |
| 1 | 108775044 | 108775 | 253 | 7 | 1010016 | 10 | 3 | 9 |
| 2 | 108775051 | 108775 | 253 | 7 | 1010017 | 11 | 1 | 9 |
| 3 | 110065001 | 110065 | 306 | 16 | 1010016 | 9 | 4 | 5 |
| 4 | 110065002 | 110065 | 306 | 16 | 1010016 | 10 | 3 | 9 |
| 5 | 110065011 | 110065 | 306 | 16 | 1010016 | 12 | 1 | 11 |
| 6 | 111565001 | 111565 | 304 | 13 | 1010016 | 9 | 4 | 5 |
| 7 | 111565003 | 111565 | 302 | 13 | 1010016 | 13 | 2 | 11 |
| 8 | 111586001 | 111586 | 273 | 6 | 1010016 | 9 | 4 | 5 |
| 9 | 111593001 | 111593 | 304 | 13 | 1010016 | 9 | 4 | 5 |

In [6]:

```
#data = pd.merge(customers, transactions, how="left", on=["customer_id", "customer_id"])
```

```
#data = pd.merge(data, articles, how="left", on=["article_id", "article_id"])
#data = customers.merge(transactions, how='left', left_on='customer_id', right_on='customer_id').merge(articles, how='left', l
eft_on='article_id', right_on='article_id')
data = transactions.merge(customers,on='customer_id').merge(articles,on='article_id')
data['ID'] = data.index
del transactions
#del customers
#saving customers for later to merge onto predictions
del articles
gc.collect()
memory_usage()
```

Out[6]:

| | Size |
|---|---|
| data | 4.07GB |
| customers | 215.71MB |
| _iii | 1.69KB |
| _i3 | 1.69KB |
| _4 | 1.18KB |
| __ | 1.18KB |
| _5 | 1.17KB |
| _ | 1.17KB |
| LabelEncoder | 1.04KB |
| _i2 | 820.00B |

In [7]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10980132 entries, 0 to 10980131
Data columns (total 25 columns):
 #   Column                      Dtype  
---  ------                      -----  
 0   t_dat                       string 
 1   customer_id                 string 
 2   article_id                  int64  
 3   price                       float64
 4   sales_channel_id            int64  
 5   year                        int64  
 6   month                       int64  
 7   FN                          float64
 8   Active                      float64
 9   club_member_status          float64
 10  fashion_news_frequency      float64
 11  age                         float64
 12  product_code                int64  
 13  product_type_no             int64  
 14  product_group_name          int64  
 15  graphical_appearance_no     int64  
 16  colour_group_code           int64  
 17  perceived_colour_value_id   int64  
 18  perceived_colour_master_id  int64  
 19  department_no               int64  
 20  index_code                  int64  
 21  index_group_no              int64  
 22  section_no                  int64  
 23  garment_group_no            int64  
 24  ID                          int64  
dtypes: float64(6), int64(17), string(2)
```

```
memory usage: 2.1 GB
```

In [8]:

```
data.head(10)
```

Out[8]:

| | t_dat | customer_id | article_id | price | sales_channel_id | year | month | FN | Active | club_member_status | ... |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 2020-01-01 | 0034b3dced3e565a43438bdfb5447e7321fea65388b398... | 835247001 | 0.033881 | 2 | 2020 | 1 | 1.0 | 1.0 | 1.0 | ... |
| 1 | 2020-03-10 | 078be7d15562e689421fcad630bee1d41aea7eb518d2b1... | 835247001 | 0.022017 | 1 | 2020 | 3 | 0.0 | 0.0 | 1.0 | ... |
| 2 | 2020-02-04 | 0c30c91b3272fc36b172d7b56636fd8ce54af75a3e9368... | 835247001 | 0.022492 | 2 | 2020 | 2 | 1.0 | 1.0 | 1.0 | ... |
| 3 | 2020-01-01 | 170126f1d3345fa450b87b147906643a3659ccb6ef2871... | 835247001 | 0.031508 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | ... |
| 4 | 2020-01-01 | 170126f1d3345fa450b87b147906643a3659ccb6ef2871... | 835247001 | 0.031576 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | ... |
| 5 | 2020-02-11 | 2414b5f86a1742ebbadc781561ddfac6ca0177a9ef08f1... | 835247001 | 0.027441 | 2 | 2020 | 2 | 0.0 | 0.0 | 1.0 | ... |
| 6 | 2020-03-15 | 31b07944e12db276190a729edce320718ecc3a67121e78... | 835247001 | 0.030492 | 2 | 2020 | 3 | 1.0 | 1.0 | 1.0 | ... |
| 7 | 2020-01-01 | 33992d5bc04fbf1ddac7df567b919ac37d814f293013af... | 835247001 | 0.032068 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | ... |
| 8 | 2020-01-01 | 4444f737c7bc41faef68d30c26957d1c41bdeaef36e926... | 835247001 | 0.033881 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | ... |
| 9 | 2020-01-01 | 459913ba6177f175e6c76a1367f839008ccbe5e46756f5... | 835247001 | 0.033881 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | ... |

10 rows × 25 columns

In [9]:

```
#elimate null values while maintaining customer count
#data.loc[data['t_dat'].isnull(),'t_dat'] = '00-00-0000'
#data.loc[data['article_id'].isnull(),'article_id'] = float(0.0)
#data.loc[data['price'].isnull(),'price'] = float(0.0)
#data.loc[data['sales_channel_id'].isnull(),'sales_channel_id'] = float(0.0)
#data.loc[data['year'].isnull(),'year'] = float(0.0)
#data.loc[data['month'].isnull(),'month'] = float(0.0)
#data.loc[data['product_code'].isnull(),'product_code'] = float(0.0)
#data.loc[data['product_type_no'].isnull(),'product_type_no'] = float(0.0)
#data.loc[data['product_group_name'].isnull(),'product_group_name'] = float(0.0)
#data.loc[data['graphical_appearance_no'].isnull(),'graphical_appearance_no'] = float(0.0)
#data.loc[data['colour_group_code'].isnull(),'colour_group_code'] = float(0.0)
#data.loc[data['perceived_colour_value_id'].isnull(),'perceived_colour_value_id'] = float(0.0)
#data.loc[data['perceived_colour_master_id'].isnull(),'perceived_colour_master_id'] = float(0.0)
#data.loc[data['department_no'].isnull(),'department_no'] = float(0.0)
#data.loc[data['index_code'].isnull(),'index_code'] = float(0.0)
#data.loc[data['index_group_no'].isnull(),'index_group_no'] = float(0.0)
#data.loc[data['section_no'].isnull(),'section_no'] = float(0.0)
#data.loc[data['garment_group_no'].isnull(),'garment_group_no'] = float(0.0)
#Turns out, this was a bad idea. The model predicts 0 all the time and becomes horribly inaccurate.

data.isnull().sum()
```

Out[9]:

```
t_dat                          0
customer_id                    0
article_id                     0
price                          0
sales_channel_id               0
year                           0
```

```
month                        0
FN                           0
Active                       0
club_member_status           0
fashion_news_frequency       0
age                          0
product_code                 0
product_type_no              0
product_group_name           0
graphical_appearance_no      0
colour_group_code            0
perceived_colour_value_id    0
perceived_colour_master_id   0
department_no                0
index_code                   0
index_group_no               0
section_no                   0
garment_group_no             0
ID                           0
dtype: int64
```

In [10]:

```
#JAX FOR RANDOM SAMPLING
import jax.numpy as jnp
from jax import random
#tried using jax_sampling methods and ran into many issues, switched to jax random generation

#Using JAX to create a random array of indexes and use that to take a sample out of the data to pass to X
seed = 1234
key = jax.random.PRNGKey(seed)
#SAMPLE SIZE
```

```
#optimized for accuracy vs time
shape = (25000,)
index_rand = jax.random.randint(key, shape, 0, len(data)-1)
index_rand = np.array(index_rand)

#JAX version
x = data.iloc[index_rand, :].copy()
y = x['article_id']
#Dropping non-numerical data for later
#going to rejoin data to data_num based on ID
x.drop(columns=['article_id', 't_dat', 'customer_id'], axis = 1, inplace = True)
indexes = data[['article_id', 't_dat', 'customer_id', 'ID']]
data.drop(columns=['article_id', 't_dat', 'customer_id'], axis = 1, inplace = True)
```

In [11]:

```
# split the dataset into the training set and test set
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
del x
del y
gc.collect()
memory_usage()
```

Out[11]:

| | Size |
|---|---|
| indexes | 2.27GB |
| data | 1.97GB |
| customers | 215.71MB |
| x_train | 3.59MB |
| x_test | 898.47KB |

| | Size |
|---|---|
| x_test | 898.47KB |
| y_train | 312.53KB |
| index_rand | 97.76KB |
| y_test | 78.16KB |
| _8 | 3.74KB |
| __ | 3.74KB |

In [12]:

```
from sklearn.neighbors import KNeighborsClassifier
x_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20000 entries, 447385 to 4698435
Data columns (total 22 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   price                     20000 non-null  float64
 1   sales_channel_id          20000 non-null  int64  
 2   year                      20000 non-null  int64  
 3   month                     20000 non-null  int64  
 4   FN                        20000 non-null  float64
 5   Active                    20000 non-null  float64
 6   club_member_status        20000 non-null  float64
 7   fashion_news_frequency    20000 non-null  float64
 8   age                       20000 non-null  float64
 9   product_code              20000 non-null  int64  
 10  product_type_no           20000 non-null  int64  
 11  product_group_name        20000 non-null  int64  
 12  graphical_appearance_no   20000 non-null  int64  
 13  colour_group_code           20000 non-null  int64
 14  perceived_colour_value_id   20000 non-null  int64
 15  perceived_colour_master_id  20000 non-null  int64
 16  department_no               20000 non-null  int64
 17  index_code                  20000 non-null  int64
 18  index_group_no              20000 non-null  int64
 19  section_no                  20000 non-null  int64
 20  garment_group_no            20000 non-null  int64
 21  ID                          20000 non-null  int64
dtypes: float64(6), int64(16)
memory usage: 3.5 MB
```

In [13]:

```
k = 1
memory_usage()
#Train Model and Predict
neighbors = KNeighborsClassifier(n_neighbors = k).fit(x_train,y_train)
```

In [14]:

```
yhat = neighbors.predict(x_test)
yhat[0:5]
```

Out[14]:

```
array([810227004, 636323002, 866383001, 806731001, 824337001])
```

In [15]:

```
from sklearn import metrics
```

```python
print("Train set Accuracy: ", metrics.accuracy_score(y_train, neighbors.predict(x_train)))
print("Test set Accuracy: ", metrics.accuracy_score(y_test, yhat))
```

```
Train set Accuracy:  1.0
Test set Accuracy:  0.6912
```

In [16]:

```python
x_train.info()
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20000 entries, 447385 to 4698435
Data columns (total 22 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   price                        20000 non-null  float64
 1   sales_channel_id             20000 non-null  int64  
 2   year                         20000 non-null  int64  
 3   month                        20000 non-null  int64  
 4   FN                           20000 non-null  float64
 5   Active                       20000 non-null  float64
 6   club_member_status           20000 non-null  float64
 7   fashion_news_frequency       20000 non-null  float64
 8   age                          20000 non-null  float64
 9   product_code                 20000 non-null  int64  
 10  product_type_no              20000 non-null  int64  
 11  product_group_name           20000 non-null  int64  
 12  graphical_appearance_no      20000 non-null  int64  
 13  colour_group_code            20000 non-null  int64  
 14  perceived_colour_value_id   20000 non-null  int64
 15  perceived_colour_master_id  20000 non-null  int64
 16  department_no               20000 non-null  int64
 17  index_code                  20000 non-null  int64
 18  index_group_no              20000 non-null  int64
 19  section_no                  20000 non-null  int64
 20  garment_group_no            20000 non-null  int64
 21  ID                          20000 non-null  int64
dtypes: float64(6), int64(16)
memory usage: 3.5 MB
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10980132 entries, 0 to 10980131
Data columns (total 22 columns):
 #   Column                      Dtype
---  ------                      -----
 0   price                       float64
 1   sales_channel_id            int64
 2   year                        int64
 3   month                       int64
 4   FN                          float64
 5   Active                      float64
 6   club_member_status          float64
 7   fashion_news_frequency      float64
 8   age                         float64
 9   product_code                int64
 10  product_type_no             int64
 11  product_group_name          int64
 12  graphical_appearance_no     int64
 13  colour_group_code           int64
 14  perceived_colour_value_id   int64
 15  perceived_colour_master_id  int64
 16  department_no               int64
 17  index_code                  int64
 18  index_group_no               int64
 19  section_no                   int64
 20  garment_group_no             int64
 21  ID                           int64
dtypes: float64(6), int64(16)
memory usage: 1.9 GB
```

In [17]:

```
result_data = neighbors.predict(data)
```

In [18]:

```
len(result_data)
len(data)
```

Out[18]:

```
10980132
```

In [19]:

```
data = data.merge(indexes, on='ID')
data['article'] = result_data.tolist()
```

In [20]:

```
data_ndg = data.drop_duplicates(subset = ['customer_id','article'])
data_ngd = data_ndg.groupby('customer_id')
data_ndg.head(10)
data_ndg.head(10)
```

Out[20]:

| | price | sales_channel_id | year | month | FN | Active | club_member_status | fashion_news_frequency | age | product_code | ... | department_no | index_code | i |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.033881 | 2 | 2020 | 1 | 1.0 | 1.0 | 1.0 | 0.5 | 42.0 | 835247 | ... | 1322 | 0 | 1 |
| 1 | 0.022017 | 1 | 2020 | 3 | 0.0 | 0.0 | 1.0 | 0.0 | 22.0 | 835247 | ... | 1322 | 0 | 1 |
| 2 | 0.022492 | 2 | 2020 | 2 | 1.0 | 1.0 | 1.0 | 0.5 | 45.0 | 835247 | ... | 1322 | 0 | 1 |
| 3 | 0.031508 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 37.0 | 835247 | ... | 1322 | 0 | 1 |
| 5 | 0.027441 | 2 | 2020 | 2 | 0.0 | 0.0 | 1.0 | 0.0 | 25.0 | 835247 | ... | 1322 | 0 | 1 |
| 6 | 0.030492 | 2 | 2020 | 3 | 1.0 | 1.0 | 1.0 | 0.5 | 42.0 | 835247 | ... | 1322 | 0 | 1 |
| 7 | 0.032068 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 29.0 | 835247 | ... | 1322 | 0 | 1 |
| 8 | 0.033881 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 49.0 | 835247 | ... | 1322 | 0 | 1 |
| 9 | 0.033881 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 37.0 | 835247 | ... | 1322 | 0 | 1 |
| 11 | 0.030492 | 2 | 2020 | 1 | 1.0 | 1.0 | 1.0 | 0.5 | 38.0 | 835247 | ... | 1322 | 0 | 1 |

10 rows × 26 columns

In [21]:

```
data_ndg['customer_id'].value_counts()
```

Out[21]:

```
b637a3e7d8b0caa947aaefd609b8d84a9ee962cf0a52a51bac507ffc2bf1b741    539
be1981ab818cf4ef6765b2ecaea7a2cbf14ccd6e8a7ee985513d9e8e53c6d91b    499
cd04ec2726dd58a8c753e0d6423e57716fd9ebcf2f14ed6012e7e5bea016b4d6    442
863f0e03da282ae32a76775ce55d8a4605a85c84a26066e1ad0e9469e8c40e68    412
a65f77281a528bf5c1e9f270141d601d116e1df33bf9df512f495ee06647a9cc    405
                                                                   ...
174d69cb658377048e5d730b1fefd62cda4024b719debfb7b03735ee027f13e7      1
a4b158c57d4483d1cb87f515fe2742bbfdff3128133d45ddb45365e6b5052a0d      1
3372a6bd1f2b06e339cd7c4ac0c3cad4d24bb5a0430c52e76a7511349d6ec6d1      1
bbe5bbdc885e15b232c6d19a1d4524c448cff17942224108ed664c0b193ac054      1
daaceb3e5ec42f538f20dc11a0baccea645c4e6dd22088988bd9289fa0715881      1
Name: customer_id, Length: 862724, dtype: Int64
```

In [22]:

```python
data_ndg.head(10)
```

Out[22]:

| | price | sales_channel_id | year | month | FN | Active | club_member_status | fashion_news_frequency | age | product_code | ... | department_no | index_code | ir |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.033881 | 2 | 2020 | 1 | 1.0 | 1.0 | 1.0 | 0.5 | 42.0 | 835247 | ... | 1322 | 0 | 1 |
| 1 | 0.022017 | 1 | 2020 | 3 | 0.0 | 0.0 | 1.0 | 0.0 | 22.0 | 835247 | ... | 1322 | 0 | 1 |
| 2 | 0.022492 | 2 | 2020 | 2 | 1.0 | 1.0 | 1.0 | 0.5 | 45.0 | 835247 | ... | 1322 | 0 | 1 |
| 3 | 0.031508 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 37.0 | 835247 | ... | 1322 | 0 | 1 |
| 5 | 0.027441 | 2 | 2020 | 2 | 0.0 | 0.0 | 1.0 | 0.0 | 25.0 | 835247 | ... | 1322 | 0 | 1 |
| 6 | 0.030492 | 2 | 2020 | 3 | 1.0 | 1.0 | 1.0 | 0.5 | 42.0 | 835247 | ... | 1322 | 0 | 1 |

| | price | sales_channel_id | year | month | FN | Active | club_member_status | fashion_news_frequency | age | product_code | ... | department_no | index_code | i |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 0.032068 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 29.0 | 835247 | ... | 1322 | 0 | 1 |
| 8 | 0.033881 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 49.0 | 835247 | ... | 1322 | 0 | 1 |
| 9 | 0.033881 | 2 | 2020 | 1 | 0.0 | 0.0 | 1.0 | 0.0 | 37.0 | 835247 | ... | 1322 | 0 | 1 |
| 11 | 0.030492 | 2 | 2020 | 1 | 1.0 | 1.0 | 1.0 | 0.5 | 38.0 | 835247 | ... | 1322 | 0 | 1 |

10 rows × 26 columns

In [23]:

```
result_data=data_ndg[['customer_id','article', 'ID']]
result_data
```

Out[23]:

| | customer_id | article | ID |
|---|---|---|---|
| 0 | 0034b3dced3e565a43438bdfb5447e7321fea65388b398... | 821673001 | 0 |
| 1 | 078be7d15562e689421fcad630bee1d41aea7eb518d2b1... | 821673001 | 1 |
| 2 | 0c30c91b3272fc36b172d7b56636fd8ce54af75a3e9368... | 821673001 | 2 |
| 3 | 170126f1d3345fa450b87b147906643a3659ccb6ef2871... | 821673001 | 3 |
| 5 | 2414b5f86a1742ebbadc781561ddfac6ca0177a9ef08f1... | 821673001 | 5 |
| ... | ... | ... | ... |
| 10980126 | 91539e75dac101c6c1761a08842f8fbe34d1e15f58abe3... | 646829002 | 10980126 |
| 10980127 | cf078f0327367735111094c34b76010dee62510152ddeb... | 907317001 | 10980127 |
| 10980128 | eaf28084976ef4eaf6b3be9c2a91ca6679e91c74959bb5... | 907317001 | 10980128 |
| 10980129 | ec794d6268ee3c75b5bbea014e3299d994bcd1180ecf6d... | 850134003 | 10980129 |
| 10980130 | ec794d6268ee3c75b5bbea014e3299d994bcd1180ecf6d... | 547365012 | 10980130 |

9445143 rows × 3 columns

In [24]:

```
result_data['customer_id'].value_counts()
```

Out[24]:

```
b637a3e7d8b0caa947aaefd609b8d84a9ee962cf0a52a51bac507ffc2bf1b741    539
be1981ab818cf4ef6765b2ecaea7a2cbf14ccd6e8a7ee985513d9e8e53c6d91b    499
cd04ec2726dd58a8c753e0d6423e57716fd9ebcf2f14ed6012e7e5bea016b4d6    442
863f0e03da282ae32a76775ce55d8a4605a85c84a26066e1ad0e9469e8c40e68    412
a65f77281a528bf5c1e9f270141d601d116e1df33bf9df512f495ee06647a9cc    405
                                                                   ...
174d69cb658377048e5d730b1fefd62cda4024b719debfb7b03735ee027f13e7      1
a4b158c57d4483d1cb87f515fe2742bbfdff3128133d45ddb45365e6b5052a0d      1
3372a6bd1f2b06e339cd7c4ac0c3cad4d24bb5a0430c52e76a7511349d6ec6d1      1
bbe5bbdc885e15b232c6d19a1d4524c448cff17942224108ed664c0b193ac054      1
daaceb3e5ec42f538f20dc11a0baccea645c4e6dd22088988bd9289fa0715881      1
Name: customer_id, Length: 862724, dtype: Int64
```

In [25]:

```
result_final = result_data.groupby('customer_id').sum().reset_index()
result_final = result_final.merge(customers,how='right', on='customer_id')
result_final = result_final[['customer_id', 'article']]
```

Out[25]:

```
customer_id         0
```

```
article        509256
dtype: int64
```

In [26]:

```
result_final.columns = ['customer_id','prediction']
#Guess most popular product for every other customer with no transaction history
most_popular_product = result_final['prediction'].mode()
result_final.loc[result_final['prediction'].isnull(),'prediction'] = int(most_popular_product)

#pad with 0 and convert to string for article_ids
result_final['prediction'] = result_final['prediction'].astype('int')
result_final['prediction'] = result_final['prediction'].astype('string')
result_final['prediction'] = result_final['prediction'].apply(lambda x: x.zfill(10))

result_final['prediction'] = result_final['prediction'].astype('string')
result_final['customer_id'] = result_final['customer_id'].astype('string')
result_final.head(20)
```

Out[26]:

| | customer_id | prediction |
|---|---|---|
| 0 | 00000dbacae5abe5e23885899a1fa44253a17956c6d1c3... | 4842791057 |
| 1 | 0000423b00ade91418cceaf3b26c6af3dd342b51fd051e... | 16928243327 |
| 2 | 000058a12d5b43e67d225668fa1f8d618c13dc232df0ca... | 8352360041 |
| 3 | 00005ca1c9ed5f5146b52ac8639a40ca9d57aeff4d1bd2... | 0706016001 |
| 4 | 00006413d8573cd20ed7128e53b7b13819fe5cfc2d801f... | 5550285086 |
| 5 | 000064249685c11552da43ef22a5030f35a147f723d5b0... | 0706016001 |
| 6 | 0000757967448a6cb83efb3ea7a3fb9d418ac7adf2379d... | 1168039017 |
| 7 | 00007d2de826758b65a93dd24ce629ed66842531df6699... | 4879841163 |
| 8 | 00007e8d4e54114b5b2a9b51586325a8d0fa74ea23ef77... | 1670037002 |
| 9 | 00008469a21b50b3d147c97135e25b4201a8c58997f787 | 0706016001 |

| | customer_id | prediction |
|---|---|---|
| 10 | 0000945f66de1a11d9447609b8b41b1bc987ba185a5496... | 1520168016 |
| 11 | 000097d91384a0c14893c09ed047a963c4fc6a5c021044... | 0706016001 |
| 12 | 00009c2aeae8761f738e4f937d9be6b49861a66339c2b1... | 1711185002 |
| 13 | 00009d946eec3ea54add5ba56d5210ea898def4b46c685... | 38705401704 |
| 14 | 0000ae1bbb25e04bdc7e35f718e852adfb3fbb72ef38b3... | 0706016001 |
| 15 | 0000b2f1829e23b24feec422ef13df3ccedaedc85368e6... | 11697524211 |
| 16 | 0000b7a134c3ec0d8842fad1fd4ca28517424c14fc4848... | 0706016001 |
| 17 | 0000b95f630aaa9313028ce9c41154bb95ac7afa34f55b... | 0706016001 |
| 18 | 0000c97821eb48d0e590fd309133f0a6c08f7750f64ccc... | 4868755058 |
| 19 | 0000d6c053fc8f9389d4565051f12402d5774aa4a9d2e5... | 0706016001 |

In [28]:

```
#check for null values
result_final.isnull().sum()
```

Out[28]:

```
customer_id    0
prediction     0
dtype: int64
```

In [27]:

```
import reprlib
submit = result_final.to_csv(index=False)
#limit output displayed
print(reprlib.repr(submit))
```

```
'customer_id,...,0706016001\n'
```