MAD_Street_Den_Task

March 25, 2019

0.0.1 Task-1

- Each record in represents an event by a visitor on an eCommerce website, with the following information: ### DataFields

clicked_epoch (UNIX timestamp in seconds)
date
user_id
product_id
price
category

- Objective: Write a python script to assign a "Session ID" to every record in the data.

```
In [3]: # loading libraries
        import pandas as pd
        from datetime import timedelta
        from datetime import datetime

In [4]: # Reading csv file
        data=pd.read_csv('C:/Users/abc/Downloads/clickStream.csv')

In [5]: data.shape

Out[5]: (413913, 6)

In [6]: data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 413913 entries, 0 to 413912
Data columns (total 6 columns):
clicked_epoch    413913 non-null float64
uuid             413913 non-null int64
date             413913 non-null object
price            413913 non-null float64
product_id       413913 non-null int64
category         413913 non-null object
dtypes: float64(2), int64(2), object(2)
memory usage: 18.9+ MB
```

```
In [7]: data.head() # printing first 5 rows
```

Out[7]:

	clicked_epoch	uuid	date	price	product_id	category
0	1.496273e+09	110971	2017-06-01	599.5	122712	kurta & kurtis
1	1.496273e+09	110971	2017-06-01	599.5	3453	kurta & kurtis
2	1.496276e+09	49864	2017-06-01	1349.1	13610	jeans
3	1.496277e+09	49864	2017-06-01	1124.1	48309	jeans
4	1.496280e+09	21453	2017-06-01	999.0	133239	kurta & kurtis

```
In [8]: new_data=data[['uuid','clicked_epoch']] # Creating a subset of data to assign session_

In [9]: T=15*60 #Setting the time after which session expires if user is inactive

In [10]: # add a column containing previous timestamp

         start=datetime.now() # Storing the current time in start variable
         new_data =  pd.concat([new_data, new_data.groupby('uuid').transform(lambda x:x.shift(

In [11]: new_data.columns = ['user_id','clicked_epoch','prev_mytimestamp']

In [12]: new_data['new_session'] = ((new_data['clicked_epoch'] - new_data['prev_mytimestamp'])>

In [13]: new_data.head()
```

Out[13]:

	user_id	clicked_epoch	prev_mytimestamp	new_session
0	110971	1.496273e+09	NaN	0
1	110971	1.496273e+09	1.496273e+09	0
2	49864	1.496276e+09	NaN	0
3	49864	1.496277e+09	1.496276e+09	0
4	21453	1.496280e+09	NaN	0

```
In [14]: new_data['increment'] = new_data.groupby("user_id")['new_session'].cumsum()
         new_data['session_id'] = new_data['user_id'].astype(str) + '_' + new_data['increment']
         end=datetime.now()

In [15]: new_data=new_data[['user_id','clicked_epoch','session_id']]
         new_data.head()
```

Out[15]:

	user_id	clicked_epoch	session_id
0	110971	1.496273e+09	110971_0
1	110971	1.496273e+09	110971_0
2	49864	1.496276e+09	49864_0
3	49864	1.496277e+09	49864_0
4	21453	1.496280e+09	21453_0

```
In [16]: print("Total Time for the completion of Task1",end-start)

Total Time for the completion of Task1 0:01:07.848225
```

0.0.2 Task-2

- For every transaction, the following fields have been provided: ### DataFields

product_id
category
date

- Objective:For each category in the dataset, find out if there is a seasonal pattern in purchase behaviour. Correspondingly, generate seasonal scores (Range: [0, 1]) for each category across seasons* to indicate seasonal relevance of the category at a given time period.

```
In [80]: # Reading csv file
         start=datetime.now()
         transact_data=pd.read_csv('C:/Users/abc/Downloads/transactions.csv')

In [62]: transact_data.shape

Out[62]: (1203105, 4)

In [63]: transact_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1203105 entries, 0 to 1203104
Data columns (total 4 columns):
transaction_id    1203105 non-null int64
product_id        1203105 non-null int64
category          1203105 non-null object
date              1203105 non-null object
dtypes: int64(2), object(2)
memory usage: 36.7+ MB


In [64]: transact_data.head() # Printing first 5 rows

Out[64]:    transaction_id  product_id          category        date
         0               1      662685      Casual Dress  2017-10-01
         1               2      154881      Casual Dress  2017-10-01
         2               3      220036      Casual Dress  2017-10-01
         3               4      220036      Casual Dress  2017-10-01
         4               5      950839  Pullover Sweater  2017-10-01

In [65]: transact_data.category.value_counts()

Out[65]: Casual Dress         776888
         Pullover Sweater     244096
         Sleeveless Blouse    175913
         Fleece Jacket          6208
         Name: category, dtype: int64
```

```
In [81]: transact_data.date = pd.to_datetime(transact_data.date)
         transact_data['month']=transact_data['date'].dt.month # Creating a new column 'month'

In [67]: transact_data.head() # printing rows after creating month column
```

Out[67]:

	transaction_id	product_id	category	date	month
0	1	662685	Casual Dress	2017-10-01	10
1	2	154881	Casual Dress	2017-10-01	10
2	3	220036	Casual Dress	2017-10-01	10
3	4	220036	Casual Dress	2017-10-01	10
4	5	950839	Pullover Sweater	2017-10-01	10

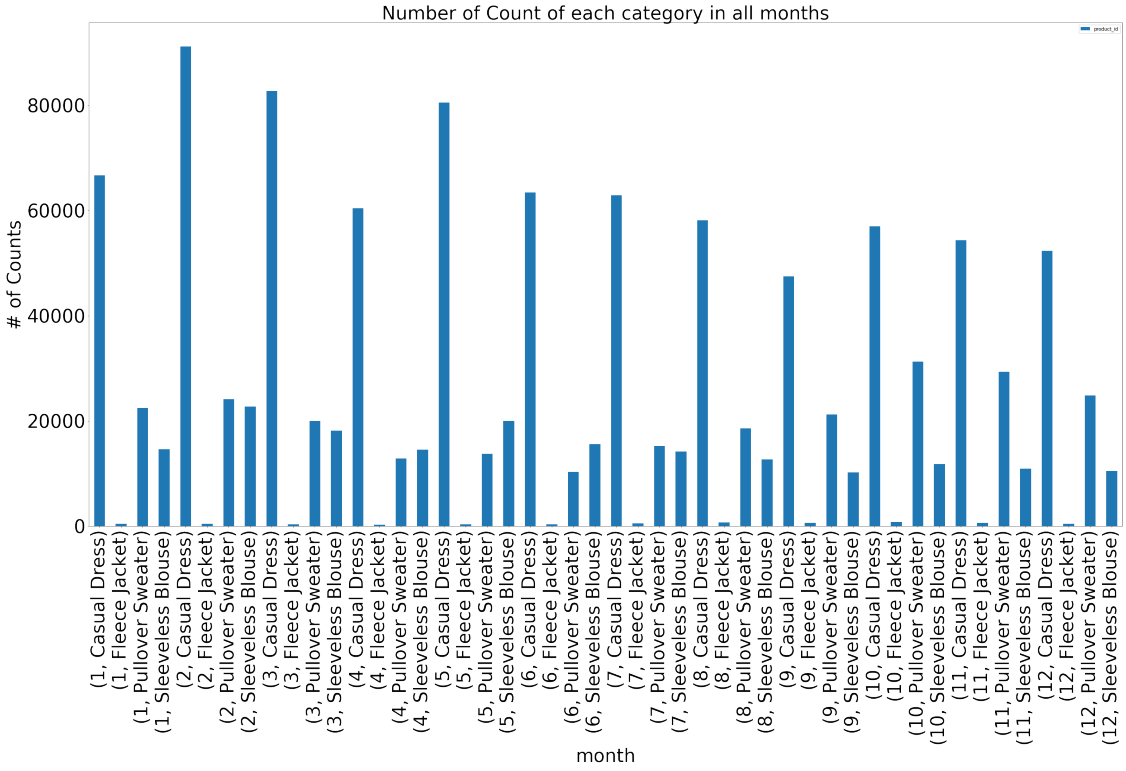
```
In [68]: # storing the count of each category per month in transact_data variable
         transact_data=transact_data[['month','category','product_id']].groupby(['month','categ

In [69]: transact_data
```

Out[69]:

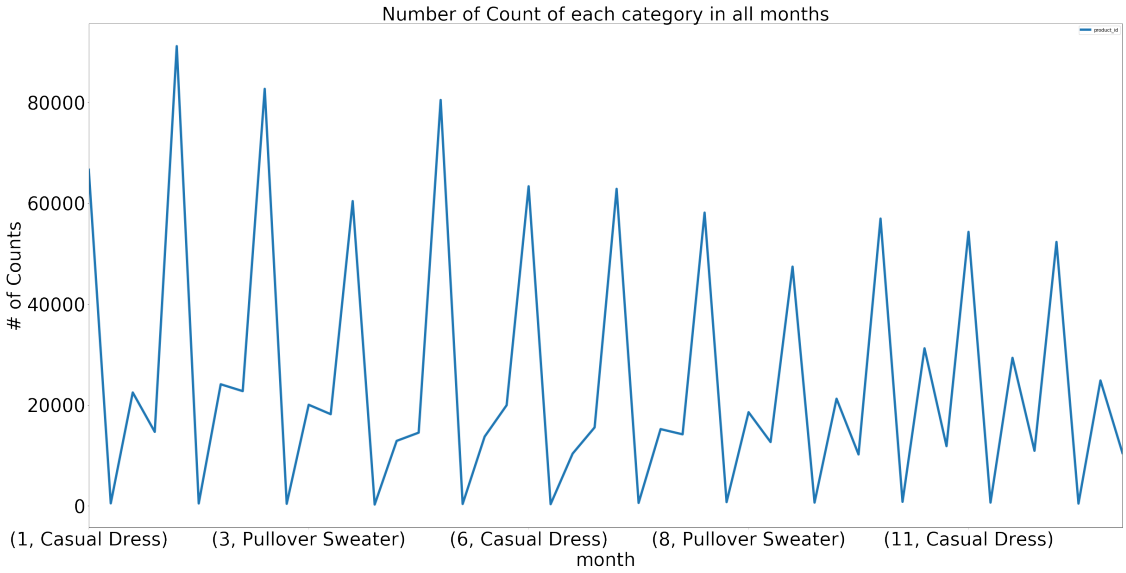
	month	category	product_id
0	1	Casual Dress	66649
1	1	Fleece Jacket	498
2	1	Pullover Sweater	22481
3	1	Sleeveless Blouse	14674
4	2	Casual Dress	91162
5	2	Fleece Jacket	474
6	2	Pullover Sweater	24109
7	2	Sleeveless Blouse	22752
8	3	Casual Dress	82695
9	3	Fleece Jacket	407
10	3	Pullover Sweater	20045
11	3	Sleeveless Blouse	18177
12	4	Casual Dress	60434
13	4	Fleece Jacket	261
14	4	Pullover Sweater	12886
15	4	Sleeveless Blouse	14516
16	5	Casual Dress	80490
17	5	Fleece Jacket	374
18	5	Pullover Sweater	13732
19	5	Sleeveless Blouse	19962
20	6	Casual Dress	63372
21	6	Fleece Jacket	336
22	6	Pullover Sweater	10356
23	6	Sleeveless Blouse	15559
24	7	Casual Dress	62865
25	7	Fleece Jacket	581
26	7	Pullover Sweater	15219
27	7	Sleeveless Blouse	14183
28	8	Casual Dress	58148
29	8	Fleece Jacket	740
30	8	Pullover Sweater	18570

```
31      8  Sleeveless Blouse       12653
32      9       Casual Dress       47438
33      9      Fleece Jacket         645
34      9    Pullover Sweater       21243
35      9  Sleeveless Blouse       10197
36     10       Casual Dress       56963
37     10      Fleece Jacket         798
38     10    Pullover Sweater       31239
39     10  Sleeveless Blouse       11841
40     11       Casual Dress       54334
41     11      Fleece Jacket         645
42     11    Pullover Sweater       29354
43     11  Sleeveless Blouse       10918
44     12       Casual Dress       52338
45     12      Fleece Jacket         449
46     12    Pullover Sweater       24862
47     12  Sleeveless Blouse       10481
```

In [82]:
```
import matplotlib.pyplot as plt
transact_data[['month','category','product_id']].groupby(['month','category']).count()
plt.xlabel('month',fontsize=40)
plt.ylabel('# of Counts',fontsize=40)
plt.title('Number of Count of each category in all months',fontsize=40)
plt.show()
```



In [83]:
```
transact_data[['month','category','product_id']].groupby(['month','category']).count()
plt.xlabel('month',fontsize=40)
plt.ylabel('# of Counts',fontsize=40)
plt.title('Number of Count of each category in all months',fontsize=40)
plt.show()
```



In [72]:
```
# storing the count of category per month in transact_data_per_month variable
transact_data_per_month=transact_data.groupby('month',as_index=False)['product_id'].su
```

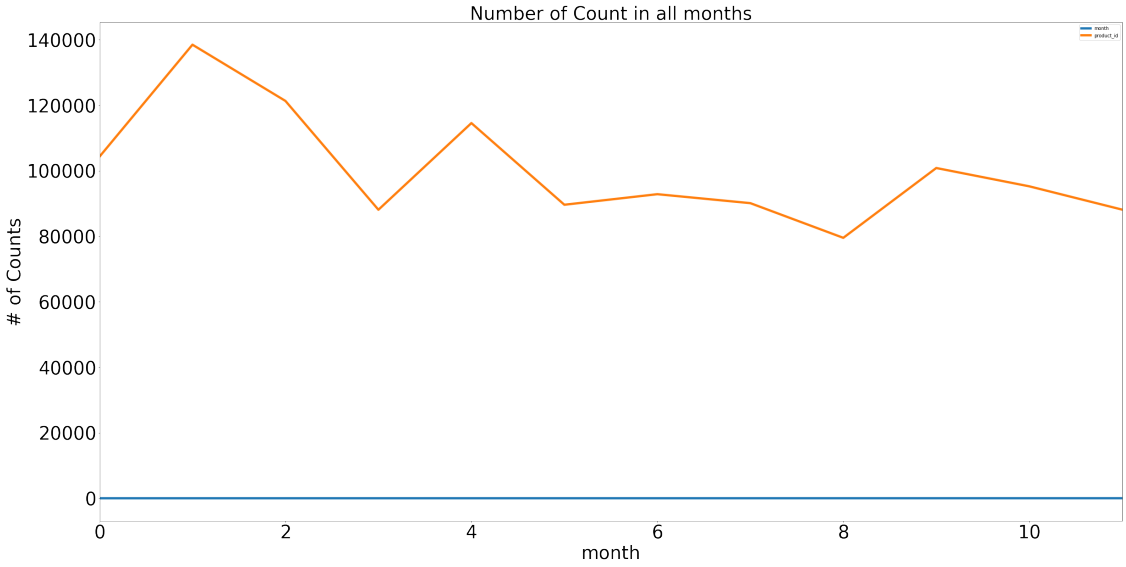
In [73]:
```
transact_data_per_month
```

Out[73]:

	month	product_id
0	1	104302
1	2	138497
2	3	121324
3	4	88097
4	5	114558
5	6	89623
6	7	92848
7	8	90111
8	9	79523
9	10	100841
10	11	95251
11	12	88130

In [74]:
```
transact_data_per_month.plot(figsize=(40,20), linewidth=5, fontsize=40)

plt.xlabel('month',fontsize=40)
plt.ylabel('# of Counts',fontsize=40)
```

```
plt.title('Number of Count in all months',fontsize=40)
plt.show()
```




```
In [75]: required_data=pd.merge(transact_data,transact_data_per_month,on='month') # merging bo

In [76]: # Calculating seasonal_score of each category per month to check the relevance
         required_data['seasonal_score']=required_data['product_id_x']/required_data['product_

In [77]: required_data
```

Out[77]:

	month	category	product_id_x	product_id_y	seasonal_score
0	1	Casual Dress	66649	104302	0.639000
1	1	Fleece Jacket	498	104302	0.004775
2	1	Pullover Sweater	22481	104302	0.215538
3	1	Sleeveless Blouse	14674	104302	0.140688
4	2	Casual Dress	91162	138497	0.658224
5	2	Fleece Jacket	474	138497	0.003422
6	2	Pullover Sweater	24109	138497	0.174076
7	2	Sleeveless Blouse	22752	138497	0.164278
8	3	Casual Dress	82695	121324	0.681605
9	3	Fleece Jacket	407	121324	0.003355
10	3	Pullover Sweater	20045	121324	0.165219
11	3	Sleeveless Blouse	18177	121324	0.149822
12	4	Casual Dress	60434	88097	0.685994
13	4	Fleece Jacket	261	88097	0.002963
14	4	Pullover Sweater	12886	88097	0.146271
15	4	Sleeveless Blouse	14516	88097	0.164773
16	5	Casual Dress	80490	114558	0.702614
17	5	Fleece Jacket	374	114558	0.003265

18	5	Pullover Sweater	13732	114558	0.119869
19	5	Sleeveless Blouse	19962	114558	0.174252
20	6	Casual Dress	63372	89623	0.707095
21	6	Fleece Jacket	336	89623	0.003749
22	6	Pullover Sweater	10356	89623	0.115551
23	6	Sleeveless Blouse	15559	89623	0.173605
24	7	Casual Dress	62865	92848	0.677074
25	7	Fleece Jacket	581	92848	0.006258
26	7	Pullover Sweater	15219	92848	0.163913
27	7	Sleeveless Blouse	14183	92848	0.152755
28	8	Casual Dress	58148	90111	0.645293
29	8	Fleece Jacket	740	90111	0.008212
30	8	Pullover Sweater	18570	90111	0.206079
31	8	Sleeveless Blouse	12653	90111	0.140416
32	9	Casual Dress	47438	79523	0.596532
33	9	Fleece Jacket	645	79523	0.008111
34	9	Pullover Sweater	21243	79523	0.267130
35	9	Sleeveless Blouse	10197	79523	0.128227
36	10	Casual Dress	56963	100841	0.564879
37	10	Fleece Jacket	798	100841	0.007913
38	10	Pullover Sweater	31239	100841	0.309785
39	10	Sleeveless Blouse	11841	100841	0.117422
40	11	Casual Dress	54334	95251	0.570430
41	11	Fleece Jacket	645	95251	0.006772
42	11	Pullover Sweater	29354	95251	0.308175
43	11	Sleeveless Blouse	10918	95251	0.114623
44	12	Casual Dress	52338	88130	0.593873
45	12	Fleece Jacket	449	88130	0.005095
46	12	Pullover Sweater	24862	88130	0.282106
47	12	Sleeveless Blouse	10481	88130	0.118927

```
In [78]: end=datetime.now()

In [79]: print('Total time taken for Task2 is',end-start)

Total time taken for Task2 is 0:00:23.830465
```