Amazon_Fashion_Discovery_Engine_EDA

June 29, 2019

Amazon Apparel Recommendations

0.0.1 [4.2] Data and Code:

https://drive.google.com/open?id=0BwNkduBnePt2VWhCYXhMV3p4dTg

0.0.2 [4.3] Overview of the data

In [2]:
```
#import all the necessary packages.

from PIL import Image
import requests
from io import BytesIO
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import warnings
from bs4 import BeautifulSoup
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import nltk
import math
import time
import re
import os
import seaborn as sns
from collections import Counter
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.metrics import pairwise_distances
from matplotlib import gridspec
from scipy.sparse import hstack
import plotly
import plotly.figure_factory as ff
from plotly.graph_objs import Scatter, Layout
```

```
        plotly.offline.init_notebook_mode(connected=True)
        warnings.filterwarnings("ignore")
```

```
In [33]: # we have give a json file which consists of all information about
         # the products
         # loading the data using pandas' read_json file.
         data = pd.read_json('tops_fashion.json')
```

```
In [34]: print ('Number of data points : ', data.shape[0], \
                'Number of features/variables:', data.shape[1])
```

```
Number of data points :  183138 Number of features/variables: 19
```

0.0.3 Terminology:

What is a dataset? Rows and columns Data-point Feature/variable

```
In [35]: # each product/item has 19 features in the raw dataset.
         data.columns # prints column-names or feature-names.
```

```
Out[35]: Index(['asin', 'author', 'availability', 'availability_type', 'brand', 'color',
                'editorial_reivew', 'editorial_review', 'formatted_price',
                'large_image_url', 'manufacturer', 'medium_image_url', 'model',
                'product_type_name', 'publisher', 'reviews', 'sku', 'small_image_url',
                'title'],
               dtype='object')
```

Of these 19 features, we will be using only 6 features in this workshop. 1. asin (Amazon standard identification number) 2. brand (brand to which the product belongs to) 3. color (Color information of apparel, it can contain many colors as a value ex: red and black stripes) 4. product_type_name (type of the apperal, ex: SHIRT/TSHIRT) 5. medium_image_url (url of the image) 6. title (title of the product.) 7. formatted_price (price of the product)

```
In [36]: data = data[['asin', 'brand', 'color', 'medium_image_url', 'product_type_name', 'title'
```

```
In [37]: print ('Number of data points : ', data.shape[0], \
                'Number of features:', data.shape[1])
         data.head() # prints the top rows in the table.
```

```
Number of data points :  183138 Number of features: 7
```

```
Out[37]:          asin         brand              color  \
         0  B016I2TS4W         FNC7C               None
         1  B01N49AI08  FIG Clothing               None
         2  B01JDPCOHO  FIG Clothing               None
         3  B01N19U5H5       Focal18               None
         4  B004GSI2OS   FeatherLite  Onyx Black/ Stone
```

```
                                          medium_image_url product_type_name  \
0  https://images-na.ssl-images-amazon.com/images...              SHIRT
1  https://images-na.ssl-images-amazon.com/images...              SHIRT
2  https://images-na.ssl-images-amazon.com/images...              SHIRT
3  https://images-na.ssl-images-amazon.com/images...              SHIRT
4  https://images-na.ssl-images-amazon.com/images...              SHIRT

                                               title formatted_price
0  Minions Como Superheroes Ironman Long Sleeve R...            None
1                      FIG Clothing Womens Izo Tunic            None
2                        FIG Clothing Womens Won Top            None
3  Focal18 Sailor Collar Bubble Sleeve Blouse Shi...            None
4  Featherlite Ladies' Long Sleeve Stain Resistan...          $26.26
```

0.0.4 [5.1] Missing data for various features.

Basic stats for the feature: product_type_name

```
In [38]: # We have total 72 unique type of product_type_names
         print(data['product_type_name'].describe())

         # 91.62% (167794/183138) of the products are shirts,
```

```
count     183138
unique        72
top        SHIRT
freq      167794
Name: product_type_name, dtype: object
```

```
In [39]: # names of different product types
         print(data['product_type_name'].unique())
```

```
['SHIRT' 'SWEATER' 'APPAREL' 'OUTDOOR_RECREATION_PRODUCT'
 'BOOKS_1973_AND_LATER' 'PANTS' 'HAT' 'SPORTING_GOODS' 'DRESS' 'UNDERWEAR'
 'SKIRT' 'OUTERWEAR' 'BRA' 'ACCESSORY' 'ART_SUPPLIES' 'SLEEPWEAR'
 'ORCA_SHIRT' 'HANDBAG' 'PET_SUPPLIES' 'SHOES' 'KITCHEN' 'ADULT_COSTUME'
 'HOME_BED_AND_BATH' 'MISC_OTHER' 'BLAZER' 'HEALTH_PERSONAL_CARE'
 'TOYS_AND_GAMES' 'SWIMWEAR' 'CONSUMER_ELECTRONICS' 'SHORTS' 'HOME'
 'AUTO_PART' 'OFFICE_PRODUCTS' 'ETHNIC_WEAR' 'BEAUTY'
 'INSTRUMENT_PARTS_AND_ACCESSORIES' 'POWERSPORTS_PROTECTIVE_GEAR' 'SHIRTS'
 'ABIS_APPAREL' 'AUTO_ACCESSORY' 'NONAPPARELMISC' 'TOOLS' 'BABY_PRODUCT'
 'SOCKSHOSIERY' 'POWERSPORTS_RIDING_SHIRT' 'EYEWEAR' 'SUIT'
 'OUTDOOR_LIVING' 'POWERSPORTS_RIDING_JACKET' 'HARDWARE' 'SAFETY_SUPPLY'
 'ABIS_DVD' 'VIDEO_DVD' 'GOLF_CLUB' 'MUSIC_POPULAR_VINYL'
 'HOME_FURNITURE_AND_DECOR' 'TABLET_COMPUTER' 'GUILD_ACCESSORIES'
 'ABIS_SPORTS' 'ART_AND_CRAFT_SUPPLY' 'BAG' 'MECHANICAL_COMPONENTS'
 'SOUND_AND_RECORDING_EQUIPMENT' 'COMPUTER_COMPONENT' 'JEWELRY'
 'BUILDING_MATERIAL' 'LUGGAGE' 'BABY_COSTUME' 'POWERSPORTS_VEHICLE_PART'
```

```
 'PROFESSIONAL_HEALTHCARE' 'SEEDS_AND_PLANTS' 'WIRELESS_ACCESSORY']
```

```
In [40]: # find the 10 most frequent product_type_names.
         product_type_count = Counter(list(data['product_type_name']))
         product_type_count.most_common(10)
```

```
Out[40]: [('SHIRT', 167794),
          ('APPAREL', 3549),
          ('BOOKS_1973_AND_LATER', 3336),
          ('DRESS', 1584),
          ('SPORTING_GOODS', 1281),
          ('SWEATER', 837),
          ('OUTERWEAR', 796),
          ('OUTDOOR_RECREATION_PRODUCT', 729),
          ('ACCESSORY', 636),
          ('UNDERWEAR', 425)]
```

Basic stats for the feature: brand

```
In [41]: # there are 10577 unique brands
         print(data['brand'].describe())

         # 183138 - 182987 = 151 missing values.
```

```
count     182987
unique     10577
top         Zago
freq          223
Name: brand, dtype: object
```

```
In [42]: brand_count = Counter(list(data['brand']))
         brand_count.most_common(10)
```

```
Out[42]: [('Zago', 223),
          ('XQS', 222),
          ('Yayun', 215),
          ('YUNY', 198),
          ('XiaoTianXin-women clothes', 193),
          ('Generic', 192),
          ('Boohoo', 190),
          ('Alion', 188),
          ('Abetteric', 187),
          ('TheMogan', 187)]
```

Basic stats for the feature: color

```
In [43]: print(data['color'].describe())


         # we have 7380 unique colors
         # 7.2% of products are black in color
         # 64956 of 183138 products have brand information. That's approx 35.4%.

count     64956
unique     7380
top       Black
freq      13207
Name: color, dtype: object


In [44]: color_count = Counter(list(data['color']))
         color_count.most_common(10)

Out[44]: [(None, 118182),
          ('Black', 13207),
          ('White', 8616),
          ('Blue', 3570),
          ('Red', 2289),
          ('Pink', 1842),
          ('Grey', 1499),
          ('*', 1388),
          ('Green', 1258),
          ('Multi', 1203)]
```

Basic stats for the feature: formatted_price

```
In [45]:
         print(data['formatted_price'].describe())

         # Only 28,395 (15.5% of whole data) products with price information

count      28395
unique      3135
top       $19.99
freq         945
Name: formatted_price, dtype: object


In [46]: price_count = Counter(list(data['formatted_price']))
         price_count.most_common(10)

Out[46]: [(None, 154743),
          ('$19.99', 945),
          ('$9.99', 749),
          ('$9.50', 601),
```

```
 ('$14.99', 472),
 ('$7.50', 463),
 ('$24.99', 414),
 ('$29.99', 370),
 ('$8.99', 343),
 ('$9.01', 336)]
```

Basic stats for the feature: title

```
In [47]: print(data['title'].describe())

         # All of the products have a title.
         # Titles are fairly descriptive of what the product is.
         # We use titles extensively in this workshop
         # as they are short and informative.
```

```
count                                                183138
unique                                               175985
top       Nakoda Cotton Self Print Straight Kurti For Women
freq                                                     77
Name: title, dtype: object
```

```
In [48]: data.to_pickle('pickels/180k_apparel_data')
```

We save data files at every major step in our processing in "pickle" files. If you are stuck anywhere (or) if some code takes too long to run on your laptop, you may use the pickle files we give you to speed things up.

```
In [49]: # consider products which have price information
         # data['formatted_price'].isnull() => gives the information
         #about the dataframe row's which have null values price == None/Null
         data = data.loc[~data['formatted_price'].isnull()]
         print('Number of data points After eliminating price=NULL :', data.shape[0])
```

```
Number of data points After eliminating price=NULL : 28395
```

```
In [50]: # consider products which have color information
         # data['color'].isnull() => gives the information about the dataframe row's which have
         data =data.loc[~data['color'].isnull()]
         print('Number of data points After eliminating color=NULL :', data.shape[0])
```

```
Number of data points After eliminating color=NULL : 28385
```

We brought down the number of data points from 183K to 28K. We are processing only 28K points so that most of the workshop participants can run this code on thier laptops in a reasonable amount of time.

For those of you who have powerful computers and some time to spare, you are recommended to use all of the 183K images.

```
In [51]: data.to_pickle('pickels/28k_apparel_data')
```

```
In [52]: # You can download all these 28k images using this code below.
         # You do NOT need to run this code and hence it is commented.


         '''
         from PIL import Image
         import requests
         from io import BytesIO

         for index, row in images.iterrows():
                 url = row['large_image_url']
                 response = requests.get(url)
                 img = Image.open(BytesIO(response.content))
                 img.save('images/28k_images/'+row['asin']+'.jpeg')


         '''
```

```
Out[52]: "\nfrom PIL import Image\nimport requests\nfrom io import BytesIO\n\nfor index, row in
```

0.0.5 [5.2] Remove near duplicate items

[5.2.1] Understand about duplicates.

```
In [97]: # read data from pickle file from previous stage
         data = pd.read_pickle('pickels/28k_apparel_data')

         # find number of products that have duplicate titles.
         print(sum(data.duplicated('title')))
         # we have 2325 products which have same title but different color
```

```
2325
```

These shirts are exactly same except in size (S, M,L,XL) :B00AQ4GMCK
:B00AQ4GMTS
:B00AQ4GMLQ
:B00AQ4GN3I

These shirts exactly same except in color :B00G278GZ6
:B00G278W6O
:B00G278Z2A
:B00G2786X8

In our data there are many duplicate products like the above examples, we need to de-dupe them for better results.

[5.2.2] Remove duplicates : Part 1

In [102]:
```
# read data from pickle file from previous stage
data = pd.read_pickle('pickels/28k_apparel_data')
```

In [103]:
```
data.head()
```

Out[103]:

	asin	brand	color	medium_image_url	product_type_name	title	formatted_price
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	Featherlite Ladies' Long Sleeve Stain Resistan...	$26.26
6	B012YX2ZPI	HX-Kingdom Fashion T-shirts	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	Women's Unique 100% Cotton T - Special Olympic...	$9.99
11	B001LOUGE4	Fitness Etc.	Black	https://images-na.ssl-images-amazon.com/images...	SHIRT	Ladies Cotton Tank 2x1 Ribbed Tank Top	$11.99
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	FeatherLite Ladies' Moisture Free Mesh Sport S...	$20.54
21	B014ICEDNA	FNC7C	Purple	https://images-na.ssl-images-amazon.com/images...	SHIRT	Supernatural Chibis Sam Dean And Castiel Short...	$7.50

In [104]:
```
# Remove All products with very few words in title
data_sorted = data[data['title'].apply(lambda x: len(x.split())>4)]
print("After removal of products with short description:", data_sorted.shape[0])
```

```
After removal of products with short description: 27949
```

In [105]:
```
# Sort the whole data based on title (alphabetical order of title)
data_sorted.sort_values('title',inplace=True, ascending=False)
data_sorted.head()
```

Out[105]:

	asin	brand	color
61973	B06Y1KZ2WB	Éclair	Black/Pink

```
133820  B010RV33VE  xiaoming         Pink
81461   B01DDSDLNS  xiaoming        White
75995   B00X5LYO9Y  xiaoming  Red Anchors
151570  B00WPJG35K  xiaoming        White

                                        medium_image_url product_type_name  \
61973   https://images-na.ssl-images-amazon.com/images...             SHIRT
133820  https://images-na.ssl-images-amazon.com/images...             SHIRT
81461   https://images-na.ssl-images-amazon.com/images...             SHIRT
75995   https://images-na.ssl-images-amazon.com/images...             SHIRT
151570  https://images-na.ssl-images-amazon.com/images...             SHIRT

                                                    title formatted_price
61973   Éclair Women's Printed Thin Strap Blouse Black...          $24.99
133820  xiaoming Womens Sleeveless Loose Long T-shirts...          $18.19
81461   xiaoming Women's White Long Sleeve Single Brea...          $21.58
75995   xiaoming Stripes Tank Patch/Bear Sleeve Anchor...          $15.91
151570  xiaoming Sleeve Sheer Loose Tassel Kimono Woma...          $14.32
```

Some examples of dupliacte titles that differ only in the last few words.

In [106]:
```
indices = []
for i,row in data_sorted.iterrows():
    indices.append(i)
```

In [107]:
```
import itertools
stage1_dedupe_asins = []
i = 0
j = 0
num_data_points = data_sorted.shape[0]
while i < num_data_points and j < num_data_points:

    previous_i = i

    # store the list of words of ith string in a, ex: a = ['tokidoki', 'The', 'Queen',
    a = data['title'].loc[indices[i]].split()

    # search for the similar products sequentially
    j = i+1
    while j < num_data_points:

        # store the list of words of jth string in b, ex: b = ['tokidoki', 'The', 'Que
        b = data['title'].loc[indices[j]].split()

        # store the maximum length of two strings
        length = max(len(a), len(b))

        # count is used to store the number of words that are matched in both strings
```

```
                    count  = 0

                    # itertools.zip_longest(a,b): will map the corresponding words in both strings
                    # example: a =['a', 'b', 'c', 'd']
                    # b = ['a', 'b', 'd']
                    # itertools.zip_longest(a,b): will give [('a','a'), ('b','b'), ('c','d'), ('d'
                    for k in itertools.zip_longest(a,b):
                        if (k[0] == k[1]):
                            count += 1

                    # if the number of words in which both strings differ are > 2 , we are conside
                    # if the number of words in which both strings differ are < 2 , we are conside
                    if (length - count) > 2: # number of words in which both sensences differ
                        # if both strings are differ by more than 2 words we include the 1st strin
                        stage1_dedupe_asins.append(data_sorted['asin'].loc[indices[i]])

                        # if the comaprision between is between num_data_points, num_data_points-1
                        if j == num_data_points-1: stage1_dedupe_asins.append(data_sorted['asin'].

                        # start searching for similar apperals corresponds 2nd string
                        i = j
                        break
                    else:
                        j += 1
                if previous_i == i:
                    break
```

```
In [108]: data = data.loc[data['asin'].isin(stage1_dedupe_asins)]
```

We removed the dupliactes which differ only at the end.

```
In [109]: print('Number of data points : ', data.shape[0])
```

```
Number of data points :  17593
```

```
In [110]: data.to_pickle('pickels/17k_apperal_data')
```

[5.2.3] Remove duplicates : Part 2

```
In [65]: data = pd.read_pickle('pickels/17k_apperal_data')
```

```
In [66]: # This code snippet takes significant amount of time.
         # O(n^2) time.
         # Takes about an hour to run on a decent computer.

         indices = []
         for i,row in data.iterrows():
             indices.append(i)
```

```
        stage2_dedupe_asins = []
        while len(indices)!=0:
            i = indices.pop()
            stage2_dedupe_asins.append(data['asin'].loc[i])
            # consider the first apperal's title
            a = data['title'].loc[i].split()
            # store the list of words of ith string in a, ex: a = ['tokidoki', 'The', 'Queen',
            for j in indices:

                b = data['title'].loc[j].split()
                # store the list of words of jth string in b, ex: b = ['tokidoki', 'The', 'Quee

                length = max(len(a),len(b))

                # count is used to store the number of words that are matched in both strings
                count  = 0

                # itertools.zip_longest(a,b): will map the corresponding words in both strings,
                # example: a =['a', 'b', 'c', 'd']
                # b = ['a', 'b', 'd']
                # itertools.zip_longest(a,b): will give [('a','a'), ('b','b'), ('c','d'), ('d',
                for k in itertools.zip_longest(a,b):
                    if (k[0]==k[1]):
                        count += 1

                # if the number of words in which both strings differ are < 3 , we are consider
                if (length - count) < 3:
                    indices.remove(j)
```

In [71]:
```
        # from whole previous products we will consider only
        # the products that are found in previous cell
        data = data.loc[data['asin'].isin(stage2_dedupe_asins)]
```

In [74]:
```
        print('Number of data points after stage two of dedupe: ',data.shape[0])
        # from 17k apperals we reduced to 16k apperals
```

```
Number of data points after stage two of dedupe:  16042
```

In [75]:
```
        data.to_pickle('pickels/16k_apperal_data')
        # Storing these products in a pickle file
        # candidates who wants to download these files instead
        # of 180K they can download and use them from the Google Drive folder.
```

1 6. Text pre-processing

In [3]:
```
        data = pd.read_pickle('pickels/16k_apperal_data')
```

```
        # NLTK download stop words. [RUN ONLY ONCE]
        # goto Terminal (Linux/Mac) or Command-Prompt (Window)
        # In the temrinal, type these commands
        # $python3
        # $import nltk
        # $nltk.download()
```

```
In [4]: # we use the list of stop words that are downloaded from nltk lib.
        stop_words = set(stopwords.words('english'))
        print ('list of stop words:', stop_words)

        def nlp_preprocessing(total_text, index, column):
            if type(total_text) is not int:
                string = ""
                for words in total_text.split():
                    # remove the special chars in review like '"#$@!%^&*()_+-~?>< etc.
                    word = ("".join(e for e in words if e.isalnum()))
                    # Conver all letters to lower-case
                    word = word.lower()
                    # stop-word removal
                    if not word in stop_words:
                        string += word + " "
                data[column][index] = string
```

```
list of stop words: {'such', 'and', 'hers', 'up', 'she', 'd', 'further', 'all', 'than', 'under',
```

```
In [5]: start_time = time.clock()
        # we take each title and we text-preprocess it.
        for index, row in data.iterrows():
            nlp_preprocessing(row['title'], index, 'title')
        # we print the time it took to preprocess whole titles
        print(time.clock() - start_time, "seconds")
```

```
3.5727220000000006 seconds
```

```
In [6]: data.head()
```

```
Out[6]:          asin                        brand              color  \
        4   B004GSI2OS                  FeatherLite  Onyx Black/ Stone
        6   B012YX2ZPI  HX-Kingdom Fashion T-shirts              White
        15  B003BSRPB0                  FeatherLite              White
        27  B014ICEJ1Q                        FNC7C             Purple
        46  B01NACPBG2                Fifth Degree              Black

                                           medium_image_url product_type_name  \
        4   https://images-na.ssl-images-amazon.com/images...             SHIRT
        6   https://images-na.ssl-images-amazon.com/images...             SHIRT
```

```
15  https://images-na.ssl-images-amazon.com/images...              SHIRT
27  https://images-na.ssl-images-amazon.com/images...              SHIRT
46  https://images-na.ssl-images-amazon.com/images...              SHIRT

                                                    title formatted_price
4   featherlite ladies long sleeve stain resistant...          $26.26
6   womens unique 100 cotton  special olympics wor...           $9.99
15  featherlite ladies moisture free mesh sport sh...          $20.54
27  supernatural chibis sam dean castiel neck tshi...           $7.39
46  fifth degree womens gold foil graphic tees jun...           $6.95
```

```
In [8]: data.to_pickle('pickels/16k_apperal_data_preprocessed')
```

2 Procedure Summary

Done basic EDA on the raw dataset

Removed duplicate data and removed rows having invalid data

After removing duplicates and invalid entries the dataset size reduced to 16K

NLP preprocessing such as stop words removal, converting to lower case, steming etc done on text features

Saved the cleaned & pre-processed data to disk

3 Conclusion

A cleaned and pre-processed dataset is prepared from the raw dataset

After cleaning & prp-processing the dataset size reduced to 16K from 183K