

Amazon Apparel Recommendations



[4.2] Data and Code:

<https://drive.google.com/open?id=0BwNkduBnePt2VWhCYXhMV3p4dTg> (<https://drive.google.com/open?id=0BwNkduBnePt2VWhCYXhMV3p4dTg>)

In [5]:

```
from google.colab import drive  
drive.mount('/content/drive')
```

Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3aietf%3awg%3aoauth%3a2.0%3aoob&response_type=code&scope=email%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3a%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3a%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:

.....

Mounted at /content/drive

[4.3] Overview of the data

In [3]:

```
#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")
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
    import pandas.util.testing as tm
```

In [0]:

```
# 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('/content/drive/My Drive/tops_fashion.json')
```

In [5]:

```
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
```

Terminology:

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

In [6]:

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

Out[6]:

```
Index(['sku', 'asin', 'product_type_name', 'formatted_price', 'author',  
       'color', 'brand', 'publisher', 'availability', 'reviews',  
       'large_image_url', 'availability_type', 'small_image_url',  
       'editorial_review', 'title', 'model', 'medium_image_url',  
       'manufacturer', 'editorial_review'],  
      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 apparel, 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 [0]:

```
data = data[['asin', 'brand', 'color', 'medium_image_url', 'product_type_name', 'title'  
, 'formatted_price']]
```

In [8]:

```
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[8]:

	asin	brand	color	medium_image_url	product_type_name	
0	B016I2TS4W	FNC7C	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	Minion Como Super Ironm Long R...
1	B01N49AI08	FIG Clothing	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	FIG C Wome Tunic
2	B01JDPCOHO	FIG Clothing	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	FIG C Wome Won
3	B01N19U5H5	Focal18	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	Focal Sailor Bubbl Sleev Blous
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	Feath Ladie Sleev Resis



[5.1] Missing data for various features.

Basic stats for the feature: product_type_name

In [9]:

```
# 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 [10]:

```
# 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 [11]:

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

Out[11]:

```
[('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 [12]:

```
# 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 [13]:

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

Out[13]:

```
[('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 [14]:

```
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 [15]:

```
color_count = Counter(list(data['color']))
color_count.most_common(10)
```

Out[15]:

```
[(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 [16]:

```
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 [17]:

```
price_count = Counter(list(data['formatted_price']))
price_count.most_common(10)
```

Out[17]:

```
[(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 [18]:

```
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 [19]:

```
...
data.to_pickle('/content/drive/My Drive/180k_apparel_data')
...
```

Out[19]:

```
"\ndata.to_pickle('/content/drive/My Drive/180k_apparel_data')\n"
```

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 [20]:

```
# 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 [21]:

```
# consider products which have color information
# data['color'].isnull() => gives the information about the dataframe row's which have
# null values price == None/NULL
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 their 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 [22]:

```
...
data.to_pickle('/content/drive/My Drive/28k_apparel_data')
...
```

Out[22]:

```
"\n data.to_pickle('/content/drive/My Drive/28k_apparel_data')\n"
```

In [23]:

```
# 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[23]:

```
"\n from PIL import Image\n import requests\n from io import BytesIO\n\n for i
index, row in images.iterrows():\n     url = row['large_image_url']\n     response = requests.get(url)\n     img = Image.open(BytesIO(response.co
ntent))\n     img.save('images/28k_images/'+row['asin']+'.jpeg')\n\n"
```

[5.2] Remove near duplicate items

[5.2.1] Understand about duplicates.

In [24]:

```
# read data from pickle file from previous stage
data = pd.read_pickle('/content/drive/My Drive/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	
	:B00AQ4GMLQ	

These shirts exactly same except in color

	:B00G278GZ6	
	:B00G278Z2A	

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 [0]:

```
# read data from pickle file from previous stage
data = pd.read_pickle('/content/drive/My Drive/pickels/28k_apparel_data')
```

In [26]:

data.head()

Out[26]:

	asin	brand	color	medium_image_url	product_type_name	
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	Featl Ladie Slee Resi
6	B012YX2ZPI	HX-Kingdom Fashion T-shirts	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	Wom Uniq 100% T - S Olym
11	B001LOUGE4	Fitness Etc.	Black	https://images-na.ssl-images-amazon.com/images...	SHIRT	Ladie Cott 2x1 F Tank
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	Featl Ladie Mois Free Spor
21	B014ICEDNA	FNC7C	Purple	https://images-na.ssl-images-amazon.com/images...	SHIRT	Sup Chib Dear Cast Shor



In [27]:

```
# 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 [28]:

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

Out[28]:

	asin	brand	color	medium_image_url	product_type_name
61973	B06Y1KZ2WB	Éclair	Black/Pink	https://images-na.ssl-images-amazon.com/images...	SHIRT
133820	B010RV33VE	xiaoming	Pink	https://images-na.ssl-images-amazon.com/images...	SHIRT
81461	B01DDSDLNS	xiaoming	White	https://images-na.ssl-images-amazon.com/images...	SHIRT
75995	B00X5LYO9Y	xiaoming	Red Anchors	https://images-na.ssl-images-amazon.com/images...	SHIRT
151570	B00WPJG35K	xiaoming	White	https://images-na.ssl-images-amazon.com/images...	SHIRT

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

Titles 1:

16. woman's place is in the house and the senate shirts for Womens XXL White
17. woman's place is in the house and the senate shirts for Womens M Grey

Title 2:

25. tokidoki The Queen of Diamonds Women's Shirt X-Large
26. tokidoki The Queen of Diamonds Women's Shirt Small
27. tokidoki The Queen of Diamonds Women's Shirt Large

Title 3:

61. psychedelic colorful Howling Galaxy Wolf T-shirt/Colorful Rainbow Animal Print Head Shirt for woman Neon Wolf t-shirt
62. psychedelic colorful Howling Galaxy Wolf T-shirt/Colorful Rainbow Animal Print Head Shirt for woman Neon Wolf t-shirt
63. psychedelic colorful Howling Galaxy Wolf T-shirt/Colorful Rainbow Animal Print Head Shirt for woman Neon Wolf t-shirt
64. psychedelic colorful Howling Galaxy Wolf T-shirt/Colorful Rainbow Animal Print Head Shirt for woman Neon Wolf t-shirt

In [0]:

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

In [0]:

```

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',
    'of', 'Diamonds', 'Women's', 'Shirt', 'X-Large']
    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', 'Queen',
        'of', 'Diamonds', 'Women's', 'Shirt', 'Small']
        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,
        it will appened None in case of unequal 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',
        None)]
        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 considering it as those two apparel are different
        # if the number of words in which both strings differ are < 2 , we are considering it as those two apparel are same, hence we are ignoring them
        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 string
            index
                stage1_dedupe_asins.append(data_sorted['asin'].loc[indices[i]])

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

```

In [0]:

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

We removed the duplicates which differ only at the end.

In [32]:

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

Number of data points : 17592

In [33]:

```
...
data.to_pickle('pickels/17k_apperal_data')
...
```

Out[33]:

```
"\n data.to_pickle('pickels/17k_apperal_data')\n"
```

[5.2.3] Remove duplicates : Part 2

In the previous cell, we sorted whole data in alphabetical order of titles. Then, we removed titles which are adjacent and very similar title

But there are some products whose titles are not adjacent but very similar.

Examples:

Titles-1

86261. UltraClub Women's Classic Wrinkle-Free Long Sleeve Oxford Shirt, Pink, X-X-Large

115042. UltraClub Ladies Classic Wrinkle-Free Long-Sleeve Oxford Light Blue XXL

Titles-2

75004. EVALY Women's Cool University Of UTAH 3/4 Sleeve Raglan Tee

109225. EVALY Women's Unique University Of UTAH 3/4 Sleeve Raglan Tees

120832. EVALY Women's New University Of UTAH 3/4-Sleeve Raglan Tshirt

In [0]:

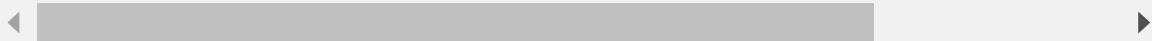
```
data = pd.read_pickle('/content/drive/My Drive/pickels/17k_apperal_data')
```

In [35]:

data.head()

Out[35]:

	asin	brand	color	medium_image_url	product_type_name	
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl- images- amazon.com/images...	SHIRT	Feat Ladi Slee Resi
6	B012YX2ZPI	HX- Kingdom Fashion T- shirts	White	https://images-na.ssl- images- amazon.com/images...	SHIRT	Wor Uniq 100% T - S Olyn
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl- images- amazon.com/images...	SHIRT	Feat Ladi Mois Free Spor
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl- images- amazon.com/images...	SHIRT	Supé Chib Dear Cast Nec.
43	B0079BMKDS	FeatherLite	White	https://images-na.ssl- images- amazon.com/images...	APPAREL	Feat Ladi Smo Piqu (Whi



In [0]:

```
# 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 apparel's title
    a = data['title'].loc[i].split()
    # store the list of words of ith string in a, ex: a = ['tokidoki', 'The', 'Queen',
    'of', 'Diamonds', 'Women's', 'Shirt', 'X-Large']
    for j in indices:

        b = data['title'].loc[j].split()
        # store the list of words of jth string in b, ex: b = ['tokidoki', 'The', 'Queen',
        'of', 'Diamonds', 'Women's', 'Shirt', 'X-Large']

        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,
        it will appened None in case of unequal 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',
        None)]
        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 considering
        it as those two apparels are same, hence we are ignoring them
        if (length - count) < 3:
            indices.remove(j)
```

In [0]:

```
# 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 [38]:

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

Number of data points after stage two of dedupe: 16435

In [39]:

```
'''  
data.to_pickle('pickels/16k_apperial_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.  
'''
```

Out[39]:

```
"\ndata.to_pickle('pickels/16k_apperial_data')\n# Storing these products in  
a pickle file\n# candidates who wants to download these files instead\n#\nof 180K they can download and use them from the Google Drive folder.\n"
```

6. Text pre-processing

In [0]:

```
import pandas as pd  
  
data = pd.read_pickle('/content/drive/My Drive/pickels/16k_apperial_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 [41]:

data.head()

Out[41]:

	asin	brand	color	medium_image_url	product_type_name	
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	Feat Ladi Slee Resi
6	B012YX2ZPI	HX- Kingdom Fashion T-shirts	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	Won Uniq 100% T - S Olyn
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	Feat Ladi Mois Free Spor
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl-images-amazon.com/images...	SHIRT	Sup Chib Deal Cast Nec.
46	B01NACPBG2	Fifth Degree	Black	https://images-na.ssl-images-amazon.com/images...	SHIRT	Fifth Won Gold Grap Tees



In [50]:

```
import nltk
nltk.download('stopwords')
# 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
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]  Unzipping corpora/stopwords.zip.
list of stop words: {'had', 'against', 'be', "wasn't", 'and', 'all', 'the',
'm', 've', 'does', "shan't", 'nor', 'down', 'from', 'hasn', 'being', 'to',
'yourself', 'just', 'been', "you're", 'd', 'they', 'hers', 'after', 'tha
t', 'once', 'we', 'hadn', 'such', "don't", 'here', 'other', 'you', 'the',
>won', 'these', 'shouldn', "didn't", 'll', 'into', 'in', 'but', "mustn't",
'with', "doesn't", 'ours', "you'll", 'haven', 'yours', 're', 'her', "tha
t'll", 'itself', 'which', 'most', 'at', 'own', 'same', 'doing', 'over', 'w
asn', 'will', 'while', 'any', 'up', 'below', 'can', 'why', "have
n't", 'further', 'through', 'some', 'than', 'aren', "wouldn't", 'yourselv
e', 'isn', 'if', "should've", 'have', "hadn't", "you'd", 'didn', 'where',
'not', 'on', 'doesn', 'until', 'having', 'him', 'there', 'o', 't', 'how',
'ourselves', 'who', 'themselves', 'has', 'did', 'were', 'between', 'or',
'whom', 'me', 'so', 'more', "hasn't", 'because', 'too', 'needn', 'our', 'm
ustn', 'each', 'then', 'no', 'few', "weren't", 'm', 'a', 'am', 'myself',
'theirs', 'an', 'himself', 'as', 'of', 'when', "isn't", 'is', 'it', 'abov
e', 'was', 'about', 'very', "needn't", 'before', 'ma', 'weren', 'i', 'hi
s', 'under', 'mightn', 'she', 'herself', 'off', 'couldn', "mightn't", "i
t's", 'wouldn', "you've", 'shan', "aren't", 'what', 'again', 'both', 'no
w', 'your', "shouldn't", 'out', 'only', 's', "couldn't", "she's", 'y', 'h
e', 'by', 'for', 'my', 'don', "won't", 'those', 'their', 'do', 'this', 'ai
n', 'should', 'during', 'its'}
```

In [51]:

```
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")
```

5.475258999999824 seconds

In [52]:

data.head()

Out[52]:

	asin	brand	color	medium_image_url	product_type_name	
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	feath ladi slee resis
6	B012YX2ZPI	HX-Kingdom Fashion T-shirts	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	wom uniq cottc spec olym wor..
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	feath ladi mois free spor
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl-images-amazon.com/images...	SHIRT	supe chibi dear neck
46	B01NACPBG2	Fifth Degree	Black	https://images-na.ssl-images-amazon.com/images...	SHIRT	fifth wom gold grap jun..

< >

In [53]:

```
'''  
data.to_pickle('/content/dive/My Drive/16k_apperal_data_preprocessed')  
'''
```

Out[53]:

```
"\n data.to_pickle('/content/dive/My Drive/16k_apperal_data_preprocessed')  
\n"
```

Stemming

In [54]:

```
from nltk.stem.porter import *
stemmer = PorterStemmer()
print(stemmer.stem('arguing'))
print(stemmer.stem('fishing'))
```

We tried using stemming on our titles and it didnot work very well.

argu
fish

[8] Text based product similarity

In [55]:

```
data = pd.read_pickle('/content/drive/My Drive/16k_apperal_data_preprocessed')
data.head()
```

Out[55]:

	asin	brand	color	medium_image_url	product_type_name	
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl- images- amazon.com/images...	SHIRT	feath ladie slee resis
6	B012YX2ZPI	HX- Kingdom Fashion T- shirts	White	https://images-na.ssl- images- amazon.com/images...	SHIRT	wom uniq cottc spec olym wor..
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl- images- amazon.com/images...	SHIRT	feath ladie mois free spor
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl- images- amazon.com/images...	SHIRT	supe chibi dear neck
46	B01NACPBG2	Fifth Degree	Black	https://images-na.ssl- images- amazon.com/images...	SHIRT	fifth wom gold grap jun..



In [0]:

```
# Utility Functions which we will use through the rest of the workshop.

#Display an image
def display_img(url,ax,fig):
    # we get the url of the apparel and download it
    response = requests.get(url)
    img = Image.open(BytesIO(response.content))
    # we will display it in notebook
    plt.imshow(img)

#plotting code to understand the algorithm's decision.
def plot_heatmap(keys, values, labels, url, text):
    # keys: list of words of recommended title
    # values: len(values) == len(keys), values(i) represents the occurrence of the word keys(i)
    # labels: len(labels) == len(keys), the values of labels depends on the model we are using
    # if model == 'bag of words': labels(i) = values(i)
    # if model == 'tfidf weighted bag of words': labels(i) = tfidf(keys(i))
    # if model == 'idf weighted bag of words': labels(i) = idf(keys(i))
    # url : apparel's url

    # we will devide the whole figure into two parts
    gs = gridspec.GridSpec(2, 2, width_ratios=[4,1], height_ratios=[4,1])
    fig = plt.figure(figsize=(25,3))

    # 1st, plotting heat map that represents the count of commonly occurred words in title2
    ax = plt.subplot(gs[0])
    # it displays a cell in white color if the word is intersection(list of words of title1 and list of words of title2), in black if not
    ax = sns.heatmap(np.array([values]), annot=np.array([labels]))
    ax.set_xticklabels(keys) # set that axis labels as the words of title
    ax.set_title(text) # apparel title

    # 2nd, plotting image of the the apparel
    ax = plt.subplot(gs[1])
    # we don't want any grid lines for image and no labels on x-axis and y-axis
    ax.grid(False)
    ax.set_xticks([])
    ax.set_yticks([])

    # we call dispaly_img based with paramete url
    display_img(url, ax, fig)

    # displays combine figure ( heat map and image together)
    plt.show()

def plot_heatmap_image(doc_id, vec1, vec2, url, text, model):

    # doc_id : index of the title1
    # vec1 : input apparels's vector, it is of a dict type {word:count}
    # vec2 : recommended apparels's vector, it is of a dict type {word:count}
    # url : apparels image url
    # text: title of recomonded apparel (used to keep title of image)
    # model, it can be any of the models,
    # 1. bag_of_words
    # 2. tfidf
```

```
# 3. idf
```

```
# we find the common words in both titles, because these only words contribute to the distance between two title vec's
intersection = set(vec1.keys()) & set(vec2.keys())

# we set the values of non intersecting words to zero, this is just to show the difference in heatmap
for i in vec2:
    if i not in intersection:
        vec2[i]=0

# for Labeling heatmap, keys contains list of all words in title2
keys = list(vec2.keys())
# if ith word in intersection(list of words of title1 and list of words of title2):
values(i)=count of that word in title2 else values(i)=0
values = [vec2[x] for x in vec2.keys()]

# Labels: Len(Labels) == Len(keys), the values of Labels depends on the model we are using
# if model == 'bag of words': Labels(i) = values(i)
# if model == 'tfidf weighted bag of words': Labels(i) = tfidf(keys(i))
# if model == 'idf weighted bag of words': Labels(i) = idf(keys(i))

if model == 'bag_of_words':
    labels = values
elif model == 'tfidf':
    labels = []
    for x in vec2.keys():
        # tfidf_title_vectorizer.vocabulary_ it contains all the words in the corpus
        # tfidf_title_features[doc_id, index_of_word_in_corpus] will give the tfidf value of word in given document (doc_id)
        if x in tfidf_title_vectorizer.vocabulary_:
            labels.append(tfidf_title_features[doc_id, tfidf_title_vectorizer.vocabulary_[x]])
        else:
            labels.append(0)
elif model == 'idf':
    labels = []
    for x in vec2.keys():
        # idf_title_vectorizer.vocabulary_ it contains all the words in the corpus
        # idf_title_features[doc_id, index_of_word_in_corpus] will give the idf value of word in given document (doc_id)
        if x in idf_title_vectorizer.vocabulary_:
            labels.append(idf_title_features[doc_id, idf_title_vectorizer.vocabulary_[x]])
        else:
            labels.append(0)

plot_heatmap(keys, values, labels, url, text)

# this function gets a list of words along with the frequency of each word given "text"
def text_to_vector(text):
    word = re.compile(r'\w+')
    words = word.findall(text)
    # words stores list of all words in given string, you can try 'words = text.split()' this will also gives same result
    return Counter(words) # Counter counts the occurrence of each word in list, it return
```

```

ns dict type object {word1:count}

def get_result(doc_id, content_a, content_b, url, model):
    text1 = content_a
    text2 = content_b

    # vector1 = dict{word11:#count, word12:#count, etc.}
    vector1 = text_to_vector(text1)

    # vector1 = dict{word21:#count, word22:#count, etc.}
    vector2 = text_to_vector(text2)

    plot_heatmap_image(doc_id, vector1, vector2, url, text2, model)

```

[8.2] Bag of Words (BoW) on product titles.

In [57]:

```

from sklearn.feature_extraction.text import CountVectorizer
title_vectorizer = CountVectorizer()
title_features = title_vectorizer.fit_transform(data['title'])
title_features.get_shape() # get number of rows and columns in feature matrix.
# title_features.shape = #data_points * #words_in_corpus
# CountVectorizer().fit_transform(corpus) returns
# the a sparse matrix of dimensions #data_points * #words_in_corpus

# What is a sparse vector?

# title_features[doc_id, index_of_word_in_corpus] = number of times the word occurred in
that doc

```

Out[57]:

(16042, 12609)

In [58]:

```
def bag_of_words_model(doc_id, num_results):
    # doc_id: apparel's id in given corpus

    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as K(X, Y) = <
    X, Y> / (||X||*||Y||)
    # http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
    pairwise_dist = pairwise_distances(title_features,title_features[doc_id])

    # np.argsort will return indices of the smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    #data frame indices of the 9 smallest distace's
    df_indices = list(data.index[indices])

    for i in range(0,len(indices)):
        # we will pass 1. doc_id, 2. title1, 3. title2, url, model
        get_result(indices[i],data['title'].loc[df_indices[0]], data['title'].loc[df_indices[i]], data['medium_image_url'].loc[df_indices[i]], 'bag_of_words')
        print('ASIN :',data['asin'].loc[df_indices[i]])
        print ('Brand:', data['brand'].loc[df_indices[i]])
        print ('Title:', data['title'].loc[df_indices[i]])
        print ('Euclidean similarity with the query image :', pdists[i])
        print('='*60)

    #call the bag-of-words model for a product to get similar products.
    bag_of_words_model(12566, 20) # change the index if you want to.
    # In the output heat map each value represents the count value
    # of the label word, the color represents the intersection
    # with inputs title.

#try 12566
#try 931
```

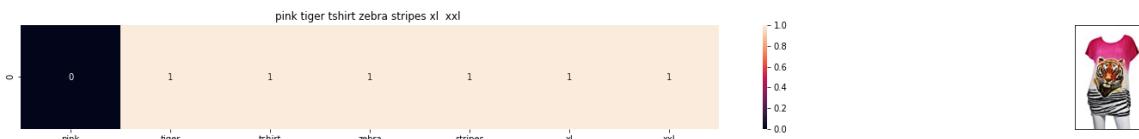


ASIN : B00JXQB5FQ

Brand: Si Row

Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean similarity with the query image : 0.0



ASIN : B00JXQASS6

Brand: Si Row

Title: pink tiger tshirt zebra stripes xl xxl

Euclidean similarity with the query image : 1.7320508075688772



ASIN : B00JXQCWT0

Brand: Si Row

Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean similarity with the query image : 2.449489742783178

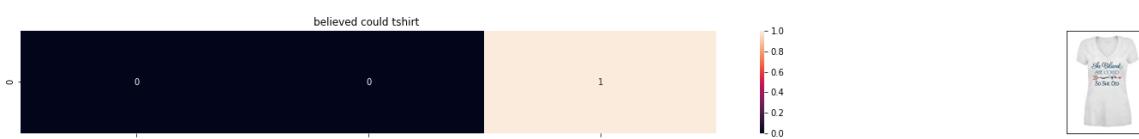


ASIN : B00JXQCUIC

Brand: Si Row

Title: yellow tiger tshirt tiger stripes l

Euclidean similarity with the query image : 2.6457513110645907

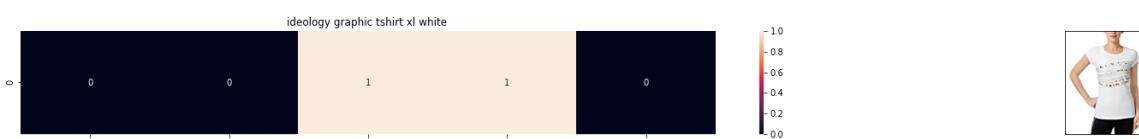


ASIN : B07568NZX4

Brand: Rustic Grace

Title: believed could tshirt

Euclidean similarity with the query image : 3.0

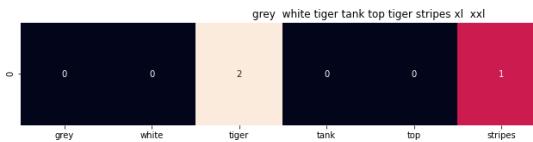


ASIN : B01NB0NKRO

Brand: Ideology

Title: ideology graphic tshirt xl white

Euclidean similarity with the query image : 3.0

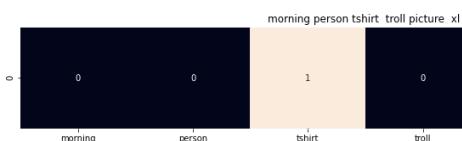


ASIN : B00JXQAFZ2

Brand: Si Row

Title: grey white tiger tank top tiger stripes xl xxl

Euclidean similarity with the query image : 3.0



ASIN : B01CLS8LMW

Brand: Awake

Title: morning person tshirt troll picture xl

Euclidean similarity with the query image : 3.1622776601683795



ASIN : B01KVZUB6G

Brand: Merona

Title: merona green gold stripes

Euclidean similarity with the query image : 3.1622776601683795

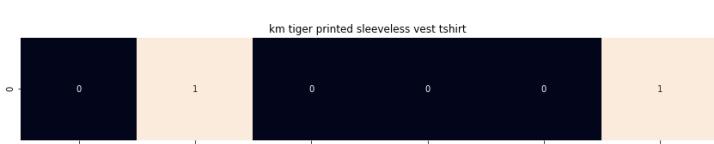


ASIN : B0733R2CJK

Brand: BLVD

Title: blvd womens graphic tshirt l

Euclidean similarity with the query image : 3.1622776601683795



ASIN : B012VQLT6Y

Brand: KM T-shirt

Title: km tiger printed sleeveless vest tshirt

Euclidean similarity with the query image : 3.1622776601683795

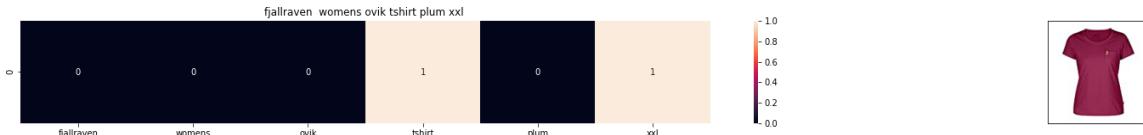


ASIN : B00JXQC8L6

Brand: Si Row

Title: blue peacock print tshirt l

Euclidean similarity with the query image : 3.1622776601683795



ASIN : B06XC3CZF6

Brand: Fjallraven

Title: fjallraven womens ovik tshirt plum xxl

Euclidean similarity with the query image : 3.1622776601683795



ASIN : B005IT80BA

Brand: Hetalia

Title: hetalia us girl tshirt

Euclidean similarity with the query image : 3.1622776601683795

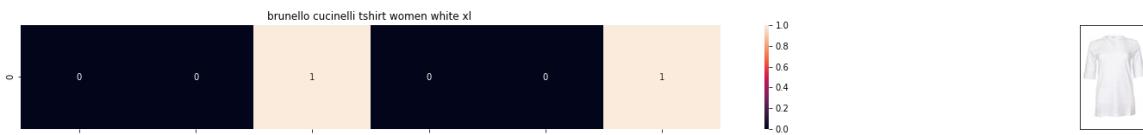


ASIN : B0088PN0LA

Brand: Red House

Title: canvas 3001 30s tshirt kelly xl

Euclidean similarity with the query image : 3.1622776601683795

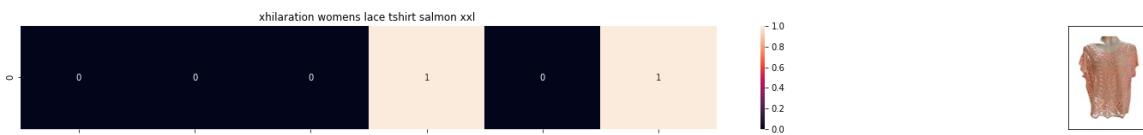


ASIN : B06X99V6WC

Brand: Brunello Cucinelli

Title: brunello cucinelli tshirt women white xl

Euclidean similarity with the query image : 3.1622776601683795

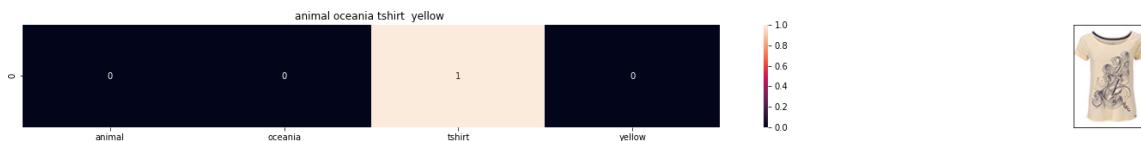


ASIN : B06Y1JPW1Q

Brand: Xhilaration

Title: xhilaration womens lace tshirt salmon xxl

Euclidean similarity with the query image : 3.1622776601683795

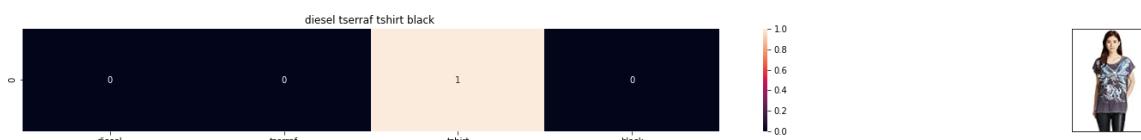


ASIN : B06X6GX6WG

Brand: Animal

Title: animal oceania tshirt yellow

Euclidean similarity with the query image : 3.1622776601683795

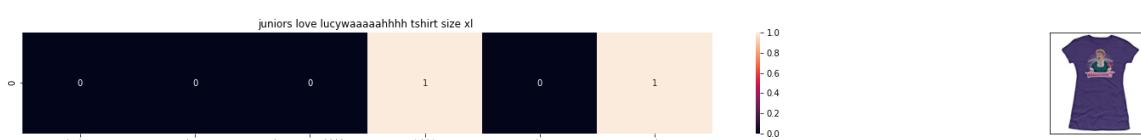


ASIN : B017X8PW9U

Brand: Diesel

Title: diesel tserraf tshirt black

Euclidean similarity with the query image : 3.1622776601683795



ASIN : B00IAA4JIQ

Brand: I Love Lucy

Title: juniors love lucywaaaaahhhh tshirt size xl

Euclidean similarity with the query image : 3.1622776601683795

[8.5] TF-IDF based product similarity

In [0]:

```
tfidf_title_vectorizer = TfidfVectorizer(min_df = 0)
tfidf_title_features = tfidf_title_vectorizer.fit_transform(data['title'])
# tfidf_title_features.shape = #data_points * #words_in_corpus
# CountVectorizer().fit_transform(courpus) returns the a sparase matrix of dimensions # data_points * #words_in_corpus
# tfidf_title_features[doc_id, index_of_word_in_corpus] = tfidf values of the word in given doc
```

In [60]:

```
def tfidf_model(doc_id, num_results):
    # doc_id: apparel's id in given corpus

    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as  $K(X, Y) = \frac{X \cdot Y}{\|X\| * \|Y\|}$ 
    # http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
    pairwise_dist = pairwise_distances(tfidf_title_features, tfidf_title_features[doc_id])

    # np.argsort will return indices of 9 smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the 9 smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    #data frame indices of the 9 smallest distace's
    df_indices = list(data.index[indices])

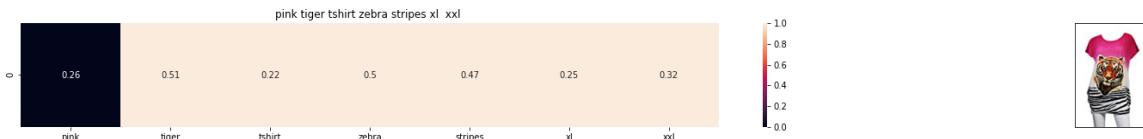
    for i in range(0,len(indices)):
        # we will pass 1. doc_id, 2. title1, 3. title2, url, model
        get_result(indices[i], data['title'].loc[df_indices[0]], data['title'].loc[df_indices[i]], data['medium_image_url'].loc[df_indices[i]], 'tfidf')
        print('ASIN :', data['asin'].loc[df_indices[i]])
        print('BRAND :', data['brand'].loc[df_indices[i]])
        print ('Eucliden distance from the given image :', pdists[i])
        print('='*125)
tfidf_model(12566, 20)
# in the output heat map each value represents the tfidf values of the label word, the color represents the intersection with inputs title
```



ASIN : B00JXQB5FQ

BRAND : Si Row

Eucliden distance from the given image : 0.0



ASIN : B00JXQASS6

BRAND : Si Row

Eucliden distance from the given image : 0.7536331912451363



ASIN : B00JXQCWT0

BRAND : Si Row

Eucliden distance from the given image : 0.9357643943769647



ASIN : B00JXQAFZ2

BRAND : Si Row

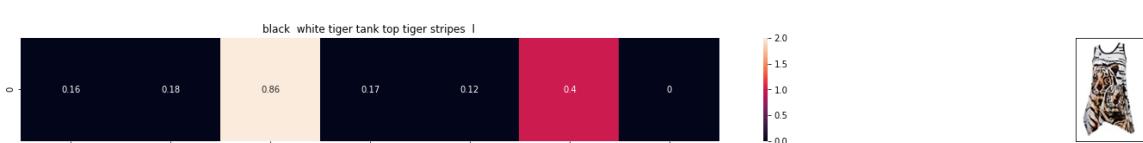
Eucliden distance from the given image : 0.9586153524200749



ASIN : B00JXQCUIC

BRAND : Si Row

Eucliden distance from the given image : 1.000074961446881



ASIN : B00JXQA094

BRAND : Si Row

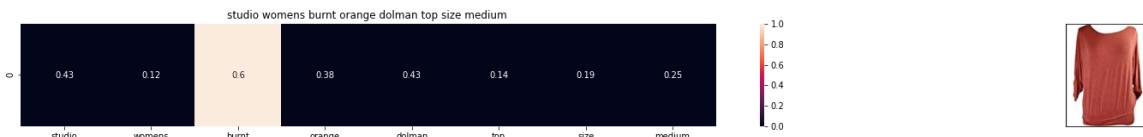
Euclidean distance from the given image : 1.023215552457452



ASIN : B00JXQAUWA

BRAND : Si Row

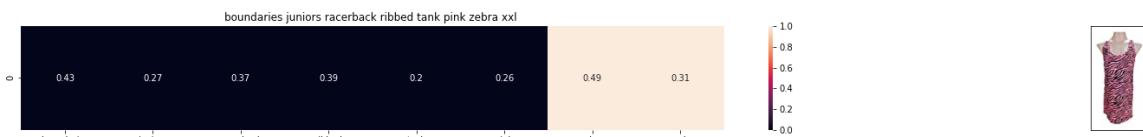
Euclidean distance from the given image : 1.031991846303421



ASIN : B06XSCVFT5

BRAND : Studio M

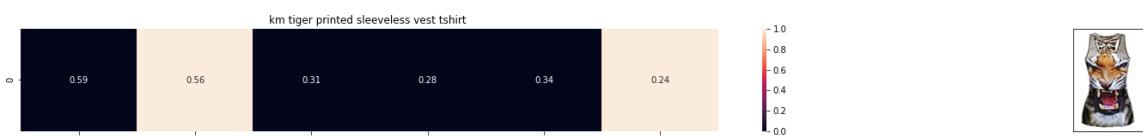
Euclidean distance from the given image : 1.2106843670424716



ASIN : B06Y2GTYPM

BRAND : No Boundaries

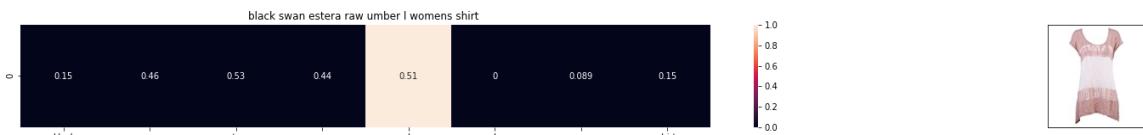
Euclidean distance from the given image : 1.2121683810720831



ASIN : B012VQLT6Y

BRAND : KM T-shirt

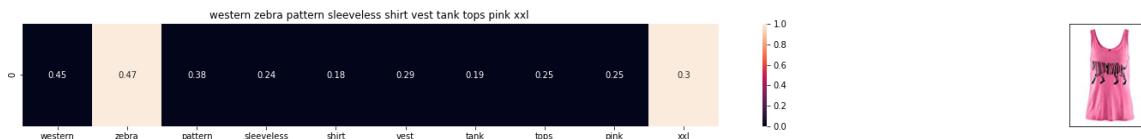
Euclidean distance from the given image : 1.219790640280982



ASIN : B06Y1VN8WQ

BRAND : Black Swan

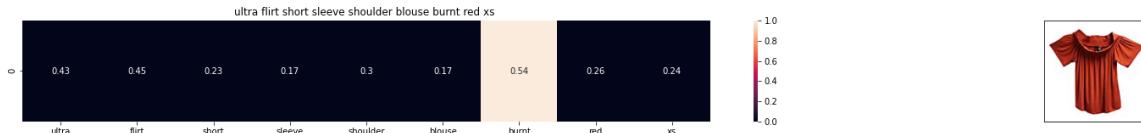
Euclidean distance from the given image : 1.2206849659998316



ASIN : B00Z6HEXWI

BRAND : Black Temptation

Eucliden distance from the given image : 1.221281392120943



ASIN : B074TR12BH

BRAND : Ultra Flirt

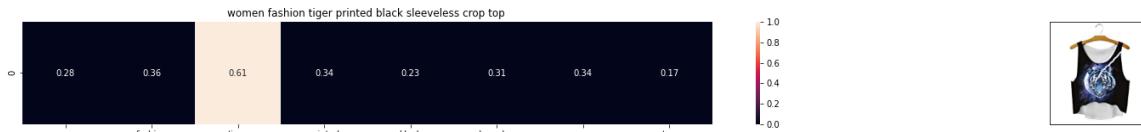
Eucliden distance from the given image : 1.2313364094597743



ASIN : B072R2JXKW

BRAND : WHAT ON EARTH

Eucliden distance from the given image : 1.2318451972624516



ASIN : B074T8ZYGX

BRAND : MKP Crop Top

Eucliden distance from the given image : 1.2340607457359425



ASIN : B071ZDF6T2

BRAND : Mossimo

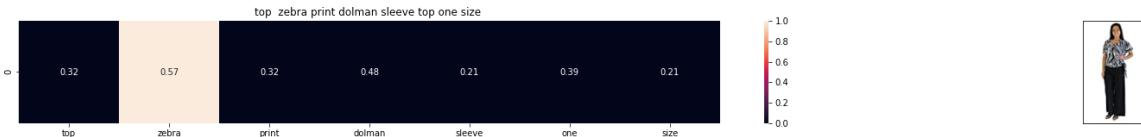
Eucliden distance from the given image : 1.2352785577664824



ASIN : B01K0H020G

BRAND : Tultex

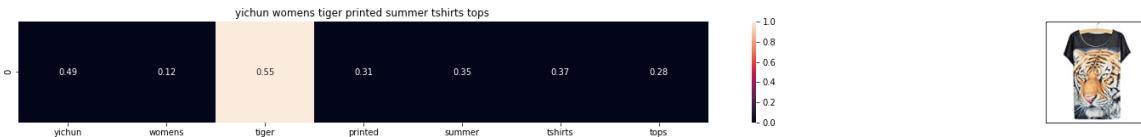
Eucliden distance from the given image : 1.236457298812782



ASIN : B00H8A6ZLI

BRAND : Vivian's Fashions

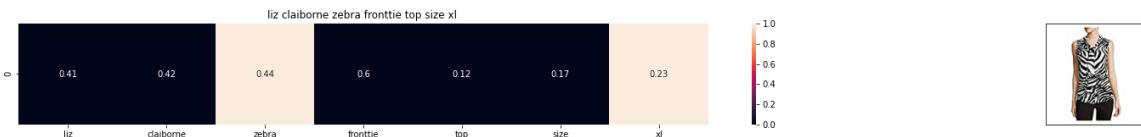
Eucliden distance from the given image : 1.24996155052848



ASIN : B010NN9RX0

BRAND : YICHUN

Eucliden distance from the given image : 1.2535461420856102



ASIN : B06XBY5QXL

BRAND : Liz Claiborne

Eucliden distance from the given image : 1.2538832938357722

[8.5] IDF based product similarity

In [0]:

```
idf_title_vectorizer = CountVectorizer()
idf_title_features = idf_title_vectorizer.fit_transform(data['title'])

# idf_title_features.shape = #data_points * #words_in_corpus
# CountVectorizer().fit_transform(courpus) returns the a sparase matrix of dimensions # 
# data_points * #words_in_corpus
# idf_title_features[doc_id, index_of_word_in_corpus] = number of times the word occure
# d in that doc
```

In [0]:

```
def nContaining(word):
    # return the number of documents which had the given word
    return sum(1 for blob in data['title'] if word in blob.split())

def idf(word):
    # idf = Log(#number of docs / #number of docs which had the given word)
    return math.log(data.shape[0] / (nContaining(word)))
```

In [0]:

```
def nContaining(word):
    # return the number of documents which had the given word
    return sum(1 for blob in data['title'] if word in blob.split())

def idf(word):
    # idf = Log(#number of docs / #number of docs which had the given word
    try:
        return math.log(data.shape[0] / (nContaining(word)))
    except ZeroDivisionError:
        return -1
```

In [0]:

```
# we need to convert the values into float
idf_title_features = idf_title_features.astype(np.float)

for i in idf_title_vectorizer.vocabulary_.keys():
    # for every word in whole corpus we will find its idf value
    idf_val = idf(i)

    # to calculate idf_title_features we need to replace the count values with the idf
    # values of the word
    # idf_title_features[:, idf_title_vectorizer.vocabulary_[i]].nonzero()[0] will return all
    # documents in which the word i present
    for j in idf_title_features[:, idf_title_vectorizer.vocabulary_[i]].nonzero()[0]:
        # we replace the count values of word i in document j with idf_value of word i
        # idf_title_features[doc_id, index_of_word_in_corpus] = idf value of word
        idf_title_features[j,idf_title_vectorizer.vocabulary_[i]] = idf_val
```

In [65]:

```
def idf_model(doc_id, num_results):
    # doc_id: apparel's id in given corpus

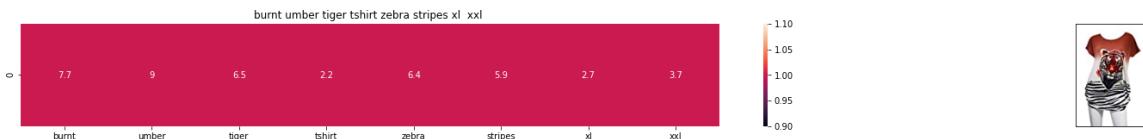
    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as  $K(X, Y) = \frac{X \cdot Y}{\|X\| * \|Y\|}$ 
    # http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
    pairwise_dist = pairwise_distances(idf_title_features,idf_title_features[doc_id])

    # np.argsort will return indices of 9 smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the 9 smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    #data frame indices of the 9 smallest distace's
    df_indices = list(data.index[indices])

    for i in range(0,len(indices)):
        get_result(indices[i],data['title'].loc[df_indices[0]], data['title'].loc[df_indices[i]], data['medium_image_url'].loc[df_indices[i]], 'idf')
        print('ASIN :',data['asin'].loc[df_indices[i]])
        print('Brand :',data['brand'].loc[df_indices[i]])
        print ('euclidean distance from the given image :', pdists[i])
        print('='*125)

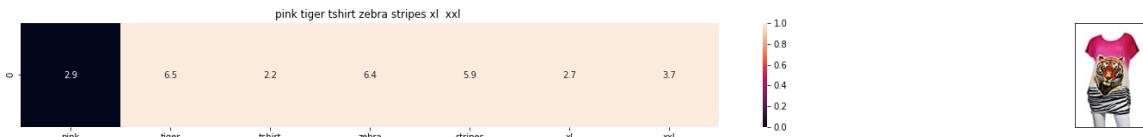
    idf_model(12566,20)
    # in the output heat map each value represents the idf values of the label word, the color represents the intersection with inputs title
```



ASIN : B00JXQB5FQ

Brand : Si Row

euclidean distance from the given image : 0.0



ASIN : B00JXQASS6

Brand : Si Row

euclidean distance from the given image : 12.20507131122177



ASIN : B00JXQCWT0

Brand : Si Row

euclidean distance from the given image : 14.468362685603465



ASIN : B00JXQAFZ2

Brand : Si Row

euclidean distance from the given image : 14.486832924778964



ASIN : B00JXQA094

Brand : Si Row

euclidean distance from the given image : 14.833392966672909



ASIN : B00JXQCUIC

Brand : Si Row

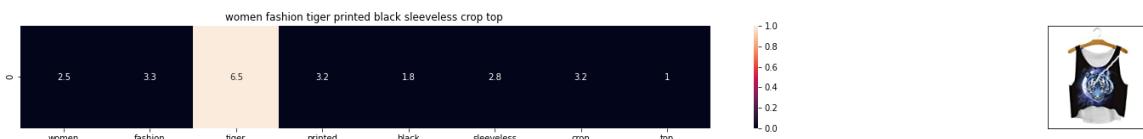
euclidean distance from the given image : 14.898744516719225



ASIN : B00JXQAUWA

Brand : Si Row

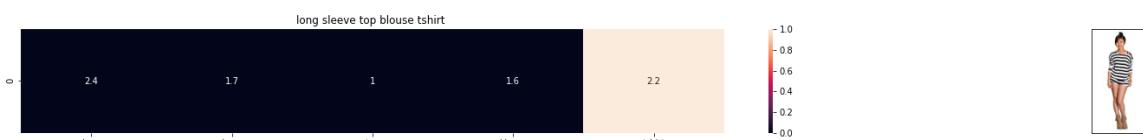
euclidean distance from the given image : 15.224458287343769



ASIN : B074T8ZYGX

Brand : MKP Crop Top

euclidean distance from the given image : 17.080812955631995



ASIN : B00KF2N5PU

Brand : Vietsbay

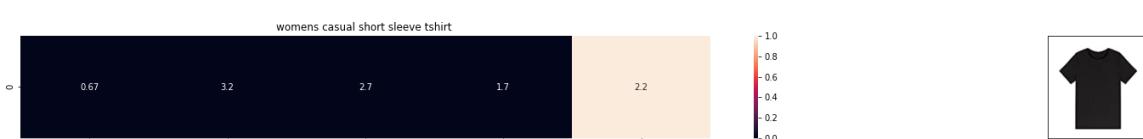
euclidean distance from the given image : 17.090168125645416



ASIN : B00JP0Z9GM

Brand : Sofra

euclidean distance from the given image : 17.153215337562703



ASIN : B074T9KG9Q

Brand : Rain

euclidean distance from the given image : 17.33671523874989



ASIN : B00H8A6ZLI

Brand : Vivian's Fashions

euclidean distance from the given image : 17.410075941001253



ASIN : B074G5G5RK

Brand : ERMANNO SCERVINO

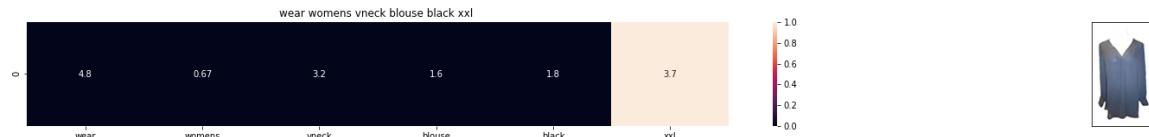
euclidean distance from the given image : 17.539921335459557



ASIN : B06XSCVFT5

Brand : Studio M

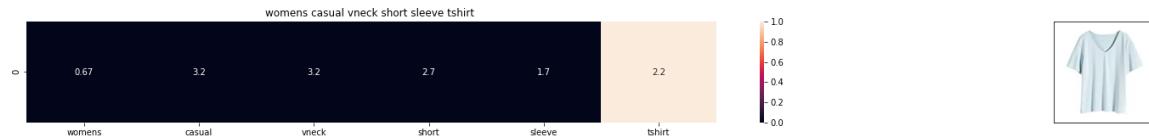
euclidean distance from the given image : 17.61275854366134



ASIN : B06Y6FH453

Brand : Who What Wear

euclidean distance from the given image : 17.623745282500135



ASIN : B074V45DCX

Brand : Rain

euclidean distance from the given image : 17.634342496835046



ASIN : B07583CQFT

Brand : Very J

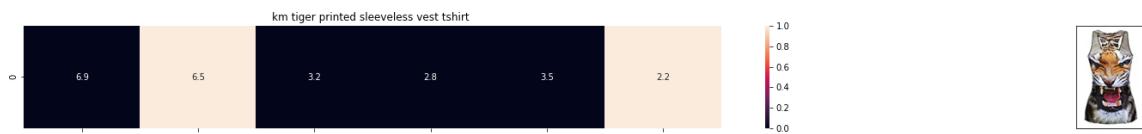
euclidean distance from the given image : 17.63753712743611



ASIN : B073GJGVBN

Brand : Ivan Levi

euclidean distance from the given image : 17.7230738913371



ASIN : B012VQLT6Y

Brand : KM T-shirt

euclidean distance from the given image : 17.762588561202364



ASIN : B00ZZMYBRG

Brand : HP-LEISURE

euclidean distance from the given image : 17.779536864674238

[9] Text Semantics based product similarity

In [66]:

```
# credits: https://www.kaggle.com/c/word2vec-nlp-tutorial#part-2-word-vectors
# Custom Word2Vec using your own text data.
# Do NOT RUN this code.
# It is meant as a reference to build your own Word2Vec when you have
# lots of data.

...
# Set values for various parameters
num_features = 300      # Word vector dimensionality
min_word_count = 1        # Minimum word count
num_workers = 4            # Number of threads to run in parallel
context = 10               # Context window size
downsampling = 1e-3        # Downsample setting for frequent words

# Initialize and train the model (this will take some time)
from gensim.models import word2vec
print ("Training model...")
model = word2vec.Word2Vec(sen_corpus, workers=num_workers,
                           size=num_features, min_count = min_word_count,
                           window = context)

...
```

Out[66]:

```
'\n# Set values for various parameters\nnum_features = 300      # Word vecto
r dimensionality\n                           \nmin_word_count = 1      # Minimum wo
rd count\n                           \nnum_workers = 4          # Number of thread
s to run in parallel\ncontext = 10                # Context window size\n\ndownsampling = 1e-3    # Downsample setting for frequent words\n\n# Initia
lize and train the model (this will take some time)\n\nfrom gensim.models i
mport word2vec\nprint ("Training model...")\nmodel = word2vec.Word2Vec(sen
_corpus, workers=num_workers,\n                           size=num_features, min_count = m
in_word_count,\n                           window = context)\n      \n'
```

In [0]:

```
from gensim.models import Word2Vec
from gensim.models import KeyedVectors
import pickle

# in this project we are using a pretrained model by google
# its 3.3G file, once you load this into your memory
# it occupies ~9Gb, so please do this step only if you have >12G of ram
# we will provide a pickle file which contains a dict ,
# and it contains all our corpus words as keys and model[word] as values
# To use this code-snippet, download "GoogleNews-vectors-negative300.bin"
# from https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTLSS21pQmM/edit
# it's 1.9GB in size.

...
model = KeyedVectors.load_word2vec_format('GoogleNews-vectors-negative300.bin', binary=True)
...

#if you do NOT have RAM >= 12GB, use the code below.
with open('/content/drive/My Drive/word2vec_model', 'rb') as handle:
    model = pickle.load(handle)
```

In [0]:

```
# Utility functions

def get_word_vec(sentence, doc_id, m_name):
    # sentence : title of the apparel
    # doc_id: document id in our corpus
    # m_name: model information it will take two values
        # if m_name == 'avg', we will append the model[i], w2v representation of word
    i
        # if m_name == 'weighted', we will multiply each w2v[word] with the idf(word)
    vec = []
    for i in sentence.split():
        if i in vocab:
            if m_name == 'weighted' and i in idf_title_vectorizer.vocabulary_:
                vec.append(idf_title_features[doc_id, idf_title_vectorizer.vocabulary_[i]] * model[i])
            elif m_name == 'avg':
                vec.append(model[i])
        else:
            # if the word in our corpus is not there in the google word2vec corpus, we
            # are just ignoring it
            vec.append(np.zeros(shape=(300,)))
    # we will return a numpy array of shape (#number of words in title * 300 ) 300 = len(w2v_model[word])
    # each row represents the word2vec representation of each word (weighted/avg) in given sentence
    return np.array(vec)

def get_distance(vec1, vec2):
    # vec1 = np.array(#number_of_words_title1 * 300), each row is a vector of length 300
    # corresponds to each word in give title
    # vec2 = np.array(#number_of_words_title2 * 300), each row is a vector of length 300
    # corresponds to each word in give title

    final_dist = []
    # for each vector in vec1 we calculate the distance(euclidean) to all vectors in vec2
    for i in vec1:
        dist = []
        for j in vec2:
            # np.linalg.norm(i-j) will result the euclidean distance between vectors i, j
            dist.append(np.linalg.norm(i-j))
        final_dist.append(np.array(dist))
    # final_dist = np.array(#number of words in title1 * #number of words in title2)
    # final_dist[i,j] = euclidean distance between vectors i, j
    return np.array(final_dist)

def heat_map_w2v(sentence1, sentence2, url, doc_id1, doc_id2, model):
    # sentence1 : title1, input apparel
    # sentence2 : title2, recommended apparel
    # url: apparel image url
    # doc_id1: document id of input apparel
    # doc_id2: document id of recommended apparel
    # model: it can have two values, 1. avg 2. weighted

    # s1_vec = np.array(#number_of_words_title1 * 300), each row is a vector(weighted/avg) of length 300 corresponds to each word in give title
    s1_vec = get_word_vec(sentence1, doc_id1, model)
```

```
#s2_vec = np.array(#number_of_words_title1 * 300), each row is a vector(weighted/average) of length 300 corresponds to each word in give title
g) s2_vec = get_word_vec(sentence2, doc_id2, model)

# s1_s2_dist = np.array(#number of words in title1 * #number of words in title2)
# s1_s2_dist[i,j] = euclidean distance between words i, j
s1_s2_dist = get_distance(s1_vec, s2_vec)

# devide whole figure into 2 parts 1st part displays heatmap 2nd part displays image of apparel
gs = gridspec.GridSpec(2, 2, width_ratios=[4,1],height_ratios=[2,1])
fig = plt.figure(figsize=(15,15))

ax = plt.subplot(gs[0])
# plotting the heap map based on the pairwise distances
ax = sns.heatmap(np.round(s1_s2_dist,4), annot=True)
# set the x axis labels as recommended apparels title
ax.set_xticklabels(sentence2.split())
# set the y axis labels as input apparels title
ax.set_yticklabels(sentence1.split())
# set title as recommended apparels title
ax.set_title(sentence2)

ax = plt.subplot(gs[1])
# we remove all grids and axis labels for image
ax.grid(False)
ax.set_xticks([])
ax.set_yticks([])
display_img(url, ax, fig)

plt.show()
```

In [0]:

```
# vocab = stores all the words that are there in google w2v model
# vocab = model.wv.vocab.keys() # if you are using Google word2Vec

vocab = model.keys()
# this function will add the vectors of each word and returns the avg vector of given sentence
def build_avg_vec(sentence, num_features, doc_id, m_name):
    # sentace: its title of the apparel
    # num_features: the lenght of word2vec vector, its values = 300
    # m_name: model information it will take two values
        # if m_name == 'avg', we will append the model[i], w2v representation of word
    i
        # if m_name == 'weighted', we will multiply each w2v[word] with the idf(word)

    featureVec = np.zeros((num_features,), dtype="float32")
    # we will intialize a vector of size 300 with all zeros
    # we add each word2vec(wordi) to this festureVec
    nwords = 0

    for word in sentence.split():
        nwords += 1
        if word in vocab:
            if m_name == 'weighted' and word in idf_title_vectorizer.vocabulary_:
                featureVec = np.add(featureVec, idf_title_features[doc_id, idf_title_vectorizer.vocabulary_[word]] * model[word])
            elif m_name == 'avg':
                featureVec = np.add(featureVec, model[word])
    if(nwords>0):
        featureVec = np.divide(featureVec, nwords)
    # returns the avg vector of given sentance, its of shape (1, 300)
    return featureVec
```

[9.2] Average Word2Vec product similarity.

In [0]:

```
doc_id = 0
w2v_title = []
# for every title we build a avg vector representation
for i in data['title']:
    w2v_title.append(build_avg_vec(i, 300, doc_id,'avg'))
    doc_id += 1

# w2v_title = np.array(# number of doc in courpus * 300), each row corresponds to a doc
w2v_title = np.array(w2v_title)
```

In [71]:

```
def avg_w2v_model(doc_id, num_results):
    # doc_id: apparel's id in given corpus

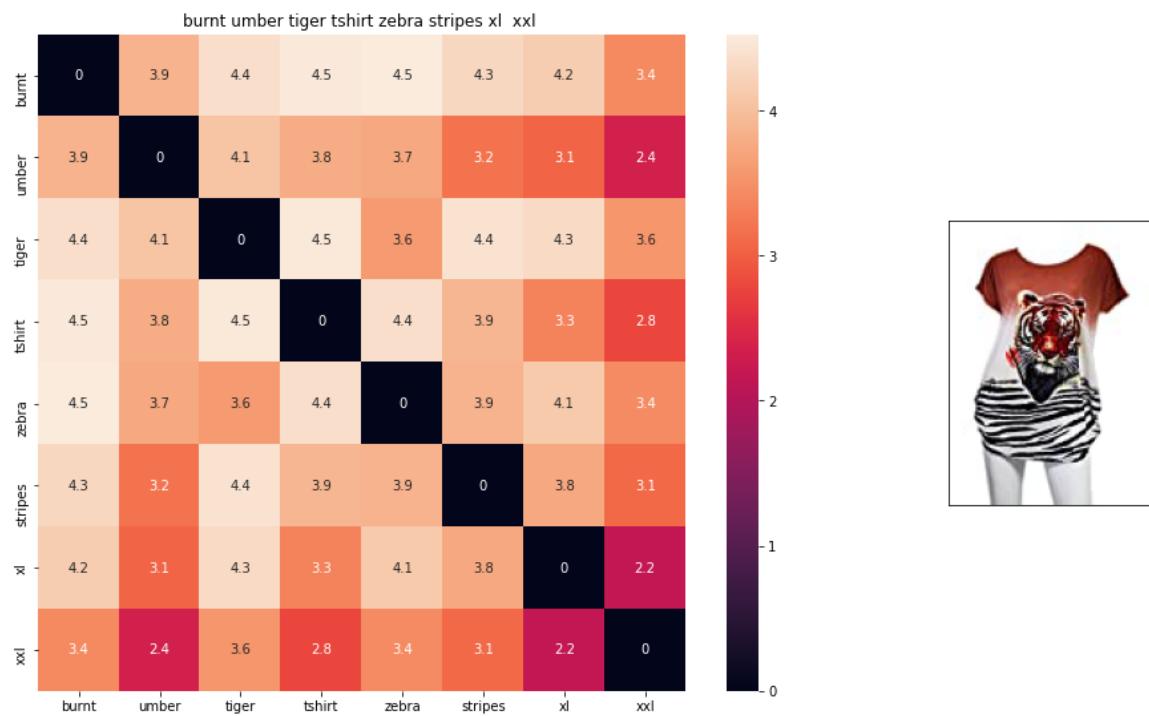
    # dist(x, y) = sqrt(dot(x, x) - 2 * dot(x, y) + dot(y, y))
    pairwise_dist = pairwise_distances(w2v_title, w2v_title[doc_id].reshape(1,-1))

    # np.argsort will return indices of 9 smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the 9 smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    #data frame indices of the 9 smallest distance's
    df_indices = list(data.index[indices])

    for i in range(0, len(indices)):
        heat_map_w2v(data['title'].loc[df_indices[0]],data['title'].loc[df_indices[i]],
data['medium_image_url'].loc[df_indices[i]], indices[0], indices[i], 'avg')
        print('ASIN :',data['asin'].loc[df_indices[i]])
        print('BRAND :',data['brand'].loc[df_indices[i]])
        print ('euclidean distance from given input image :', pdists[i])
        print('='*125)

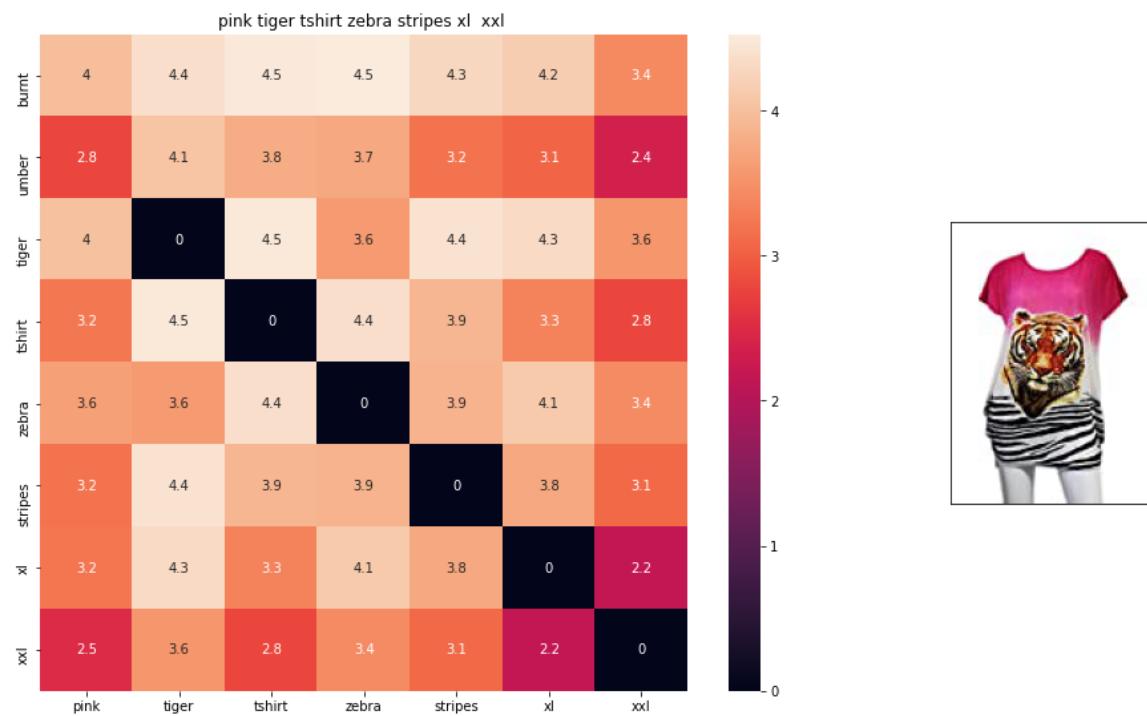
avg_w2v_model(12566, 20)
# in the give heat map, each cell contains the euclidean distance between words i, j
```



ASIN : B00JXQB5FQ

BRAND : Si Row

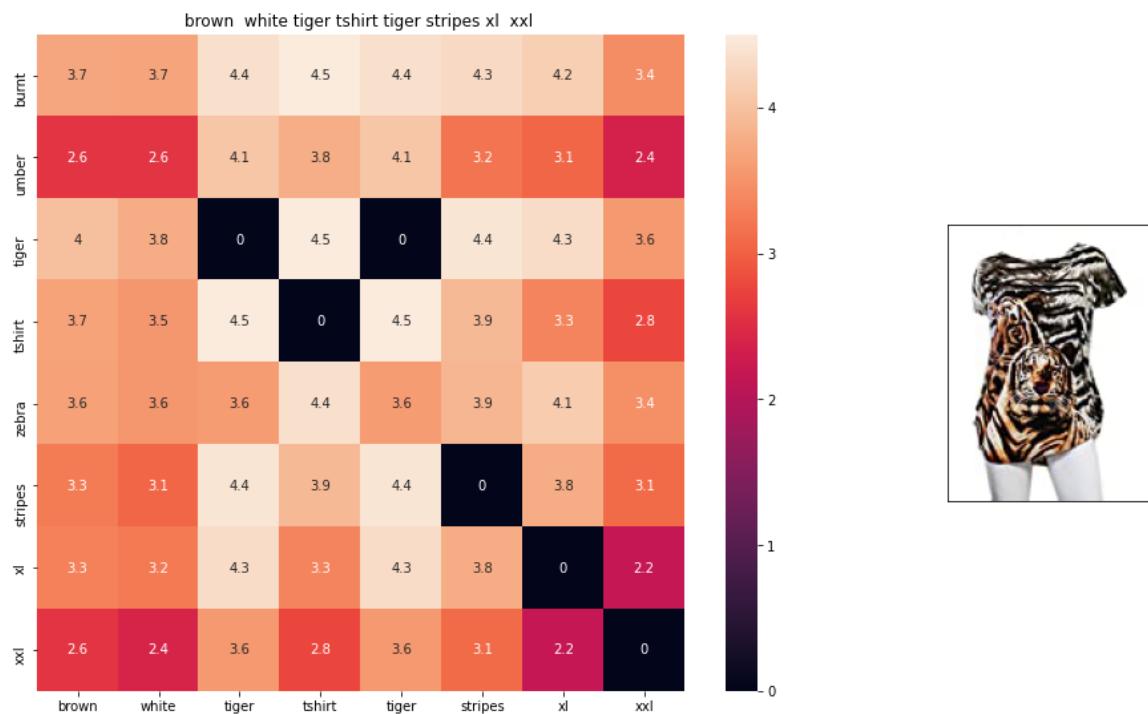
euclidean distance from given input image : 0.0



ASIN : B00JXQASS6

BRAND : Si Row

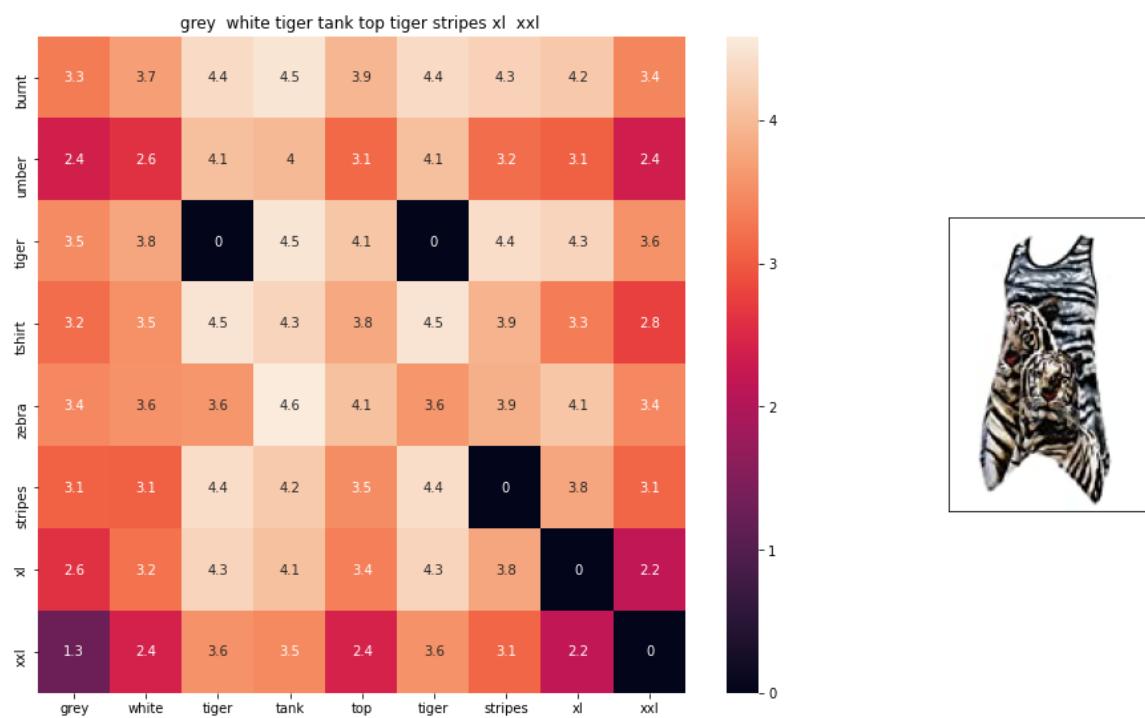
euclidean distance from given input image : 0.5891926



ASIN : B00JXQCWTO

BRAND : Si Row

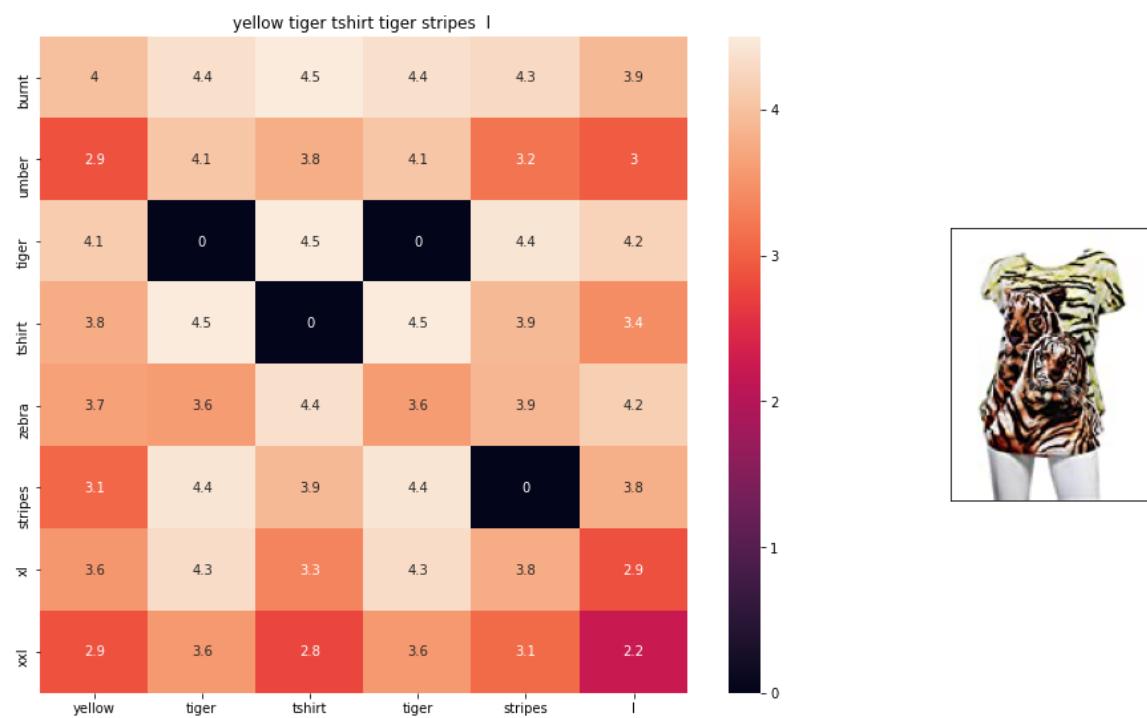
euclidean distance from given input image : 0.7003438



ASIN : B00JXQAFZ2

BRAND : Si Row

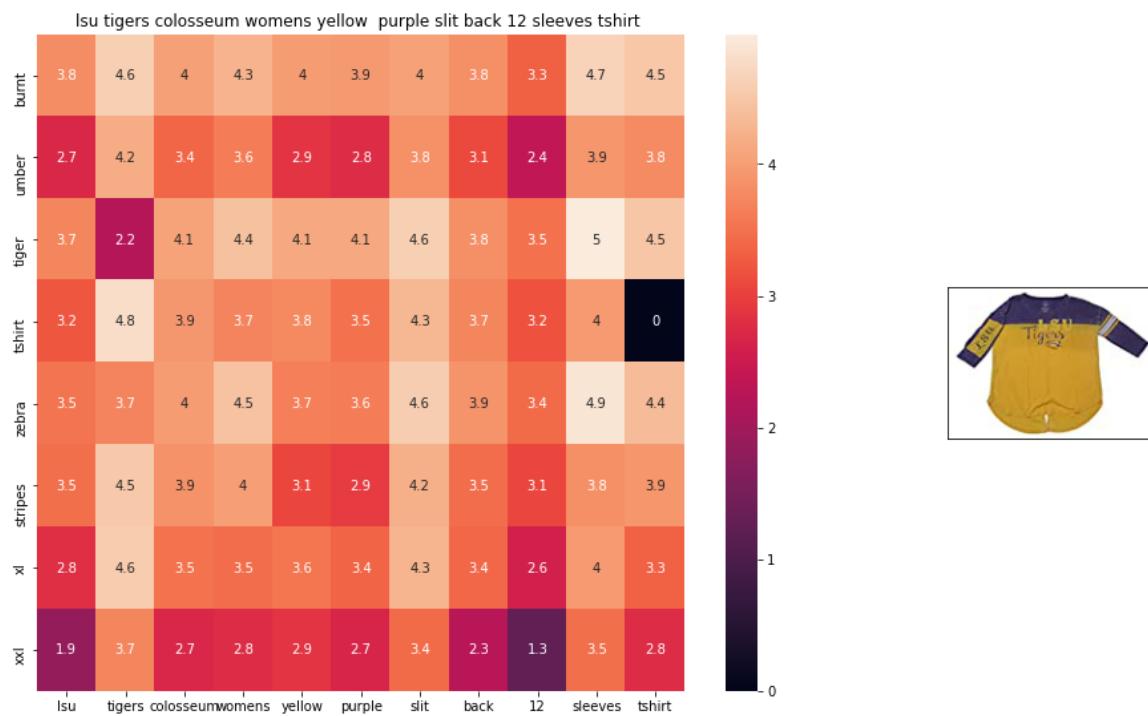
euclidean distance from given input image : 0.89283955



ASIN : B00JXQCUIC

BRAND : Si Row

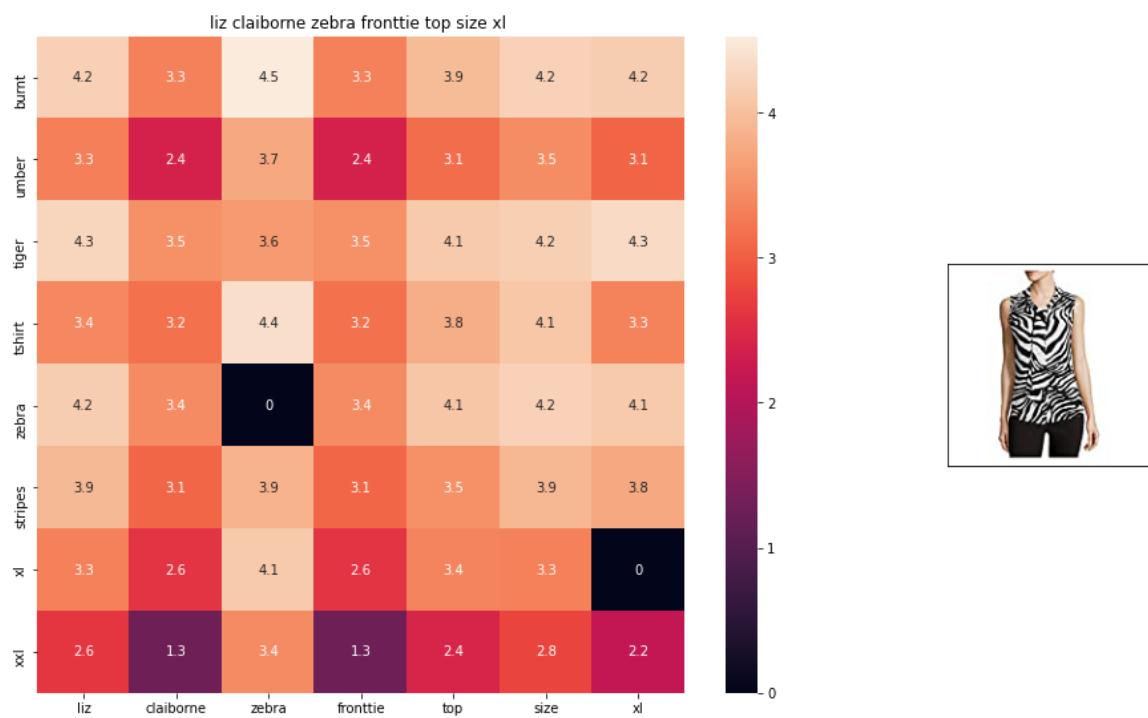
euclidean distance from given input image : 0.95601255



ASIN : B073R5Q8HD

BRAND : Colosseum

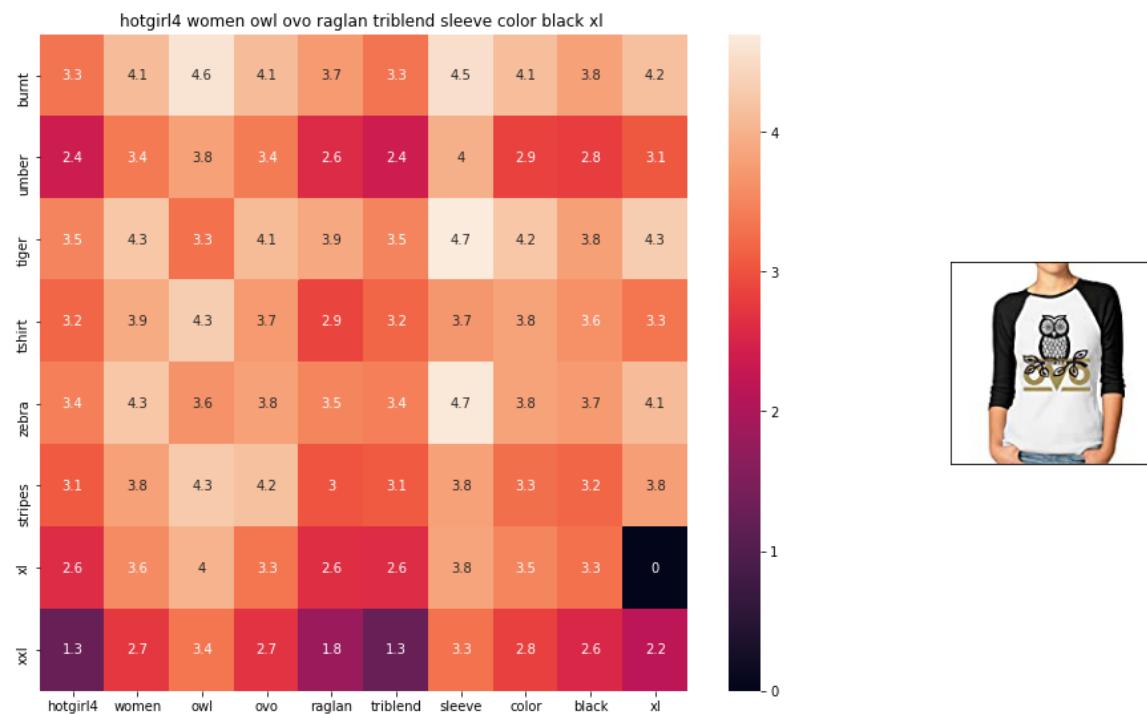
euclidean distance from given input image : 1.022969



ASIN : B06XBY5QXL

BRAND : Liz Claiborne

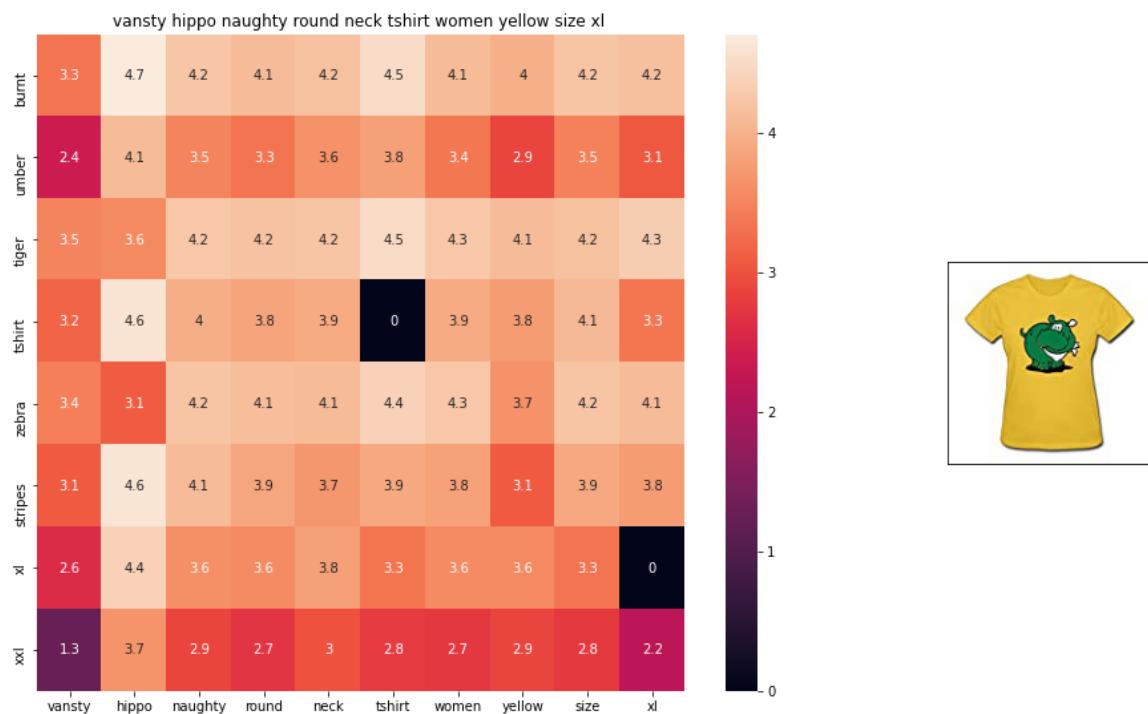
euclidean distance from given input image : 1.0669324



ASIN : B01L8L73M2

BRAND : Hotgirl4 Raglan Design

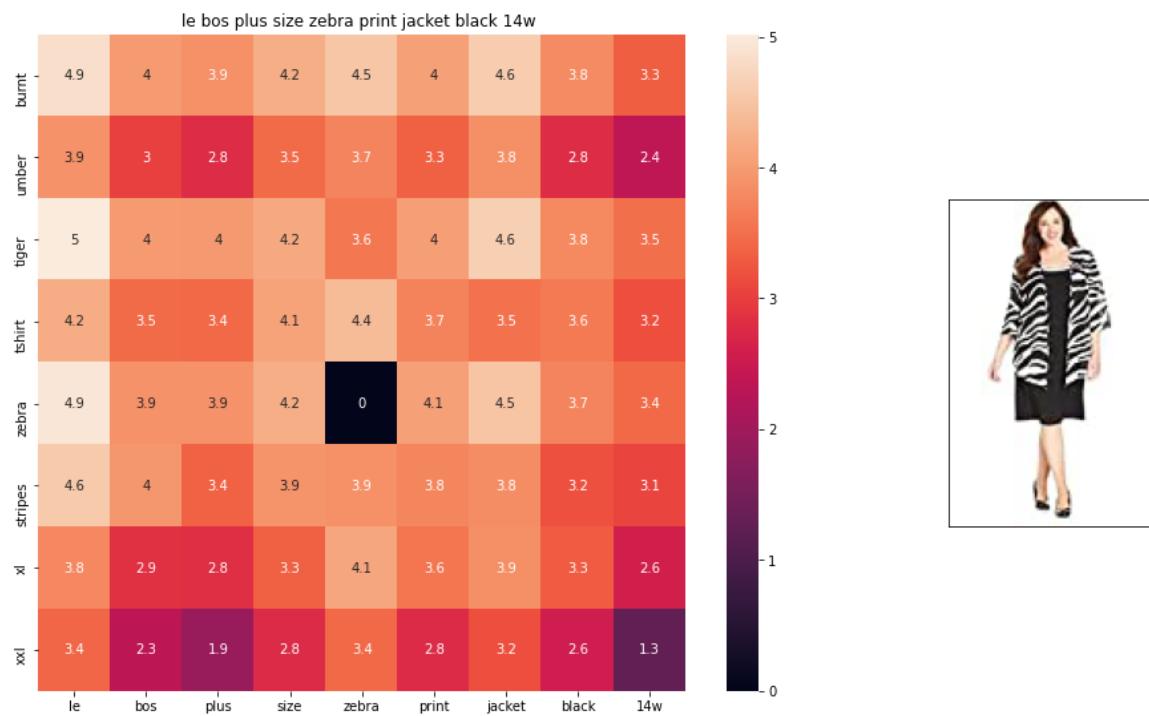
euclidean distance from given input image : 1.0731405



ASIN : B01EJS5H06

BRAND : Vansty

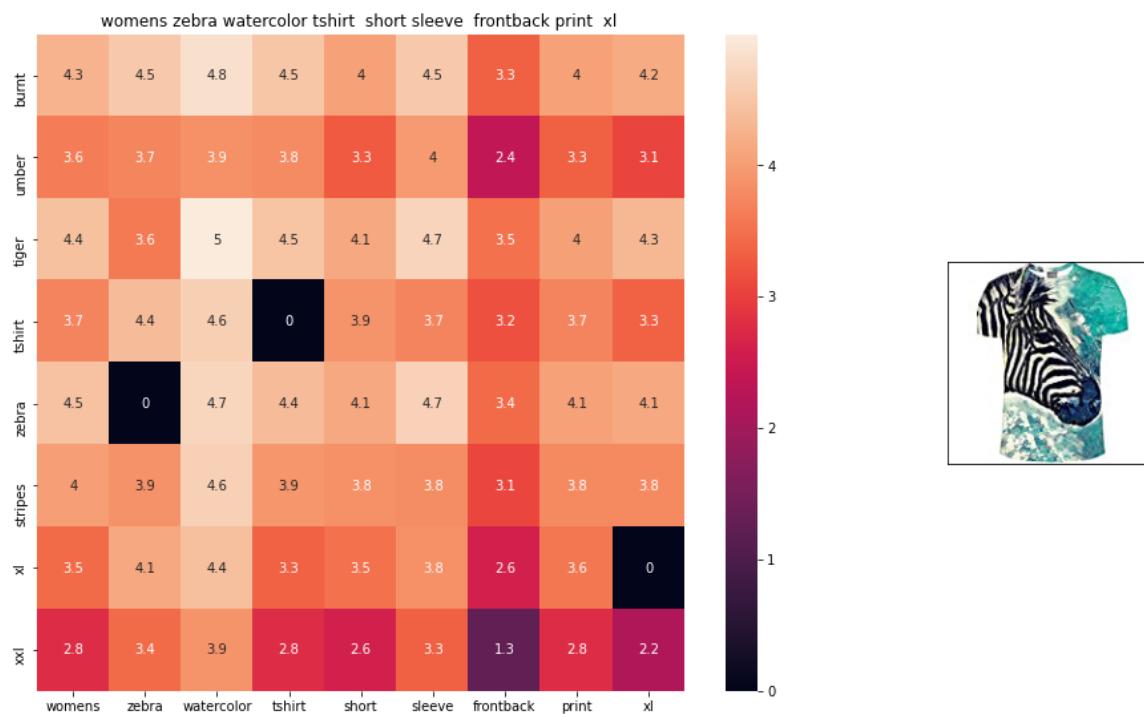
euclidean distance from given input image : 1.075719



ASIN : B01B01XRK8

BRAND : Le Bos

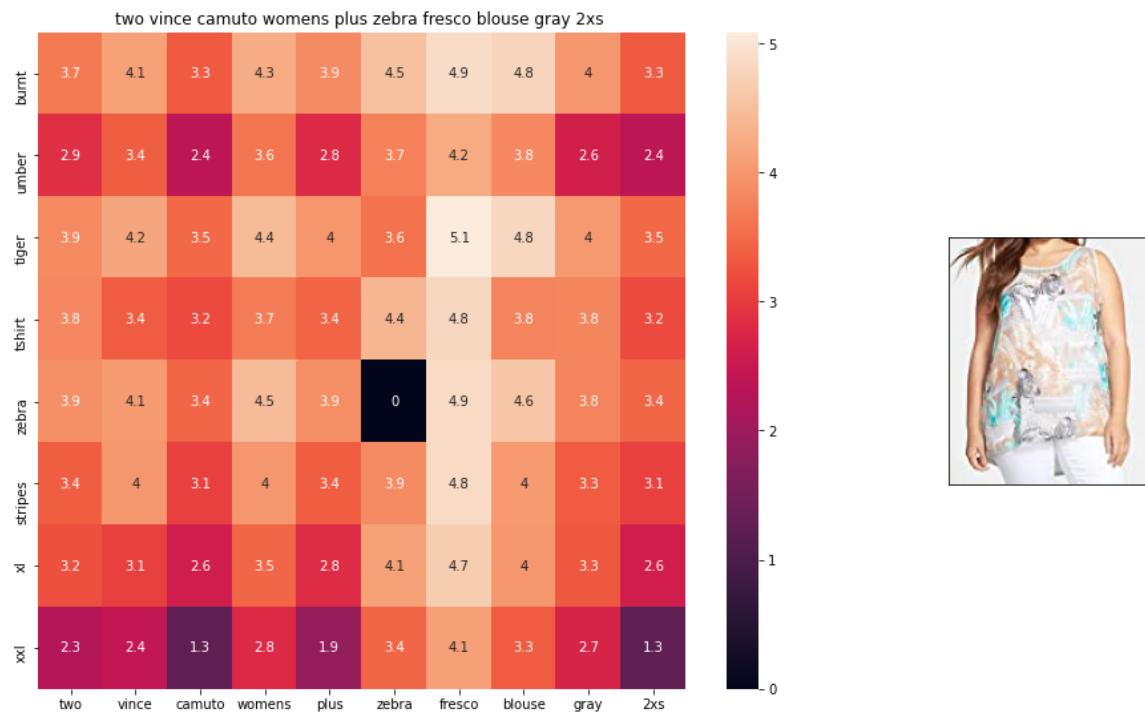
euclidean distance from given input image : 1.0839964



ASIN : B072R2JXKW

BRAND : WHAT ON EARTH

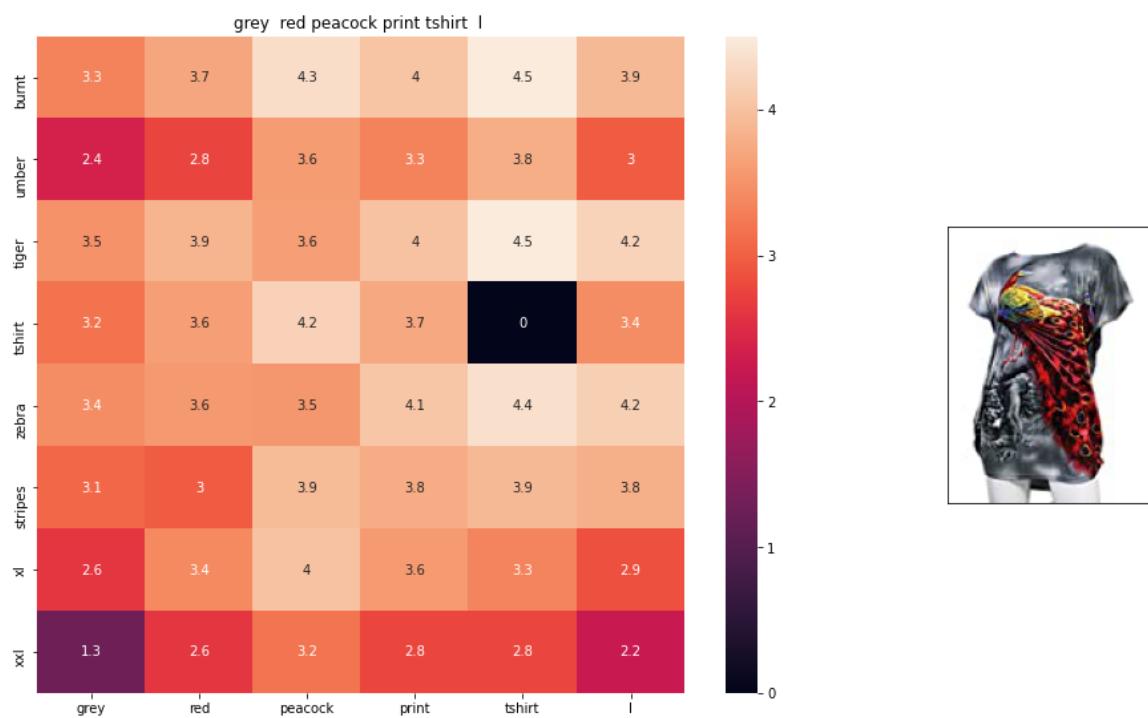
euclidean distance from given input image : 1.0842218



ASIN : B074MJRGW6

BRAND : Two by Vince Camuto

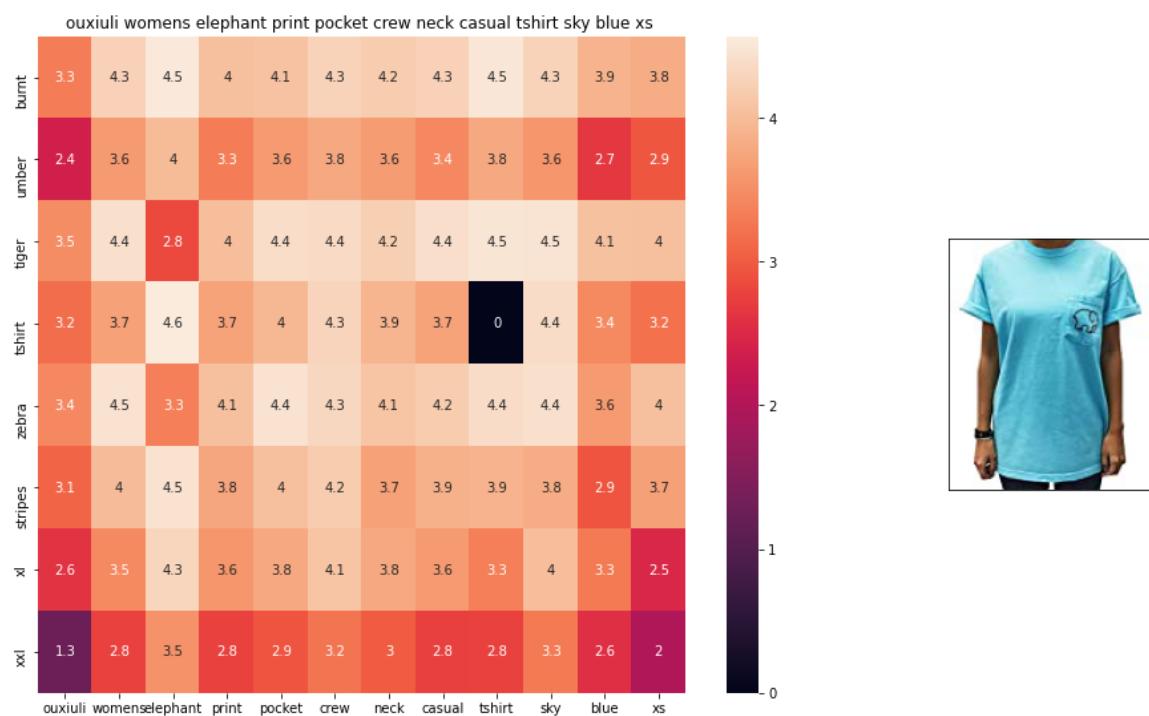
euclidean distance from given input image : 1.0895038



ASIN : B00JXQCFRS

BRAND : Si Row

euclidean distance from given input image : 1.0900588



ASIN : B01I53HU6K

BRAND : ouxiuli

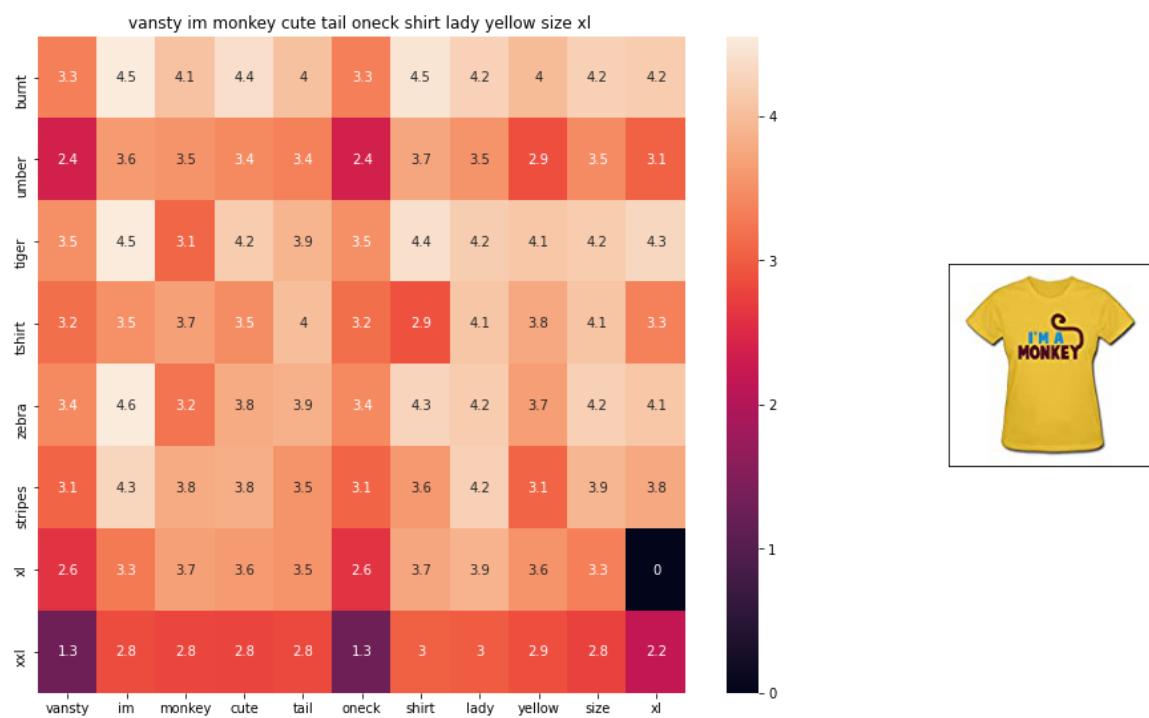
euclidean distance from given input image : 1.0920111



ASIN : B0711NGTQM

BRAND : THILFIGER RTW

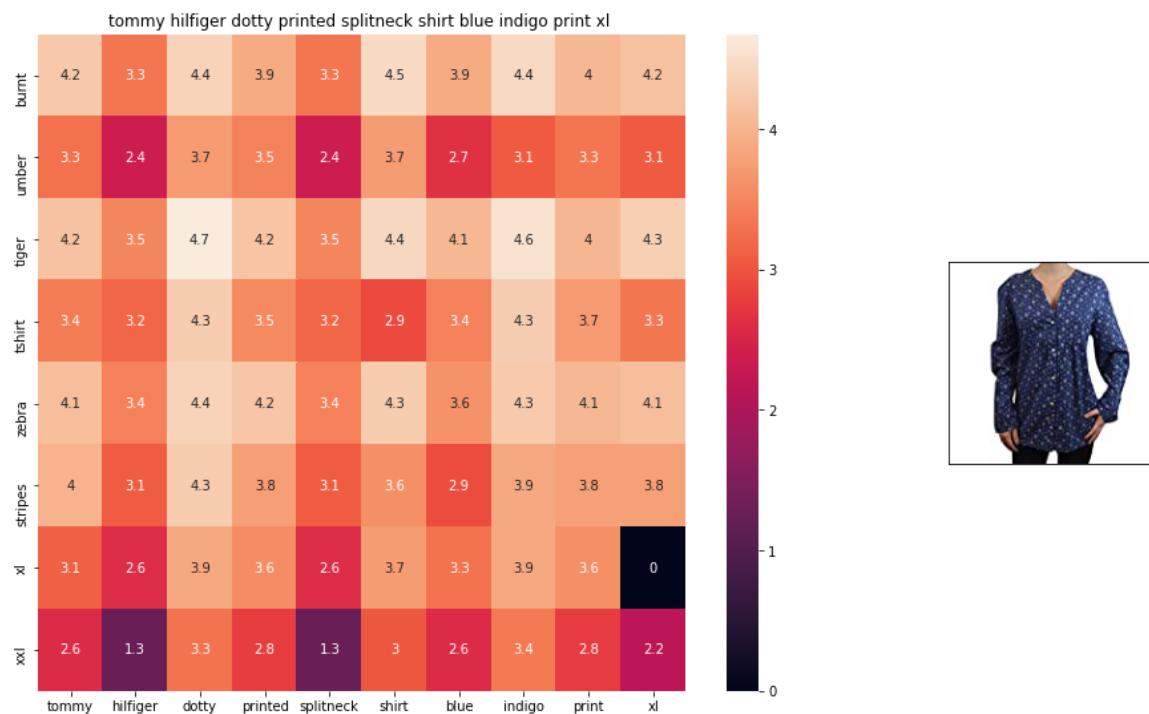
euclidean distance from given input image : 1.0923415



ASIN : B01EFSL08Y

BRAND : Vansty

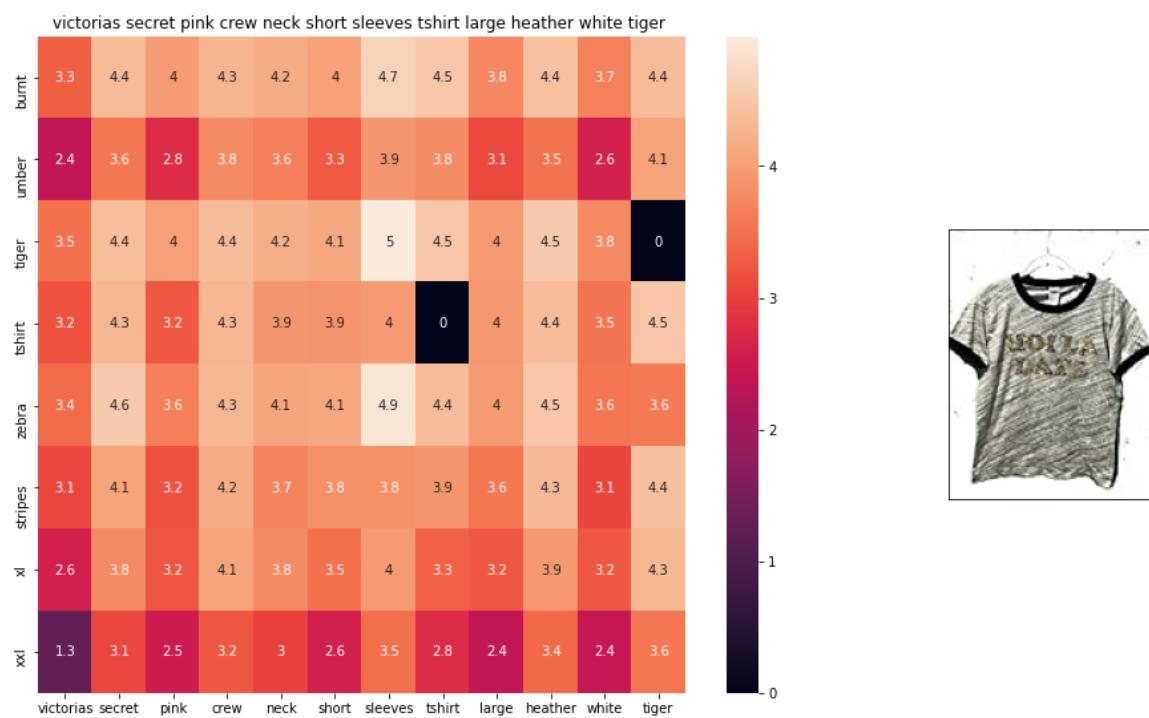
euclidean distance from given input image : 1.0934004



ASIN : B0716TVWQ4

BRAND : THILFIGER RTW

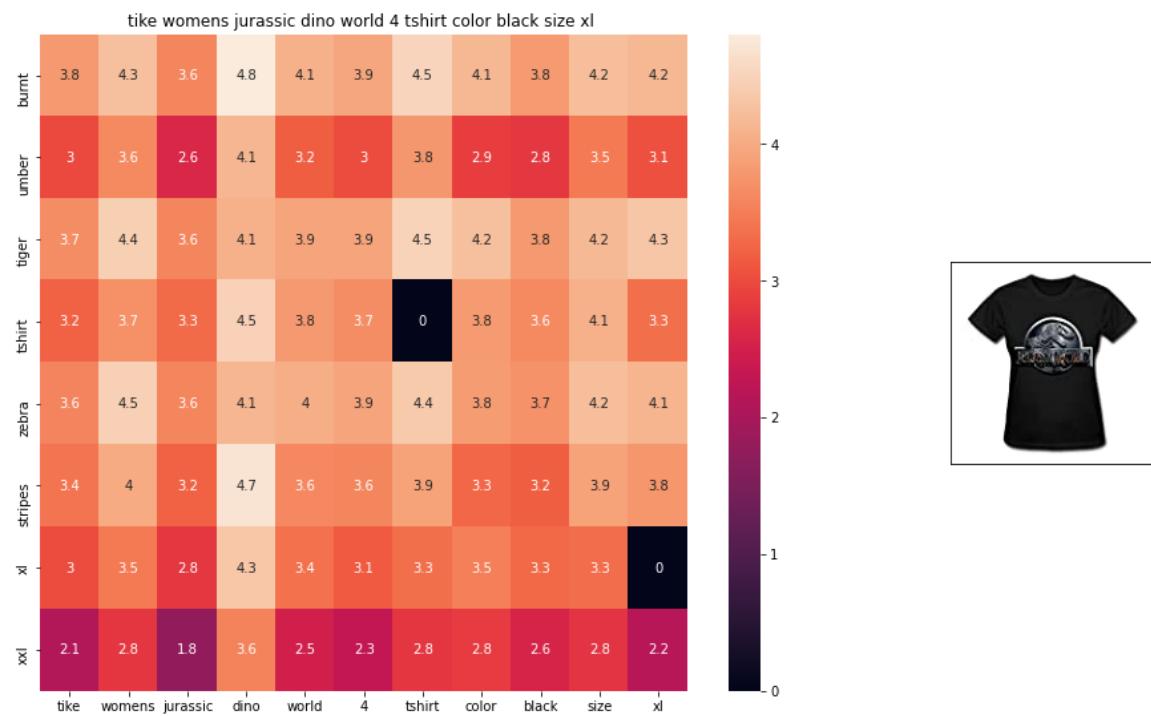
euclidean distance from given input image : 1.0942024



ASIN : B0716MVPGV

BRAND : V.Secret

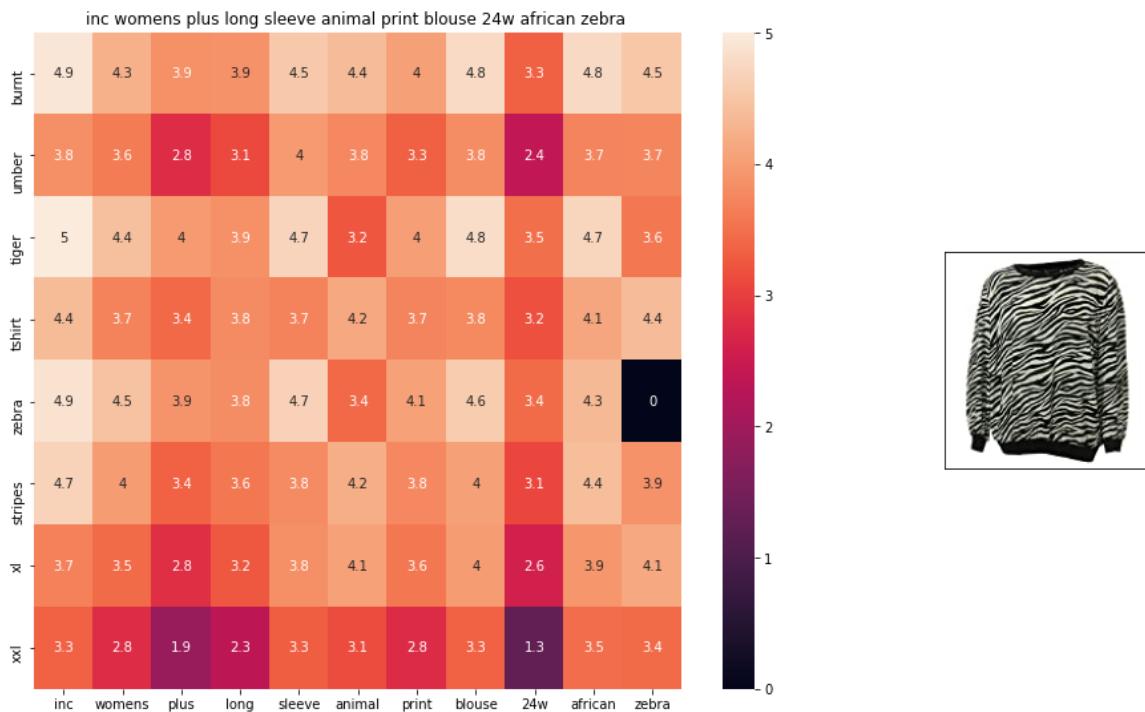
euclidean distance from given input image : 1.0948304



ASIN : B016OPN40I

BRAND : TIKE Fashions

euclidean distance from given input image : 1.0951275



ASIN : B018WDJCUA

BRAND : INC - International Concepts Woman

euclidean distance from given input image : 1.0966892

[9.4] IDF weighted Word2Vec for product similarity

In [0]:

```
doc_id = 0
w2v_title_weight = []
# for every title we build a weighted vector representation
for i in data['title']:
    w2v_title_weight.append(build_avg_vec(i, 300, doc_id,'weighted'))
    doc_id += 1
# w2v_title = np.array(# number of doc in courpus * 300), each row corresponds to a doc
w2v_title_weight = np.array(w2v_title_weight)
```

In [73]:

```
def weighted_w2v_model(doc_id, num_results):
    # doc_id: apparel's id in given corpus

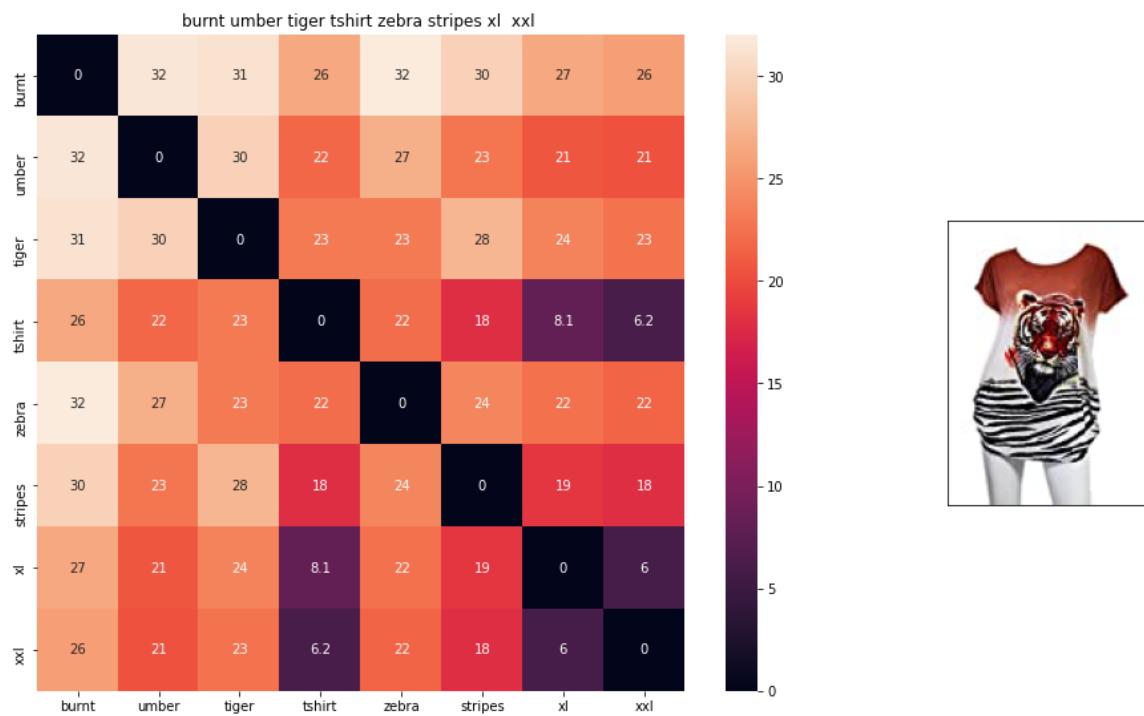
    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as  $K(X, Y) = \frac{X, Y}{\|X\| * \|Y\|}$ 
    # http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
    pairwise_dist = pairwise_distances(w2v_title_weight, w2v_title_weight[doc_id].reshape(1,-1))

    # np.argsort will return indices of 9 smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the 9 smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    #data frame indices of the 9 smallest distace's
    df_indices = list(data.index[indices])

    for i in range(0, len(indices)):
        heat_map_w2v(data['title'].loc[df_indices[0]], data['title'].loc[df_indices[i]],
data['medium_image_url'].loc[df_indices[i]], indices[0], indices[i], 'weighted')
        print('ASIN :', data['asin'].loc[df_indices[i]])
        print('Brand :', data['brand'].loc[df_indices[i]])
        print('euclidean distance from input :', pdists[i])
        print('='*125)

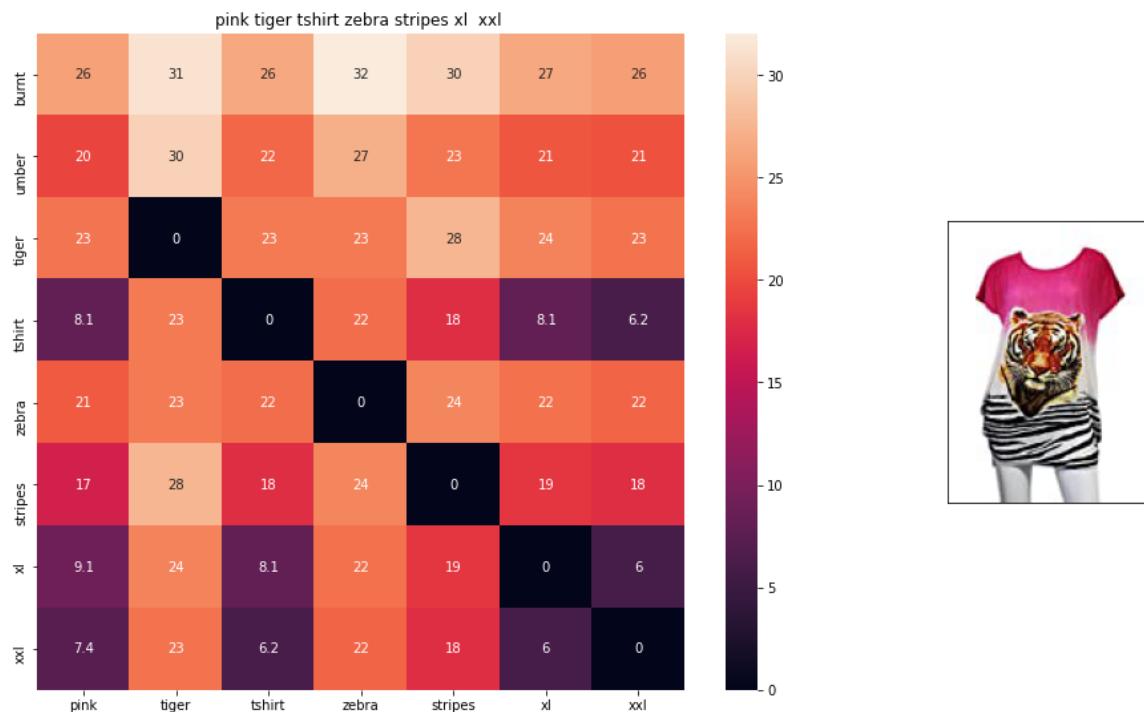
weighted_w2v_model(12566, 20)
#931
#12566
# in the give heat map, each cell contains the euclidean distance between words i, j
```



ASIN : B00JXQB5FQ

Brand : Si Row

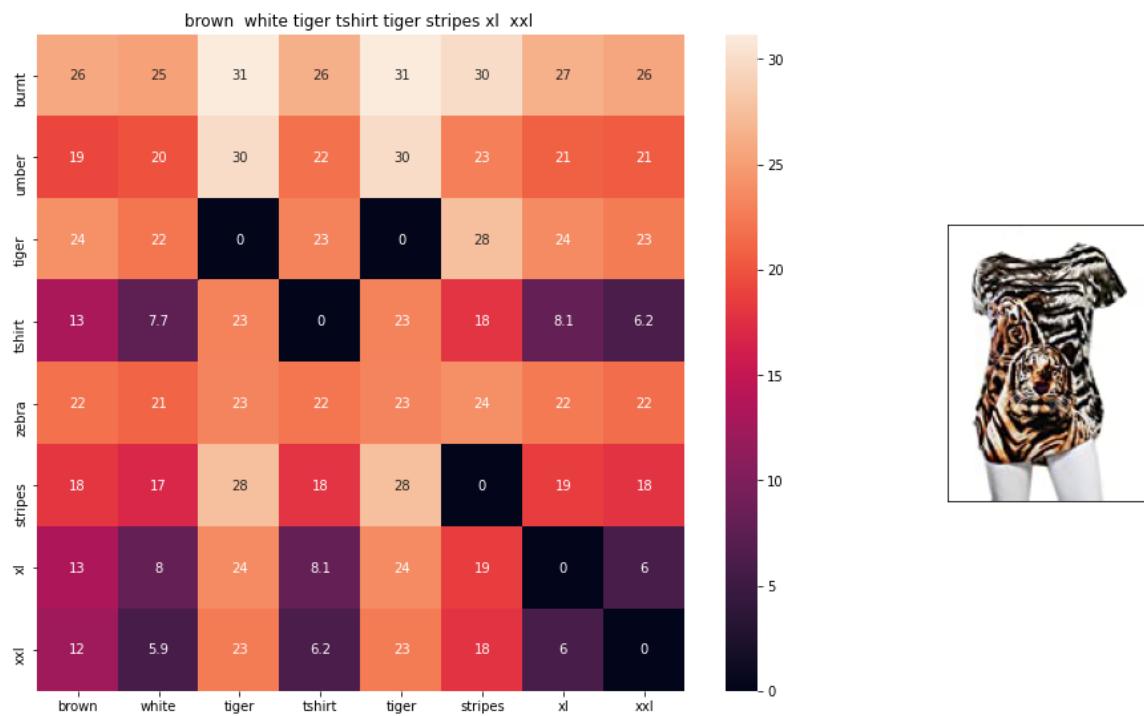
euclidean distance from input : 0.0



ASIN : B00JXQASS6

Brand : Si Row

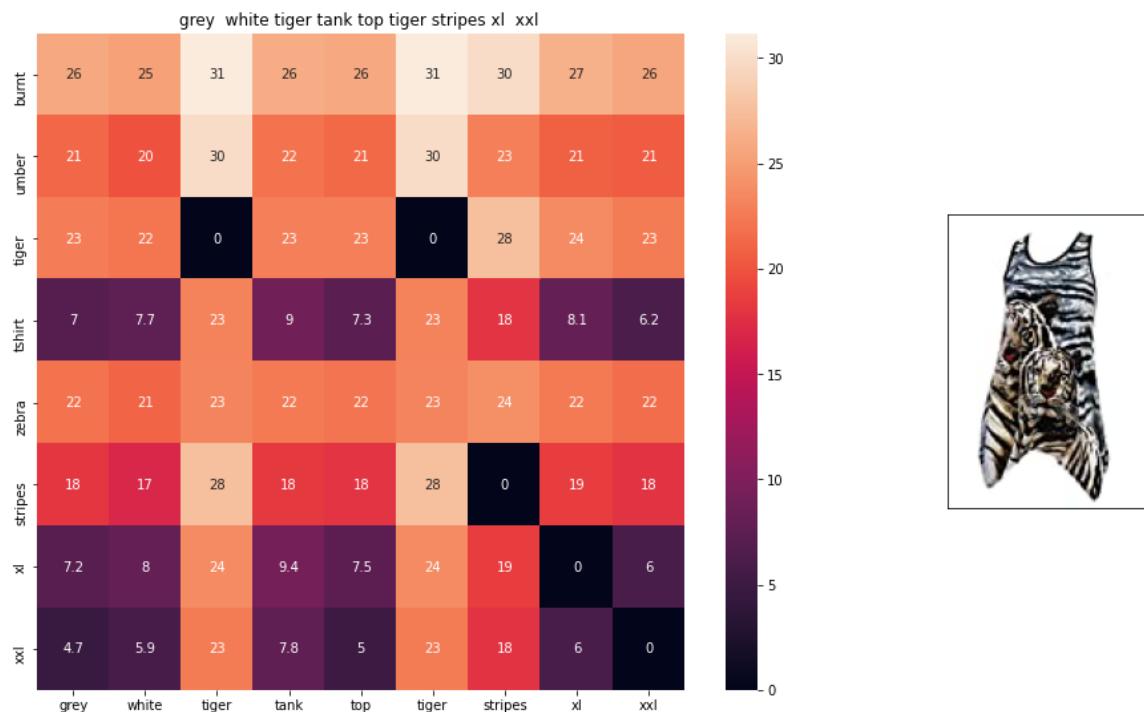
euclidean distance from input : 4.0638866



ASIN : B00JXQCWT0

Brand : Si Row

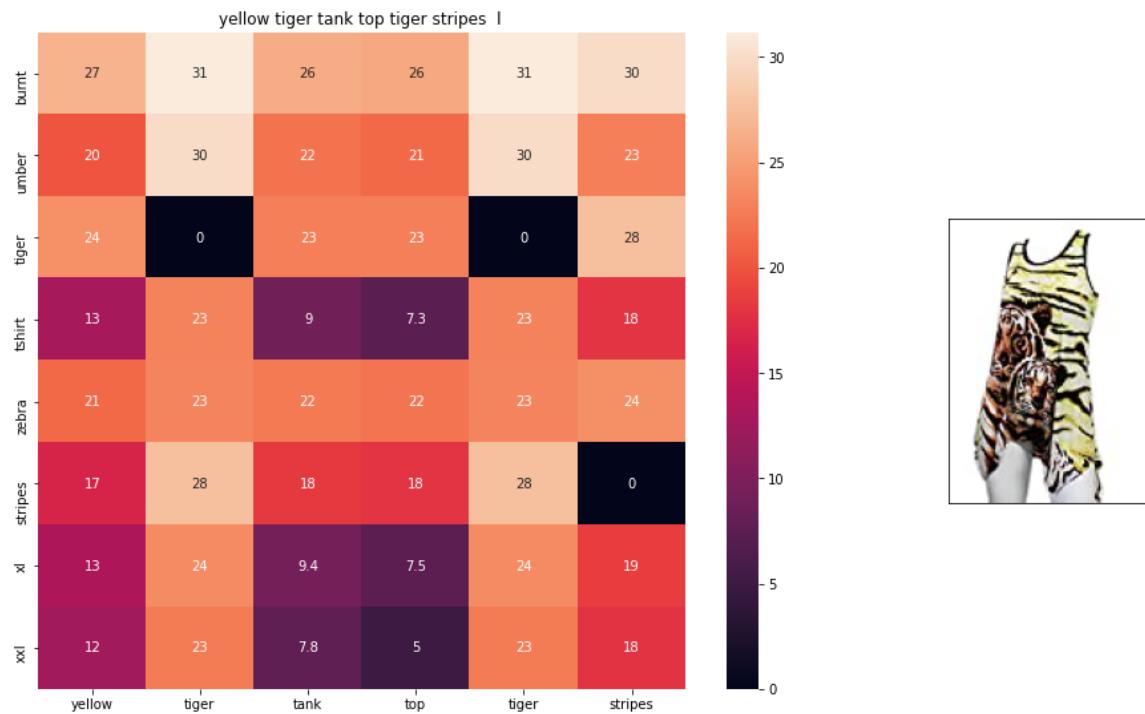
euclidean distance from input : 4.7709413



ASIN : B00JXQAFZ2

Brand : Si Row

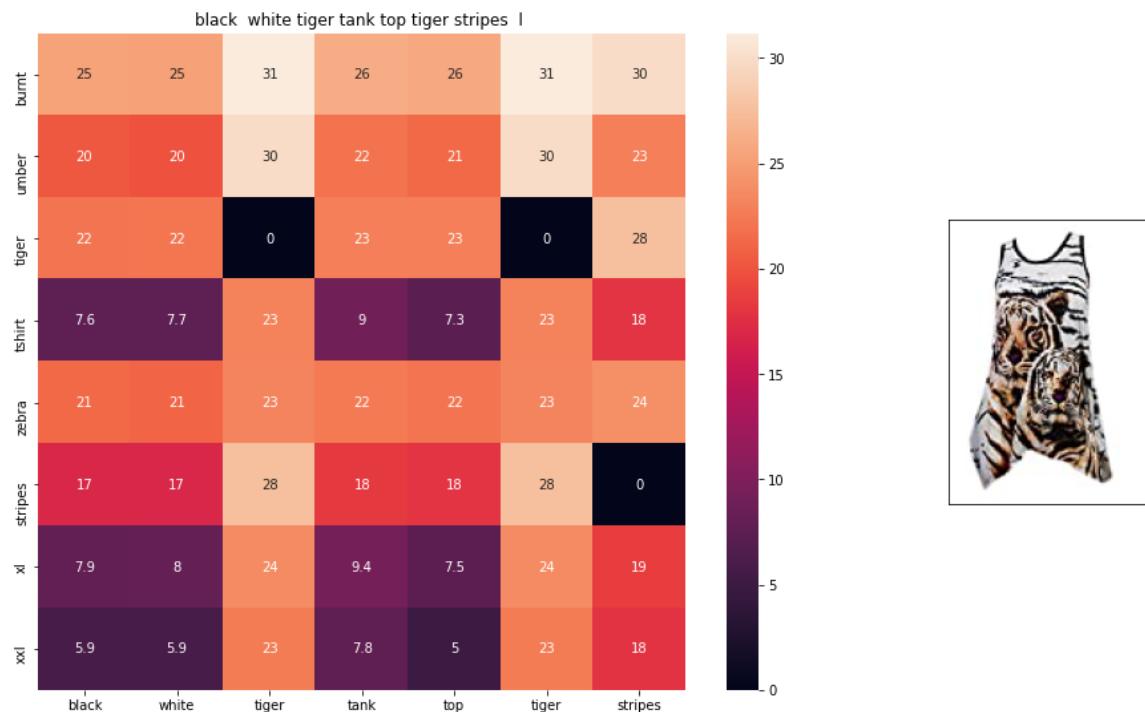
euclidean distance from input : 5.3601604



ASIN : B00JXQAUWA

Brand : Si Row

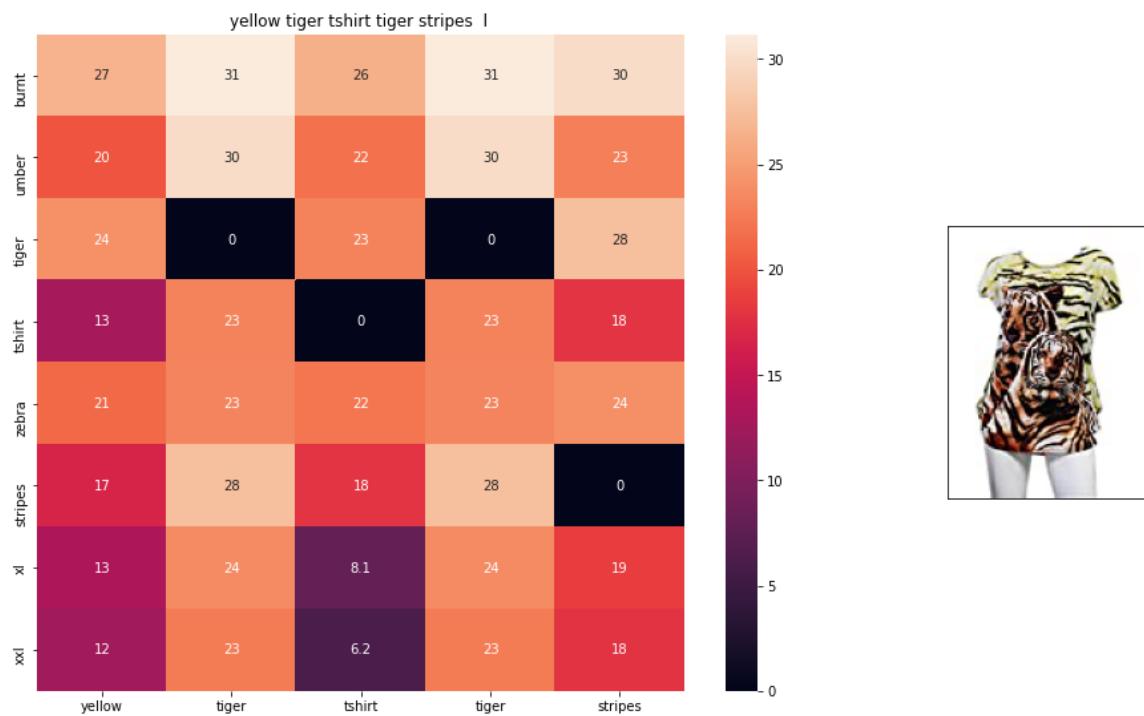
euclidean distance from input : 5.6895227



ASIN : B00JXQA094

Brand : Si Row

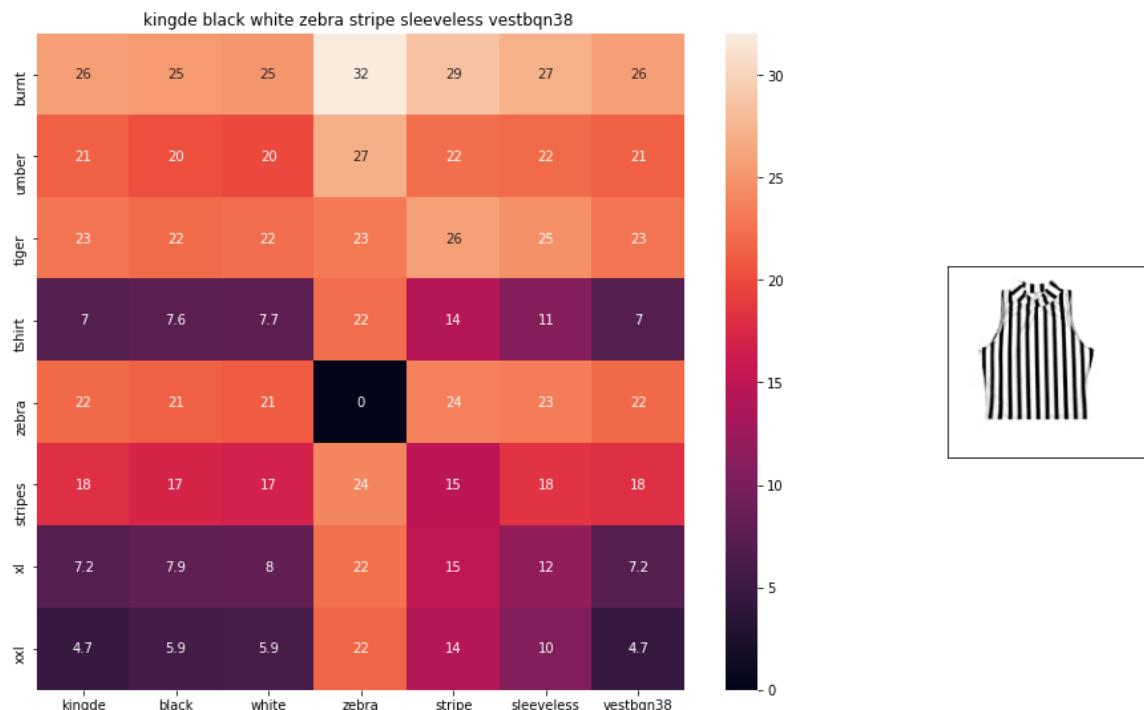
euclidean distance from input : 5.693021



ASIN : B00JXQCUIC

Brand : Si Row

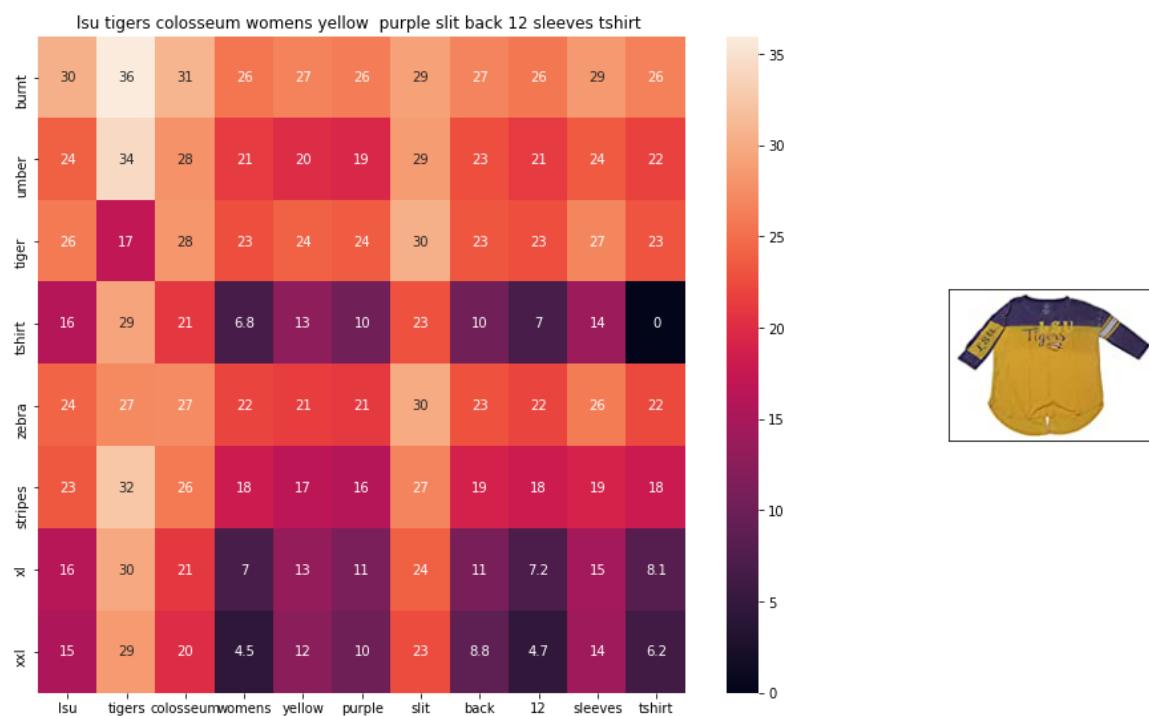
euclidean distance from input : 5.893442



ASIN : B015H41F6G

Brand : KINGDE

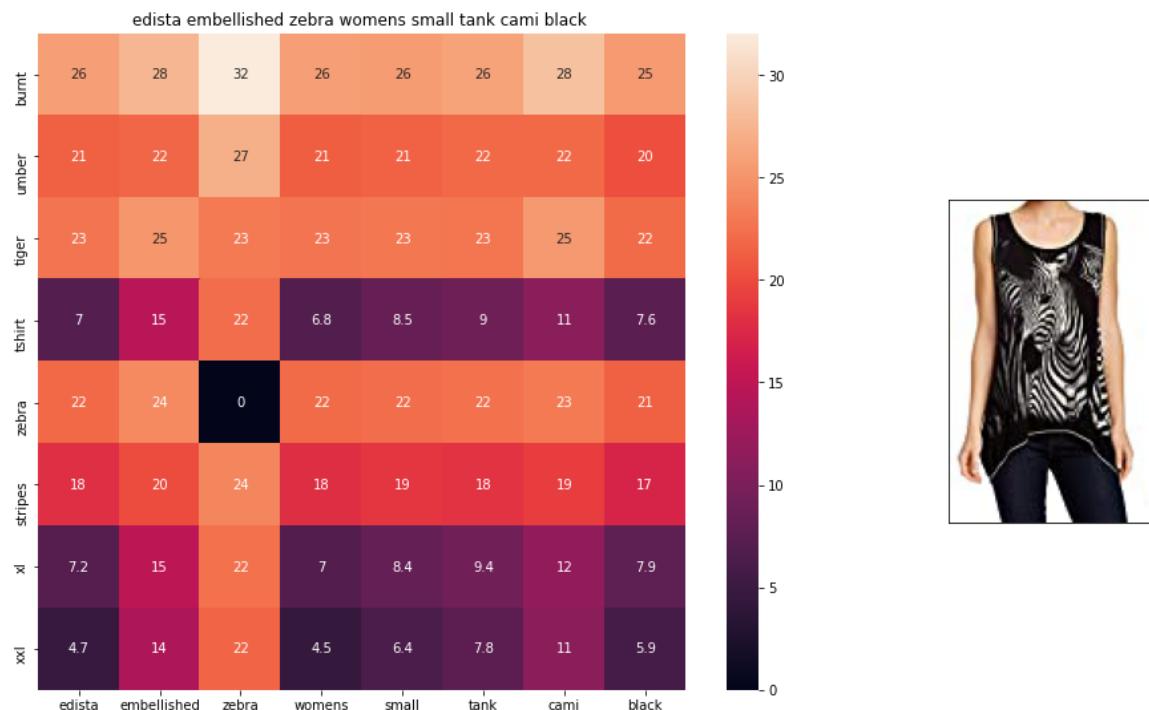
euclidean distance from input : 6.1329894



ASIN : B073R5Q8HD

Brand : Colosseum

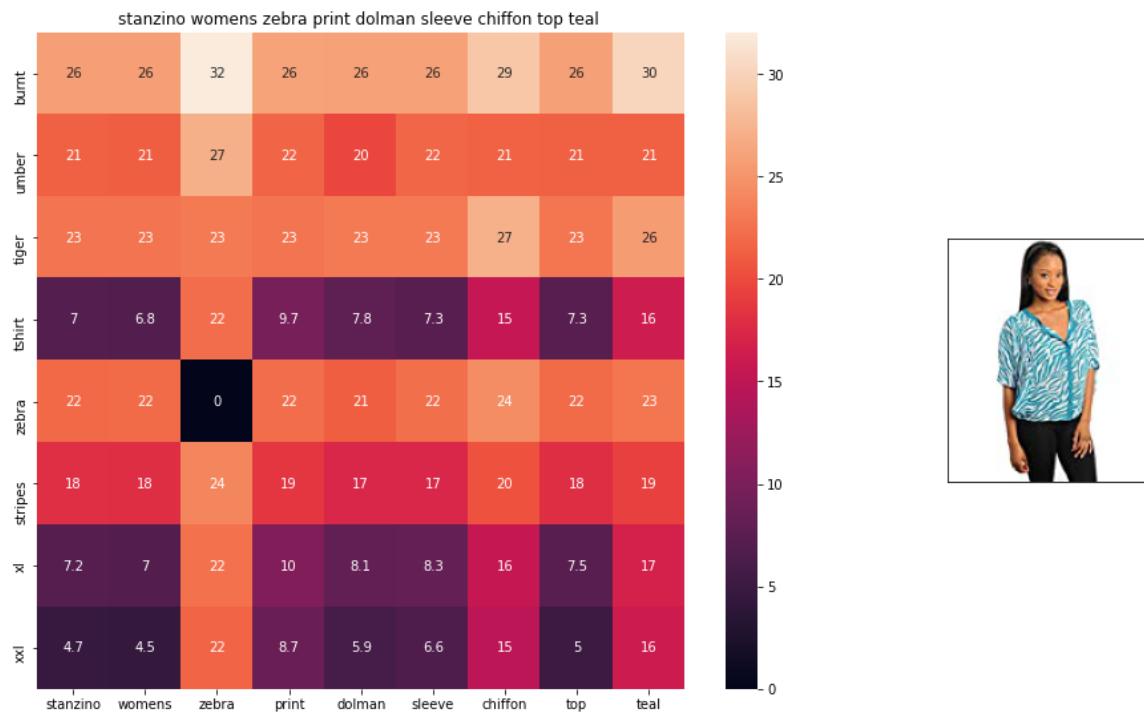
euclidean distance from input : 6.2567053



ASIN : B074P8MD22

Brand : Edista

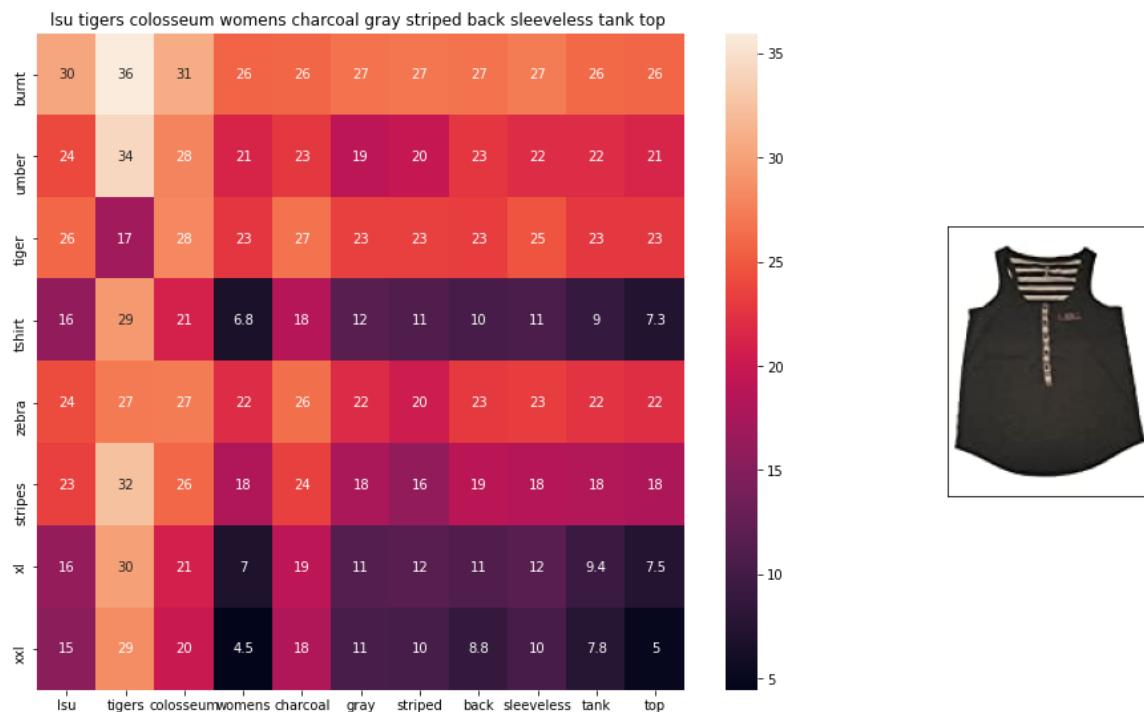
euclidean distance from input : 6.3922033



ASIN : B00C0I3U3E

Brand : Stanzino

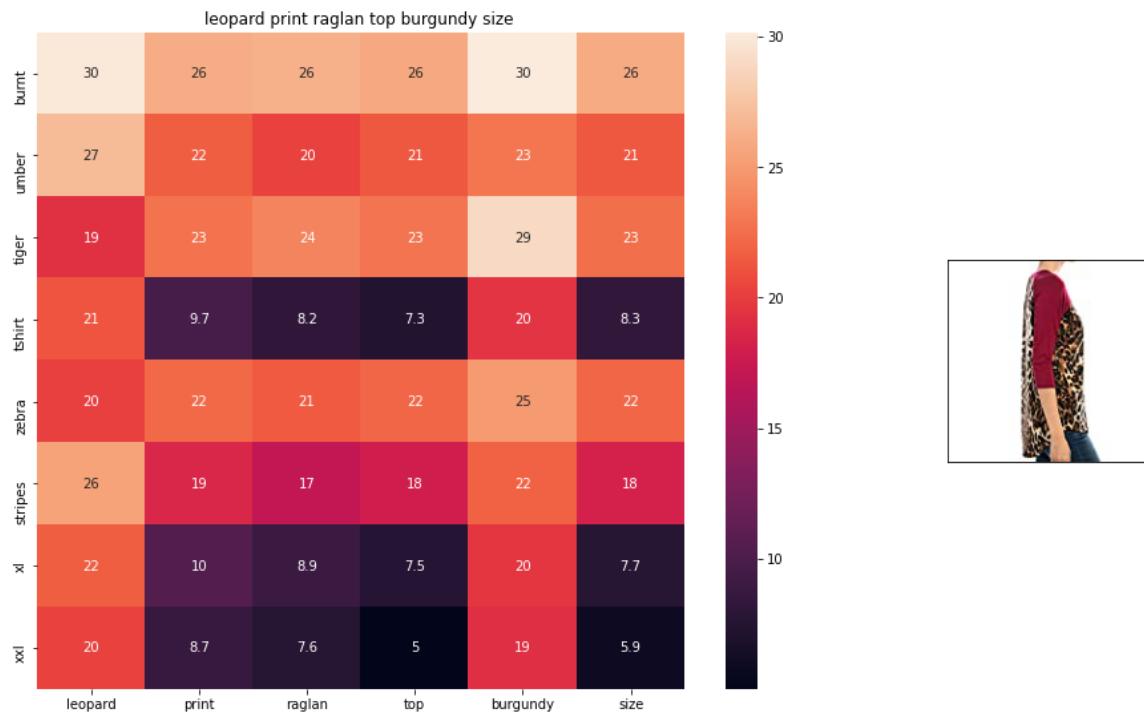
euclidean distance from input : 6.4149003



ASIN : B073R4ZM7Y

Brand : Colosseum

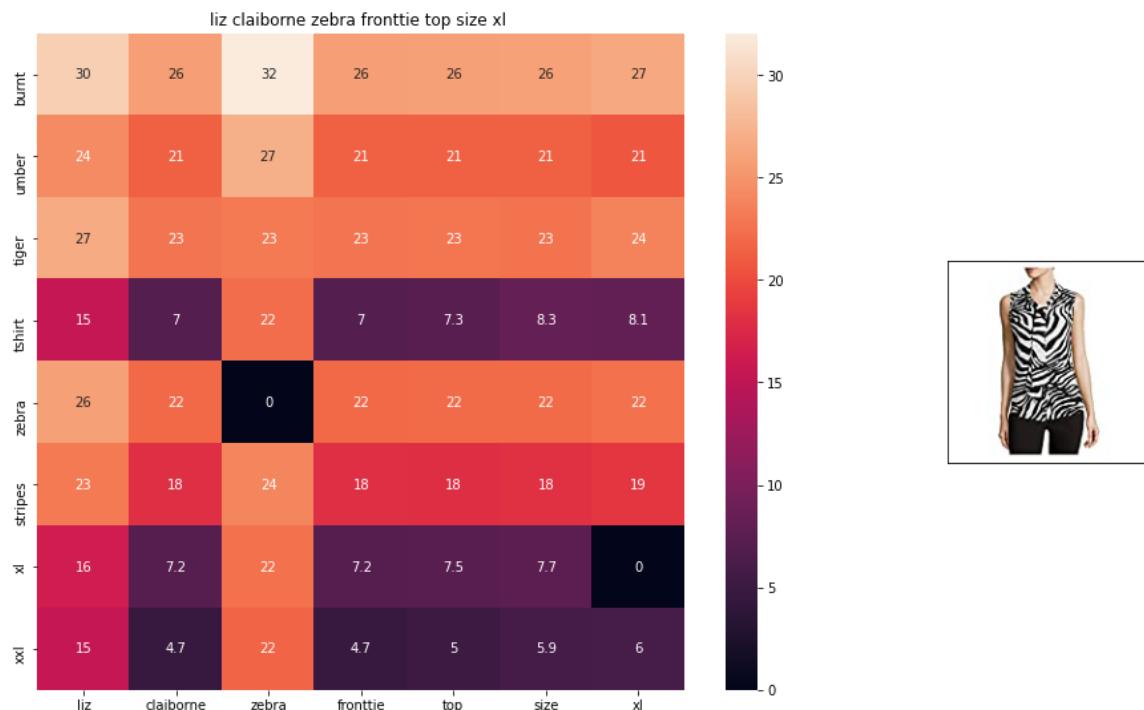
euclidean distance from input : 6.450959



ASIN : B01C60RLDQ

Brand : 1 Mad Fit

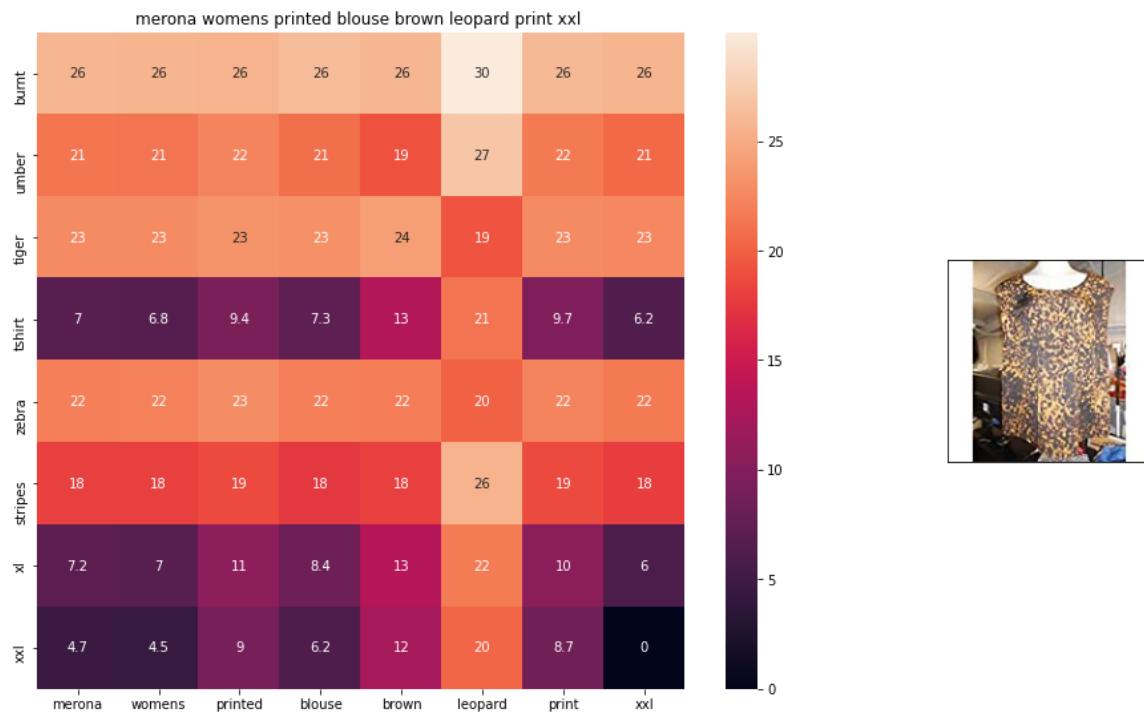
euclidean distance from input : 6.463409



ASIN : B06XBY5QXL

Brand : Liz Claiborne

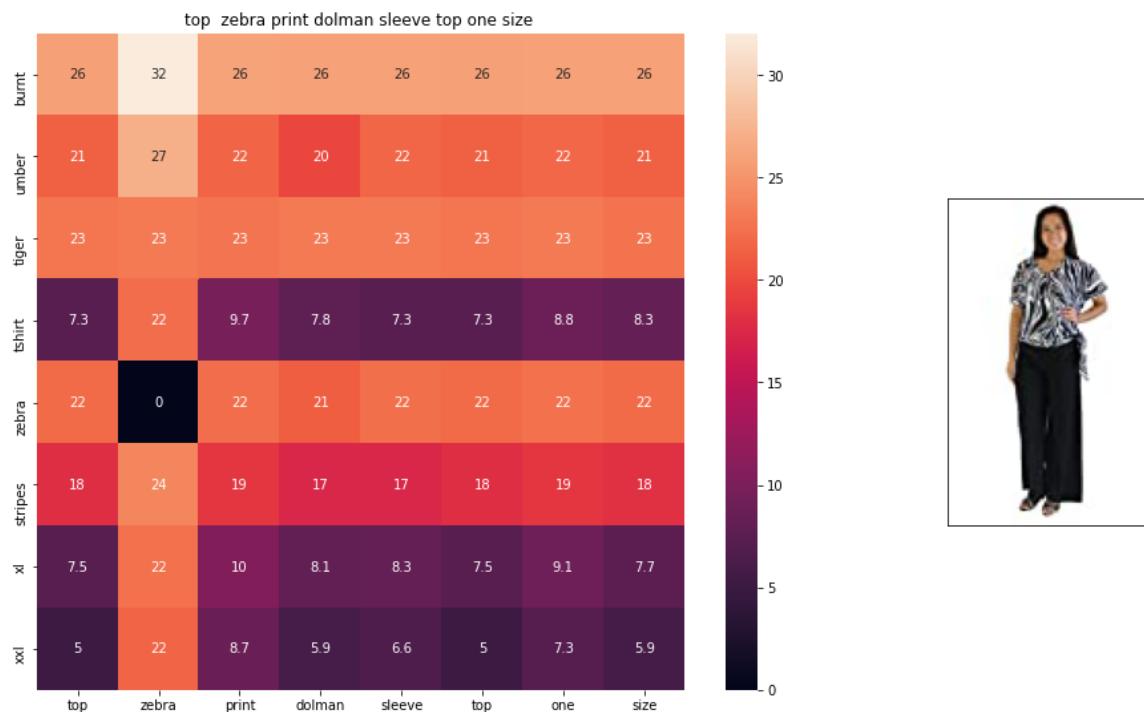
euclidean distance from input : 6.5392227



ASIN : B071YF3WDD

Brand : Merona

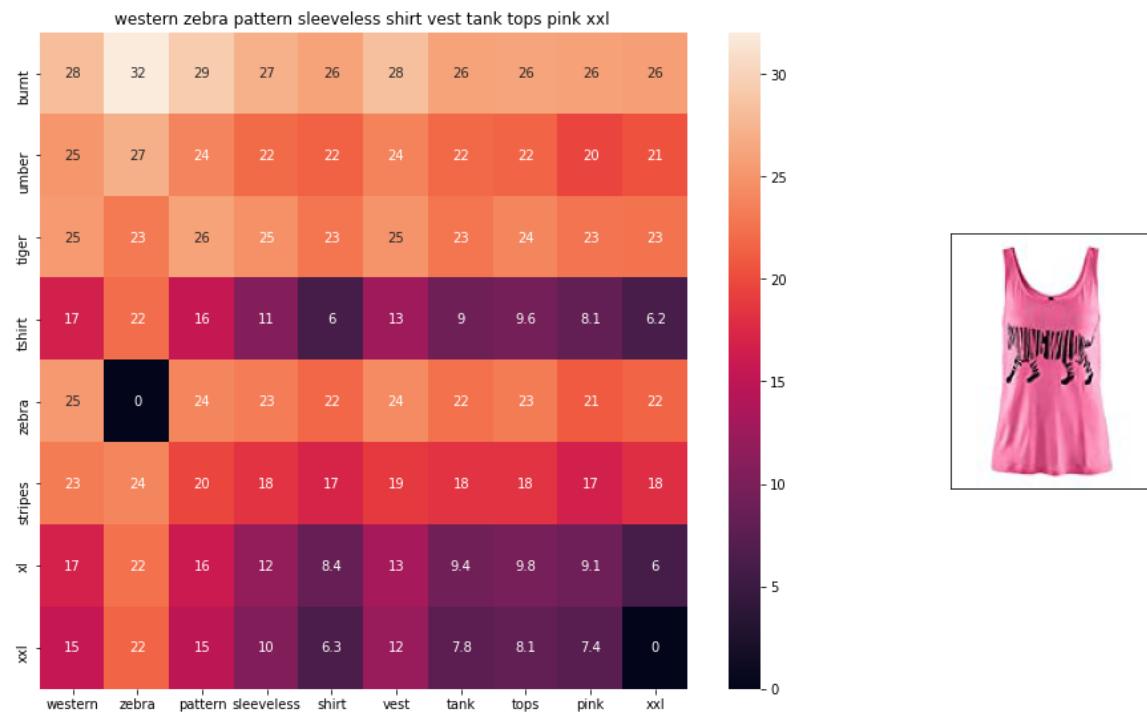
euclidean distance from input : 6.5755024



ASIN : B00H8A6ZLI

Brand : Vivian's Fashions

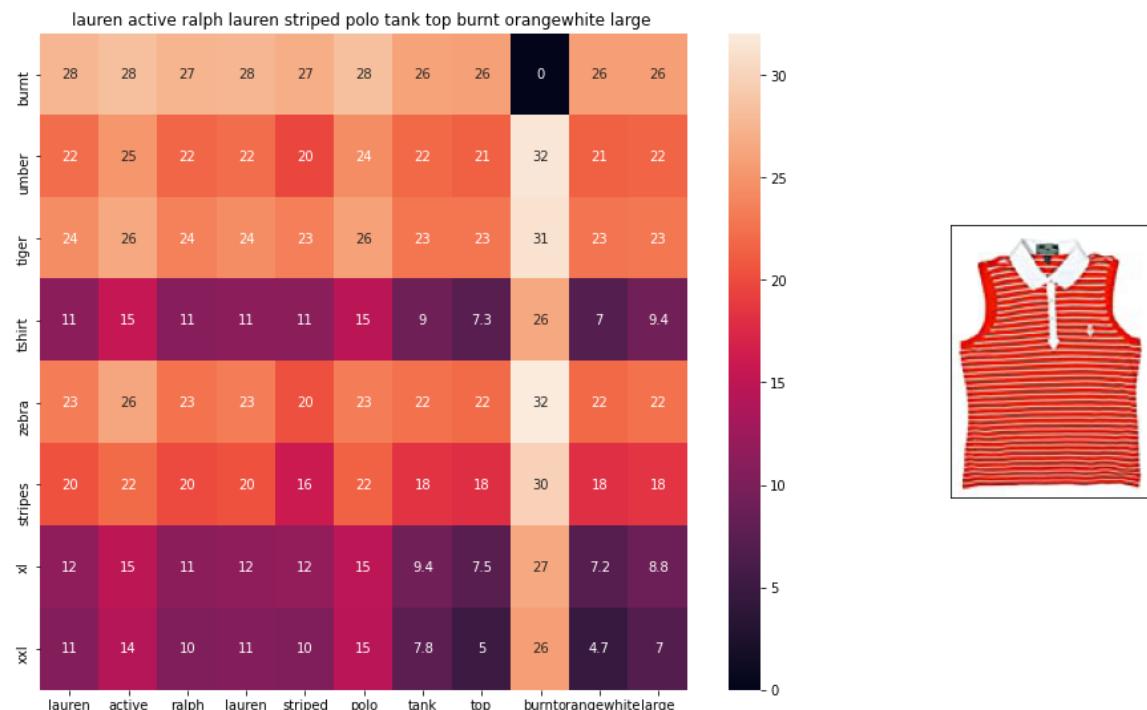
euclidean distance from input : 6.6382146



ASIN : B00Z6HEXWI

Brand : Black Temptation

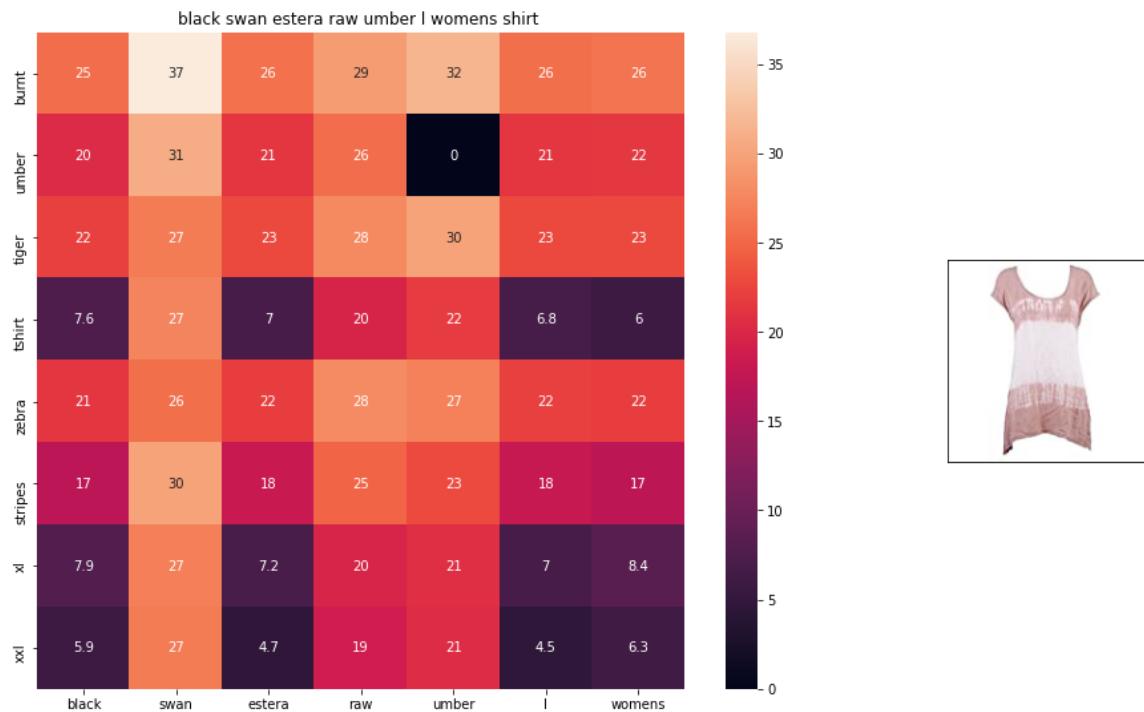
euclidean distance from input : 6.6607366



ASIN : B00ILGH50Y

Brand : Ralph Lauren Active

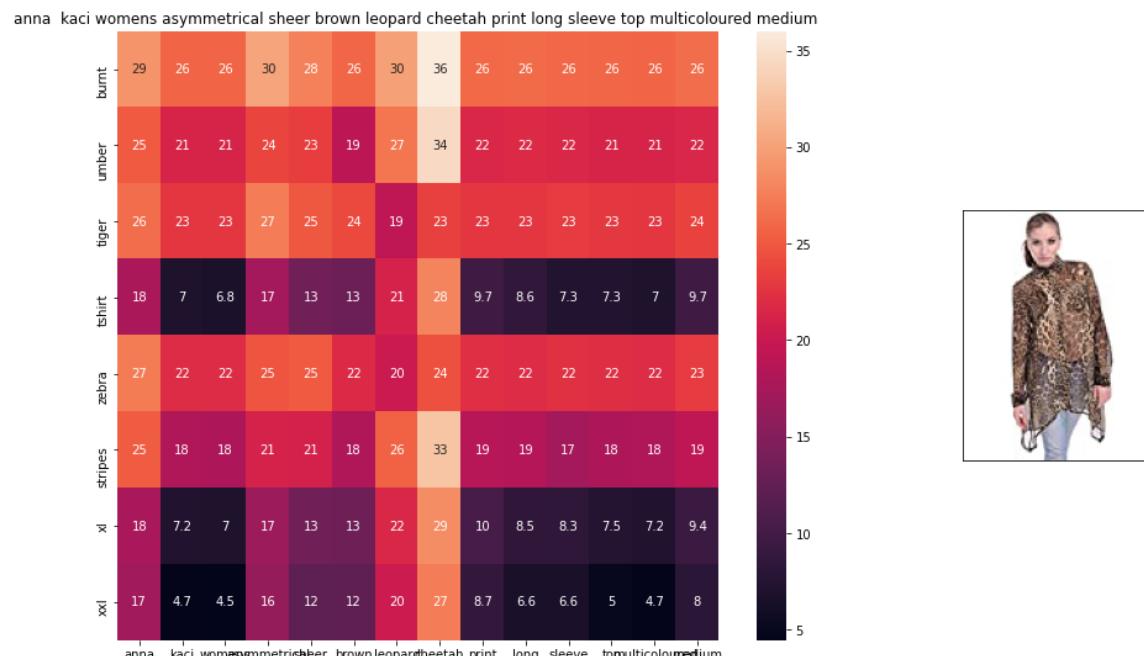
euclidean distance from input : 6.6839046



ASIN : B06Y1VN8WQ

Brand : Black Swan

euclidean distance from input : 6.705763



ASIN : B00KSNTY7Y

Brand : Anna-Kaci

euclidean distance from input : 6.7061243

[9.6] Weighted similarity using brand and color.

In [0]:

```
# some of the brand values are empty.  
# Need to replace Null with string "NULL"  
data['brand'].fillna(value="Not given", inplace=True )  
  
# replace spaces with hyphen  
brands = [x.replace(" ", "-") for x in data['brand'].values]  
types = [x.replace(" ", "-") for x in data['product_type_name'].values]  
colors = [x.replace(" ", "-") for x in data['color'].values]  
  
brand_vectorizer = CountVectorizer()  
brand_features = brand_vectorizer.fit_transform(brands)  
  
type_vectorizer = CountVectorizer()  
type_features = type_vectorizer.fit_transform(types)  
  
color_vectorizer = CountVectorizer()  
color_features = color_vectorizer.fit_transform(colors)  
  
extra_features = hstack((brand_features, type_features, color_features)).tocsr()
```

In [0]:

```

def heat_map_w2v_brand(sentance1, sentance2, url, doc_id1, doc_id2, df_id1, df_id2, model):
    # sentance1 : title1, input apparel
    # sentance2 : title2, recommended apparel
    # url: apparel image url
    # doc_id1: document id of input apparel
    # doc_id2: document id of recommended apparel
    # df_id1: index of document1 in the data frame
    # df_id2: index of document2 in the data frame
    # model: it can have two values, 1. avg 2. weighted

    #s1_vec = np.array(#number_of_words_title1 * 300), each row is a vector(weighted/avg)
    # of length 300 corresponds to each word in give title
    s1_vec = get_word_vec(sentance1, doc_id1, model)
    #s2_vec = np.array(#number_of_words_title2 * 300), each row is a vector(weighted/avg)
    # of length 300 corresponds to each word in give title
    s2_vec = get_word_vec(sentance2, doc_id2, model)

    # s1_s2_dist = np.array(#number of words in title1 * #number of words in title2)
    # s1_s2_dist[i,j] = euclidean distance between words i, j
    s1_s2_dist = get_distance(s1_vec, s2_vec)

    data_matrix = [['Asin', 'Brand', 'Color', 'Product type'],
                   [data['asin'].loc[df_id1], brands[doc_id1], colors[doc_id1], types[doc_id1]], # input apparel's features
                   [data['asin'].loc[df_id2], brands[doc_id2], colors[doc_id2], types[doc_id2]]] # recommended apparel's features

    colorscale = [[0, '#1d004d'], [.5, '#f2e5ff'], [1, '#f2e5d1']] # to color the headings of each column

    # we create a table with the data_matrix
    table = ff.create_table(data_matrix, index=True, colorscale=colorscale)
    # plot it with plotly
    plotly.offline.iplot(table, filename='simple_table')

    # devide whole figure space into 25 * 1:10 grids
    gs = gridspec.GridSpec(25, 15)
    fig = plt.figure(figsize=(25,5))

    # in first 25*10 grids we plot heatmap
    ax1 = plt.subplot(gs[:, :-5])
    # plotting the heap map based on the pairwise distances
    ax1 = sns.heatmap(np.round(s1_s2_dist,6), annot=True)
    # set the x axis labels as recommended apparels title
    ax1.set_xticklabels(sentance2.split())
    # set the y axis labels as input apparels title
    ax1.set_yticklabels(sentance1.split())
    # set title as recommended apparels title
    ax1.set_title(sentance2)

    # in Last 25 * 10:15 grids we display image
    ax2 = plt.subplot(gs[:, 10:16])
    # we dont display grid lines and axis labels to images
    ax2.grid(False)
    ax2.set_xticks([])
    ax2.set_yticks([])
```

```
# pass the url it display it  
display_img(url, ax2, fig)  
  
plt.show()
```

In [76]:

```

def idf_w2v_brand(doc_id, w1, w2, num_results):
    # doc_id: apparel's id in given corpus
    # w1: weight for w2v features
    # w2: weight for brand and color features

    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as  $K(X, Y) = \frac{X \cdot Y}{\|X\| * \|Y\|}$ 
    # http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
    idf_w2v_dist = pairwise_distances(w2v_title_weight, w2v_title_weight[doc_id].reshape(1,-1))
    ex_feat_dist = pairwise_distances(extra_features, extra_features[doc_id])
    pairwise_dist = (w1 * idf_w2v_dist + w2 * ex_feat_dist)/float(w1 + w2)

    # np.argsort will return indices of 9 smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the 9 smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

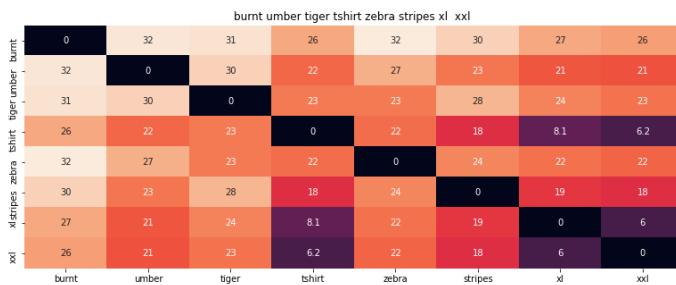
    #data frame indices of the 9 smallest distace's
    df_indices = list(data.index[indices])

    for i in range(0, len(indices)):
        heat_map_w2v_brand(data['title'].loc[df_indices[0]],data['title'].loc[df_indices[i]], data['medium_image_url'].loc[df_indices[i]], indices[0], indices[i],df_indices[0], df_indices[i], 'weighted')
        print('ASIN :',data['asin'].loc[df_indices[i]])
        print('Brand :',data['brand'].loc[df_indices[i]])
        print('euclidean distance from input :', pdists[i])
        print('='*125)

    idf_w2v_brand(12566, 5, 5, 20)
    # in the give heat map, each cell contains the euclidean distance between words i, j

```

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

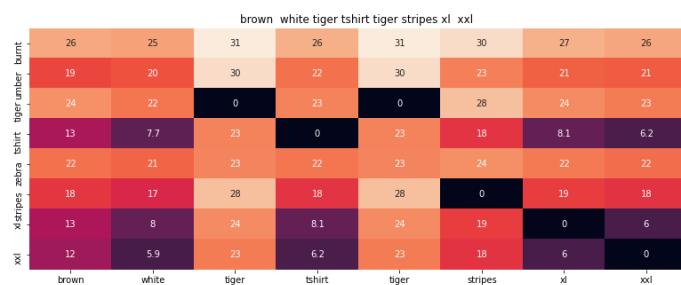


ASIN : B00JXQB5FQ

Brand : Si Row

euclidean distance from input : 0.0

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

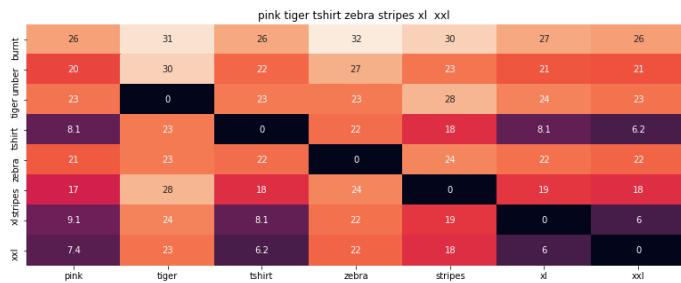


ASIN : B00JXQCWTO

Brand : Si Row

euclidean distance from input : 2.3854705810546877

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

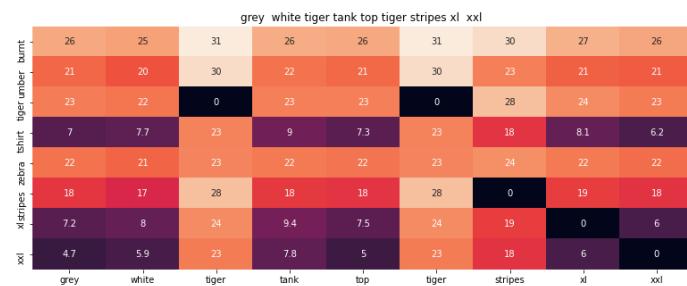


ASIN : B00JXQASS6

Brand : Si Row

euclidean distance from input : 2.739050102414575

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

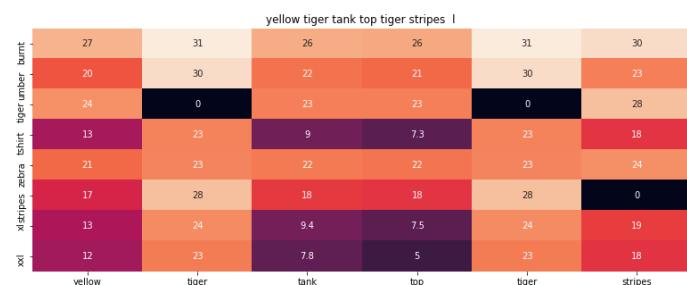


ASIN : B00JXQAFZ2

Brand : Si Row

euclidean distance from input : 3.387187004270044

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQAUWA

Brand : Si Row

euclidean distance from input : 3.5518680574316646

=====

=====

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQA094

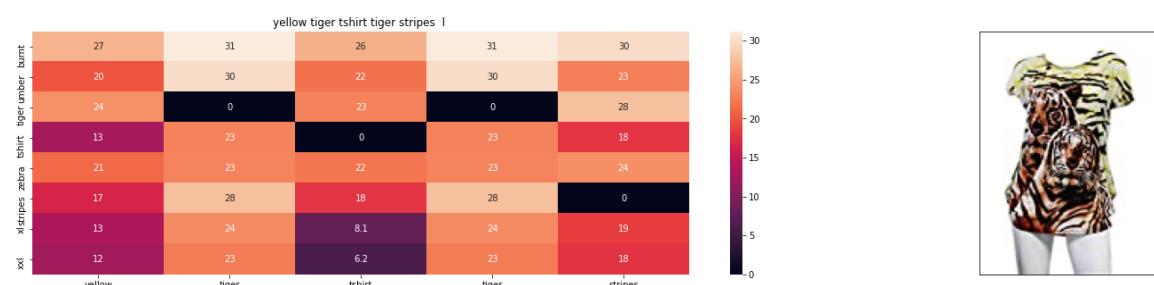
Brand : Si Row

euclidean distance from input : 3.5536170961279536

=====

=====

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQCUIC

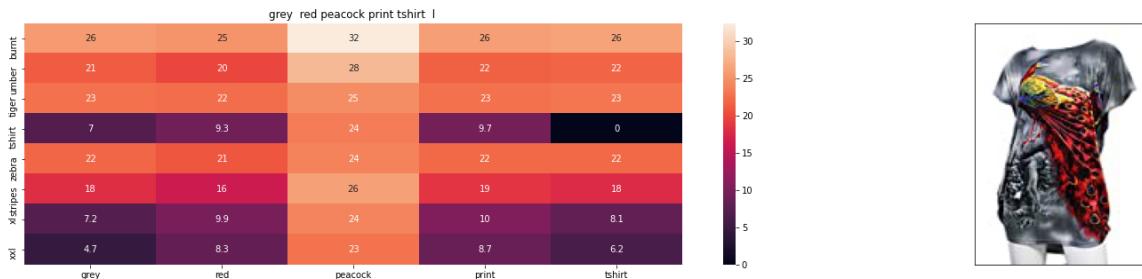
Brand : Si Row

euclidean distance from input : 3.6538278581518795

=====

=====

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

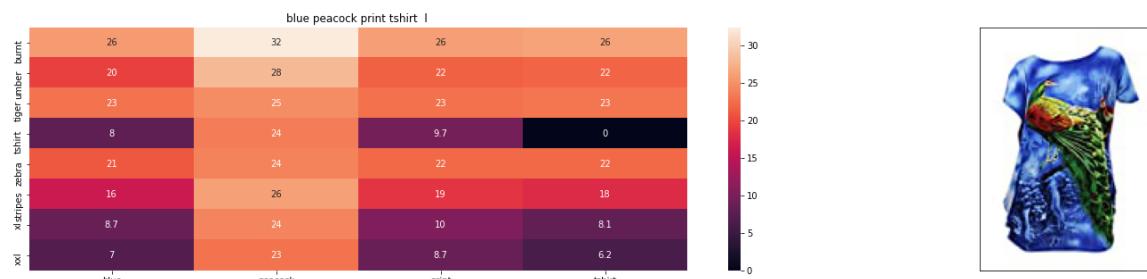


ASIN : B00JXQC8RS

Brand : Si Row

euclidean distance from input : 4.128811264218774

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

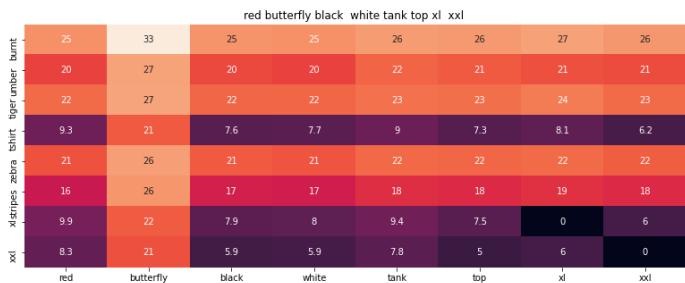


ASIN : B00JXQC8L6

Brand : Si Row

euclidean distance from input : 4.203900146665063

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

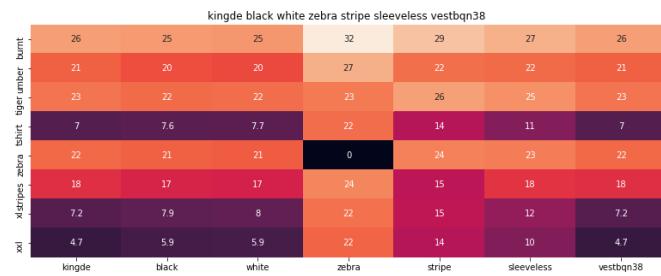


ASIN : B00JV63CW2

Brand : Si Row

euclidean distance from input : 4.286586380185571

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

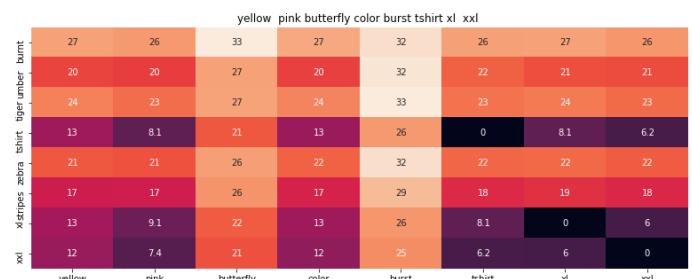


ASIN : B015H41F6G

Brand : KINGDE

euclidean distance from input : 4.389370406508858

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

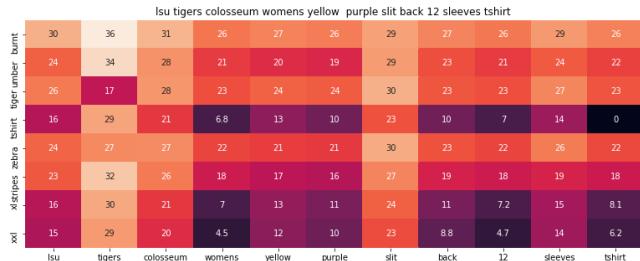


ASIN : B00JXQBBMI

Brand : Si Row

euclidean distance from input : 4.397909927548852

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B073R5Q8HD

Brand : Colosseum

euclidean distance from input : 4.451228392959053

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

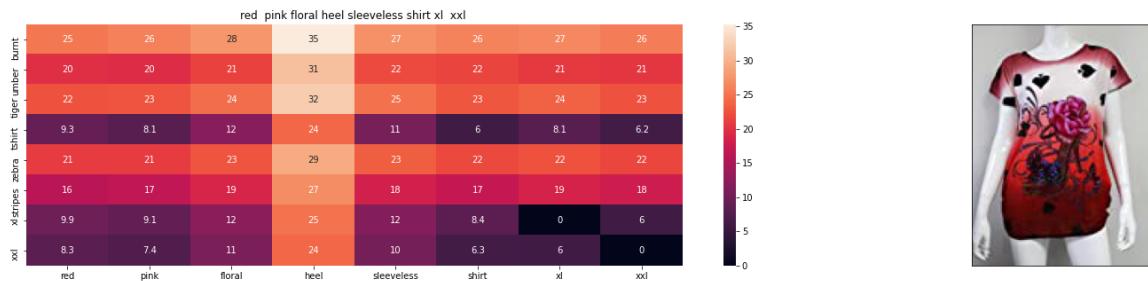


ASIN : B074P8MD22

Brand : Edista

euclidean distance from input : 4.518977416396553

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JV63QQE

Brand : Si Row

euclidean distance from input : 4.529374695004907

=====

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00C0I3U3E

Brand : Stanzino

euclidean distance from input : 4.530325759292061

=====

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



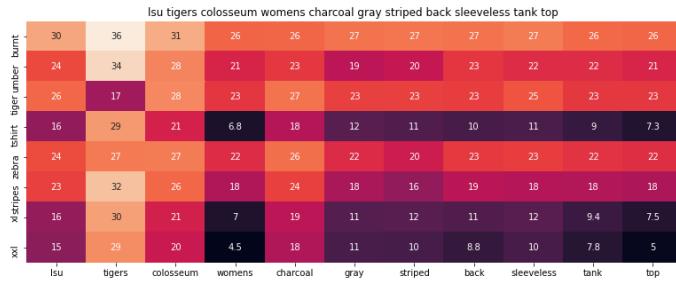
ASIN : B01ER18406

Brand : GuPoBoU168

euclidean distance from input : 4.546816642558488

=====

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

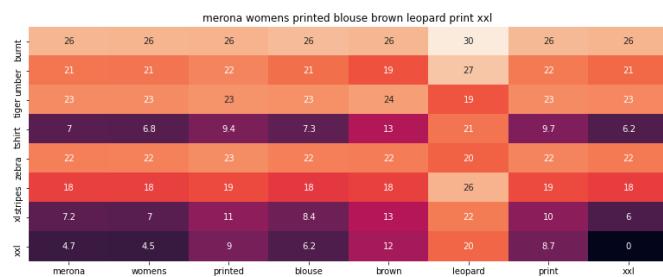


ASIN : B073R4ZM7Y

Brand : Colosseum

euclidean distance from input : 4.548355162978584

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B071YF3WDD

Brand : Merona

euclidean distance from input : 4.610626662612374

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B01C60RLDQ

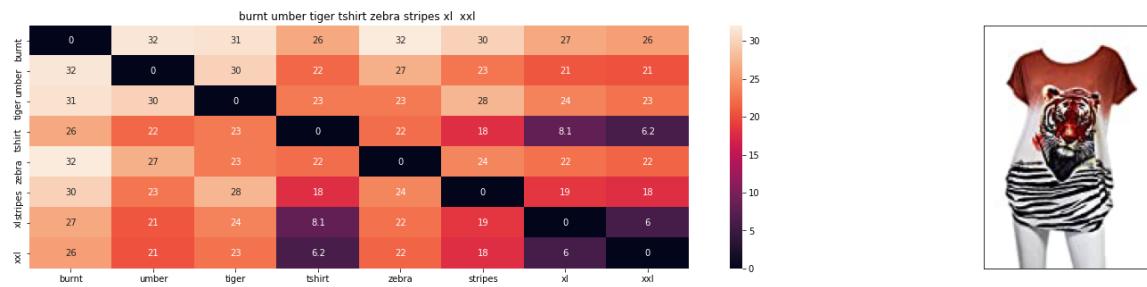
Brand : 1 Mad Fit

euclidean distance from input : 4.645917892817431

In [77]:

```
# brand and color weight =50  
# title vector weight = 5  
  
idf_w2v_brand(12566, 5, 50, 20)
```

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

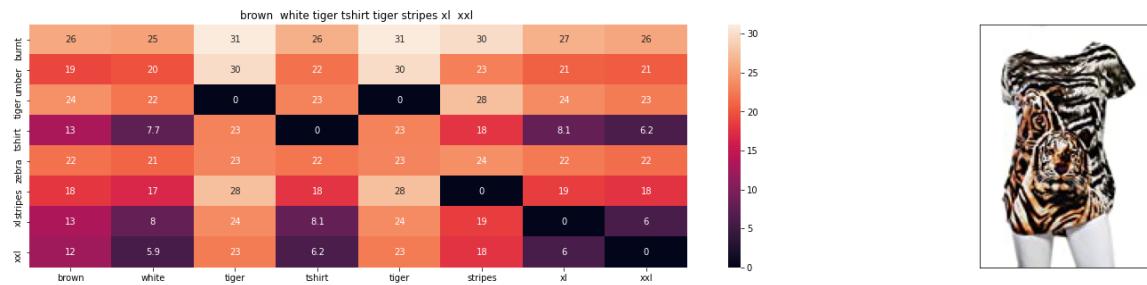


ASIN : B00JXQB5FQ

Brand : Si Row

euclidean distance from input : 0.0

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

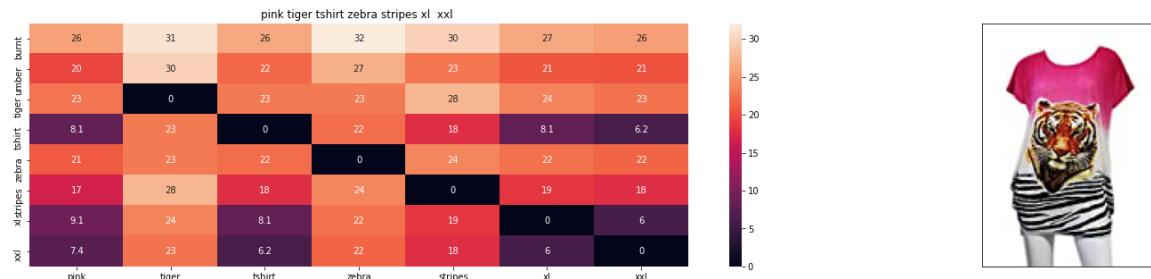


ASIN : B00JXQCWT0

Brand : Si Row

euclidean distance from input : 0.433721923828125

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQASS6

Brand : Si Row

euclidean distance from input : 1.6550929332897277

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

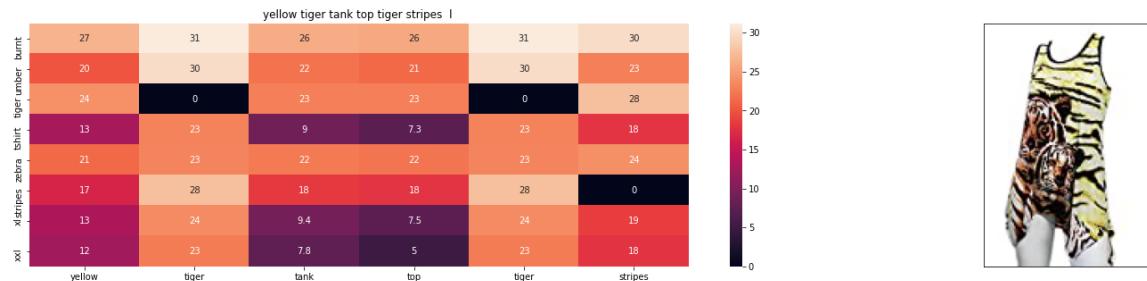


ASIN : B00JXQAFZ2

Brand : Si Row

euclidean distance from input : 1.7729360063543584

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQAUWA

Brand : Si Row

euclidean distance from input : 1.8028780160201077

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

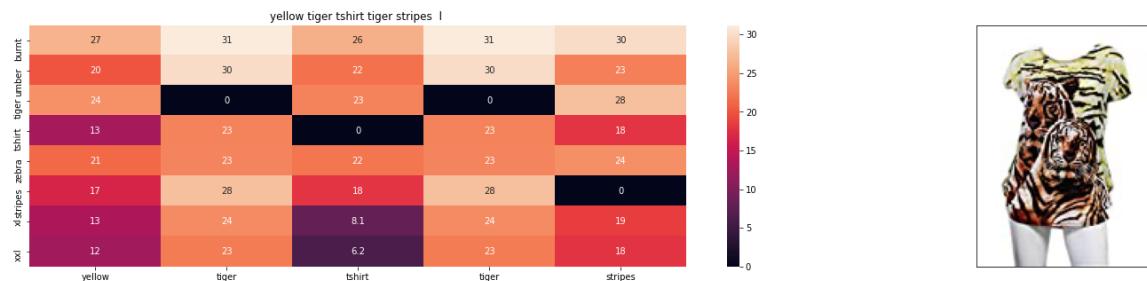


ASIN : B00JXQA094

Brand : Si Row

euclidean distance from input : 1.8031960230557966

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

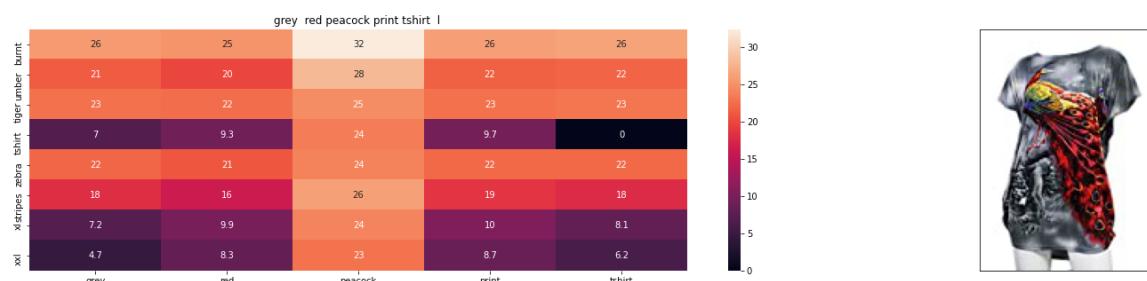


ASIN : B00JXQCUIC

Brand : Si Row

euclidean distance from input : 1.8214161616056013

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

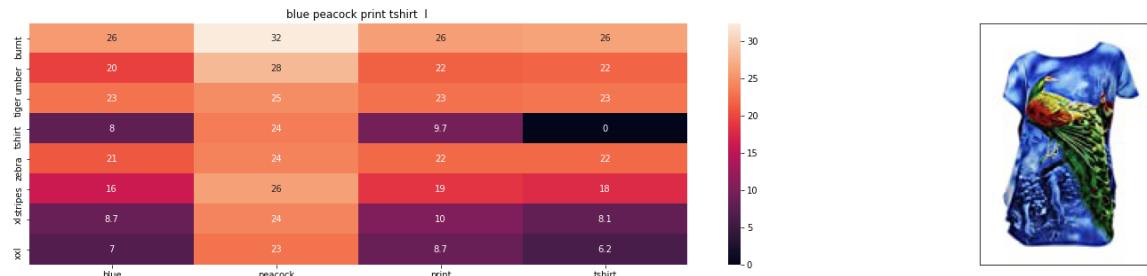


ASIN : B00JXQCFRS

Brand : Si Row

euclidean distance from input : 1.9077767808904913

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

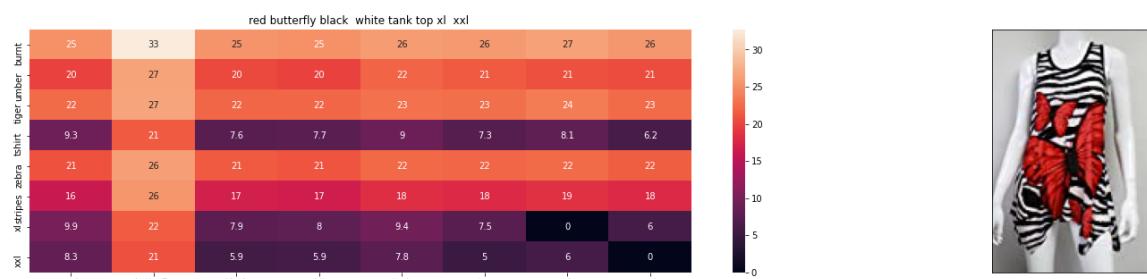


ASIN : B00JXQC8L6

Brand : Si Row

euclidean distance from input : 1.9214293049716347

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

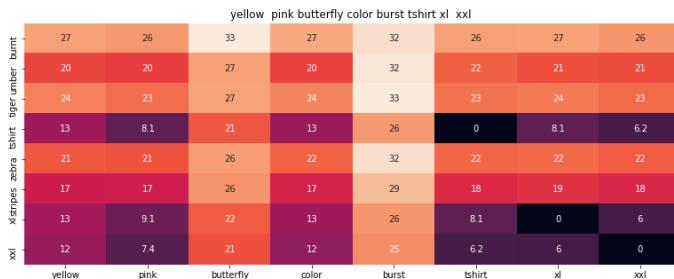


ASIN : B00JV63CW2

Brand : Si Row

euclidean distance from input : 1.936463165611727

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQBBMI

Brand : Si Row

euclidean distance from input : 1.956703810586869

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

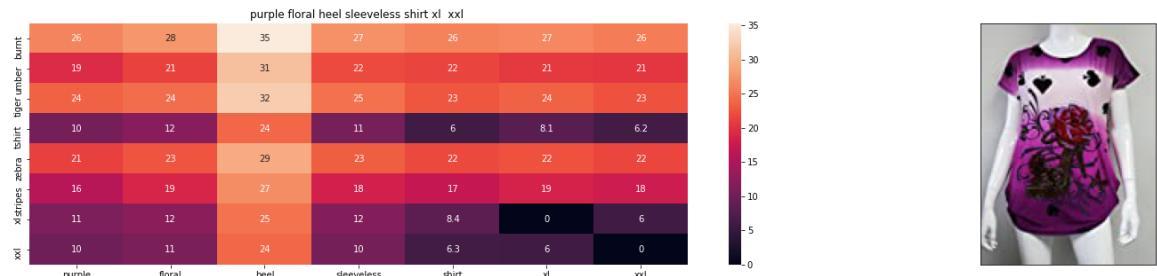


ASIN : B00JV63QQE

Brand : Si Row

euclidean distance from input : 1.9806064955788791

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

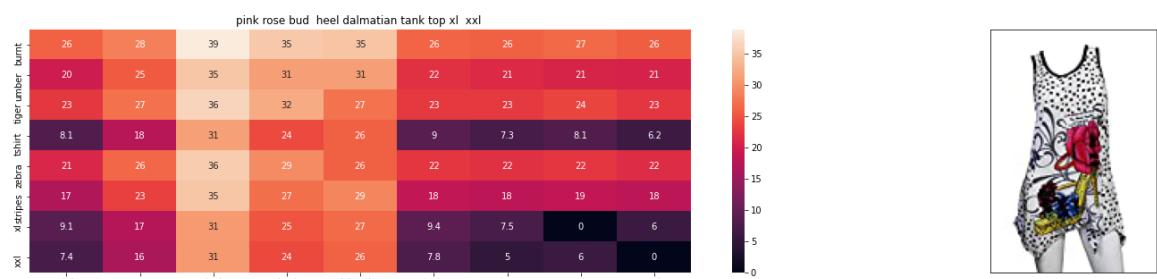


ASIN : B00JV63VC8

Brand : Si Row

euclidean distance from input : 2.012185530557572

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B00JXQAX2C

Brand : Si Row

euclidean distance from input : 2.01335171819074

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

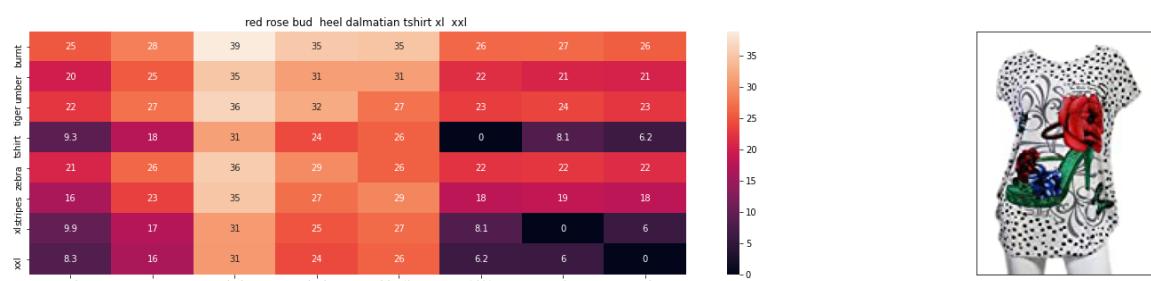


ASIN : B00JXQC0C8

Brand : Si Row

euclidean distance from input : 2.0138832789151717

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

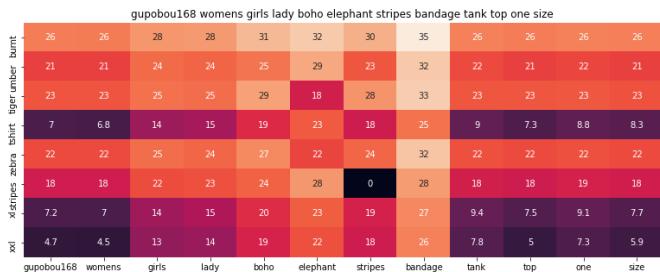


ASIN : B00JXQABB0

Brand : Si Row

euclidean distance from input : 2.0367257554998663

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B01ER18406

Brand : GuPoBoU168

euclidean distance from input : 2.656204098419553

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

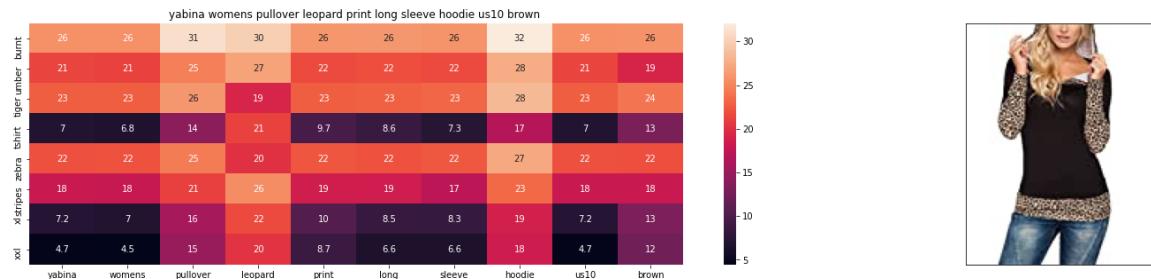


ASIN : B01LZ7BQ4H

Brand : WAYF

euclidean distance from input : 2.6849067129437008

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown

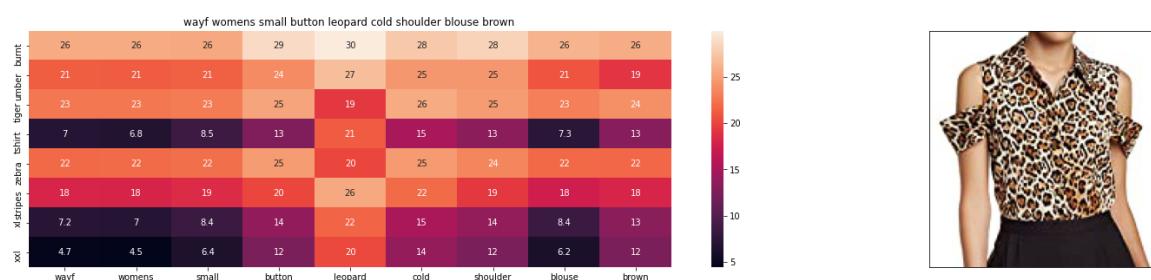


ASIN : B01KJUM6JI

Brand : YABINA

euclidean distance from input : 2.6858381232997024

Asin	Brand	Color
B00JXQB5FQ	Si-Row	Brown



ASIN : B01M06V4X1

Brand : WAYF

euclidean distance from input : 2.694761948650377

[10.2] Keras and Tensorflow to extract features

In [78]:

```
import numpy as np
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
from sklearn.metrics import pairwise_distances
import matplotlib.pyplot as plt
import requests
from PIL import Image
import pandas as pd
import pickle
```

Using TensorFlow backend.

In [79]:

```
...
# https://gist.github.com/fchollet/f35fbc80e066a49d65f1688a7e99f069
# Code reference: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html

# This code takes 40 minutes to run on a modern GPU (graphics card)
# like Nvidia 1050.
# GPU (Nvidia 1050): 0.175 seconds per image

# This codse takes 160 minutes to run on a high end i7 CPU
# CPU (i7): 0.615 seconds per image.

#Do NOT run this code unless you want to wait a few hours for it to generate output

# each image is converted into 25088 Length dense-vector
...

...
# dimensions of our images.
img_width, img_height = 224, 224

top_model_weights_path = 'bottleneck_fc_model.h5'
train_data_dir = 'images2/'
nb_train_samples = 16042
epochs = 50
batch_size = 1

def save_bottlebeck_features():

    #Function to compute VGG-16 CNN for image feature extraction.

    asins = []
    datagen = ImageDataGenerator(rescale=1. / 255)

    # build the VGG16 network
    model = applications.VGG16(include_top=False, weights='imagenet')
    generator = datagen.flow_from_directory(
        train_data_dir,
        target_size=(img_width, img_height),
        batch_size=batch_size,
        class_mode=None,
        shuffle=False)

    for i in generator.filenames:
        asins.append(i[2:-5])

    bottleneck_features_train = model.predict_generator(generator, nb_train_samples // batch_size)
    bottleneck_features_train = bottleneck_features_train.reshape((16042, 25088))

    np.save(open('16k_data_cnn_features.npy', 'wb'), bottleneck_features_train)
    np.save(open('16k_data_cnn_feature_asins.npy', 'wb'), np.array(asins))

save_bottlebeck_features()
```

```
Out[79]:
```

```
"\n# dimensions of our images.\nimg_width, img_height = 224, 224\n\n\ntop_mo  
del_weights_path = 'bottleneck_fc_model.h5'\ntrain_data_dir = 'images2/'\nn  
nb_train_samples = 16042\nepochs = 50\nbatch_size = 1\n\n\ndef save_bottle  
beck_features():\n    \n        #Function to compute VGG-16 CNN for image feat  
ure extraction.\n        \n        asins = []\n        datagen = ImageDataGenerator(re  
scale=1. / 255)\n        \n        # build the VGG16 network\n        model = applicat  
ions.VGG16(include_top=False, weights='imagenet')\n        generator = datage  
n.flow_from_directory(\n            train_data_dir,\n            target_size=(img_  
width, img_height),\n            batch_size=batch_size,\n            class_mode=None,  
            \n            shuffle=False)\n\n        for i in generator.filenames:\n            asins.append(i[2:-5])\n\n        bottleneck_features_train = model.predict_gen  
erator(generator, nb_train_samples // batch_size)\n        bottleneck_features_<br>  
_train = bottleneck_features_train.reshape((16042,25088))\n        \n        np.sa  
ve(open('16k_data_cnn_features.npy', 'wb'), bottleneck_features_train)\n        np.save(open('16k_data_cnn_feature_asins.npy', 'wb'), np.array(asins))\n\n\ndef save_bottlebeck_features():\n    \n"
```

[10.3] Visual features based product similarity.

In [80]:

```
#Load the features and corresponding ASINS info.
bottleneck_features_train = np.load('/content/drive/My Drive/16k_data_cnn_features.npy')
)
asins = np.load('/content/drive/My Drive/16k_data_cnn_feature_asins.npy')
asins = list(asins)

# Load the original 16K dataset
data = pd.read_pickle('/content/drive/My Drive/16k_apperial_data_preprocessed')
df_asins = list(data['asin'])

from IPython.display import display, Image, SVG, Math, YouTubeVideo

#get similar products using CNN features (VGG-16)
def get_similar_products_cnn(doc_id, num_results):
    doc_id = asins.index(df_asins[doc_id])
    pairwise_dist = pairwise_distances(bottleneck_features_train, bottleneck_features_train[doc_id].reshape(1,-1))

    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    for i in range(len(indices)):
        rows = data[['medium_image_url','title']].loc[data['asin']==asins[indices[i]]]
        for indx, row in rows.iterrows():
            display(Image(url=row['medium_image_url'], embed=True))
            print('Product Title: ', row['title'])
            print('Euclidean Distance from input image:', pdists[i])
            print('Amazon Url: www.amazon.com/dp/' + asins[indices[i]])


get_similar_products_cnn(12566, 20)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 6.32596e-06

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 30.05017

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 41.261116

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 44.000156

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: kawaii pastel tops tees pink flower design
Euclidean Distance from input image: 47.38248
Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: womens thin style tops tees pastel watermelon print
Euclidean Distance from input image: 47.71842
Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: kawaii pastel tops tees baby blue flower design
Euclidean Distance from input image: 47.90206
Amazon Url: www.amazon.com/dp/B071SBCY9W



Product Title: edv cheetah run purple multi xl
Euclidean Distance from input image: 48.046482
Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: danskin womens vneck loose performance tee xsmall pink ombre

Euclidean Distance from input image: 48.101837

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: summer alpaca 3d pastel casual loose tops tee design

Euclidean Distance from input image: 48.118866

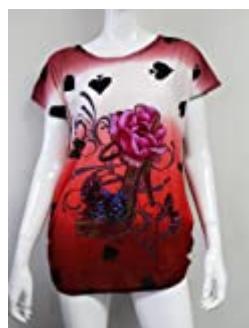
Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: misschievous juniors striped peplum tank top medium shadow peach

Euclidean Distance from input image: 48.13122

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: red pink floral heel sleeveless shirt xl xxl

Euclidean Distance from input image: 48.16945

Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: moana logo adults hot v neck shirt black xxl

Euclidean Distance from input image: 48.256786

Amazon Url: www.amazon.com/dp/B01LX6H43D



Product Title: abaday multicolor cartoon cat print short sleeve longline shirt large

Euclidean Distance from input image: 48.265686

Amazon Url: www.amazon.com/dp/B01CR57YY0



Product Title: kawaii cotton pastel tops tees peach pink cactus design

Euclidean Distance from input image: 48.362602

Amazon Url: www.amazon.com/dp/B071WYLBZS



Product Title: chicago chicago 18 shirt women pink

Euclidean Distance from input image: 48.383606

Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: yichun womens tiger printed summer tshirts tops

Euclidean Distance from input image: 48.449356

Amazon Url: www.amazon.com/dp/B010NN9RX0



Product Title: nancy lopez whimsy short sleeve whiteblacklemon drop xs

Euclidean Distance from input image: 48.47889

Amazon Url: www.amazon.com/dp/B01MPX6IDX



Product Title: womens tops tees pastel peach ice cream cone print

Euclidean Distance from input image: 48.557957

Amazon Url: www.amazon.com/dp/B0734GRKZL



Product Title: uswomens mary j blige without tshirts shirt

Euclidean Distance from input image: 48.614372

Amazon Url: www.amazon.com/dp/B01M0XXFKK

Assignment

In [6]:

```
#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")

import numpy as np
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
from sklearn.metrics import pairwise_distances
import matplotlib.pyplot as plt
import requests
from PIL import Image
import pandas as pd
import pickle
```

In [7]:

```
data = pd.read_pickle('/content/drive/My Drive/pickels/16k_apperal_data_preprocessed')
data.head()
```

Out[7]:

	asin	brand	color	medium_image_url	product_type_name	
4	B004GSI2OS	FeatherLite	Onyx Black/Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	feath ladie slee resis
6	B012YX2ZPI	HX-Kingdom Fashion T-shirts	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	wom uniq cott spec olym wor..
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	feath ladie mois free spor
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl-images-amazon.com/images...	SHIRT	supe chibi dear neck
46	B01NACPBG2	Fifth Degree	Black	https://images-na.ssl-images-amazon.com/images...	SHIRT	fifth wom gold grap jun..

In [0]:

```
import pickle
with open('/content/drive/My Drive/word2vec_model', 'rb') as handle:
    model = pickle.load(handle)
```

In [0]:

```
idf_title_vectorizer = CountVectorizer()
idf_title_features = idf_title_vectorizer.fit_transform(data['title'])

# idf_title_features.shape = #data_points * #words_in_corpus
# CountVectorizer().fit_transform(courpus) returns the a sparase matrix of dimensions #
# data_points * #words_in_corpus
# idf_title_features[doc_id, index_of_word_in_corpus] = number of times the word occure
d in that doc
```

In [0]:

```
def nContaining(word):
    # return the number of documents which had the given word
    return sum(1 for blob in data['title'] if word in blob.split())

def idf(word):
    # idf = log(#number of docs / #number of docs which had the given word)
    return math.log(data.shape[0] / (nContaining(word)))
```

In [0]:

```
# we need to convert the values into float
idf_title_features = idf_title_features.astype(np.float)

for i in idf_title_vectorizer.vocabulary_.keys():
    # for every word in whole corpus we will find its idf value
    idf_val = idf(i)

    # to calculate idf_title_features we need to replace the count values with the idf
    # values of the word
    # idf_title_features[:, idf_title_vectorizer.vocabulary_[i]].nonzero()[0] will retu
    rn all documents in which the word i present
    for j in idf_title_features[:, idf_title_vectorizer.vocabulary_[i]].nonzero()[0]:
        # we replace the count values of word i in document j with idf_value of word i
        # idf_title_features[doc_id, index_of_word_in_courpus] = idf value of word
        idf_title_features[j,idf_title_vectorizer.vocabulary_[i]] = idf_val
```

In [0]:

```
# vocab = stores all the words that are there in google w2v model
# vocab = model.wv.vocab.keys() # if you are using Google word2Vec

vocab = model.keys()
# this function will add the vectors of each word and returns the avg vector of given sentence

def build_avg_vec(sentence, num_features, doc_id, m_name):
    # sentace: its title of the apparel
    # num_features: the lenght of word2vec vector, its values = 300
    # m_name: model information it will take two values
        # if m_name == 'avg', we will append the model[i], w2v representation of word
    i
        # if m_name == 'weighted', we will multiply each w2v[word] with the idf(word)

    featureVec = np.zeros((num_features,), dtype="float32")
    # we will intialize a vector of size 300 with all zeros
    # we add each word2vec(wordi) to this festureVec
    nwords = 0

    for word in sentence.split():
        nwords += 1
        if word in vocab:
            if m_name == 'weighted' and word in idf_title_vectorizer.vocabulary_:
                featureVec = np.add(featureVec, idf_title_features[doc_id, idf_title_vectorizer.vocabulary_[word]] * model[word])
            elif m_name == 'avg':
                featureVec = np.add(featureVec, model[word])
    if(nwords>0):
        featureVec = np.divide(featureVec, nwords)
    # returns the avg vector of given sentance, its of shape (1, 300)
    return featureVec
```

In [0]:

```
doc_id = 0
w2v_title_weight = []
# for every title we build a weighted vector representation
for i in data['title']:
    w2v_title_weight.append(build_avg_vec(i, 300, doc_id,'weighted'))
    doc_id += 1
# w2v_title = np.array(# number of doc in courpus * 300), each row corresponds to a doc
w2v_title_weight = np.array(w2v_title_weight)
```

In [0]:

```
# Utility functions

def get_word_vec(sentence, doc_id, m_name):
    # sentence : title of the apparel
    # doc_id: document id in our corpus
    # m_name: model information it will take two values
        # if m_name == 'avg', we will append the model[i], w2v representation of word
    i
        # if m_name == 'weighted', we will multiply each w2v[word] with the idf(word)
    vec = []
    for i in sentence.split():
        if i in vocab:
            if m_name == 'weighted' and i in idf_title_vectorizer.vocabulary_:
                vec.append(idf_title_features[doc_id, idf_title_vectorizer.vocabulary_[i]] * model[i])
            elif m_name == 'avg':
                vec.append(model[i])
        else:
            # if the word in our corpus is not there in the google word2vec corpus, we
            # are just ignoring it
            vec.append(np.zeros(shape=(300,)))
    # we will return a numpy array of shape (#number of words in title * 300 ) 300 = len(w2v_model[word])
    # each row represents the word2vec representation of each word (weighted/avg) in given sentence
    return np.array(vec)

def get_distance(vec1, vec2):
    # vec1 = np.array(#number_of_words_title1 * 300), each row is a vector of length 300
    # corresponds to each word in give title
    # vec2 = np.array(#number_of_words_title2 * 300), each row is a vector of length 300
    # corresponds to each word in give title

    final_dist = []
    # for each vector in vec1 we calculate the distance(euclidean) to all vectors in vec2
    for i in vec1:
        dist = []
        for j in vec2:
            # np.linalg.norm(i-j) will result the euclidean distance between vectors i, j
            dist.append(np.linalg.norm(i-j))
        final_dist.append(np.array(dist))
    # final_dist = np.array(#number of words in title1 * #number of words in title2)
    # final_dist[i,j] = euclidean distance between vectors i, j
    return np.array(final_dist)

def heat_map_w2v(sentence1, sentence2, url, doc_id1, doc_id2, model):
    # sentance1 : title1, input apparel
    # sentance2 : title2, recommended apparel
    # url: apparel image url
    # doc_id1: document id of input apparel
    # doc_id2: document id of recommended apparel
    # model: it can have two values, 1. avg 2. weighted

    #s1_vec = np.array(#number_of_words_title1 * 300), each row is a vector(weighted/avg) of length 300 corresponds to each word in give title
    s1_vec = get_word_vec(sentence1, doc_id1, model)
    #s2_vec = np.array(#number_of_words_title1 * 300), each row is a vector(weighted/avg) of length 300 corresponds to each word in give title
    s2_vec = get_word_vec(sentence2, doc_id2, model)
```

```
g) of length 300 corresponds to each word in give title
s2_vec = get_word_vec(sentence2, doc_id2, model)

# s1_s2_dist = np.array(#number of words in title1 * #number of words in title2)
# s1_s2_dist[i,j] = euclidean distance between words i, j
s1_s2_dist = get_distance(s1_vec, s2_vec)

# devide whole figure into 2 parts 1st part displays heatmap 2nd part displays image
# of apparel
gs = gridspec.GridSpec(2, 2, width_ratios=[4,1],height_ratios=[2,1])
fig = plt.figure(figsize=(15,15))

ax = plt.subplot(gs[0])
# plotting the heap map based on the pairwise distances
ax = sns.heatmap(np.round(s1_s2_dist,4), annot=True)
# set the x axis labels as recommended apparels title
ax.set_xticklabels(sentence2.split())
# set the y axis labels as input apparels title
ax.set_yticklabels(sentence1.split())
# set title as recommended apparels title
ax.set_title(sentence2)

ax = plt.subplot(gs[1])
# we remove all grids and axis labels for image
ax.grid(False)
ax.set_xticks([])
ax.set_yticks([])
display_img(url, ax, fig)

plt.show()
```

In [0]:

```
# Utility Functions which we will use through the rest of the workshop.

#Display an image
def display_img(url,ax,fig):
    # we get the url of the apparel and download it
    response = requests.get(url)
    img = Image.open(BytesIO(response.content))
    # we will display it in notebook
    plt.imshow(img)

#plotting code to understand the algorithm's decision.
def plot_heatmap(keys, values, labels, url, text):
    # keys: list of words of recommended title
    # values: len(values) == len(keys), values(i) represents the occurrence of the word keys(i)
    # labels: len(labels) == len(keys), the values of labels depends on the model we are using
    # if model == 'bag of words': labels(i) = values(i)
    # if model == 'tfidf weighted bag of words': labels(i) = tfidf(keys(i))
    # if model == 'idf weighted bag of words': labels(i) = idf(keys(i))
    # url : apparel's url

    # we will devide the whole figure into two parts
    gs = gridspec.GridSpec(2, 2, width_ratios=[4,1], height_ratios=[4,1])
    fig = plt.figure(figsize=(25,3))

    # 1st, plotting heat map that represents the count of commonly occurred words in title2
    ax = plt.subplot(gs[0])
    # it displays a cell in white color if the word is intersection(list of words of title1 and list of words of title2), in black if not
    ax = sns.heatmap(np.array([values]), annot=np.array([labels]))
    ax.set_xticklabels(keys) # set that axis labels as the words of title
    ax.set_title(text) # apparel title

    # 2nd, plotting image of the the apparel
    ax = plt.subplot(gs[1])
    # we don't want any grid lines for image and no labels on x-axis and y-axis
    ax.grid(False)
    ax.set_xticks([])
    ax.set_yticks([])

    # we call dispaly_img based with paramete url
    display_img(url, ax, fig)

    # displays combine figure ( heat map and image together)
    plt.show()

def plot_heatmap_image(doc_id, vec1, vec2, url, text, model):

    # doc_id : index of the title1
    # vec1 : input apparels's vector, it is of a dict type {word:count}
    # vec2 : recommended apparels's vector, it is of a dict type {word:count}
    # url : apparels image url
    # text: title of recomonded apparel (used to keep title of image)
    # model, it can be any of the models,
    # 1. bag_of_words
    # 2. tfidf
```

```

# 3. idf

# we find the common words in both titles, because these only words contribute to the distance between two title vec's
intersection = set(vec1.keys()) & set(vec2.keys())

# we set the values of non intersecting words to zero, this is just to show the difference in heatmap
for i in vec2:
    if i not in intersection:
        vec2[i]=0

# for Labeling heatmap, keys contains list of all words in title2
keys = list(vec2.keys())
# if ith word in intersection(list of words of title1 and list of words of title2):
values(i)=count of that word in title2 else values(i)=0
values = [vec2[x] for x in vec2.keys()]

# Labels: Len(Labels) == Len(keys), the values of Labels depends on the model we are using
# if model == 'bag of words': Labels(i) = values(i)
# if model == 'tfidf weighted bag of words': Labels(i) = tfidf(keys(i))
# if model == 'idf weighted bag of words': Labels(i) = idf(keys(i))

if model == 'bag_of_words':
    labels = values
elif model == 'tfidf':
    labels = []
    for x in vec2.keys():
        # tfidf_title_vectorizer.vocabulary_ it contains all the words in the corpus
        # tfidf_title_features[doc_id, index_of_word_in_corpus] will give the tfidf value of word in given document (doc_id)
        if x in tfidf_title_vectorizer.vocabulary_:
            labels.append(tfidf_title_features[doc_id, tfidf_title_vectorizer.vocabulary_[x]])
        else:
            labels.append(0)
elif model == 'idf':
    labels = []
    for x in vec2.keys():
        # idf_title_vectorizer.vocabulary_ it contains all the words in the corpus
        # idf_title_features[doc_id, index_of_word_in_corpus] will give the idf value of word in given document (doc_id)
        if x in idf_title_vectorizer.vocabulary_:
            labels.append(idf_title_features[doc_id, idf_title_vectorizer.vocabulary_[x]])
        else:
            labels.append(0)

plot_heatmap(keys, values, labels, url, text)

# this function gets a list of words along with the frequency of each word given "text"
def text_to_vector(text):
    word = re.compile(r'\w+')
    words = word.findall(text)
    # words stores list of all words in given string, you can try 'words = text.split()' this will also gives same result
    return Counter(words) # Counter counts the occurrence of each word in list, it returns dict type object {word1:count}

```

```
def get_result(doc_id, content_a, content_b, url, model):
    text1 = content_a
    text2 = content_b

    # vector1 = dict{word11:#count, word12:#count, etc.}
    vector1 = text_to_vector(text1)

    # vector1 = dict{word21:#count, word22:#count, etc.}
    vector2 = text_to_vector(text2)

    plot_heatmap_image(doc_id, vector1, vector2, url, text2, model)
```

Weighted similarity using brand and color.

In [0]:

```
# some of the brand values are empty.
# Need to replace Null with string "NULL"
data['brand'].fillna(value="Not given", inplace=True )

# replace spaces with hyphen
brands = [x.replace(" ", "-") for x in data['brand'].values]
types = [x.replace(" ", "-") for x in data['product_type_name'].values]
colors = [x.replace(" ", "-") for x in data['color'].values]

brand_vectorizer = CountVectorizer()
brand_features = brand_vectorizer.fit_transform(brands)

type_vectorizer = CountVectorizer()
type_features = type_vectorizer.fit_transform(types)

color_vectorizer = CountVectorizer()
color_features = color_vectorizer.fit_transform(colors)

extra_features = hstack((brand_features, type_features, color_features)).tocsr()
```

In [0]:

```

def heat_map_w2v_brand(sentance1, sentance2, url, doc_id1, doc_id2, df_id1, df_id2, model):
    # sentance1 : title1, input apparel
    # sentance2 : title2, recommended apparel
    # url: apparel image url
    # doc_id1: document id of input apparel
    # doc_id2: document id of recommended apparel
    # df_id1: index of document1 in the data frame
    # df_id2: index of document2 in the data frame
    # model: it can have two values, 1. avg 2. weighted

    #s1_vec = np.array(#number_of_words_title1 * 300), each row is a vector(weighted/avg)
    # of length 300 corresponds to each word in give title
    s1_vec = get_word_vec(sentance1, doc_id1, model)
    #s2_vec = np.array(#number_of_words_title2 * 300), each row is a vector(weighted/avg)
    # of length 300 corresponds to each word in give title
    s2_vec = get_word_vec(sentance2, doc_id2, model)

    # s1_s2_dist = np.array(#number of words in title1 * #number of words in title2)
    # s1_s2_dist[i,j] = euclidean distance between words i, j
    s1_s2_dist = get_distance(s1_vec, s2_vec)

    data_matrix = [['Asin', 'Brand', 'Color', 'Product type'],
                  [data['asin'].loc[df_id1], brands[doc_id1], colors[doc_id1], types[doc_id1]], # input apparel's features
                  [data['asin'].loc[df_id2], brands[doc_id2], colors[doc_id2], types[doc_id2]]] # recommended apparel's features

    colorscale = [[0, '#1d004d'], [.5, '#f2e5ff'], [1, '#f2e5d1']] # to color the headings of each column

    # we create a table with the data_matrix
    table = ff.create_table(data_matrix, index=True, colorscale=colorscale)
    # plot it with plotly
    plotly.offline.iplot(table, filename='simple_table')

    # devide whole figure space into 25 * 1:10 grids
    gs = gridspec.GridSpec(25, 15)
    fig = plt.figure(figsize=(25,5))

    # in first 25*10 grids we plot heatmap
    # devide whole figure space into 25 * 1:10 grids
    gs = gridspec.GridSpec(25, 15)
    fig = plt.figure(figsize=(25,5))

    # in first 25*10 grids we plot heatmap
    ax1 = plt.subplot(gs[:, :-5])
    # plotting the heatmap based on the pairwise distances
    ax1 = sns.heatmap(np.round(s1_s2_dist,6), annot=True)
    # set the x axis labels as recommended apparels title
    ax1.set_xticklabels(sentance2.split())
    # set the y axis labels as input apparels title
    ax1.set_yticklabels(sentance1.split())
    # set title as recommended apparels title
    ax1.set_title(sentance2)

    # in last 25 * 10:15 grids we display image
    ax2 = plt.subplot(gs[:, 10:16])

```

```
# we dont display grid lines and axis labels to images
ax2.grid(False)
ax2.set_xticks([])
ax2.set_yticks([])

# pass the url it display it
display_img(url, ax2, fig)

plt.show()
```

In [22]:

```

def idf_w2v_brand_color_cnn(doc_id, w1, w2, w3, num_results):
    #Load the features and corresponding ASINS info.
    bottleneck_features_train = np.load('/content/drive/My Drive/16k_data_cnn_features.npy')
    asins = np.load('/content/drive/My Drive/16k_data_cnn_feature_asins.npy')
    asins = list(asins)

    # Load the original 16K dataset
    data = pd.read_pickle('/content/drive/My Drive/pickels/16k_apperal_data_preprocessed')
    df_asins = list(data['asin'])
    from IPython.display import display, Image, SVG, Math, YouTubeVideo

    idf_w2v_dist = pairwise_distances(w2v_title_weight, w2v_title_weight[doc_id].reshape(1,-1))
    ex_feat_dist = pairwise_distances(extra_features, extra_features[doc_id])
    doc_id = asins.index(df_asins[doc_id])
    cnn_feat_dist = pairwise_distances(bottleneck_features_train, bottleneck_features_train[doc_id].reshape(1,-1))
    pairwise_dist = (w1 * idf_w2v_dist + w2 * ex_feat_dist + w3*cnn_feat_dist)/float(w1 + w2 + w3)

    # np.argsort will return indices of 9 smallest distances
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    #pdists will store the 9 smallest distances
    pdists = np.sort(pairwise_dist.flatten())[0:num_results]

    #data frame indices of the 9 smallest distance's
    df_indices = list(data.index[indices])
    for i in range(len(indices)):
        rows = data[['medium_image_url','title']].loc[data['asin']==asins[indices[i]]]
        for indx, row in rows.iterrows():
            display(Image(url=row['medium_image_url'], embed=True))
            print('Product Title: ', row['title'])
            print('Euclidean Distance from input image:', pdists[i])
            print('Amazon Url: www.amazon.com/dp/' + asins[indices[i]]))

    idf_w2v_brand_color_cnn(12566, 5,5 ,5, 20)
    # in the give heat map, each cell contains the euclidean distance between words i, j

```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 3.50476844597102

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 13.771039440209613

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 17.173916411956426

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 18.3198087459408

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 19.304608408730484

Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: kawaii pastel tops tees pink flower design

Euclidean Distance from input image: 19.425534057617188

Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: summer alpaca 3d pastel casual loose tops tee design

Euclidean Distance from input image: 19.478585561116535

Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: womens thin style tops pastel watermelon print

Euclidean Distance from input image: 19.581652832272166

Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: chicago chicago 18 shirt women pink

Euclidean Distance from input image: 19.5831621808236

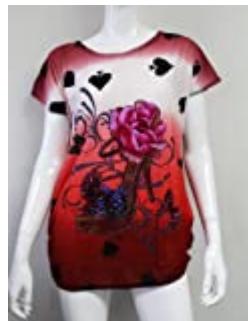
Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: edv cheetah run purple multi xl

Euclidean Distance from input image: 19.594672942718145

Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: red pink floral heel sleeveless shirt xl xxl

Euclidean Distance from input image: 19.639882405839877

Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: womens tops tees pastel peach ice cream cone print

Euclidean Distance from input image: 19.647568257890658

Amazon Url: www.amazon.com/dp/B0734GRKZL



Product Title: foxcroft nyc womens pinpoint oxford shirt noniron stretch poplin blouse xlarge white

Euclidean Distance from input image: 19.68741658528646

Amazon Url: www.amazon.com/dp/B072277HVB



Product Title: miss chievous juniors striped peplum tank top medium shadow peach

Euclidean Distance from input image: 19.697577921790398

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: dasy womens neck electric light orchestra logo shirt xxlarge black

Euclidean Distance from input image: 19.718012532154994

Amazon Url: www.amazon.com/dp/B016EXUZC4



Product Title: cling womens hardy boyz wwe logo vneck tshirt black

Euclidean Distance from input image: 19.7685994468555

Amazon Url: www.amazon.com/dp/B01A82FFBI



Product Title: kingde star pink flower dog stamp sleeveless vestbqn24

Euclidean Distance from input image: 19.785620666106816

Amazon Url: www.amazon.com/dp/B015H3W9BM



Product Title: women japanese mixed martial arts teeblack tshirts hgiorgis xxl black

Euclidean Distance from input image: 19.810214528004604

Amazon Url: www.amazon.com/dp/B01JQ096HW



Product Title: danskin womens vneck loose performance tee xsmall pink ombre

Euclidean Distance from input image: 19.816824595133465

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: officially licensed merchandise popeye welcome gunshow gi
rly tshirt red xxlarge

Euclidean Distance from input image: 19.818007709671217

Amazon Url: www.amazon.com/dp/B01BFLPAK2

Smaller equal weights for all

In [23]:

```
idf_w2v_brand_color_cnn(12566, 5,5 ,5, 5)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 3.50476844597102

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 13.771039440209613

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 17.173916411956426

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 18.3198087459408

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 19.304608408730484

Amazon Url: www.amazon.com/dp/B01KUN4VWE

More weight for idf

In [24]:

```
idf_w2v_brand_color_cnn(12566, 100, 5 ,5, 5)
```



Product Title: foxcroft nyc womens pinpoint oxford shirt noniron stretch poplin blouse xl
white

Euclidean Distance from input image: 2.684647716175426

Amazon Url: www.amazon.com/dp/B072277HVB



Product Title: kirkland signature ladies size medium cotton vneck tee white

Euclidean Distance from input image: 6.216651985878465

Amazon Url: www.amazon.com/dp/B073ZRD5FL



Product Title: exotic india yellow gray jamawar wrap fauxfur collar

Euclidean Distance from input image: 7.025206409801136

Amazon Url: www.amazon.com/dp/B073ZHRBV8



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 7.1157216953801035

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: miss chievous womens short sleeve knit top wsharkbite hem charcoal 1x

Euclidean Distance from input image: 7.636244271034715

Amazon Url: www.amazon.com/dp/B06XR9NTVL

More weight for brand and color

In [25]:

```
idf_w2v_brand_color_cnn(12566, 5,100 ,5, 20)
```



Product Title: foxcroft nyc womens pinpoint oxford shirt noniron stretch poplin blouse xlarge white

Euclidean Distance from input image: 2.684647716175426

Amazon Url: www.amazon.com/dp/B072277HVB



Product Title: exotic india yellow gray jamawar wrap fauxfur collar

Euclidean Distance from input image: 2.9048481334339487

Amazon Url: www.amazon.com/dp/B073ZHRBV8



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 2.9206554867313943

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 3.8538128596036483

Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: kirkland signature ladies size medium cotton vneck tee white

Euclidean Distance from input image: 3.928297892812322

Amazon Url: www.amazon.com/dp/B073ZRD5FL



Product Title: kirkland signature ladies striped active yoga tee large green

Euclidean Distance from input image: 4.05578072320672

Amazon Url: www.amazon.com/dp/B01LX08SGX



Product Title: fylo ladies size medium 14 zip vneck blouse pinkgrey animal

Euclidean Distance from input image: 4.11497112653813

Amazon Url: www.amazon.com/dp/B073X71H8K



Product Title: blouseuploter cute cartoon vest summer pajamas

Euclidean Distance from input image: 4.185148308822399

Amazon Url: www.amazon.com/dp/B01LPJRYI0



Product Title: material girl womens juniors cutout swing shift caviar black combo xxl

Euclidean Distance from input image: 4.192392176176358

Amazon Url: www.amazon.com/dp/B072L5CGR3



Product Title: divas love snow henley long sleeve tshirt 4xlarge violettu rquoise

Euclidean Distance from input image: 4.222756646224742

Amazon Url: www.amazon.com/dp/B071NHMRB2



Product Title: miss chievous womens short sleeve knit top wsharkbite hem charcoal 1x

Euclidean Distance from input image: 4.2283809838545805

Amazon Url: www.amazon.com/dp/B06XR9NTVL



Product Title: tahari asl ruffledfront polkadot top blackwhite xl

Euclidean Distance from input image: 4.2348922906172755

Amazon Url: www.amazon.com/dp/B074MZTZS5



Product Title: ideology womens small crewneck solid tee tshirt black

Euclidean Distance from input image: 4.272896090749334

Amazon Url: www.amazon.com/dp/B0721T22DL



Product Title: hippie rose juniors marled dolmansleeve tunic top size

Euclidean Distance from input image: 4.284973769082963

Amazon Url: www.amazon.com/dp/B072HXX3W7



Product Title: maison margiela mm6 beige 3/4 sleeve womens blouse top us 4

2

Euclidean Distance from input image: 4.293816063949351

Amazon Url: www.amazon.com/dp/B01N70M7H4



Product Title: romeo juliet navy womens medium jeweled knit top green

Euclidean Distance from input image: 4.302626592791237

Amazon Url: www.amazon.com/dp/B071P49MCK



Product Title: socialite teal women medium crocheted offshoulder blouse blue

Euclidean Distance from input image: 4.395271370956188

Amazon Url: www.amazon.com/dp/B01M0DVT93



Product Title: yellow tiger tshirt tiger stripes 1

Euclidean Distance from input image: 4.626864733913478

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: hnan womens death note ryuk silhouette oneck tshirt pink x 1

Euclidean Distance from input image: 4.724947994903928

Amazon Url: www.amazon.com/dp/B011I80ET4



Product Title: confess juniors plaid highlow buttonfront shirt size xs

Euclidean Distance from input image: 4.741856709891944

Amazon Url: www.amazon.com/dp/B06XHNGBV6

More weight for cnn features

In [26]:

```
idf_w2v_brand_color_cnn(12566, 5,5 ,100, 20)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 0.47792843323395884

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 27.83028933595188

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 37.97649990978028

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 40.49829070900791

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: kawaii pastel tops tees pink flower design
Euclidean Distance from input image: 43.57016962224787
Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: womens thin style tops tees pastel watermelon print
Euclidean Distance from input image: 43.881586386973545
Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: kawaii pastel tops tees baby blue flower design
Euclidean Distance from input image: 44.0727749619376
Amazon Url: www.amazon.com/dp/B071SBCY9W



Product Title: edv cheetah run purple multi xl
Euclidean Distance from input image: 44.166692045677834
Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: summer alpaca 3d pastel casual loose tops tee design

Euclidean Distance from input image: 44.213374293934216

Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: danskin womens vneck loose performance tee xs small pink ombre

Euclidean Distance from input image: 44.24478888078169

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: miss chievous juniors striped peplum tank top medium shadow peach

Euclidean Distance from input image: 44.253905695167795

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: red pink floral heel sleeveless shirt xl xxl

Euclidean Distance from input image: 44.279052873124115

Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: abaday multicolor cartoon cat print short sleeve longline shirt large

Euclidean Distance from input image: 44.38771972659535

Amazon Url: www.amazon.com/dp/B01CR57YY0



Product Title: moana logo adults hot v neck shirt black xxl

Euclidean Distance from input image: 44.41747901226208

Amazon Url: www.amazon.com/dp/B01LX6H43D



Product Title: chicago chicago 18 shirt women pink

Euclidean Distance from input image: 44.45627049536088

Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: kawaii cotton pastel tops tees peach pink cactus design

Euclidean Distance from input image: 44.47038265575062

Amazon Url: www.amazon.com/dp/B071WYLBZS



Product Title: yichun womens tiger printed summer tshirts tops

Euclidean Distance from input image: 44.55524312799627

Amazon Url: www.amazon.com/dp/B010NN9RX0



Product Title: nancy lopez whimsy short sleeve whiteblacklemon drop xs

Euclidean Distance from input image: 44.591757554433606

Amazon Url: www.amazon.com/dp/B01MPX6IDX



Product Title: womens tops tees pastel peach ice cream cone print

Euclidean Distance from input image: 44.615633114934305

Amazon Url: www.amazon.com/dp/B0734GRKZL



Product Title: women japanese mixed martial arts teeblack tshirts hgiorgis xxl black

Euclidean Distance from input image: 44.695044679067394

Amazon Url: www.amazon.com/dp/B01JQ096HW

More weight for all features

In [27]:

```
idf_w2v_brand_color_cnn(12566, 100,100 ,100, 20)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 3.5047685476962323

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 13.771039440209613

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 17.173917632659553

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 18.319808949391323

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 19.30460744234051

Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: kawaii pastel tops tees pink flower design

Euclidean Distance from input image: 19.425534261067707

Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: summer alpaca 3d pastel casual loose tops tee design

Euclidean Distance from input image: 19.478586018880208

Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: womens thin style tops tees pastel watermelon print

Euclidean Distance from input image: 19.58165323917321

Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: chicago chicago 18 shirt women pink
Euclidean Distance from input image: 19.583161214433627
Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: edv cheetah run purple multi xl
Euclidean Distance from input image: 19.59467406169601
Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: red pink floral heel sleeveless shirt xl xxl
Euclidean Distance from input image: 19.639882202389355
Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: womens tops tees pastel peach ice cream cone print
Euclidean Distance from input image: 19.647569376868525
Amazon Url: www.amazon.com/dp/B0734GRKZL



Product Title: foxcroft nyc womens pinpoint oxford shirt noniron stretch poplin blouse xlarge white

Euclidean Distance from input image: 19.687415364583334

Amazon Url: www.amazon.com/dp/B072277HVB



Product Title: miss chievous juniors striped peplum tank top medium shade wpeach

Euclidean Distance from input image: 19.697577311438835

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: dasy womens neck electric light orchestra logo shirt xxlarge black

Euclidean Distance from input image: 19.718012481292366

Amazon Url: www.amazon.com/dp/B016EXUZC4



Product Title: cling womens hardy boyz wwe logo vneck tshirt black

Euclidean Distance from input image: 19.768599853756545

Amazon Url: www.amazon.com/dp/B01A82FFBI



Product Title: kingde star pink flower dog stamp sleeveless vestbqn24

Euclidean Distance from input image: 19.78562092041997

Amazon Url: www.amazon.com/dp/B015H3W9BM



Product Title: women japanese mixed martial arts teeblack tshirts hgiorgis xxl black

Euclidean Distance from input image: 19.81021381592778

Amazon Url: www.amazon.com/dp/B01JQ096HW



Product Title: danskin womens vneck loose performance tee xsmall pink ombre

Euclidean Distance from input image: 19.816824137369792

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: officially licensed merchandise popeye welcome gunshow gi
rly tshirt red xxlarge

Euclidean Distance from input image: 19.81800842174804

Amazon Url: www.amazon.com/dp/B01BFLPAK2

More weight for idf, brand and color

In [28]:

```
idf_w2v_brand_color_cnn(12566, 100, 100, 5, 20)
```



Product Title: foxcroft nyc womens pinpoint oxford shirt noniron stretch poplin blouse xlARGE white

Euclidean Distance from input image: 1.4405426769721799

Amazon Url: www.amazon.com/dp/B072277HVB



Product Title: exotic india yellow gray jamawar wrap fauxfur collar

Euclidean Distance from input image: 3.76962295160061

Amazon Url: www.amazon.com/dp/B073ZHRBV8



Product Title: kirkland signature ladies size medium cotton vneck tee white

Euclidean Distance from input image: 3.991131740839391

Amazon Url: www.amazon.com/dp/B073ZRD5FL



Product Title: miss chievous womens short sleeve knit top wsharkbite hem charcoal 1x

Euclidean Distance from input image: 4.752864186532989

Amazon Url: www.amazon.com/dp/B06XR9NTVL



Product Title: kirkland signature ladies striped active yoga tee large green

Euclidean Distance from input image: 4.814501655568812

Amazon Url: www.amazon.com/dp/B01LX08SGX



Product Title: blouseuploader cute cartoon vest summer pajamas

Euclidean Distance from input image: 4.882297423004025

Amazon Url: www.amazon.com/dp/B01LPJRYI0



Product Title: tahari asl ruffledfront polkadot top blackwhite xl

Euclidean Distance from input image: 5.0034886803877905

Amazon Url: www.amazon.com/dp/B074MZTZS5



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 5.128926650452192

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: fylo ladies size medium 14 zip vneck blouse pinkgrey anima
1

Euclidean Distance from input image: 5.379369075300138

Amazon Url: www.amazon.com/dp/B073X71H8K



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 5.488643739457912

Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: hippie rose juniors marled dolmansleeve tunic top size

Euclidean Distance from input image: 5.5401845049876375

Amazon Url: www.amazon.com/dp/B072HXX3W7



Product Title: divas love snow henley long sleeve tshirt 4xlarge violettu
rquoise

Euclidean Distance from input image: 5.58343595196744

Amazon Url: www.amazon.com/dp/B071NHMRB2



Product Title: hanro womens leonie 3/4 sleeve shirt 76214 dim brown xsmall

Euclidean Distance from input image: 5.681882368297514

Amazon Url: www.amazon.com/dp/B06Y3YRLP1



Product Title: boden womens embellished safari spot top us sz 4 brownblac
kwhite

Euclidean Distance from input image: 5.77027898962945

Amazon Url: www.amazon.com/dp/B01H62TIZZ



Product Title: material girl womens juniors cutout swing shift caviar bla
ck combo xxl

Euclidean Distance from input image: 5.792166212304711

Amazon Url: www.amazon.com/dp/B072L5CGR3



Product Title: h bordeaux womens plus short sleeve hilow hem vneck shirt
1x peony

Euclidean Distance from input image: 5.829908715577409

Amazon Url: www.amazon.com/dp/B01I75WDQ8



Product Title: style co plus size shortsleeve flouncehem top 2x pink

Euclidean Distance from input image: 5.8303718226134436

Amazon Url: www.amazon.com/dp/B01N6U2KIT



Product Title: kingsize big tall gauze laceup shirt bright orangetall x
1

Euclidean Distance from input image: 5.864200282945782

Amazon Url: www.amazon.com/dp/B06Y4R5Q33



Product Title: ira soleil womens ethnic viscose knit red pink anarkali s hrug kurta kurti large red

Euclidean Distance from input image: 5.865545192230441

Amazon Url: www.amazon.com/dp/B00P9790T2



Product Title: cavalli womens sparkle loose style blouse top us 40

Euclidean Distance from input image: 5.876851582760167

Amazon Url: www.amazon.com/dp/B01CAIM90E

More weight for cnn, brand and color

In [29]:

```
idf_w2v_brand_color_cnn(12566, 5,100 ,100, 20)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 1.5671839244225538

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 16.470298448968144

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 21.60371397379008

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 22.956870012400028

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: kawaii pastel tops tees pink flower design
Euclidean Distance from input image: 24.769359309498856
Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: womans wicked musical poster shirts red xxl
Euclidean Distance from input image: 24.829900881315528
Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: womens thin style tops tees pastel watermelon print
Euclidean Distance from input image: 24.85695160691697
Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: kawaii pastel tops tees baby blue flower design
Euclidean Distance from input image: 24.874885952996443
Amazon Url: www.amazon.com/dp/B071SBCY9W



Product Title: edv cheetah run purple multi xl

Euclidean Distance from input image: 24.925280485735108

Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: miss chievous juniors striped peplum tank top medium shadow peach

Euclidean Distance from input image: 25.05673269911876

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: red pink floral heel sleeveless shirt xl xxl

Euclidean Distance from input image: 25.070226306802642

Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: summer alpaca 3d pastel casual loose tops tee design

Euclidean Distance from input image: 25.114493523574456

Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: abaday multicolor cartoon cat print short sleeve longline shirt large

Euclidean Distance from input image: 25.12853535012867

Amazon Url: www.amazon.com/dp/B01CR57YY0



Product Title: danskin womens vneck loose performance tee xsmall pink ombre

Euclidean Distance from input image: 25.13135013115115

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: moana logo adults hot v neck shirt black xxl

Euclidean Distance from input image: 25.144503747315692

Amazon Url: www.amazon.com/dp/B01LX6H43D



Product Title: nancy lopez whimsy short sleeve whiteblacklemon drop xs

Euclidean Distance from input image: 25.153364417262594

Amazon Url: www.amazon.com/dp/B01MPX6IDX



Product Title: chicago chicago 18 shirt women pink

Euclidean Distance from input image: 25.165318689466275

Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: women japanese mixed martial arts teeblack tshirts hgiorgis xxl black

Euclidean Distance from input image: 25.208786776822187

Amazon Url: www.amazon.com/dp/B01JQ096HW



Product Title: kingde star pink flower dog stamp sleeveless vestbqn24

Euclidean Distance from input image: 25.229777902161604

Amazon Url: www.amazon.com/dp/B015H3W9BM



Product Title: womens tops tees pastel peach ice cream cone print

Euclidean Distance from input image: 25.250830338993474

Amazon Url: www.amazon.com/dp/B0734GRKZL

More weight for idf, cnn features

In [30]:

```
idf_w2v_brand_color_cnn(12566, 100, 5 ,100, 20)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 3.8181950607706416

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 18.615768180385956

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 23.906482513398682

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 25.583396635445176

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: kawaii pastel tops tees pink flower design
Euclidean Distance from input image: 27.037367211318596
Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: summer alpaca 3d pastel casual loose tops tee design
Euclidean Distance from input image: 27.115003930068596
Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: womens thin style tops tees pastel watermelon print
Euclidean Distance from input image: 27.34534338976134
Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: chicago chicago 18 shirt women pink
Euclidean Distance from input image: 27.347550182825362
Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: red pink floral heel sleeveless shirt xl xxl

Euclidean Distance from input image: 27.430556506663017

Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: womens tops tees pastel peach ice cream cone print

Euclidean Distance from input image: 27.441806030291065

Amazon Url: www.amazon.com/dp/B0734GRKZL



Product Title: edv cheetah run purple multi xl

Euclidean Distance from input image: 27.449052897354473

Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: miss chievous juniors striped peplum tank top medium shadow peach

Euclidean Distance from input image: 27.51498837356469

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 27.59527777695956

Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: danskin womens vneck loose performance tee xsmall pink ombre

Euclidean Distance from input image: 27.60998654249238

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: kawaii cotton pastel tops tees peach pink cactus design

Euclidean Distance from input image: 27.614131871665396

Amazon Url: www.amazon.com/dp/B071WYLBZS



Product Title: cling womens hardy boyz wwe logo vneck tshirt black

Euclidean Distance from input image: 27.618923801346703

Amazon Url: www.amazon.com/dp/B01A82FFBI



Product Title: dasy womens neck electric light orchestra logo shirt xxlarge black
Euclidean Distance from input image: 27.62954814554426

Amazon Url: www.amazon.com/dp/B016EXUZC4



Product Title: new balance womens blackpink love ladies shirt us small
Euclidean Distance from input image: 27.660038943407013

Amazon Url: www.amazon.com/dp/B071ZM3N95



Product Title: kawaii cotton pastel tops tees pink flower design
Euclidean Distance from input image: 27.691510381948838

Amazon Url: www.amazon.com/dp/B071P4YKH5



Product Title: abaday multicolor cartoon cat print short sleeve longline shirt large
Euclidean Distance from input image: 27.704800266190453

Amazon Url: www.amazon.com/dp/B01CR57YY0

Random values

In [31]:

```
idf_w2v_brand_color_cnn(12566, 10,50 ,70, 20)
```



Product Title: burnt umber tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 1.6790809071275064

Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl

Euclidean Distance from input image: 18.067741759436572

Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes l

Euclidean Distance from input image: 23.820880584258614

Amazon Url: www.amazon.com/dp/B00JXQCUIC



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 25.349490344249602

Amazon Url: www.amazon.com/dp/B00JXQCWT0



Product Title: kawaii pastel tops tees pink flower design
Euclidean Distance from input image: 27.274730036808894
Amazon Url: www.amazon.com/dp/B071FCWD97



Product Title: womens thin style tops tees pastel watermelon print
Euclidean Distance from input image: 27.413013869343487
Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: kawaii pastel tops tees baby blue flower design
Euclidean Distance from input image: 27.496713244640226
Amazon Url: www.amazon.com/dp/B071SBCY9W



Product Title: edv cheetah run purple multi xl
Euclidean Distance from input image: 27.51122376616216
Amazon Url: www.amazon.com/dp/B01CUPYBM0



Product Title: womans wicked musical poster shirts red xxl

Euclidean Distance from input image: 27.55989996117039

Amazon Url: www.amazon.com/dp/B01KUN4VWE



Product Title: summer alpaca 3d pastel casual loose tops tee design

Euclidean Distance from input image: 27.626842029278095

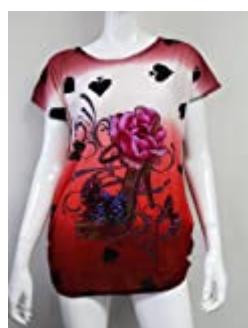
Amazon Url: www.amazon.com/dp/B01I80A93G



Product Title: miss chievous juniors striped peplum tank top medium shade wpeach

Euclidean Distance from input image: 27.630290339821254

Amazon Url: www.amazon.com/dp/B0177DM70S



Product Title: red pink floral heel sleeveless shirt xl xxl

Euclidean Distance from input image: 27.63461843665118

Amazon Url: www.amazon.com/dp/B00JV63QQE



Product Title: danskin womens vneck loose performance tee xsmall pink ombre

Euclidean Distance from input image: 27.69703867985652

Amazon Url: www.amazon.com/dp/B01F7PHXY8



Product Title: chicago chicago 18 shirt women pink

Euclidean Distance from input image: 27.72037253774418

Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: abaday multicolor cartoon cat print short sleeve longline shirt large

Euclidean Distance from input image: 27.722283466323656

Amazon Url: www.amazon.com/dp/B01CR57YY0



Product Title: moana logo adults hot v neck shirt black xxl

Euclidean Distance from input image: 27.78153956514427

Amazon Url: www.amazon.com/dp/B01LX6H43D



Product Title: nancy lopez whimsy short sleeve whiteblacklemon drop xs

Euclidean Distance from input image: 27.798158545402696

Amazon Url: www.amazon.com/dp/B01MPX6IDX



Product Title: womens tops tees pastel peach ice cream cone print

Euclidean Distance from input image: 27.815703524354007

Amazon Url: www.amazon.com/dp/B0734GRKZL



Product Title: kawaii cotton pastel tops tees peach pink cactus design

Euclidean Distance from input image: 27.81804428100586

Amazon Url: www.amazon.com/dp/B071WYLBZS



Product Title: women japanese mixed martial arts teeblack tshirts hgiorgi s xxl black

Euclidean Distance from input image: 27.827617104879106

Amazon Url: www.amazon.com/dp/B01JQ096HW

1. Had considered 16k datapoints for above experiment
2. Had built 'idf-w2v-weightage' , 'Brand & Color' and 'Image feature' weightage feature model.
3. Had given max and min weightage to these features interchangeably.