

Amazon Apparel Recommendations

[4.2] Data and Code:

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

[4.3] Overview of the data

In [1]: *#import all the necessary packages.*

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

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

In [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('tops_fashion.json')*

```
In [0]: 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 [0]: # each product/item has 19 features in the raw dataset.
data.columns # prints column-names or feature-names.
```

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

Of these 19 features, we will be using only 6 features in this workshop.

1. asin (Amazon standard identification number)
2. brand (brand to which the product belongs to)
3. color (Color information of apparel, it can contain many colors as a value ex: red and black stripes)
4. product_type_name (type of the 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 [0]: 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[0]:

	asin	brand	color	medium_image_url	product_type_name	title	formatted_price
0	B016I2TS4W	FNC7C	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	Minions Como Superheroes Ironman Long Sleeve R...	None
1	B01N49AI08	FIG Clothing	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	FIG Clothing Womens Izo Tunic	None
2	B01JDPCOHO	FIG Clothing	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	FIG Clothing Womens Won Top	None
3	B01N19U5H5	Focal18	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	Focal18 Sailor Collar Bubble Sleeve Blouse Shi...	None
4	B004GSI2OS	FeatherLite	Onyx Black/Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	Featherlite Ladies' Long Sleeve Stain Resistan...	\$26.26

[5.1] Missing data for various features.

Basic stats for the feature: product_type_name

```
In [0]: # 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 [0]: # 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 [0]: # find the 10 most frequent product_type_names.  
product_type_count = Counter(list(data['product_type_name']))  
product_type_count.most_common(10)
```

```
Out[0]: [('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 [0]: # 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 [0]: brand_count = Counter(list(data['brand']))
brand_count.most_common(10)
```

```
Out[0]: [('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 [0]: 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 [0]: color_count = Counter(list(data['color']))
color_count.most_common(10)
```

```
Out[0]: [(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 [0]:
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 [0]: price_count = Counter(list(data['formatted_price']))
price_count.most_common(10)
```

```
Out[0]: [(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 [0]: 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 [0]: data.to_pickle('pickels/180k_apparel_data')
```

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

```
In [0]: # 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 [0]: # 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 [0]: data.to_pickle('pickels/28k_apparel_data')
```

```
In [0]: # 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[0]: "\nfrom PIL import Image\nimport requests\nfrom io import BytesIO\nfor 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('workshop/images/28k_images/'+row['asin']+'.jpeg')\n\n"
```

[5.2] Remove near duplicate items

[5.2.1] Understand about duplicates.

```
In [0]: # read data from pickle file from previous stage  
data = pd.read_pickle('pickels/28k_apparel_data')  
  
# find number of products that have duplicate titles.  
print(sum(data.duplicated('title')))  
# we have 2325 products which have same title but different color
```

2325

These shirts are exactly same except in size (S, M,L,XL)

	:B00AQ4GMCK		:B00AQ4GMTS
	:B00AQ4GMLQ		:B00AQ4GN3I

These shirts exactly same except in color

	:B00G278GZ6		:B00G278W6O
	:B00G278Z2A		:B00G278X8

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('pickels/28k_apparel_data')
```

In [0]: `data.head()`

Out[0]:

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

In [0]: `# 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 [0]: # Sort the whole data based on title (alphabetical order of title)
data_sorted.sort_values('title', inplace=True, ascending=False)
data_sorted.head()
```

Out[0]:

	asin	brand	color	medium_image_url	product_type_name	title	formatted_price
61973	B06Y1KZ2WB	Éclair	Black/Pink	https://images-na.ssl-images-amazon.com/images...	SHIRT	Éclair Women's Printed Thin Strap Blouse Black...	\$24.99
133820	B010RV33VE	xiaoming	Pink	https://images-na.ssl-images-amazon.com/images...	SHIRT	xiaoming Womens Sleeveless Loose Long T-shirts...	\$18.19
81461	B01DDSDLNS	xiaoming	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	xiaoming Women's White Long Sleeve Single Brea...	\$21.58
75995	B00X5LYO9Y	xiaoming	Red Anchors	https://images-na.ssl-images-amazon.com/images...	SHIRT	xiaoming Stripes Tank Patch/Bear Sleeve Anchor...	\$15.91
151570	B00WPJG35K	xiaoming	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	xiaoming Sleeve Sheer Loose Tassel Kimono Woma...	\$14.32

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 appearals are different
        # if the number of words in which both strings differ are < 2 , we are considering it as those two appear
```

```
als 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]])

    # if the comaprision between is between num_data_points, num_data_points-1 strings and they differ in more than 2 words we include both
    if j == num_data_points-1: stage1_dedupe_asins.append(data_sorted['asin'].loc[indices[j]])

    # start searching for similar apperals 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 [0]: print('Number of data points : ', data.shape[0])
Number of data points : 17593
```

```
In [0]: data.to_pickle('pickels/17k_apperial_data')
```

[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, XX-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('pickels/17k_apperial_data')
```

```
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 appended 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 apparel 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_deduplicate_asins)]
```

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

Number of data points after stage two of dedupe: 16042

```
In [0]: 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.
```

6. Text pre-processing

```
In [0]: data = pd.read_pickle('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 [0]: # we use the list of stop words that are downloaded from nltk lib.
stop_words = set(stopwords.words('english'))
print ('list of stop words:', stop_words)

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

list of stop words: {'such', 'and', 'hers', 'up', 'she', 'd', 'further', 'all', 'than', 'under', 'is', 'off', 'b
oth', 'most', 'few', 'should', 're', 'very', 'just', 'then', 'didn', 'myself', 'in', 'too', 's', 'shouldn', 'her
self', 'because', 'how', 'itself', 'what', 'shan', 'weren', 'doing', 'them', 'couldn', 'their', 'so', 'ain', 'ha
ven', 'yourself', 'now', 'll', 'isn', 'about', 'over', 'into', 'before', 'during', 'on', 'as', 'aren', 'agains
t', 'above', 'down', 'they', 'below', 'me', 'again', 'for', 'why', 'been', 'yourselves', 'more', 'her', 'that',
'can', 'am', 'was', 'themselves', 'mightn', 'does', 'those', 'only', 'hasn', 'any', 'ma', 'are', 'nor', 'out',
'you', 'ourselves', 'the', 'an', 'has', 'where', 'i', 'while', 'ours', 'its', 'your', 'had', 'were', 'being', 'n
o', 'or', 'needn', 've', 'y', 'a', 'each', 'have', 'through', 'when', 'mustn', 'by', 'won', 'from', 'own', 'wil
l', 'there', 't', 'him', 'these', 'doesn', 'theirs', 'my', 'did', 'of', 'who', 'until', 'wouldn', 'we', 'do', 'h
aving', 'yours', 'other', 'wasn', 'it', 'with', 'once', 'here', 'don', 'o', 'whom', 'this', 'if', 'but', 'hadn',
'our', 'some', 'm', 'not', 'between', 'himself', 'same', 'at', 'be', 'he', 'after', 'which', 'to', 'his'}

```
In [0]: start_time = time.clock()
# we take each title and we text-preprocess it.
for index, row in data.iterrows():
    nlp_preprocessing(row['title'], index, 'title')
# we print the time it took to preprocess whole titles
print(time.clock() - start_time, "seconds")
```

3.572722000000006 seconds

In [0]: `data.head()`

Out[0]:

	asin	brand	color	medium_image_url	product_type_name	title	formatted_price
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	featherlite ladies long sleeve stain resistant...	\$26.26
6	B012YX2ZPI	HX-Kingdom Fashion T-shirts	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	womens unique 100 cotton special olympics wor...	\$9.99
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl-images-amazon.com/images...	SHIRT	featherlite ladies moisture free mesh sport sh...	\$20.54
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl-images-amazon.com/images...	SHIRT	supernatural chibis sam dean castiel neck tshi...	\$7.39
46	B01NACPBG2	Fifth Degree	Black	https://images-na.ssl-images-amazon.com/images...	SHIRT	fifth degree womens gold foil graphic tees jun...	\$6.95

In [0]: `data.to_pickle('pickels/16k_apperial_data_preprocessed')`

Stemming

```
In [0]: 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 [23]: data = pd.read_pickle('./pickels/16k_apperial_data_preprocessed')
data.head()
```

Out[23]:

	asin	brand	color	medium_image_url	product_type_name	title	formatted_price
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl- images- amazon.com/images...	SHIRT	featherlite ladies long sleeve stain resistant...	\$26.26
6	B012YX2ZPI	HX-Kingdom Fashion T- shirts	White	https://images-na.ssl- images- amazon.com/images...	SHIRT	womens unique 100 cotton special olympics wor...	\$9.99
15	B003BSRPB0	FeatherLite	White	https://images-na.ssl- images- amazon.com/images...	SHIRT	featherlite ladies moisture free mesh sport sh...	\$20.54
27	B014ICEJ1Q	FNC7C	Purple	https://images-na.ssl- images- amazon.com/images...	SHIRT	supernatural chibis sam dean castiel neck tshi...	\$7.39
46	B01NACPBG2	Fifth Degree	Black	https://images-na.ssl- images- amazon.com/images...	SHIRT	fifth degree womens gold foil graphic tees jun...	\$6.95

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 parameter 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 recommended 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 titles 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 {word : 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 [25]: 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 occured in that doc
```

Out[25]: (16042, 12609)

```
In [26]: 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  $K(X, Y) = \langle X, Y \rangle / (\|X\| * \|Y\|)$ 
        # 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

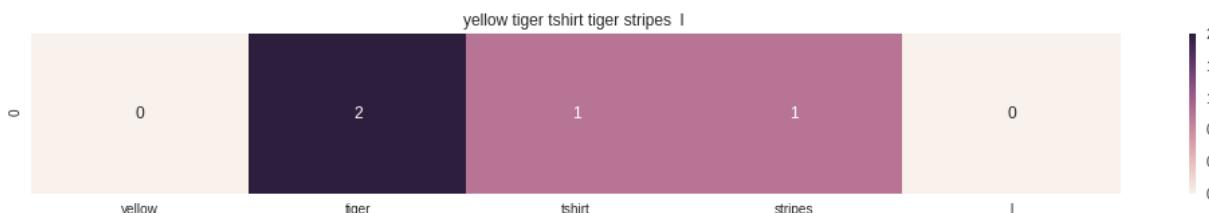


ASIN : B00JXQCWT0

Brand: Si Row

Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean similarity with the query image : 2.449489742783178

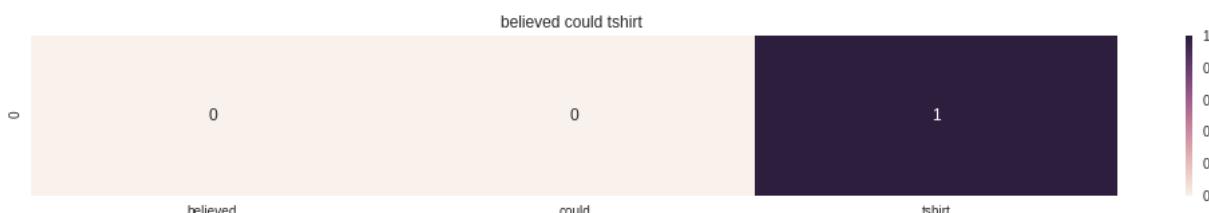


ASIN : B00JXQCUIC

Brand: Si Row

Title: yellow tiger tshirt tiger stripes 1

Euclidean similarity with the query image : 2.6457513110645907

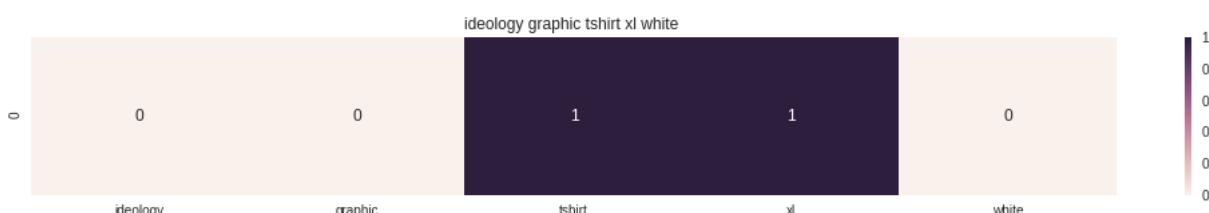


ASIN : B07568NZX4

Brand: Rustic Grace

Title: believed could tshirt

Euclidean similarity with the query image : 3.0



ASIN : B01NB0NKR0

Brand: Ideology

Title: ideology graphic tshirt xl white

Euclidean similarity with the query image : 3.0

AmazonFashionDiscoveryEngine (1)



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

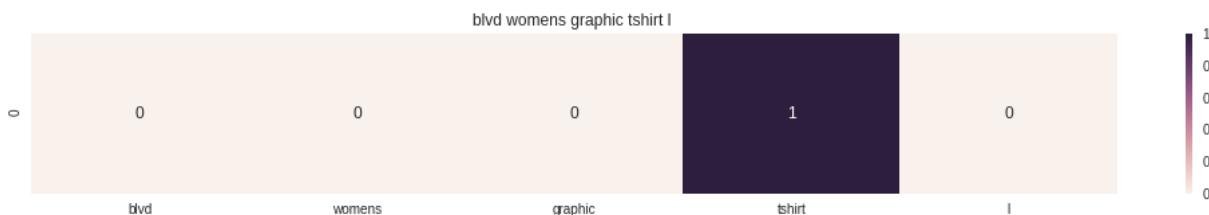


ASIN : B01KVZUB6G

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

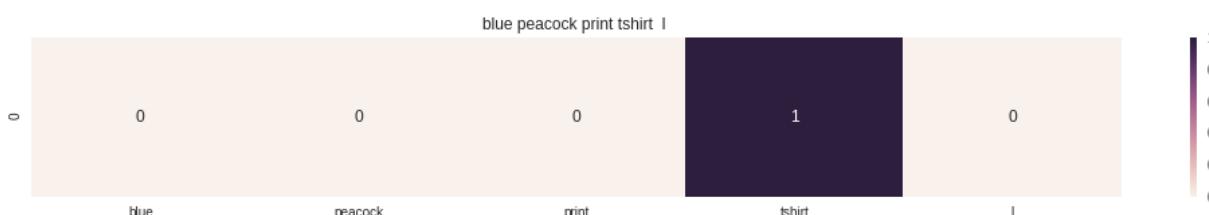


ASIN : B012VQLT6Y

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

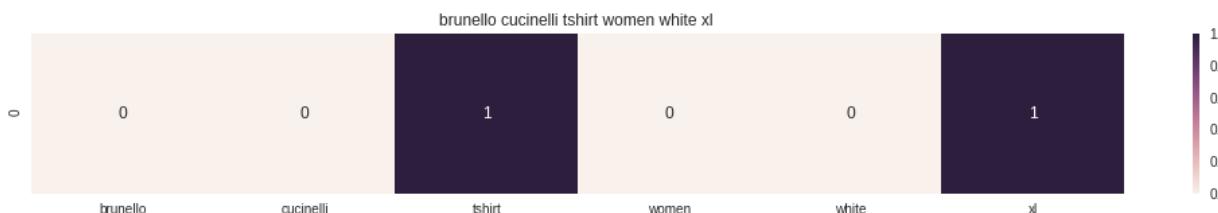


ASIN : B0088PN0LA

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

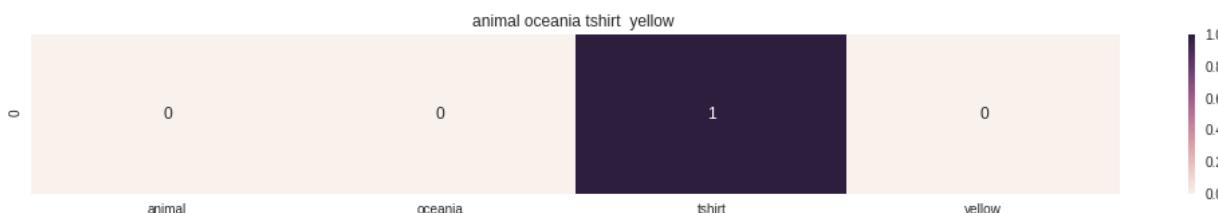


ASIN : B06Y1JPW1Q

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 sparse matrix of dimensions #data_points * #words_in_c
orpus
# tfidf_title_features[doc_id, index_of_word_in_corpus] = tfidf values of the word in given doc
```

```
In [28]: 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  $K(X, Y) = \langle X, Y \rangle / (\|X\| * \|Y\|)$ 
    # 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 : 0.7536331912451361

=====



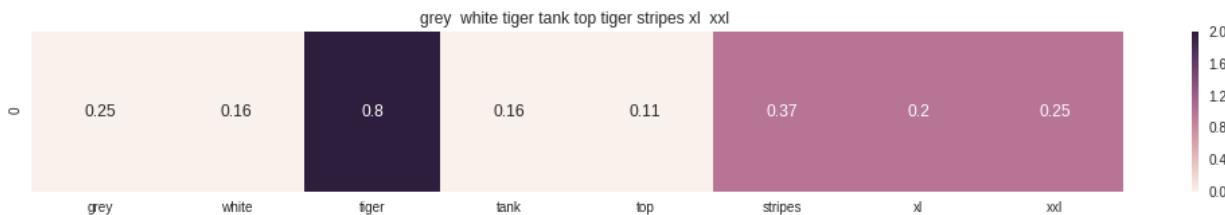
ASIN : B00JXQCWT0

BRAND : Si Row

Eucliden distance from the given image : 0.9357643943769645

=====

AmazonFashionDiscoveryEngine (1)

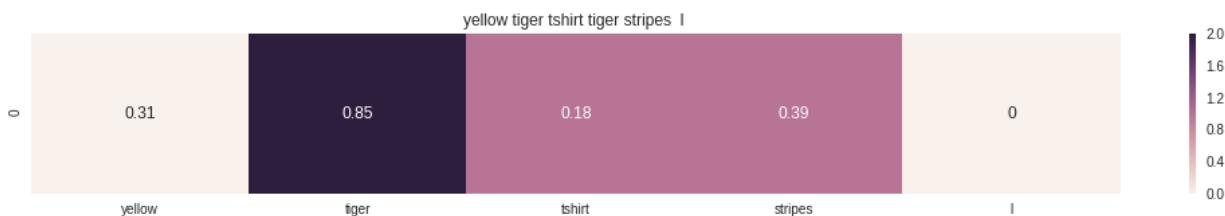


ASIN : B00JXQAFZ2

BRAND : Si Row

Eucliden distance from the given image : 0.9586153524200749

=====

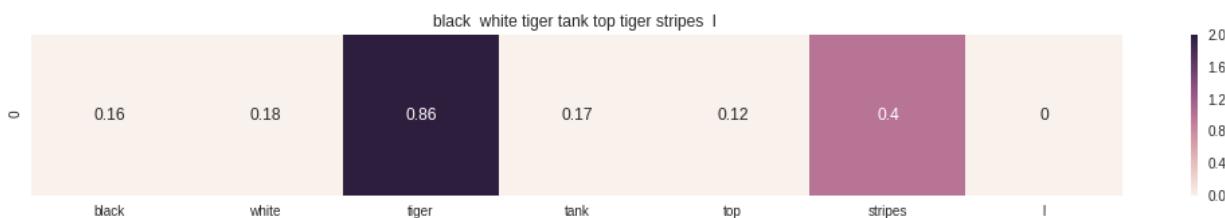


ASIN : B00JXQCUIC

BRAND : Si Row

Eucliden distance from the given image : 1.000074961446881

=====



ASIN : B00JXQA094

BRAND : Si Row

Eucliden distance from the given image : 1.023215552457452

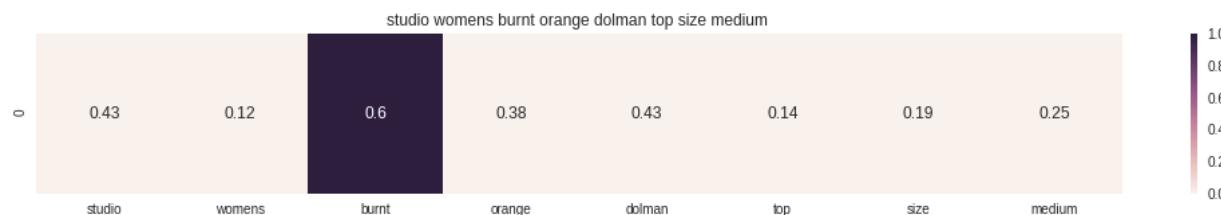
=====



ASIN : B00JXQAUWA

BRAND : Si Row

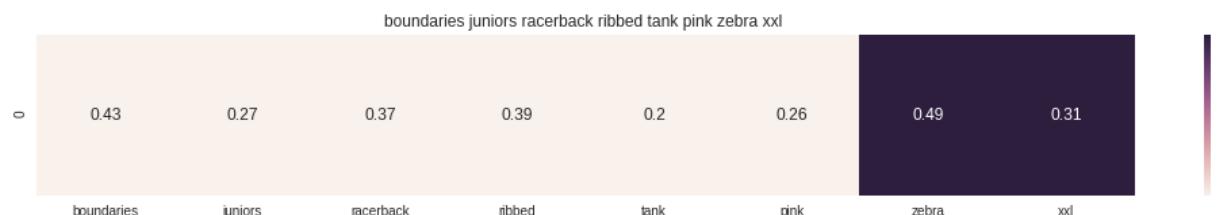
Eucliden distance from the given image : 1.031991846303421



ASIN : B06XSCVFT5

BRAND : Studio M

Eucliden distance from the given image : 1.2106843670424716



ASIN : B06Y2GTYPM

BRAND : No Boundaries

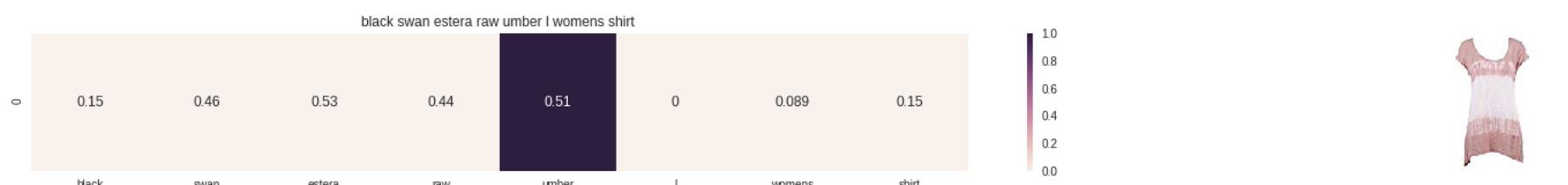
Eucliden distance from the given image : 1.212168381072083



ASIN : B012VQLT6Y

BRAND : KM T-shirt

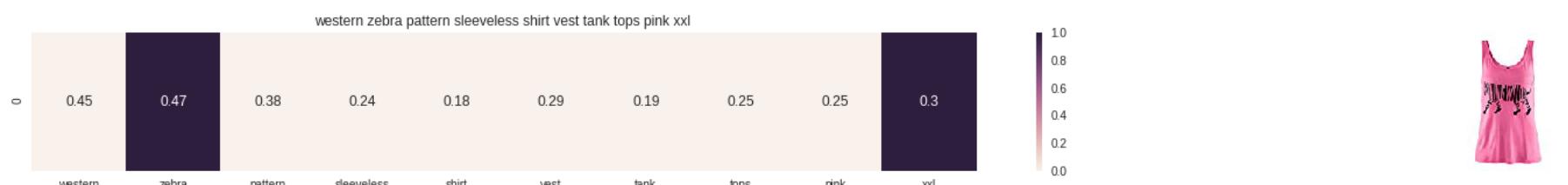
Eucliden distance from the given image : 1.219790640280982



ASIN : B06Y1VN8WQ

BRAND : Black Swan

Eucliden 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 : 1.2318451972624518



ASIN : B074T8ZYGX

BRAND : MKP Crop Top

Eucliden distance from the given image : 1.2340607457359425



ASIN : B071ZDF6T2

BRAND : Mossimo

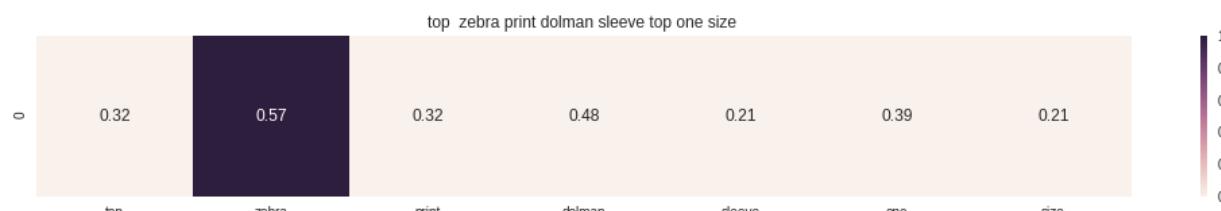
Eucliden distance from the given image : 1.2352785577664824



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 : 1.25354614208561



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_c
orpus
# idf_title_features[doc_id, index_of_word_in_corpus] = number of times the word occured 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 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 [32]: 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  $K(X, Y) = \langle X, Y \rangle / (\|X\| * \|Y\|)$ 
    # 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

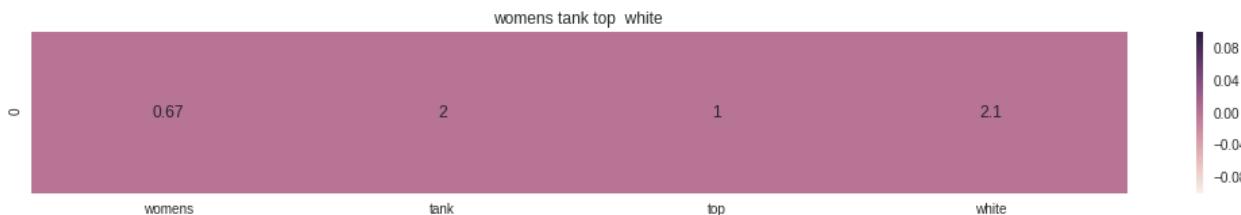
euclidean distance from the given image : 17.080812955631995



ASIN : B00KF2N5PU

Brand : Vietsbay

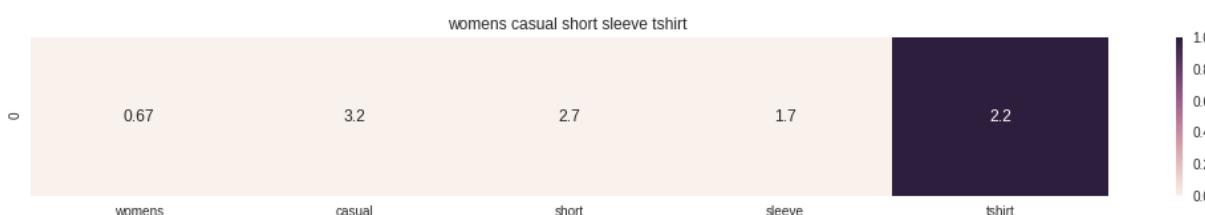
euclidean distance from the given image : 17.090168125645416



ASIN : B00JPOZ9GM

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

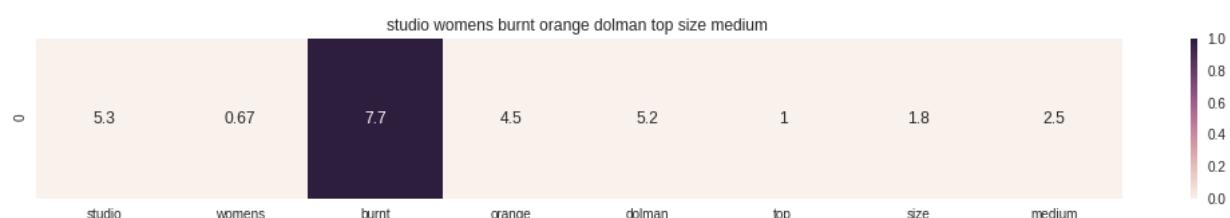
euclidean distance from the given image : 17.410075941001253



ASIN : B074G5G5RK

Brand : ERMANNO SCERVINO

euclidean distance from the given image : 17.539921335459557



ASIN : B06XSCVFT5

Brand : Studio M

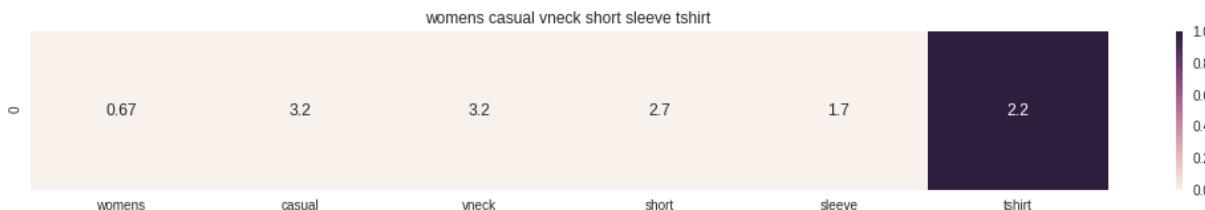
euclidean distance from the given image : 17.61275854366134



ASIN : B06Y6FH453

Brand : Who What Wear

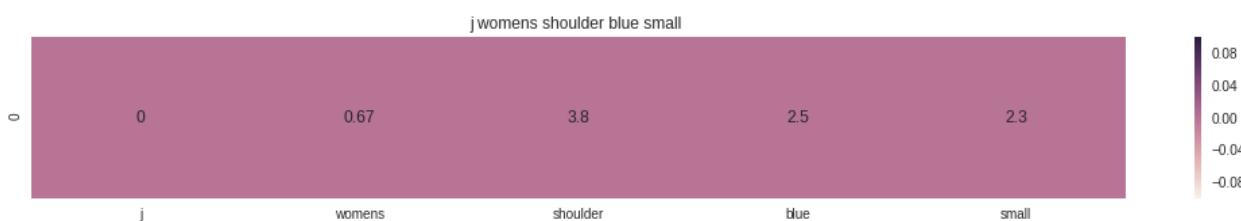
euclidean distance from the given image : 17.623745282500135



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

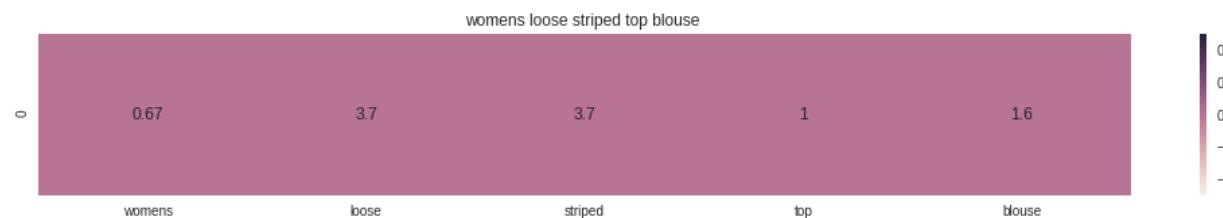
euclidean distance from the given image : 17.7230738913371



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 [0]: # 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)

...
```

```
In [34]: !pip install paramiko
```

```
Collecting paramiko
  Downloading https://files.pythonhosted.org/packages/cf/ae/94e70d49044ccc234bfdb20114fa947d7ba6eb68a2e452d89b9
20e62227/paramiko-2.4.2-py2.py3-none-any.whl (193kB)
    100% |██████████| 194kB 7.9MB/s
Collecting pynacl>=1.0.1 (from paramiko)
  Downloading https://files.pythonhosted.org/packages/27/15/2cd0a203f318c2240b42cd9dd13c931ddd61067809fee3479f44
f086103e/PyNaCl-1.3.0-cp34-abi3-manylinux1_x86_64.whl (759kB)
    100% |██████████| 768kB 25.1MB/s
Collecting bcrypt>=3.1.3 (from paramiko)
  Downloading https://files.pythonhosted.org/packages/d0/79/79a4d167a31cc206117d9b396926615fa9c1fdbd52017bcced80
937ac501/bcrypt-3.1.6-cp34-abi3-manylinux1_x86_64.whl (55kB)
    100% |██████████| 61kB 24.2MB/s
Requirement already satisfied: pyasn1>=0.1.7 in /usr/local/lib/python3.6/dist-packages (from paramiko) (0.4.5)
Collecting cryptography>=1.5 (from paramiko)
  Downloading https://files.pythonhosted.org/packages/5b/12/b0409a94dad366d98a8eee2a77678c7a73aaaf8c0e4b835abea6
34ea3896/cryptography-2.6.1-cp34-abi3-manylinux1_x86_64.whl (2.3MB)
    100% |██████████| 2.3MB 12.8MB/s
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from pynacl>=1.0.1->paramiko) (1.1
1.0)
Requirement already satisfied: cffi>=1.4.1 in /usr/local/lib/python3.6/dist-packages (from pynacl>=1.0.1->parami
ko) (1.12.2)
Collecting asn1crypto>=0.21.0 (from cryptography>=1.5->paramiko)
  Downloading https://files.pythonhosted.org/packages/ea/cd/35485615f45f30a510576f1a56d1e0a7ad7bd8ab5ed7cdc600ef
7cd06222/asn1crypto-0.24.0-py2.py3-none-any.whl (101kB)
    100% |██████████| 102kB 29.6MB/s
Requirement already satisfied: pycparser in /usr/local/lib/python3.6/dist-packages (from cffi>=1.4.1->pynacl>=1.
0.1->paramiko) (2.19)
Installing collected packages: pynacl, bcrypt, asn1crypto, cryptography, paramiko
Successfully installed asn1crypto-0.24.0 bcrypt-3.1.6 cryptography-2.6.1 paramiko-2.4.2 pynacl-1.3.0
```

```
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('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 courpus 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 sentance
    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 caluclate 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 correspo
    nds 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 correspo
    nds 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])
    # ploting 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):
    # sentence: its title of the apparel
    # num_features: the length 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 initialize a vector of size 300 with all zeros
    # we add each word2vec(wordi) to this featureVec
    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 sentence, 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 [39]: def avg_w2v_model(doc_id, num_results):
    # doc_id: apparel's id in given corpus

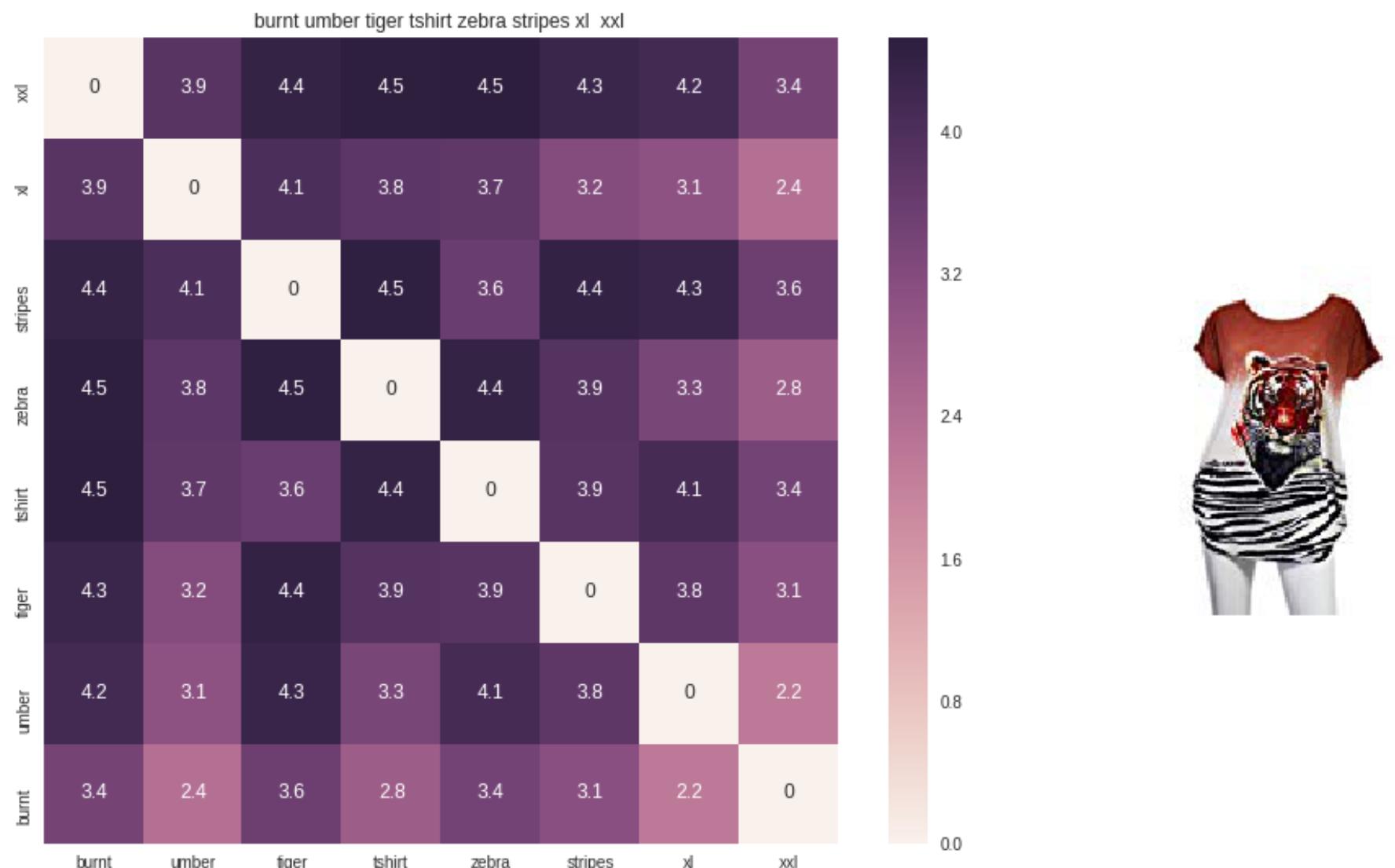
    # dist(x, y) = sqrt(dot(x, x) - 2 * dot(x, y) + dot(y, y))
    pairwise_dist = pairwise_distances(w2v_title, w2v_title[doc_id].reshape(1,-1))

    # 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)

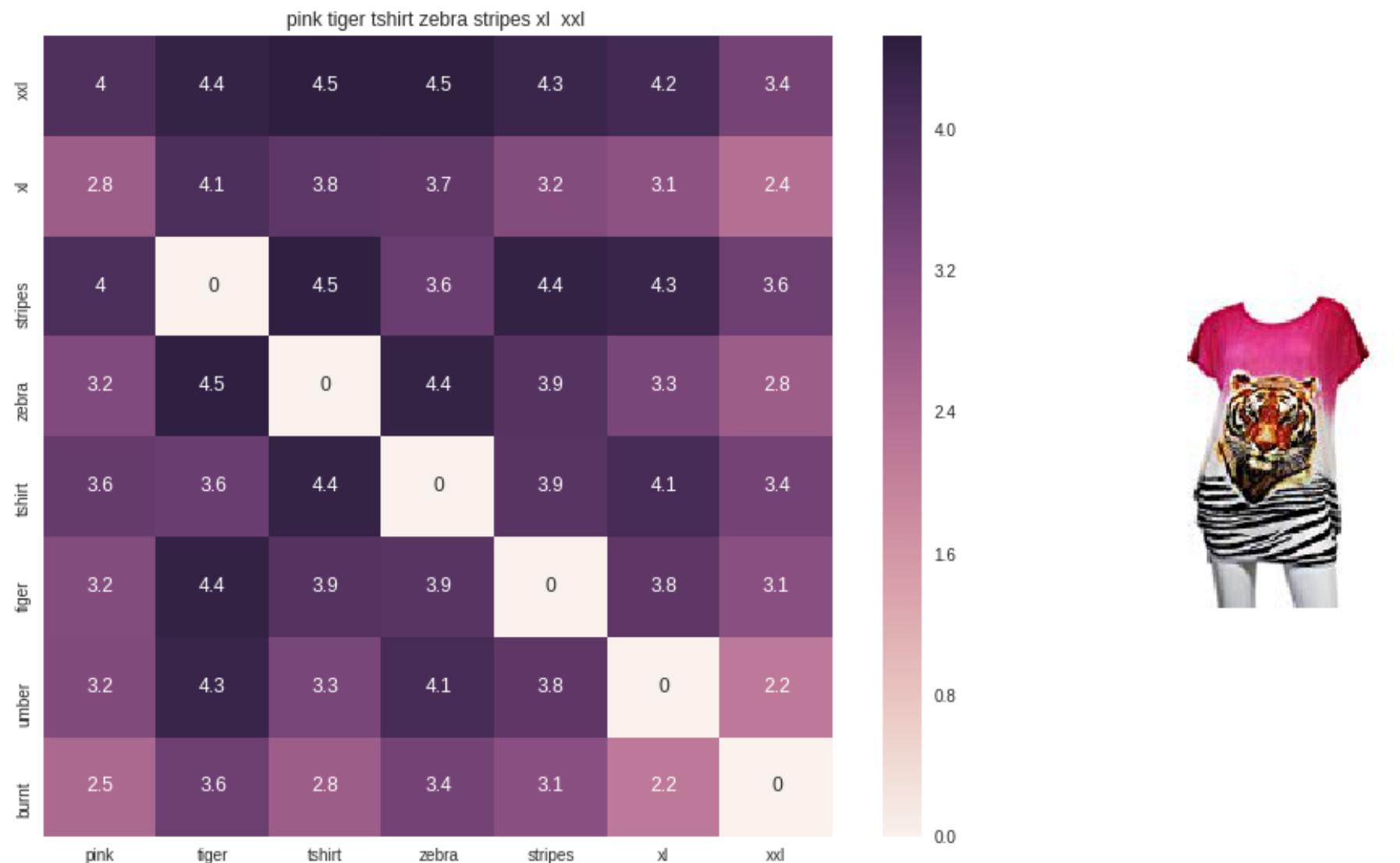
avg_w2v_model(12566, 20)
# in the give heat map, each cell contains the euclidean distance between words i, j
```



ASIN : B00JXQB5FQ

BRAND : Si Row

euclidean distance from given input image : 0.0



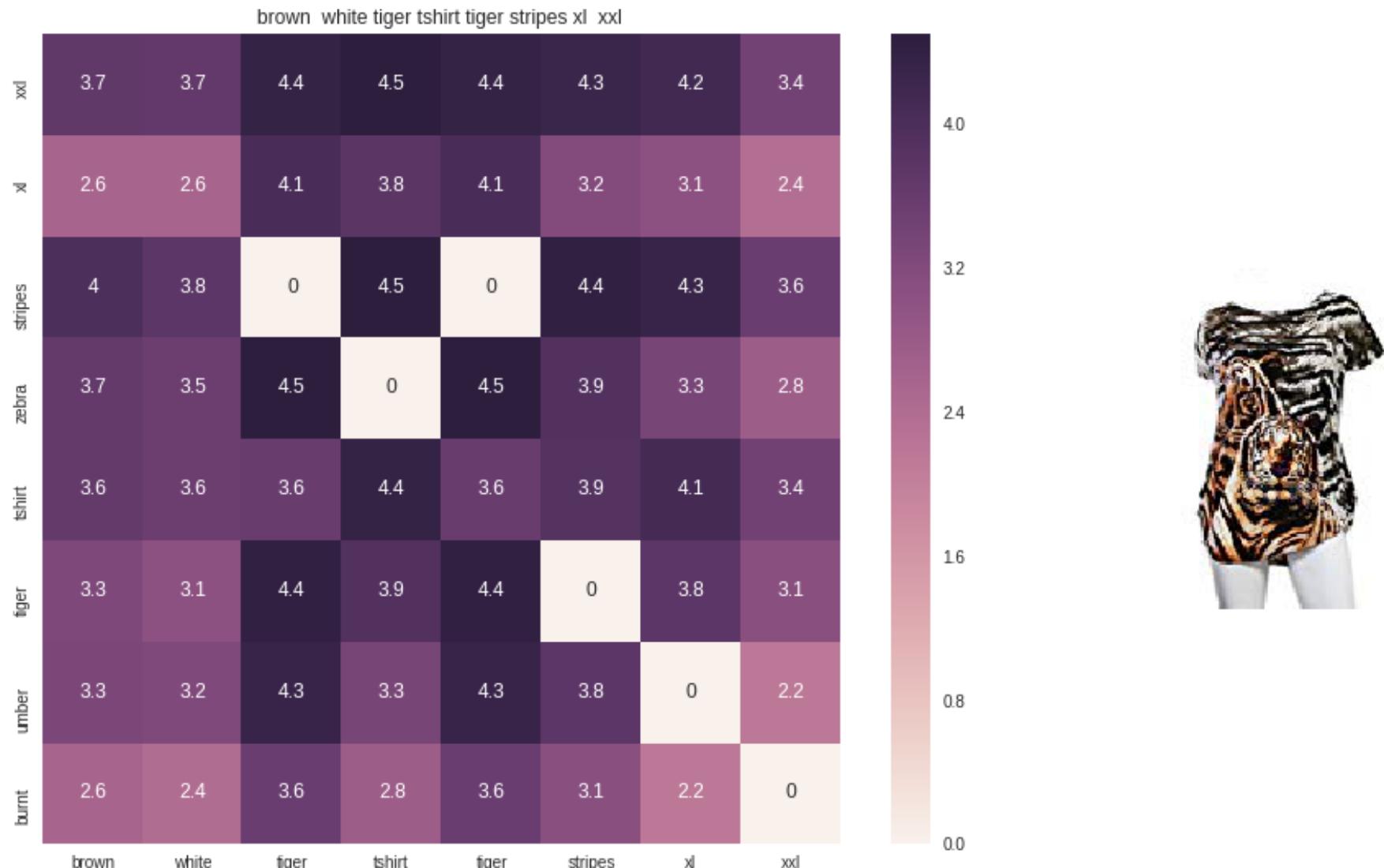
ASIN : B00JXQASS6

BRAND : Si Row

euclidean distance from given input image : 0.5891928

=====

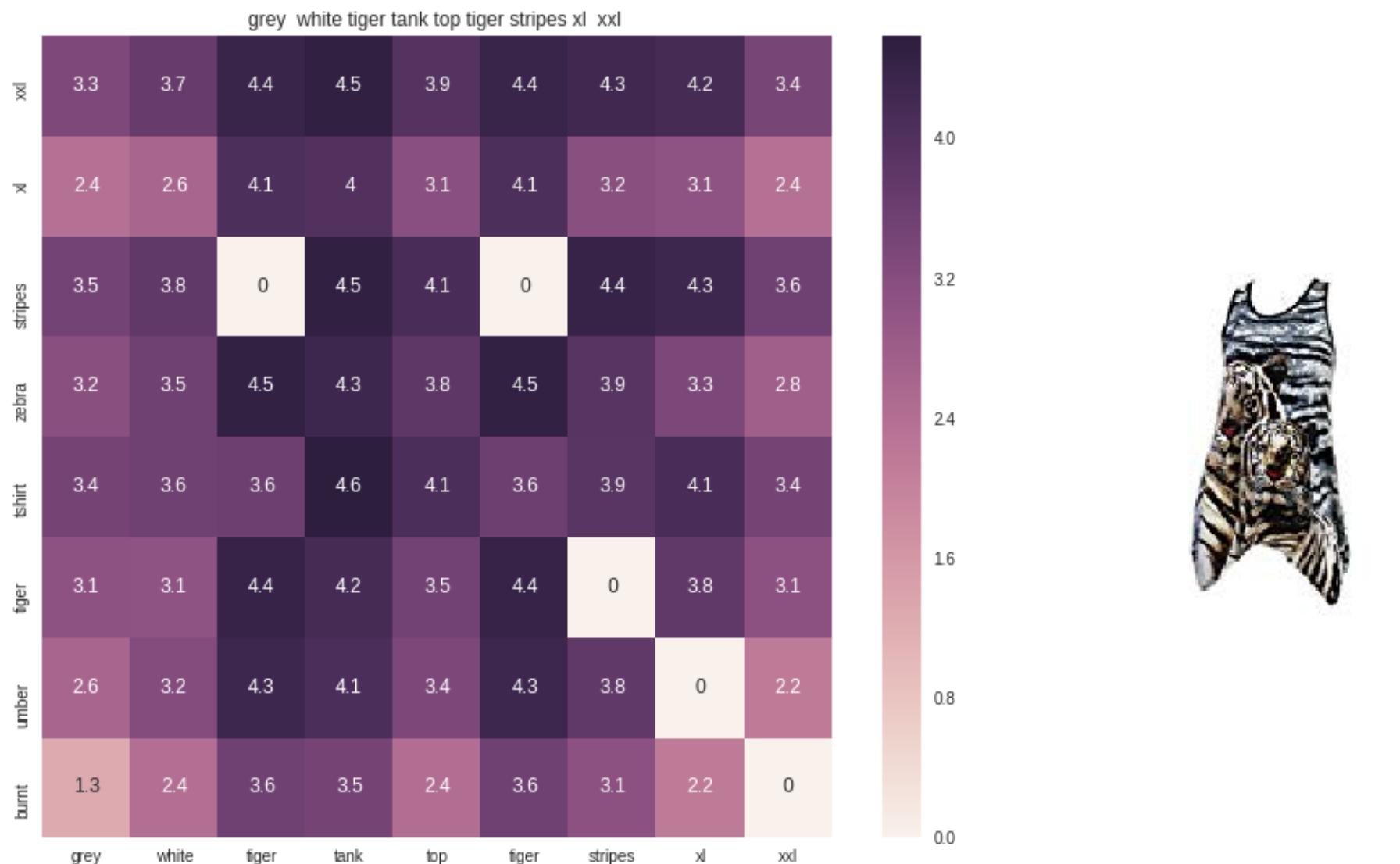
=====



ASIN : B00JXQCWT0

BRAND : Si Row

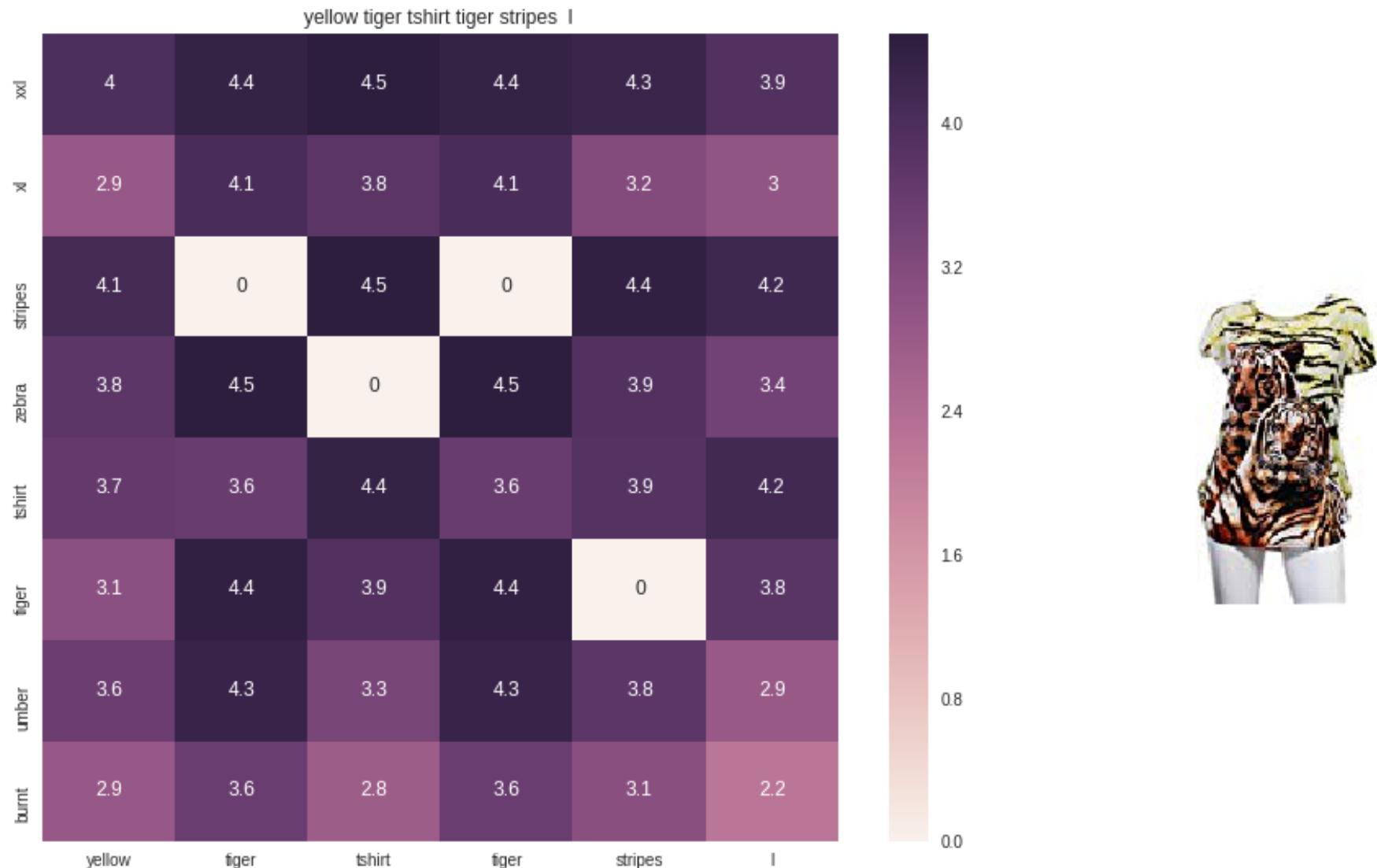
euclidean distance from given input image : 0.7003436



ASIN : B00JXQAFZ2

BRAND : Si Row

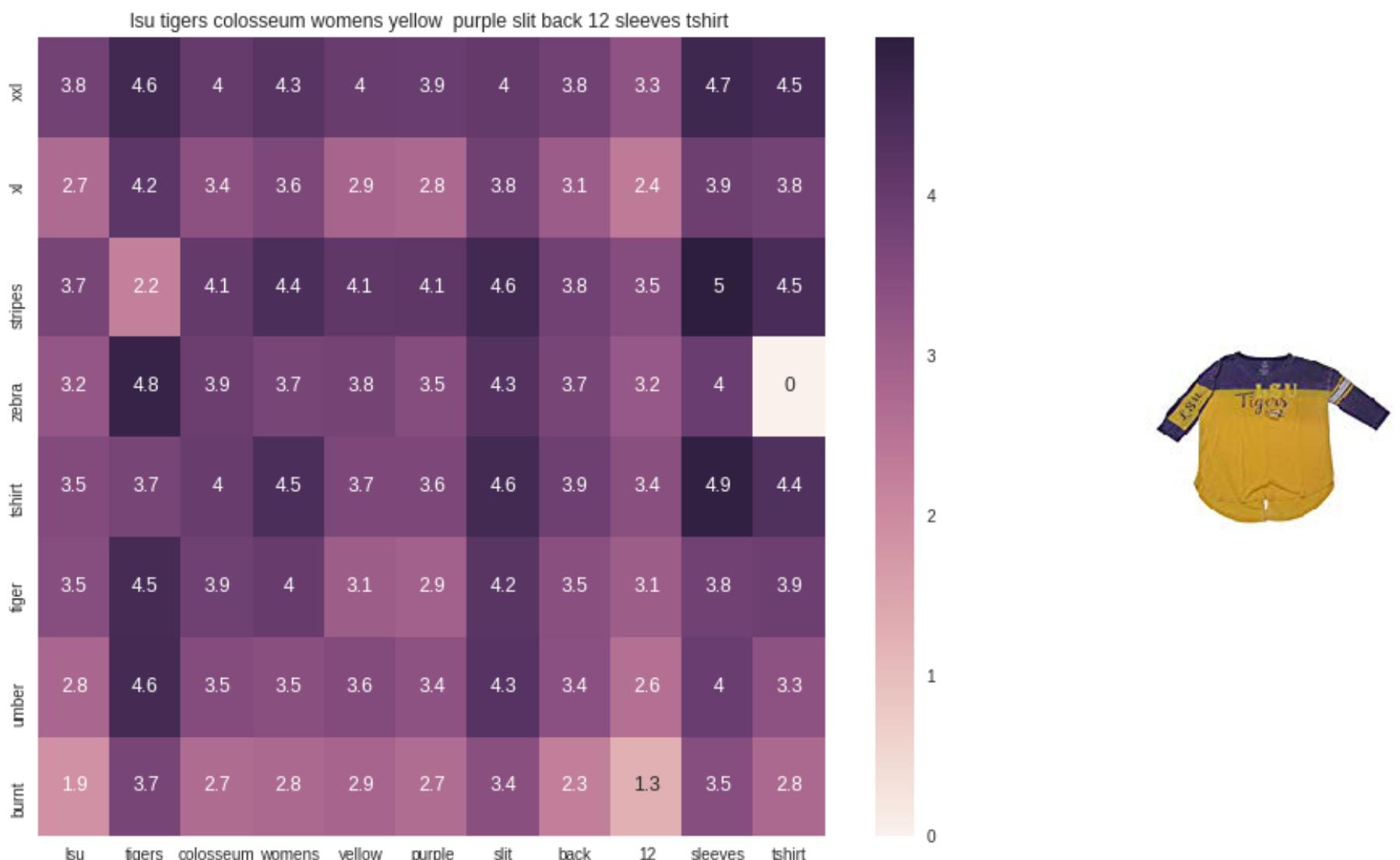
euclidean distance from given input image : 0.8928398



ASIN : B00JXQCUIC

BRAND : Si Row

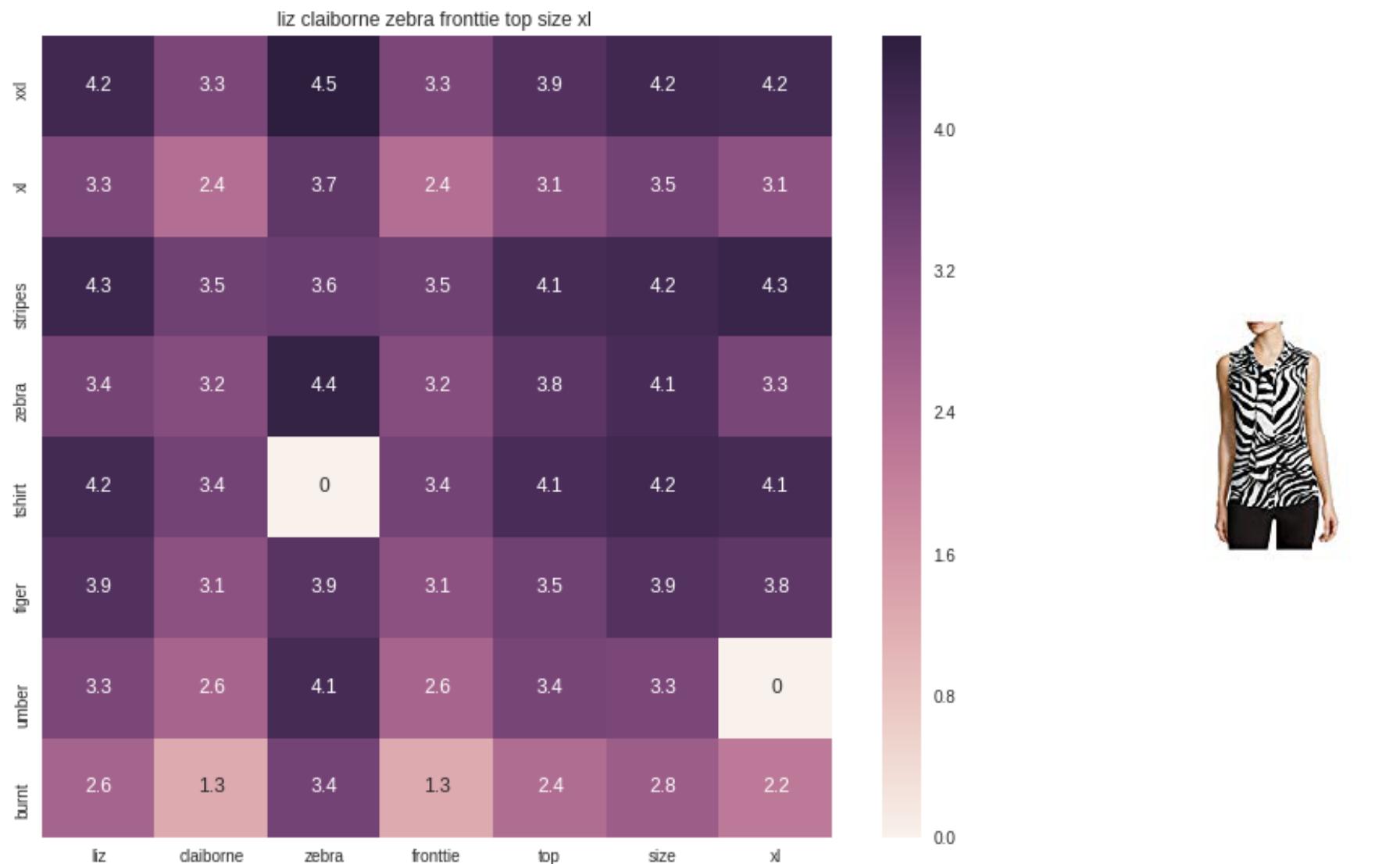
euclidean distance from given input image : 0.9560129



ASIN : B073R5Q8HD

BRAND : Colosseum

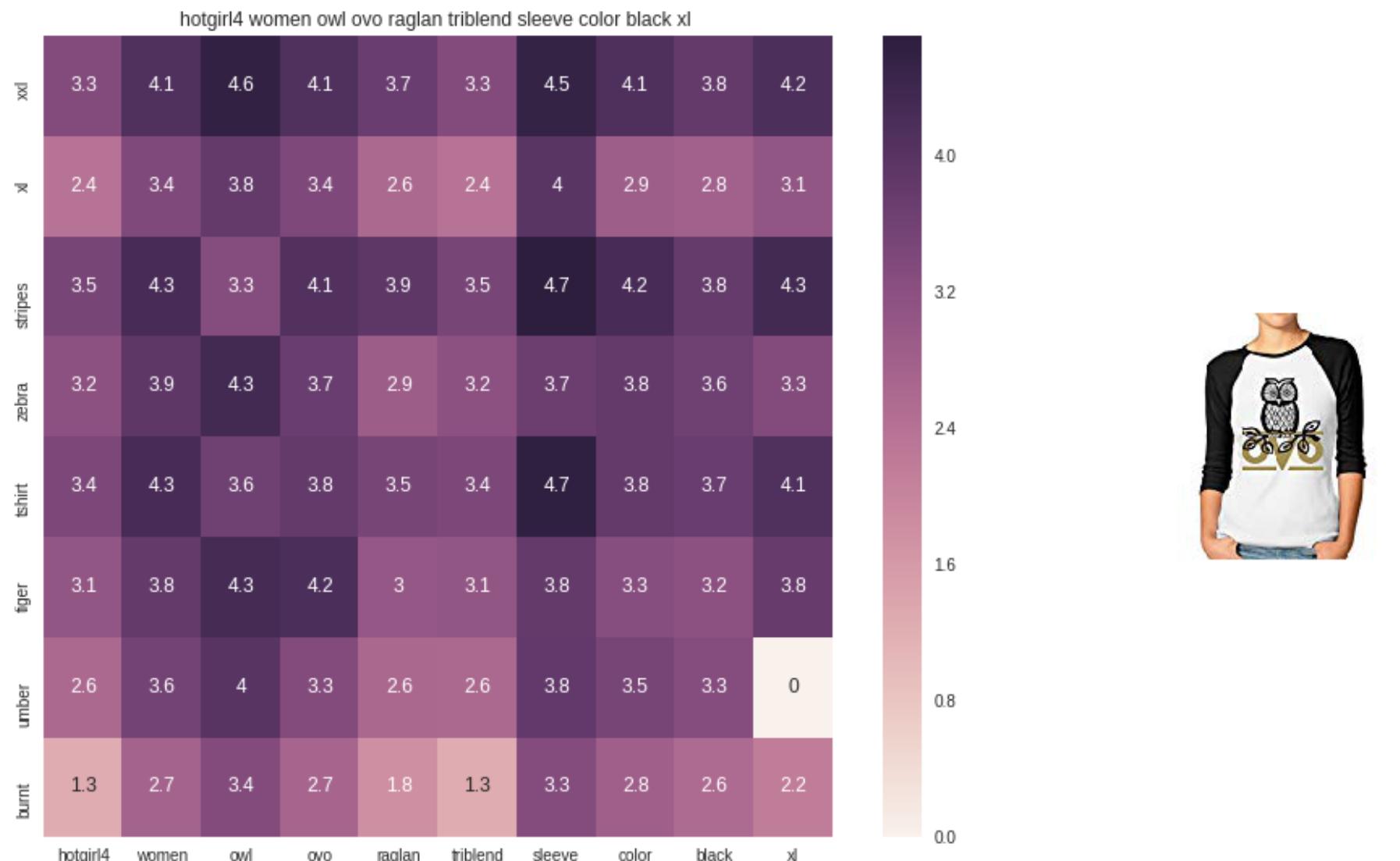
euclidean distance from given input image : 1.0229691



ASIN : B06XBY5QXL

BRAND : Liz Claiborne

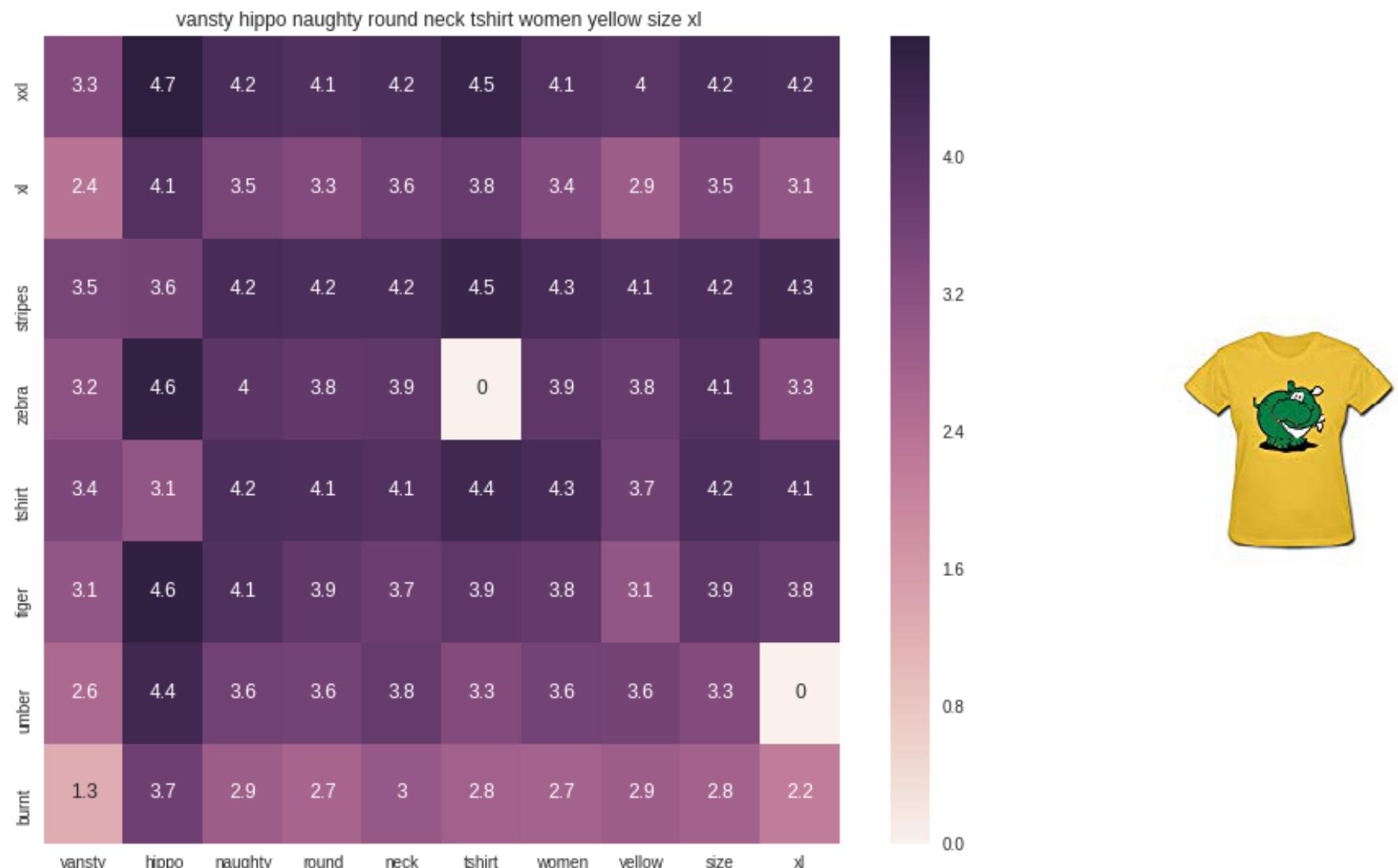
euclidean distance from given input image : 1.0669324



ASIN : B01L8L73M2

BRAND : Hotgirl4 Raglan Design

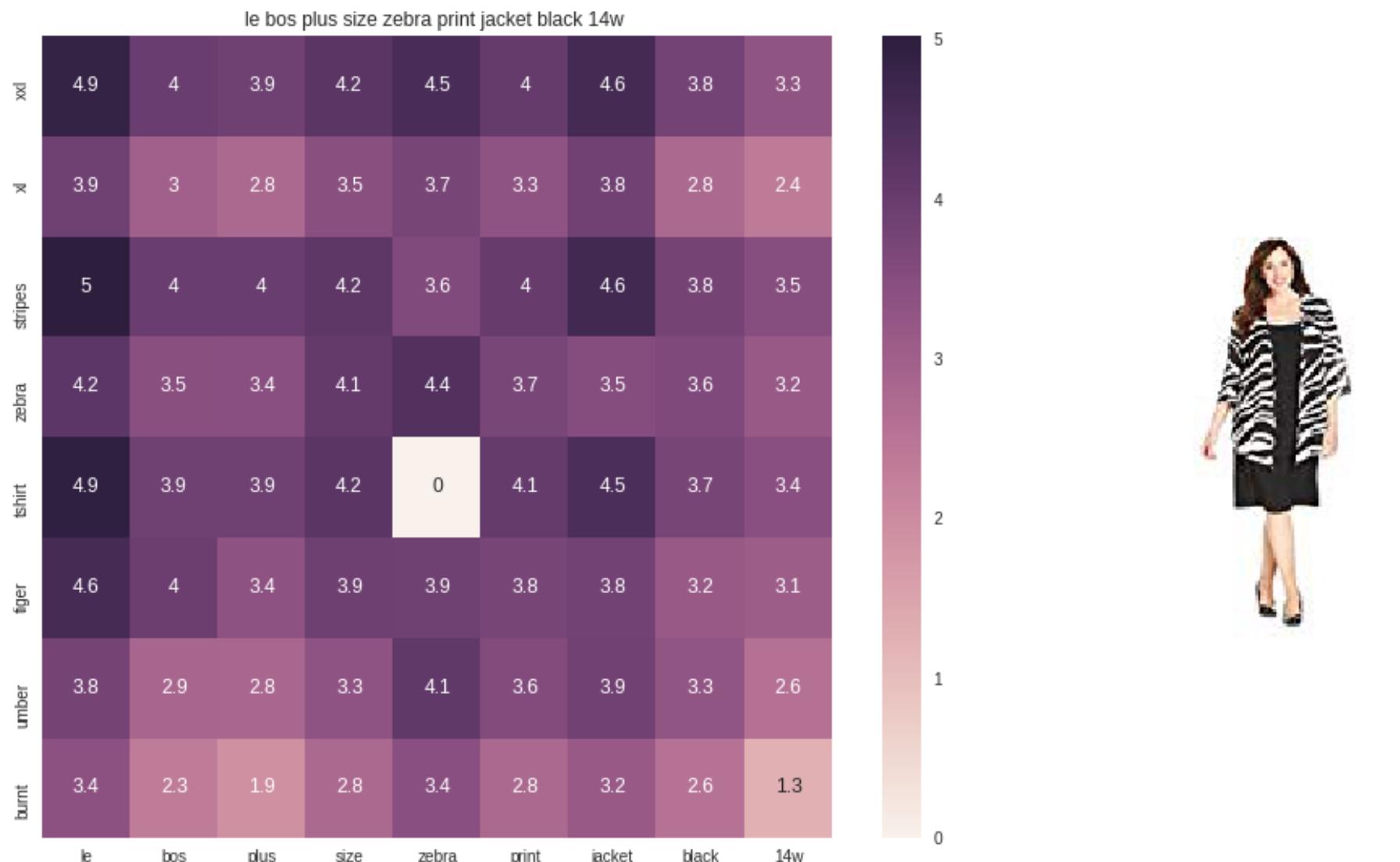
euclidean distance from given input image : 1.0731406



ASIN : B01EJS5H06

BRAND : Vansty

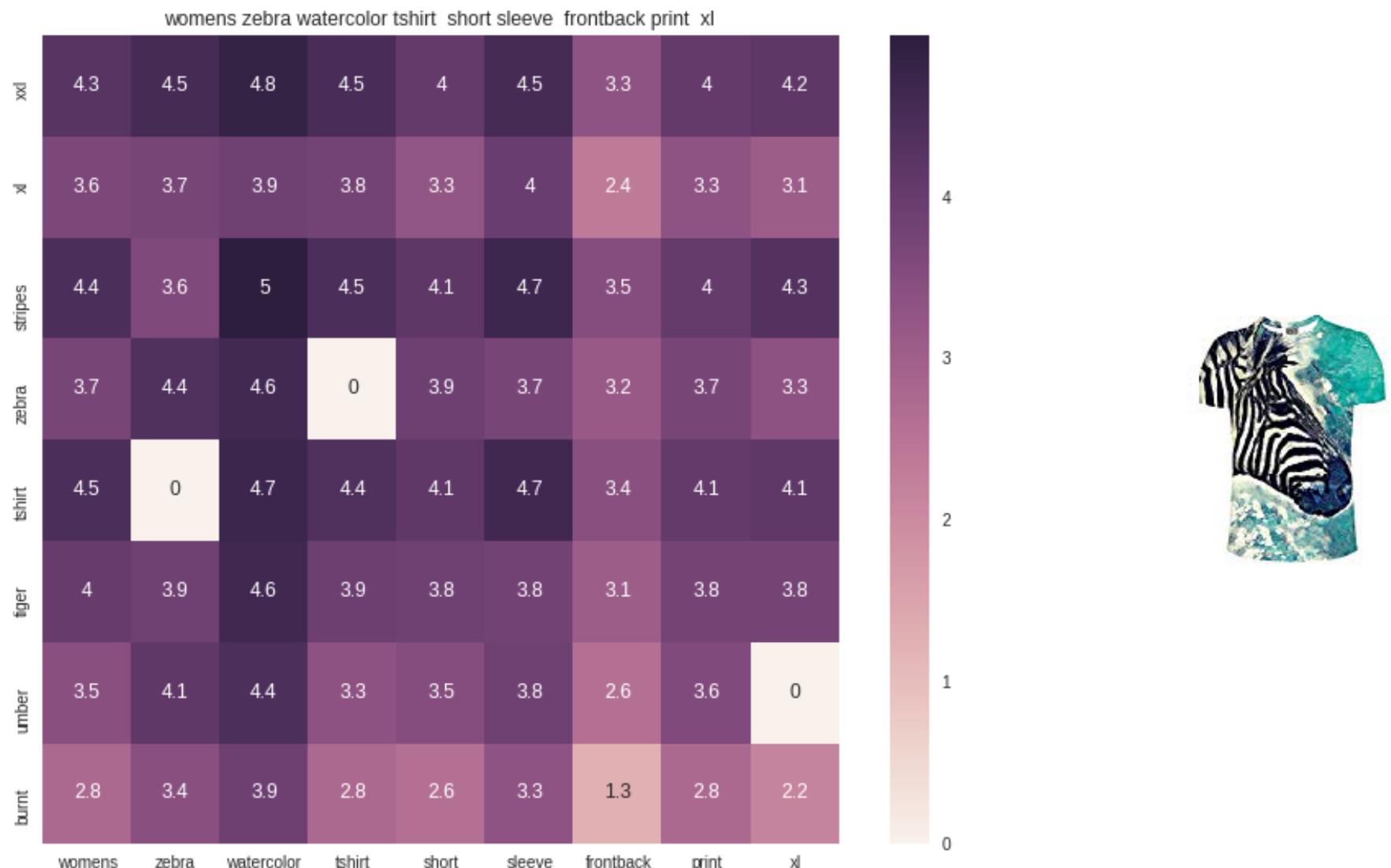
euclidean distance from given input image : 1.0757191



ASIN : B01B01XRK8

BRAND : Le Bos

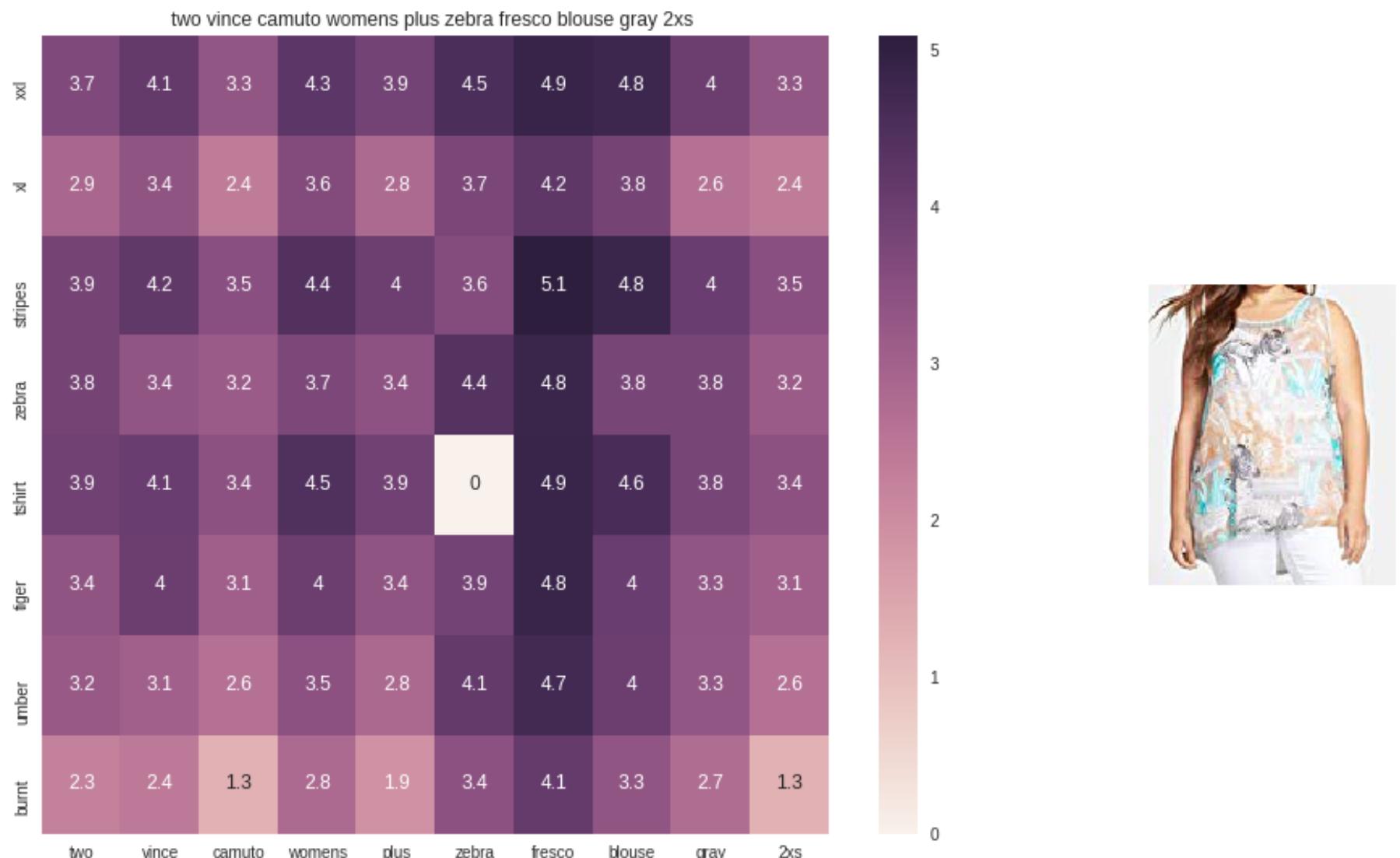
euclidean distance from given input image : 1.0839965



ASIN : B072R2JXKW

BRAND : WHAT ON EARTH

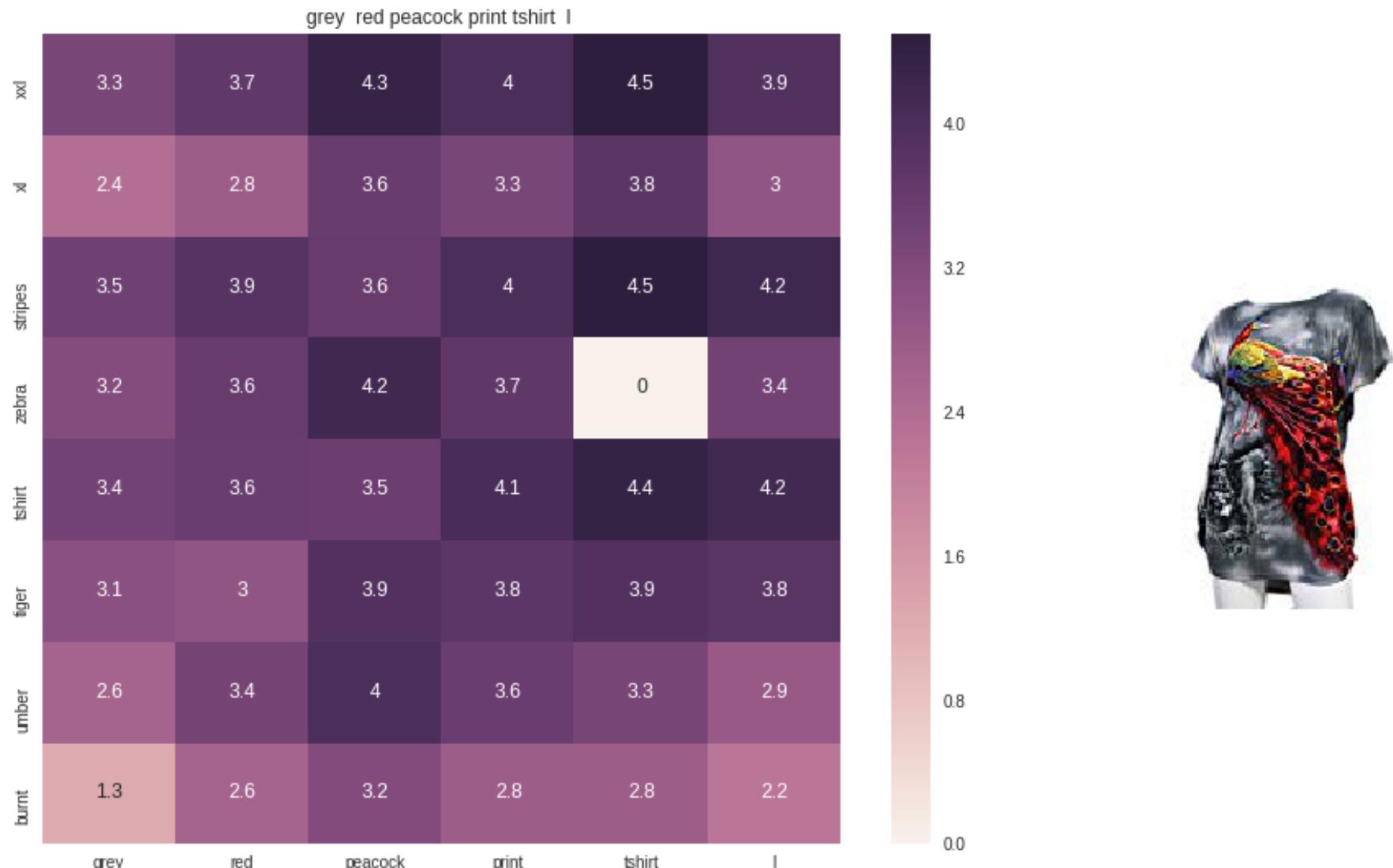
euclidean distance from given input image : 1.0842218



ASIN : B074MJRGW6

BRAND : Two by Vince Camuto

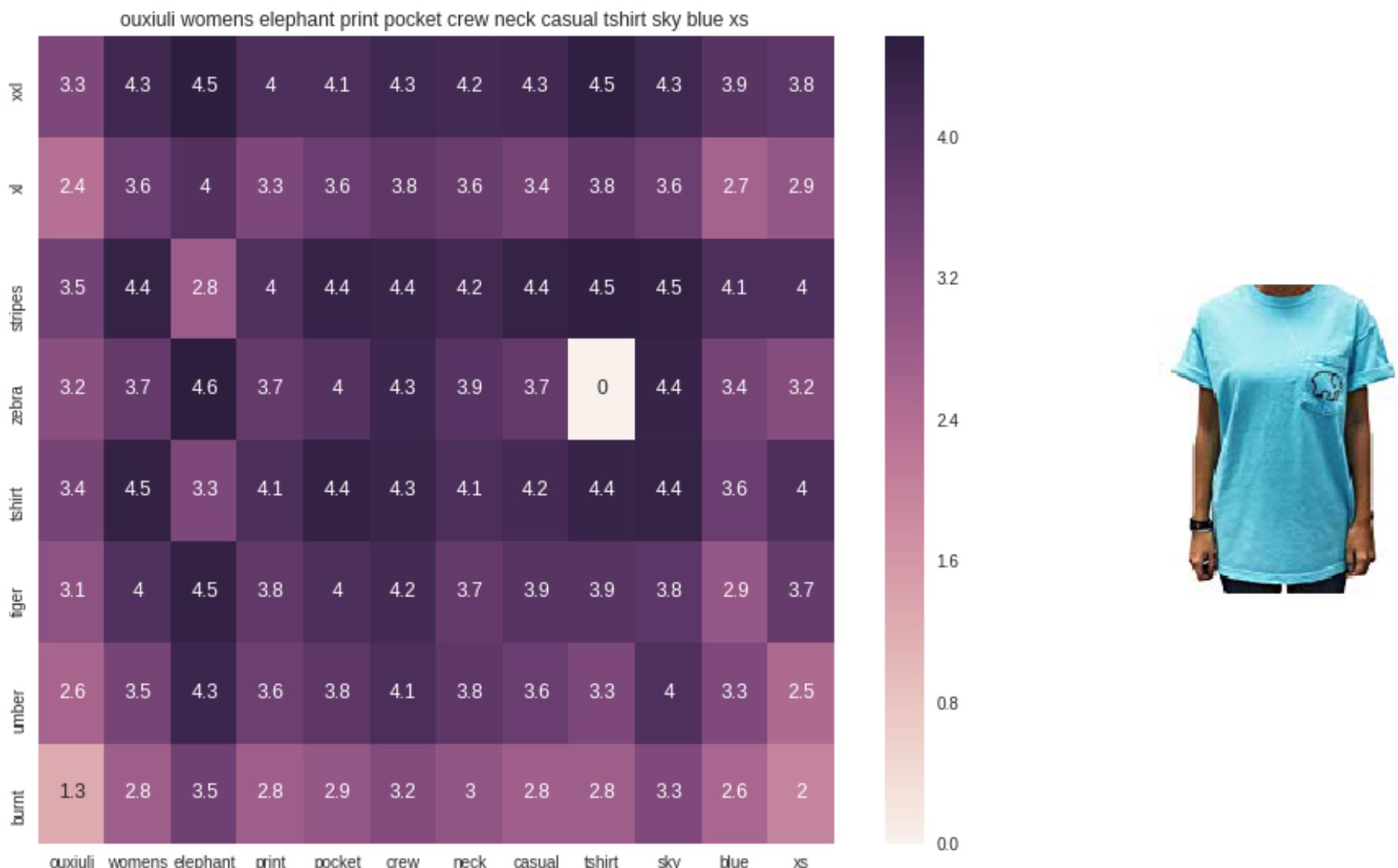
euclidean distance from given input image : 1.0895039



ASIN : B00JXQCFRS

BRAND : Si Row

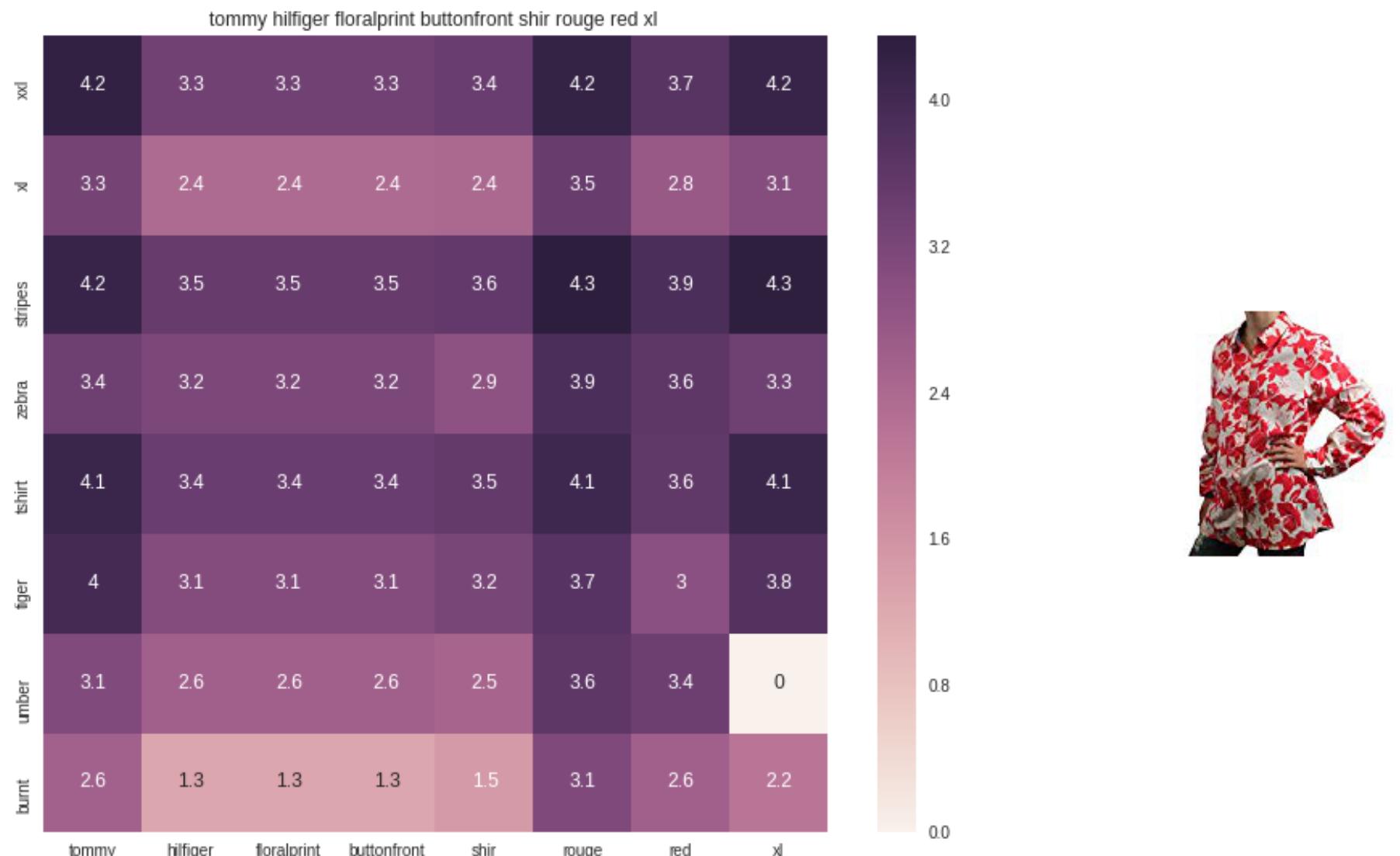
euclidean distance from given input image : 1.0900588



ASIN : B01I53HU6K

BRAND : ouxiuli

euclidean distance from given input image : 1.0920112



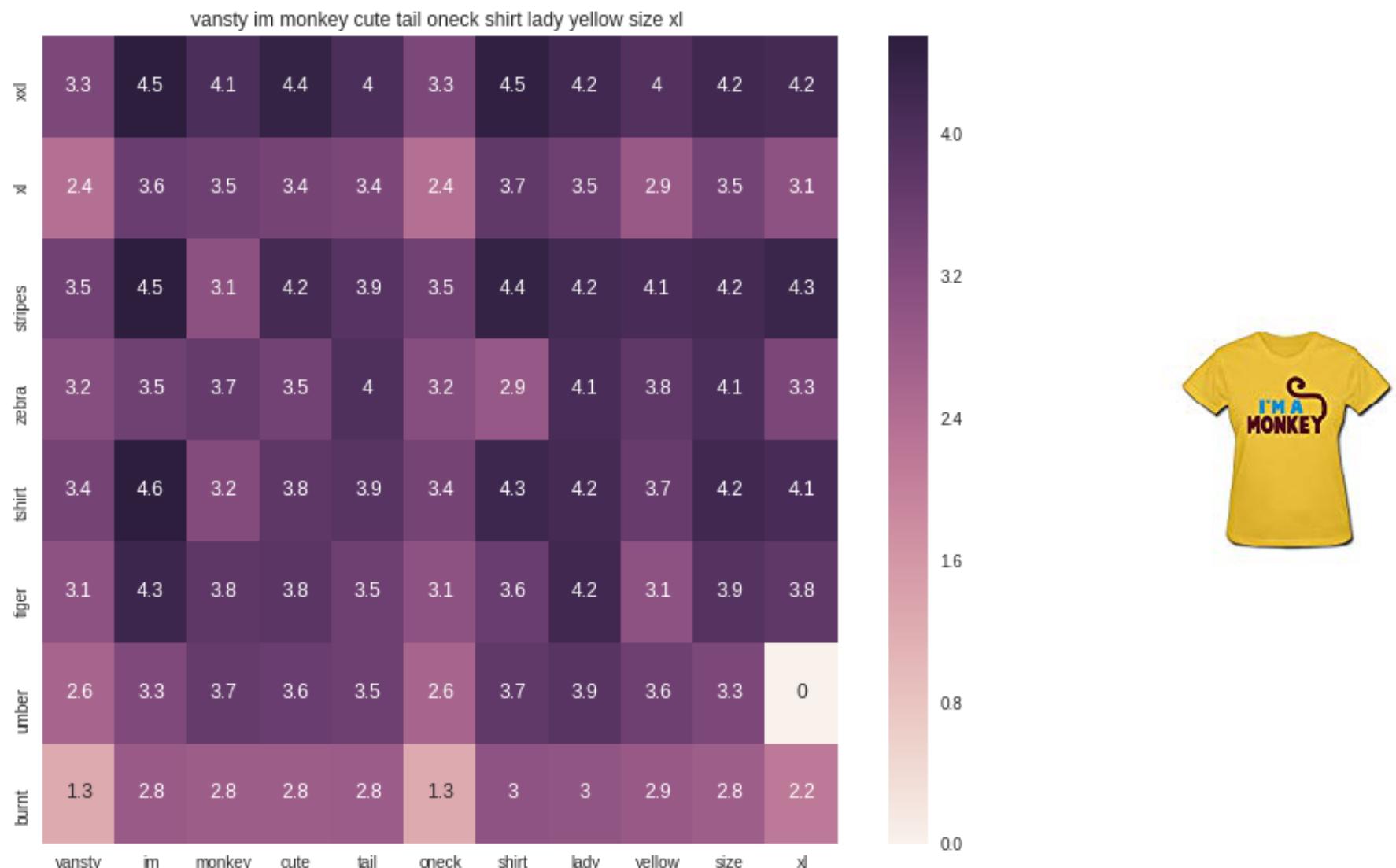
ASIN : B0711NGTQM

BRAND : THILFIGER RTW

euclidean distance from given input image : 1.0923418

=====

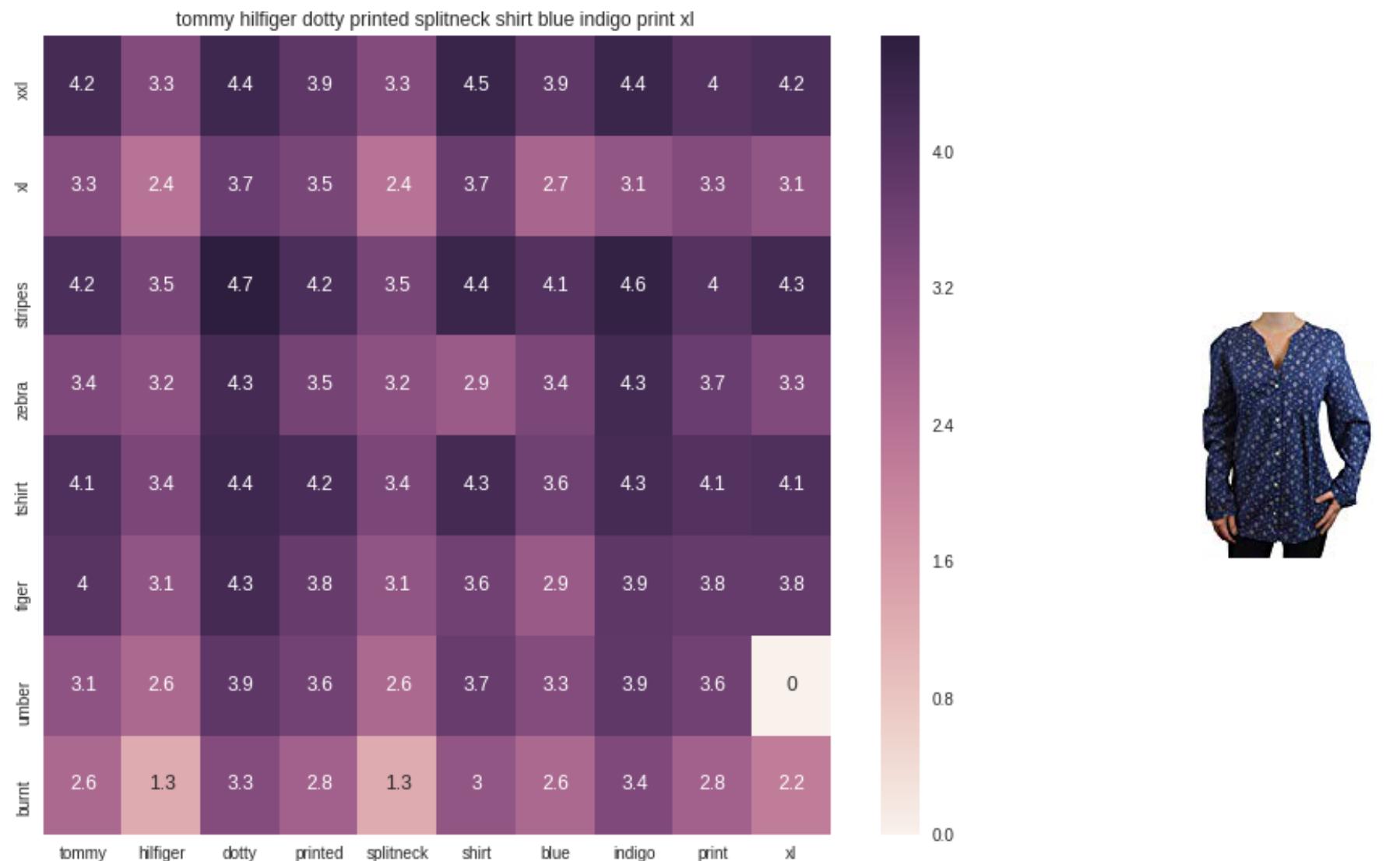
=====



ASIN : B01EFSL08Y

BRAND : Vansty

euclidean distance from given input image : 1.0934006



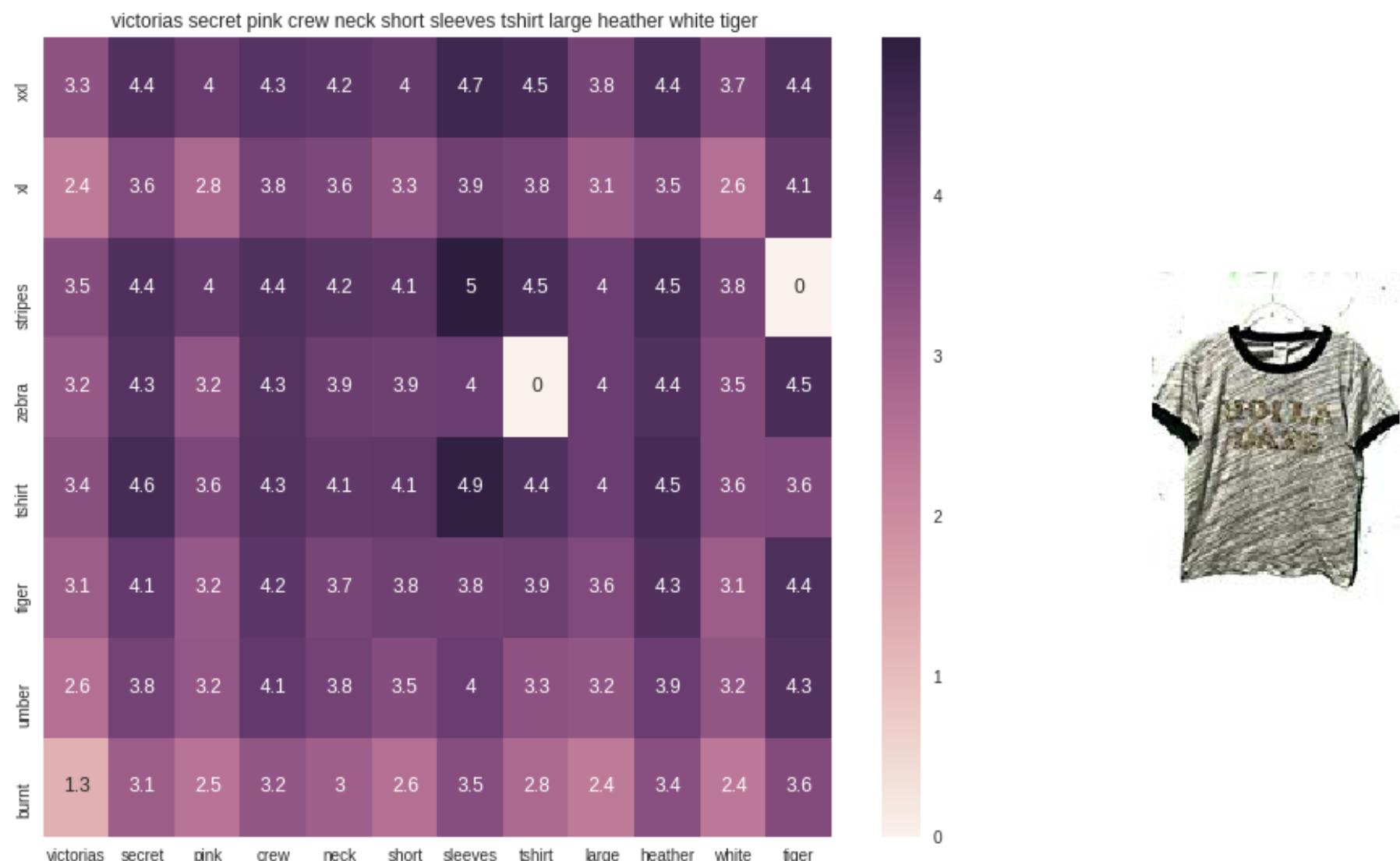
ASIN : B0716TVWQ4

BRAND : THILFIGER RTW

euclidean distance from given input image : 1.0942025

=====

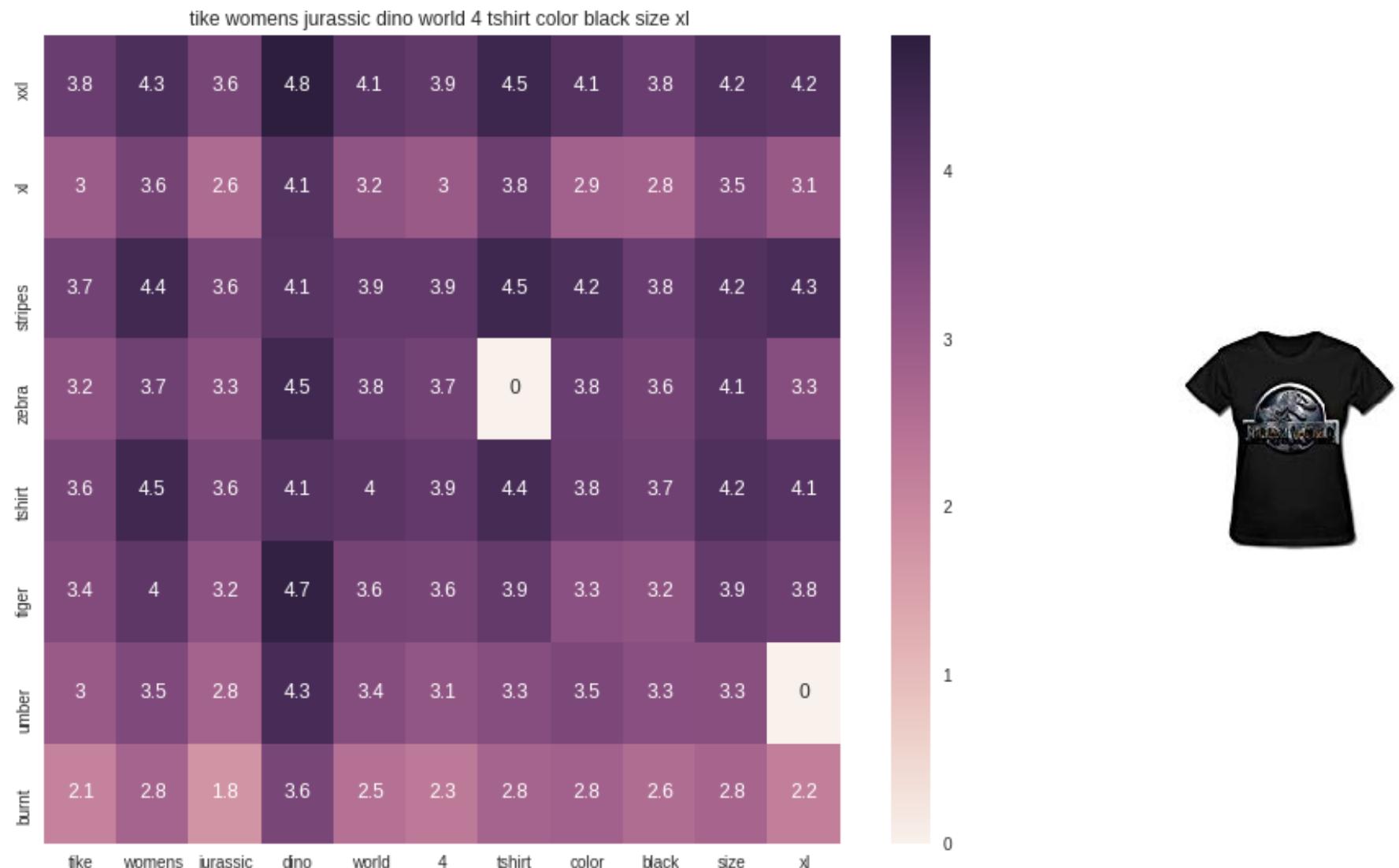
=====



ASIN : B0716MVPGV

BRAND : V.Secret

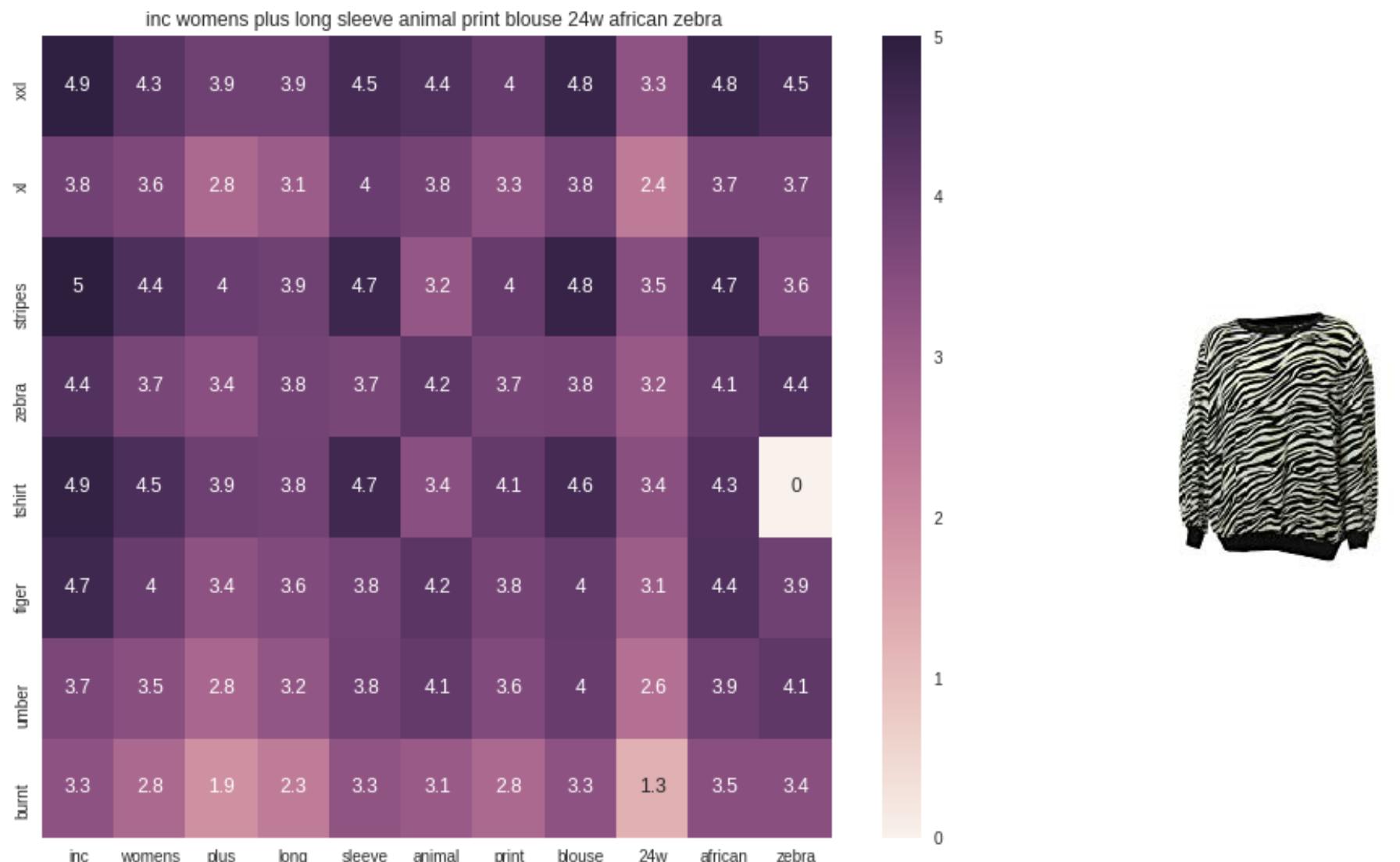
euclidean distance from given input image : 1.0948305



ASIN : B0160PN40I

BRAND : TIKE Fashions

euclidean distance from given input image : 1.0951276



ASIN : B018WDJCUA

BRAND : INC - International Concepts Woman

euclidean distance from given input image : 1.0966893

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

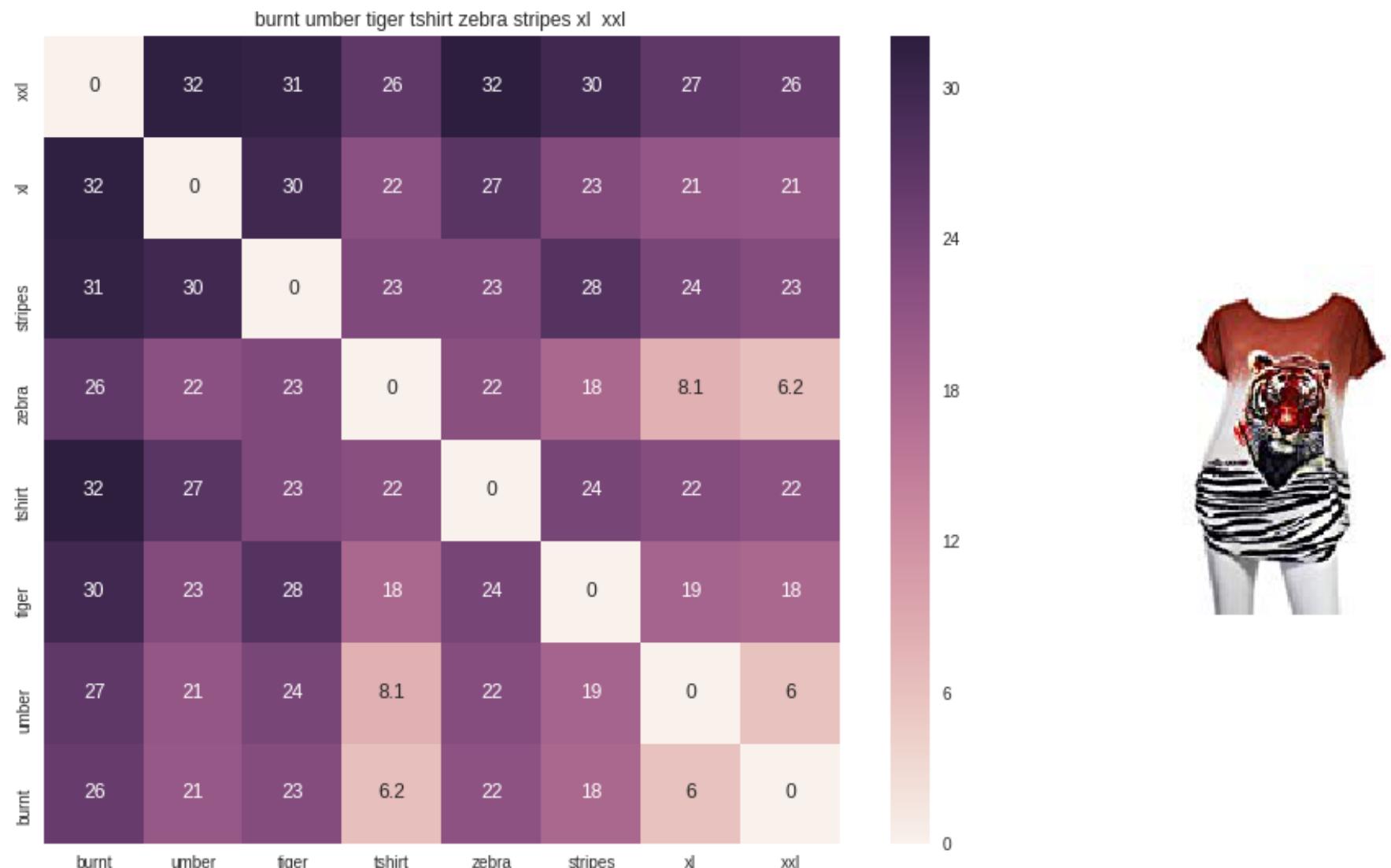
    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as  $K(X, Y) = \langle X, Y \rangle / (\|X\| * \|Y\|)$ 
    # http://scikit-Learn.org/stable/modules/metrics.html#cosine-similarity
    pairwise_dist = 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)

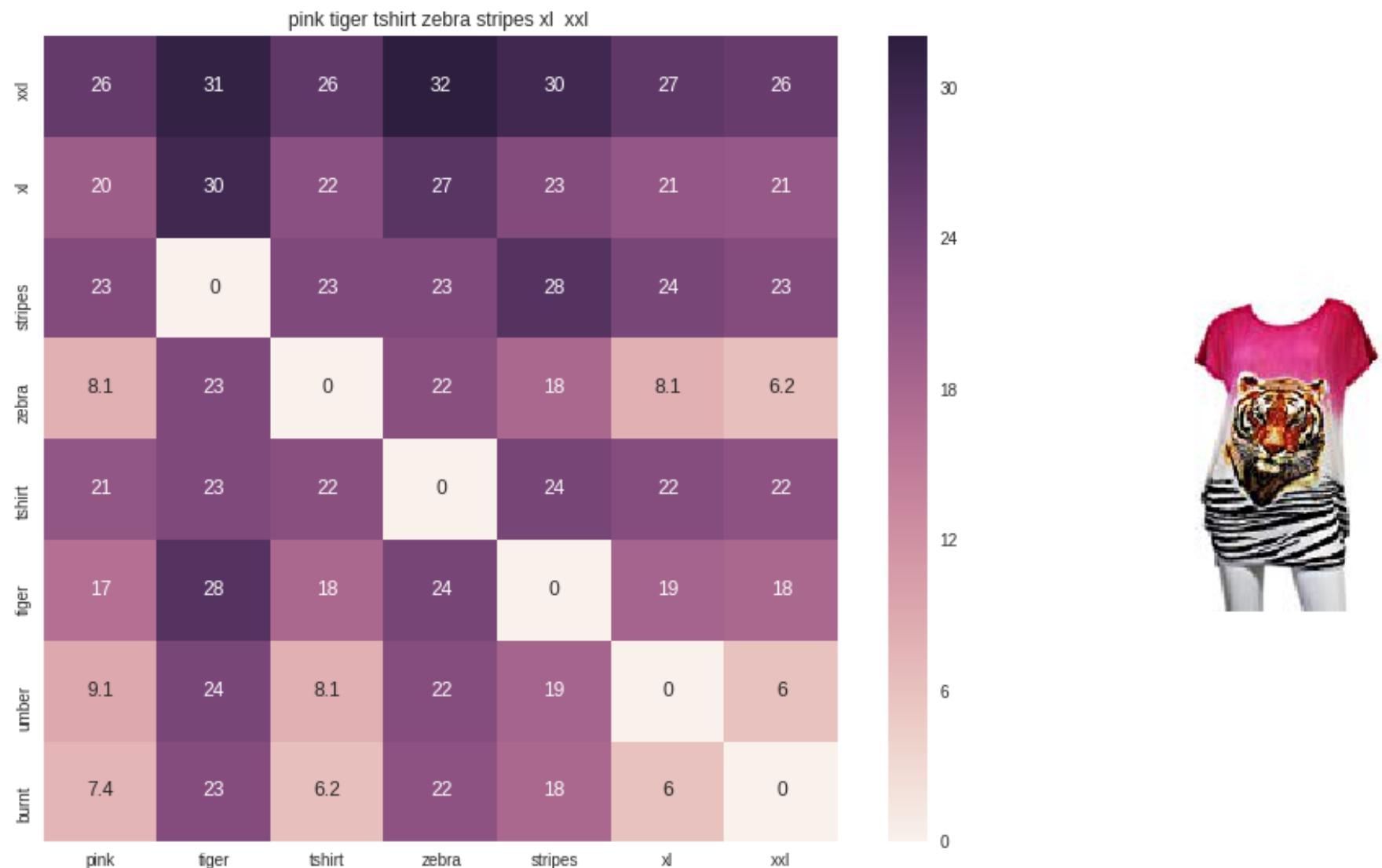
weighted_w2v_model(12566, 20)
#931
#12566
# in the give heat map, each cell contains the euclidean distance between words i, j
```



ASIN : B00JXQB5FQ

Brand : Si Row

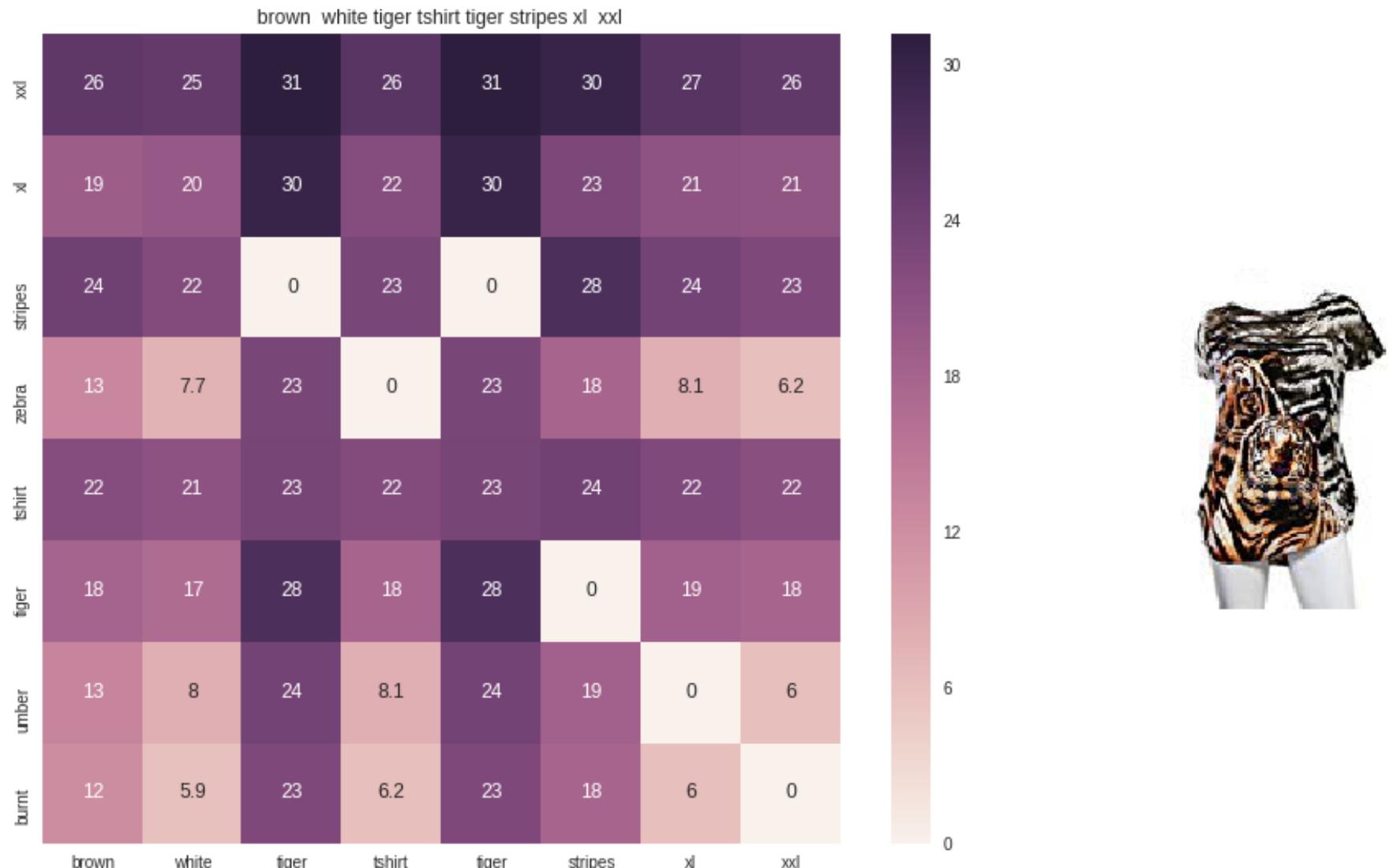
euclidean distance from input : 0.00390625



ASIN : B00JXQASS6

Brand : Si Row

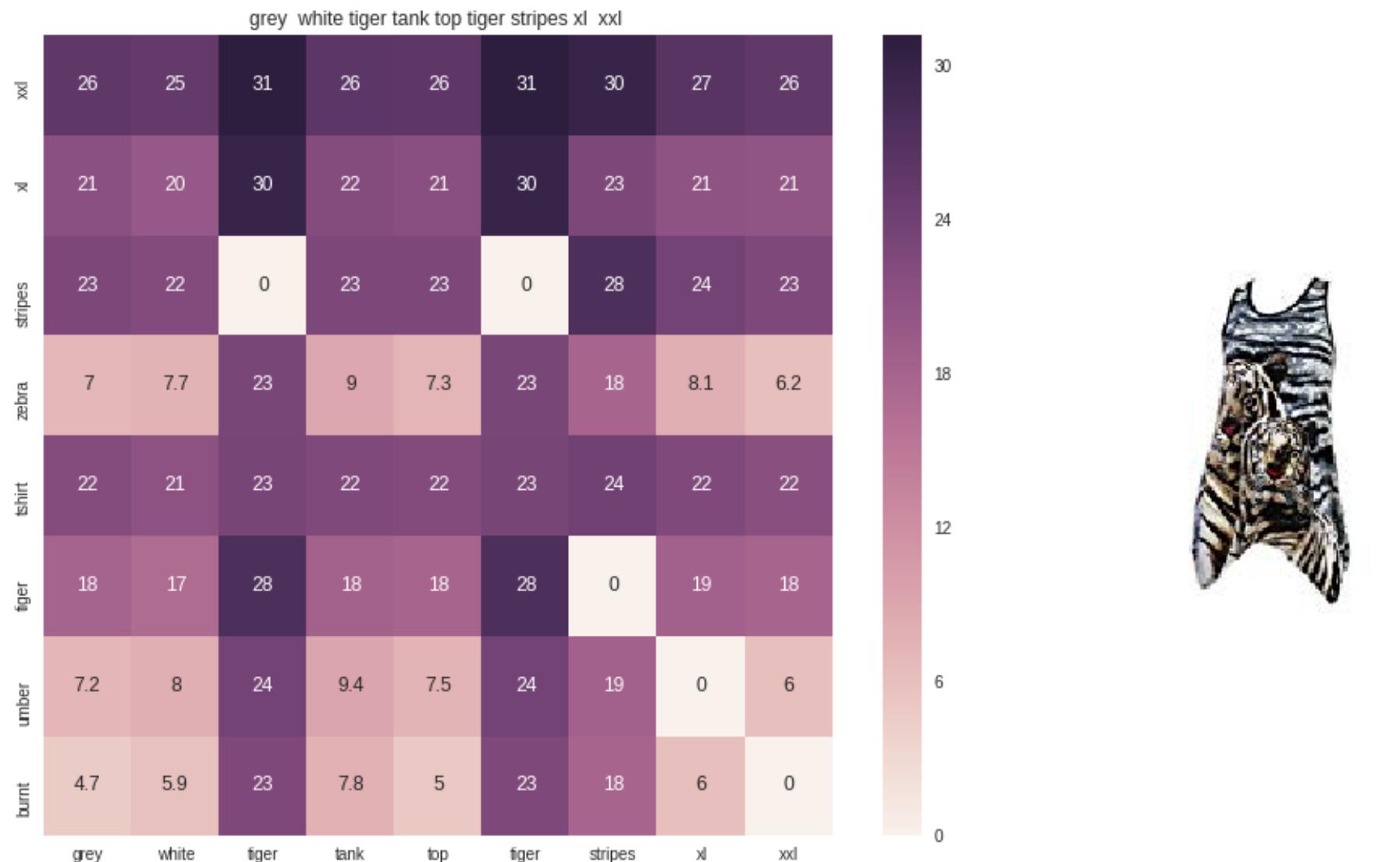
euclidean distance from input : 4.0638876



ASIN : B00JXQCWT0

Brand : Si Row

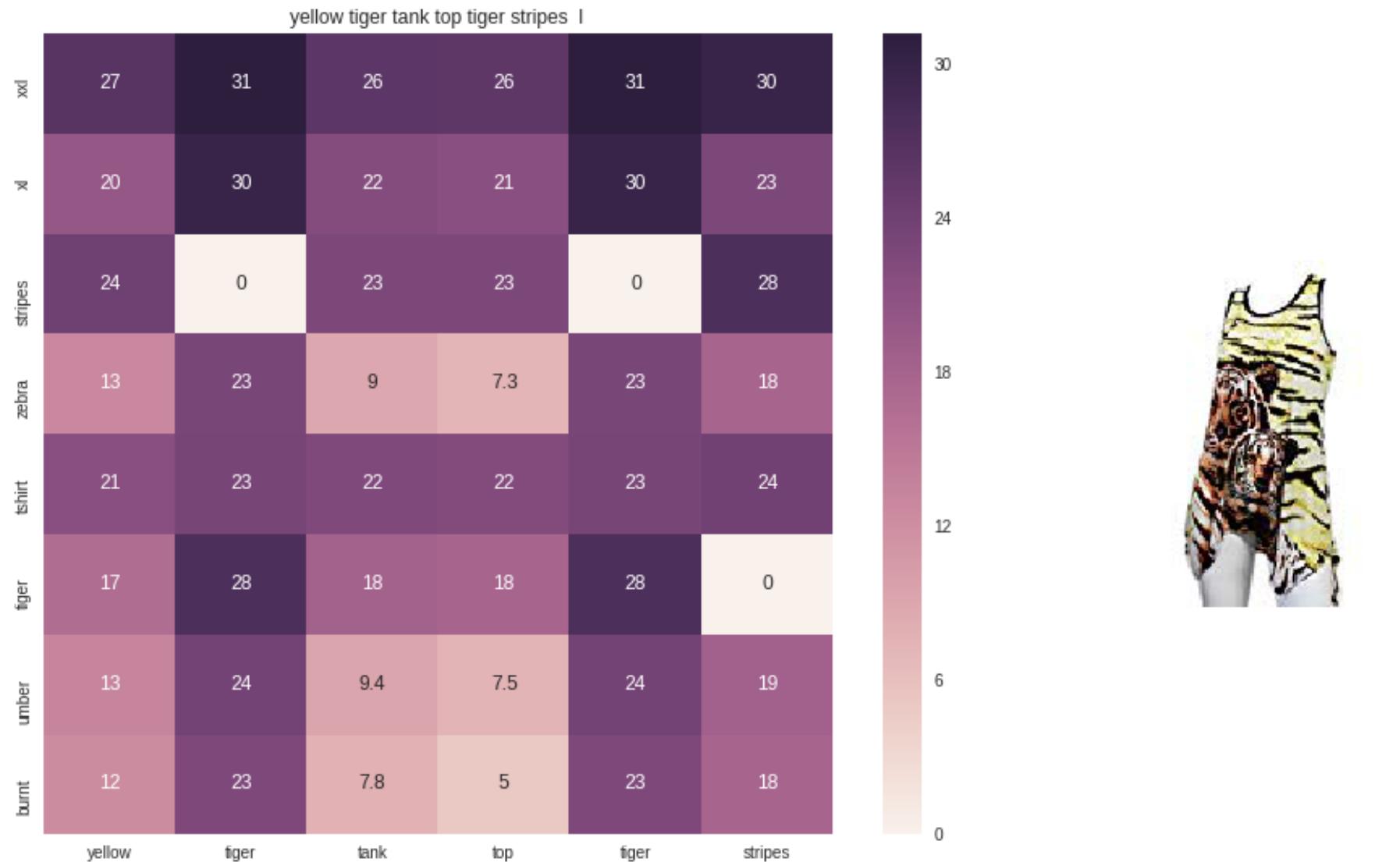
euclidean distance from input : 4.770942



ASIN : B00JXQAFZ2

Brand : Si Row

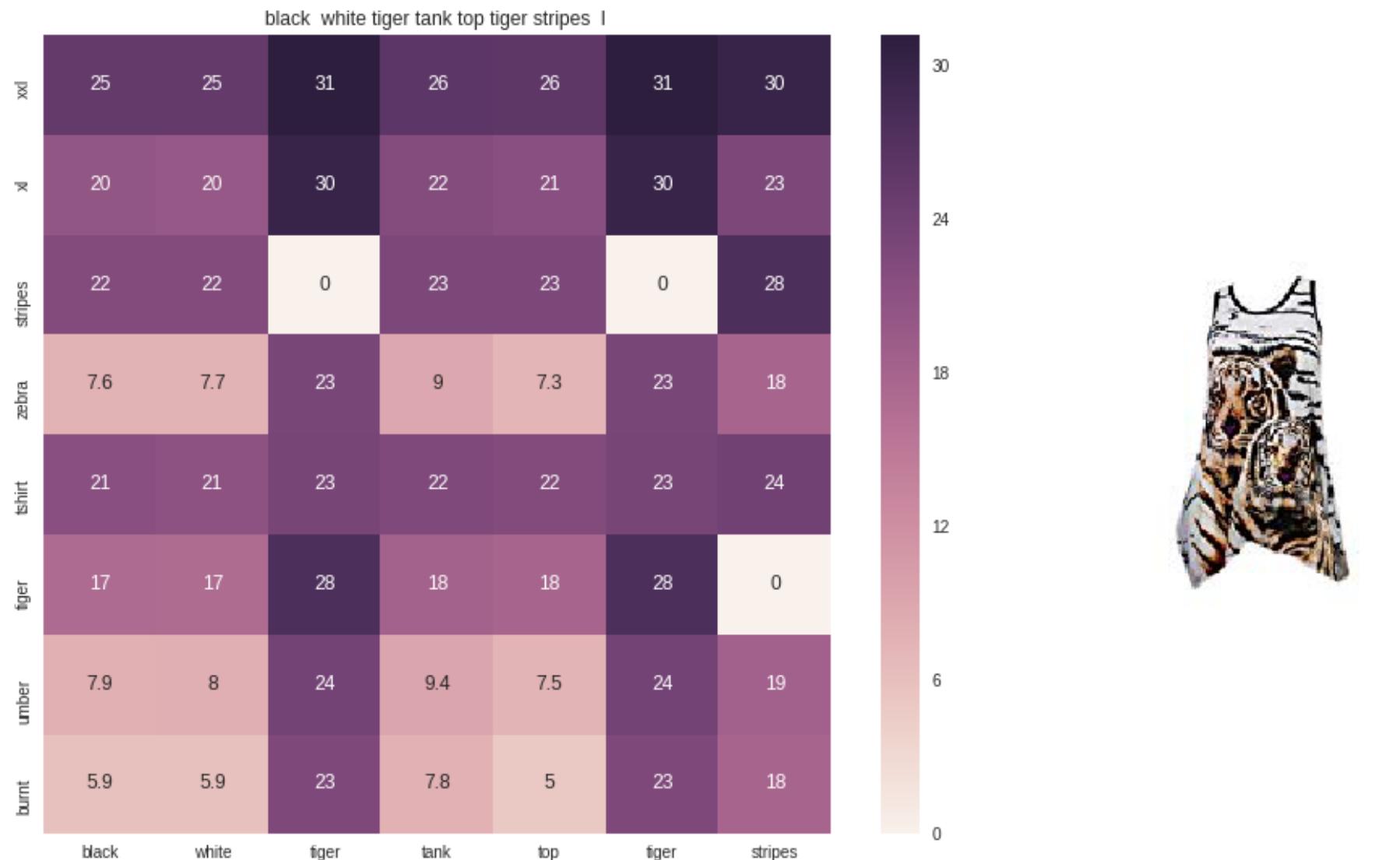
euclidean distance from input : 5.3601613



ASIN : B00JXQAUWA

Brand : Si Row

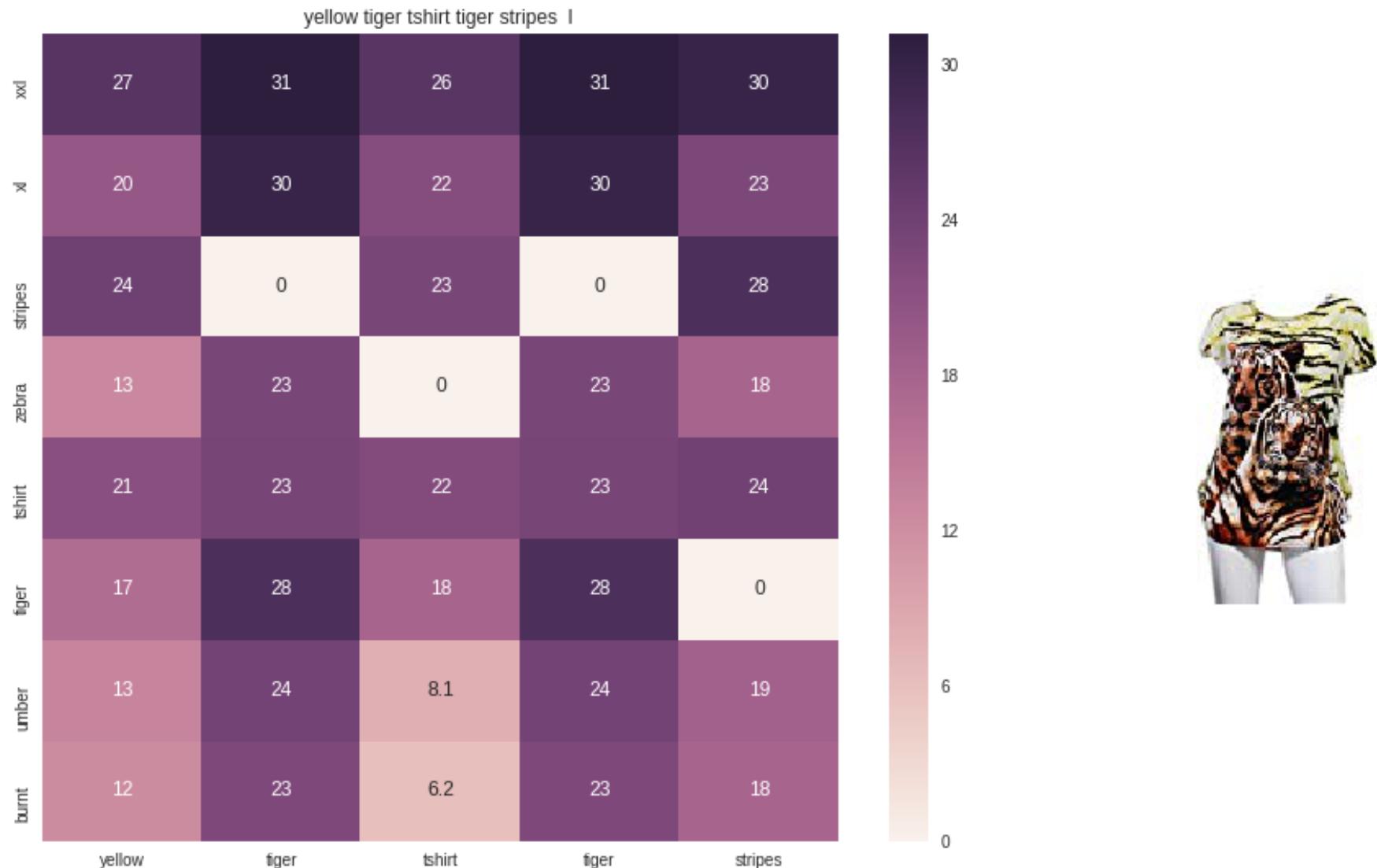
euclidean distance from input : 5.689523



ASIN : B00JXQA094

Brand : Si Row

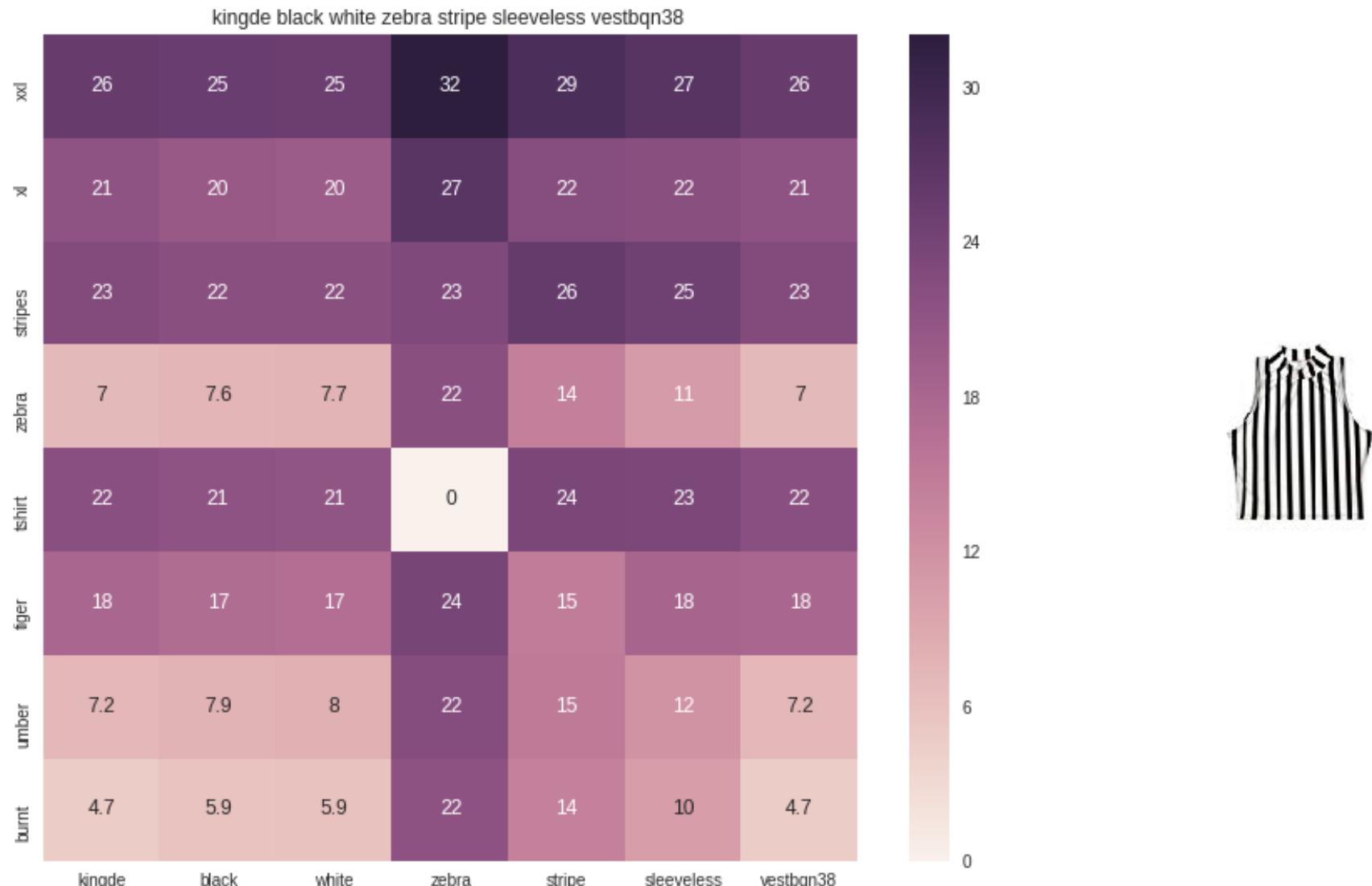
euclidean distance from input : 5.6930227



ASIN : B00JXQCUIC

Brand : Si Row

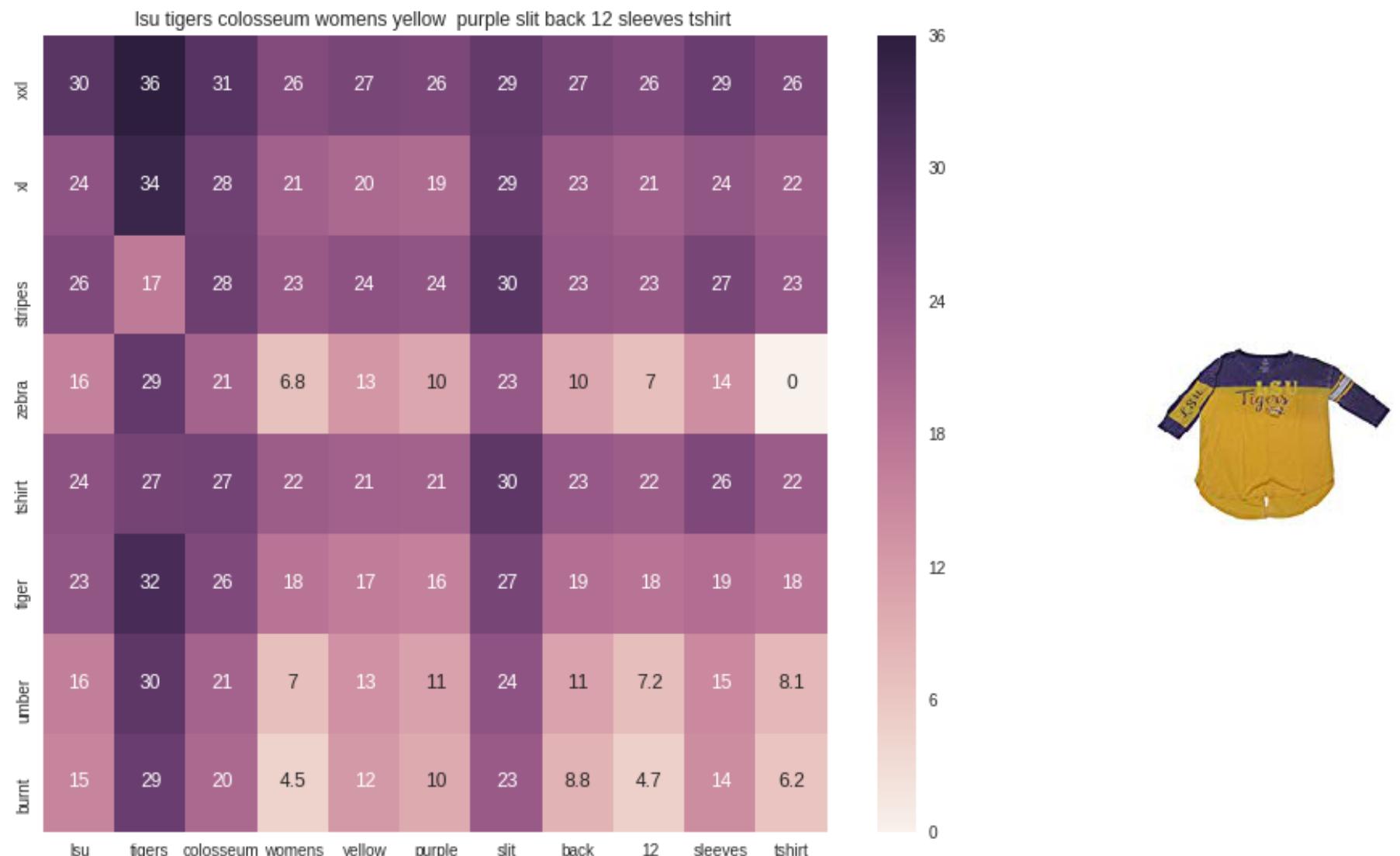
euclidean distance from input : 5.8934426



ASIN : B015H41F6G

Brand : KINGDE

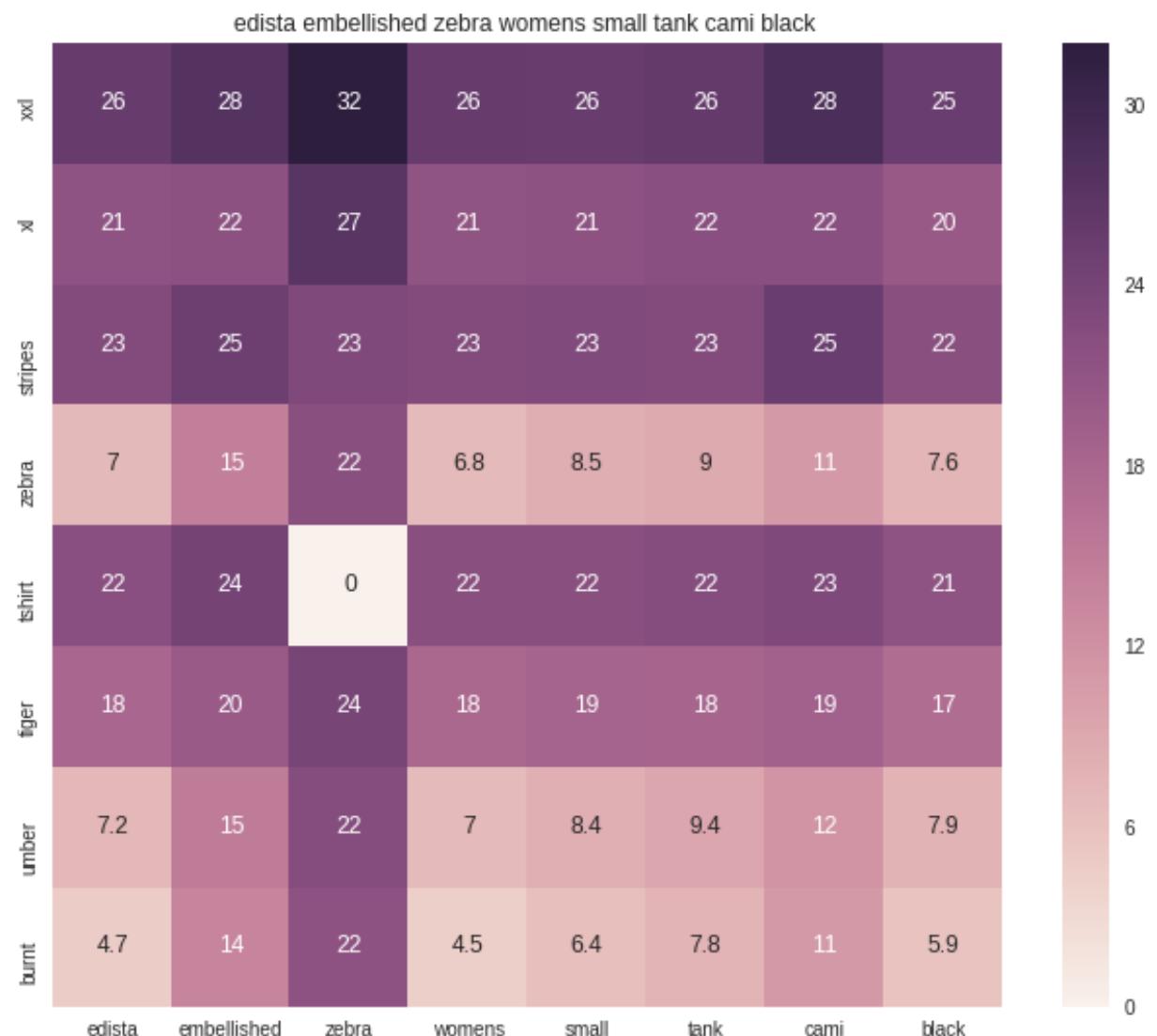
euclidean distance from input : 6.13299



ASIN : B073R5Q8HD

Brand : Colosseum

euclidean distance from input : 6.2567058

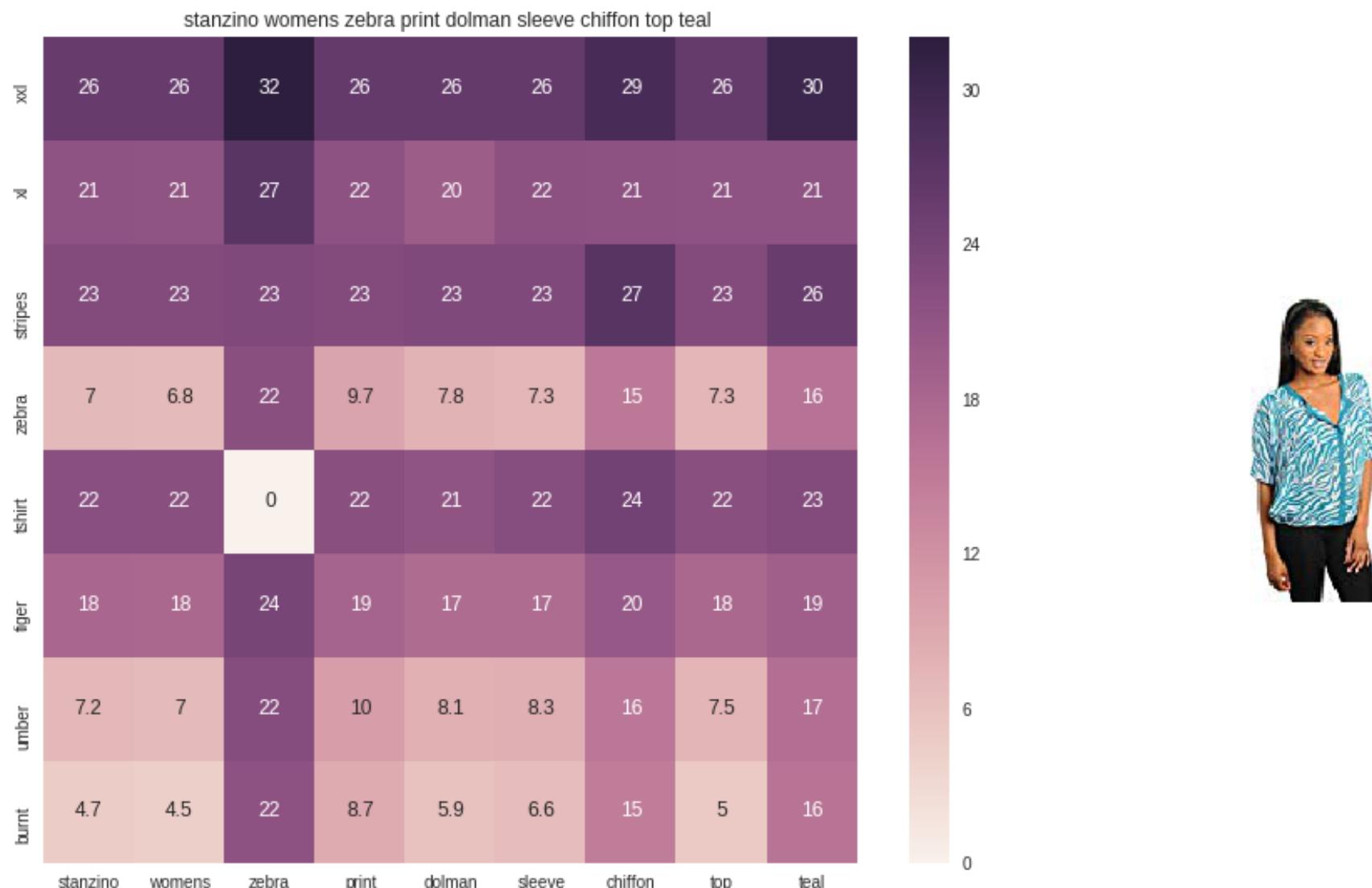


ASIN : B074P8MD22

Brand : Edista

euclidean distance from input : 6.3922043

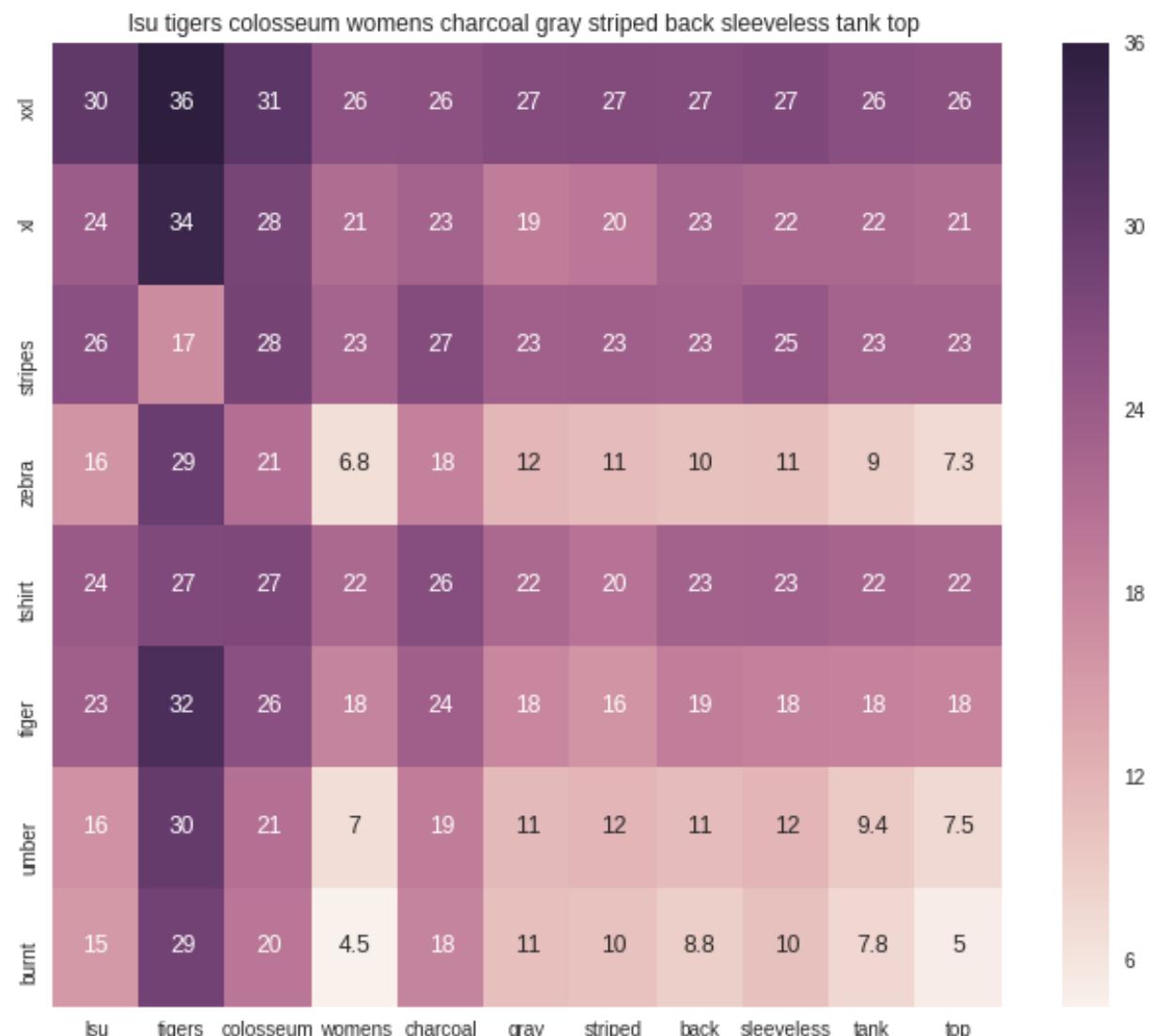
=====



ASIN : B00C0I3U3E

Brand : Stanzino

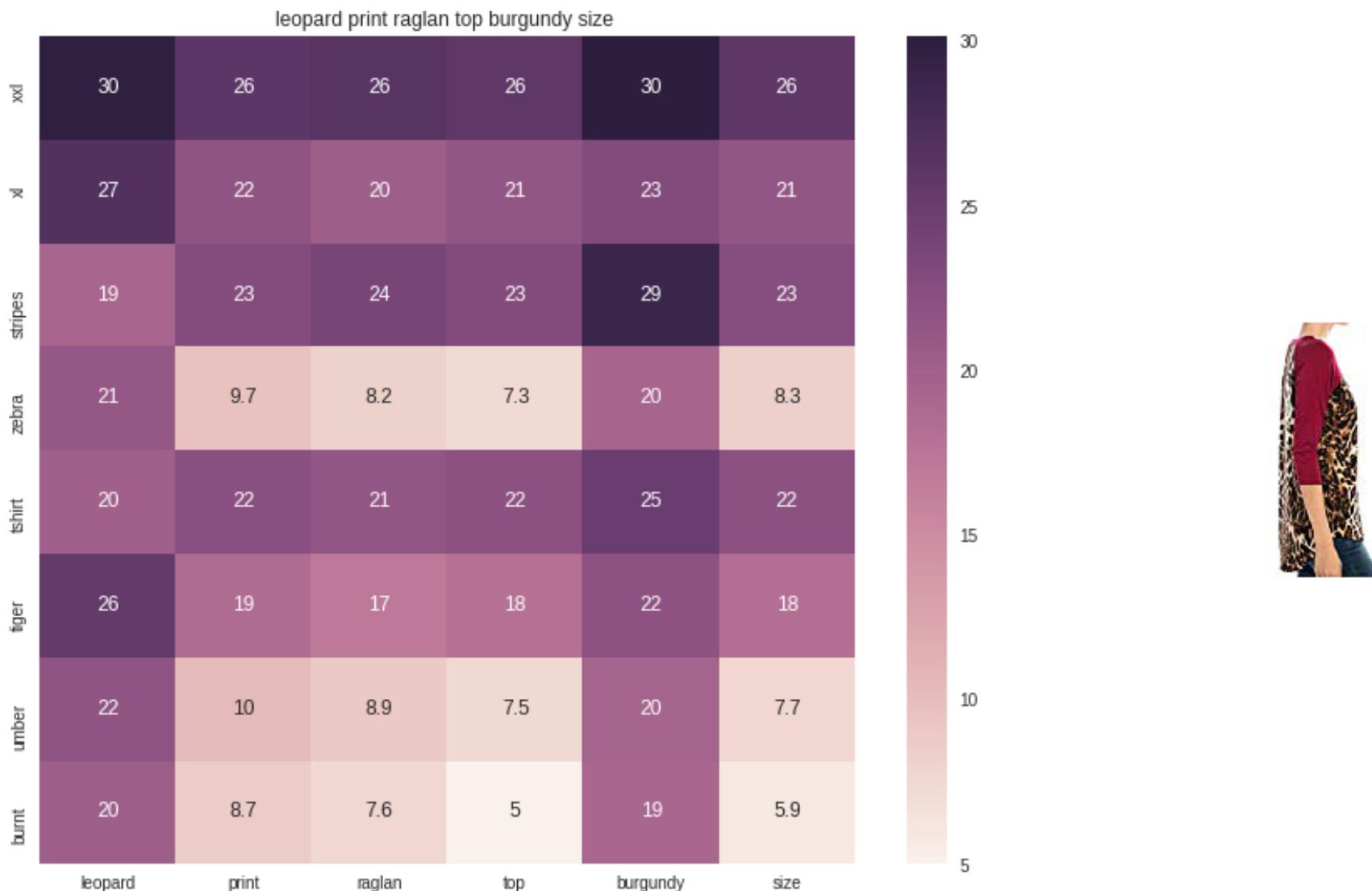
euclidean distance from input : 6.414901



ASIN : B073R4ZM7Y

Brand : Colosseum

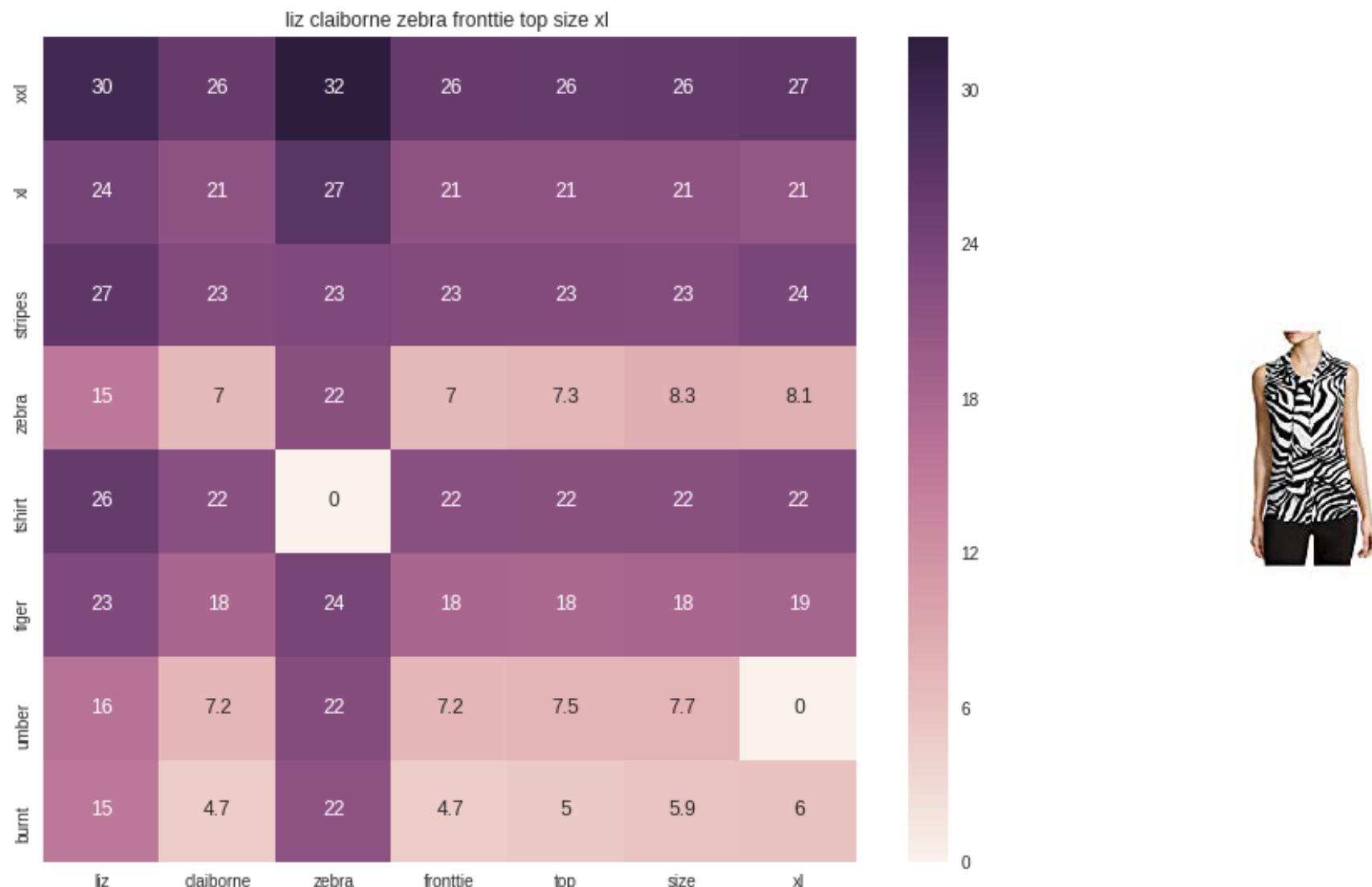
euclidean distance from input : 6.4509606



ASIN : B01C60RLDQ

Brand : 1 Mad Fit

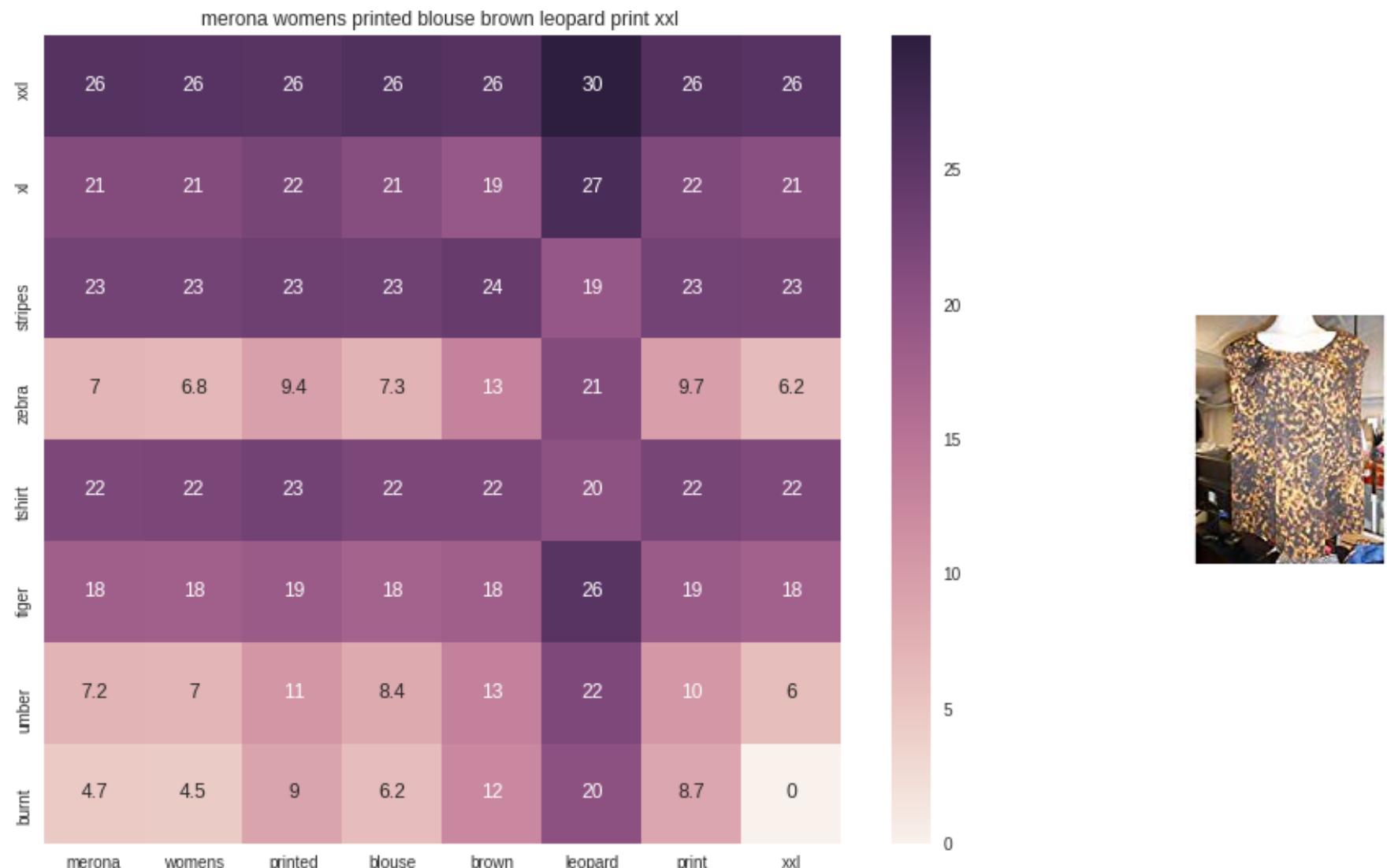
euclidean distance from input : 6.4634094



ASIN : B06XBY5QXL

Brand : Liz Claiborne

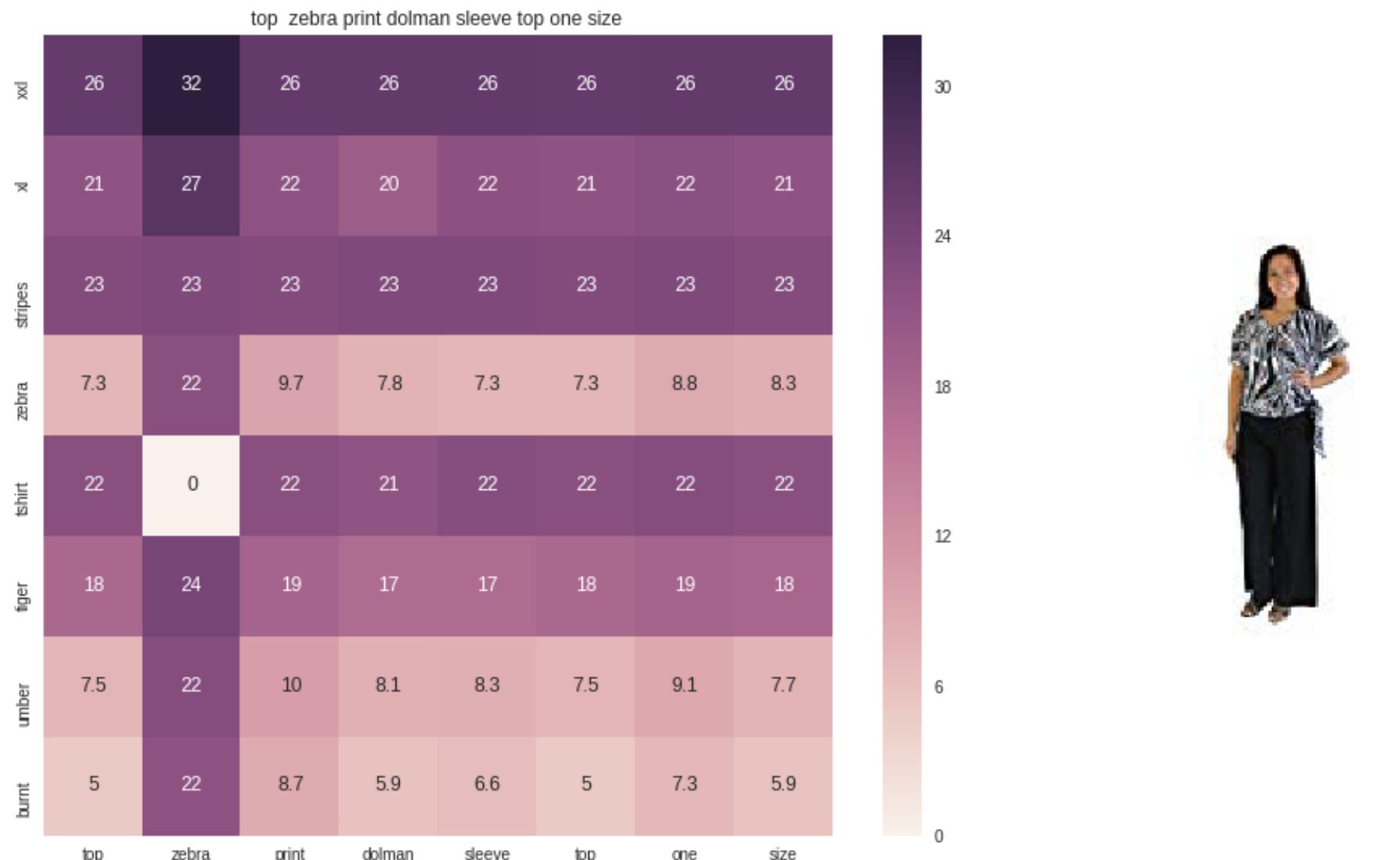
euclidean distance from input : 6.5392237



ASIN : B071YF3WDD

Brand : Merona

euclidean distance from input : 6.575504

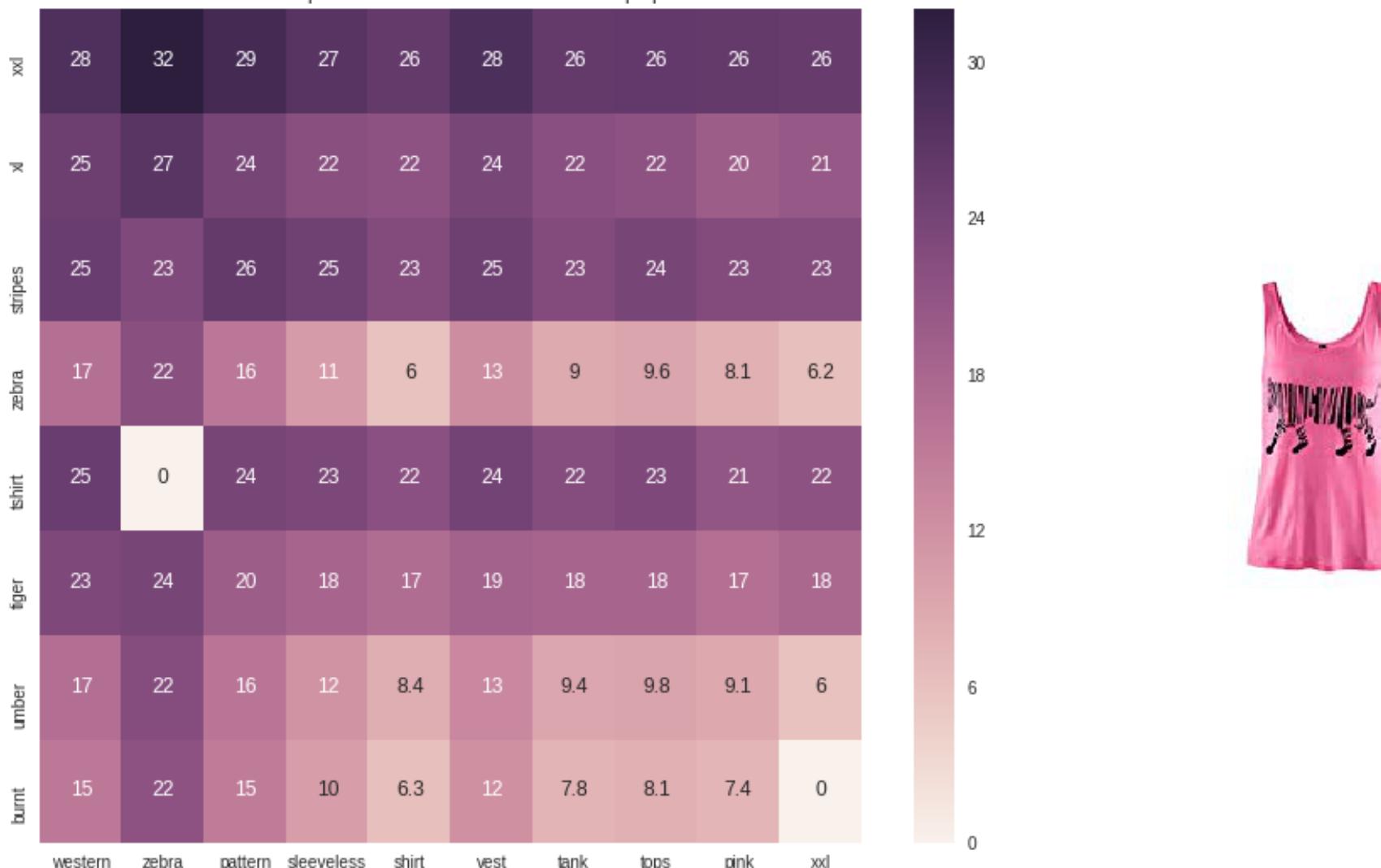


ASIN : B00H8A6ZLI

Brand : Vivian's Fashions

euclidean distance from input : 6.6382155

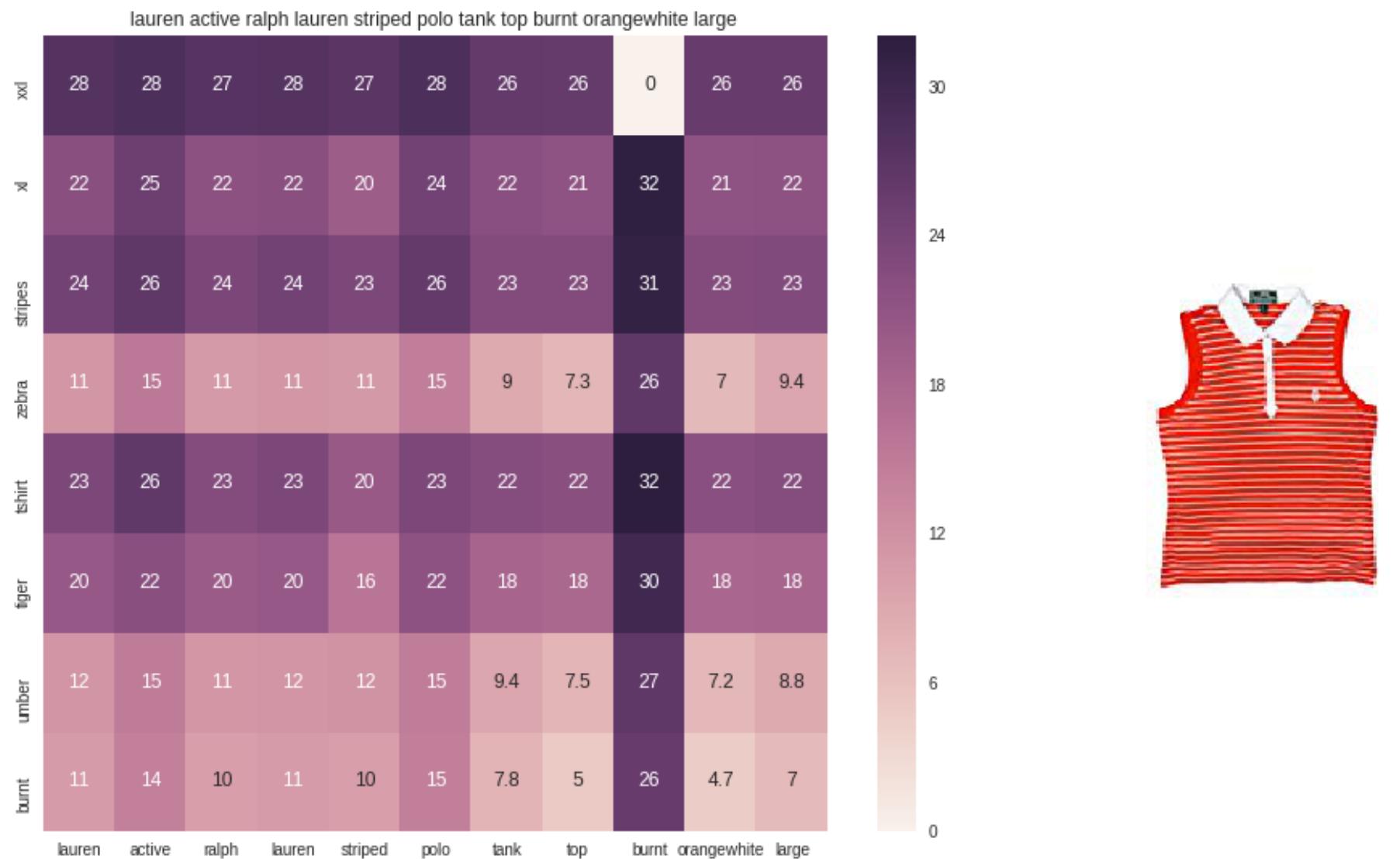
western zebra pattern sleeveless shirt vest tank tops pink xxl



ASIN : B00Z6HEXWI

Brand : Black Temptation

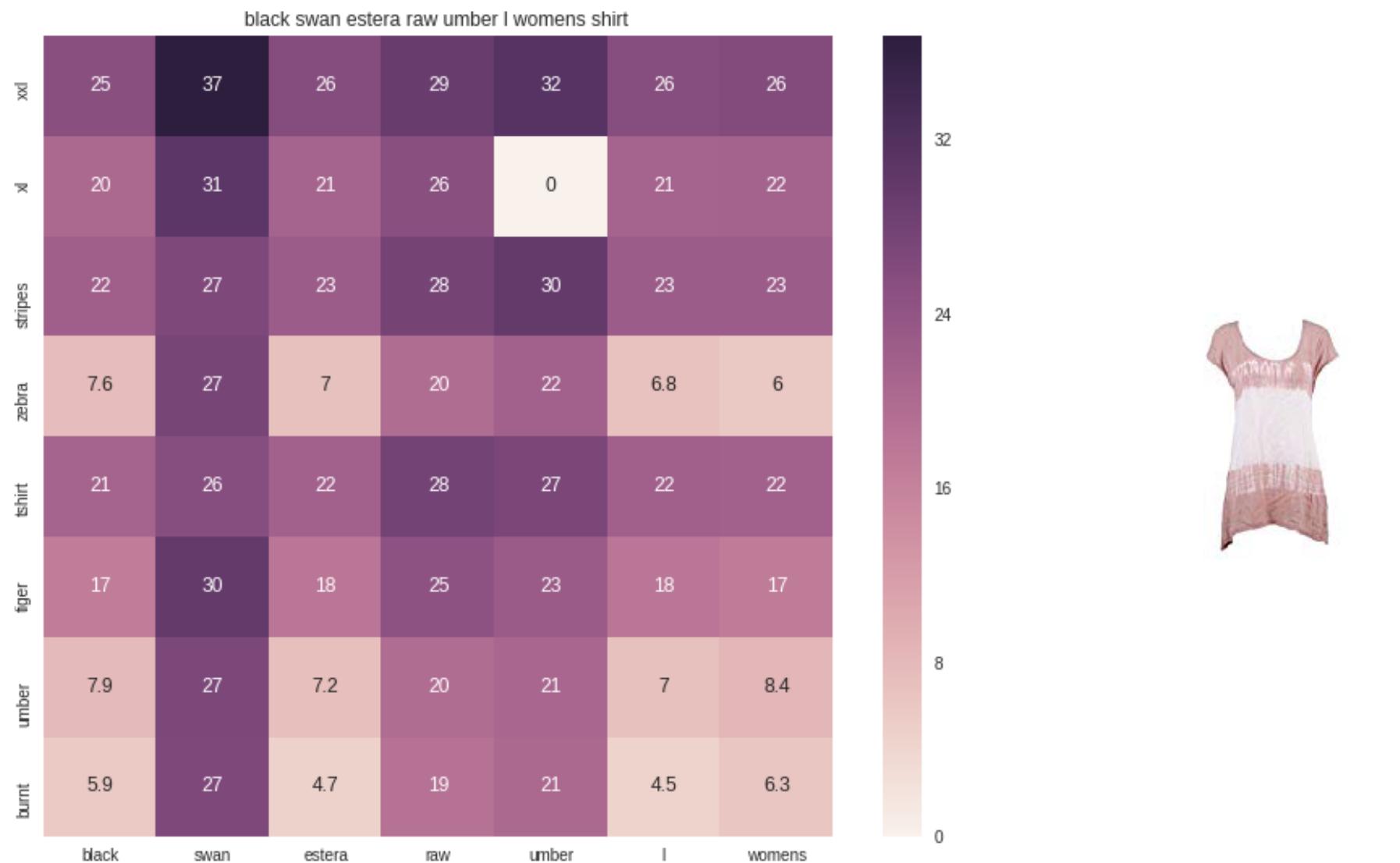
euclidean distance from input : 6.660737



ASIN : B00ILGH50Y

Brand : Ralph Lauren Active

euclidean distance from input : 6.6839056

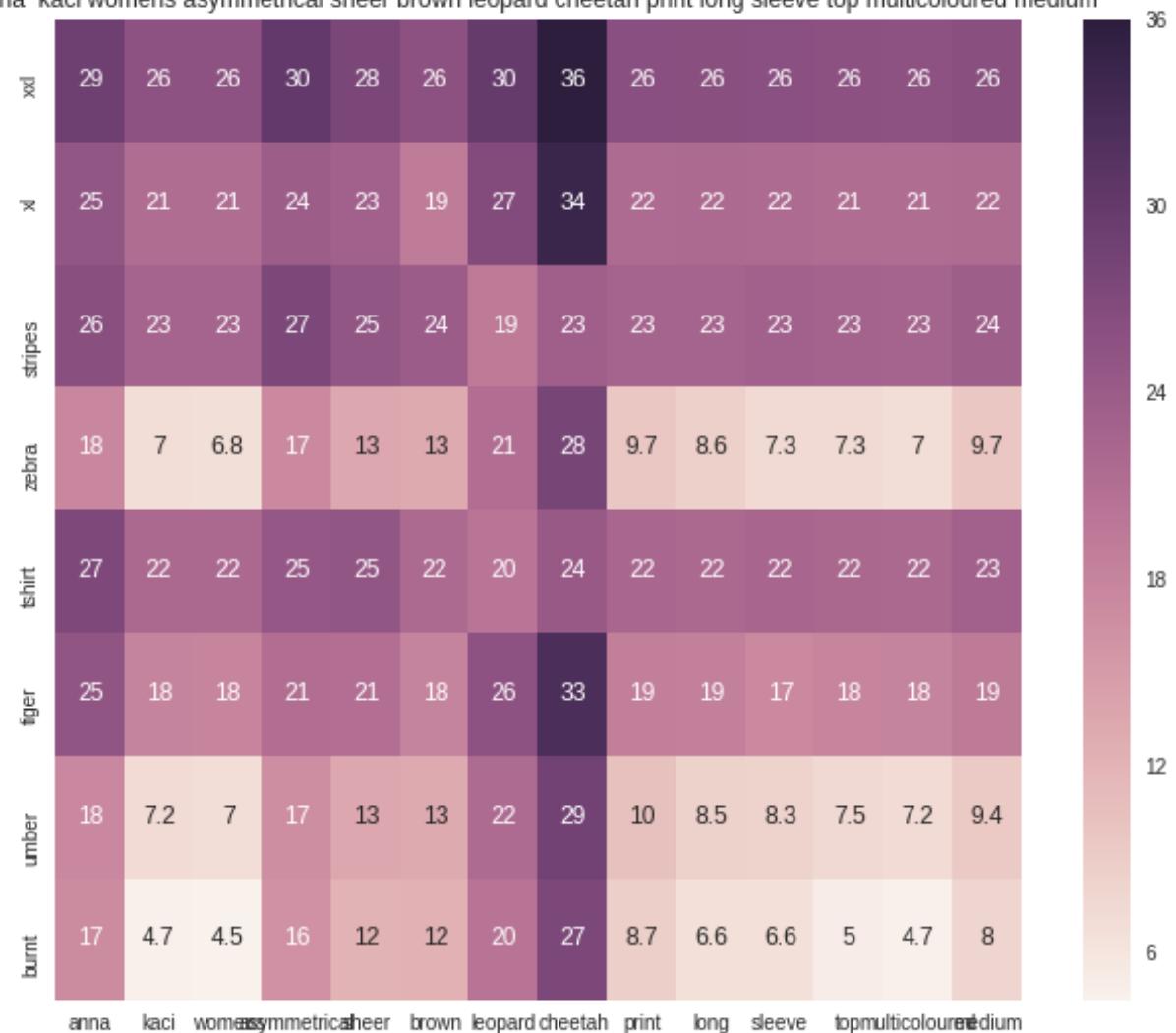


ASIN : B06Y1VN8WQ

Brand : Black Swan

euclidean distance from input : 6.7057643

anna kaci womens asymmetrical sheer brown leopard cheetah print long sleeve top multicoloured medium



ASIN : B00KSNTY7Y

Brand : Anna-Kaci

euclidean distance from input : 6.7061253

=====

[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 correspo  
        nds 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 correspo  
        nds 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 fe  
                      features  
                      [data['asin'].loc[df_id2],brands[doc_id2], colors[doc_id2], types[doc_id2]]] # recommended appare  
                      l'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 [44]: 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

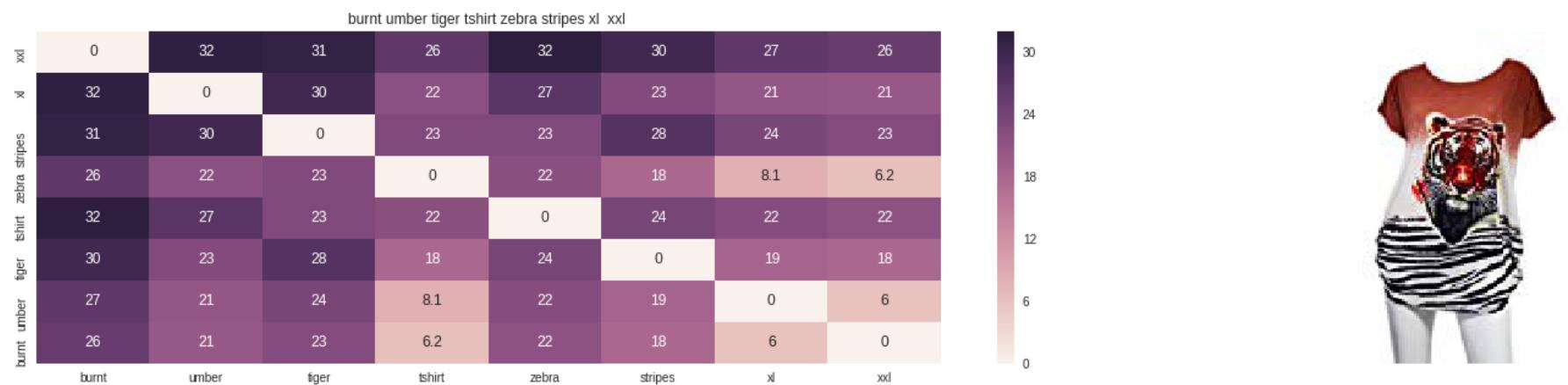
    # pairwise_dist will store the distance from given input apparel to all remaining apparels
    # the metric we used here is cosine, the coside distance is mesured as  $K(X, Y) = \langle X, Y \rangle / (\|X\| * \|Y\|)$ 
    # http://scikit-Learn.org/stable/modules/metrics.html#cosine-similarity
    idf_w2v_dist = 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)

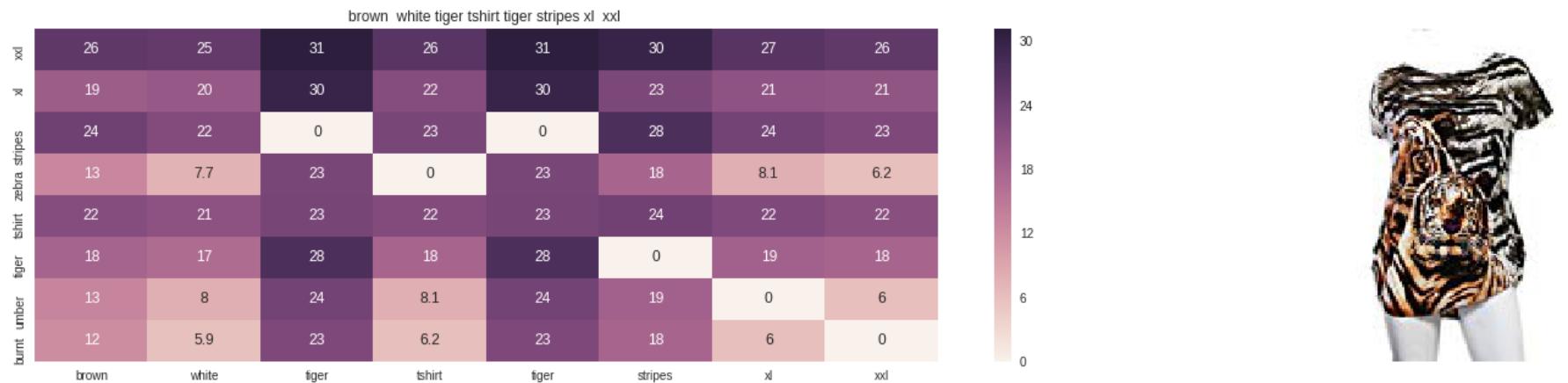
    idf_w2v_brand(12566, 5, 5, 20)
    # in the give heat map, each cell contains the euclidean distance between words i, j
```



ASIN : B00JXQB5FQ

Brand : Si Row

euclidean distance from input : 0.001953125



ASIN : B00JXQCWT0

Brand : Si Row

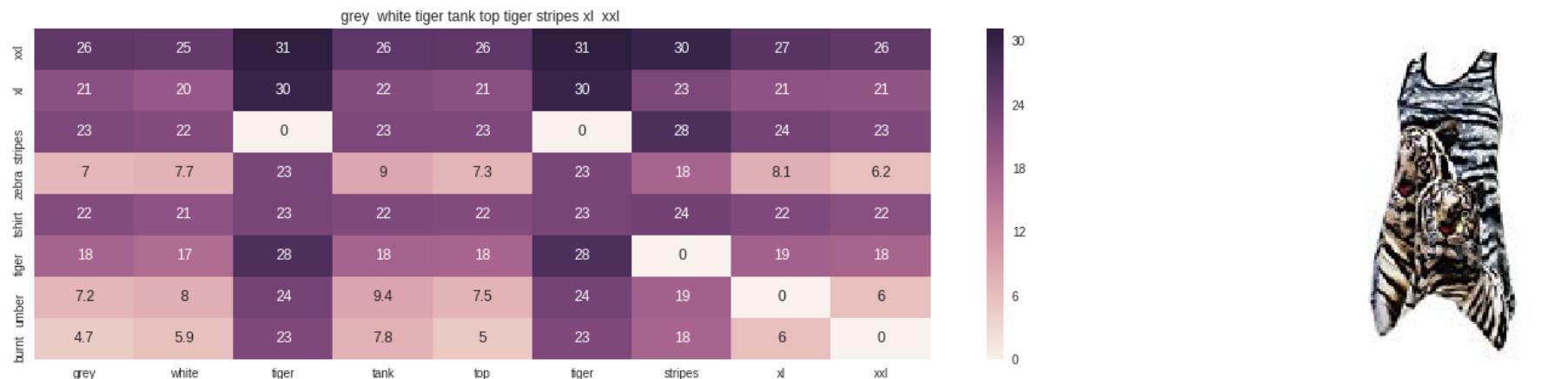
euclidean distance from input : 2.385471153259277



ASIN : B00JXQASS6

Brand : Si Row

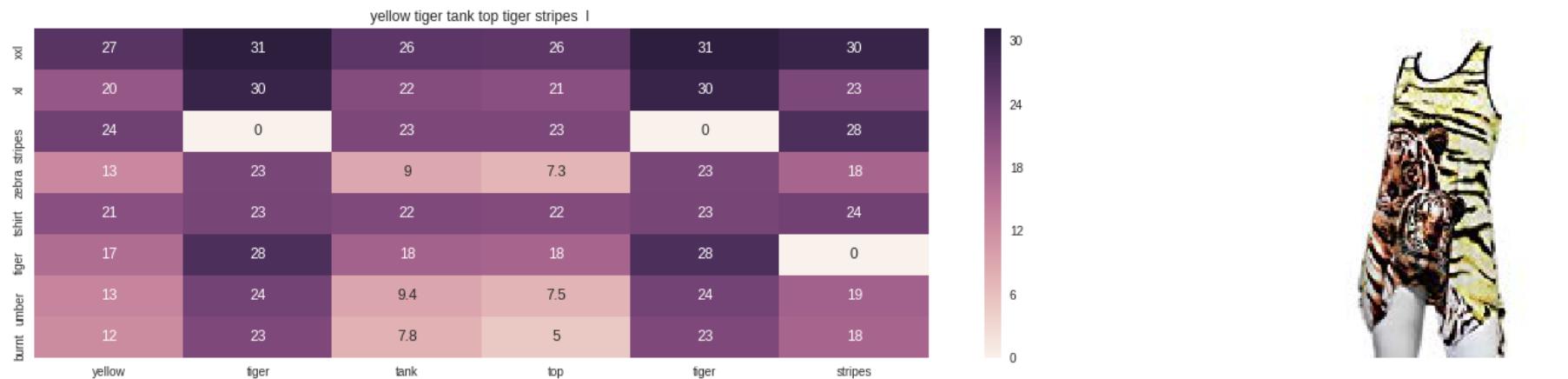
euclidean distance from input : 2.7390506746191647



ASIN : B00JXQAFZ2

Brand : Si Row

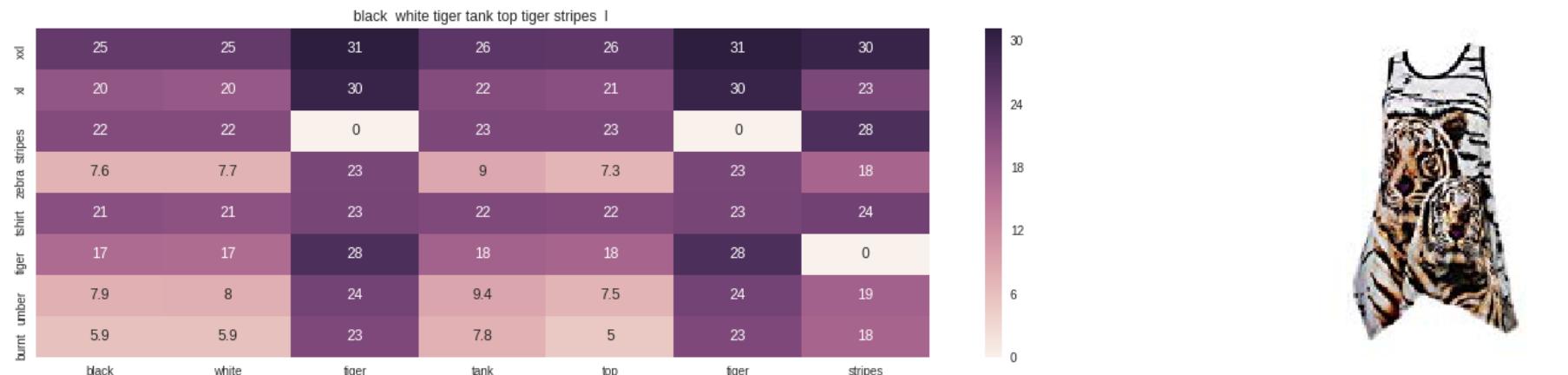
euclidean distance from input : 3.3871873857397703



ASIN : B00JXQAUWA

Brand : Si Row

euclidean distance from input : 3.5518684389013915



ASIN : B00JXQA094

Brand : Si Row

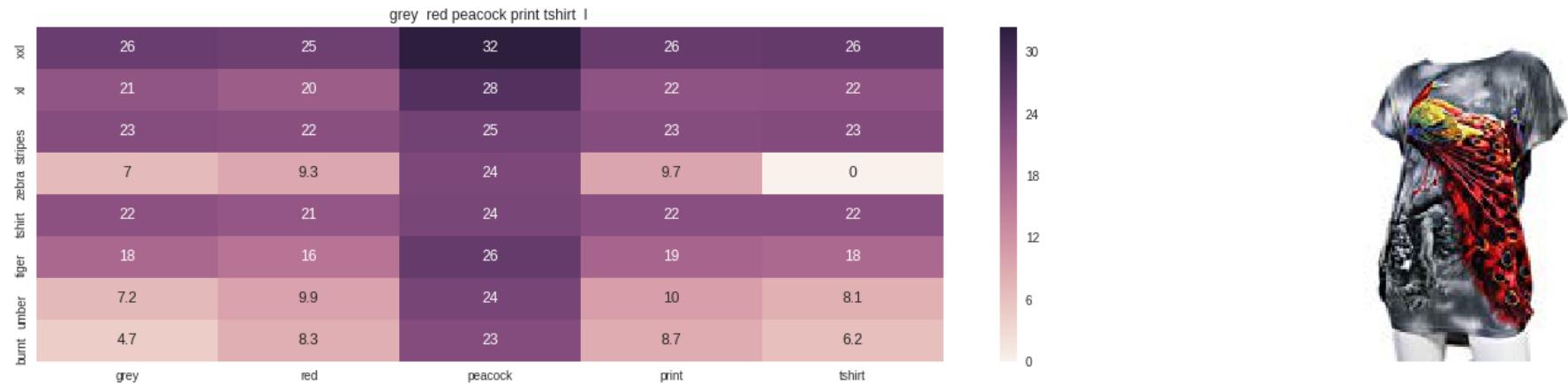
euclidean distance from input : 3.5536182405371335



ASIN : B00JXQCUIC

Brand : Si Row

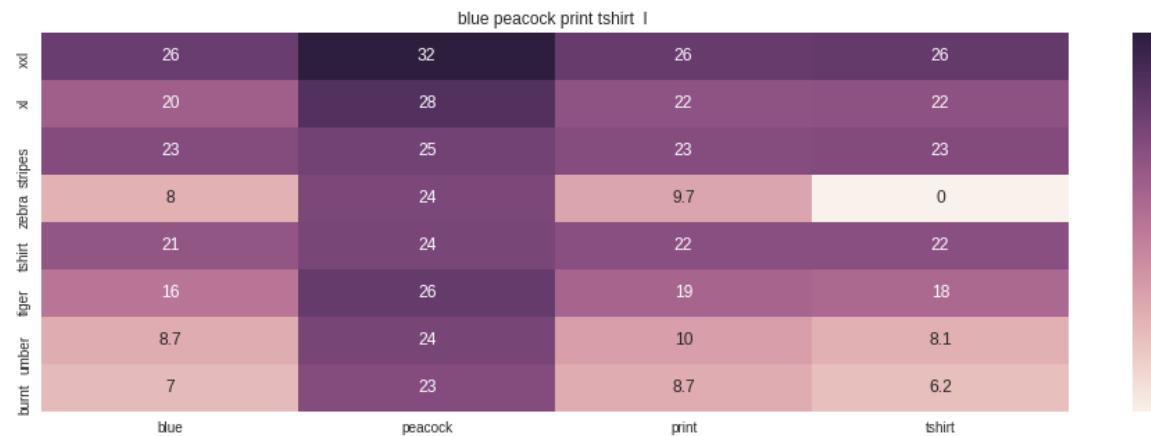
euclidean distance from input : 3.653828048886743



ASIN : B00JXQCFRS

Brand : Si Row

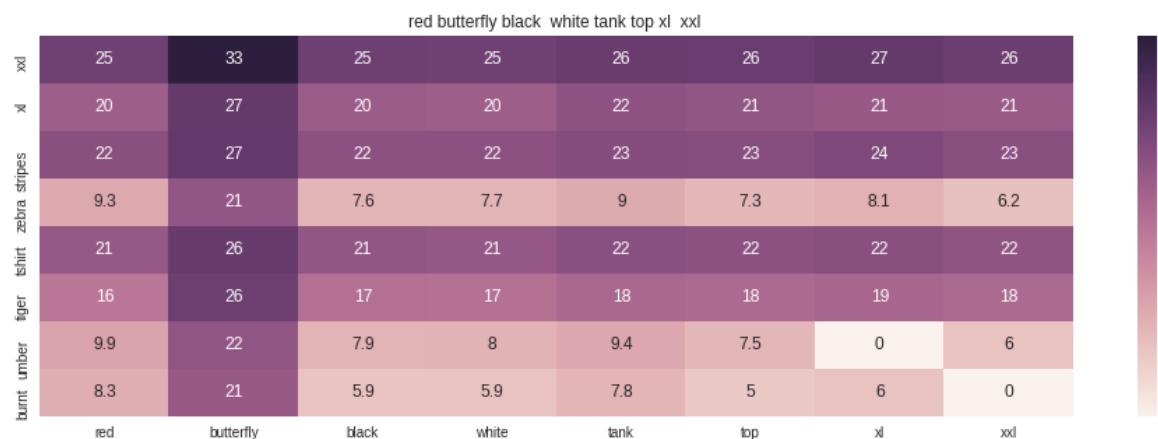
euclidean distance from input : 4.128811645688501



ASIN : B00JXQC8L6

Brand : Si Row

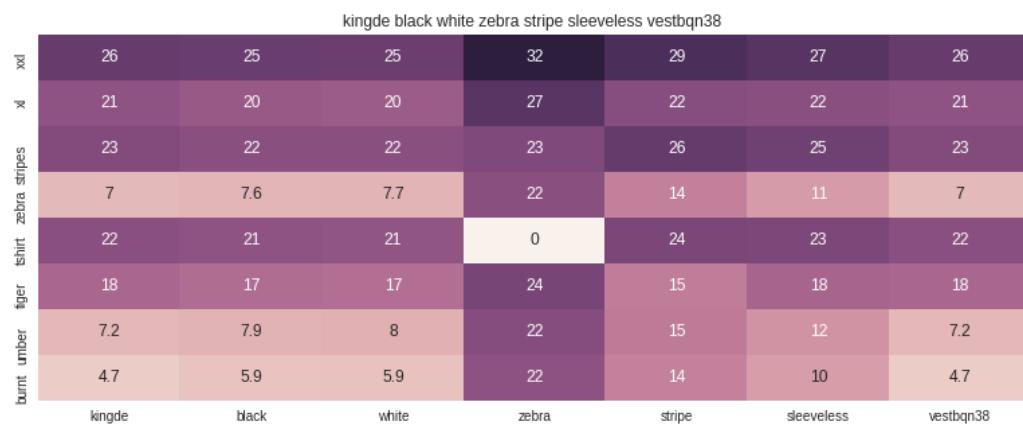
euclidean distance from input : 4.203900146665063



ASIN : B00JV63CW2

Brand : Si Row

euclidean distance from input : 4.286586761655298

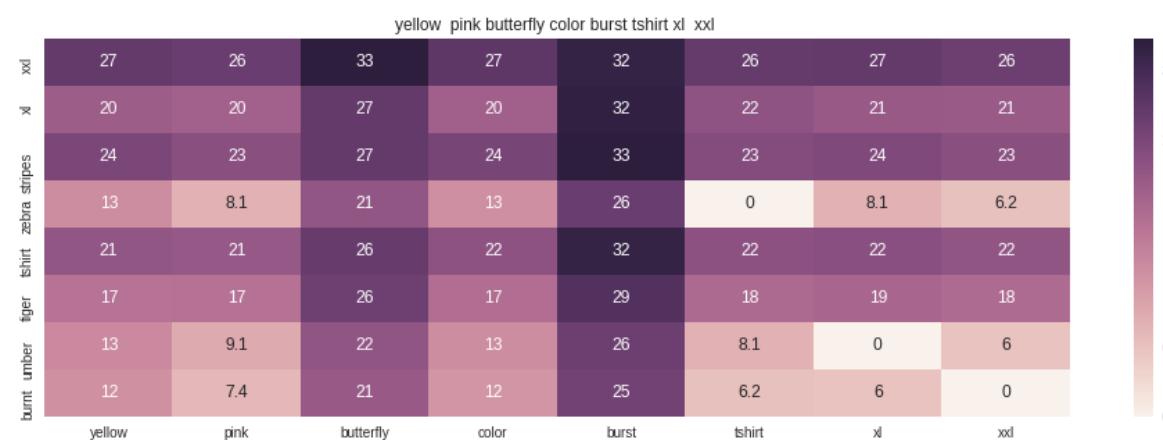


ASIN : B015H41F6G

Brand : KINGDE

euclidean distance from input : 4.389370597243721

=====

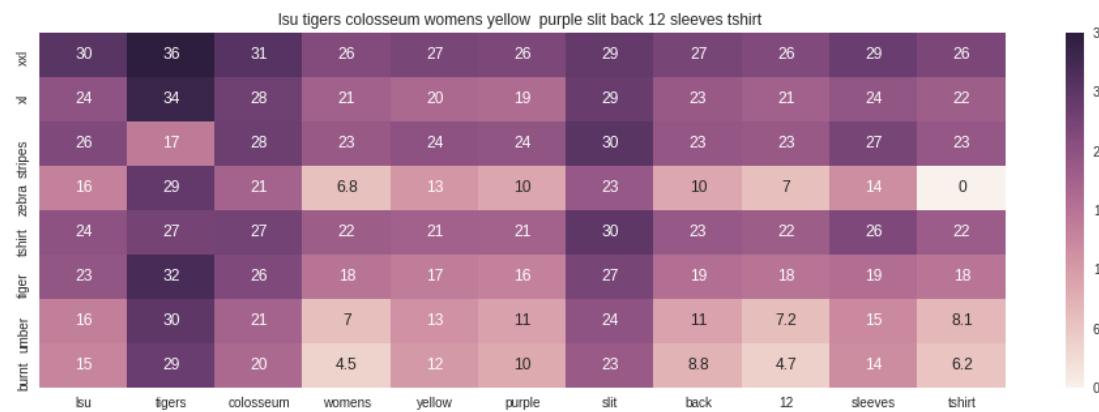


ASIN : B00JXQBBMI

Brand : Si Row

euclidean distance from input : 4.397909927548852

=====



ASIN : B073R5Q8HD

Brand : Colosseum

euclidean distance from input : 4.451228583693917



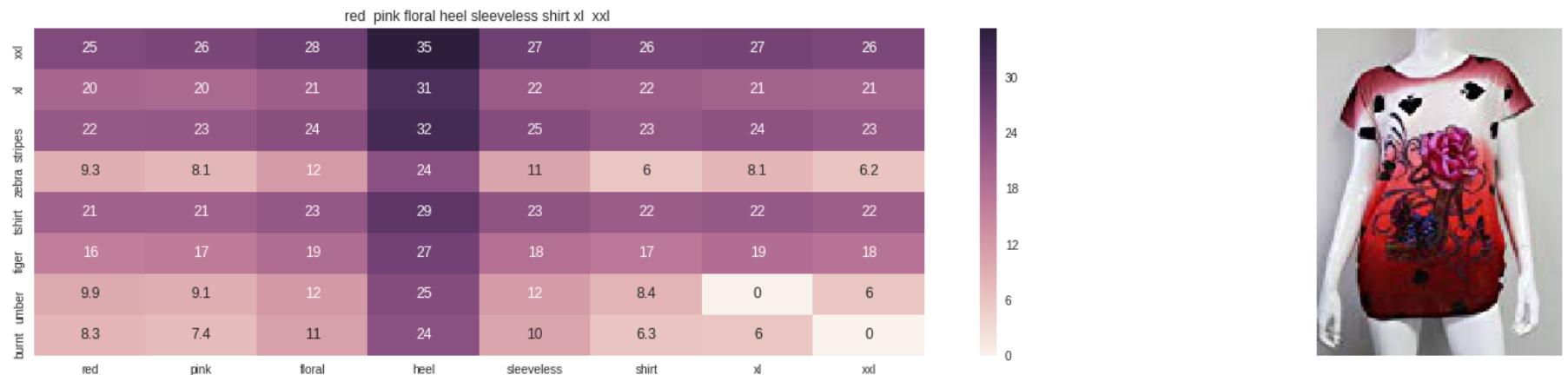
ASIN : B074P8MD22

Brand : Edista

euclidean distance from input : 4.518977797866279

=====

=====



ASIN : B00JV63QQE

Brand : Si Row

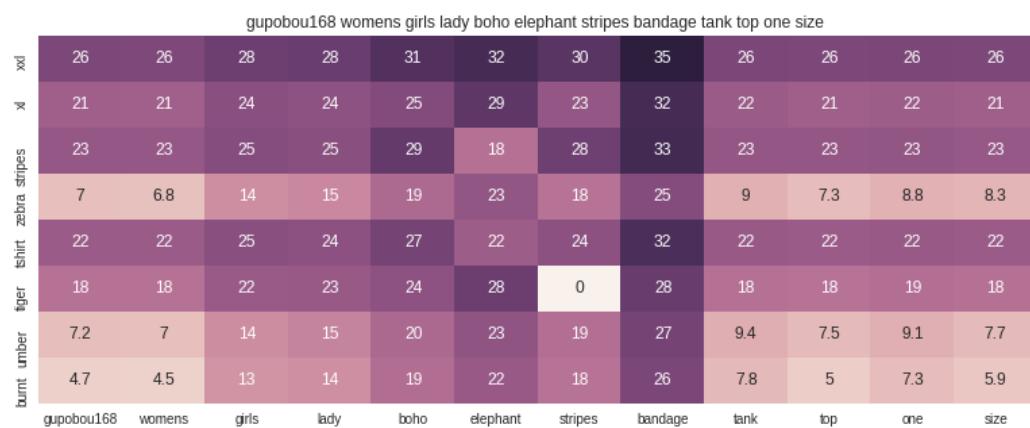
euclidean distance from input : 4.52937545794436



ASIN : B00C0I3U3E

Brand : Stanzino

euclidean distance from input : 4.530326140761788



ASIN : B01ER18406

Brand : GuPoBoU168

euclidean distance from input : 4.546817024028215

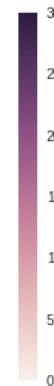
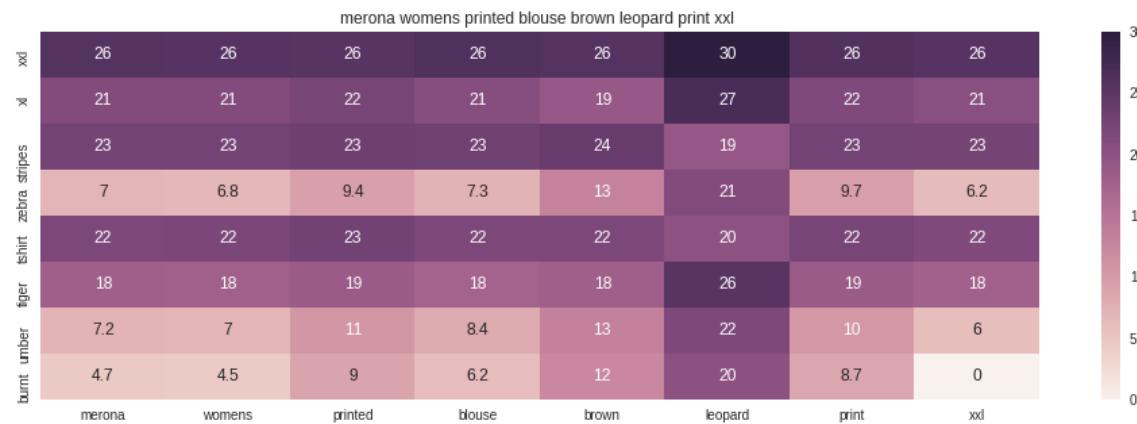
lsu tigers colosseum womens charcoal gray striped back sleeveless tank top											
zoom	30	36	31	26	26	27	27	27	27	26	26
x	24	34	28	21	23	19	20	23	22	22	21
z	26	17	28	23	27	23	23	23	25	23	23
zebra stripes	16	29	21	6.8	18	12	11	10	11	9	7.3
tshirt	24	27	27	22	26	22	20	23	23	22	22
iger	23	32	26	18	24	18	16	19	18	18	18
burnt umber	16	30	21	7	19	11	12	11	12	9.4	7.5
top	15	29	20	4.5	18	11	10	8.8	10	7.8	5
lsu	tigers	colosseum	womens	charcoal	gray	striped	back	sleeveless	tank	top	



ASIN : B073R4ZM7Y

Brand : Colosseum

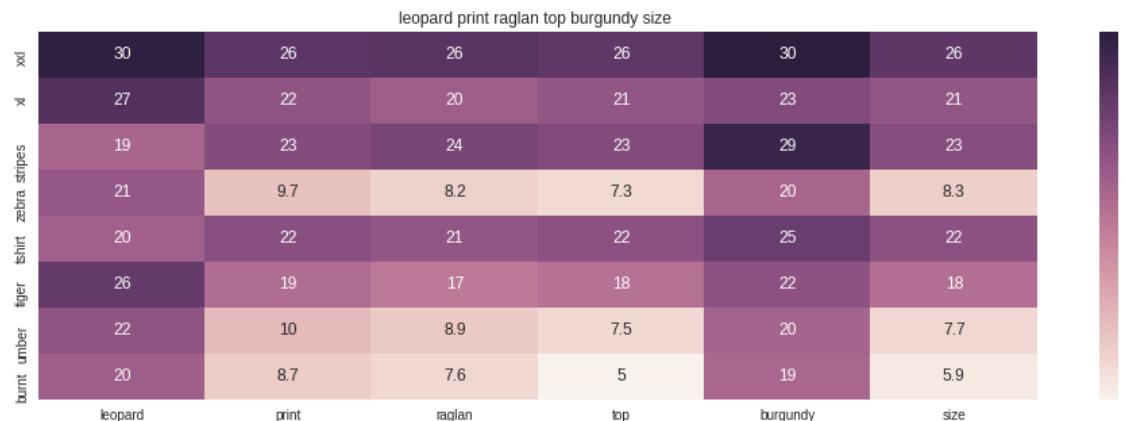
euclidean distance from input : 4.548355925918037



ASIN : B071YF3WDD

Brand : Merona

euclidean distance from input : 4.610627425551827



ASIN : B01C60RLDQ

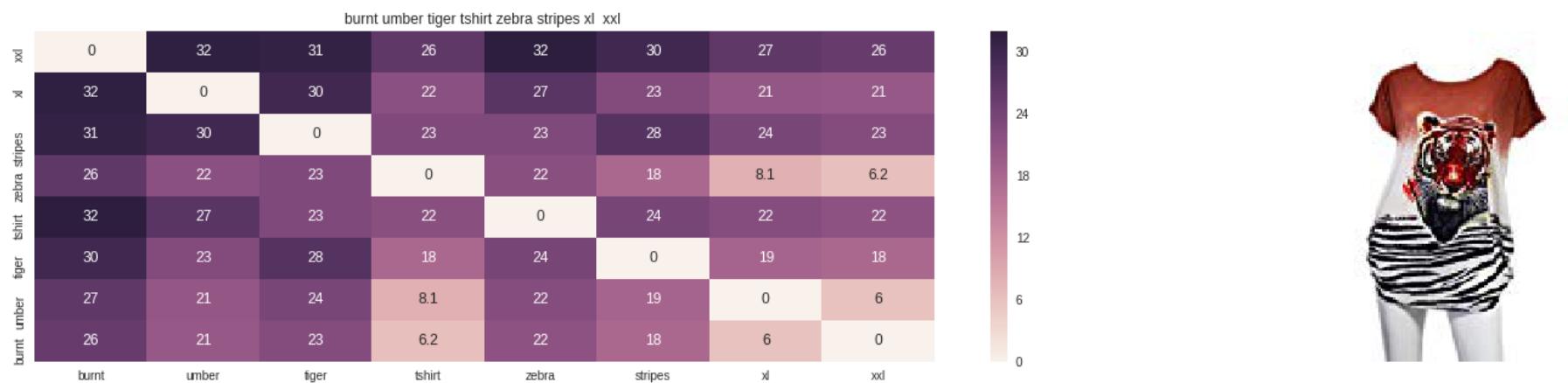
Brand : 1 Mad Fit

euclidean distance from input : 4.645918274287157

=====

=====

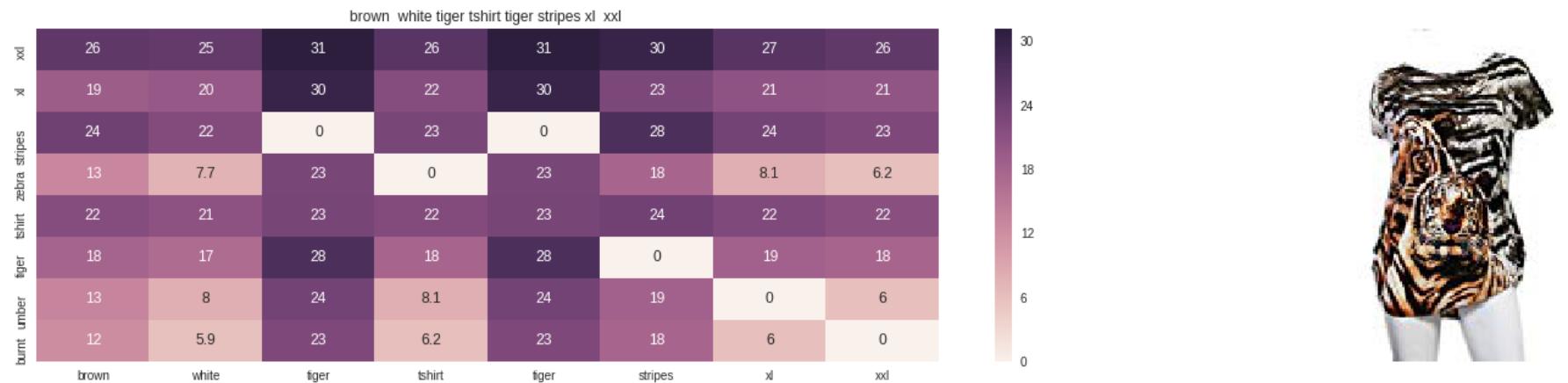
```
In [45]: # brand and color weight =50  
# title vector weight = 5  
  
idf_w2v_brand(12566, 5, 50, 20)
```



ASIN : B00JXQB5FQ

Brand : Si Row

euclidean distance from input : 0.0003551136363636364



ASIN : B00JXQCWT0

Brand : Si Row

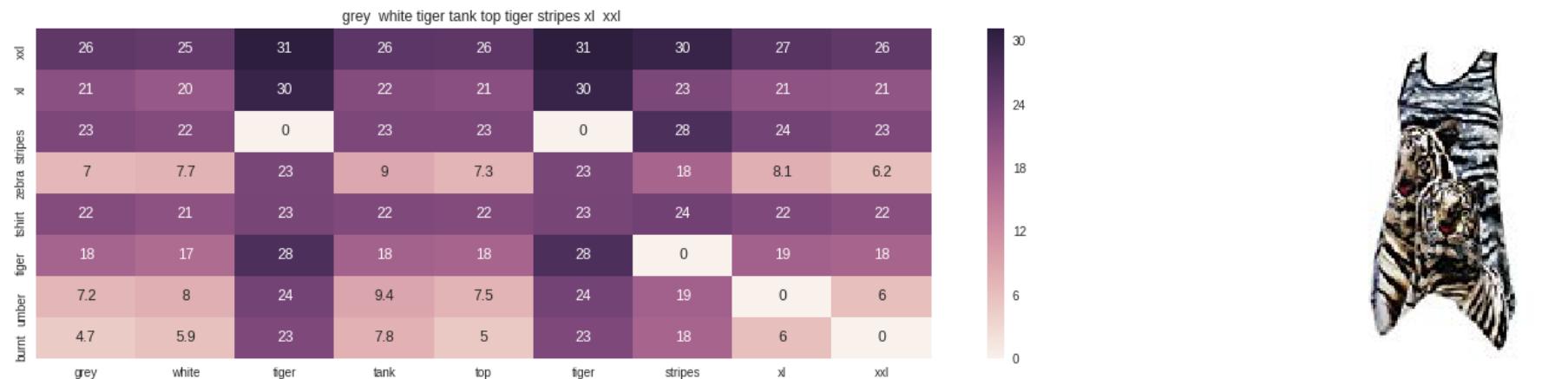
euclidean distance from input : 0.43372202786532316



ASIN : B00JXQASS6

Brand : Si Row

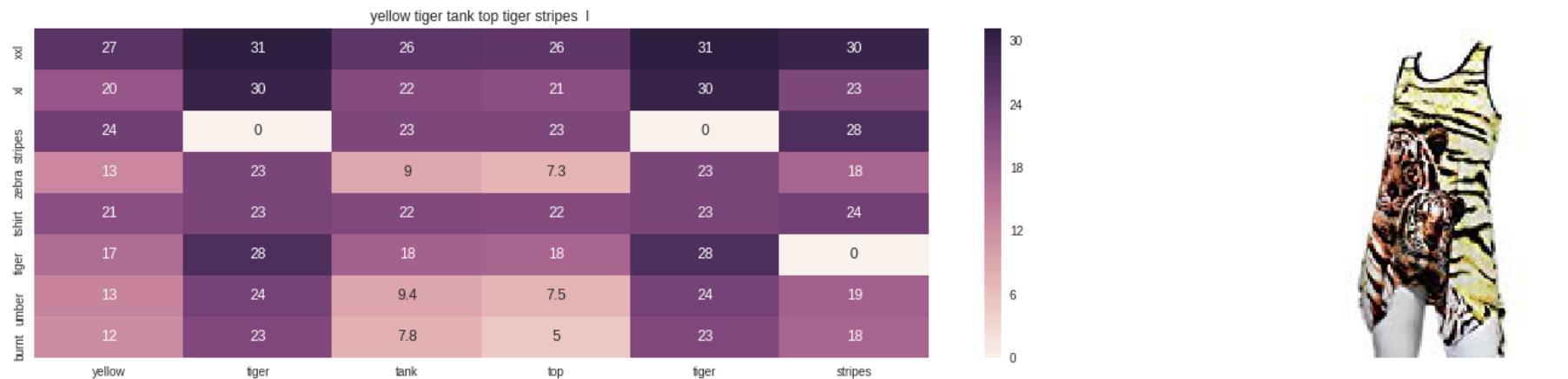
euclidean distance from input : 1.655093037326926



ASIN : B00JXQAFZ2

Brand : Si Row

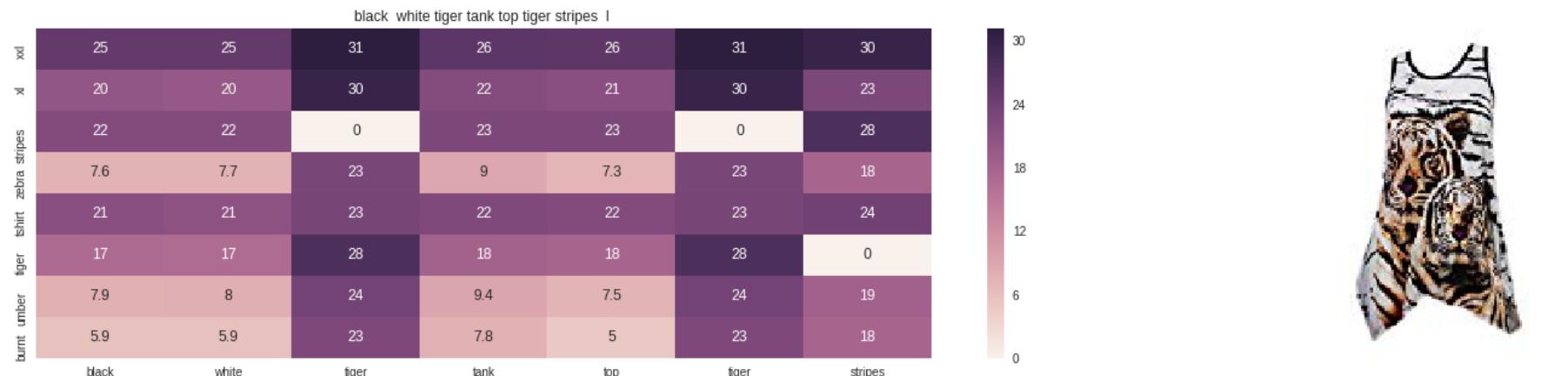
euclidean distance from input : 1.7729360757124906



ASIN : B00JXQAUWA

Brand : Si Row

euclidean distance from input : 1.8028780853782398



ASIN : B00JXQA094

Brand : Si Row

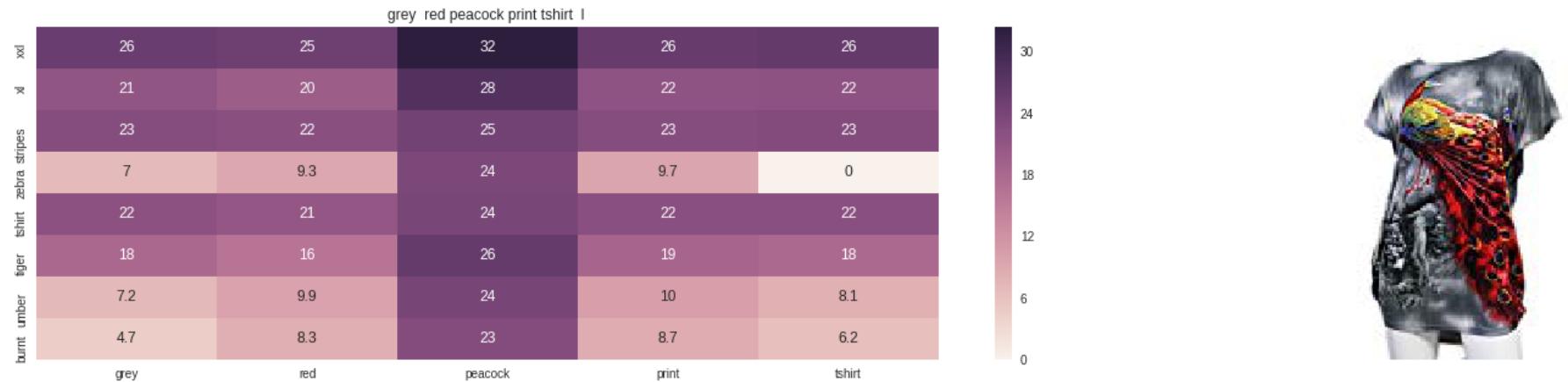
euclidean distance from input : 1.803196231130193



ASIN : B00JXQCUIC

Brand : Si Row

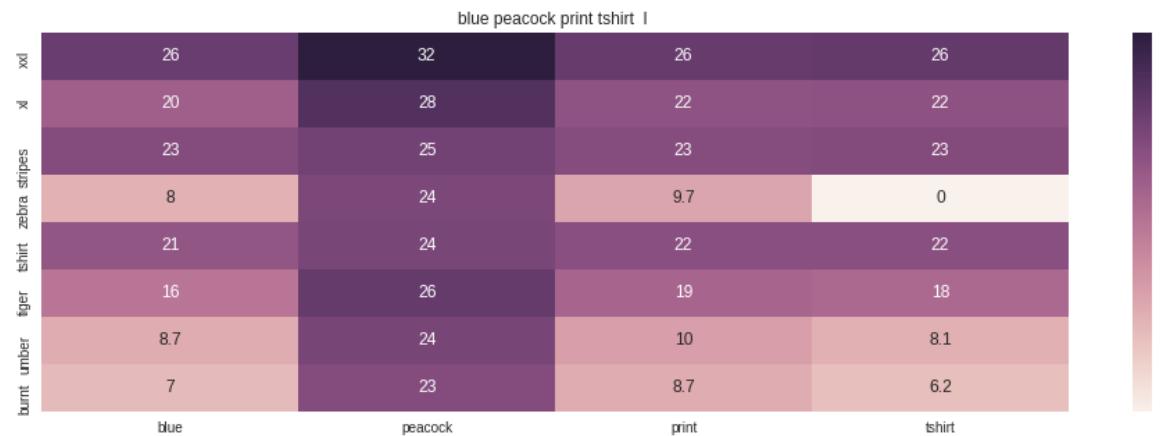
euclidean distance from input : 1.8214161962846673



ASIN : B00JXQCFRS

Brand : Si Row

euclidean distance from input : 1.9077768502486234



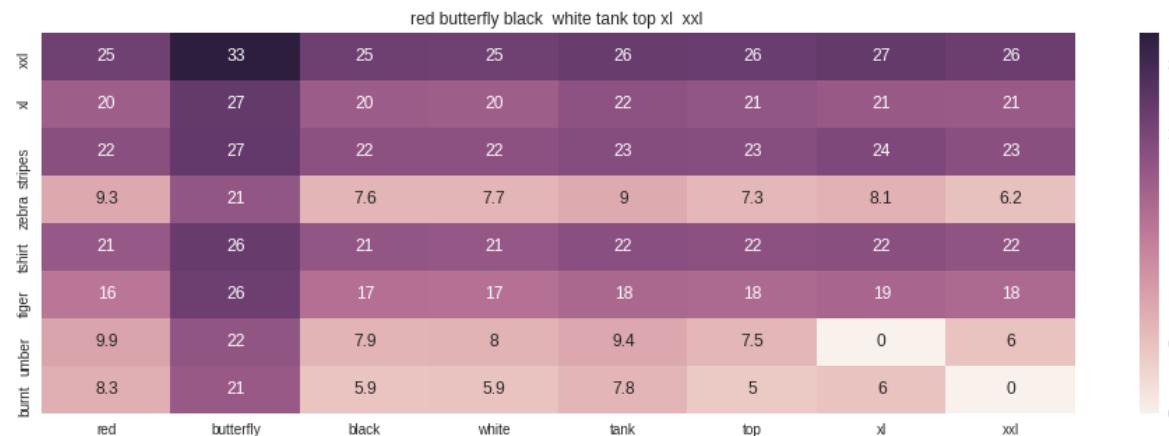
ASIN : B00JXQC8L6

Brand : Si Row

euclidean distance from input : 1.9214293049716347

=====

=====



ASIN : B00JV63CW2

Brand : Si Row

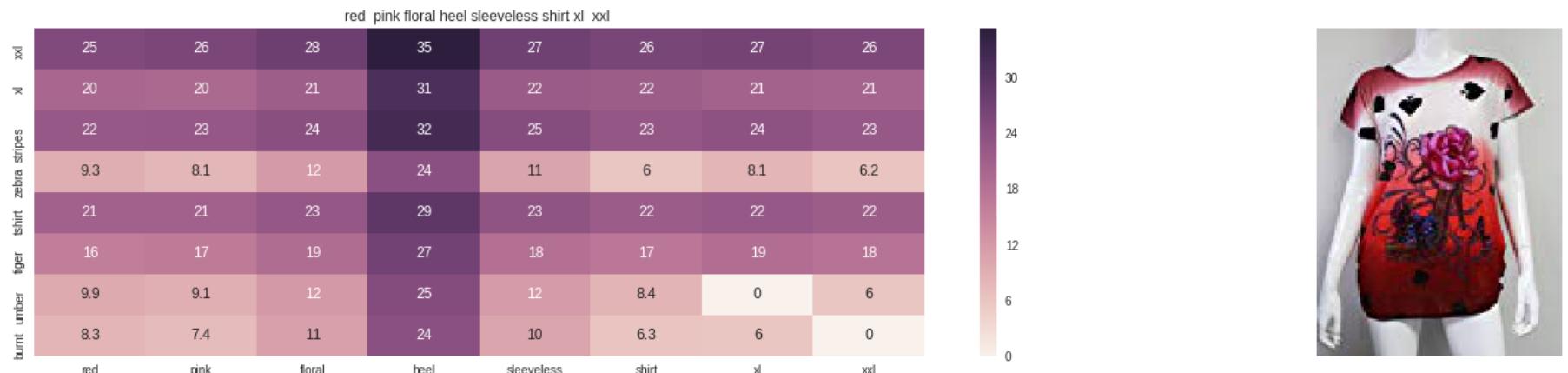
euclidean distance from input : 1.9364632349698592



ASIN : B00JXQBBMI

Brand : Si Row

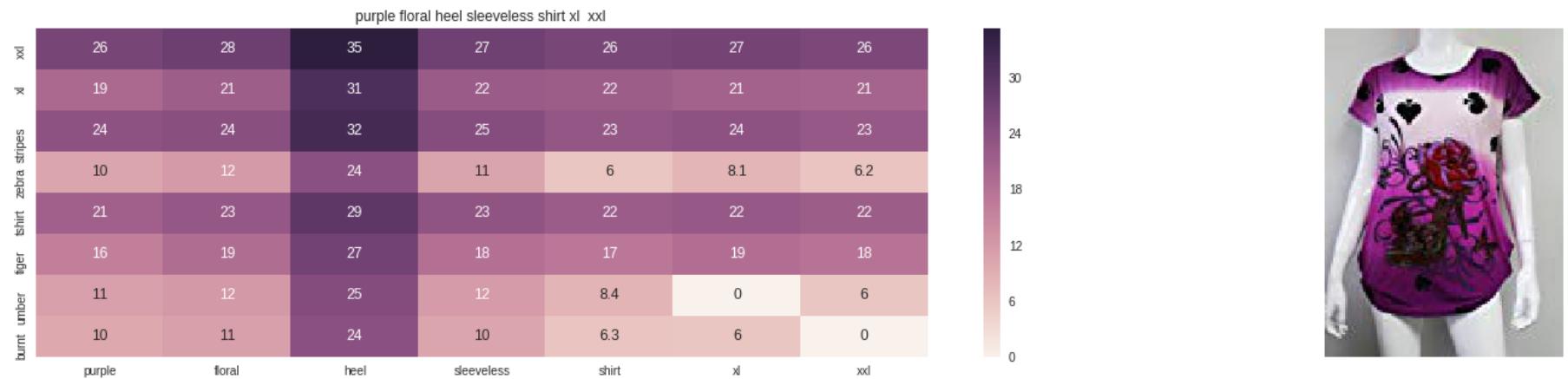
euclidean distance from input : 1.956703810586869



ASIN : B00JV63QQE

Brand : Si Row

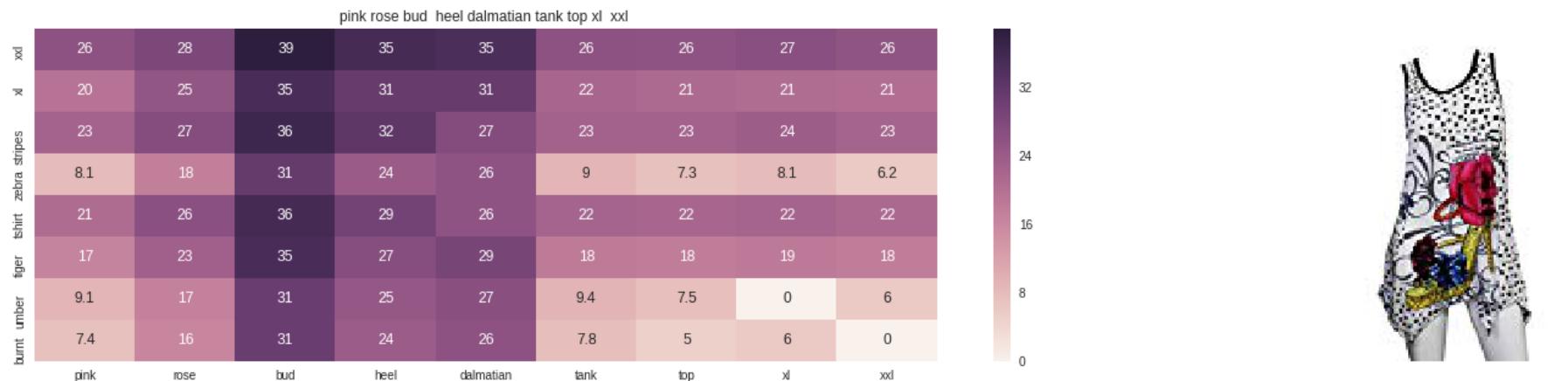
euclidean distance from input : 1.9806066342951432



ASIN : B00JV63VC8

Brand : Si Row

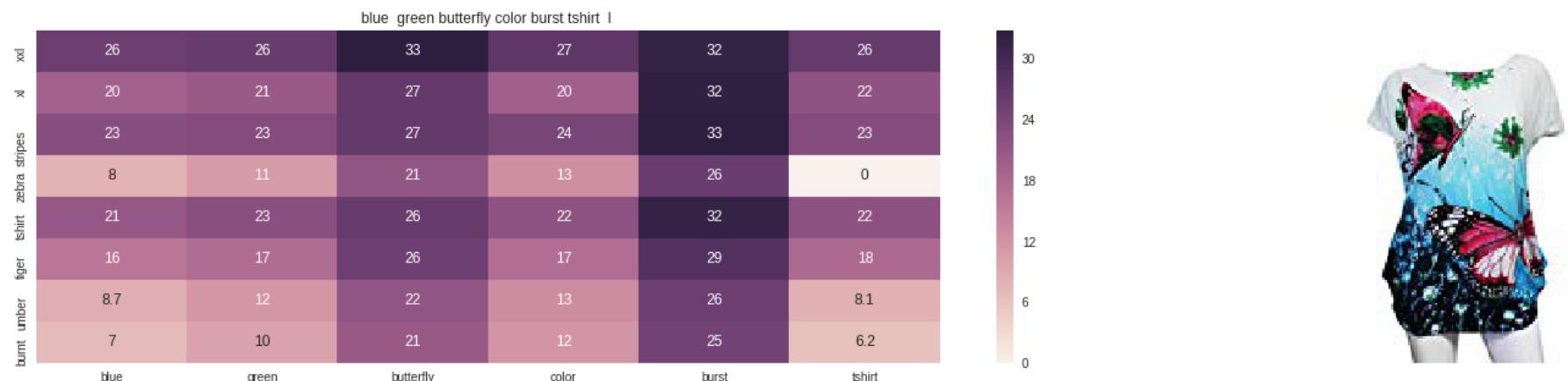
euclidean distance from input : 2.0121855999157043



ASIN : B00JXQAX2C

Brand : Si Row

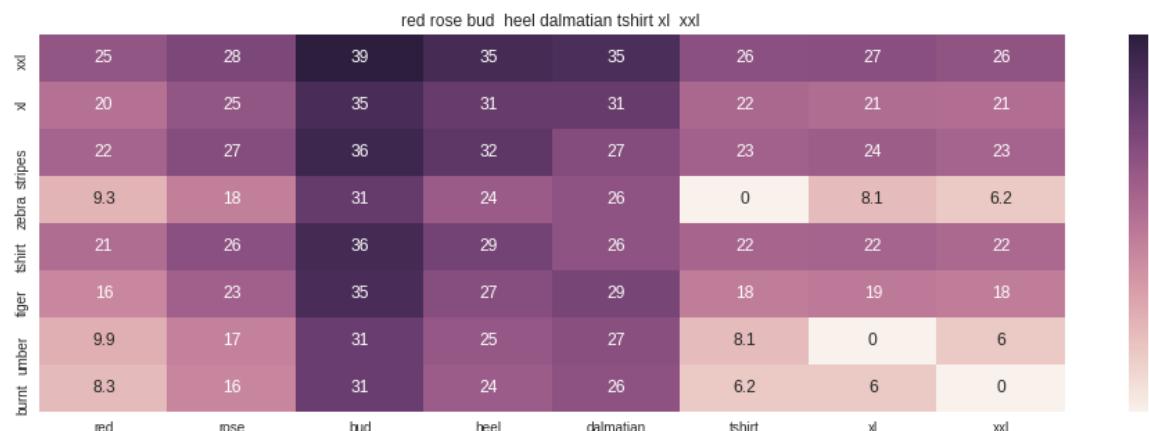
euclidean distance from input : 2.013351787548872



ASIN : B00JXQC0C8

Brand : Si Row

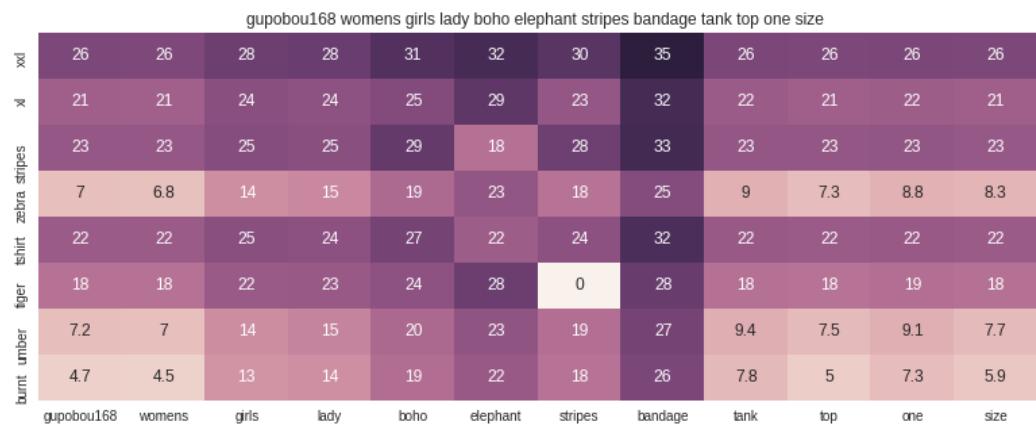
euclidean distance from input : 2.013883348273304



ASIN : B00JXQABB0

Brand : Si Row

euclidean distance from input : 2.0367257554998663



ASIN : B01ER18406

Brand : GuPoBoU168

euclidean distance from input : 2.6562041677776853

=====

=====



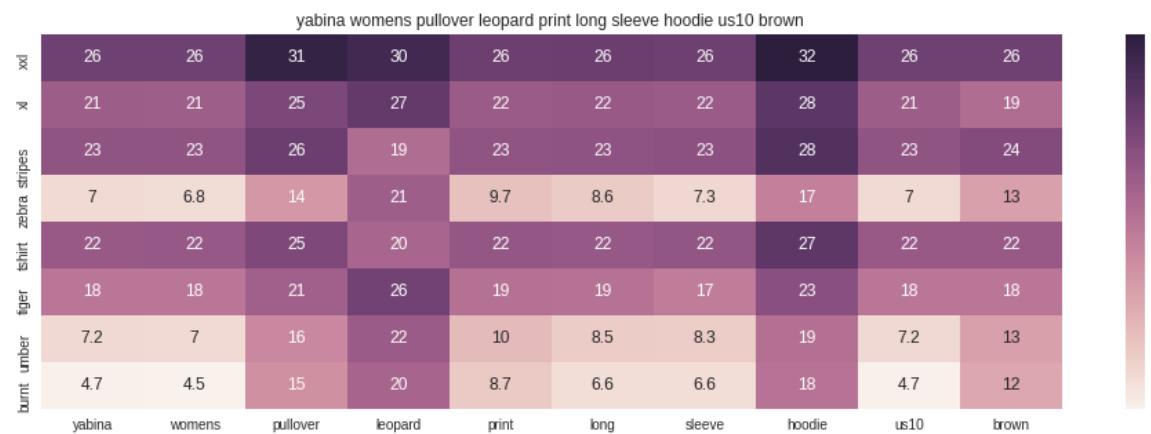
ASIN : B01LZ7BQ4H

Brand : WAYF

euclidean distance from input : 2.684906782301833

=====

=====

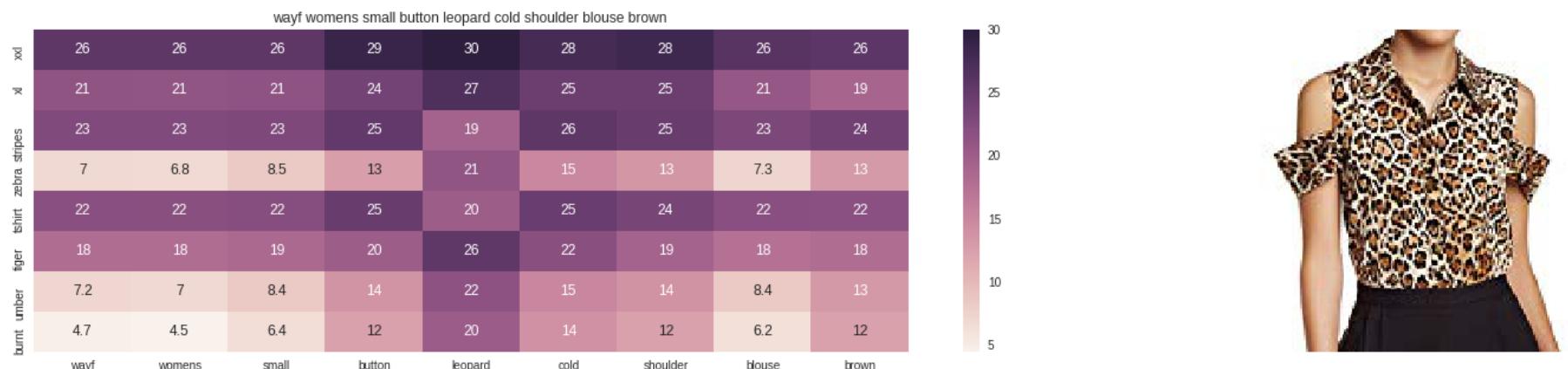


ASIN : B01KJUM6JI

Brand : YABINA

euclidean distance from input : 2.6858381926578345

=====



ASIN : B01M06V4X1

Brand : WAYF

euclidean distance from input : 2.694761948650377

[10.2] Keras and Tensorflow to extract features

```
In [46]: 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 [0]: # https://gist.github.com/fchollet/f35fbc80e066a49d65f1688a7e99f069
# Code reference: https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.htm
l

# 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()  
  
'''
```

[10.3] Visual features based product similarity.

```
In [47]: #Load the features and corresponding ASINS info.
bottleneck_features_train = np.load('16k_data_cnn_features.npy')
asins = np.load('16k_data_cnn_feature_asins.npy')
asins = list(asins)

# Load the original 16K dataset
data = pd.read_pickle('./pickels/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: 0.044194173
Amazon Url: www.amazon.com/dp/B00JXQB5FQ



Product Title: pink tiger tshirt zebra stripes xl xxl
Euclidean Distance from input image: 30.050056
Amazon Url: www.amazon.com/dp/B00JXQASS6



Product Title: yellow tiger tshirt tiger stripes 1

Euclidean Distance from input image: 41.261112

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



Product Title: brown white tiger tshirt tiger stripes xl xxl

Euclidean Distance from input image: 44.0002

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



Product Title: kawaii pastel tops tees pink flower design

Euclidean Distance from input image: 47.38251

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



Product Title: womens thin style tops tees pastel watermelon print
Euclidean Distance from input image: 47.71839
Amazon Url: www.amazon.com/dp/B01JUNHBRM



Product Title: kawaii pastel tops tees baby blue flower design
Euclidean Distance from input image: 47.9021
Amazon Url: www.amazon.com/dp/B071SBCY9W



Product Title: edv cheetah run purple multi xl

Euclidean Distance from input image: 48.046467

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



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

Euclidean Distance from input image: 48.101875

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



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

Euclidean Distance from input image: 48.118896

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



Product Title: miss chievous juniors striped peplum tank top medium shadowpeach
Euclidean Distance from input image: 48.13128
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: 48.25678

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



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

Euclidean Distance from input image: 48.265644

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



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

Euclidean Distance from input image: 48.362583

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



Product Title: chicago chicago 18 shirt women pink
Euclidean Distance from input image: 48.383648
Amazon Url: www.amazon.com/dp/B01GXAZTRY



Product Title: yichun womens tiger printed summer tshirts tops
Euclidean Distance from input image: 48.449345
Amazon Url: www.amazon.com/dp/B010NN9RX0



Product Title: nancy lopez whimsy short sleeve whiteblacklemon drop xs

Euclidean Distance from input image: 48.478893

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



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

Euclidean Distance from input image: 48.557983

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



Product Title: uswomens mary j blige without tshirts shirt

Euclidean Distance from input image: 48.614395

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

weighted euclidean distance similarity model with title, brand, color and image, where each feature can be modified by applying weights.

```
In [0]: def idf_w2v_one_hot_cnn_model(doc_id, result_count, wt, wb, wc, wi):  
  
    # pairwise_dist will store the distance from given input apparel to all remaining apparels for title feature  
    title_distance = pairwise_distances(title_features,title_features[doc_id].reshape(1,-1))  
  
    # pairwise_dist will store the distance from given input apparel to all remaining apparels for brand feature  
    brand_distance = pairwise_distances(brand_features,brand_features[doc_id])  
  
    # pairwise_dist will store the distance from given input apparel to all remaining apparels for color feature  
    color_distance = pairwise_distances(color_features,color_features[doc_id])  
  
    # pairwise_dist will store the distance from given input apparel to all remaining apparels for image feature  
    #image_distance = pairwise_distances(image_features,image_features[doc_id].reshape(1,-1))  
    image_distance = pairwise_distances(bottleneck_features_train, bottleneck_features_train[doc_id].reshape(1,-1))  
    # Weighted pairwise distance for all feature.  
    weighted_pairwise_dist = (wt * title_distance + wb * brand_distance + wc * color_distance + wi * image_distance)/float(wt + wb + wc + wi)  
  
    # Nearest indices of datapoints with respective to input doc_id  
    nearest_indices = np.argsort(weighted_pairwise_dist.flatten())[0:result_count]  
  
    # Nearest distance values of datapoints with respective to input doc_id  
    nearest_distances = np.sort(weighted_pairwise_dist.flatten())[0:result_count]  
  
    # List of nearest data points  
    list_nearest_data_points = list(data.index[nearest_indices])  
  
    # Just to print the output  
    for i in range(0,len(nearest_indices)):  
        display(Image(url=data['medium_image_url'].loc[list_nearest_data_points[i]], embed=True))  
        print('ASIN : ',data['asin'].loc[list_nearest_data_points[i]])  
        print('TITLE : ',data['title'].loc[list_nearest_data_points[i]])  
        print('BRAND : ',data['brand'].loc[list_nearest_data_points[i]])  
        print('COLOR : ',data['color'].loc[list_nearest_data_points[i]])  
        print('Euclidean Distance from input is  :',nearest_distances[i])  
        print('*125)
```



```
In [70]: #giving more weight to title  
idf_w2v_one_hot_cnn_model(12566,20,50,1,1,1)
```



ASIN : B00JXQB5FQ
TITLE : burnt umber tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 0.0

=====



ASIN : B00JXQASS6
TITLE : pink tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Pink
Euclidean Distance from input is : 2.5759735787633873

=====



ASIN : B00JXQCWTO
TITLE : brown white tiger tshirt tiger stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 3.3589129866551577

=====



ASIN : B00JXQCUIC
TITLE : yellow tiger tshirt tiger stripes 1
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 3.5071471724946033

=====



ASIN : B00JXQAFZ2
TITLE : grey white tiger tank top tiger stripes xl xxl
BRAND : Si Row
COLOR : Grey
Euclidean Distance from input is : 3.874353919845778



ASIN : B07568NZX4
TITLE : believed could tshirt
BRAND : Rustic Grace
COLOR : White
Euclidean Distance from input is : 3.8798706198706525



ASIN : B01K0H020G

TITLE : tpain tiger juniors tshirt size xlarge

BRAND : Tultex

COLOR : Black

Euclidean Distance from input is : 3.9470737438815893

=====

=====



ASIN : B06X99V6WC

TITLE : brunello cucinelli tshirt women white xl

BRAND : Brunello Cucinelli

COLOR : White

Euclidean Distance from input is : 3.957565044723635

=====

=====



ASIN : B01NB0NKRO
TITLE : ideology graphic tshirt xl white
BRAND : Ideology
COLOR : White
Euclidean Distance from input is : 3.9605419466266323

=====



ASIN : B074VMNZN9
TITLE : womens fashion beautiful tshirt
BRAND : simple
COLOR : Black
Euclidean Distance from input is : 3.994750765987832

=====



ASIN : B06XC3CZF6
TITLE : fjallraven womens ovik tshirt plum xxl
BRAND : Fjallraven
COLOR : Plum
Euclidean Distance from input is : 3.9970020131779114



ASIN : B00JXQAUWA
TITLE : yellow tiger tank top tiger stripes 1
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 4.018999808506064



ASIN : B0088PN0LA
TITLE : canvas 3001 30s tshirt kelly xl
BRAND : Red House
COLOR : Deep Red
Euclidean Distance from input is : 4.024932607561387
=====



ASIN : B00JXQABB0
TITLE : red rose bud heel dalmatian tshirt xl xxl
BRAND : Si Row
COLOR : Red
Euclidean Distance from input is : 4.0318590810130654
=====



ASIN : B00IAA4JIQ
TITLE : juniors love lucywaaaahhhh tshirt size xl
BRAND : I Love Lucy
COLOR : Purple
Euclidean Distance from input is : 4.057034859384049



ASIN : B005IT80BA
TITLE : hetalia us girl tshirt
BRAND : Hetalia
COLOR : Khaki
Euclidean Distance from input is : 4.058082010114638



ASIN : B00JXQBBMI

TITLE : yellow pink butterfly color burst tshirt xl xxl

BRAND : Si Row

COLOR : Yellow

Euclidean Distance from input is : 4.068996958279428

=====

=====



ASIN : B01CLS8LMW

TITLE : morning person tshirt troll picture xl

BRAND : Awake

COLOR : White

Euclidean Distance from input is : 4.0733783703464335

=====

=====



ASIN : B017X8PW9U
TITLE : diesel tserraf tshirt black
BRAND : Diesel
COLOR : Black
Euclidean Distance from input is : 4.086997847240596



ASIN : B00JXQC8L6
TITLE : blue peacock print tshirt 1
BRAND : Si Row
COLOR : Blue
Euclidean Distance from input is : 4.094486549482074

```
In [72]: #giving more weight to brand  
idf_w2v_one_hot_cnn_model(12566,20,1,50,1,1)
```



ASIN : B00JXQB5FQ
TITLE : burnt umber tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 0.0

=====



ASIN : B00JXQASS6
TITLE : pink tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Pink
Euclidean Distance from input is : 0.9746435868600859

=====



ASIN : B00JXQCUIC
TITLE : yellow tiger tshirt tiger stripes 1
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 1.0610752056613022



ASIN : B00JXQCWTO
TITLE : brown white tiger tshirt tiger stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 1.09429039427071



ASIN : B00JXQAUWA
TITLE : yellow tiger tank top tiger stripes 1
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 1.0953846132560523



ASIN : B00JXQAFZ2
TITLE : grey white tiger tank top tiger stripes xl xxl
BRAND : Si Row
COLOR : Grey
Euclidean Distance from input is : 1.1007690141854003



ASIN : B00JXQABB0
TITLE : red rose bud heel dalmatian tshirt xl xxl
BRAND : Si Row
COLOR : Red
Euclidean Distance from input is : 1.108243885763054



ASIN : B00JXQA094
TITLE : black white tiger tank top tiger stripes 1
BRAND : Si Row
COLOR : White
Euclidean Distance from input is : 1.1313236804557412



ASIN : B00JV63QQE
TITLE : red pink floral heel sleeveless shirt xl xxl
BRAND : Si Row
COLOR : Red
Euclidean Distance from input is : 1.1388508817488043

=====



ASIN : B00JXQBBMI
TITLE : yellow pink butterfly color burst tshirt xl xxl
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 1.145381763029416

=====



ASIN : B00JXQC0C8
TITLE : blue green butterfly color burst tshirt 1
BRAND : Si Row
COLOR : Blue
Euclidean Distance from input is : 1.1590096134612895



ASIN : B00JXQC8L6
TITLE : blue peacock print tshirt 1
BRAND : Si Row
COLOR : Blue
Euclidean Distance from input is : 1.1708713542320621



ASIN : B00JXQCFRS
TITLE : grey red peacock print tshirt l
BRAND : Si Row
COLOR : Grey
Euclidean Distance from input is : 1.1972379784593157

=====



ASIN : B00JV63CW2
TITLE : red butterfly black white tank top xl xxl
BRAND : Si Row
COLOR : Red
Euclidean Distance from input is : 1.2082538283649231

=====



ASIN : B00JV63VC8
TITLE : purple floral heel sleeveless shirt xl xxl
BRAND : Si Row
COLOR : Purple
Euclidean Distance from input is : 1.2092630558472874

=====



ASIN : B00JXQAX2C
TITLE : pink rose bud heel dalmatian tank top xl xxl
BRAND : Si Row
COLOR : Pink
Euclidean Distance from input is : 1.2093677329045396

=====



ASIN : B06XJYVM9Q

TITLE : woolrich womens size large buttondown flannel shirt deep iron

BRAND : W

COLOR : Deep Iron

Euclidean Distance from input is : 2.299164979521401

=====

=====



ASIN : B071XD3WT1

TITLE : alc womens jery silk top 6 black

BRAND : A.L.C.

COLOR : Black/Eggshell

Euclidean Distance from input is : 2.314169596951642

=====

=====



ASIN : B073GDVFT2

TITLE : joe elle womens black lacedinset peasant blouse top shirt small

BRAND : J&E

COLOR : Black

Euclidean Distance from input is : 2.3912527267971084

=====

=====



ASIN : B06ZY8FFRW

TITLE : alc tesi white linen cross back tee 1

BRAND : A.L.C.

COLOR : White

Euclidean Distance from input is : 2.402867727541622

=====

=====

```
In [73]: #giving more weight to color  
idf_w2v_one_hot_cnn_model(12566,20,1,1,50,1)
```



ASIN : B00JXQB5FQ
TITLE : burnt umber tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 0.0

=====



ASIN : B074MJN1K9
TITLE : hip latter crochet back womens small hilow blouse brown
BRAND : Hip
COLOR : Brown
Euclidean Distance from input is : 0.9681274218304653

=====



ASIN : B072BVB47Z

TITLE : h bordeaux white womens small striped tee shirt brown

BRAND : H By Bordeaux

COLOR : Brown

Euclidean Distance from input is : 0.9753476268840287

=====

=====



ASIN : B014OUHUZY

TITLE : leisure vest modal tank top loose condole belt large size backing shirtkhaki

BRAND : Black Temptation

COLOR : Brown

Euclidean Distance from input is : 1.0200680144534855

=====

=====



ASIN : B074QVMXSQ

TITLE : bellatrix women large petite sheer button blouse brown pl

BRAND : bellatrix

COLOR : Brown

Euclidean Distance from input is : 1.0610392680633294

=====

=====



ASIN : B071LDTQ1F

TITLE : hip small junior sheer printed button tank top 22 brown

BRAND : Hip

COLOR : Brown

Euclidean Distance from input is : 1.0621013660922884

=====

=====



ASIN : B0758356K3

TITLE : soprano womens small flawless asymmetric camisole top brown

BRAND : Soprano

COLOR : Brown

Euclidean Distance from input is : 1.0631825952277514

=====

=====



ASIN : B003SPYNAM

TITLE : marrikas viscose bamboo ring top mushroom small 68

BRAND : Marrikas

COLOR : Brown

Euclidean Distance from input is : 1.0645846764114189

=====

=====



ASIN : B017YBAI9A

TITLE : 1 world sarongs womens animal print tunic coverup small

BRAND : La Fleva

COLOR : Brown

Euclidean Distance from input is : 1.0683883810943027

=====

=====



ASIN : B074J7BCYM

TITLE : dark brown lao laos laotian sleeveless blouse classic neckline size 36 sl36f

BRAND : Nanon

COLOR : Brown

Euclidean Distance from input is : 1.0707089647031636

=====

=====



ASIN : B073ZCN5LG

TITLE : brunello cucinelli top womens brown slim fit cotton casual xs

BRAND : Brunello Cucinelli

COLOR : Brown

Euclidean Distance from input is : 1.0728456605360066

=====

=====



ASIN : B00BTJKAQ0

TITLE : annakaci sm fit brown three fashion friends graphic print paisley panel top

BRAND : Anna-Kaci

COLOR : Brown

Euclidean Distance from input is : 1.075467927161696

=====

=====



ASIN : B01CE40W16

TITLE : marolaya womens caftan poncho tunic plus size sexy tassel cover

BRAND : Marolaya

COLOR : Brown

Euclidean Distance from input is : 1.078909426072573

=====

=====



ASIN : B072M4ZF89

TITLE : abound womens medium striped pocketfront knit top 24 brown

BRAND : Abound

COLOR : Brown

Euclidean Distance from input is : 1.0817406224679949

=====

=====



ASIN : B01KJUM6JI

TITLE : yabina womens pullover leopard print long sleeve hoodie us10 brown

BRAND : YABINA

COLOR : Brown

Euclidean Distance from input is : 1.0863957334686518

=====

=====



ASIN : B074P8YWV4

TITLE : bobeau womens small petite striped tank cami top brown ps

BRAND : Bobeau

COLOR : Brown

Euclidean Distance from input is : 1.091843624991905

=====

=====



ASIN : B016MGC5VW

TITLE : vogue code bows print long sleeve base shirt loose fit plus size blouse

BRAND : VOGUE CODE

COLOR : Brown

Euclidean Distance from input is : 1.0920194750356498

=====

=====



ASIN : B00JXQCWT0

TITLE : brown white tiger tshirt tiger stripes xl xxl

BRAND : Si Row

COLOR : Brown

Euclidean Distance from input is : 1.09429039427071

=====

=====



ASIN : B07288KFHF

TITLE : soprano olive large junior cropped ribbed knit top 18 brown 1

BRAND : Soprano

COLOR : Brown

Euclidean Distance from input is : 1.0954410284642297

=====

=====



ASIN : B00MJPVIDW

TITLE : new ladies plus size long batwing sleeve shoulder baggy tops brown 1x

BRAND : Xclusive Collection

COLOR : Brown

Euclidean Distance from input is : 1.0991811649961227

=====

=====

```
In [74]: #giving more weight to cnn_model  
idf_w2v_one_hot_cnn_model(12566,20,1,1,1,50)
```



ASIN : B00JXQB5FQ
TITLE : burnt umber tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 0.0

=====



ASIN : B06XBHNM7J
TITLE : womens crochet trim shirts olive tree large xhilaration
BRAND : Xhilaration
COLOR : Olive Tree
Euclidean Distance from input is : 35.22367869350496

=====



ASIN : B016CU40IY

TITLE : breast cancer awareness juniors vneck shirt fight cancer

BRAND : Juiceclouds

COLOR : Black

Euclidean Distance from input is : 36.28389388226766

=====

=====



ASIN : B074LTBWSW

TITLE : completely liz lange long flyaway vest 249682 turquoise

BRAND : Liz Lange

COLOR : Turquoise

Euclidean Distance from input is : 37.27591400147166

=====

=====



ASIN : B074MXY984

TITLE : free people free malibu thermal henley pullover black small

BRAND : We The Free

COLOR : Black

Euclidean Distance from input is : 37.28713979243469

=====

=====



ASIN : B018H5AZXQ

TITLE : buffalo david bitton nipaw logo graphic tank white combo xxl

BRAND : Buffalo

COLOR : White Combo

Euclidean Distance from input is : 37.46353764471368

=====

=====



ASIN : B06XYP1X1F
TITLE : j brand womens pinstripe shirt xs blue
BRAND : J Brand Jeans
COLOR : Navy/Blk Stp
Euclidean Distance from input is : 37.49566225980352

=====



ASIN : B01BMSFYW2
TITLE : tommy hilfiger graphic lounge cami white cloud dancer xlarge
BRAND : igertommy hilf
COLOR : white cloud dancer
Euclidean Distance from input is : 37.637816188101276

=====



ASIN : B01FJVZST2
TITLE : kongyii womens charlotte hornets à sport pique polo
BRAND : KONGYII
COLOR : White
Euclidean Distance from input is : 38.554666027810605



ASIN : B01L7ROZNC
TITLE : bila size small womens sleeveless blouse red
BRAND : Bila
COLOR : Red
Euclidean Distance from input is : 38.84605707660188



ASIN : B01L9F153U
TITLE : girls fairy tail exceed tee shirts black
BRAND : ATYPEMX
COLOR : Black
Euclidean Distance from input is : 39.365090832498105

=====



ASIN : B01EXXFS4M
TITLE : boundaries juniors 3/4 sleeve space dye hi lo knit top pink medium
BRAND : No Boundaries
COLOR : Pink
Euclidean Distance from input is : 39.47907064242685

=====



ASIN : B0756JTS1F

TITLE : salvatore ferragamo geometric print silk top 42 6

BRAND : Salvatore Ferragamo

COLOR : Multi-color

Euclidean Distance from input is : 39.64422363232361

=====

=====



ASIN : B06Y41MRCH

TITLE : byoung womens henya womens light blue shirt size 401 light blue

BRAND : Byoung

COLOR : Chambray Blue

Euclidean Distance from input is : 39.73244011744364

=====

=====



ASIN : B074MK6LV2

TITLE : 1state womens medium chambray crochet solid blouse blue

BRAND : 1.State

COLOR : Blue

Euclidean Distance from input is : 39.80414439701541

=====

=====



ASIN : B074Z5C98D

TITLE : sexy sheer mesh print long sleeves bodysuit

BRAND : Ariella's closet

COLOR : Multi Color Black & Pink

Euclidean Distance from input is : 39.82058277851335

=====

=====



ASIN : B01M8GB3AL

TITLE : maven west striped sleeveless lace peplum peasant blouse yellow large

BRAND : Maven West

COLOR : Yellow

Euclidean Distance from input is : 40.07541860907061

=====

=====



ASIN : B00DP4VHWI

TITLE : stanzino womens long sleeve graphic print plus size top fuchsia xl

BRAND : Stanzino

COLOR : Fuchsia

Euclidean Distance from input is : 40.114904276154434

=====

=====



ASIN : B01G7XE50E

TITLE : womens ultimate scoop tee fresh white xl merona

BRAND : Merona

COLOR : White

Euclidean Distance from input is : 40.133908495704546

=====

=====



ASIN : B00JMAASRO

TITLE : hot sexy fashion women loose chiffon short sleeve tops blouse shirt

BRAND : Wotefusi

COLOR : Multicolor

Euclidean Distance from input is : 40.146927278025615

=====

=====

In [77]: *#giving same weights to brand and cnn_model*
idf_w2v_one_hot_cnn_model(12566,20,10,30,10,5)



ASIN : B00JXQB5FQ
TITLE : burnt umber tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 0.0

=====



ASIN : B00JXQASS6
TITLE : pink tiger tshirt zebra stripes xl xxl
BRAND : Si Row
COLOR : Pink
Euclidean Distance from input is : 4.982033973689734

=====



ASIN : B06XBHNM7J

TITLE : womens crochet trim shirts olive tree large xhilaration

BRAND : Xhilaration

COLOR : Olive Tree

Euclidean Distance from input is : 5.367656713033046

=====

=====



ASIN : B016CU40IY

TITLE : breast cancer awareness juniors vneck shirt fight cancer

BRAND : Juiceclouds

COLOR : Black

Euclidean Distance from input is : 5.456287301550389

=====

=====



ASIN : B00JXQCUIC
TITLE : yellow tiger tshirt tiger stripes 1
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 5.481540910049749



ASIN : B00JXQCWTO
TITLE : brown white tiger tshirt tiger stripes xl xxl
BRAND : Si Row
COLOR : Brown
Euclidean Distance from input is : 5.495170897835578



ASIN : B018H5AZXQ

TITLE : buffalo david bitton nipaw logo graphic tank white combo xxl

BRAND : Buffalo

COLOR : White Combo

Euclidean Distance from input is : 5.583497832972677

=====

=====



ASIN : B01FJVZST2

TITLE : kongyii womens charlotte hornets à sport pique polo

BRAND : KONGYII

COLOR : White

Euclidean Distance from input is : 5.608569285376071

=====

=====



ASIN : B01L7ROZNC

TITLE : bila size small womens sleeveless blouse red

BRAND : Bila

COLOR : Red

Euclidean Distance from input is : 5.6366487864414125

=====

=====



ASIN : B074LTBWSW

TITLE : completely liz lange long flyaway vest 249682 turquoise

BRAND : Liz Lange

COLOR : Turquoise

Euclidean Distance from input is : 5.653873790459881

=====

=====



ASIN : B01L9F153U
TITLE : girls fairy tail exceed tee shirts black
BRAND : ATYPEMX
COLOR : Black
Euclidean Distance from input is : 5.686664600095532



ASIN : B00JXQAUWA
TITLE : yellow tiger tank top tiger stripes 1
BRAND : Si Row
COLOR : Yellow
Euclidean Distance from input is : 5.693806953459245



ASIN : B00JXQAFZ2
TITLE : grey white tiger tank top tiger stripes xl xxl
BRAND : Si Row
COLOR : Grey
Euclidean Distance from input is : 5.704997392199938



ASIN : B06XYP1X1F
TITLE : j brand womens pinstripe shirt xs blue
BRAND : J Brand Jeans
COLOR : Navy/Blk Stp
Euclidean Distance from input is : 5.733989531082479



ASIN : B01G7XE50E
TITLE : womens ultimate scoop tee fresh white xl merona
BRAND : Merona
COLOR : White
Euclidean Distance from input is : 5.737111604719133

=====



ASIN : B074MK6LV2
TITLE : 1state womens medium chambray crochet solid blouse blue
BRAND : 1.State
COLOR : Blue
Euclidean Distance from input is : 5.751836451300007

=====



ASIN : B00JXQABB0

TITLE : red rose bud heel dalmatian tshirt xl xxl

BRAND : Si Row

COLOR : Red

Euclidean Distance from input is : 5.755765405163791

=====

=====



ASIN : B00DP4VHWI

TITLE : stanzino womens long sleeve graphic print plus size top fuchsia xl

BRAND : Stanzino

COLOR : Fuchsia

Euclidean Distance from input is : 5.803941193954922

=====

=====



ASIN : B06XK2ZRFH

TITLE : acquaa womens long sleeve stripe pocket fashion tshirt picture

BRAND : Acqua

COLOR : Y#4

Euclidean Distance from input is : 5.805637655782668

=====

=====



ASIN : B01HT0PRMY

TITLE : scally crop top grey large

BRAND : Lushfox

COLOR : Grey

Euclidean Distance from input is : 5.813499183202551

=====

=====

CONCLUSION:

- With giving more weight to the brand we can get very similar items.
- But which model to use is strictly a business decision.

PROCEDURE:

1. Read the data from tops_fashion.json and load into dataframe.
2. Analyzing the data and understand it.
3. Cleaning the data i.e., removing duplicates, removing the rows which don't have brand,color,price and removing the titles which have less than 3 words.
4. The final size of the data we get is 16k.
5. Removing stopwors for the titles.
6. Finally performed similarity techniques.