In [32]:

```
#import all the necessary packages.

import PIL.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
import pickle
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import Dropout, Flatten, Dense
from keras import applications
from gensim.models import Word2Vec
from gensim.models import KeyedVectors
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
from IPython.display import display, Image, SVG, Math, YouTubeVideo

import pickle
plotly.offline.init_notebook_mode(connected=True)
warnings.filterwarnings("ignore")
data = pd.read_pickle('pickels/16k_apperal_data_preprocessed')
data.head()
# vocab = stores all the words that are there in google w2v model
# vocab = model.wv.vocab.keys() # if you are using Google word2Vec
with open('word2vec_model', 'rb') as handle:
    model = pickle.load(handle)
vocab = model.keys()
idf_title_vectorizer = CountVectorizer()
idf_title_features = idf_title_vectorizer.fit_transform(data['title'])

#load the features and corresponding ASINS info.
# 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 hypen
```

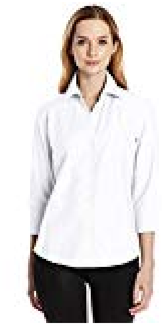
```
brands = [x.replace(" ", "-") for x in data['brand'].values]
types = [x.replace(" ", "-") for x in data['product_type_name'].values]
colors = [x.replace(" ", "-") for x in data['color'].values]

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
()
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_apperal_data_preprocessed')
df_asins = list(data['asin'])


from IPython.display import display, Image, SVG, Math, YouTubeVideo
# this function will add the vectors of each word and returns the avg vector o
f given sentance
def build_avg_vec(sentence, num_features, doc_id, m_name):
    # sentace: its title of the apparel
    # num_features: the lenght of word2vec vector, its values = 300
    # m_name: model information it will take two values
        # if  m_name == 'avg', we will append the model[i], w2v representation
of word i
        # if m_name == 'weighted', we will multiply each w2v[word] with the id
f(word)

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

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

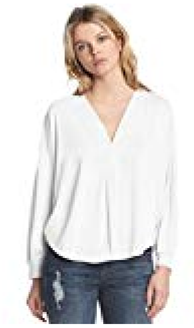
In [34]:

```
# Utility functions

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

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 id
f(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.voc
abulary_[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 c
orpus, 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/a
vg) 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 vect
ors in vec2
    for i in vec1:
        dist = []
        for j in vec2:
            # np.linalg.norm(i-j) will result the euclidean distance between v
ectors 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 t
itle2)
    # final_dist[i,j] = euclidean distance between vectors i, j
    return np.array(final_dist)
```

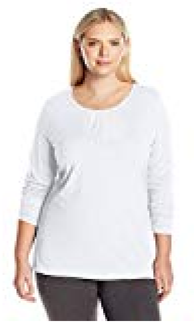
```python
def heat_map_w2v_brand(sentance1, sentance2, url, doc_id1, doc_id2, df_id1, df
_id2, model):

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

    # s1_s2_dist = np.array(#number of words in title1 * #number of words in t
itle2)
    # 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], typ
es[doc_id1]], # input apparel's features
               [data['asin'].loc[df_id2],brands[doc_id2], colors[doc_id2], typ
es[doc_id2]]] # recommonded apparel's features

    colorscale = [[0, '#1d004d'],[.5, '#f2e5ff'],[1, '#f2e5d1']] # to color th
e 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])
    # ploting 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 lins 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()


def idf_w2v_brand_img(doc_id, w_text , w_brand, w_color,w_img, 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 re
maining apparels
    # the metric we used here is cosine, the coside distance is mesured as K
(X, Y) = <X, Y> / (||X||*||Y||)
    # http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity
    idf_w2v_dist  = pairwise_distances(w2v_title_weight, w2v_title_weight[doc_
id].reshape(1,-1))
    pair_img_dist = pairwise_distances(bottleneck_features_train, bottleneck_f
eatures_train[doc_id].reshape(1,-1))
    brand_dist = pairwise_distances(brand_features, brand_features[doc_id])
    color_dist = pairwise_distances(color_features, color_features[doc_id])
    pairwise_dist   = (w_text * idf_w2v_dist +  w_brand * brand_dist+ w_color*
color_dist+w_img*pair_img_dist)/float(w_text + w_brand+ w_color+w_img)

    # 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(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], indi
ces[i],df_indices[0], df_indices[i], 'weighted')
        rows = data[['medium_image_url','title']].loc[data['asin']==asins[indi
ces[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.amzon.com/dp/'+ asins[indices[i]])
            print('='*125)

idf_w2v_brand_img(12566, 5, 5,5, 5,20)

# in the give heat map, each cell contains the euclidean distance between word
s i, j
```



```
Product Title:  foxcroft nyc womens pinpoint oxford shirt noniron stretch pop
lin blouse xlarge white
Euclidean Distance from input image: 0.011221176944673061
Amazon Url: www.amzon.com/dp/B072277HVB
=============================================================================
================================================
```



```
Product Title:  kiind longsleeve swing top white
Euclidean Distance from input image: 10.51571851138942
Amazon Url: www.amzon.com/dp/B0142Q3C6G
=============================================================================
================================================
```



```
Product Title:  size comfortblend shirred crewneck longsleeve tshirt white 2x
Euclidean Distance from input image: 10.961295725816793
Amazon Url: www.amzon.com/dp/B01N5YHV9R
=============================================================================
================================================
```

```
Product Title:  karen scott womens plus short sleeve solid henley top white 1
x
Euclidean Distance from input image: 11.031367397398693
Amazon Url: www.amzon.com/dp/B0196H3Z2W
=============================================================================
================================================
```

```
Product Title:  free people scoopneck highlow thermal tunic oatmeal heather
Euclidean Distance from input image: 11.048943094061539
Amazon Url: www.amzon.com/dp/B00ZH2X45E
=============================================================================
================================================
```

```
Product Title:  karen scott wmen short sleeve henley top plus size 0x bright
white
Euclidean Distance from input image: 11.164481452930868
Amazon Url: www.amzon.com/dp/B0196H7P5A
=============================================================================
================================================
```

```
Product Title:  masion jules longsleeve paneled sweatshirt xxl grey
Euclidean Distance from input image: 11.226435160636902
Amazon Url: www.amzon.com/dp/B06X16WTFX
=============================================================================
================================================
```

```
Product Title:  vitamina usa soft vneckline belted blouse st6356 wht
Euclidean Distance from input image: 11.266915869712829
Amazon Url: www.amzon.com/dp/B0155ICSDS
=============================================================================
================================================
```

```
Product Title:  dg2 diane gilman turtleneck fleece pullover pockets small gra
y
Euclidean Distance from input image: 11.3570208806935
Amazon Url: www.amzon.com/dp/B01MSP0KAM
=============================================================================
================================================
```

```
Product Title:  jessica simpson womens frida peasant top antiquewhite x small
Euclidean Distance from input image: 11.41253459643752
Amazon Url: www.amzon.com/dp/B01KW297J0
=============================================================================
================================================
```

```
Product Title:  red house  ladies nailhead noniron buttondown shirt3xl white
Euclidean Distance from input image: 11.532396771425313
Amazon Url: www.amzon.com/dp/B008LOSLYO
=============================================================================
================================================
```

```
Product Title:  american living womens dipdye peplum peplum top red l
Euclidean Distance from input image: 11.610639381499036
Amazon Url: www.amzon.com/dp/B01LYFIRIL
=============================================================================
================================================
```

```
Product Title:  jenni kayne womens crepe tee white small
Euclidean Distance from input image: 11.66481273364455
Amazon Url: www.amzon.com/dp/B07531DL6X
=============================================================================
================================================
```

```
Product Title:  bb dakota womens plus size tereza top optic white 2x
Euclidean Distance from input image: 11.700085618013448
Amazon Url: www.amzon.com/dp/B00JG6EBEA
=============================================================================
================================================
```

```
Product Title:  jm collection womens bright white top size xl
Euclidean Distance from input image: 11.719918562883443
Amazon Url: www.amzon.com/dp/B01N4NYF25
=============================================================================
================================================
```

```
Product Title:  isaac mizrahi woven tunic multicolored soutache a223828 pink
purple
Euclidean Distance from input image: 11.752072932147346
Amazon Url: www.amzon.com/dp/B074HFNCWK
=============================================================================
================================================
```

```
Product Title:  robert graham womens blouse trish lavender small
Euclidean Distance from input image: 11.777045180315083
Amazon Url: www.amzon.com/dp/B071NQ52LM
=============================================================================
================================================
```

```
Product Title:  view walter baker long slv knit top a263076 white
Euclidean Distance from input image: 11.810799602316543
Amazon Url: www.amzon.com/dp/B074Q572HW
=============================================================================
================================================
```

```
Product Title:  hengsong women casual half sleeve sexy v lace splicing croche
t tops blouse shirts
Euclidean Distance from input image: 11.832138824553235
Amazon Url: www.amzon.com/dp/B01DXPBBLK
=============================================================================
================================================
```

```
Product Title:  tuxe rookie bodysuit b1421ixs
Euclidean Distance from input image: 11.840807223410351
Amazon Url: www.amzon.com/dp/B00M7DFH9C
============================================================================
===============================================
```

summary:

1.Used Text, brand, color and image features to recommend similar products.

2.Given weights feature for all 4 features and played around with it, to see how image recommendations varry based on features.

```
In [ ]:
```