Feature engineering

I Use Keras' Pre-trained VGG16 Models for Feature Extraction in Image Clustering(reference https://keras.io/applications/#vgg16):

1. set the output layer as the second-to-last fully connected layer 'fc2' (see picture 1) of shape 4096.

```
base_model = VGG16(weights='imagenet', include_top=True)
model = Model(inputs=base_model.input,
              outputs=base_model.get_layer('fc2').output)
```

2. for each image resizes it into 244 * 244 *3 as the model input required
3. by using VGG16 Model, get the features of each image, which is an array of shape 4096.
4. use features of all images to do hierarchical clustering (use Euclidean distance metric and similar images are put together in a cluster).
5. use the similarity index sim=0...1 to define the height at which we cut through the dendogram tree built by the hierarchical clustering. sim=0 is the root of the dendogram where there is only one node (that means all images in one cluster). sim=1 is equal to the top of the dendogram tree, where each image is its own cluster. By varying the index between 0 and 1, increase the number of clusters from 1 to the number of images.
6. By a lot of tests, I choose sim=0.6 to get all the clusters

Picture 1

```
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
.
.
.
block5_pool (MaxPooling2D)   (None, 7, 7, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 25088)             0
_________________________________________________________________
fc1 (Dense)                  (None, 4096)              102764544
_________________________________________________________________
fc2 (Dense)                  (None, 4096)              16781312
_________________________________________________________________
predictions (Dense)          (None, 1000)              4097000
=================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0
_________________________________________________________________
```