Artificial Intelligence and Machine Learning Fundamentals

Activity 13: Shape Recognition with the Mean Shift Algorithm

In this section, we will learn how images can be clustered. Imagine you are working for a company that detects human emotions from photos. Your task is to extract pixels making up a face in an avatar photo.
Create a clustering algorithm with Mean Shift to cluster pixels of images. Examine the results of the Mean Shift algorithm and check whether any of the clusters contain a face when used on avatar images.
Then, apply the k-means, algorithm with a fixed default number of clusters (8, in this case). Compare your results with the Mean Shift clustering algorithm:

1. Select an image you would like to cluster and load the image.
2. We chose this image from the Author's Youtube channel:

3. The image size has been significantly reduced so that our algorithm would terminate more quickly.

```
image = Image.open('destructuring.jpg')
pixels = image.load()
```

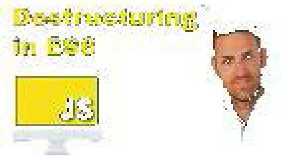
4. Transform the pixels into a data frame to perform clustering

```
import pandas
data_frame = pandas.DataFrame(
[[x,y,pixels[x,y][0], pixels[x,y][1],
pixels[x,y][2]]
for x in range(image.size[0])
for y in range(image.size[1])
],
columns=['x', 'y', 'r', 'g', 'b']
)
```

5. Perform Mean Shift clustering on the image using scikit-learn. Note that this time we will skip normalization of the features, because proximity of the pixels and proximity of color components are represented in close to equal weight. The largest difference in pixels distance is 750, while the largest difference in a color component is 256.

```
from sklearn.cluster import MeanShift
mean_shift_model = MeanShift()
mean_shift_model.fit(data_frame)
for i in
range(len(mean_shift_model.cluster_centers_)):
image = Image.open('destructuring.jpg')
pixels = image.load()
for j in range(len(data_frame)):
if (mean_shift_model.labels_[j] != i ):
pixels[ int(data_frame['x'][j]),
int(data_frame['y'][j]) ] = (255, 255, 255)
image.save( 'cluster' + str(i) + '.jpg' )
```

6. The algorithm found the following two clusters:



7. The Mean Shift algorithm treated my skin and the yellow JavaScript and Destructuring text close enough to each other to form the same cluster.

8. Let's use the k-means algorithm to formulate eight clusters on the same data.

```
k_means_model = KMeans(n_clusters=8)
k_means_model.fit(data_frame)
for i in range(len(k_means_model.cluster_centers_)):
image = Image.open('destructuring.jpg')
pixels = image.load()
for j in range(len(data_frame)):
if (k_means_model.labels_[j] != i):
pixels[int(data_frame['x'][j]),
int(data_frame['y'][j])] =
(255, 255, 255)
image.save('kmeanscluster' + str(i) + '.jpg')
```

9. The 8 clusters are the following:

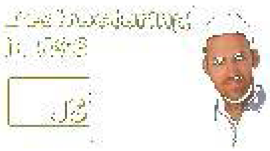
The output for the first is as follows:

The output for the second is as follows:

The output for the third is as follows:

The output for the fourth is as follows:

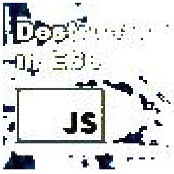
The output for the fifth is as follows:



The output for the sixth is as follows:

The output for the seventh is as follows:

The output for the eighth is as follows:



As you can see, the fifth cluster recognized my face quite well. The clustering algorithm indeed located data points that are close and contain similar colors.