Capstone Project Report

Image segmentation using KMeans

Name: Polluri Kodanda Rama Durgarao
Course: AI & ML (Batch - 4)

Problem Statement
Take a bright colorful image (Eg: image having fruits in it) and implement image segmentation using K-Means. You can first try to implement K-Means on iris dataset to understand its working and then extend the same logic, using the image pixels as the data points. Hint: All the K centroids will represent a color and therefore, you can initialize all the pixels to belong to a cluster randomly and then start the training of the centroids.

Prerequisites
Along with Python below packages needed to be installed

Matplotlib
Numpy
PIL
Sklearn

Dataset Used
Test image of fruits

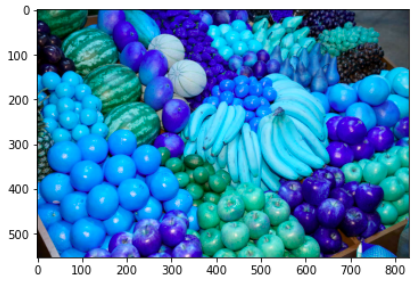
Implementation
Import required libraries and load data

```
In [1]: import cv2
        import matplotlib.pyplot as plt
        import numpy as np
        from sklearn.cluster import KMeans
```

```
In [2]: original = cv2.imread('fruits.jpeg')
```

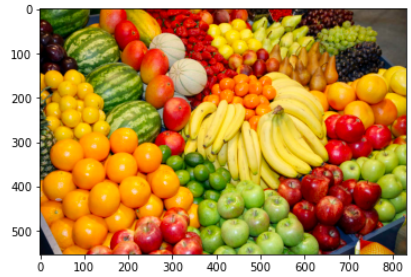
Visualize the image loaded

```
In [4]: plt.imshow(original)
Out[4]: <matplotlib.image.AxesImage at 0x7fcd63ba8160>
```



Convert to RGB colored image

```
In [6]: img=cv2.cvtColor(original,cv2.COLOR_BGR2RGB)
        plt.imshow(img)
Out[6]: <matplotlib.image.AxesImage at 0x7fcd64190910>
```



Vectorize and apply KMeans model

```
In [7]: vectorized = img.reshape((-1,3))
        print(vectorized.shape)
        print(vectorized[0])

        (458990, 3)
        [45 67  3]

In [8]: kmeans = KMeans(n_clusters=5)
        kmeans.fit(vectorized)
Out[8]: KMeans(n_clusters=5)
```

Use the clusters that are created by KMeans for image segmentation

In [11]:
```
center = np.uint8(kmeans.cluster_centers_)
res = center[kmeans.labels_.flatten()]
result_image = res.reshape(img.shape)
print(center[1])
print(kmeans.labels_[1])
print(center[kmeans.labels_[1]])
print(kmeans.labels_[1].flatten())
print(center[[3]])
print(center[3].shape)
print(center[[3]].shape)
print(res.shape)
```

```
[132 135  57]
2
[59 38 21]
[2]
[[223 199 113]]
(3,)
(1, 3)
(458990, 3)
```

In [12]:
```
plt.imshow(result_image)
```

Out[12]:
```
<matplotlib.image.AxesImage at 0x7fcd64630b80>
```