CS498 AML HW8

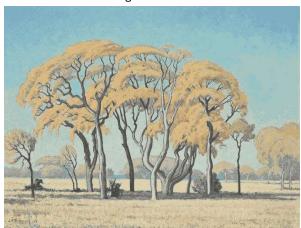
Yidi Yang (yyang160) Huiyun Wu (hwu63)

Image 1, tree.jpg.

Segments = 10



Segments = 20



Segments = 50



Image 2, RobertMixed03.jpg

Segments = 10



Segments = 20



Segments = 50



Image 3, smallstrelitzia.jpg.

Segments = 10



Segments = 20



Segments = 50



Image 4, smallsunset.jpg.

Segments = 10



Segments = 20



Segments = 50



Tree image with 20 segments with 5 different initial points (5 different random state number in the K-means)

The difference is hardly observed, but they are indeed different.











Code Snippets

EM initialization (with random numbers as the random state in KMeans)

```
11
          def segment(img, number):
12
               rs = np.random.randint(0, 100)
13
               kmeans = KMeans(n_clusters=number, random_state=rs).fit(img)
14
               print(kmeans.cluster_centers_)
               return kmeans.cluster_centers_
15
EM Updates (E-Step, M-Step, and find cluster based on results)
        def EM(img, number, mus):
17
18
19
             pis = np.zeros((number,))
             for i in range(number):
20
21
                 pis[i] = 1/number
22
             w = np.zeros((len(img), number))
                                                     # number of iteration
23
             it = 0
24
             converge = False
25
             ##### E Step #######
26
             while(not converge):
27
                 it += 1
                 print("Iteration: ", it)
28
29
                 for px in range(len(img)):
30
                     denom = 0
                     for k in range(number):
31
32
                         distk = img[px] - mus[k]
33
                         denom += np.exp(-0.5 * np.dot(distk, distk) * pis[k])
34
                     wj = np.zeros((number, ))
35
                     for k in range(number):
                         distk = img[px] - mus[k]
36
                         wj[k] = np.exp(-0.5 * np.dot(distk, distk)) * pis[k]
37
38
                     if denom == 0:
                         denom = 0.1
39
40
                     w[px, :] = wj/denom
42
                  ###### M Step #######
43
                newmus = np.zeros((number, 3))
44
                newpis = np.zeros((number, ))
45
                 for j in range(number):
46
                    nom = 0
47
                    # total = 0
                    total = np.sum(w[:, j])
48
49
                    for px in range(len(img)):
50
                        nom += img[px] * w[px, j]
                        # total += w[px, j]
51
                    newmus[j] = nom/total
52
53
                    newpis[j] = total/len(img)
54
                diff = np.abs(np.sum(mus) - np.sum(newmus))
                print("diff: ", diff)
55
                if(diff < 0.1):
56
57
                    converge = True
58
                mus = newmus
59
                pis = newpis
60
             print("===
                            ==Result=======")
61
             print(mus)
62
             print(pis)
63
             return mus, pis
64
        def findCluster(value, mus):
65
66
            min = math.inf
67
             cluster = None
68
             for i, mu in enumerate(mus):
                dist = np.linalg.norm(value-mu)
69
70
                 if dist < min:</pre>
71
                    cluster = mu
                    min = dist
72
73
             return cluster
```