Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Компьютерных сетей и систем

Кафедра Информатики

ЛАБОРАТОРНАЯ РАБОТА №6
«Кластеризация»

БГУИР 1-40 81 04

Магистрант:
гр. 858642
Кукареко А.В.

Проверил:
Стержанов М. В.

Минск, 2019

ХОД РАБОТЫ

Задание.

Набор данных ex6data1.mat представляет собой файл формата *.mat (т.е. сохраненного из Matlab). Набор содержит две переменные X1 и X2 - координаты точек, которые необходимо кластеризовать.

Набор данных bird_small.mat представляет собой файл формата *.mat (т.е. сохраненного из Matlab). Набор содержит массив размером (16384, 3) - изображение 128x128 в формате RGB.

1. Загрузите данные ex6data1.mat из файла.
2. Реализуйте функцию случайной инициализации K центров кластеров.
3. Реализуйте функцию определения принадлежности к кластерам.
4. Реализуйте функцию пересчета центров кластеров.
5. Реализуйте алгоритм K-средних.
6. Постройте график, на котором данные разделены на K=3 кластеров (при помощи различных маркеров или цветов), а также траекторию движения центров кластеров в процессе работы алгоритма
7. Загрузите данные bird_small.mat из файла.
8. С помощью алгоритма K-средних используйте 16 цветов для кодирования пикселей.
9. Насколько уменьшился размер изображения? Как это сказалось на качестве?
10. Реализуйте алгоритм K-средних на другом изображении.
11. Реализуйте алгоритм иерархической кластеризации на том же изображении. Сравните полученные результаты.
12. Ответы на вопросы представьте в виде отчета..

Результат выполнения:

1. Загрузите данные ex6data1.mat из файла.

```
data1 = scipy.io.loadmat('ex6data1.mat')
X1 = data1['X']
X1.shape
```

```
(50, 2)
```

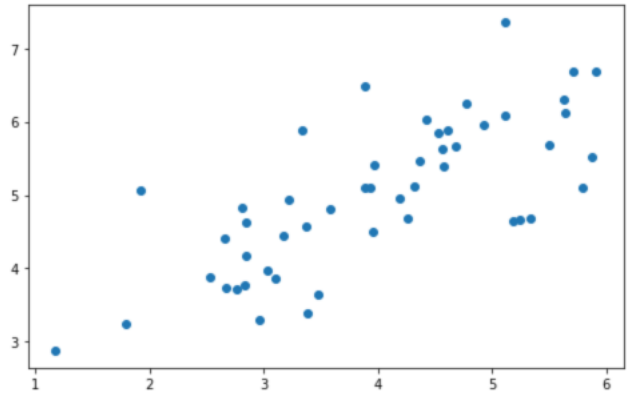

Рисунок 1 – исходные данные файла ex6data1.mat.

2. Реализуйте функцию случайной инициализации K центров кластеров.

Функция инициализации K центров кластеров:

```
def rand_init_centroids(X, K):
    indexes = random.sample(range(0, len(X)), K)

    return X[indexes]
```

```
rand_init_centroids(X1, 3)
```

```
array([[3.89067196, 6.48838087],
       [2.85051337, 4.62645627],
       [3.38156267, 3.38911268]])
```

3. Реализуйте функцию определения принадлежности к кластерам.

Функция определения принадлежности к кластерам:

```
def assign_clusters(X, centroids):
    m = len(X)

    c = np.zeros([m, 1])

    for x_i in range(m):
        x = X[x_i]
        x_distance = 100000 # инициализируем большим занчением

        for c_i in range(len(centroids)):
            centroid = centroids[c_i]
            dist = euclidean_distance(x, centroid)

            if dist < x_distance:
                x_distance = dist
                c[x_i] = c_i

    return c
```

4. Реализуйте функцию пересчета центров кластеров.

Функция пересчета центров кластеров:

```
def move_centroids(X, clusters):
    cluster_valaues = split_data_by_clusters(X, clusters)
    return np.array([np.mean(c_vals, axis=0) for c_vals in cluster_valaues])
```

Вспомогательная функция подсчета принадлежности точек к кластерам:

```
def split_data_by_clusters(X, clusters):
    m = len(X)
    cluster_indexes = np.unique(clusters)

    cluster_values = []

    for c_index in cluster_indexes:
        # получаем все значения для центроида c_index
        values = np.array([X[i] for i in range(m) if clusters[i] == c_index])
        cluster_values.append(values)



    return cluster_values
```

5. Реализуйте алгоритм K-средних.

Функция реализующая алгоритм «K-средних»:

```
def k_means(X, K, max_iter = 10):
    centroids = rand_init_centroids(X, K)
    centroids_history = [centroids]
    clusters = np.zeros([len(X), 1])

    for i in range(max_iter):
        clusters = assign_clusters(X, centroids)
        centroids = move_centroids(X, clusters)
        centroids_history.append(centroids)

    return centroids, clusters, np.array(centroids_history)
```

6. Постройте график, на котором данные разделены на K=3 кластеров (при помощи различных маркеров или цветов), а также траекторию движения центров кластеров в процессе работы алгоритма.

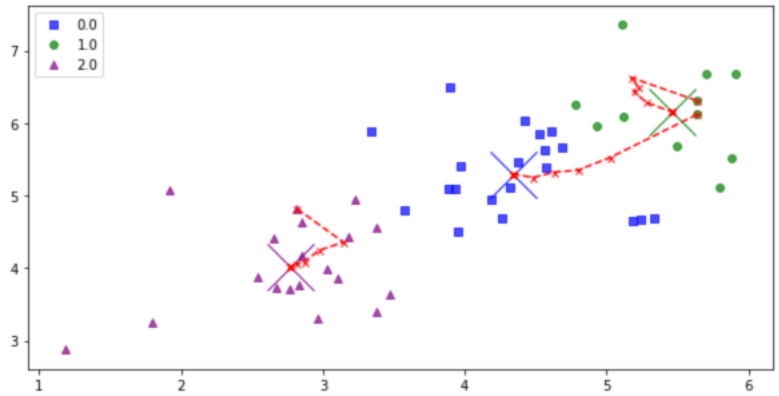

Рисунок 2 – график разделения данных на 3 кластера и траектории движения центроидов.

7. Загрузите данные bird_small.mat из файла..

```
bird_data = scipy.io.loadmat('bird_small.mat')
Xb = bird_data['A']
Xb.shape
```

```
(128, 128, 3)
```

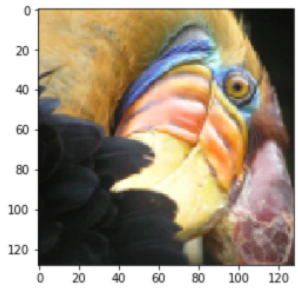

Рисунок 3 – исходные данные файла bird_small.mat.

8. С помощью алгоритма K-средних используйте 16 цветов для кодирования пикселей.

```
bird_k = 16
```

```
bird_centr, bird_clust, _ = best_k_means(Xb.reshape(-1, 3), bird_k, 20, 10)
```

```
Лучший результат: 19.020840958268977 был достигнут на 4 итерации.
```

9. Насколько уменьшился размер изображения? Как это сказалось на качестве?

Оригинальный размер bird_small.mat файла составляет: 128 x 128 x 3 = 49152

Размер нового файла составляет: 16 x 3 + 16384 = 16432, что почти в 3 раза меньше оригинального размера.

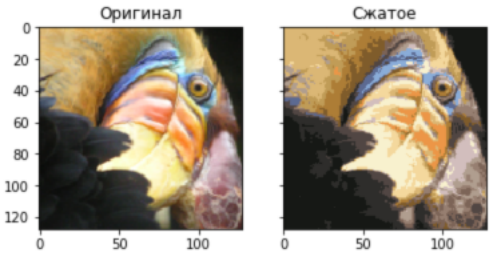

Рисунок 4 – сравнение исходного изображения и сжатого.

Если посмотреть на рисунок 4, то к сожалению можно заметить разницу в качестве невооруженным взглядом.

10. Реализуйте алгоритм K-средних на другом изображении.

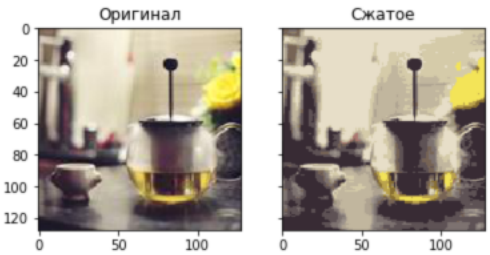

Рисунок 5 – сравнение другого исходного изображения и сжатого.

11. Реализуйте алгоритм иерархической кластеризации на том же изображении. Сравните полученные результаты.

```
cluster = AgglomerativeClustering(n_clusters=bird_k)
cluster.fit_predict(img2.reshape(-1, 3))
```

Иерархическая кластеризация была осуществлена с помощью готового алгоритма «AgglomerativeClustering» из пакета «sklearn.cluster». В качестве метрики связанности использовалось евклидово расстояние, а в качестве оценки расстояния между кластерами был использован метод "ward" (метод Уорда).

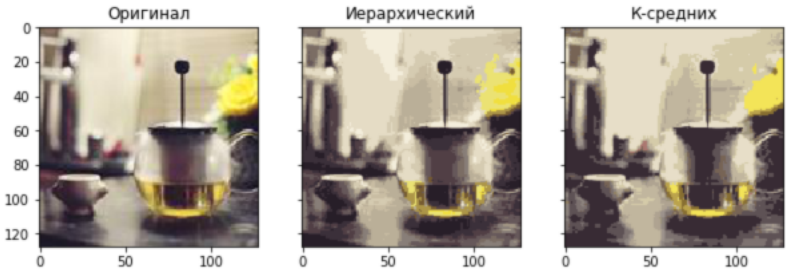

Рисунок 6 – сравнение оригинального изображения, с сжатыми с помощью алгоритмов «иерархической кластеризации» и «к-средних».

Если взглянуть на рисунок 6, то можно заметить, что картинка сжатая с помощью «алгоритма иерархической кластеризации» выглядит лучше, чем картинка сжатая с помощью алгоритма «к-средних».

Вывод.

В ходе выполнения лабораторной работы я ознакомился с одним из направлений «обучения без учителя» - кластеризацией, реализовал алгоритм к-средних и применил его для разбиения набора точек на группы и для сжатия изображения. Так же ознакомился с алгоритмом «иерархической кластеризации» и применил его для сжатия изображения.