Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»
Кафедра информатики

Лабораторная работа №6
«Кластеризация»

Выполнил: Чёрный Родион Павлович
магистрант кафедры информатики
группа №858642

Проверил: доцент кафедры информатики
Стержанов Максим Валерьевич

Минск 2019

Постановка задачи

Набор данных **ex6data1.mat** представляет собой файл формата *.mat (т.е. сохраненного из Matlab). Набор содержит две переменные X_1 и X_2 - 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.

Задание.

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.

Описание реализации

1. Загрузим данные из ex6data1.mat:

```
mat = loadmat("ex6data1")
x = mat['X']
```

2. Функция случайной инициализации K центров кластеров:

```
def initialize_centroids(self):
    idx = np.random.randint(0, self.n, self.k)
    centroids = self.x[idx, :]
    return centroids
```

3. Реализация определения принадлежности к кластеру среди списка кластером, центры которых составляют вектор centroids:

```
@staticmethod
def detect_cluster(point, centroids):
    k = np.argmin([np.linalg.norm(point - centroid) for centroid in centroids])
    return k
```

4. Реализация метода, осуществляющего пересчет пересчет центров кластеров:

```
def recalculate_centroids(self, clusters):
    new_centroids = []
    for cluster in range(self.k):
        cluster_point_indexes = np.argwhere(clusters == cluster)
        cluster_points = self.x[cluster_point_indexes]
        centroid = np.mean(cluster_points, axis=0)[0]
        new_centroids.append(centroid)
    return np.array(new_centroids)
```

5. Имея готовые методы инициализации центров кластеров, функции определения принадлежности к кластеру и пересчета центров кластеров, реализуем алгоритм K-средний кластеризации данных:

```
def clusterize(self, number_of_iterations=10):
    clusters = np.zeros(self.n)
    centroids = self.initialize_centroids()
    for _ in range(number_of_iterations):
        for i, vector in enumerate(self.x):
            k = self.detect_cluster(vector, centroids)
            clusters[i] = k
        centroids = self.recalculate_centroids(clusters)
    return clusters, centroids
```

6. Результат кластеризации, а также движения центров кластеров изображены на рисунке 1. Крестиками обозначены центры сформировавшихся кластеров.

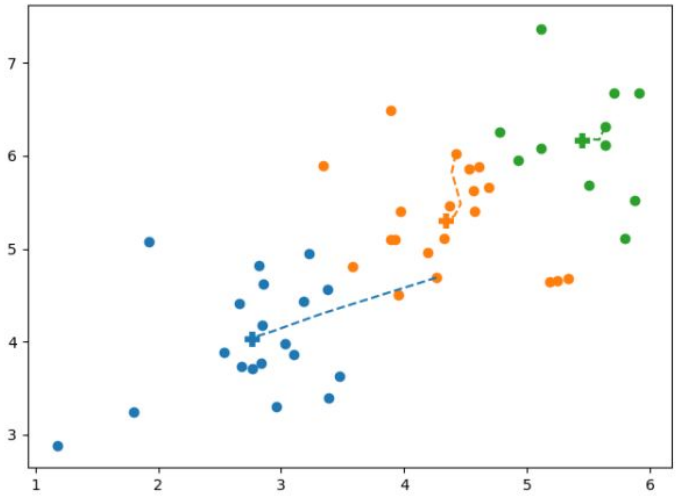

Рисунок 1 - сформированные кластеры исходного набора данных

7. Загрузим данные из файла bird_small.mat и изобразим исходную картинку на рисунке 2.

```
mat = loadmat("bird_small")
data = mat['A']
```

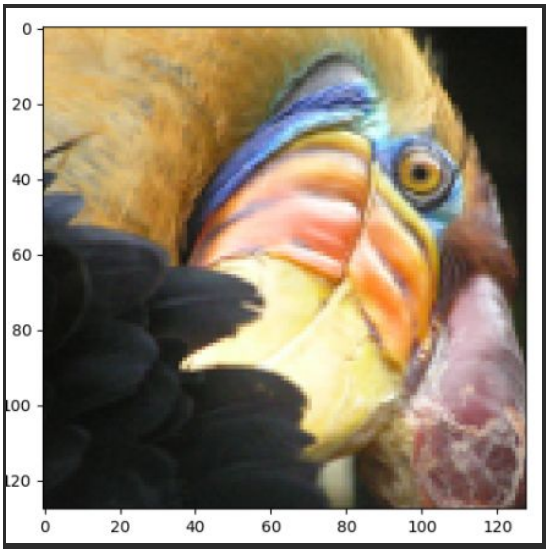

Рисунок 2 - Исходное изображение

8. Запустим 10 итераций алгоритма K-средних с числом K равным 16. После чего заменим в исходном изображении цвета на центры кластеров, к которым они относятся. Результат изображен на рисунке 3.

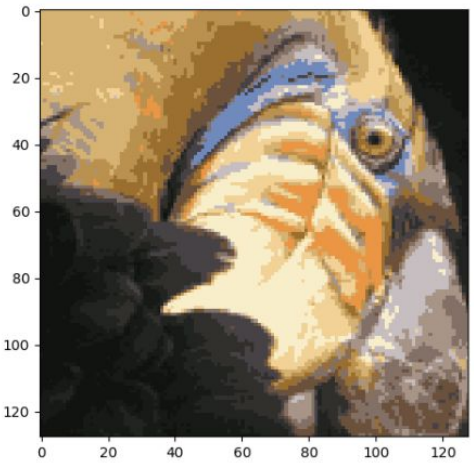

Рисунок 3 - Исходное изображение, закодированное 16 цветами

9. Размер изображения уменьшился с 32.3КБ до 8.89КБ, при этом качество изображение заметно ухудшилось.

10. Применим алгоритм K-средних с теми же параметрами на другом изображении. На рисунке 4 изображено исходное изображение, а на рисунке 5 - его закодированный вариант.

Рисунок 4 - Исходное изображение

Рисунок 5 - 16 цветное изображение