|  |  |  |  |
| --- | --- | --- | --- |
| Министерство образования Республики Беларусь  Учреждение образования  Белорусский Государственный Университет Информатики и Радиоэлектроники | | | |
| Факультет | Компьютерных сетей и систем | | |
| Кафедра | Информатики  Дисциплина: Конструирование те технологии электронных вычислительных средств | | |
|  |  | | |
| **лабораторная работа №6**  по курсу Машинное обучение  на тему  **Кластеризация** | | | |
| Магистрант:  гр. 858341  Кудин Н.И. | |  | Проверил:  Стержанов М.В. |
| Минск, 2019 | | | |

**Постановка задачи**

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.

**Ход работы**

1. Загрузим данные **ex6data1.mat** из файла с помощью scipy.

mat = scipy.io.loadmat("ex6data1.mat")  
X = mat['X']

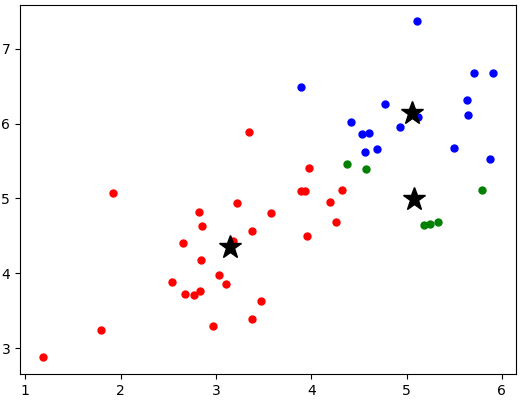
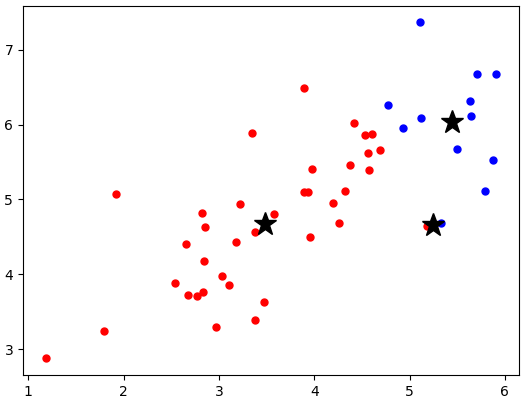
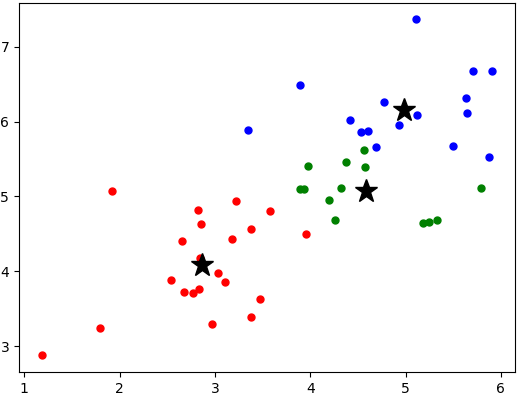
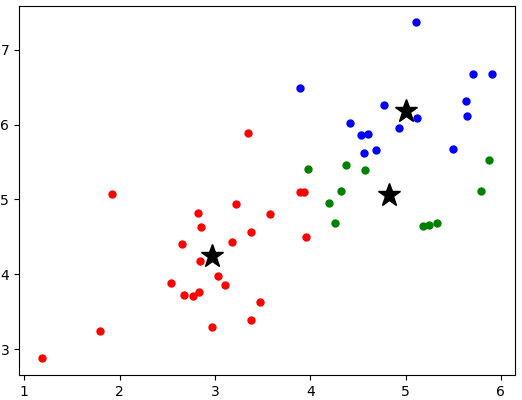
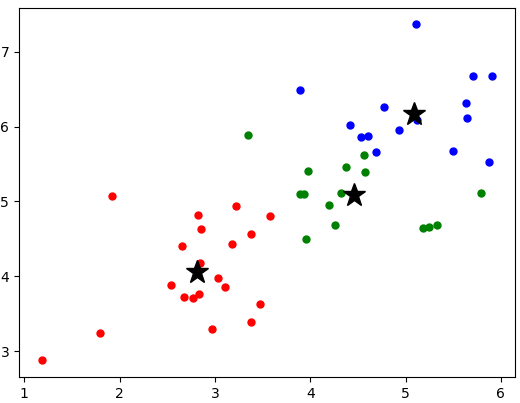
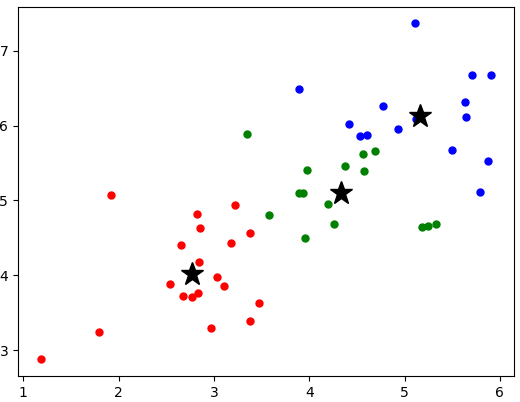
1. Реализуем функцию случайной инициализации K центров кластеров.

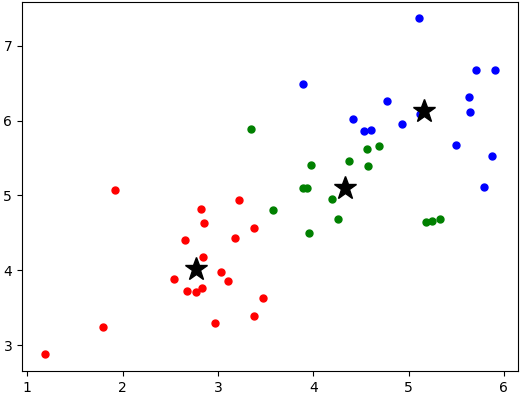
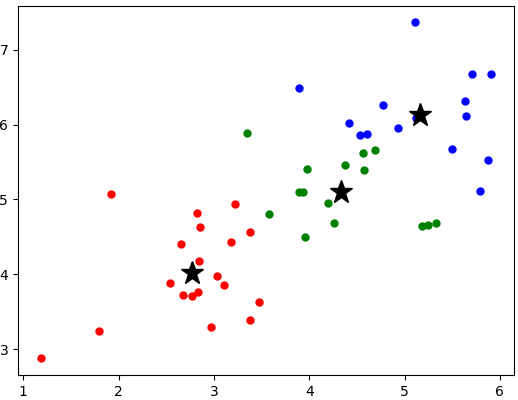
def kMeansInitCentroids(X, K):  
  
 randidx = np.random.permutation(X.shape[0])  
 centroids = X[randidx[:K], :]  
  
  
 return centroids

1. Реализуем функцию определения принадлежности к кластерам и пересчета центров кластеров.

def runkMeans(X, initial\_centroids, max\_iters, plot=False):  
 K = shape(initial\_centroids)[0]  
 centroids = copy(initial\_centroids)  
 idx = None  
  
 for iteration in range(0, max\_iters):  
 idx = findClosestCentroids(X, centroids)  
 centroids = computeCentroids(X, idx, K)  
  
 if plot is True:  
 data = c\_[X, idx]  
  
 data\_1 = data[data[:, 2] == 1]  
 pyplot.plot(data\_1[:, 0], data\_1[:, 1], 'ro', markersize=5)  
  
 data\_2 = data[data[:, 2] == 2]  
 pyplot.plot(data\_2[:, 0], data\_2[:, 1], 'go', markersize=5)  
  
 data\_3 = data[data[:, 2] == 3]  
 pyplot.plot(data\_3[:, 0], data\_3[:, 1], 'bo', markersize=5)  
  
 pyplot.plot(centroids[:, 0], centroids[:, 1], 'k\*', markersize=17)  
  
 pyplot.show(block=True)  
  
 return centroids, idx

1. Построим график, на котором данные разделены на K=3 кластеров (при помощи различных маркеров или цветов). Несколькими изображениями отобразим траекторию движения центров кластеров в процессе работы алгоритма

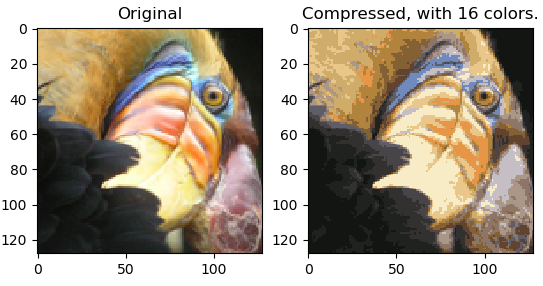
 

1. Загрузим данные **bird\_small.mat** из файла.

mat = scipy.io.loadmat('bird\_small.mat')  
A = mat["A"]

1. С помощью алгоритма K-средних используем 16 цветов для кодирования пикселей.

idx = fcc.findClosestCentroids(X, centroids)  
  
# Essentially, now we have represented the image X as in terms of the  
# indices in idx.  
  
# We can now recover the image from the indices (idx) by mapping each pixel  
# (specified by it's index in idx) to the centroid value  
X\_recovered = centroids[idx, :]  
  
# Reshape the recovered image into proper dimensions  
X\_recovered = X\_recovered.reshape(img\_size[0], img\_size[1], 3, order='F')  
  
# Display the original image  
plt.close()  
plt.subplot(1, 2, 1)  
plt.imshow(A)  
plt.title('Original')  
  
# Display compressed image side by side  
plt.subplot(1, 2, 2)  
plt.imshow(X\_recovered)  
plt.title('Compressed, with {:d} colors.'.format(K))  
plt.show(block=False)  
plt.savefig('results1.png')  
print('results1.png has bee saved')



Размер изображения уменьшился в несколько раз, качество стало намного хуже, что и предполагалось изначально.

**Заключение**

В ходе лабораторной работа были изучены различные инструменты для кластеризации изображения. На практике были изучены способы сжатия изображения с помощью K-means алгоритма.

Определены минусы данного алгоритма

* необходимо заранее знать количество кластеров.
* алгоритм чувствителен к выбору начальных центров кластеров. Классический вариант подразумевает случайный выбор класторов, что очень часто являлось источником погрешности. Возможный вариант решения - необходимо проводить исследования объекта для более точного определения центров начальных кластеров.
* не справляется с задачей, когда объект принадлежит к разным кластерам в равной степени или не принадлежит ни одному.

Но, к примеру в алгоритмах глубокого обучения метод k-средних иногда применяют не по прямому назначению (классификация разбивкой на кластеры), а для создания так называемых фильтров (ядер свёртки, словарей).