Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Кафедра информатики

Отчет по лабораторной работе:

**Лабораторная работа №6 “ Кластеризация”**

Выполнил: Карп Александр Игоревич

магистрант кафедры информатики

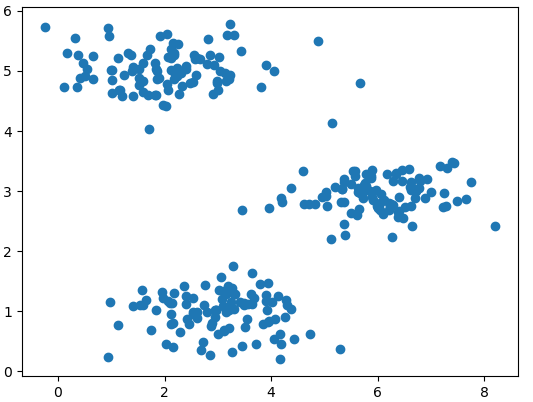
группа №858641

Минск 2019

1. Загрузите данные ex6data1.mat из файла.

(Данные были взяты из 7 лабы, так как, судя по всему, произошла ошибка, и они были перепутаны)

#1  
data = sio.loadmat('ex6data1.mat')  
X = data.get('X')



1. Реализуйте функцию случайной инициализации K центров кластеров.

def get\_random\_centoids(k, low, high):  
 return np.random.randint(low, high, size=(k, 2))

1. Реализуйте функцию определения принадлежности к кластерам.

def findClosestCentroids(X, centroids):  
 *"""  
 Returns the closest centroids in idx for a dataset X where each row is a single example.  
 """* K = centroids.shape[0]  
 idx = np.zeros((X.shape[0], 1))  
 temp = np.zeros((centroids.shape[0], 1))  
  
 for i in range(X.shape[0]):  
 for j in range(K):  
 dist = X[i, :] - centroids[j, :]  
 length = np.sum(dist \*\* 2)  
 temp[j] = length  
 idx[i] = np.argmin(temp) + 1  
 return idx

1. Реализуйте функцию пересчета центров кластеров.

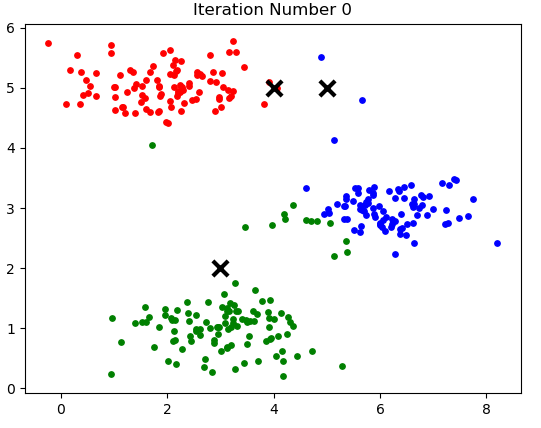
def computeCentroids(X, idx, K):  
 *"""  
 returns the new centroids by computing the means of the data points assigned to each centroid.  
 """* m, n = X.shape[0], X.shape[1]  
 centroids = np.zeros((K, n))  
 count = np.zeros((K, 1))  
  
 for i in range(m):  
 index = int((idx[i] - 1)[0])  
 centroids[index, :] += X[i, :]  
 count[index] += 1  
  
 return centroids / count

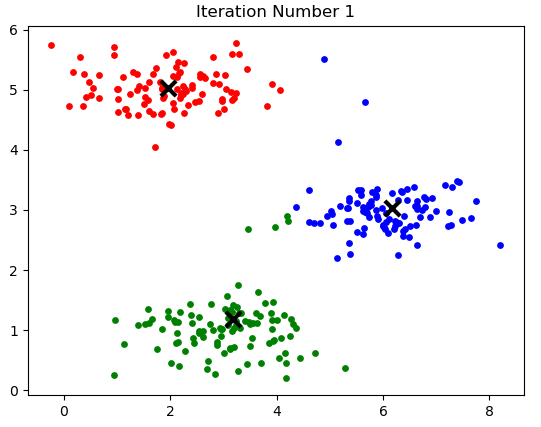
1. Реализуйте алгоритм K-средних.

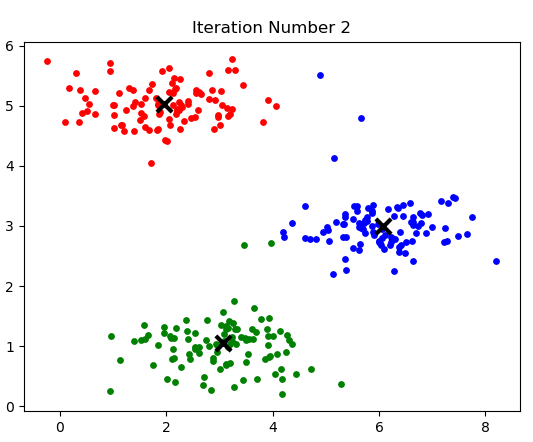
def k\_means(X, idx, K, num\_iters):  
 for i in range(num\_iters):  
 # Compute the centroids mean  
 centroids = computeCentroids(X, idx, K)  
  
 # assign each training example to the nearest centroid  
 idx = findClosestCentroids(X, centroids)  
  
 return [centroids, idx]

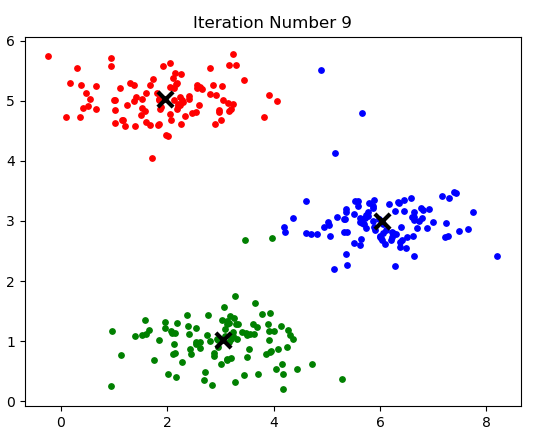
1. Постройте график, на котором данные разделены на K=3 кластеров (при помощи различных маркеров или цветов), а также траекторию движения центров кластеров в процессе работы алгоритма

def plotKmeans(X, centroids, idx, K, num\_iters):  
 *"""  
 plots the data points with colors assigned to each centroid  
 """* m, n = X.shape[0], X.shape[1]  
 for i in range(num\_iters):  
 # Visualisation of data  
 color = "rgb"  
 for k in range(1, K + 1):  
 grp = (idx == k).reshape(m, 1)  
 plt.scatter(X[grp[:, 0], 0], X[grp[:, 0], 1], c=color[k - 1], s=15)  
 # visualize the new centroids  
 plt.scatter(centroids[:, 0], centroids[:, 1], s=120, marker="x", c="black", linewidth=3)  
 title = "Iteration Number " + str(i)  
 plt.title(title)  
  
 [centroids, idx] = k\_means(X, idx, K, 1)  
 # Compute the centroids mean  
 # centroids = computeCentroids(X, idx, K)  
 #  
 # # assign each training example to the nearest centroid  
 # idx = findClosestCentroids(X, centroids)  
 plt.show()









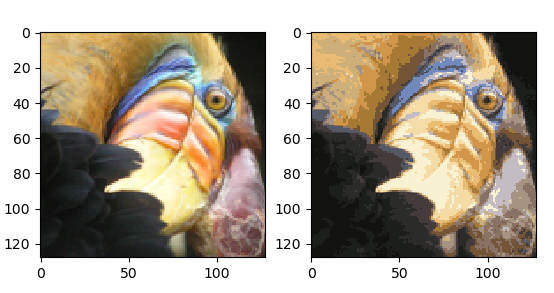
Cлучайные данные были удачно подобраны, поэтому уже на первой итерации центры были определены достаточно точно.

1. Загрузите данные bird\_small.mat из файла.

#7  
data = sio.loadmat('bird\_small.mat')  
X = data.get('A')

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

X2 = (X/255).reshape(128\*128,3)  
K2 = 16  
num\_iters = 10  
  
# preprocess and reshape the image  
X2 = (X / 255).reshape(128 \* 128, 3)  
centroids2, idx2 = k\_means(X2, kMeansInitCentroids(X2, K2), K2, num\_iters)

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

Размер уменьшился в 7 раз. На сравнении изображений можно увидеть, что качество изображения после алгоритма ухудшилось.

**Выводы**

В рамках лабораторной работы был рассмотрен алгоритм k-средних.

Алгоритм просто в использовании и понимании, однако имеется ряд недостатков:

* необходимо заранее знать количество кластеров. Мной было предложено метод определения количества кластеров, который основывался на нахождении кластеров, распределенных по некоему закону (в моем случае все сводилось к нормальному закону). После этого выполнялся классический алгоритм k-means, который давал более точные результаты.
* алгоритм очень чувствителен к выбору начальных центров кластеров. Классический вариант подразумевает случайный выбор класторов, что очень часто являлось источником погрешности. Как вариант решения, необходимо проводить исследования объекта для более точного определения центров начальных кластеров. В моем случае на начальном этапе предлагается принимать в качестве центов самые отдаленные точки кластеров.
* не справляется с задачей, когда объект принадлежит к разным кластерам в равной степени или не принадлежит ни одному.