

实验四KMeans报告

**江辉--171250506**



**2020-4-20**

**南京大学**

**软件学院**

目录

[一、 实验内容描述 1](#_Toc38310359)

[1、 项目目录描述 1](#_Toc38310360)

[二、 实验结果图 2](#_Toc38310361)

[三、 算法原理阐述 4](#_Toc38310362)

[四、 核心代码讲解 4](#_Toc38310363)

[1、 KMeans算法类 4](#_Toc38310364)

# 实验内容描述

本程序用python编写，手写KMeans算法的实现。100个随机点和随机数量个核(我代码里随机范围是3-13)，通过 Matplotlib绘制每一次迭代的聚类过程

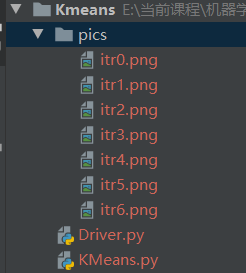
## 项目目录描述

本项目推荐使用pycharm打开

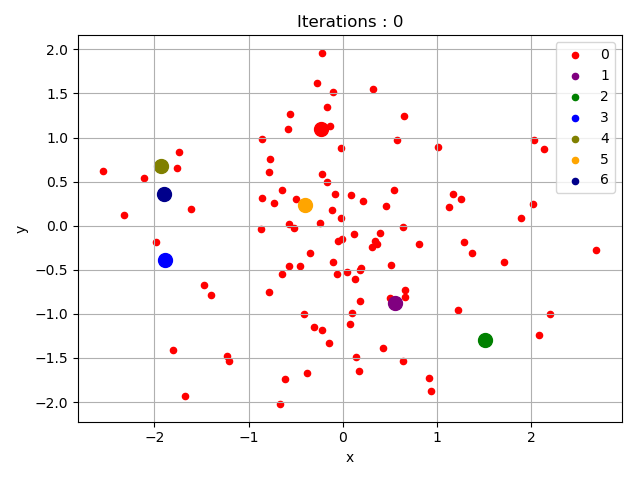
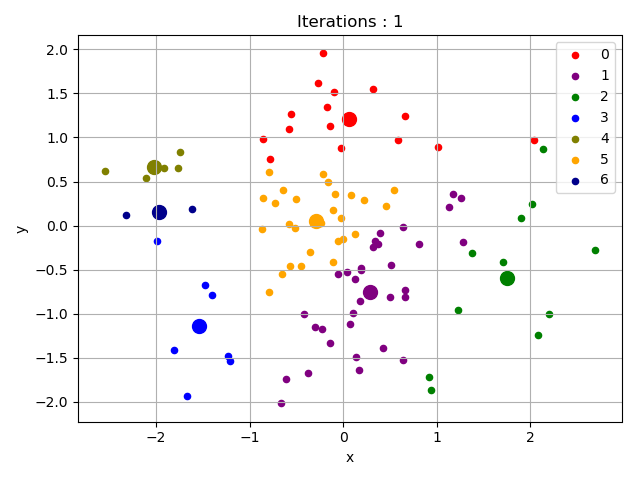
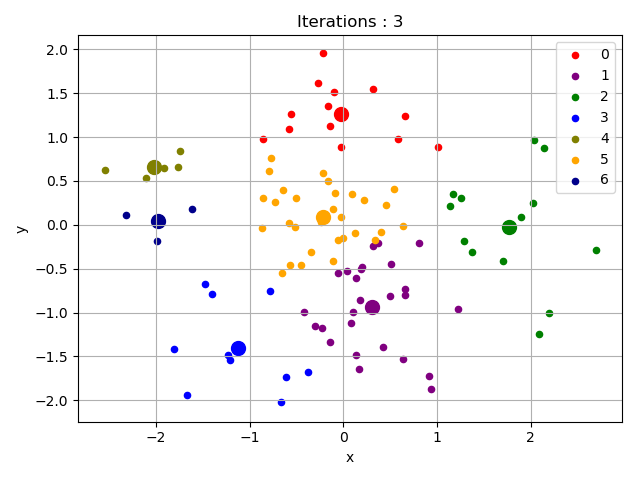
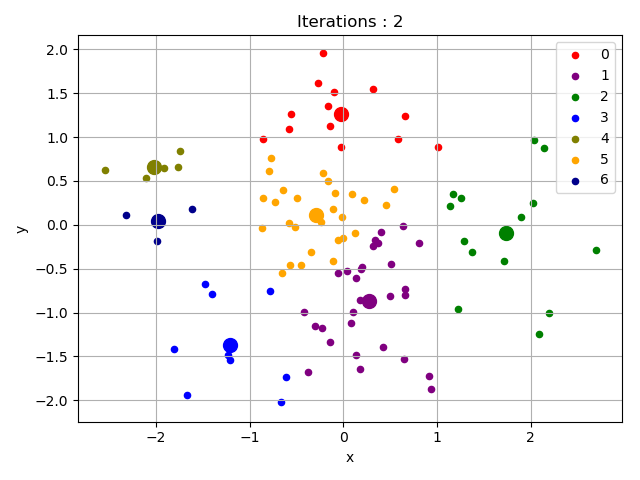
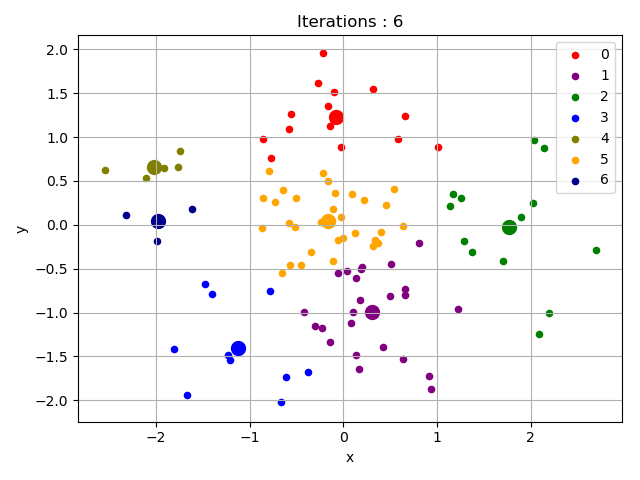
**KMeans.py**里写了实现的代码，同时附带了pyplot画图代码

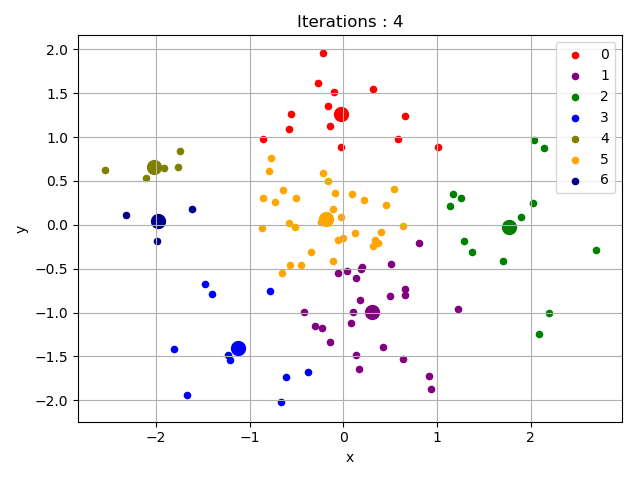
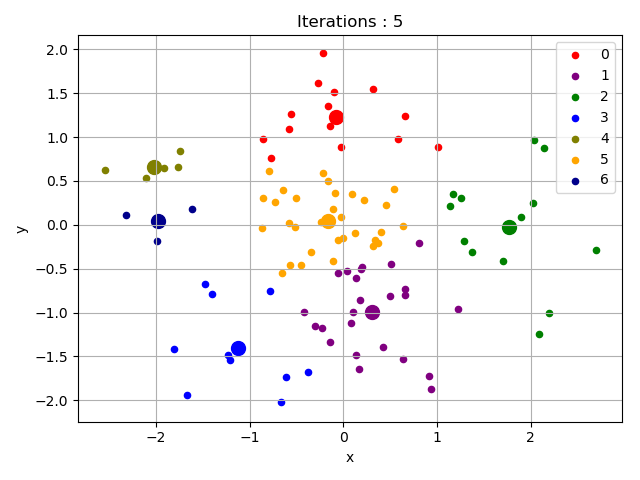
**Driver.py**里是main函数入口

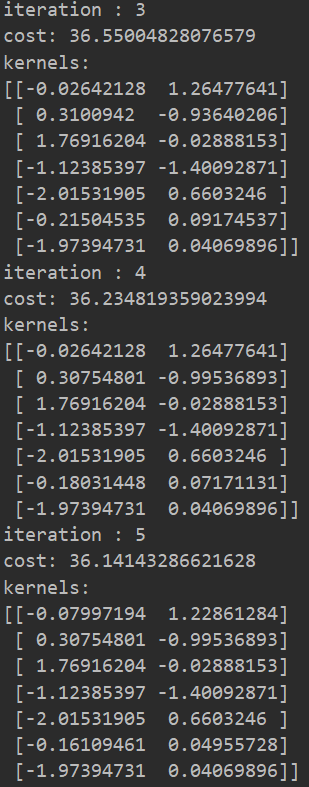
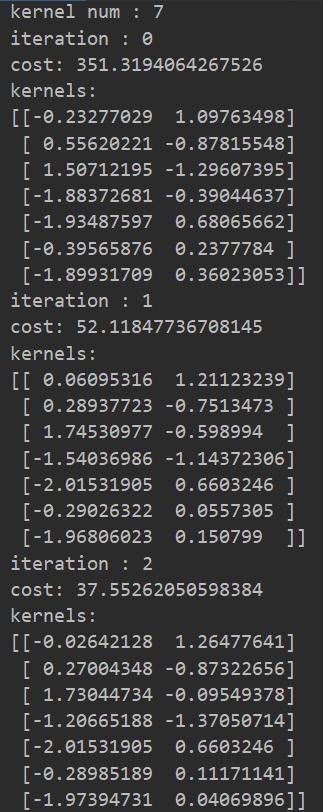
**pics**目录下是每一次迭代画出来的图

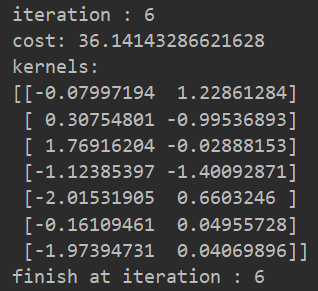


# 实验结果图

以下是用pyplot画的各迭代的图。该用例运行的时候随机到有7个聚类中心，各个颜色类的中心点用比较大的点表示。可以看到其实在前几次计算后各个中心点就大致确定方位了，在后续的聚类过程中只有个别中心点的微调(从相对网格的位置线可以看出来，图中蓝色点中心变化比较明显)，所以每次迭代时中心点的移动都不大，在第6次迭代的时候就完全稳定了



下面是cost变化和中心点坐标变化，可以发现，cost在刚开始下降很快，后面就下降很少了，也就是说基本稳定了，只有个别点还在变动，第6次迭代就已经收敛了



# 算法原理阐述

K-Means算法是一种无监督分类算法，只需要数据，而不需要标签。假设预先给定n个训练样本（无标记），如：{X1,X2,…,Xn}，同时给定聚类的个数K。

目标：把比较“接近”的样本放到一个簇类（cluster）里，总共得到K个簇类（cluster）

具体做法：

1. 初始化k个中心点n1、n2、、、nk
2. 计算所有点Xi与所有中心点n的距离(可以是欧氏距离、曼哈顿距离核函数映射后距离等)，取距离最近的的中心点作为该点的簇中心
3. 根据归入各个簇的样本点，分别计算各个簇新的簇中心(可以取簇内所有样本点均值，也可以取最中心的样本点作为簇中心等)
4. 重复步骤2、3直至所有簇中心不再变化

# 核心代码讲解

## KMeans算法类

下图是我实现的KMeans算法类

is\_convergent函数用于判断是否收敛

classify用于重新将素有点进行归类

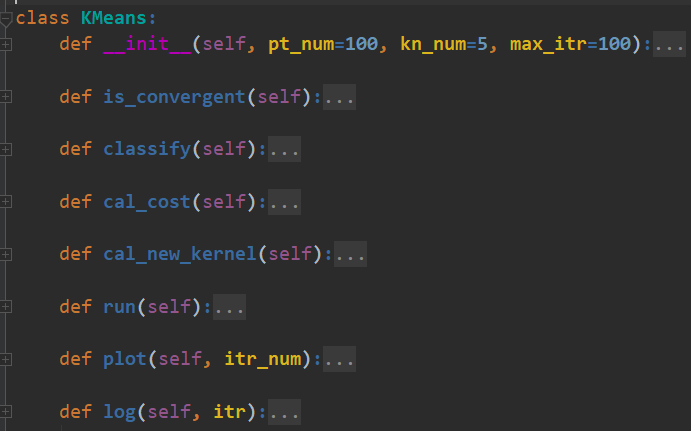
cal\_new\_kernel是重新计算中心点坐标

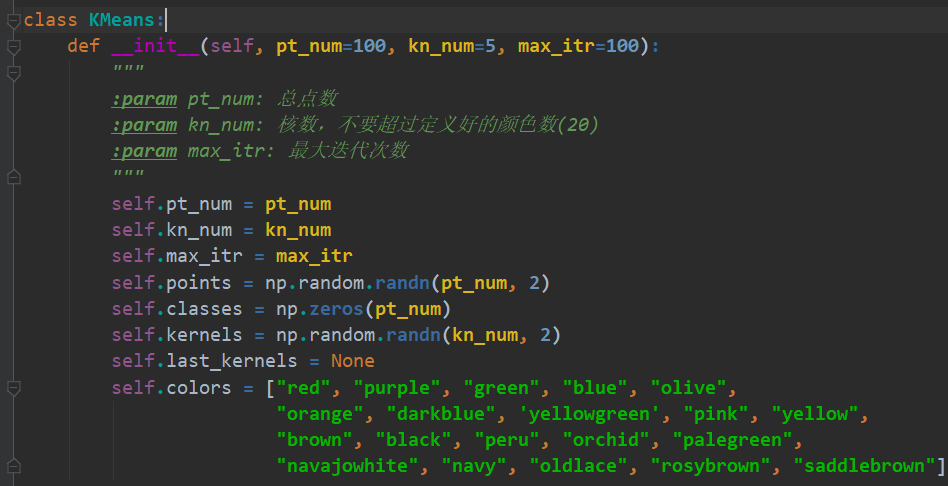
cal\_cost是计算误差值

run是kmeans主体

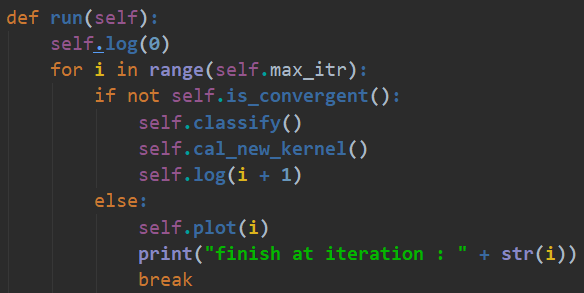
log是做绘图、打印中心点坐标、cost值等

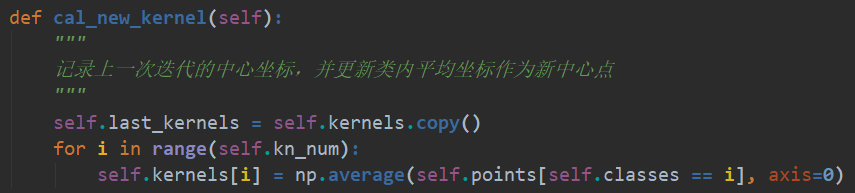
plot是绘图方法



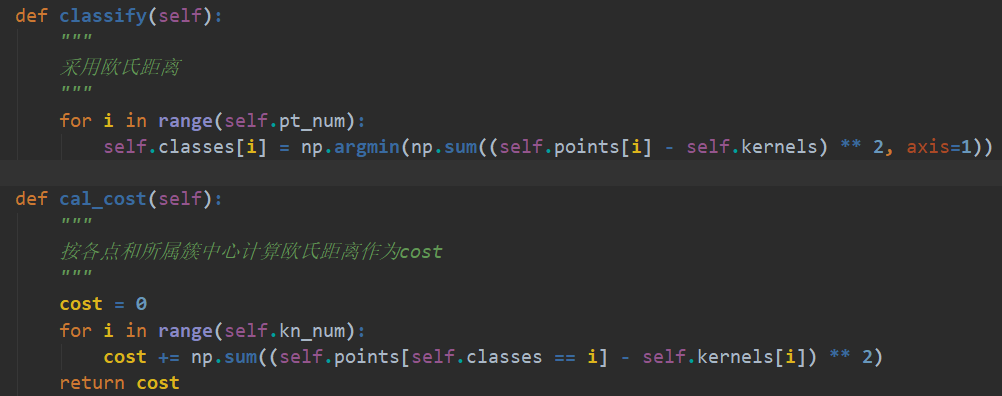
\_\_init\_\_中可以看到就是默认参数是100个随机点，5个中心点，最多100次迭代(在main函数里刷随机数传入即可修改) 。classes存了各个点对应的簇中心序号，last\_kernels是记录上一次迭代时的中心点坐标，用于判断是否收敛

整个KMeans算法流程就是对每一次迭代先判断是否已经收敛，收敛则退出，不收敛则依次进行点归类、重算中心点、画图、打印中心点坐标和计算cost

  
记录并更新簇中心用的是簇内平均坐标



距离计算采用的是欧氏距离，下面的classify方法是将每个点归类，计算与所有簇中心的欧氏距离，取最小距离的类保留在classes数组内



判断聚类是否停止，我在每次迭代后都会记录的上一次的簇中心坐标，因此只需要比对每次迭代所有簇中心坐标是否都和上一次迭代一样即可

