# 实验五：K-Means在数据分析中的应用实验

学号: 18401190103 姓名： 谢威

学号: 18401190202 姓名： 陆镇涛

学号: 18401190120 姓名： 曹鹏霄

学号: 18401190107 姓名： 王云浩 提交时间： 2020-11-5

## 一、实验目的

1．了解聚类算法的相关概念；

2．理解并掌握K-Means算法原理；

3．进一步熟悉Pandas对文件操作；

4．初步掌握K-Means算法在生活中的应用；

## 实验内容

1. 运用聚类算法对给出红酒数据进行聚类（聚为三类）。
2. 给定一张JPEG格式彩色图像，采用聚类的算法将其二值化（黑白）。

## 实现过程分析

**K-Means的工作原理：**

1选取 K 个点作为初始的类中心点，这些点一般都是从数据集中随机抽取的；

2将每个点分配到最近的类中心点，这样就形成了 K 个类，然后重新计算每个类的中心点；（这个怎么算最近，一般是欧几里得距离公式， 那么怎么重新计算每个类的中心点， 每个维度的平均值就可以的）；

3重复第二步，直到类不发生变化，或者你也可以设置最大迭代次数，这样即使类中心点发生变化，但是只要达到最大迭代次数就会结束。

**1、将红酒数据聚类为三类**

**数据分析：**首先要将13列数据规范化，你可以把这些值划分到[0,1]或者按照均值为 0，方差为 1 的正态分布进行规范化。

**模型建立：**随机取3个品种的酒作为数据13种属性的中心点，通过计算每行代码到该3种酒的欧氏距离，选取欧氏距离最近的点作出聚类。再取三类数据中的中心点作为新的三类参考点，所有数据再进行聚类，不断迭代至稳定状态。

**2、图像二值化**

**数据分析：**定义一个 load\_data()进行图像加载和数据规范化。能在获取图像数据以外，还需要获取图像的尺寸和通道数，然后基于图像中每个通道的数值进行数据规范化。采用 Min-Max 规范化对数据进行规范化。

**模型建立：**由于要将图像转变为黑白两色，即灰度值为0或255。所以我们要将图像中每个像素通过K-Means聚类分为两类，类似于将图像矩阵用0或1 替代。最终根据矩阵对应的类来赋予新图像中该位置像素对应的灰度值。

## 源代码

**1、将红酒数据聚类为三类**

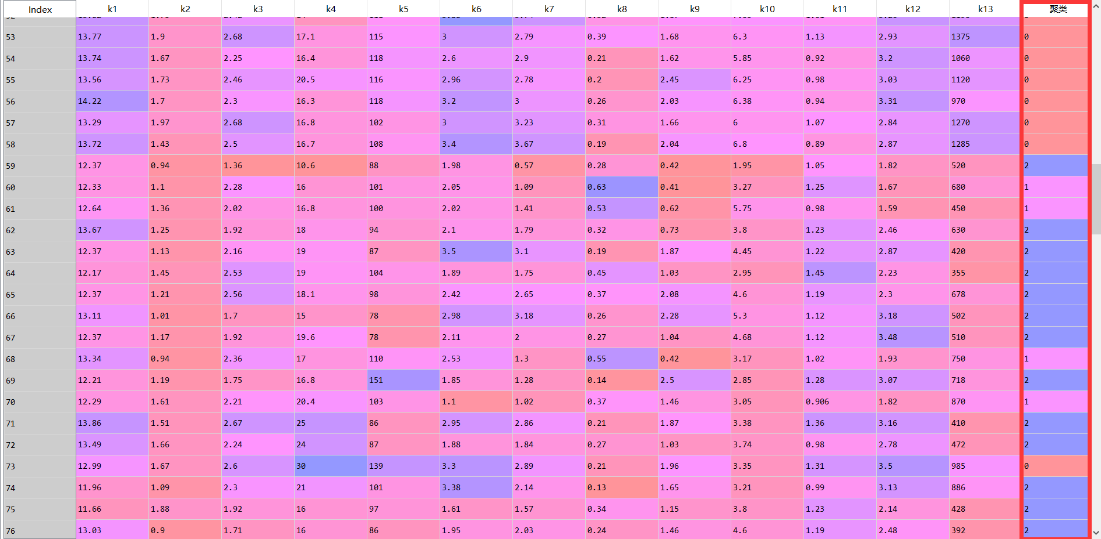
1. # coding: utf-8
2. **from** sklearn.cluster **import** KMeans
3. **from** sklearn **import** preprocessing
4. **import** pandas as pd
5. # 输入数据
6. data = pd.read\_csv('wine\_Kmeans.data', delimiter=',')
7. train\_x = data[["k1","k2","k3","k4","k5","k6","k7","k8","k9","k10","k11","k12","k13"]]
8. df = pd.DataFrame(train\_x)
9. kmeans = KMeans(n\_clusters=3)
10. # 规范化到[0,1]空间
11. min\_max\_scaler=preprocessing.MinMaxScaler()
12. train\_x=min\_max\_scaler.fit\_transform(train\_x)
13. # kmeans算法
14. kmeans.fit(train\_x)
15. predict\_y = kmeans.predict(train\_x)
16. # 合并聚类结果，插入到原数据中
17. result = pd.concat((data,pd.DataFrame(predict\_y)),axis=1)
18. result.rename({0:u'聚类'},axis=1,inplace=True)
19. **print**(result)

**2、图像二值化**

1. # -\*- coding: utf-8 -\*-
2. # 使用K-means对图像进行聚类，显示分割标识的可视化
3. **import** numpy as np
4. **import** PIL.Image as image
5. **from** sklearn.cluster **import** KMeans
6. **from** sklearn **import** preprocessing
8. # 加载图像，并对数据进行规范化
9. **def** load\_data(filePath):
10. # 读文件
11. f = open(filePath,'rb')
12. data = []
13. # 得到图像的像素值
14. img = image.open(f)
15. # 得到图像尺寸
16. width, height = img.size
17. **for** x **in** range(width):
18. **for** y **in** range(height):
19. # 得到点(x,y)的三个通道值
20. c1, c2, c3 = img.getpixel((x, y))
21. data.append([c1, c2, c3])
22. f.close()
23. # 采用Min-Max规范化
24. mm = preprocessing.MinMaxScaler()
25. data = mm.fit\_transform(data)
26. **return** np.mat(data), width, height
28. # 加载图像，得到规范化的结果img，以及图像尺寸
29. img, width, height = load\_data('./weixin.jpg')
31. # 用K-Means对图像进行2聚类
32. kmeans =KMeans(n\_clusters=2)
33. kmeans.fit(img)
34. label = kmeans.predict(img)
35. # 将图像聚类结果，转化成图像尺寸的矩阵
36. label = label.reshape([width, height])
37. # 创建个新图像pic\_bin,保存图像聚类的结果，并设置为0或255（黑白）
38. pic\_bin = image.new("L", (width, height))#L为灰度显示
39. **for** x **in** range(width):
40. **for** y **in** range(height):
41. # 根据类别设置图像灰度, 类别0 灰度值为255（白）， 类别1 灰度值为0（黑）
42. pic\_bin.putpixel((x, y), int(255\*(label[x][y])))
43. pic\_bin.save("weixin\_bin.jpg", "JPEG")

## 实验结果

1. **将红酒数据聚类为三类**



**红酒数据分为三类**

**2、图像二值化**

****

**微信黑白图**

## 结论

**1、将红酒数据聚类为三类**

由于每次运行所选取的基础中心点（第一次迭代）以及迭代次数的不同，聚类的结果也会产生不同的变化。其中观察实验结果可以发现数据头尾两侧的相似度较高，每次实验聚类后同类的概率非常高。

**2、图像二值化**

图像的二值化总体与其聚类后对应的灰度值有关，通过聚类为两类让其赋予不同的灰度值从而展现出黑白的效果。同理可以通过不同的灰度值来展现出不同灰度的两色灰度图，或者是根据聚类的类型数量不同来改变图片的色彩数量。