**北京科技大学**

**《模式识别基础》实验报告**

学院： 计通 专业： 计科 班级： 计213

姓名： 王佳 学号： U202142840 实验日期： 2024.04.03

**实验名称：**

利用scikit-learn实现k均值聚类

**实验原理：**

1. 聚类是一种无监督学习方法，用于将相似的数据对象归为一组。K-Means Clustering是常用的基于欧氏距离的聚类算法，其认为两个目标的距离越近，相似度越大。算法步骤是：选择初始化的k个样本作为初始聚类中心a=a1,a2...ak;针对数据集中的每个样本xi计算它到k个聚类中心的距离并将其分到距离最小的聚类中心所对应的类中；针对每个类别aj，重新计算它的聚类中心;重复以上操作直到达到某个终止条件（迭代次数，最小误差变化等）

2）Scikit-learn是一个Python机器学习库，通过该库可以实现各种监督和非监督学习模型，还可以进行数据的预处理，特征工程，数据集切分，模型评估，获取丰富的数据集等。

3）Sklearn..cluster.KMeans类实现了K-Means。

类实例化参数：

·n\_clusters:要形成的簇数以及要生成的质心数

·init:初始化方法

·n\_init:用不同的初始化质心运行算法的次数

·max\_iter: 最大的迭代次数

·tol:相对容忍度与Frobenius范数

·algorithm:表示K-means要使用的算法，有”auto”,”full”,”elkan”三种选择

类属性：

·cluster\_centers\_:簇中心坐标

·labels\_:每一点的标签

·inertia\_:样本到其最近聚类中心的平方距离之和

·n\_iter\_:运行的迭代次数

方法：

·\_init\_:初始化self

·\_fit(self,X[,y,sample):计算k-均值聚类，sX是要聚类的训练实例，sample\_weight是X中每个观测值的权重，返回拟合的估计器

·fit\_predict:计算聚类中心并预测每个样本的聚类索引，返回值是labels

·fit\_tranform:计算聚类并将X变换为簇距离空间

·get\_params:获取估计器的参数

·predict:预测X中每个样本所属最近的聚类,返回值是labels

·score:K-均值目标上X值的相反

·set\_params:设置此估计器的参数

·transform:将X转换为簇距空间

**实验内容：**

1）实现第二章作业中的聚类问题

2）学习一个复杂一点的聚类示例，学会如何对聚类结果可视化，聚类中心，不同类别上色

**实验过程及代码：**

1）实现第二章作业的聚类问题

1. **from** sklearn.cluster **import** KMeans
2. **import** numpy as np
3. x=np.array([[0,0],[0,1],[4,4],[4,5],[5,4],[5,5],[1,0]])
4. kmeans=KMeans(n\_clusters=2,random\_state=0).fit(x)
5. **print**(kmeans.labels\_)
6. **print**(kmeans.cluster\_centers\_)

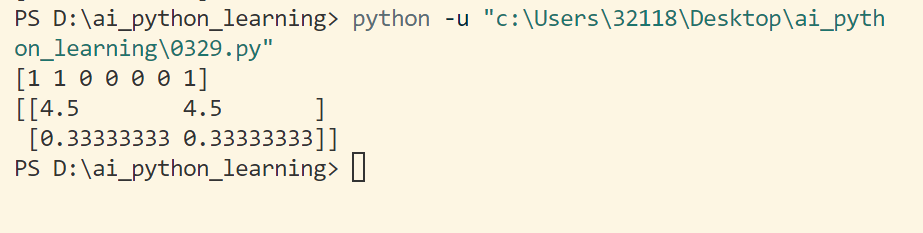
2）该程序导入matplotlib包中的pyplot模块用于绘图，导入numpy用于数据分析，sklearn.cluster和sklearn.datasets用于生成数据和聚类分析

1. **import** time
2. **import** numpy as np
3. **import** matplotlib.pyplot as plt
4. **from** sklearn.cluster **import** MiniBatchKMeans, KMeans
5. **from** sklearn.metrics.pairwise **import** pairwise\_distances\_argmin
6. **from** sklearn.datasets **import** make\_blobs
7. np.random.seed(0)
8. # #############################################################################
9. batch\_size = 45
10. centers = [[1, 1], [-1, -1], [1, -1]]
11. n\_clusters = len(centers)
12. #生成3000个样本数据，三个类别，类别方差为0.7
13. X, labels\_true = make\_blobs(n\_samples=3000, centers=n\_clusters, cluster\_std=0.7)
14. # #############################################################################
15. k\_means = KMeans(init='k-means++', n\_clusters=3, n\_init=10)
16. t0 = time.time()
17. k\_means.fit(X)
18. t\_batch = time.time() - t0
19. # #############################################################################
20. # Compute clustering with MiniBatchKMeans
22. mbk = MiniBatchKMeans(init='k-means++', n\_clusters=3, batch\_size=batch\_size,
23. n\_init=10, max\_no\_improvement=10, verbose=0)
24. t0 = time.time()
25. mbk.fit(X)
26. t\_mini\_batch = time.time() - t0
27. # #############################################################################
28. #创建了一个大小为8\*3英寸的绘图
29. fig = plt.figure(figsize=(8, 3))
30. # 调整子图位置
31. fig.subplots\_adjust(left=0.02, right=0.98, bottom=0.05, top=0.9)
32. colors = ['#4EACC5', '#FF9C34', '#4E9A06']
34. k\_means\_cluster\_centers = k\_means.cluster\_centers\_
35. order = pairwise\_distances\_argmin(k\_means.cluster\_centers\_,
36. mbk.cluster\_centers\_)
37. # order变量包含了用来重新排列mbk.cluster\_centers\_的索引顺序，通过使用mbk.cluster\_centers\_[order]确保两个聚类中心坐标对应
38. mbk\_means\_cluster\_centers = mbk.cluster\_centers\_[order]
39. # 得到包含每个样本聚类标签的数组
40. k\_means\_labels = pairwise\_distances\_argmin(X, k\_means\_cluster\_centers)
41. mbk\_means\_labels = pairwise\_distances\_argmin(X, mbk\_means\_cluster\_centers)
42. # KMeans
43. #创建一个子图，添加到fig的1行3列的第1列位置
44. ax = fig.add\_subplot(1, 3, 1)
46. # range用于生成整数序列0~n\_cluster-1
47. # zip函数用于将多个可迭代对象中对应位置的元素打包成一个元组，可以让我们可视化时为每个聚类指定不同的颜色以区分
48. **for** k, col **in** zip(range(n\_clusters), colors):
49. #创建了一个布尔数组，k\_means\_labels是包含每个样本所属聚类的标签数组，k是当前遍历地聚类索引，将与当前聚类索引k相等的元素设置为true
50. my\_members = k\_means\_labels == k
51. #k\_means\_cluster\_centers是包含了所有聚类中心坐标的数组
52. cluster\_center = k\_means\_cluster\_centers[k]
53. #用于绘制属于当前聚类的样本点
54. ax.plot(X[my\_members, 0], X[my\_members, 1], 'w',
55. markerfacecolor=col, marker='.')
56. #这行代码用于绘制当前聚类中心
57. ax.plot(cluster\_center[0], cluster\_center[1], 'o', markerfacecolor=col,
58. markeredgecolor='k', markersize=6)
59. ax.set\_title('KMeans')
60. ax.set\_xticks(())
61. ax.set\_yticks(())
62. plt.text(-3.5, 1.8,  'train time: %.2fs\ninertia: %f' % (
63. t\_batch, k\_means.inertia\_))
64. # MiniBatchKMeans
65. ax = fig.add\_subplot(1, 3, 2)
66. **for** k, col **in** zip(range(n\_clusters), colors):
67. my\_members = mbk\_means\_labels == k
68. cluster\_center = mbk\_means\_cluster\_centers[k]
69. ax.plot(X[my\_members, 0], X[my\_members, 1], 'w',
70. markerfacecolor=col, marker='.')
71. ax.plot(cluster\_center[0], cluster\_center[1], 'o', markerfacecolor=col,
72. markeredgecolor='k', markersize=6)
73. ax.set\_title('MiniBatchKMeans')
74. ax.set\_xticks(())
75. ax.set\_yticks(())
76. plt.text(-3.5, 1.8, 'train time: %.2fs\ninertia: %f' %
77. (t\_mini\_batch, mbk.inertia\_))
78. # Initialise the different array to all False
79. different = (mbk\_means\_labels == 4)
80. ax = fig.add\_subplot(1, 3, 3)
81. **for** k **in** range(n\_clusters):
82. different += ((k\_means\_labels == k) != (mbk\_means\_labels == k))
83. identic = np.logical\_not(different)
84. ax.plot(X[identic, 0], X[identic, 1], 'w',
85. markerfacecolor='#bbbbbb', marker='.')
86. ax.plot(X[different, 0], X[different, 1], 'w',
87. markerfacecolor='m', marker='.')
88. ax.set\_title('Difference')
89. ax.set\_xticks(())
90. ax.set\_yticks(())
91. plt.show()

代码得到包含每个样本聚类标签的数组后，通过循环遍历所有的样本，利用plot(x,y,color,markerfacecolor,marker)函数将得到的属于不同类的二维样本数据以不同的颜色绘制在子图上，并将聚类中心以区别于类样本的颜色一并绘制在子图上。并利用defferent数组将K-Means方法和MiniBatchKMeans方法得到的类别标签不同的样本数据绘制在第三个子图上。

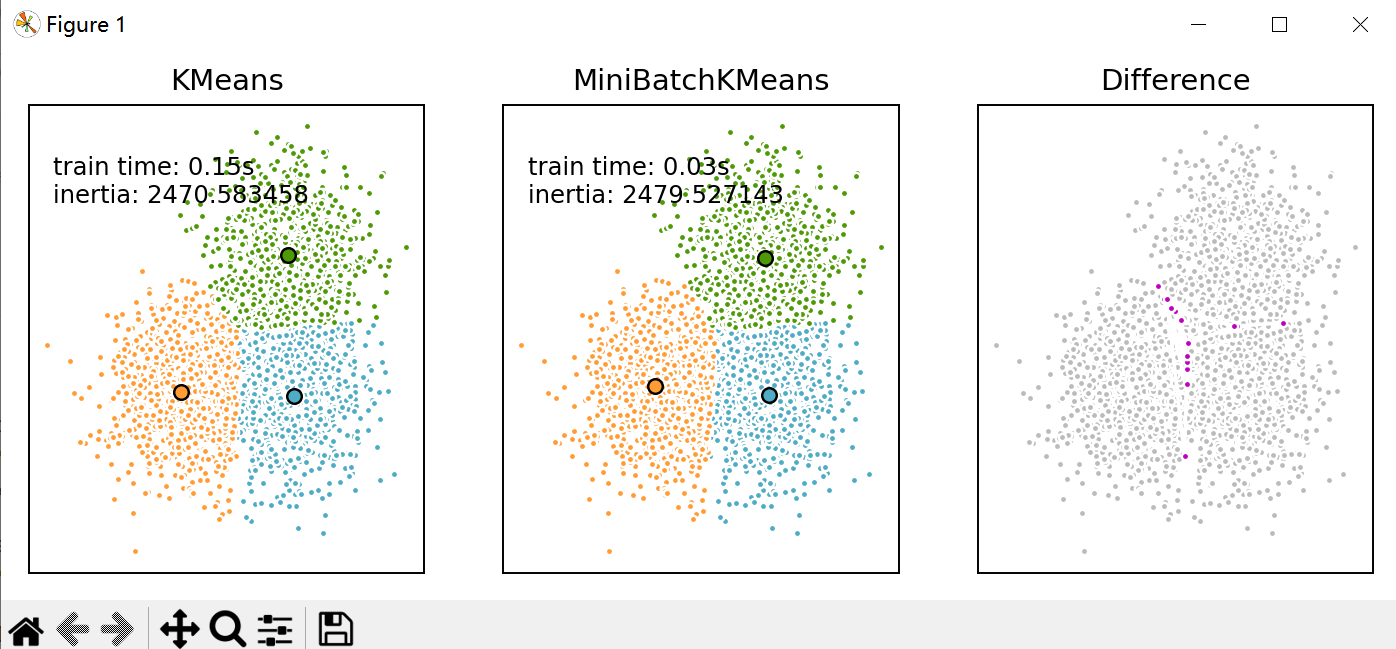
**实验结果与分析：**

1）



结果显示（0，0）（0，1）（1，0）类别标签都是1，（4，4）（4，5）（5，4）（5，5）类别标签都是0，并且输出两个类别聚类中心的坐标（1/3，1/3）（9/2，9/2），与手算结果一致，聚类成功。

**2）**



第一张子图是KMeans方法可视化的聚类结果，第二张图是MiniBatchKMeans方法可视化的聚类结果，通过第三张子图可得Kmeans和MiniBatchKMeans方法得到的聚类结果区别主要在于边界样本的类别。

**实验体会与收获：**

通过本次实验，学会了使用numpy包进行数据分析，使用sklearn包的datasets模块进行聚类问题模拟数据集的生成，和clustre模块的KMeans算法模型的应用，也学会了使用matplotlib包中pyplot模块进行画图来实现数据的可视化。通过实践，初步体会到机器学习算法的有趣和高效，为未来对人工智能领域进一步的学习打下基础