# 模式识别实验报告

## 实验一 K-均值聚类

学院：

姓名：

学号：

1. **实验内容**
2. 使用python或Matlab编程实现K-均值聚类算法：要求独立完成算法编程，禁止调用已有函数库或工具箱中的函数；
3. 使用仿真数据测试算法的正确性：将下列19个样本聚成2个聚类：



1. MNIST数据集测试：ClusterSamples中的10000个784维特征手写数字样本聚类为10个类别，根据SampleLabels中的标签统计每个聚类中不同样本的数量。测试不同初始值对聚类结果的影响。
2. **程序代码**

（K-均值算法部分代码）

======================

KMeans应用于仿真数据

======================

|  |
| --- |
|  |
|  | import numpy as np  import matplotlib.pyplot as plt |
|  | import copy |
|  |  |
|  | class KMeans(object): |
|  | def \_\_init\_\_(self, data): |
|  | self.data = np.array(data).astype(float) |
|  | self.centroids = self.data[np.random.randint(0, self.data.shape[0], 2), :] |
|  | self.label = np.random.randint(0, 2, self.data.shape[0]) |
|  | def EM(self): |
|  | centroids\_error = float('inf') |
|  | data\_row = self.data.shape[0] |
|  | centroids\_row = self.centroids.shape[0] |
|  | while centroids\_error > 1e-20: |
|  | # E step, clustering |
|  | for i in range(0, data\_row): |
|  | dist = np.sqrt(np.sum((self.data[i, :] - self.centroids)\*\*2, axis=1)) |
|  | self.label[i] = np.argmin(dist) |
|  | # M step, recalculate random centers after clustering |
|  | pre\_centroids = copy.deepcopy(self.centroids) |
|  | for j in range(0, centroids\_row): |
|  | n = self.label[self.label == j].size |
|  | self.centroids[j] = np.sum(self.data[self.label == j, :], axis=0)/n |
|  | centroids\_error = np.sum((pre\_centroids-self.centroids)\*\*2) |
|  | def plot(self): |
|  | x0 = self.data[self.label==0, 0] |
|  | y0 = self.data[self.label==0, 1] |
|  | x1 = self.data[self.label==1, 0] |
|  | y1 = self.data[self.label==1, 1] |
|  |  |
|  | plt.axis([-1,10,-1,10]) |
|  | plt.plot(x0, y0, 'go') |
|  | plt.plot(x1, y1, 'bo') |
|  | plt.plot(self.centroids[0][0], self.centroids[0][1], 'r+', ms=10) |
|  | plt.plot(self.centroids[1][0], self.centroids[1][1], 'r+', ms=10) |
|  | plt.show() |
|  |  |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | data = [[0,0],[1,0],[0,1],[1,1], |
|  | [2,1],[1,2],[2,2],[3,2], |
|  | [6,6],[7,6],[8,6],[7,7], |
|  | [8,7],[9,7],[7,8],[8,8], |
|  | [9,8],[8,9],[9,9]] |
|  | test = KMeans(data) |
|  | test.EM() |
|  | test.plot() |

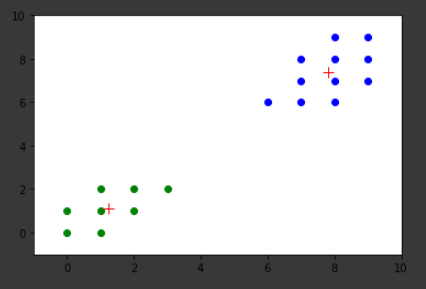
======================================

KMeans应用于Minist数据集

======================================

|  |
| --- |
|  |
|  | import numpy as np  import matplotlib.pyplot as plt |
|  | import pandas as pd |
|  | import csv |
|  | import copy |
|  |  |
|  | def readData(data, filename): |
|  | csvObject = csv.reader(open(filename, 'r')) |
|  | for row in csvObject: |
|  | data.append(row) |
|  |  |
|  | class KMeans(object): |
|  | def \_\_init\_\_(self, data, k): |
|  | self.data = np.array(data).astype(float) |
|  | self.centroids = self.data[np.random.randint(0, self.data.shape[0], k), :] |
|  | self.label = np.zeros((1, self.data.shape[0]))[0] |
|  | def EM(self): |
|  | error = float('inf') |
|  | dataRow = self.data.shape[0] |
|  | centroidsRow = self.centroids.shape[0] |
|  | while error > 1e-20: |
|  | # clustering |
|  | for i in range(0, dataRow): |
|  | dist = np.sqrt(np.sum((self.data[i, :]-self.centroids)\*\*2, axis=1)) |
|  | self.label[i] = np.argmin(dist) |
|  | # recalculate centroids after clustering and calculate the error between precentroids and newcentroids |
|  | preCentroids = copy.deepcopy(self.centroids) |
|  | for j in range(0, centroidsRow): |
|  | n = self.label[self.label==j].size |
|  | self.centroids[j] = np.sum(self.data[self.label==j, :], axis=0)/n |
|  | error = np.sum((preCentroids-self.centroids)\*\*2) |
|  | if \_\_name\_\_ == '\_\_main\_\_': |
|  | data = [] |
|  | label = [] |
|  | readData(data, 'ClusterSamples.csv') |
|  | readData(label, 'SampleLabels.csv') |
|  | label = np.array(label).astype(int).transpose() |
|  | label = label[0, :] |
|  |  |
|  | test = KMeans(data, 10) |
|  | test.EM() |
|  |  |
|  | labelFrame = np.zeros((10, 10)).astype(int) |
|  | for i in range(0, 10): |
|  | for j in range(0, 10): |
|  | realClusterIndex = np.arange(label.size)[label==i] |
|  | testCluster = test.label[realClusterIndex] |
|  | labelFrame[i, j] = testCluster[testCluster==j].size |
|  | df = pd.DataFrame(labelFrame, index=range(10), columns=range(10)) |
|  | print(df) |

1. **实验结果**
2. 仿真数据实验结果：（可以列出每个聚类中包含的样本，也可以画图显示不同聚类）



1. MNIST数据集实验结果：

**每个聚类中包含不同类别样本数量统计表**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 聚类0 | 2 | 22 | 2 | 2 | 30 | 452 | 25 | 2 | 419 | 4 |
| 聚类1 | 2 | 3 | 1101 | 7 | 0 | 0 | 3 | 10 | 0 | 0 |
| 聚类2 | 5 | 40 | 117 | 725 | 28 | 1 | 20 | 9 | 17 | 32 |
| 聚类3 | 8 | 645 | 68 | 32 | 13 | 4 | 160 | 37 | 17 | 6 |
| 聚类4 | 284 | 0 | 30 | 2 | 22 | 0 | 3 | 253 | 1 | 364 |
| 聚类5 | 65 | 313 | 147 | 3 | 30 | 10 | 273 | 40 | 34 | 26 |
| 聚类6 | 0 | 7 | 64 | 24 | 744 | 13 | 13 | 0 | 27 | 87 |
| 聚类7 | 477 | 0 | 68 | 5 | 0 | 0 | 4 | 406 | 1 | 97 |
| 聚类8 | 31 | 190 | 109 | 11 | 14 | 4 | 564 | 25 | 4 | 27 |
| 聚类9 | 314 | 14 | 22 | 1 | 1 | 5 | 10 | 390 | 0 | 257 |