大数据分析与知识发现实验报告

西安交通大學

Kmeans 算法设计和实现

自动化 94--胡欣盈--2194323176

目　录

一、实验目的

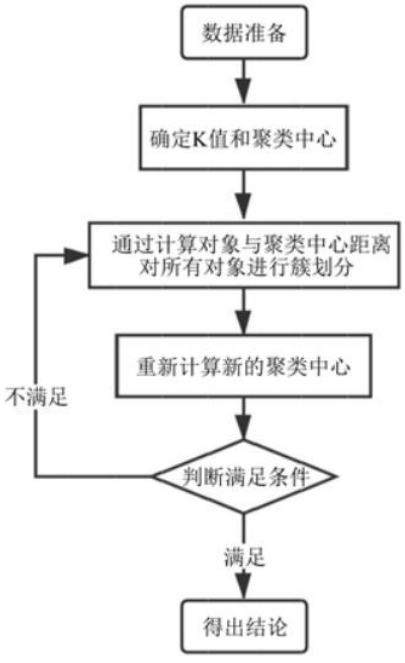
1、编写 K-means 算法实现对某数据集的聚类，对不同 K 值的聚类准确度进行比较；

2、通过实验加强对 k-means 聚类算法的理解，提升实践能力。

二、算法原理

Kmeans 算法是最常用的聚类算法，主要思想是:在给定 K 值和 K 个初始类簇中心点的情况下，把每个点(亦即数据记录)分到离其最近的类簇中心点所代表的类簇中，所有点分配完毕之后，根据一个类簇内的所有点重新计算该类簇的中心点(取平均值)，然后再迭代的进行分配点和更新类簇中心点的步骤，直至类簇中心点的变化很小，或者达到指定的迭代次数。

算法流程图与伪代码如下图所示，算法主要通过循环实现，停止迭代的条件是均值向量不再更新。循环主体是：将数据点分配到距离最短的聚类中心的簇中，更新聚类中心，然后再次进行分配。



输入：样本集 $D=\{x_1,x_2,x_3,\ldots,x_m\}$；聚类簇数k.
过程：
1：从D中随机选择k个样本作为初始均值向量 $\{\mu_1,\mu_2,\mu_3,\ldots,\mu_k\}$
2：**repeat**
3：令 $C_i=\varnothing(1\leqslant i\leqslant k)$
4：**for** *j=1,2,...,m* **do**
5：计算样本 x_j 与各均值向量 $\mu_i(1\leqslant i\leqslant k)$ 的距离：$d_{ji}=\|x_j-\mu_i\|_2$；
6：*根据距离最近的均值向量确定* x_j 的簇标记：$\lambda_j=argmin_{i\in\{1,2,3,\ldots,k\}}d_{ji}$；
7：将样本 x_j 划入相应的簇：$C_{\lambda_j}=C_{\lambda_j}\cup\{x_j\}$；
8：**end for**
9：**for** i=1,2,...,k **do**
10: 计算新均值向量：$\mu_i^{'}=\frac{1}{|C_i|}\sum_{x\in Ci}x$；
11：**if** $\mu_i^{'}\neq\mu_i$ **then**
12：将当前均值向量 μ_i 更新为 $\mu_i^{'}$
13：**else**
14：保持当前均值不变
15：**end if**
16：**end for**
17：**until** 当前均值向量均未更新
输出：簇划分 $C=\{C_1,C_2,\ldots,C_k\}$

三、实验内容与结果

（1）数据集介绍与预处理

实验中使用的是 Iris 数据集（安德森鸢尾花卉数据集）进行聚类并计算聚类的准确率。Iris 包含 150 个样本，每个样本包含了花萼长度、花萼宽度、花瓣长度、花瓣宽度四个特征（前 4 列），也包含了品种信息，即目标属性（第 5 列，也叫 target 或 label）。

为方便使用 Iris 数据集经过了一些整理，这里将最后一列的带字符串的标签 Iris-setosa，Iris-versicolor，Iris-virginica 分别用数字 1,2,3 代替并移到了第一列。整理后的部分数据集如下图所示。

	A	B	C	D	E
1	1	5.1	3.5	1.4	0.2
2	1	4.9	3	1.4	0.2
3	1	4.7	3.2	1.3	0.2
4	1	4.6	3.1	1.5	0.2
5	1	5	3.6	1.4	0.2
6	1	5.4	3.9	1.7	0.4
7	1	4.6	3.4	1.4	0.3
8	1	5	3.4	1.5	0.2
9	1	4.4	2.9	1.4	0.2

（2）算法编写

为了方便应用我们将其编写为一个 M 函数 KMeans()，输入输出参数如下表所示，关键代码如下图所示：

输入		输出	
Data	不带分类标号的数据	Idx	返回的分类标号
K	分类数	centroids	每一类的中心
iniCentriods	自行指定初始聚类中心	Distance	类内总距离
Iterations	迭代次数		

```
%% 迭代
for iter=1:iterations
    pre_centroids=centroids;% 上一次求得的中心位置

    tags=zeros(numOfData,K);
    %% 寻找最近中心，更新中心
    for i=1:numOfData
        D=zeros(1,K);% 每个数据点与每个聚类中心的标准差
        Dist=D;

        % 计算每个点到每个中心点的标准差
        for j=1:K
            Dist(j)=norm(data(i,:)-centroids(j,:),2);
        end

        [minDistance,index]=min(Dist);% 寻找距离最小的类别索引
        tags(i,index)=1;% 标记最小距离所处的位置（类别）
    end
```

```
%% 取均值更新聚类中心点
for i=1:K
    if sum(tags(:,i))~=0
        % 未出现空类，计算均值作为下一聚类中心
        for j=1:numOfAttr
            centroids(i,j)=sum(tags(:,i).*data(:,j))/sum(tags(:,i));
        end
    else % 如果出现空类，从数据集中随机选中一个点作为中心
        randidx = randperm(size(data, 1));
        centroids(i,:) = data(randidx(1),:);
        tags(randidx,:)=0;
        tags(randidx,i)=1;
    end
end


if sum(norm(pre_centroids-centroids,2))<0.001  % 不断迭代直到位置不再变化
    break;
end
```

```
%% 计算输出结果
Distance=zeros(numOfData,1);
Idx=zeros(numOfData,1);
for i=1:numOfData
    D=zeros(1,K);% 每个数据点与每个聚类中心的标准差
    Dist=D;
    % 计算每个点到每个中心点的标准差
    for j=1:K
        Dist(j)=norm(data(i,:)-centroids(j,:),2);
    end

    [distance,idx]=min(Dist);% 寻找距离最小的类别索引
    distance=Dist(idx);

    Distance(i)=distance;
    Idx(i)=idx;
end
Distance=sum(Distance,1);% 计算类内总距离
end
```

（3）调用与结果

3.3.1 指定初始聚类中心

读取 Iris 数据集，自行指定初始聚类中心调用前面编写的 KMeans 函数进行聚类，然后计算聚类的准确率，其代码如下

```
clear
data=load('Iris .txt');
data=data(:,2:end);

matrix=[5.9016,2.7484,4.3935,1.4339;6.8500,3.0737,5.7421,2.0711;5.0060,3.4280,1.4620,0.2460];
[Idx,C,distance]=KMeans(data,3,matrix,500);
Distance=sum(distance)

c1=Idx(1:50,1);c2=Idx(51:100,1);c3=Idx(101:150,1);
accuracy=(sum(c1==mode(Idx(1:50,1)))+sum(c2==mode(Idx(51:100,1)))+sum(c3==mode(Idx(101:150,1))))/150
```

准确率的计算：因为不能直接用 KMeans 计算后得到的标号跟原数据集中的标号对比计算准确率，KMeans 只需要也只能将那些“相似”的数据点聚集到一类中，而给这一类数据的标号却是可能跟原数据集不同的。采用一个简单的方法，从原数据集的标签可以看出第 1-50 个数据点为一类（Iris-setosa），第 51-100 为一类（Iris-versicolor），第 101-150 为一类（Iris-virginica），因此只需确定每 50 个数据点中的聚类标号是不是一致。取它们之中数目最多的标号作为正确的个数，最终比上数据集的总数即为准确率。以上代码运行结果如下所示。

```
Distance =

   97.3259


accuracy =

    0.8933
```

3.3.2 随机产生初始聚类中心

KMeans 算法本身思想比较简单，但是合理的确定 K 值和 K 个初始类簇中心点对于聚类效果的好坏有很大的影响。最简单的确定初始类簇中心点的方法是随机产生数据大小范围内的 K 个点作为初始的簇类中心点。随机产生初始点并进行测试的程序代码如下，改变 K 值运行几次以下代码，可以看出由于初始点事随机选取的每次运行得到的结果有所差异。

```
clear
data=load('Iris .txt');
data=data(:,2:end);
K=2;
```

```
%% 产生随机初始点
[numOfData,numOfAttr]=size(data);   % numOfData是数据个数，numOfAttr是数据维数

centroids=zeros(K,numOfAttr);       % 随机初始化，最终迭代到每一类的中心位置
maxAttr=zeros(numOfAttr);          % 每一维最大的数
minAttr=zeros(numOfAttr);          % 每一维最小的数
for i=1:numOfAttr
    maxAttr(i)=max(data(:,i));     % 每一维最大的数
    minAttr(i)=min(data(:,i));     % 每一维最小的数
    for j=1:K
        centroids(j,i)=maxAttr(i)+(minAttr(i)-maxAttr(i))*rand();  % 随机初始化，选取每一维[min max]中初始化
    end
end

[Idx,C,distance]=KMeans(data,K,centroids,500);% 调用KMeans
Distance=sum(distance)% 计算类内距离之和
```

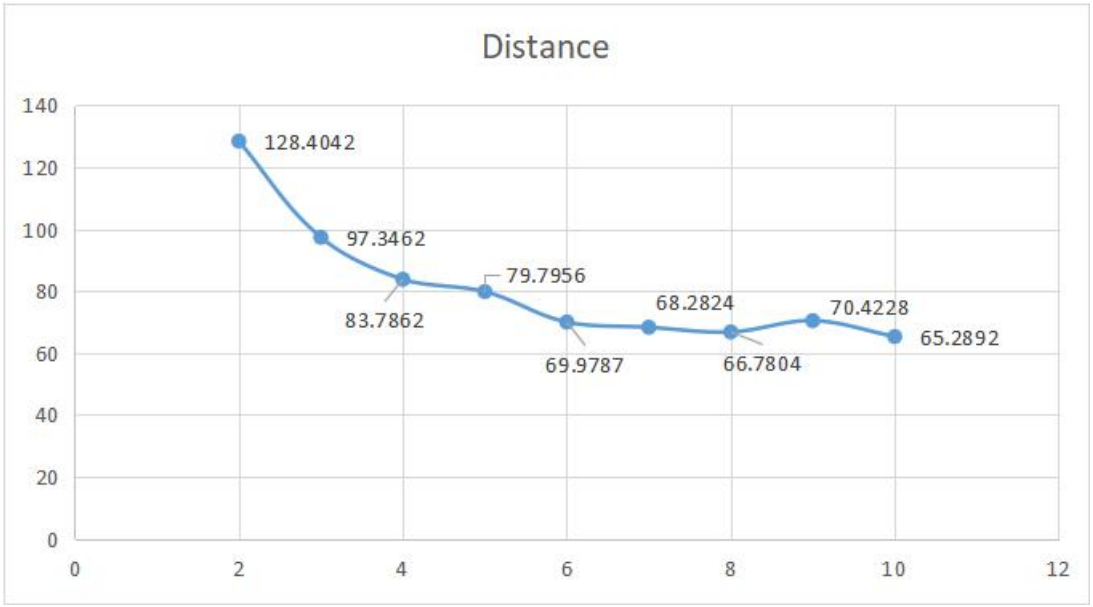
```
%% 计算准确率
c1=Idx(1:50,1);c2=Idx(51:100,1);c3=Idx(101:150,1);
Accuracy=(sum(c1==mode(Idx(1:50,1)))+sum(c2==mode(Idx(51:100,1)))+sum(c3==mode(Idx(101:150,1))))/numOfData
```

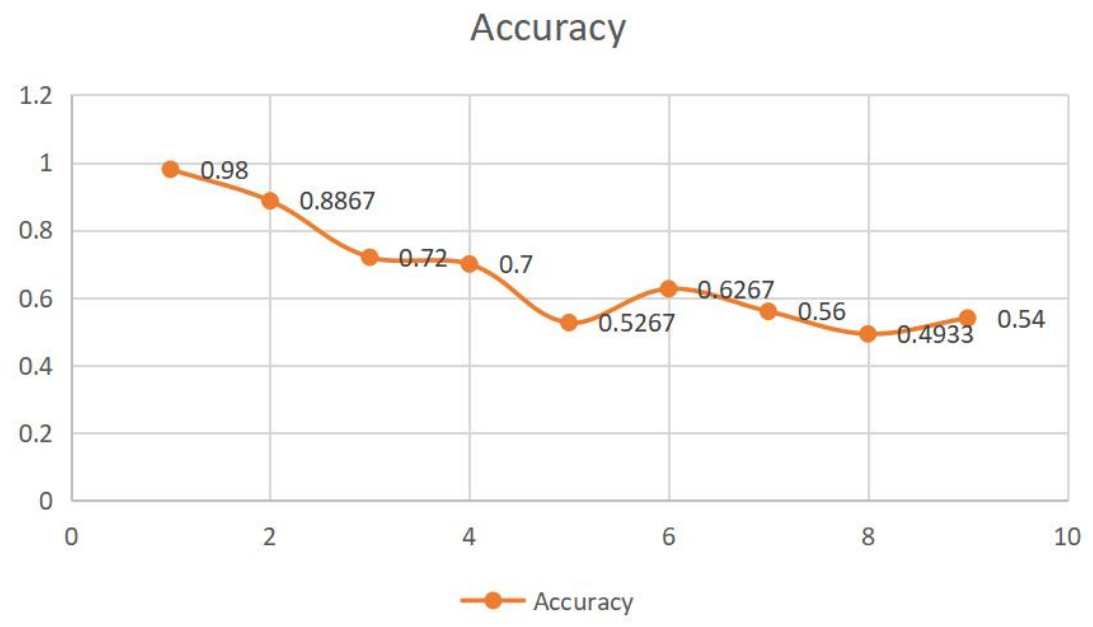
四、实验结果分析

（1）K 值的影响

调整 k 值，观察指标变化如下图所示。当 K 值与实际类别相差较多时，聚类精确度意义不大，随着 K 增大，类内距离有减小的趋势，这与经验相符。

	A	B	C	D	E	F	G	H	I	J
1	K值	2	3	4	5	6	7	8	9	10
2	Distance	128.4042	97.3462	83.7862	79.7956	69.9787	68.2824	66.7804	70.4228	65.2892
3	Accuracy	0.98	0.8867	0.72	0.7	0.5267	0.6267	0.56	0.4933	0.54





（2）算法优缺点分析

K-Means 的主要优点有：

1）原理比较简单，实现也是很容易，收敛速度快。

2）聚类效果较优。

3）算法的可解释度比较强。

4）主要需要调参的参数仅仅是簇数 k。

K-Means 的主要缺点有：

1）K 值的选取不好把握。

2）对于不是凸的数据集比较难收敛。

3）如果各隐含类别的数据不平衡，比如各隐含类别的数据量严重失衡，或者各隐含类别的方差不同，则聚类效果不佳。

4） 采用迭代方法，得到的结果只是局部最优。

5） 对噪音和异常点比较的敏感。

五、实验总结

通过本次实验，我对 K-means 算法有了更深入的了解，通过编程实现 K-means 和 K-means++算法，以对安德森鸢尾花卉数据集进行聚类分析。通过比较不同 k 值情况下的聚类结果，更深入的理解了 K 值选取对 K-means 聚类的影响，更加清晰了 K-means 算法的优缺点及适用情况。