k-means的分布式实现

2017年9月23日 星期六　　下午 9:42

问题：

1. 给定一个包括m个n维连续型变量的数据集X={Xi|i=1...m}, 用K-means聚类算法对X进行聚类（可任选一种编程语言）

要求：

a. 如果数据量很大，单台机器无法完成任务（数据无法完全倒入单台机器的内存，并且单台机器无法在合理的时间内完成相应的计算量），如何利用多台机器实现分布式的操作

b. 完成具体的代码(备注：机器之间的数据传输可以用伪代码)

k-means算法：

定义以所有训练样本为元素的集合为$\boldsymbol{X} = \{\boldsymbol{x}_i|\mathrm{i} = 1,2,\dots,m\}$,其中$\boldsymbol{x}_i$为某一个训练样本。

定义c_i为$\boldsymbol{x}_i$所属的类别。

k-means算法流程如下：

初始化k个中心点$\mu_1^{(1)},\mu_2^{(1)},\dots,\mu_{\mathrm{k}}^{(1)}$

重复直至收敛{

(第t次迭代)

Step1：$\forall i,\quad c_i^{(t+1)} = \mathrm{argmin}_j\,||\lambda_i - u_j^{(t)}||_2$

Step2：$\forall j,\quad \mu_{\mathrm{j}}^{(\mathrm{t}+1)} = \frac{\sum_{i=1}^{m} 1\left\{c_i^{(t+1)}=j\right\}x_i}{\sum_{i=1}^{m} 1\left\{c_i^{(t+1)}=j\right\}}$

}

收敛条件：

$\forall i,\quad c_i^{(t+1)} = c_i^{(t)}$

使用的初始化方法：

1：随机选取一个样点作为中心点$\mu_1^{(1)}$

重复直至找到k个中心点{

Step1：对每个$\boldsymbol{x}_i$计算它到各个中心点的距离，取这些距离的最小值，记为$\boldsymbol{d}_i$

Step2：选择最大的$\boldsymbol{d}_i$对应的样点$\boldsymbol{x}_i$作为新的中心点。

}

分布式的k-means：

分布式的计算需要对样本进行拆分。现将样本集合$\boldsymbol{X}$划分为互不相交的子集$T_0,T_1,\dots,T_{\mathrm{L}-1}$，这些样本子集将被作为分布式系统中各个计算机的训练样本。总共有L台计算机。

算法的迭代部分的第一步，若每台计算机存储有更新后的中心点，那么它在各台计算机内部运行就能实现。

算法的迭代部分的第二步，这一步需要计算机间协作才能完成。具体分析如下：

$$\mu_{\mathrm{j}}^{(\mathrm{t}+1)} = \frac{\sum_{i=1}^{m} 1\left\{c_i^{(t+1)}=j\right\}x_i}{\sum_{i=1}^{m} 1\left\{c_i^{(t+1)}=j\right\}}$$

其中

$$\sum_{i=1}^{m} 1\left\{c_i^{(t+1)}=j\right\} = \sum_{l=0}^{L-1}\sum_{x_i\in T_l} 1\left\{c_i^{(t+1)}=j\right\}$$

$$\sum_{i=1}^{m} 1\left\{c_i^{(t+1)}=j\right\}x_i = \sum_{l=0}^{L-1}\sum_{x_i\in T_l} 1\left\{c_i^{(n+1)}=j\right\}x_i$$

$\sum_{x_i \in T_l} 1\left\{ c_i^{(t+1)} = j \right\} = CN_{l,j}$表示第$l$号计算机中类别为j的样本数目。

$\sum_{x_i \in T_l} 1\left\{ c_i^{(t+1)} = j \right\} x_i = CX_{l,j}$ 表示第l号计算机中类别为j的样本特征的和。

所有计算机在迭代的第二步时计算出类别为1,2,3,...,k的样本数目（1*k维向量）、第l号计算机中类别为1,2,3,...,k的样本特征的和（n*k维矩阵）。 然后将这两个变量通过数据传输发送给某一台指定的计算机（这里指定这台计算机为第0号计算机），然后0号计算接收其他计算机发来的数据计算

$$\forall j, \mu_{\mathrm{j}}^{(\mathrm{t}+1)} = \frac{\sum_{l=0}^{L-1} CX_{l,j}}{\sum_{l=0}^{L-1} CN_{l,j}}$$

然后计算机0将得到的$\mu_1^{(t+1)}, \mu_2^{(t+1)}, \ldots, \mu_{\mathrm{k}}^{(t+1)}$通过数据传输发送给所有其他计算机。此时算法的一次迭代完成。

★ k-means迭代部分的分布式实现：

（第t次迭代）

Step1：

对每一台计算机，计算它所有样本点到各个中心点的距离，并将每一个样本纳入到它距离最近中心点所对应的类别。并检测样本点的类别是否有改动，生成changeMark变量。

Step2：

1、每一台计算机计算出类别为1,2,3,...,k的样本数目$CN_{l,j}$。

2、每一台计算机计算出类别为1,2,3,...,k的样本特征的和$CX_{l,j}$。

3、每一台计算机将上面两个变量和changeMark通过数据传输发送到第0号计算机。

4、第0号计算机接收数据。

5、第0号计算机根据每台计算机发送的changeMark判断是否收敛，如果收敛则中止程序，否则：

6、第0号计算机接收到其他计算机发来的数据，通过计算下式更新中心点。

$$\forall j, \qquad \mu_{\mathrm{j}}^{(\mathrm{t}+1)} = \frac{\sum_{l=0}^{L-1} CX_{l,j}}{\sum_{l=0}^{L-1} CN_{l,j}}$$

7、第0号计算机将更新后的中心点$\mu_1^{(t+1)}, \mu_2^{(t+1)}, \ldots, \mu_{\mathrm{k}}^{(t+1)}$通过数据传输发送给所有其他计算机。

8、所有计算机接收来自第0号计算机发来的更新后的中心点数据。

★ k-means初始化的分布式实现：

1：随机选取一个中心点$\mu_1^{(1)}$，存储在第0号计算机上

重复直至找到k个中心点{

1、第0号计算机将新的中心点发送给其他计算机。

2、其他计算机接收来自第0号计算机的新的中心点的数据。

3、对每一台计算机，计算它的每一个样本到目前已确定的各个中心点的距离，取其中最短距离。

4、每一台计算机选出最短距离最大的样本点。

5、每一台计算机将第二步中的样本点和它对应的最短距离发送给第0号计算机。

6、第0号计算机接收所有其他计算机发来的数据。

7、第0号计算机在接收到的数据中选择最短距离最大的样本点作为新的中心点。

}

代码部分：

代码部分使用python语言编写。为了模拟分布式计算的情况，对样本进行了分批次处理。使用欧氏距离。

代码由两个文件组成：main.py和BatchKmeansProcess.py。

BatchKmeansProcess.py文件创建了类BatchKmeansProcess。创建类BatchKmeansProcess的目的是希望它能清晰地模拟多台计算机分

布式地计算K-Means算法。每个BatchKmeansProcess实例表示分布式系统中的某一台计算机。它的成员变量和成员函数表示每一台计算机内部运算时所需的变量和操作。分布式系统中计算机间的数据通信对应着实例之间的参数传递模拟。

运行main.py后，会对随机产生的样本进行k-means分类。在文件中可以设置样本数目，样本维度，分类数目，批次数等参数

main.py整体代码为3个部分：

1、准备阶段：分布式计算机的生成，训练样本的生成和划分。
2、K-means算法部分。
3、作图部分。

各个部分开头都会有注释标出