第9章 聚类算法

《人工智能算法》

清华大学出版社
2022年7月

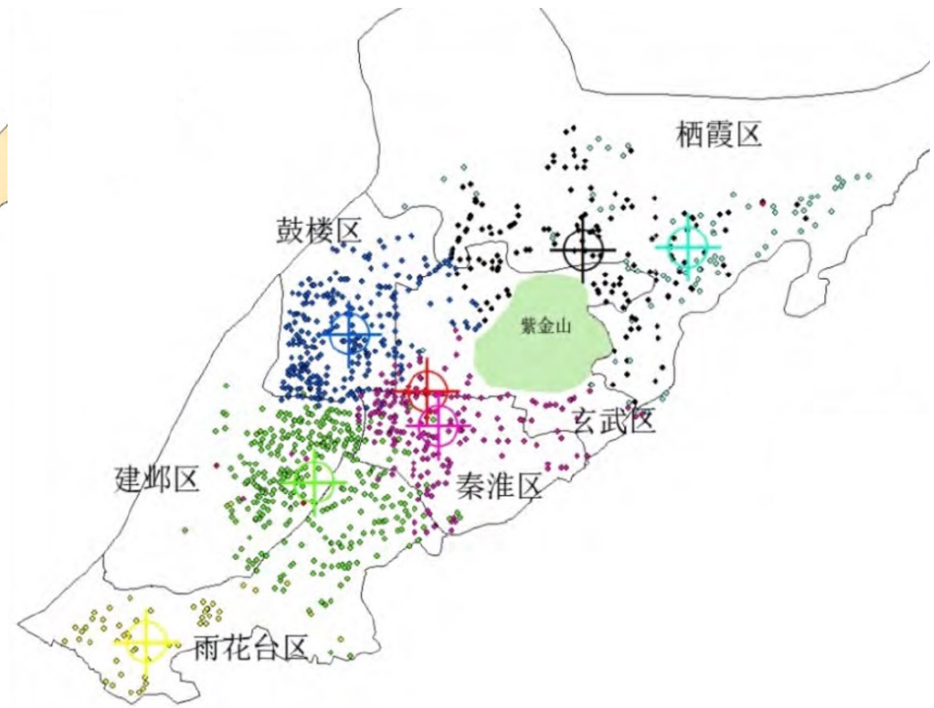
提纲

- 引例
- 聚类算法的基本思想
- 聚类算法分类
- k-均值算法
- 基于MapReduce的k-均值并行聚类算法
- 总结

引例

- 共享单车停放点问题

共享单车分布示意图



共享单车停放站点示意图

- 空间上呈现数量多、较为聚集的特点

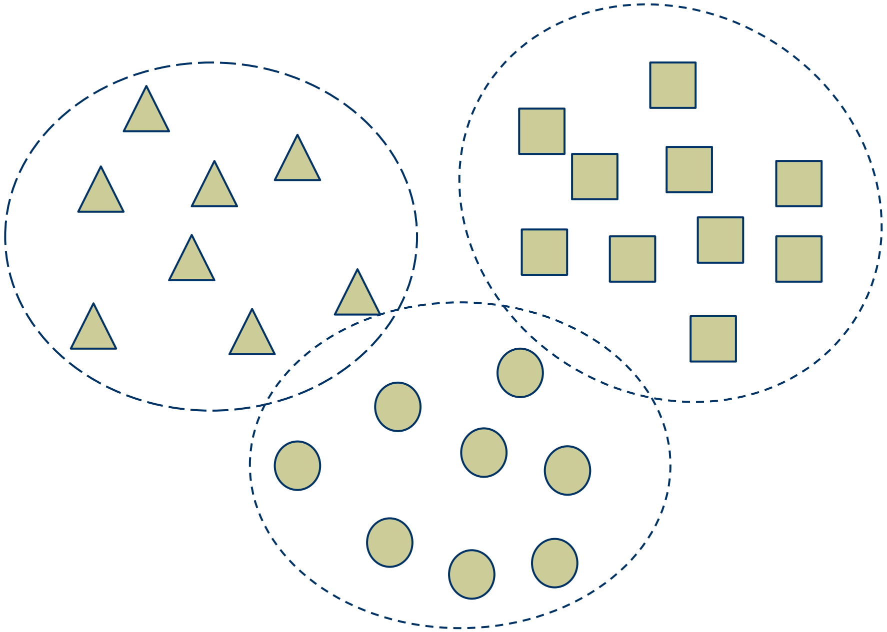
- 聚集区域可视为共享单车停放站点
- 利用聚类分析技术找到聚集区域中心点

提纲

- 引例
- 聚类算法的基本思想
- 聚类算法分类
- k-均值算法
- 基于MapReduce的k-均值并行聚类算法
- 总结

聚类算法的基本思想(1)

- **聚类目标**
  - 将一组给定的数据对象划分为多个互不相交的子集，每个子集称为一个簇（Cluster）。
  - 簇内数据对象之间相似度高，簇间数据对象之间差异性大。



聚类算法的基本思想(2)

- **定义数据对象之间的相似度**
  - 闵可夫斯基距离（Minkowski Distance）
  - 欧氏距离（Euclidean Distance）
  - 曼哈顿距离（Manhattan Distance）
- **聚类目标函数（聚类停止判别条件）**
  - 判断多个划分结果哪个是有效的
  - 划分结果达到聚类目标函数时终止算法运行
- **簇别划分策略（算法）**
  - 通过何种簇别划分方式使得划分结果达到目标函数

提纲

- 引例
- 聚类算法的基本思想
- 聚类算法分类
- k-均值算法
- 基于MapReduce的k-均值并行聚类算法
- 总结

聚类算法分类 (1)

传统聚类算法

- **基于划分的聚类算法**
  - 先将数据集任意划分为k个不相交的簇
  - 迭代优化逐步改善簇的划分
  - 目标函数收敛时，得到最终的聚类结果

  k-均值（k-Means）算法、最大最小距离（Max-Min Distance）算法
- **基于密度的聚类算法**
  - 通过数据密度（单位区域内的实例数）来发现任意形状的类簇

聚类算法分类 (2)

传统聚类算法

- **层次聚类算法**

（1）自底向上的聚合型层次聚类

– 先将每个数据对象作为一个聚类簇

– 计算簇间的相似度进行分层合并，直至最后只有一个簇或满足目标函数时终止

（2）自顶向下的分裂型层次聚类

– 先将所有数据对象看作一个聚类簇

– 逐层分裂，直至每个簇中只包含一个数据对象或满足满足目标函数时终止

聚类算法分类 (3)

传统聚类算法

- **基于网格的聚类算法**
  - 先将数据空间划分为网格单元，并将数据对象映射到网格单元
  - 判断每个网格单元是否形成类簇
- **基于模型的聚类算法**
  - 先为每个聚类假设一个模型
  - 发现符合模型的数据对象

聚类算法分类 (4)

智能聚类算法

- **大数据聚类算法**

（1）分布式聚类（Distributed Clustering）算法

  - 使用MapReduce框架对传统聚类算法进行扩展

（2）并行聚类（Parallel Clustering）算法

  - 使用并行框架对传统聚类算法进行扩展

- **基于深度学习的聚类算法**
  - 利用深度学习模型将高维的原始数据映射为低维特征向量
  - 再利用特征向量进行聚类

提纲

- 引例
- 聚类算法的基本思想
- 聚类算法分类
- k-均值算法
- 基于MapReduce的k-均值并行聚类算法
- 总结

k-均值算法 (1)

基本思想：

- 基于划分的聚类算法。以k为参数，将n个数据对象划分为k个簇。
- 簇内数据对象之间具有较高的相似性，簇间数据对象之间具有较低的相似度。相似度基于簇内数据对象的平均值来计算。
- 给定数据集$D=\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$，簇的数目$k$，k-均值算法针对聚类所得的$k$个簇划分$C=\{c_1, c_2, \cdots, c_k\}$，最小化平方误差

$$J(C)=\sum_{j=1}^{k} J(c_j)=\sum_{j=1}^{k}\sum_{\mathbf{x}_i\in c_j}\left\|\mathbf{x}_i-\mathbf{r}_j\right\|^2$$

其中， $\mathbf{r}_j$ 是簇c_j的均值向量

- $J(C)$值在一定程度的上刻画了簇内数据对象围绕簇中心点$\mathbf{r}_j$的紧密程度，$J(C)$值越小，簇内数据对象相似度越高。

k-均值算法 (2)

- **Step1** 指定需要划分簇的个数k值
- **Step2** 随机选择k个数据对象作为初始的簇中心点
- **Step3** 计算其余数据对象到k个簇中心点的欧式距离，将其划分到最近的簇中
- **Step4** 调整新簇，并重新计算每个簇的平均值
- **Step5** 计算聚类目标函数$J(C)$，若不满足收敛条件，重复Step2-Step4

k-均值算法 (3)

k-Means (D, k)

Repeat

$C_j \leftarrow \emptyset \ (1 \le j \le k)$

For $i \leftarrow 1$ **To** n **Do**

$d_{ij} \leftarrow \|\mathbf{x}_i - \mathbf{r}_j\|_2$

$\lambda_i \leftarrow \mathrm{argmin}_{j \in \{1,2,\cdots,k\}} d_{ij}$

$C_{\lambda_i} \leftarrow C_{\lambda_i} \cup \{\mathbf{x}_i\}$

End For

For $j \leftarrow 1$ **To** k **Do**

$\mathbf{r}_j^* = \frac{1}{|C_j|} \sum_{\mathbf{x} \in C_j} \mathbf{x}$

If $\mathbf{r}_j \neq \mathbf{r}_j^*$ **Then**

$\mathbf{r}_j \leftarrow \mathbf{r}_j^*$

End If

End For

Until $\{\mathbf{r}_j\}$未发生变化

时间复杂度：
$O(n \times k \times t)$
t为迭代次数

k-均值算法 (4)

- **K-均值聚类示例**

考虑二维空间中的数据集$D=\{\mathbf{x}_1=(2, 3), \mathbf{x}_2=(1, 2), \mathbf{x}_3=(1, 1), \mathbf{x}_4=(2, 2), \mathbf{x}_5=(4, 2), \mathbf{x}_6=(4, 1), \mathbf{x}_7=(5, 1)\}$，假设$k=2$，初始时随机选择2个簇的中心，$\mathbf{r}_{10}=\mathbf{x}_1=(2, 3)$，$\mathbf{r}_{20}=\mathbf{x}_2=(1, 2)$

（1）第一趟计算

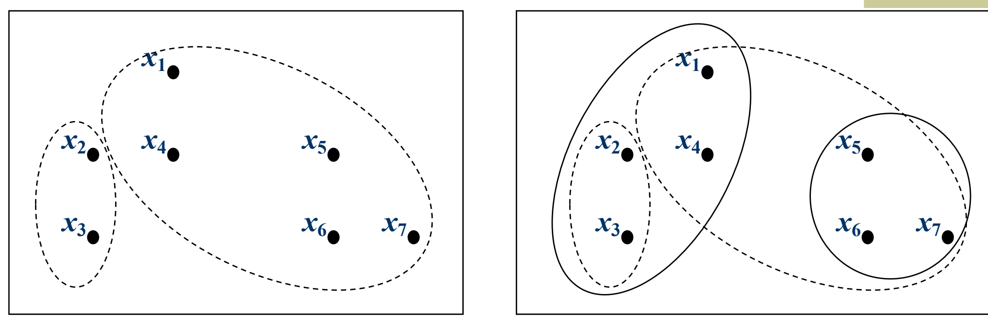
$d(\mathbf{x}_3, \mathbf{r}_{10})=\sqrt{5}$，$d(\mathbf{x}_3, \mathbf{r}_{20})=1$ 则 $\mathbf{x}_3 \Rightarrow c_2$

类似计算得到 $\Rightarrow$ $c_1=\{\mathbf{x}_1, \mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7\}$ $c_2=\{\mathbf{x}_2, \mathbf{x}_3\}$

重新调整簇中心 $\Rightarrow$ $\mathbf{r}_1=(\mathbf{x}_1+\mathbf{x}_4+\mathbf{x}_5+\mathbf{x}_6+\mathbf{x}_7)/5=(3.4, 1.8)$，$\mathbf{r}_2=(\mathbf{x}_2+\mathbf{x}_3)/2=(1.0, 1.5)$

$\mathbf{r}_{10} \neq \mathrm{r}_1$， $\mathrm{r}_{20} \neq \mathbf{r}_2$ $\Rightarrow$ 不满足收敛性

k-均值算法 (5)



（2）第二趟计算

计算各点到两个簇中心r_1和r_2的距离 ⟹ $c_1=\{\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7\}$， $c_2=\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$

重新调整簇中心 ⟹ $\mathbf{r}_1=(4.333, 1.333)$， $\mathbf{r}_2=(1.5, 2.0)$ ⟹ 不满足收敛性

（3）第三趟计算

产生了与第二趟相同的簇 ⟹ 收敛性满足

得到最终的聚类结果 ⟹ $c_1=\{\mathbf{x}_5, \mathbf{x}_6, \mathbf{x}_7\}$， $c_2=\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$

提纲

- 引例
- 聚类算法的基本思想
- 聚类算法分类
- k-均值算法
- 基于MapReduce的k-均值并行聚类算法
- 总结

基于MapReduce的k-均值并行聚类算法 (1)

分而治之

- **面向大规模数据的聚类算法**
- **基于MapReduce框架**
  - Map将输入数据转化为<key, value>序列
  - Combine阶段对Map阶段的输出结果进行合并和处理
  - Reduce阶段并行地合并聚类结果

基于MapReduce的k-均值并行聚类算法 (2)

- **Map阶段**

Step1： Map函数以<key, value>对的形式读入待处理数据集

Step2： 分布式缓存中取出上一轮聚类的k个簇中心点

Step3： k-均值聚类算法将数据对象划分到与其距离最近的簇中

Step4： 输出中间结果至Combine函数

Map ({r_j}, <key, value>**)**

For j=1 **To** k **Do**

  If $d(\mathbf{r}_j, instance) < minDistance$ **Then**

    $minDistance \leftarrow d(\mathbf{r}_j, instance)$

    $index \leftarrow \mathbf{r}_j$

  End If

End For

$key' \leftarrow index$

$value' \leftarrow instance$

基于MapReduce的k-均值并行聚类算法 (3)

- **Combine阶段**

Step1： 从Map函数输出的value中提取所有数据对象

Step2： 合并属于同一簇中的数据对象

Step3： Combine函数统计同一簇的数据对象个数，并计算该簇所有数据对象的均值

Step4： 输出每个簇中心的局部聚类结果至Reduce函数

基于MapReduce的k-均值并行聚类算法 (4)

- **Reduce阶段**

Step1： 从Combine函数输出中提取所有的数据对象，并聚合所有簇中心的局部结果

Step2： 根据聚类结果重新计算出每个簇的中心点

Step3： 计算目标函数，若不满足收敛条件，执行下一次迭代

Reduce (<key, values>)

While *values*.hasNext() **Do**

从*values*.Next()读取数据对象*instance*

For i=1 **To** *dimension* **Do**

$S[i] \leftarrow S[i]+instance[i]$

End For

$num \leftarrow num+1$

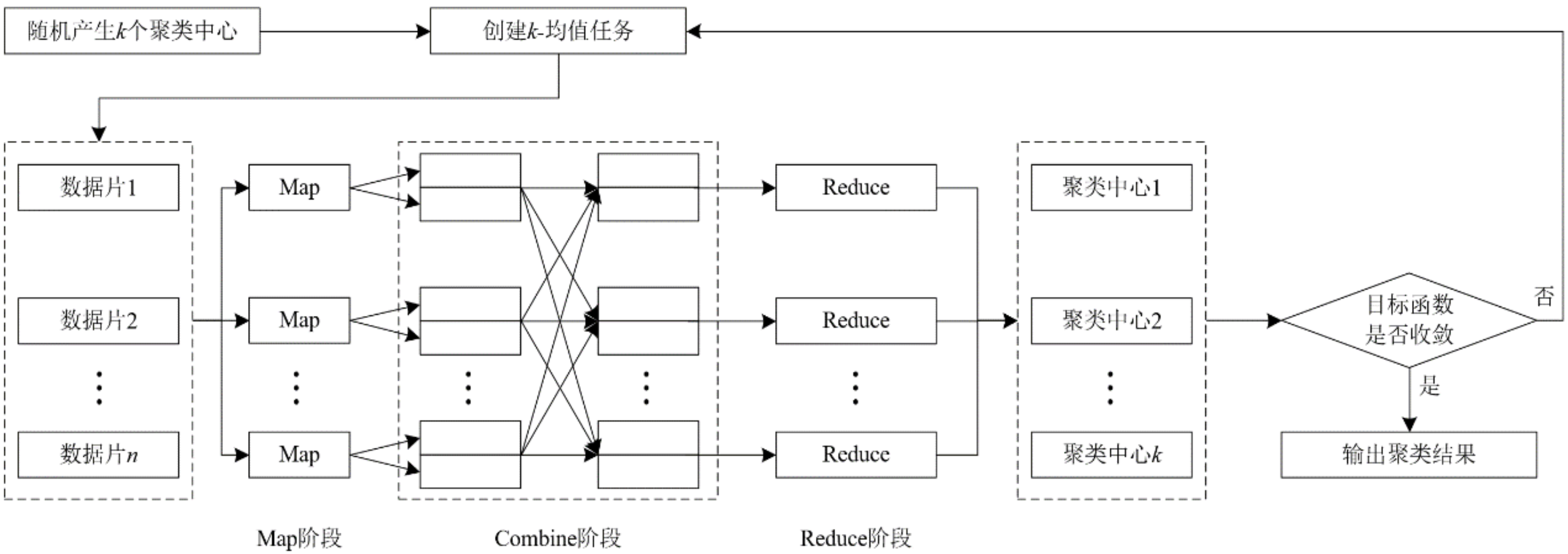
End While

For i=1 **To** *dimension*

$mean[i] \leftarrow S[i]/num$

End For

基于MapReduce的k-均值并行聚类算法 (5)



时间复杂度：

$O((N \times k \times t)/m)$

m为Map任务数

提纲

- 引例
- 聚类算法的基本思想
- 聚类算法分类
- k-均值算法
- 基于MapReduce的k-均值并行聚类算法
- 总结

总结

- 聚类算法的基本思想和分类
- k-均值算法的基本思想、算法步骤和改进策略
- k-均值算法的优缺点
- 基于MapReduce的k-均值并行聚类算法

  - Map阶段

  - Combine阶段

  - Reduce阶段

结语

谢谢！