算法知识点梳理(8) - KMeans算法评估及BIRCH聚类算法

不讲道理的瞬间 不讲道理的瞬间 2019-03-23

这是不讲道理的瞬间的第 **8** 次知识点学习

今天我们主要讨论一下如何评估**KMeans算法的效果**和概述**BIRCH聚类算法的原理**

01

KMeans

如何判断K均值算法的效果?

1. **inertia**：所有点到所属聚类中心的距离和的平均

```
model = KMeans(n_clusters=5).fit(X)
inertial = model.inertia_
```

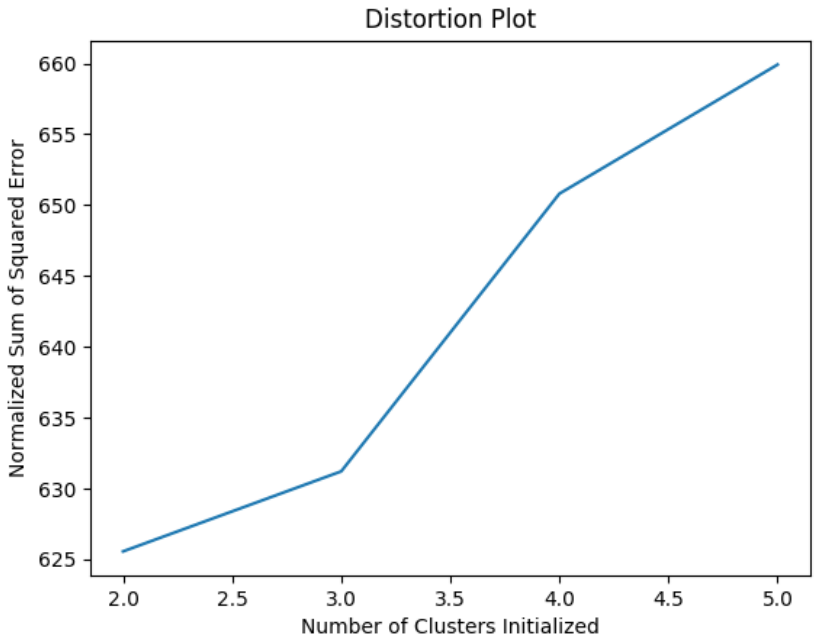
2. **轮廓系数(silhouette score)**：每个点i到同簇样本点的平均距离为ai， 到其它某簇样本点的平均距离的最小值为bi；然后根据以下公式计算轮廓系数，最终求得所有样本点的平均轮廓系数：如果si

等于1，说明样本i聚类合理；如果si等于-1，说明样本i应该被分类到其它的簇；如果si等于0，说明样本i应该在两簇的边界上。

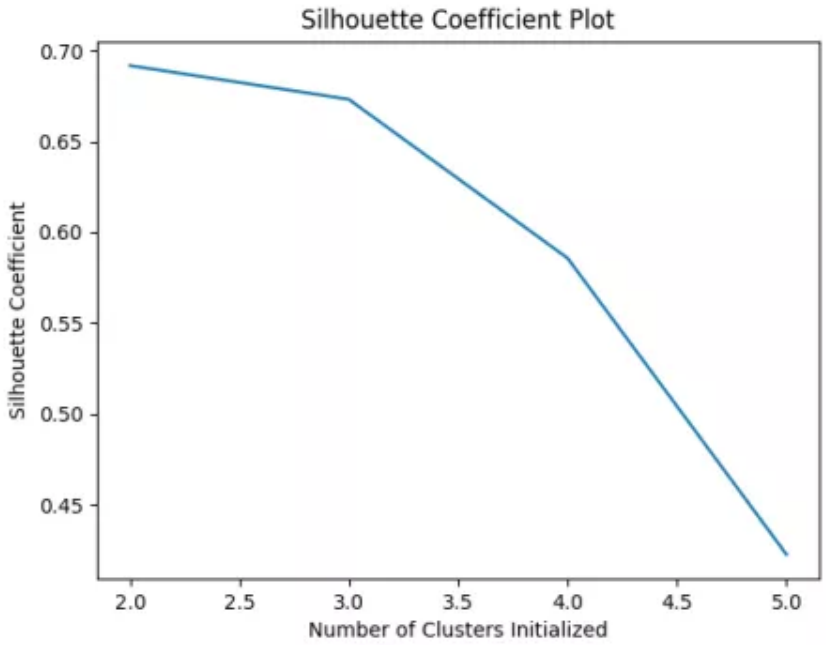
$$s(i)=\frac{b(i)-a(i)}{\max\{a(i),b(i)\}} \qquad s(i)=\begin{cases}1-\dfrac{a(i)}{b(i)}, & a(i)<b(i)\\ 0, & a(i)=b(i)\\ \dfrac{b(i)}{a(i)}-1, & a(i)>b(i)\end{cases}$$

如何选择最佳聚类个数K?

1. inertia：指定一个范围，对范围内的每个K值训练一个KMeans模型，得到inertia分数；选择最小inertial分数对应的K值；画出的折线图称为**肘部图(distorition plot)**



2. 轮廓系数(silhouette score): 指定一个范围，对范围内的每个K值训练一个KMeans模型，得到**轮廓系数**；选择最大轮廓系数对应的K值；画出对应折线图



02

BIRCH

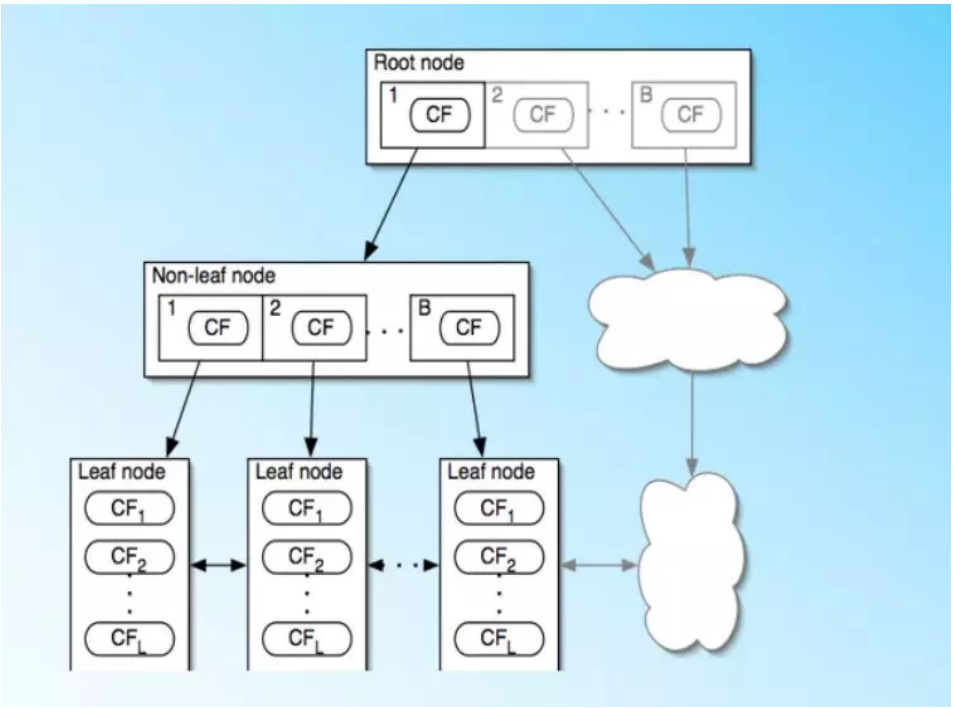
https://www.cnblogs.com/pinard/p/6179132.html

BIRCH概述

BIRCH全称Balanced Iterative Reducing and Clustering Using Herarchical, 中文翻译过来是利用层次方法的平衡迭代规约和聚类。

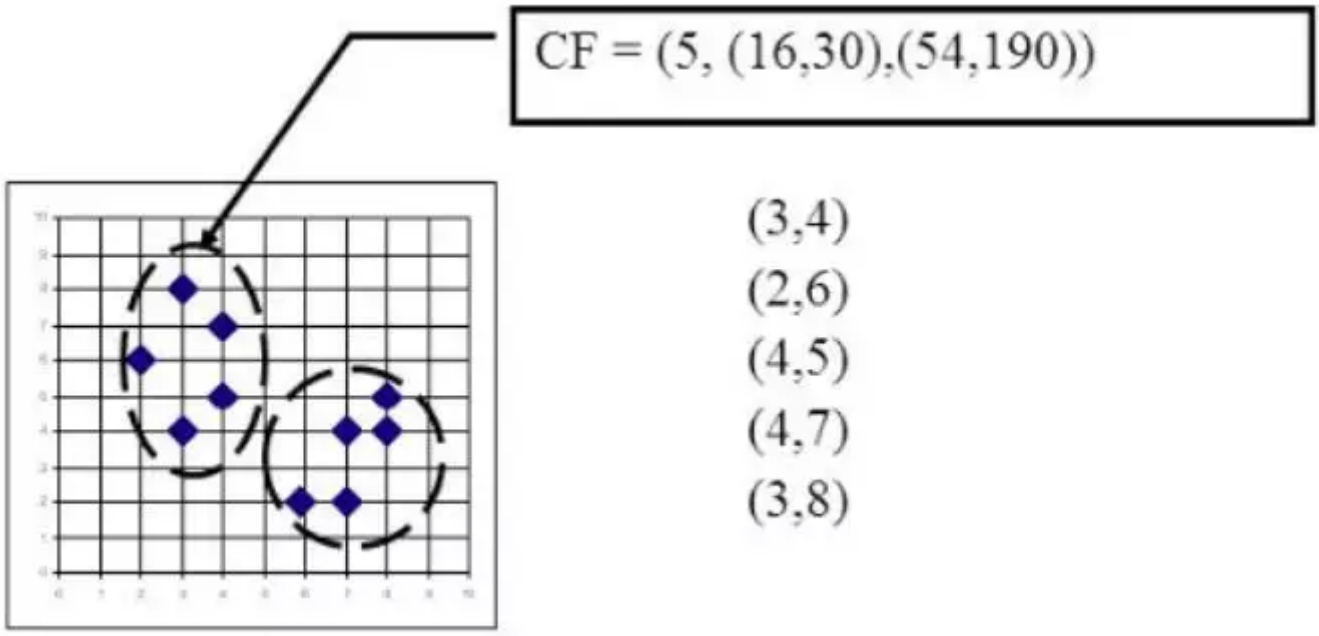
BIRCH算法适合于数据量大以及类别数K多的情况；它的运行速度很快，只需要单遍扫描数据就能进行聚类。

BIRCH算法利用聚类特征树(Clustering Tree)的结果帮助我们进行聚类，树的每个节点由若干个聚类特征(Cluster Features)构成



聚类特征CF概述

一个CF里有多个样本点，每个CF由一个三元组组成(N, LS, SS)：N代表CF中样本点个数，如下图N=5；LS代表拥有的样本点各特征维度的和向量；SS代表拥有的样本点各特征维度的平方和向量

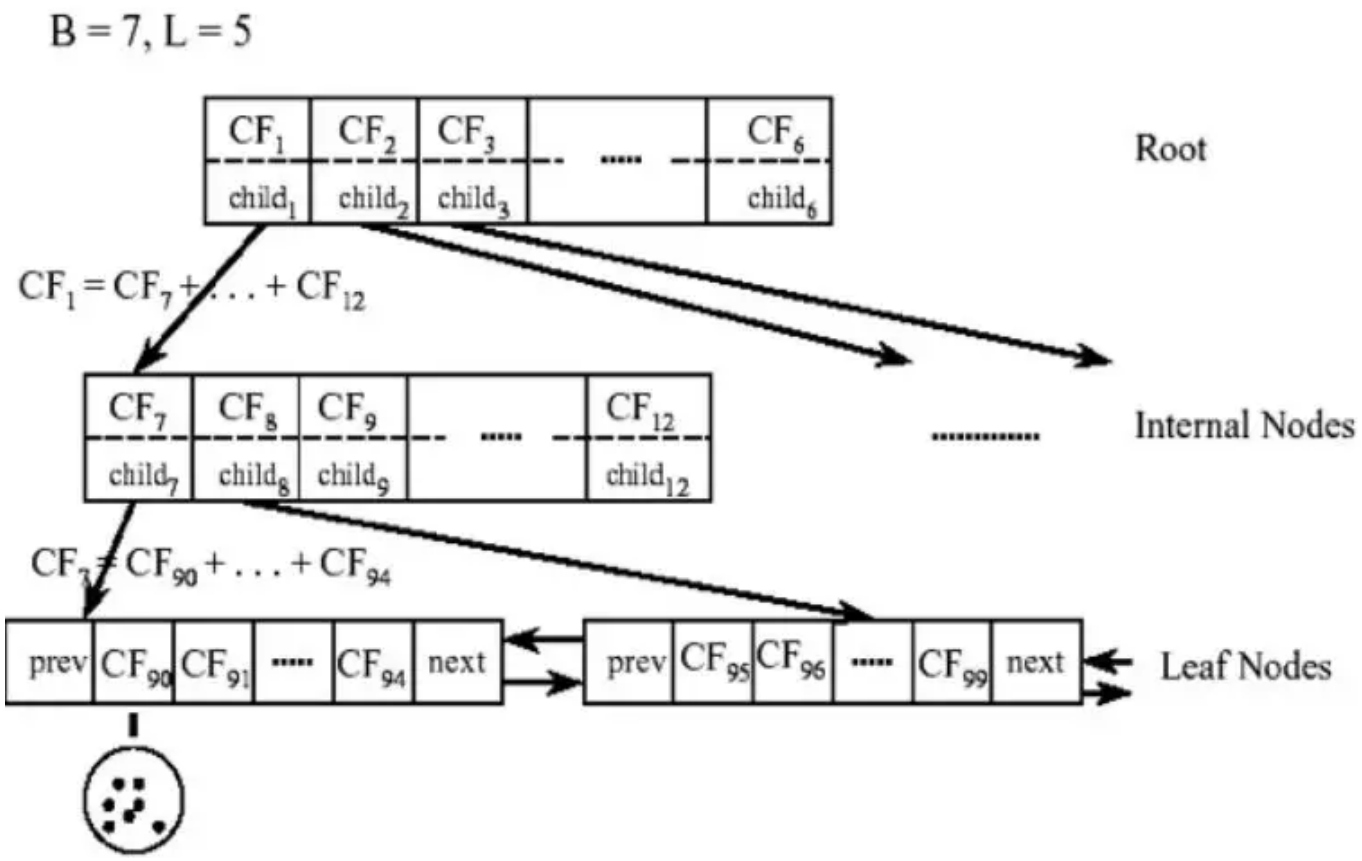


CF树的重要参数

1) 每个内部节点(除叶子节点以外的)的最大CF数B

2) 每个叶子节点的最大CF数L

3) 叶子节点每个CF的最大样本半径阈值T，也就是说，在这个CF里的所有样本点都要在半径小于T的超球体内



CF树的样本点插入

1) 从根节点向下寻找和新样本点距离最近的叶子节点和叶子节点里最近的CF节点
2) 如果新样本加入后，这个CF节点对应的超球体半径仍然小于阈值T，则更新路径上的所有CF三元组，插入结束，否则跳转到3)
3) 如果当前叶子节点的CF个数小于阈值L，则在该叶子节点创建新的CF节点，放入新样本，并更新路径上的所有CF三元组，插入结束，否则跳转到4)
4) 将当前叶子节点划分为两个新叶子节点，选择旧叶子节点中所有CF元组里超球体距离最远的两个CF元组，分布作为两个新叶子节点的第一个CF节点。将其他元组和新样本元组按照距离远近原则放入对应的叶子节点。依次向上检查父节点是否也要分裂，如果需要按和叶子节点分裂方式相同。

BIRCH算法的优缺点

优点：
1) 节省内存，所有样本点都在磁盘上，CF Tree仅仅保留了CF节点和对应的指针
2) 聚类速度快，只需要一遍扫描数据集就可以建立CF Tree

3) 可以识别噪音点

缺点：

1）由于CF Tree的节点的CF个数限制，导致聚类结果和真实类别分布不同

2) 对高维特征的聚类效果不好，此时建议使用Mini Batch K-Means

3) 如果数据集的分布簇不是类似于超球体，或者说不是凸的，则聚类效果不好