Spark mllib 分析报告-Kmeans

**一：算法介绍**

Kmeans 是一种聚类算法，是一种无监督学习方法。

**二：算法过程**

Kmeans 是一种迭代式算法。

算法初始化：

1>:可以随机初始化

2>:采用Kmeans|| , 是Kmeans++ 的变种

Kmeans|| 思想待学习\*\*\*\*

算法过程：

主要计算过程是迭代以下两步：

1：计算每个样本点所属的中心点，得到一些集合。

2：计算每个集合中的点的均值，得到新的样本点。

**三：算法评价**

评价标准：

1：收敛速度

2：正确性：SSE(sum of square error)

目前Mllib实现了SSE评价标准

**四：算法并行化**

Kmeans的并行化核心部分接住了RDD,从而实现计算过程的并行化。

计算过程：

初始化：

1：初始化中心点

迭代直至收敛或者精度达到SSE指定的精度：

1：broadcast中心点

2：计算sum and count of each points mapping to each center

rdd.mapPartition().reduceByKey(mergeFunc).collectAsMap()

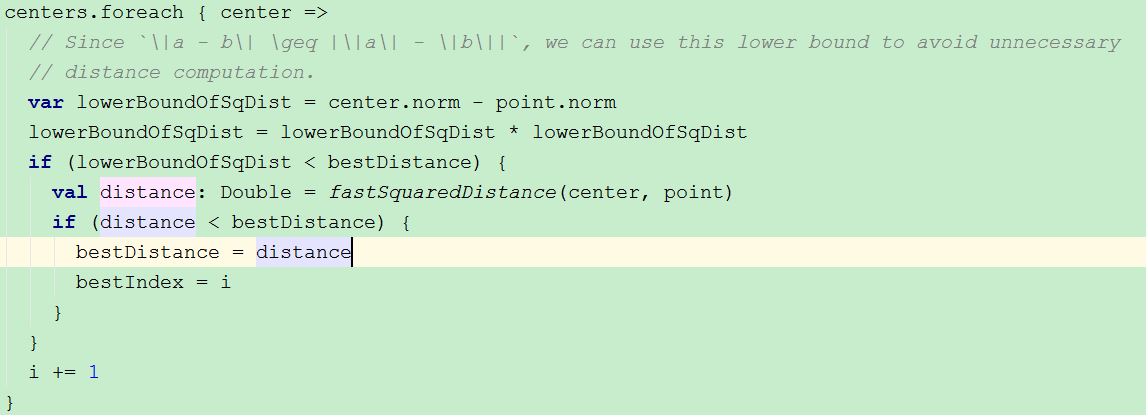
3：更新中心点：这一步在driver进行

但是应该没有driver瓶颈，因为每个executor 只返回K个（sum,count）这样的数据给driver

算法有多次迭代，所以RDD最好被cached

**算法中值得借鉴的地方：通过三角不等式和L2范数优化欧式距离的计算**

**比如：计算每个点到中心点的距离：**



**利用了三角不等式 |a - b| >= |a| -|b|**

**可以通过L2 范式快速计算出 |a| -|b|**

**另外fastSquareDistance 还可以根据范式快速的计算距离**

**五：数据流图**

**mapPartition -> reduceByKey -> Driver**

**六：测试**

|  |  |  |  |
| --- | --- | --- | --- |
| **DataSet Size** | **Configuration** | **Run time** | **Error** |
| **57M**  **1 Million points**  **2 dims/point** | **4 Executor**  **4G/Executor** | **102S** | **No** |
| **570M**  **10 Million points**  **2 Dims/point** | **4 Executor**  **4G/Executor** | **32Mins** | **No** |

**算法可扩展性:**

**可以看到当数据量增加10倍的时候，计算时间增长是多于10倍的。**

**七：Others**

**Spark 还实现了 Bisecting KMeans和 StreamingKMeans**

**TODO：算法优化：**

**1：参考目前Kmeans 最好单机版实现：**

**<http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview>**

**高维数据稀疏性**

**2：Community**

**SPARK-14174 Accelerate Kmeans wia Mini-Batch EM**

**思路：抽样**

**加速距离计算**

**保存个态，避免重复计算**

**关于Mini-Batch Kmeans,可以减少算法时间，准确性只比标准Kmeans 差一点，见下图。**

