实验 4: K-均值聚类

1 实验要求

实验任务

使用 Hadoop MapReduce 实现课堂上介绍的“K-Means 聚类算法”。算法描述如下：

Algorithm 1: K-Means 算法

Input: 数据集 $D = \{x_1, x_2, \ldots, x_m\}$; 聚类簇数 k;

Output: 簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$

从 D 中随机选择 k 个样本作为初始均值向量 $\{\mu_1, \mu_2, \ldots, \mu_k\}$

repeat

  令 $C_i = \emptyset (1 \le i \le k)$

  for $j = 1, 2, \ldots, m$ **do**

    计算样本 x_i 与各均值向量 $\mu_i (1 \le i \le k)$ 的距离: $d_{ji} = \|x_j - \mu_i\|_2$;

    根据距离最近的均值向量确定 x_j 的簇标记：$\lambda_j = \arg\min_{i \in \{1,2,\ldots,k\}} d_{ji}$;

    将样本 x_j 划入相应的簇：$C_{\lambda_j} = C_{\lambda_j} \bigcup \{x_j\}$;

  for $i = 1, 2, \ldots, k$ **do**

    计算新均值向量：$\mu_i' = \frac{1}{|C_i|} \sum_{x \in C_i} x$;

    if $\mu_i' \neq \mu_i$ **then**

      将均值向量 μ_i 更新为 μ_i'

    else

      保持当前均值向量不变

until 当前均值向量均未更新;

Result: 聚类中心

注意为了便于验证实验结果，本次实验不需要随机选择初始聚类中心，而是使用给定的初始聚类中心执行聚类。同时，由于给定了初始聚类中心，聚类簇数也是确定的，在本次实验中 $k = 20$。实验的**基本任务**是在 K-Means 算法执行完毕后，输出最终得到的聚类中心。

输出格式

本次实验的基本任务是输出 K-Means 聚类算法完成后得到的聚类中心，共有 20 个聚类中心。输出格式要求如下：

聚类中心 ID(0-19) \TAB 聚类中心向量 (向量的各分量以, 分隔)

下图展示了输出文件的一个片段，其中输出数据的精度可以自行控制，**注意以下示例仅供参考，不对应正确结果：**

```
0   4.49367,4.501744,4.49628,4.49701,4.4969,4.500036,4.500506,4.491124,4.499306,4.503578
1   14.50264,14.500742,14.495192,14.506466,14.495812,14.50276,14.497252,14.501858,14.496668,14.497848
2   24.501768,24.50761,24.50257,24.495495,24.497847,24.506466,24.502674,24.507177,24.501446,24.507666
```

图 1: 输出格式

选作内容

在基础任务中，已经利用 K-Means 算法对数据进行聚类，并得到各个簇的聚类中心。选作部分的任务就是将所属不同聚类的数据划分到不同的文件中。最终产生 k 个输出文件，每个文件对应一个簇，文件中存储属于该簇的所有数据。

提示： Hadoop 的”MultipleOutputs”

2 实验数据

本次实验的数据集包括一个存储所有参与聚类向量的文件和一个存储初始聚类中心的文件。这两个文件的格式如图2所示。数据集共包含 $1,000,000$ 条数据，每条数据表示一个向量，每个向量的维度为 16。每条数据的冒号”:” 之前为该数据条目的 ID，不属于向量的分量。

```
0: 7, 2, 1, 0, 1, 1, 1, 6, 2, 8, 3, 1, 0, 6, 0, 4
1: 7, 4, 0, 6, 0, 9, 2, 9, 3, 0, 7, 3, 7, 8, 0, 6
2: 6, 8, 5, 9, 4, 1, 7, 2, 0, 4, 9, 9, 2, 0, 1, 8
3: 9, 5, 9, 5, 7, 7, 0, 4, 7, 4, 6, 7, 1, 6, 4, 5
4: 9, 7, 6, 1, 7, 1, 7, 8, 7, 5, 8, 5, 8, 1, 8, 0
5: 5, 5, 6, 7, 2, 1, 4, 1, 8, 5, 8, 1, 5, 6, 5, 3
6: 3, 1, 4, 6, 5, 0, 5, 7, 2, 5, 6, 2, 3, 4, 4, 4
7: 2, 4, 2, 0, 0, 1, 7, 3, 6, 7, 2, 9, 5, 2, 0, 4
8: 6, 5, 3, 7, 4, 8, 3, 5, 5, 4, 4, 7, 3, 9, 8, 5
9: 3, 5, 4, 7, 0, 0, 0, 7, 1, 2, 9, 1, 1, 4, 1, 3
10: 5, 6, 6, 4, 6, 1, 7, 2, 9, 6, 2, 2, 3, 2, 1, 7
```

(a) 聚类数据

```
0: 12, 5, 9, 6, 6, 10, 10, 10, 8, 3, 6, 11, 4, 8, 10, 5
1: 16, 19, 22, 19, 13, 21, 16, 17, 21, 17, 15, 15, 14, 17, 18, 20
2: 29, 31, 29, 23, 32, 28, 29, 30, 25, 25, 26, 29, 30, 27, 29, 29
3: 40, 42, 42, 37, 35, 37, 35, 41, 37, 40, 41, 39, 34, 40, 41, 39
4: 44, 48, 49, 51, 49, 49, 43, 49, 48, 51, 47, 46, 44, 45, 49, 49
5: 53, 57, 61, 56, 58, 56, 53, 59, 59, 56, 53, 53, 53, 58, 61, 53
6: 72, 64, 71, 67, 68, 64, 68, 64, 68, 65, 66, 66, 71, 65, 64, 71
7: 79, 82, 81, 73, 75, 75, 75, 82, 74, 80, 81, 77, 74, 79, 78, 80
8: 84, 83, 86, 89, 84, 85, 89, 83, 83, 87, 88, 90, 88, 91, 89, 86
```

(b) 初始聚类中心

图 2: 输入文件格式

单机测试样例：提供完整数据集的一个子集作为单机测试样例，单机测试样例中有 $10,000$ 条数据。该数据集主要供本地调试使用。

完整数据集：完整数据集位于集群中的 HDFS 存储上，存储位置为 hdfs://master001:9000/data/exp4。

注意：最终每个小组的程序必须在课程指定集群上运行，而且输入数据集是完整数据集。结果输出到集群的 *HDFS* 上。*HDFS* 上数据集的文件名为 *dataset.data*，与测试样例中的文件名不同。提交前请注意修改。

3 提示

- 每一轮 K-Means 运算，即计算一次新的聚类中心作为一个 MapReduce 任务。需要在 Job Driver 中控制迭代，在下一轮任务中需要利用上一轮任务产生的聚类中心 (利用 Distributed Cache 或 HDFS 文件系统接口) 。直到聚类中心不再变化时，终止迭代。
- 为了维持 MapReduce 任务迭代运行，注意处理输出文件夹 (删除或更换) 和本次生成的聚类中心。
- 可以使用自定义 Writable 的方式减少字符串和浮点数组之间的反复转换。

示例代码

由于本次实验可能需要使用 HDFS 文件系统接口，下面提供一些使用该接口的示例代码以供参考，更详细的功能请同学们自行查阅相关文档。

```
// 删除文件或目录
public static void deletePath(String pathStr, boolean isDeleteDir) throws IOException {
  if(isDeleteDir) pathStr = pathStr.substring(0, pathStr.lastIndexOf('/'));
  Path path = new Path(pathStr);
  Configuration configuration = new Configuration();
  // 获取 HDFS 文件系统接口
  FileSystem fileSystem = path.getFileSystem(configuration);
  fileSystem.delete(path, true);
}
```

```
// 将一个文件的内容拷贝到另一个文件中
public static void copyFile(String from_path, String to_path) throws IOException {
  Path path_from = new Path(from_path);
  Path path_to = new Path(to_path);
  Configuration configuration = new Configuration();
  // 获取 HDFS 文件系统接口
  FileSystem fileSystem = path.getFileSystem(configuration);

  FSDataInputStream inputStream = fileSystem.open(path_from);
  LineReader lineReader = new LineReader(inputStream, configuration);

  FSDataOutputStream outputStream = fileSystem.create(path_to);

  Text line = new Text();

  while(lineReader.readLine(line) > 0) {
    String str = line.toString() + "\n";
    outputStream.write(str.getBytes());
  }

  lineReader.close();
```

```
    outputStream.close();
}
```

4 实验报告要求

在最后提交的压缩包中，除了包含源代码、JAR 包、JAR 包执行方式说明，还需要包含一个实验报告。实验报告中请包含：

1. Map 和 Reduce 的设计思路（含 Key、Value 类型）。
2. MapReduce 中 Map 和 Reduce 的伪代码（或者带注释的实际代码，如果使用实际代码，请做好排版）。
3. 输出结果文件的部分截图。输出结果文件在 HDFS 上的路径（某些情况下助教会检查 HDFS 上的输出文件）。
4. 请在报告中包含在集群上执行作业后，Yarn Resource Manager 的 WebUI 执行报告内容。注意只需要 K-Means 任务中最后一个任务的执行报告内容即可，如果实验中包含了选做内容，也需要给出选做任务的执行报告。