Assignment Document

> - **By 1183710109 郭茁宁**

> Problem Restatement:
>
> - 生成1万个随机数，存储在HDFS文件系统中的data1.txt中，每个数字之间用“,”分隔。在Spark平台上实现如下的功能：
>
> 1. 从HDFS上读入data1.txt，生成RDD；
> 2. 找到这个数据集合的**中位数**（精确的）；
> 3. 只能使用RDD有关API，并且不能调用Spark提供的中位数计算的API；
> 4. 在Spark平台上实现中位数算法后，验证结果的正确性。

Data Generating

> `generator.py`

- 根据要求生成数据：限定数据量`N = 10000`，数据范围为`M = 100`，即`-50~+50`
- 打开输出文件

```
filename = ".\data1.txt"
with open(filename, "w", encoding="utf-8") as file:
    ...
```

- 导入`random`库，使用`random.random()`生成`0~1`的数据，转换为指定范围
- 用','分隔，并保留6位小数'

```
for i in range(N):
    if i != 0:
        file.write(",")
    file.write(str(round((random.random() - 0.5) * M, 6)))
```

Preliminary

> `work.ipynb`

- 生成`sc`和`spark`对象

```
from pyspark import SparkConf, SparkContext
from pyspark.sql import SparkSession
from pyspark.sql.functions import col
sc = SparkContext(conf=SparkConf())
spark =
SparkSession.builder.master("local[*]").appName("FirstApp").getOrCreate()
```

Calculate

> `work.ipynb`

- **总体思路：通过设置键值对和利用`RDD.sortByKey()`方法对数据排序，取中位数**

- 打开文件，导入数据，利用`str.split()`分隔字符串

```
file = open("hdfs://master:9000/sparkdata/data1.txt")
dataStrList = file.read().split(',')
```

- 将字符串转换为浮点类型

```
dataFloatList = []
for i in dataStrList:
    dataFloatList.append(float(i))
```

- 创建`RDD`，生成键值对
- 排序
- 取出`Mapped RDD`键集合

```
data = sc.parallelize(dataFloatList)
dataMapped = data.map(lambda x: (x, x))
dataSorted = dataMapped.sortByKey(lambda x: x[0]).keys()
```

- 统计元素数
- 区分奇偶情况（可省略）
- 计算中位数

```
N = dataSorted.count()
if N % 2 == 0:
    preHalfList = dataSorted.take(N / 2 + 1)
    avg = (preHalfList[-1] + preHalfList[-2]) / 2
else:
    avg = dataSorted.take(N / 2 + 1)[-1]
```

- 得出结果

```
>>> avg
>>> -0.435513
```

Verification

> `work.ipynb`

- **总体思路：通过计算比得出的`avg`更大数字的个数`cntM`和更小得数字的个数`cntN`是否相等，判断是否为中位数**
- 初始化

```
cntM = 0
cntN = 0
```

- 通过`RDD.collect()`将`RDD`转换为`list`

```
dataSortedList = dataSorted.collect()
```

- 统计比`avg`更大数字的个数`cntM`和更小得数字的个数`cntN`

```
for i in dataSortedList:
    if (i >= avg):
        cntM += 1
    else:
        cntN +=1
```

- 输出结果验证

```
>>> cntM, cntN, cntM == cntN
>>> (5000, 5000, True)
```

Conclusion

- 首先随机生成10000个浮点数，存储到`data1.txt`中；
- 然后基于`pyspark`环境导入文件、分隔字符串并转换为浮点类型、构建`RDD`对象；
- 再生成`Map`键值对，进行排序，取出中间两个数字做平均值，作为**中位数**；
- 最后统计原数组中比该“中位数”大/小的数字的个数，相等则验证该“求中位数算法”正确。

The End