

Spark 编程 & Spark SQL

李振安 151278019

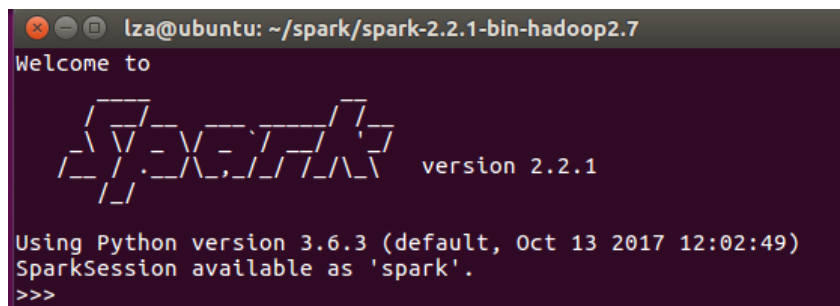
一、 环境配置

下载 Spark 2.2.1, 并解压安装

配置环境变量, 修改/etc/profile, 添加 Spark 路径

修改 conf/spark-env.sh, 添加 java 路径、Hadoop 路径

运行 pyspark



```
lza@ubuntu: ~/spark/spark-2.2.1-bin-hadoop2.7
Welcome to

      _/ _ \| | | | _/_ \| | | |
     / _ \| |_| | | |/_ \| |_| | |
    /_/ _ \|_| |_| |_|/_ \|_| |_| |
   version 2.2.1

Using Python version 3.6.3 (default, Oct 13 2017 12:02:49)
SparkSession available as 'spark'.
>>>
```

二、 代码说明

spark 编程, 实现中文分词、统计词频、sparkSQL 语句查询

使用两种方法过滤

1.使用 RDD 的 filter 方法过滤

2.使用 SQL 查询语句查询

输入格式 <input> <output> <k>

三、 Spark 编程 (Python)

中文分词函数:

#中文分词

```
def chiwordsplit(x):
    seg=jieba.cut(x)
    #停用词文件
    f=open("stopwords.txt",encoding="utf8")
    stopwords=f.readlines()
    for i in range(0,len(stopwords)):
        stopwords[i]=stopwords[i].replace("\n","")
    words=[]
    for word in seg:
        if word not in stopwords and word != " ":
            words.append(word)
    return words
```

MapReduce :

```
#MapReduce
lines = spark.read.text(sys.argv[1]).rdd.map(lambda r: r[0])
counts = lines.flatMap(lambda x: chiwordsplit(x)) \
               .map(lambda x: (x, 1)) \
               .reduceByKey(add)\
               .sortBy(lambda x: -x[1])
```

使用 RDD 的 filter 方法过滤词频在 k 次以下的单词 :

```
#使用RDD的filter方法过滤词频在k次以下的单词
Filteredcounts = counts.filter(lambda x: x[1] >= k)
output = Filteredcounts.collect()
```

Spark SQL :

```
#RDD创建DataFrame
df = spark.createDataFrame(output, ['word', 'count'])
df.createOrReplaceTempView("wordlist")
#SQL Spark SQL查询统计结果中次数超过k次的词语
df2 = spark.sql("SELECT * from wordlist WHERE count > "+str(k))
#把查询结果输出到文件
SQLoutput=df2.collect()
f=open("SQL"+sys.argv[2], 'w')
for (word, count) in SQLoutput:
    print("%s: %i" % (word, count))
    f.write(word+ " : "+str(count)+"\n")
f.close()
```

四、 一些主要类的方法说明

map(f, preservesPartitioning=False)

Return a new RDD by applying a function to each element of this RDD.

```
>>> rdd = sc.parallelize(["b", "a", "c"])

>>> sorted(rdd.map(lambda x: (x, 1)).collect())

[('a', 1), ('b', 1), ('c', 1)]
```

sortBy(keyfunc, ascending=True, numPartitions=None)

Sorts this RDD by the given keyfunc

```
>>> tmp = [('a', 1), ('b', 2), ('1', 3), ('d', 4), ('2', 5)]

>>> sc.parallelize(tmp).sortBy(lambda x: x[0]).collect()
```

```
[('1', 3), ('2', 5), ('a', 1), ('b', 2), ('d', 4)]
```

```
>>> sc.parallelize(tmp).sortBy(lambda x: x[1]).collect()
```

```
[('a', 1), ('b', 2), ('1', 3), ('d', 4), ('2', 5)]
```

filter()

Return a new RDD containing only the elements that satisfy a predicate.

```
>>> rdd = sc.parallelize([1, 2, 3, 4, 5])
```

```
>>> rdd.filter(lambda x: x % 2 == 0).collect()
```

```
[2, 4]
```