作业 3

1、在 pyspark 中编程完成如下任务：

（1）data = [1,5,7,10,23,20,6,5,10,7,10]，求 data 的平均值。

```
>>>data = [1,5,7,10,23,20,6,5,10,7,10]
>>>rdd_data = sc.parallelize(data)
>>>s = rdd_data.reduce(lambda x,y:x+y+0.0)
>>>n = rdd_data.count()
>>>avg = s/n
>>>print("average:",avg)
```

（2）data = [1,5,7,10,23,20,7,5,10,7,10]，求 data 中出现次数最多的数，若有多个，求这些数的平均值。

```
>>>data =  [1,5,7,10,23,20,7,5,10,7,10]
>>>rdd_data = sc.parallelize(data)
>>>rdd_count = rdd_data.map(lambda x:(x,1)).reduceByKey(lambda x,y:x+y)
>>>max_count = rdd_count.map(lambda x:x[1]).reduce(lambda x,y: x if x>=y else y)
>>>rdd_mode = rdd_count.filter(lambda x:x[1]==max_count).map(lambda x:x[0])
>>>mode = rdd_mode.reduce(lambda x,y:x+y+0.0)/rdd_mode.count()
>>>print("mode:",mode)
```

（3）有一批学生信息存放于如下 students 列表中，包括 name,age,score, 找出 score 排名前 3 的学生, score 相同可以任取。

```
students = [("LiLei",18,87),("HanMeiMei",16,77),("DaChui",16,66),("Jim",18,77),("RuHua",18,50)]
>>>students = [("LiLei",18,87),("HanMeiMei",16,77),("DaChui",16,66),("Jim",18,77),("RuHua",18,50)]
>>>n = 3
>>>rdd_students = sc.parallelize(students)
>>>rdd_sorted = rdd_students.sortBy(lambda x:x[2],ascending = False)
>>>students_topn = rdd_sorted.take(n)
>>>print(students_topn)
```

（4）data = [1,7,8,5,3,18,34,9,0,12,8]，按从小到大排序并返回序号, 大小相同的序号可以不同。

```
>>>data = [1,7,8,5,3,18,34,9,0,12,8]
>>>rdd_data = sc.parallelize(data)
>>>rdd_sorted = rdd_data.map(lambda x:(x,1)).sortByKey().map(lambda x:x[0])
>>>rdd_sorted_index = rdd_sorted.zipWithIndex()
>>>print(rdd_sorted_index.collect())
```

2、（20 分）在“/usr/local/spark/mycode/sparksql”文件夹下有 JSON 格式的文件 employee.json，其内容如下所示：

```
{ "id":1 , "name":" Ella" , "age":36 }
{ "id":2, "name":"Bob","age":29 }
{ "id":3 , "name":"Jack","age":29 }
{ "id":4 , "name":"Jim","age":28 }
{ "id":4 , "name":"Jim","age":28 }
{ "id":5 , "name":"Damon" }
{ "id":5 , "name":"Damon" }
```

（1）为 employee.json 创建 DataFrame，查询所有数据；

```
>>> df = spark.read.json("file:///usr/local/spark/mycode/sparksql/employee.json")
>>> df.show()
```

（2）查询所有数据，并去除重复的数据；

```
>>> df.distinct().show()
```

（3）取出前 3 行数据；

```
>>> df.take(3)
```

（4）查询所有记录的 name 列，并为其取别名为 username；

```
>>> df.select(df.name.alias("username")).show()
```

（5）查询年龄 age 的平均值；

```
>>> df.agg({"age": "mean"}).show()
```

3、（30 分）在目录“/usr/local/spark/mycode/rdd/file”下有若干个以文本文件保存的订单数据集，每个文本文件里面包含了很多行数据，每行数据由 4 个字段的值构成，不同字段值之间用逗号隔开，4 个字段分别是 orderid、userid、payment 和 productid，如下为一个样例文件 file1.txt：

```
1,1768,50,155
2,1218, 600,211
3,2239,788,242
4,3101,28,599
5,4899,290,129
6,3110,54,1201
7,4436,259,877
8,2369,7890,27
```

请按照 payment 值的 Top 10 获取相应的各行数据，并保存于 MySQL 数据库 db1 的 order 表中，表结构如下：order(orderid char(20), userid char(4), payment int(6), productid char(20))，再将 order 表的数据读取并显示出来。请编程完成上述任务。

```
from pyspark import SparkContext, SparkConf
from pyspark.sql import SparkSession

if __name__ == "__main__":
    spark = SparkSession.builder.config(conf=SparkConf()).getOrCreate()
    sc = spark.sparkContext
    lines = sc.textFile("file:///usr/local/spark/mycode/rdd/file")
    result1 = lines.filter(lambda line: (len(line.strip()) > 0) and (len(line.split(",")) == 4))
    result2 = result1.map(lambda x: x.split(","))
    result3 = result2.map(lambda x: [x[0].strip(), x[1].strip(), int(x[2].strip()), x[3].strip()])
    result4 = result3.repartition(1)
    result5 = result4.sortBy(lambda x: x[2], False)
    result6 = result5.take(10)
    df = spark.createDataFrame(result6,['orderid','userid','payment','productid'])
    prop = {}
    prop['user'] = 'root'
    prop['password'] = '123456'
    prop['driver'] = "com.mysql.jdbc.Driver"
    df.write.jdbc("jdbc:mysql://localhost:3306/db1", 'order1', 'append', prop)
    df1 = spark.read.format("jdbc").option("url","jdbc:mysql://localhost:3306/db1") \
        .option("driver","com.mysql.jdbc.Driver").option("dbtable", "order1") \
        .option("user","root").option("password", "123456").load()
    df1.show()
```

4、（30 分）在 MySQL 数据库 db1 中有表 student，表结构与内容如下所示：

student(id int(4) primary key not null, name char(10), address char(100), label int(2));

id	name	address	label
1	zhangsan	hebei road, xicheng district, Beijing	1
2	lisi	Jiangxi road, xiqing district, Tianjin	0
3	wangwu	Nanjing road, pudong district, Shanghai	0
4	chenliu	Shandong road, haidian district, Beijing	1
...	...	...	...
100	zhaoqian	guanxian road, shibei district, Qingdao	0
101	wuqi	wangfujing road, dongcheng district, Beijing	
102	zhuba	guangxi road, tianhe district, Guangzhou	

…	…	…	…
130	linshu	wener road, xihu district, Hangzhou	

其中，id、name、address、label 分别表示序号、姓名、住址和分类标记，表中共 130 条记录，前 100 条记录分类标记字段已有相应的值，地址中含有”Beijing”的，分类标记字段值为 1，地址中不含”Beijing”的，分类标记字段值为 0；后 30 条记录分类标记字段为空。

请采用 Spark MLlib 的逻辑斯蒂回归算法实现对表中后 30 条记录的分类标记进行预测，并显示预测结果。

```
from pyspark import SparkConf
from pyspark.sql import SparkSession
from pyspark.ml import Pipeline
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.feature import HashingTF, Tokenizer

if __name__ == "__main__":
    spark = SparkSession.builder.config(conf=SparkConf()).getOrCreate()
    df1 = spark.read.format("jdbc").option("url","jdbc:mysql://localhost:3306/db1") \
        .option("driver","com.mysql.jdbc.Driver").option("dbtable", "student") \
        .option("user","root").option("password", "123456").load() #读取表中的数据
    df1.createOrReplaceTempView('student') #创建临时表
    trainDF = spark.sql("select id,name,address,label from student where id<=100") #提取训练数据
    testDF = spark.sql("select id,name,address from student where id>100") #提取测试数据
    tokenizer = Tokenizer(inputCol="address", outputCol="words") #分词器
    hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(),outputCol="features") #构建特征向量
    lr = LogisticRegression(maxIter=10, regParam=0.001) #构建逻辑斯蒂回归分类器
    pipeline = Pipeline(stages=[tokenizer,hashingTF,lr]) #机器学习构建流水线
    model = pipeline.fit(trainDF) #用训练数据集对流水线进行训练，得到流水线模型
    prediction = model.transform(testDF) #用流水线模型对测试数据进行测试
    selected = prediction.select("id","name","address","prediction") #从测试结果数据框中选择要显示的列
    for row in selected.collect():
        id,name,address,prediction = row
        print("%d,%s,%s,%d" % (id,name,address,prediction))
```