

0 安装配置Spark

0.1 官网下载并解压

用虚拟机登录官网<https://archive.apache.org/dist/spark/>下载spark-3.5.3/spark-3.5.3-bin-hadoop3.tgz

```
1 # 进入解压包存放的Downloads文件夹,然后解压缩包至/home/chenmiao
2 sudo tar -zxf spark-3.5.3-bin-hadoop3.tgz -C /home/chenmiao
3 # 进入刚刚解压后存放的目录下,将该文件夹的名字重命名为spark
4 sudo mv ./spark-3.5.3-bin-hadoop3/ ./spark
```

0.2 安装依赖

安装 Spark 的 Python 包:

```
1 pip install pyspark
```

```
chenmiao@chenmiao-virtual-machine:~$ pip install pyspark
Defaulting to user installation because normal site-packages is not writeable
Collecting pyspark
  Downloading pyspark-3.5.3.tar.gz (317.3 MB)
    317.3/317.3 MB 1.5 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting py4j==0.10.9.7
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
    200.5/200.5 KB 0.6 MB/s eta 0:00:00
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.3-py2.py3-none-any.whl size=317840648 sha256=d498a2cde028a7ad737dc2c0e0819baa731eb72526ff08aff89ceaf3e3c58653
  Stored in directory: /home/chenmiao/.cache/pip/wheels/1b/3a/92/28b93e2fbfdbb07509ca4d6f50c5e407f48dce4ddbda69a4ab
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.7 pyspark-3.5.3
chenmiao@chenmiao-virtual-machine:~$
```

0.3 配置环境变量

```
1 export SPARK_HOME=/home/chenmiao/spark
2 export PATH=$PATH:$SPARK_HOME/bin
3 export SPARK_CONF_DIR=$SPARK_HOME/conf
4
5 source ~/.bashrc # 激活环境变量
```

0.4 配置文件

1. 配置spark-env.sh

```
1 # 将spark-env.sh.template重命名为spark-env.sh
2 # 加入以下内容
3 export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
4 export HADOOP_HOME=/usr/local/hadoop
5 export SPARK_MASTER_HOST=localhost
6 export SPARK_LOCAL_IP=localhost
7 export SPARK_MASTER_PORT=7077
```

2. 配置spark-defaults.conf

```
1 # 将spark-defaults.conf.template重命名为spark-defaults.conf
2 # 加入以下内容
3 spark.master                spark://localhost:7077
4 spark.driver.memory         2g
5 spark.executor.memory       4g
6 spark.eventLog.enabled      true
```

3. 配置slaves（需要自己新建）

```
1 localhost
```

0.5 启动Spark

注：可访问<http://localhost:8080>检查节点状态

```
1 # 启动Spark Master
2 $SPARK_HOME/sbin/start-master.sh
3 # 启动Spark worker并连接到Master节点
4 $SPARK_HOME/sbin/start-worker.sh spark://localhost:7077
```

终端运行结果如下：

```
chenmiao@chenmiao-virtual-machine:~$ $SPARK_HOME/sbin/start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /home/chenmiao/spark/
logs/spark-chenmiao-org.apache.spark.deploy.master.Master-1-chenmiao-virtual-mac
hine.out
chenmiao@chenmiao-virtual-machine:~$ $SPARK_HOME/sbin/start-worker.sh spark://lo
calhost:7077
starting org.apache.spark.deploy.worker.Worker, logging to /home/chenmiao/spark/
logs/spark-chenmiao-org.apache.spark.deploy.worker.Worker-1-chenmiao-virtual-mac
hine.out
chenmiao@chenmiao-virtual-machine:~$
```

网页上节点状态显示如下：

Spark Master at spark://localhost:7077

URL: spark://localhost:7077
Alive Workers: 1
Cores in use: 4 Total, 0 Used
Memory in use: 14.6 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory	Resources
worker-20241212163400-127.0.0.1-43689	127.0.0.1:43689	ALIVE	4 (0 Used)	14.6 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	------------------------	----------------	------	-------	----------

```
1 # 停止Spark worker
2 $SPARK_HOME/sbin/stop-worker.sh
3 # 停止Spark Master
4 $SPARK_HOME/sbin/stop-master.sh
```

0.6 遇到的问题

问题：后续 `spark-submit <python文件名>` 运行时，遇到报错，原因是Spark日志事件目录不存在。Spark默认会尝试将事件日志写入 `/tmp/spark-events`，但该目录不存在，因此导致 `java.io.FileNotFoundException`。

解决方法：手动创建日志事件目录

```
1 mkdir -p /tmp/spark-events
2 chmod 777 /tmp/spark-events
```

Task1：Spark RDD编程

1.1 查询资金流入和流出情况

使用 `user_balance_table`，计算出所有用户在每一天的总资金流入和总资金流出量。

输出格式：<日期> <资金流入量> <资金流出量>

1.1.1 设计思路

1. 初始化Spark环境

通过 `sparkConf` 和 `sparkContext` 创建Spark运行环境，配置应用名称和运行模式。

`local[*]` 表示在本地运行。

2. 读取文件并过滤表头

3. 解析数据

提取 `report_date`、`total_purchase_amt` 和 `total_redeem_amt` 字段，并将缺失值处理为0。

4. 转换数据格式

5. 聚合数据

使用 `reduceByKey` 操作按 `report_date` 聚合数据，计算总的资金流入和流出。

6. 排序

7. 保存和输出结果

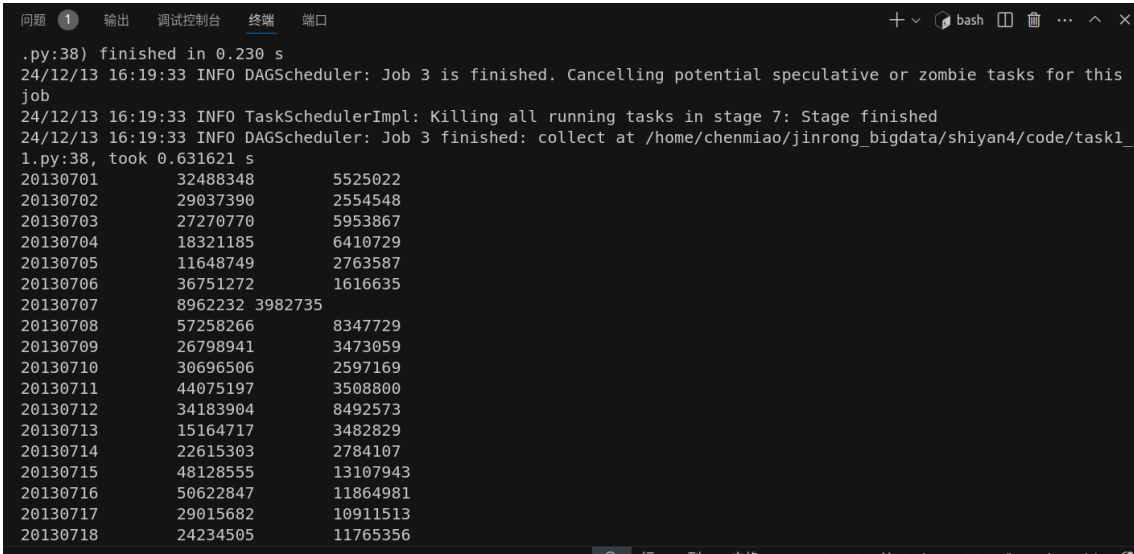
保存到CSV文件并在终端输出结果。

8. 停止SparkContext

1.1.2 运行结果

在终端输入 `spark-submit task1_1.py` 运行代码

- 终端输出结果（节选）



```
问题 1 输出 调试控制台 终端 端口
.py:38) finished in 0.230 s
24/12/13 16:19:33 INFO DAGScheduler: Job 3 is finished. Cancelling potential speculative or zombie tasks for this job
24/12/13 16:19:33 INFO TaskSchedulerImpl: Killing all running tasks in stage 7: Stage finished
24/12/13 16:19:33 INFO DAGScheduler: Job 3 finished: collect at /home/chenmiao/jinrong_bigdata/shiyan4/code/task1_1.py:38, took 0.631621 s
20130701      32488348      5525022
20130702      29037390      2554548
20130703      27270770      5953867
20130704      18321185      6410729
20130705      11648749      2763587
20130706      36751272      1616635
20130707      8962232 3982735
20130708      57258266      8347729
20130709      26798941      3473059
20130710      30696506      2597169
20130711      44075197      3508800
20130712      34183904      8492573
20130713      15164717      3482829
20130714      22615303      2784107
20130715      48128555      13107943
20130716      50622847      11864981
20130717      29015682      10911513
20130718      24234505      11765356
```

- 保存为csv文件截图（节选）

	A	B	C	D	E	F	G	H	I	J	K
1	report_date	total_purchase_amt	total_redeem_amt								
2	20130701	32488348	5525022								
3	20130702	29037390	2554548								
4	20130703	27270770	5953867								
5	20130704	18321185	6410729								
6	20130705	11648749	2763587								
7	20130706	36751272	1616635								
8	20130707	8962232	3982735								
9	20130708	57258266	8347729								
10	20130709	26798941	3473059								
11	20130710	30696506	2597169								
12	20130711	44075197	3508800								
13	20130712	34183904	8492573								
14	20130713	15164717	3482829								
15	20130714	22615303	2784107								
16	20130715	48128555	13107943								
17	20130716	50622847	11864981								
18	20130717	29015682	10911513								
19	20130718	24234505	11765356								
20	20130719	33680124	9244769								
21	20130720	20439079	4601143								
22	20130721	21142394	2681331								
23	20130722	40448896	19144267								
24	20130723	58136147	24404051								
25	20130724	48422518	36258592								
26	20130725	57433418	38212836								
27	20130726	44721817	39192369								
28	20130727	17194451	15058893								
29	20130728	36255382	7683211								
30	20130729	53512076	18599364								
31	20130730	47481243	13048582								
32	20130731	54569637	9534040								
33	20130801	53369962	18864468								
34	20130802	38064536	40150769								
35	20130803	21595789	20701797								
36	20130804	45745254	8263965								
37	20130805	43632203	15797507								
38	20130806	50866184	16401387								
39	20130807	43908081	29708706								
40	20130808	44493490	29551691								
41	20130809	33425186	30131015								
42	20130810	20615307	26614408								
43	20130811	55519543	15372680								
44	20130812	94520502	28586669								

1.1.3 程序分析

进一步对可能的改进之处进行分析。

1. 可扩展性

不足：代码中字段位置通过硬编码指定，若表结构发生变化，代码需要修改，灵活性较差。

改进分析：将字段名参数化，避免硬编码。

2. 数据加载与解析

不足：CSV文件的解析是手动完成的，使用 `split(",")`。如果数据格式复杂（如字段中包含逗号、引号等特殊字符），可能会导致解析错误。

改进分析：使用 `csv` 库进行CSV文件加载和解析。

1.2 活跃用户分析

使用 `user_balance_table`，定义活跃用户为在指定月份内有至少5天记录的用户，统计2014年8月的活跃用户总数。

1.2.1 设计思路

1. 初始化Spark环境

通过 `sparkconf` 和 `sparkcontext` 创建Spark运行环境，配置应用名称和运行模式。

`local[*]` 表示在本地运行。

2. 读取文件并过滤表头

3. 解析数据并筛选

定义目标月份；

将每行数据按逗号分隔成字段，提取 `user_id` 和 `report_date` ；

筛选出 `report_date` 中以目标月份开头的记录；

过滤掉 `None` 值。

4. 用户活跃天数统计

使用 `map` 操作将筛选后的记录转换为 `(user_id,report_date)` 的格式，然后使用 `distinct` 操作去除重复的 `user_id`，接着使用 `map` 操作将每个 `user_id` 映射为1，最后使用 `reduceByKey` 操作累加每个用户的活跃天数。

5. 筛选并统计活跃用户

使用 `filter` 函数，筛选出活跃天数大于等于5的用户；

对活跃用户RDD调用 `count`，统计活跃用户的总数。

6. 输出结果

7. 停止SparkContext

1.2.2 运行结果

在终端输入 `spark-submit task1_2.py` 运行代码。

- 终端输出结果

得到2014年8月的活跃用户总数为：12767

```
问题 2 输出 调试控制台 终端 端口
24/12/13 17:04:51 INFO TaskSetManager: Finished task 4.0 in stage 3.0 (TID 15) in 102 ms on localhost (executor driver) (5/5)
24/12/13 17:04:51 INFO TaskSchedulerImpl: Removed TaskSet 3.0, whose tasks have all completed, from pool
24/12/13 17:04:51 INFO DAGScheduler: ResultStage 3 (count at /home/chenmiao/jinrong_bigdata/shiyan4/code/task1_2.py:38) finished in 0.240 s
24/12/13 17:04:51 INFO DAGScheduler: Job 1 is finished. Cancelling potential speculative or zombie tasks for this job
24/12/13 17:04:51 INFO TaskSchedulerImpl: Killing all running tasks in stage 3: Stage finished
24/12/13 17:04:51 INFO DAGScheduler: Job 1 finished: count at /home/chenmiao/jinrong_bigdata/shiyan4/code/task1_2.py:38, took 14.049440 s
2014年8月的活跃用户总数为： 12767
24/12/13 17:04:51 INFO SparkContext: SparkContext is stopping with exitCode 0.
24/12/13 17:04:51 INFO SparkUI: Stopped Spark web UI at http://localhost:4040
24/12/13 17:04:51 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/12/13 17:04:51 INFO MemoryStore: MemoryStore cleared
24/12/13 17:04:51 INFO BlockManager: BlockManager stopped
24/12/13 17:04:51 INFO BlockManagerMaster: BlockManagerMaster stopped
24/12/13 17:04:51 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
24/12/13 17:04:51 INFO SparkContext: Successfully stopped SparkContext
24/12/13 17:04:52 INFO ShutdownHookManager: Shutdown hook called
24/12/13 17:04:52 INFO ShutdownHookManager: Deleting directory /tmp/spark-1d83cbb0-7289-43cf-9b44-ac3be6eadd16
24/12/13 17:04:52 INFO ShutdownHookManager: Deleting directory /tmp/spark-b8d7f44b-36a5-4ed0-9d1b-61218aaa29ef
24/12/13 17:04:52 INFO ShutdownHookManager: Deleting directory /tmp/spark-b8d7f44b-36a5-4ed0-9d1b-61218aaa29ef/pyspark-10f7da55-12a5-42a7-b5ac-d551ca05e2a4
(syenv) chenmiao@chenmiao-virtual-machine:~/jinrong_bigdata/shiyan4/code$
```

注：因为只有一个数字结果所以就没再写成输出文件了。

1.2.3 程序分析

进一步对可能的改进之处进行分析。

1. 可扩展性

不足：代码中字段位置通过硬编码指定，若表结构发生变化，代码需要修改，灵活性较差。

改进分析：将字段名参数化，避免硬编码。

2. 数据加载与解析

不足：CSV文件的解析是手动完成的，使用 `split(",")`。如果数据格式复杂（如字段中包含逗号、引号等特殊字符），可能会导致解析错误。

改进分析：使用 `csv` 库进行CSV文件加载和解析。

3. 性能

不足：数据集没有进行分区，处理效率较低。

改进分析：可以对数据进行分区，以提高处理效率。

Task2: Spark SQL编程

2.1 统计特定日期平均余额

按城市统计2014年3月1日的平均余额：计算每个城市在2014年3月1日的用户平均余额(`tBalance`)，按平均余额降序排列。

输出格式：<城市ID> <平均余额>

2.1.1 设计思路

1. 初始化SparkSession

2. 读取文件并筛选

使用 `spark.read.csv` 方法读取两张表 `user_profile_table` 和 `user_balance_table`，筛选 `user_balance_table` 表中目标日期的数据。

3. 将两个表按`user_id`进行关联

4. 聚合数据

按城市分组（`groupBy("City")`），对 `tBalance` 字段计算平均值（`avg("tBalance")`），然后使用 `orderBy` 方法对结果进行降序排序。

5. 收集结果到本地

6. 保存和输出结果

保存到CSV文件并在终端输出结果。

7. 停止SparkContext

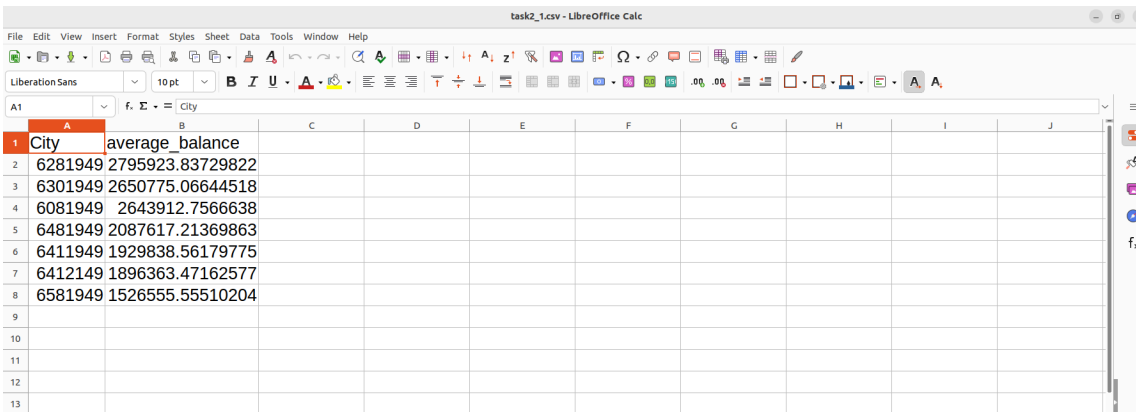
2.1.2 运行结果

在终端输入 `spark-submit task2_1.py` 运行代码。

- 终端输出结果

```
问题 4 输出 调试控制台 终端 窗口
:37) finished in 0.249 s
24/12/13 17:56:19 INFO DAGScheduler: Job 8 is finished. Cancelling potential speculative or zombie tasks for this job
24/12/13 17:56:19 INFO TaskSchedulerImpl: Removed TaskSet 12.0, whose tasks have all completed, from pool
24/12/13 17:56:19 INFO TaskSchedulerImpl: Killing all running tasks in stage 12: Stage finished
24/12/13 17:56:19 INFO DAGScheduler: Job 8 finished: collect at /home/chenmiao/jinrong_bigdata/shiyan4/code/task2_1.py
:37, took 0.295280 s
6281949 2795923.837298216
6301949 2650775.0664451825
6081949 2643912.7566638007
6481949 2087617.2136986302
6411949 1929838.5617977527
6412149 1896363.471625767
6581949 1526555.5551020408
24/12/13 17:56:19 INFO SparkContext: SparkContext is stopping with exitCode 0.
24/12/13 17:56:19 INFO SparkUI: Stopped Spark web UI at http://localhost:4040
24/12/13 17:56:19 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/12/13 17:56:19 INFO MemoryStore: MemoryStore cleared
24/12/13 17:56:19 INFO BlockManager: BlockManager stopped
24/12/13 17:56:19 INFO BlockManagerMaster: BlockManagerMaster stopped
24/12/13 17:56:19 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
24/12/13 17:56:19 INFO SparkContext: Successfully stopped SparkContext
每个城市在2014年3月1日的平均余额计算已完成，结果保存到： /home/chenmiao/jinrong_bigdata/shiyan4/output/task2_1.csv
24/12/13 17:56:20 INFO ShutdownHookManager: Shutdown hook called
```

- 保存为csv文件截图



	A	B	C	D	E	F	G	H	I	J
1	City	average balance								
2		6281949 2795923.83729822								
3		6301949 2650775.06644518								
4		6081949 2643912.7566638								
5		6481949 2087617.21369863								
6		6411949 1929838.56179775								
7		6412149 1896363.47162577								
8		6581949 1526555.55510204								
9										
10										
11										
12										
13										

2.1.3 程序分析

进一步对可能的改进之处进行分析。

1. 数据存储

不足：当前结果以CSV格式保存，效率较低。

改进分析：对于大规模数据，可以使用更高效的存储格式（如Parquet或ORC），以便后续查询和处理。

2. CSV字段类型

不足：没有直接指定CSV每列的字段，而是进行了类型推断，可能带来的性能问题。

改进分析：在读取CSV文件时，可以显式指定每列的数据类型。

2.2 统计每个城市总流量前3高的用户

统计每个城市中每个用户在2014年8月的总流量（定义为total_purchase_amt+total_redeem_amt），并输出每个城市总流量排名前三的用户ID及其总流量。

输出格式：<城市ID> <用户ID> <总流量>

2.2.1 设计思路

1. 初始化SparkSession

2. 读取数据并创建临时视图

使用 `spark.read.csv` 读取CSV文件 (`user_profile_table.csv` 和 `user_balance_table.csv`)，并设置 `header=True` 和 `inferSchema=True`，让Spark自动识别列名和数据类型。

使用 `createOrReplaceTempView` 将DataFrame注册为Spark SQL临时视图。这使得我们可以直接在SQL查询中操作DataFrame。

3. SQL查询 (这部分直接化用了实验三用到的代码)

```
1  # 定义临时结果集CTE user_city_activity
2  # 存储2014年8月每个用户的user_id、city、total_traffic (总流量)
3  WITH user_city_activity AS (
4      SELECT up.user_id, up.city,
5             (SUM(ub.total_purchase_amt) + SUM(ub.total_redeem_amt)) AS
total_traffic
6      # 通过user_id将两表连接
7      FROM user_balance_table ub
8      JOIN user_profile_table up ON ub.user_id = up.user_id
9      WHERE ub.report_date >= '20140801' AND ub.report_date <= '20140831'
10     # 筛选日期
11     GROUP BY up.user_id, up.city # 按用户id和城市分组
12 )
13 # 定义临时结果集CTE ranked_users
14 # 使用CTE user_city_activity对每个城市的用户进行排名
15 ranked_users AS (
16     SELECT user_id, city, total_traffic,
17            # 按照city分组，总流量降序排列，从1开始分配排名
18            ROW_NUMBER() OVER (PARTITION BY city ORDER BY total_traffic
DESC) AS rank
19     FROM user_city_activity
20 )
21
22 # 从CTE ranked_users中选择user_id、city和total_traffic
23 SELECT CAST(user_id AS BIGINT) AS user_id,
24        city,
25        CAST(total_traffic AS BIGINT) AS total_traffic
26 FROM ranked_users
27 WHERE rank <= 3      # 筛选每个城市的前三高
28 ORDER BY city, rank;
```

4. 执行查询并获取结果

使用 `spark.sql` 执行SQL查询，并将结果存储在DataFrame中。

5. 收集结果到本地

6. 保存和输出结果

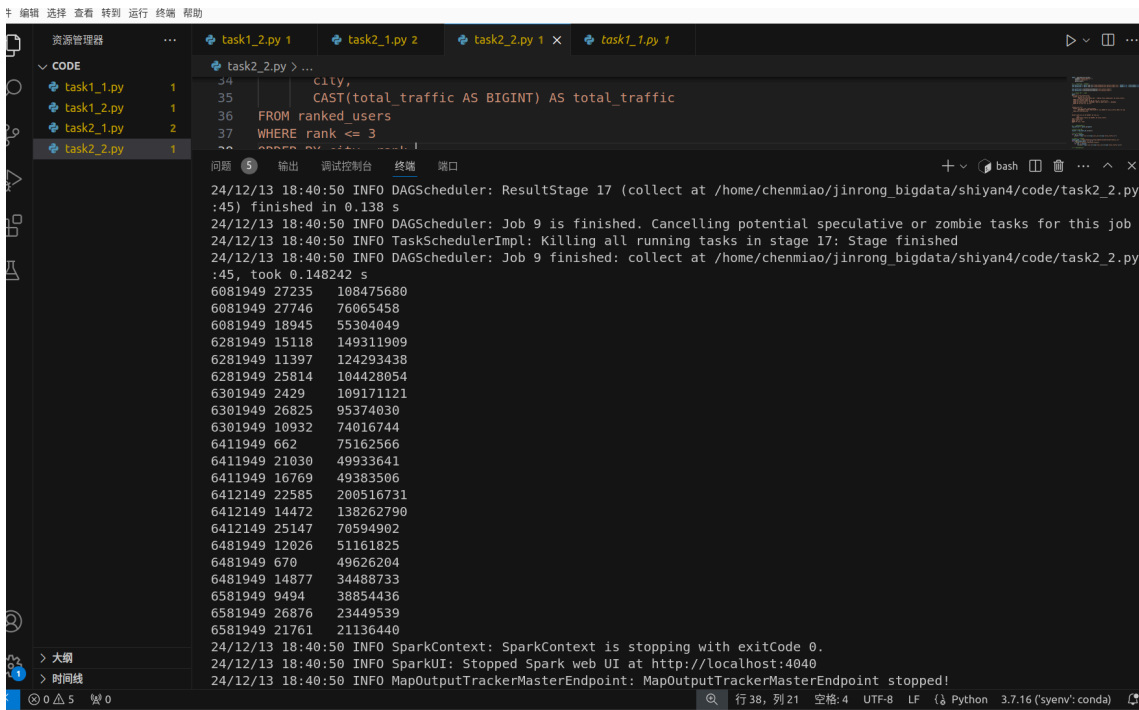
保存到CSV文件并在终端输出结果。

7. 停止SparkContext

2.2.2 运行结果

在终端输入 `spark-submit task2_2.py` 运行代码。

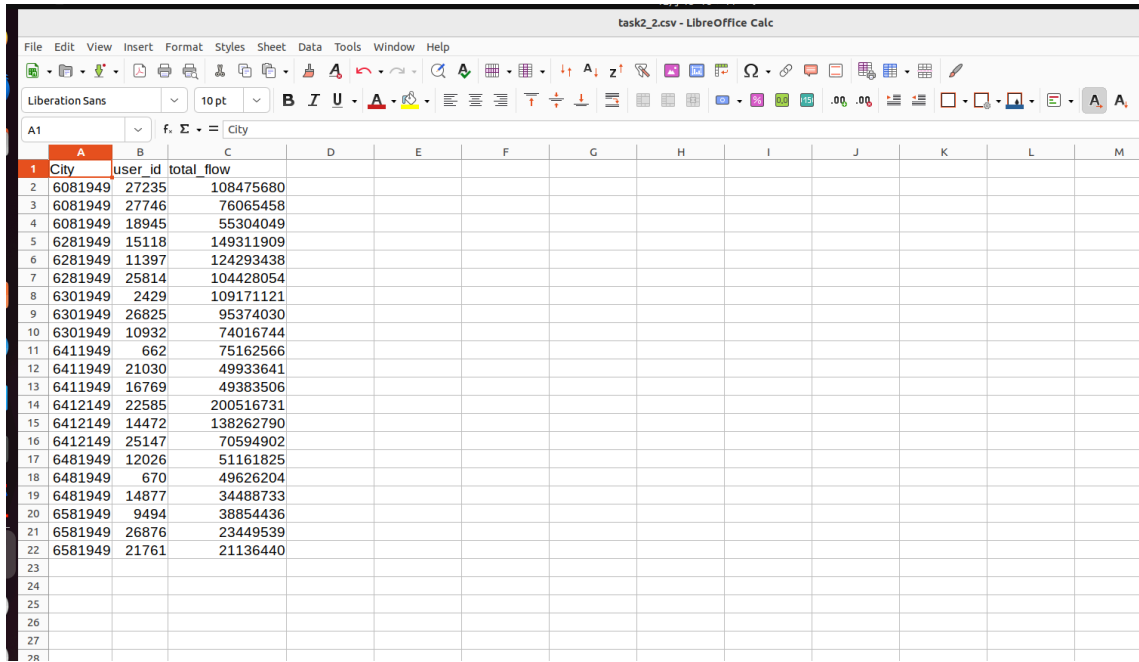
- 终端输出结果



```
task2_2.py > ...
34 city,
35 CAST(total_traffic AS BIGINT) AS total_traffic
36 FROM ranked_users
37 WHERE rank <= 3
38 ORDER BY total_flow DESC

24/12/13 18:40:50 INFO DAGScheduler: ResultStage 17 (collect at /home/chenmiao/jinrong_bigdata/shiyan4/code/task2_2.py
:45) finished in 0.138 s
24/12/13 18:40:50 INFO DAGScheduler: Job 9 is finished. Cancelling potential speculative or zombie tasks for this job
24/12/13 18:40:50 INFO TaskSchedulerImpl: Killing all running tasks in stage 17: Stage finished
24/12/13 18:40:50 INFO DAGScheduler: Job 9 finished: collect at /home/chenmiao/jinrong_bigdata/shiyan4/code/task2_2.py
:45, took 0.148242 s
6081949 27235 108475680
6081949 27746 76065458
6081949 18945 55304049
6281949 15118 149311909
6281949 11397 124293438
6281949 25814 104428054
6301949 2429 109171121
6301949 26825 95374030
6301949 10932 74016744
6411949 662 75162566
6411949 21030 49933641
6411949 16769 49383506
6412149 22585 200516731
6412149 14472 138262790
6412149 25147 70594902
6481949 12026 51161825
6481949 670 49626204
6481949 14877 34488733
6581949 9494 38854436
6581949 26876 23449539
6581949 21761 21136440
24/12/13 18:40:50 INFO SparkContext: SparkContext is stopping with exitCode 0.
24/12/13 18:40:50 INFO SparkUI: Stopped Spark web UI at http://localhost:4040
24/12/13 18:40:50 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
```

- 保存为csv文件截图



A	B	C
City	user_id	total_flow
6081949	27235	108475680
6081949	27746	76065458
6081949	18945	55304049
6281949	15118	149311909
6281949	11397	124293438
6281949	25814	104428054
6301949	2429	109171121
6301949	26825	95374030
6301949	10932	74016744
6411949	662	75162566
6411949	21030	49933641
6411949	16769	49383506
6412149	22585	200516731
6412149	14472	138262790
6412149	25147	70594902
6481949	12026	51161825
6481949	670	49626204
6481949	14877	34488733
6581949	9494	38854436
6581949	26876	23449539
6581949	21761	21136440

2.2.3 程序分析

进一步对可能的改进之处进行分析。

1. 代码健壮性

不足：没有对异常情况的处理，代码健壮性较弱。

改进分析：增加异常处理逻辑，确保文件读取、SQL执行、文件写入等过程不会因异常终止。

赛制

赛题与数据

排行榜

FAQ

论坛

使用天池实验室打比赛

提交结果

我的成绩

我的团队

长期赛

日期: 2024-12-16 16:42:40

分数: 99.8652

日期: 2024-12-16 15:46:30

分数: 13.3675

日期: 2024-12-16 15:42:27

分数:

ERROR Bad input file

暂无更多数据

长期赛: 无

注：提交文件需要把表头那行删除

3.3 程序分析

进一步对可能的改进之处进行分析。

1. 数据太过片面

不足：只使用了过去7个月每天的总资金流入、总资金流出量，并没有结合收益率表和银行间拆借利率表做更加准确的预测。

改进分析：结合收益率表和银行间拆借利率表中的数据进一步完善模型。

2. 随机森林的参数并不是最优的

不足：没有试着调整随机森林的参数使得预测效果更好

改进分析：引入交叉验证，设置不同的树的数量（`numTrees`）、树的最大深度（`maxDepth`）参数，找到这些参数的最佳组合。