



LOREM IPSUM

大数据•Hadoop•Spark

傅友、曾友雯、李傲



CONTENTS

- | | | |
|----|------------------|-------|
| 01 | 大数据平台 | 部署 |
| 02 | Spark & Hadoop案例 | 分析与实现 |
| 03 | 踩过的坑 | 解决方法 |
| 04 | 反思与总结 | 收获 |

01

大数据平台

部署

完全分布式HADOOP & SPARK部署

```
liyuanshuo@li01:~$ ssh li01
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

685 个可升级软件包。
384 个安全更新。

Last login: Fri Nov 23 21:52:03 2018 from 192.168.0.6
liyuanshuo@li01:~$ jps
2690 DataNode
3742 Jps
2815 Worker
liyuanshuo@li01:~$
```

```
liyuanshuo@li02:~$ ssh li02
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-31-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

686 个可升级软件包。
383 个安全更新。

Last login: Fri Nov 23 21:52:10 2018 from 192.168.0.6
liyuanshuo@li02:~$ jps
3569 Jps
2775 Worker
2651 DataNode
liyuanshuo@li02:~$
```

```
liyuanshuo@liyuanshuo:~/Desktop$ jps
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
7347 Master
30260 Jps
6838 NameNode
7082 SecondaryNameNode
liyuanshuo@liyuanshuo:~/Desktop$
```

完全分布式HADOOP & SPARK部署

```
liyuanshuo@liyuanshuo:~/Desktop$ jps
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
7347 Master
30260 Jps
6838 NameNode
7082 SecondaryNameNode
liyuanshuo@liyuanshuo:~/Desktop$ pyspark
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
Python 3.6.5 (default, May 11 2018, 13:30:17)
[GCC 7.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
2018-11-24 14:32:45 WARN NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
Welcome to

   / \ / \ / \ / \ / \
  / \ \ / \ / \ / \ / \ / \
 / \ / . \ / \ / \ / \ / \ / \
 / \ / \ / \ / \ / \ / \ / \
version 2.3.0

Using Python version 3.6.5 (default, May 11 2018 13:30:17)
SparkSession available as 'spark'.
>>> █
```

完全分布式HADOOP & SPARK部署

Configured Capacity:	20.42 GB
DFS Used:	2.45 GB (12.01%)
Non DFS Used:	11.2 GB
DFS Remaining:	5.68 GB (27.82%)
Block Pool Used:	2.45 GB (12.01%)
DataNodes	0 / 115.00% / 115.00% / 2.07%

Apache Spark 2.3.0

Spark Master at spark://liyuanshuo:7077

URL: spark://liyuanshuo:7077

REST URL: spark://liyuanshuo:6066 (cluster mode)

Alive Workers: 2

Cores in use: 4 Total, 0 Used

Memory in use: 4.0 GB Total, 0.0 B Used

Applications: 0 Running, 0 Completed

Drivers: 0 Running, 0 Completed

Status: ALIVE

Workers (2)

Worker Id	Address	State	Cores	Memory
worker-20181123215218-192.168.0.7-33019	192.168.0.7:33019	ALIVE	2 (0 Used)	2.0 GB (0.0 B Used)
worker-20181123215218-192.168.0.9-40711	192.168.0.9:40711	ALIVE	2 (0 Used)	2.0 GB (0.0 B Used)

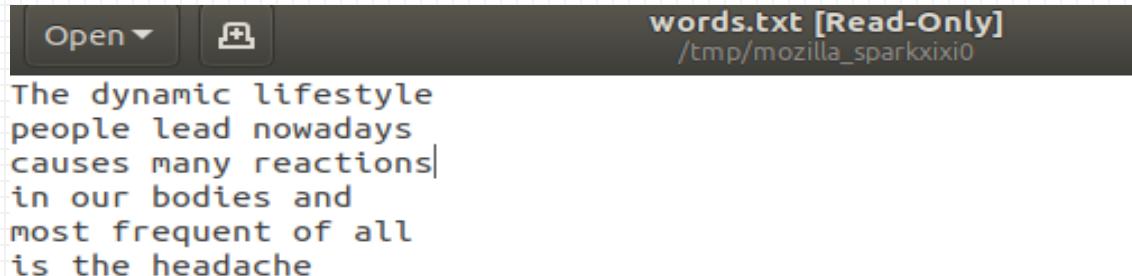
02

Spark & Hadoop案例

分析与实现

Wordcount小样例

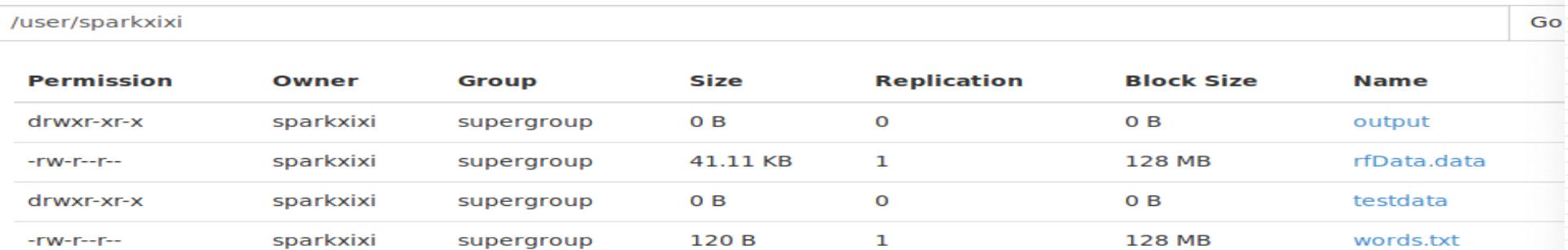
数据集：



The dynamic lifestyle
people lead nowadays
causes many reactions|
in our bodies and
most frequent of all
is the headache

```
hadoop fs - put words.txt hdfs://localhost:9000/user/sparkxixi
```

Browse Directory



/user/sparkxixi							Go
Permission	Owner	Group	Size	Replication	Block Size	Name	
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	output	
-rw-r--r--	sparkxixi	supergroup	41.11 KB	1	128 MB	rfData.data	
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	testdata	
-rw-r--r--	sparkxixi	supergroup	120 B	1	128 MB	words.txt	

Wordcount小样例

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** firstusespark (~/PycharmProjects/firstusespark)
- File:** wordcount.py
- Code Content:**

```
1 import sys
2 from pyspark import SparkContext
3 from operator import add
4
5
6 def main():
7     sc = SparkContext(appName=__"wordscount")
8     lines = sc.textFile("words.txt")
9     counts = lines.flatMap(lambda x:x.split(' '))
10    .map(lambda x:(x,1))
11    .reduceByKey(add)
12    output = counts.collect()
13    print(output)
14    for (word, count) in output:
15        print("%s:%i" %(word, count), end=" ")
16
17    # sc.setLogLevel('ERROR')
18
19    sc.stop()
20
21 if __name__ == "__main__":
22     main()
```
- Run Tab:** Shows the command run: /usr/bin/python3.6 /home/sparkxixi/PycharmProjects/firstusespark/wordcount.py and the output log.
- Output Log:**

```
/usr/bin/python3.6 /home/sparkxixi/PycharmProjects/firstusespark/wordcount.py
2018-11-20 03:28:53 WARN  Utils:66 - Your hostname, ubuntu resolves to a loopback address: 127.0.1.1; using 192.168.37.128 instead
2018-11-20 03:28:53 WARN  Utils:66 - Set SPARK_LOCAL_IP if you need to bind to another address
2018-11-20 03:28:53 WARN  NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes instead.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
[('The', 1), ('lifestyle', 1), ('causes', 1), ('in', 1), ('of', 1), ('is', 1), ('dynamic', 1), ('people', 1), ('lead', 1), ('nowadays', 1), ('bodies', 1), ('and', 1), ('most', 1), ('fre', 1), ('reactions', 1), ('our', 1), ('many', 1)]
Process finished with exit code 0
```

森林植被分类预测

数据集：使用UCI的数据集解压后大小为71M（官方数据描述很清楚）

下载地址：<https://archive.ics.uci.edu/ml/machine-learning-databases/covtype/>

数据集记录了美国科罗拉多州不同地块森林植被特征：海拔、坡度、到水源的距离、遮阳情况和土壤类型，并且随同给出了地块的已知森林植被类型。有581012个样本。

字段描述：54个属性字段，一个类别字段环境属性10个：数值属性，描述海拔、方位、斜角等环境信息；野生区域属性4个：二元属性，标定区域类型；土壤类型属性40个：二元属性，标记土壤类型；森林植被类型：1~7，标记七种不同类型

数据集部署到hadoop+spark平台上：

```
liyuanshuo@liyuanshuo:~/PycharmProjects$ ls
AS_SSOIER_10.py from covtype.data.gz      MySpark_02      spark_01.tar: Thu Nov 22 14:44:55 yunqiCrawl 192.168.0
BigDataHomework    DecisionTree           python3_crawl  spiderbookuo@li02:~$ jps
BigDataHomework02  finalBigDataHomework ShenCeCup       Subject-and-Sentiment-Analysis
covtype.data        ML                  spark_01       TitanicJps
liyuanshuo@liyuanshuo:~/PycharmProjects$ hadoop fs -put covtype.data@li02:~$ jps
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp5 DataNode
liyuanshuo@liyuanshuo:~/PycharmProjects$ hadoop fs -ls   2818 Jps
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp0 Worker
Found 1 items
-rw-r--r--  2 liyuanshuo supergroup  75169317 2018-11-22 15:37 covtype.data
liyuanshuo@liyuanshuo:~/PycharmProjects$ r[2.3.0/sbin
```

查看数据集：

从HDFS文件系统中读取查看数据集的大致信息

```
/usr/bin/python3.6 /home/liyuanshuo/PycharmProjects/finalBigDataHomework/fileInfo.py
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
2018-11-22 15:29:19 WARN NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
*****查看数据集的前两个*****
2018-11-22 15:29:24 WARN Utils:66 - Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.debug.maxToStringFields' in SparkEnv.conf.
[Row( c0='2596', c1='51', c2='3', c3='258', c4='0', c5='510', c6='221', c7='232', c8='148', c9='6279', c10='1', c11='0', c12='0', c13='0', c14='0', c15='0', c16='0', c17='0', c18='0', c19='0', c20='0', c21='0', c22='0', c23='0', c24='0', c25='0', c26='0', c27='0', c28='0', c29='0', c30='0', c31='0', c32='0', c33='0', c34='0', c35='0', c36='0', c37='0', c38='0', c39='0', c40='0', c41='0', c42='1', c43='0', c44='0', c45='0', c46='0', c47='0', c48='0', c49='0', c50='0', c51='0', c52='0', c53='0', c54='5'), Row( c0='2590', c1='56', c2='2', c3='212', c4='6', c5='390', c6='220', c7='235', c8='151', c9='6225', c10='1', c11='0', c12='0', c13='0', c14='0', c15='0', c16='0', c17='0', c18='0', c19='0', c20='0', c21='0', c22='0', c23='0', c24='0', c25='0', c26='0', c27='0', c28='0', c29='0', c30='0', c31='0', c32='0', c33='0', c34='0', c35='0', c36='0', c37='0', c38='0', c39='0', c40='0', c41='0', c42='1', c43='0', c44='0', c45='0', c46='0', c47='0', c48='0', c49='0', c50='0', c51='0', c52='0', c53='0', c54='5')]
*****数据集的大小*****
[Stage 2:=====] (3 + 1) / 4] 581012
*****数据集的描述*****
DataFrame[summary: string, c0: string, c1: string, c2: string, c3: string, c4: string, c5: string, c6: string, c7: string, c8: string, c9: string, c10: string, c11: string, c12: string, c13: string, c14: string, c15: string, c16: string, c17: string, c18: string, c19: string, c20: string, c21: string, c22: string, c23: string, c24: string, c25: string, c26: string, c27: string, c28: string, c29: string, c30: string, c31: string, c32: string, c33: string, c34: string, c35: string, c36: string, c37: string, c38: string, c39: string, c40: string, c41: string, c42: string, c43: string, c44: string, c45: string, c46: string, c47: string, c48: string, c49: string, c50: string, c51: string, c52: string, c53: string, c54: string]
*****
Process finished with exit code 0
```

数据格式转换数据预处理：

Pyspark要求数据格式为LabeledPoint的RDD形式的，原始数据格式不符合，进行一下格式转换
在调用Pyspark提供的机器学习库的时候，需要对数据进行预处理，把数据范围进行相应的调整
每个算法都需要进行一次，这里只展示的是贝叶斯算法的数据预处理的输出

```
/usr/bin/python3.6 /home/liyuanshuo/PycharmProjects/finalBigDataHomework/dataTransform.py
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp
2018-11-22 16:56:04 WARN NativeCodeLoader:62 - Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
*****格式转换和数据预处理之后的数据*****
(581012, 55)
[[5.        1.1055    0.42382271 ... 0.        0.        ],
 [5.        1.0965    0.46537396 ... 0.        0.        ],
 [2.        1.4175    1.15512465 ... 0.        0.        ],
 ...
 [3.        0.7905    1.32132964 ... 0.        0.        ],
 [3.        0.7875    1.41274238 ... 0.        0.        ],
 [3.        0.786     1.37119114 ... 0.        0.        ]]
*****
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=gasp

Process finished with exit code 0
```

可以参考网上的数据预处理，这里不存在缺失值，直接对数据范围进行转化

数据格式转换数据预处理：

/liyuanshuo	Go!
-------------	---------------------

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	liyuanshuo	supergroup	3.72 KB	2018/11/14 下午7:22:23	2	16 MB	README.md
-rw-r--r--	liyuanshuo	supergroup	71.69 MB	2018/11/22 下午4:40:00	2	16 MB	covtype.data
drwxr-xr-x	liyuanshuo	supergroup	0 B	2018/11/14 下午8:19:36	0	0 B	csvs_per_year
-rw-r--r--	liyuanshuo	supergroup	60.95 MB	2018/11/22 下午4:57:10	2	16 MB	data.data
-rw-r--r--	liyuanshuo	supergroup	156.46 MB	2018/11/22 下午5:59:39	2	16 MB	decisionTreeData.data
-rw-r--r--	liyuanshuo	supergroup	207.64 MB	2018/11/23 下午8:09:49	2	16 MB	knnData.data
-rw-r--r--	liyuanshuo	supergroup	3.54 MB	2018/11/23 下午10:06:47	2	16 MB	knnData_10000.data
-rw-r--r--	liyuanshuo	supergroup	1.77 MB	2018/11/23 下午10:13:46	2	16 MB	knnData_5000.data
-rw-r--r--	liyuanshuo	supergroup	17.71 MB	2018/11/23 下午9:52:29	2	16 MB	knnData_50000.data
-rw-r--r--	liyuanshuo	supergroup	3.72 KB	2018/11/14 下午7:16:51	2	16 MB	mySpark
-rw-r--r--	liyuanshuo	supergroup	156.46 MB	2018/11/22 下午9:47:46	2	16 MB	randomForestData.data

问题分析：

需要预测的结果：森林植被类型：1~7，标记七种不同类型

得出：这是一个分类问题

选择算法：

1、朴素贝叶斯分类：<http://spark.apache.org/docs/latest/mllib-naive-bayes.html>

2、决策树：<http://spark.apache.org/docs/latest/mllib-decision-tree.html>

3、决策树升级->随机森林：

<http://spark.apache.org/docs/latest/mllib-ensembles.html#random-forests>

4、K近邻-官方没有现成的库，但是算法原理比较简单，且有其他版本的实现可供参考

考虑自己实现KNN算法

<https://github.com/evancasey/spark-knn-recommender>

https://github.com/ldearHui/sparkML_project

前三种算法参考样例实现，第四种算法参考民间代码复现

朴素贝叶斯：

数据预处理代码：

```
data[:, i] = (data[:, i]-data[:, i].min())/(data[:, i].max()-data[:, i].min()+1)*3
```

训练集设置60%， 测试集设置40%

分类预测代码：

```
# 获得训练模型，第一个参数为数据，第二个参数为平滑参数lamda，默认为1，可改
```

```
model = NaiveBayes.train(training, lamda: 1.0)
```

```
2018-11-22 17:42:15 INFO TaskSetManager:54 - Finished task 0.0 in stage 5.0 (TID 14) in 2853 ms on localhost (executor driver) (4/4)
2018-11-22 17:42:15 INFO TaskSchedulerImpl:54 - Removed TaskSet 5.0, whose tasks have all completed, from pool
2018-11-22 17:42:15 INFO DAGScheduler:54 - ResultStage 5 (count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/navieBayes.py:25) finished in 2.863 s
2018-11-22 17:42:15 INFO DAGScheduler:54 - Job 4 finished: count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/navieBayes.py:25, took 2.866287 s
*****打印朴素贝叶斯的准确率： (0.6 / 0.4) *****
0.640302966293491
*****
2018-11-22 17:42:15 INFO SparkContext:54 - Invoking stop() from shutdown hook
2018-11-22 17:42:15 INFO AbstractConnector:318 - Stopped Spark@68c3d986{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
2018-11-22 17:42:15 INFO SparkUI:54 - Stopped Spark web UI at http://liyuanshuo:4040
2018-11-22 17:42:15 INFO MapOutputTrackerMasterEndpoint:54 - MapOutputTrackerMasterEndpoint stopped!
```

决策树：
70%的训练集，30%的测试集

```
# 训练模型
# 空categoricalFeaturesInfo表示所有特征是连续的。代码直接参照官方的样例就可以，只不过部分参数需要自己设置一下
# Empty categoricalFeaturesInfo indicates all features are continuous.
# 参数说明
# numClasses: 分类数，需比实际类别数量大，这里设置为8;
# categoricalFeaturesInfo: 特征类别信息，为空，意为所有特征为连续型变量;
# impurity: 信息纯度度量，进行分类时可选择熵或基尼，这里设置为基尼;
# maxDepth: 决策树最大深度，这里设为15;
# maxBins: 特征分裂时的最大划分数量，这里设为32。
model = DecisionTree.trainClassifier(trainingData, numClasses=8, categoricalFeaturesInfo={},
                                      impurity='gini', maxDepth=15, maxBins=32)
```

```
2018-11-22 21:39:42 INFO PythonRunner:54 - Times: total = 1243, boot = -3542, init = 3543, tfinish = 1242
2018-11-22 21:39:42 INFO Executor:54 - Finished task 8.0 in stage 37.0 (TID 360). 1354 bytes result sent to driver
2018-11-22 21:39:42 INFO TaskSetManager:54 - Finished task 8.0 in stage 37.0 (TID 360) in 1252 ms on localhost (executor driver) (10/10)
2018-11-22 21:39:42 INFO TaskSchedulerImpl:54 - Removed TaskSet 37.0, whose tasks have all completed, from pool
2018-11-22 21:39:42 INFO DAGScheduler:54 - ResultStage 37 (count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/decisionTree.py:26) finished in 3.872 s
2018-11-22 21:39:42 INFO DAGScheduler:54 - Job 21 finished: count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/decisionTree.py:26, took 3.875128 s
*****打印决策树的准确率: (0.7 / 0.3) *****
Decision Tree precision: 0.8535617476798677
*****
2018-11-22 21:39:42 INFO SparkContext:54 - Invoking stop() from shutdown hook
2018-11-22 21:39:42 INFO AbstractConnector:318 - Stopped Spark@eabb2c6{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
2018-11-22 21:39:42 INFO SparkUI:54 - Stopped Spark web UI at http://liyuanshuo:4040
2018-11-22 21:39:42 INFO MapOutputTrackerMasterEndpoint:54 - MapOutputTrackerMasterEndpoint stopped!
2018-11-22 21:39:42 INFO MemoryStore:54 - MemoryStore cleared
```

森林植被分类预测

随机森林：
参数说明：

numClasses:分类数，
需比实际类别数量大，
这里设置为8；

categoricalFeaturesInfo:
特征类别信息，为空，
意为所有特征为连续型
变量；

numTrees:森林中树的
数量，这里设为20；

featureSubsetStrategy:
特征子集采样策略，auto
表示算法自主选取；

impurity:信息纯度度量，
进行分类时可选择熵或
基尼，这里设置为基尼；

maxDepth:决策树最大
深度，这里设为18；

maxBins:特征分裂时的
最大划分数量,这里设为
32。

```
2018-11-22 22:05:57 INFO DAGScheduler:54 - Job 24 finished: count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/randomForest.py:26, took 3.999709 s
*****打印随机森林的相关信息*****
Random Forest precision: 0.7987568831733216
Learned classification forest model:
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1087
2018-11-22 22:05:57 INFO BlockManagerInfo:54 - Removed broadcast_62_piece0 on liyuanshuo:33503 in memory (size: 5.3 KB, free: 294.6 MB)
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1083
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1089
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1091
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1076
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1099
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1081
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1092
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1094
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1078
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1079
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1082
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1097
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1088
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1077
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1093
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1095
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1100
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1085
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1086
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1080
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1084
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1096
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1098
2018-11-22 22:05:57 INFO ContextCleaner:54 - Cleaned accumulator 1090
TreeEnsembleModel classifier with 20 trees
```

森林植被分 K近邻：

7/3的数据比例

参数说明：

k为6，距离权重
均为1，样本间
距离使用欧几里
得距离

```

liyuanshuo@liyuanshuo:~/PycharmProjects/finalBigDataHomework
1854685
2018-11-23 22:34:00 INFO PythonRunner:54 - Times: total = 29, boot = -19, init = 28, finish = 20
2018-11-23 22:34:00 INFO Executor:54 - Finished task 0.0 in stage 3.0 (TID 3). 1354 bytes result sent to driver
2018-11-23 22:34:00 INFO TaskSetManager:54 - Finished task 0.0 in stage 3.0 (TID 3) in 44 ms on localhost (executor driver) (1/1)
2018-11-23 22:34:00 INFO TaskSchedulerImpl:54 - Removed TaskSet 3.0, whose tasks have all completed, from pool
2018-11-23 22:34:00 INFO DAGScheduler:54 - ResultStage 3 (count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/knn2.py:168) finished in 0.050 s
2018-11-23 22:34:00 INFO DAGScheduler:54 - Job 3 finished: count at /home/liyuanshuo/PycharmProjects/finalBigDataHomework/knn2.py:168, took 0.056728 s
这是KNN算法的准确率： 0.752
*****KNN的准确率打印完成*****
*
2018-11-23 22:34:00 INFO AbstractConnector:318 - Stopped Spark@52000e5d{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
2018-11-23 22:34:00 INFO SparkUI:54 - Stopped Spark web UI at http://liyuanshuo:4040
2018-11-23 22:34:00 INFO MapOutputTrackerMasterEndpoint:54 - MapOutputTrackerMasterEndpoint stopped!
2018-11-23 22:34:00 INFO MemoryStore:54 - MemoryStore cleared
2018-11-23 22:34:00 INFO BlockManager:54 - BlockManager stopped
2018-11-23 22:34:00 INFO BlockManagerMaster:54 - BlockManagerMaster stopped
2018-11-23 22:34:00 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint:54 - OutputCommitCoordinator stopped!
2018-11-23 22:34:00 INFO SparkContext:54 - Successfully stopped SparkContext
*****打印程序运行时间: 单位 (s) *****
*****
60.479187965393066
*****
2018-11-23 22:34:01 INFO ShutdownHookManager:54 - Shutdown hook called
2018-11-23 22:34:01 INFO ShutdownHookManager:54 - Deleting directory /tmp/spark-6e67eeab-cea6-4077-a720-c6b3ff0b4756
2018-11-23 22:34:01 INFO ShutdownHookManager:54 - Deleting directory /tmp/spark-6e67eeab-cea6-4077-a720-c6b3ff0b4756/pyspark-10578fa6-1991-4eb6-8baa-03c9c4cf2a2b
2018-11-23 22:34:01 INFO ShutdownHookManager:54 - Deleting directory /tmp/spark-62364560-c6e5-4d6a-85a6-519269a8972c
liyuanshuo@liyuanshuo:~/PycharmProjects/finalBigDataHomework$ 

```

Talk is cheap show me the code

数据预处理代码：

<https://github.com/liyuanshuo/finalBigDataHomework/blob/master/dataTransform.py>

朴素贝叶斯代码：

<https://github.com/liyuanshuo/finalBigDataHomework/blob/master/navieBayes.py>

决策树代码：

<https://github.com/liyuanshuo/finalBigDataHomework/blob/master/decisionTree.py>

随机森林代码：

<https://github.com/liyuanshuo/finalBigDataHomework/blob/master/randomForest.py>

KNN代码：

<https://github.com/liyuanshuo/finalBigDataHomework/blob/master/knn.py>

<https://github.com/liyuanshuo/finalBigDataHomework/blob/master/knn2.py>

整个项目代码：

<https://github.com/liyuanshuo/finalBigDataHomework>

利用Mahout进行K-means聚类分析

Mahout简介

Mahout 是一个很强大的数据挖掘工具，是一个分布式机器学习算法的集合，包括：被称为Taste的分布式协同过滤的实现、分类、聚类等。Mahout最大的优点就是基于hadoop实现，把很多以前运行于单机上的算法，转化为了MapReduce模式，这样大大提升了算法可处理的数据量和处理性能。



利用Mahout进行K-means聚类分析

Mahout安装配置：

1. 下载 Mahout 发布版
2. 解压安装 Mahout 将压缩文件解压到 /home/hadoop/
3. 配置环境变量



```
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$SPARK_HOME/bin:$PATH:$SCALA_HOME/bin:$PATH:$MAHOUT_HOME/conf:$MAHOUT_HOME/bin:$PATH  
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib:$HADOOP_COMMON_LIB_NATIVE_DIR"  
  
export SPARK_HOME=/usr/local/spark  
export PYTHONPATH=$SPARK_HOME/python:$SPARK_HOME/python/lib/py4j-0.10.7-src.zip:$PYTHONPATH  
export PYSPARK_PYTHON=/usr/bin/python3.6  
export PYSPARK_DRIVER_PYTHON=/usr/bin/python3.6  
export SCALA_HOME=/usr/share/scala-2.11  
  
export MAHOUT_HOME=/usr/local/hadoop/apache-mahout-distribution-0.11.0  
export MAHOUT_CONF_DIR=$MAHOUT_HOME/conf
```

利用Mahout进行K-means聚类分析

数据集：

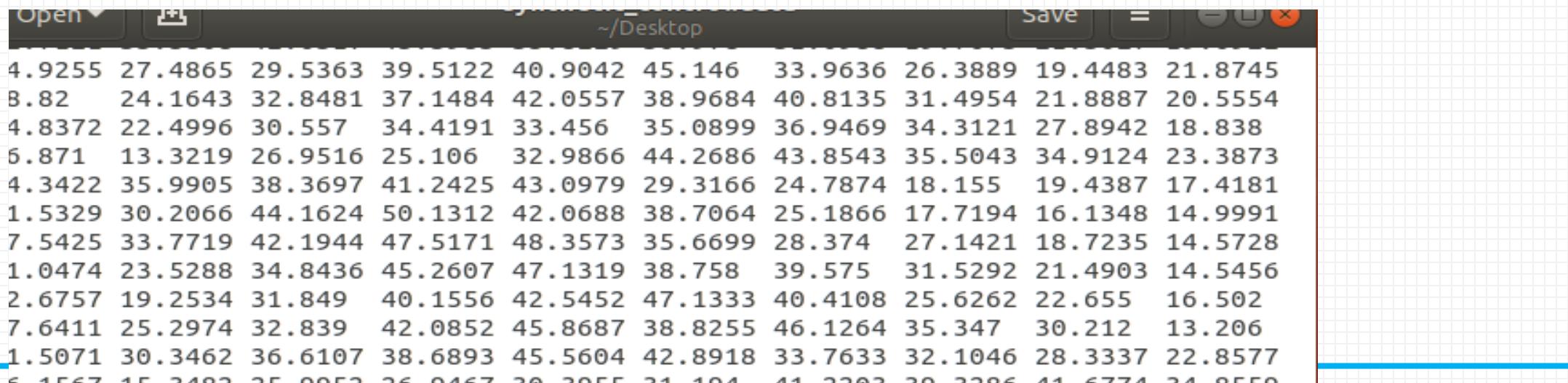
http://archive.ics.uci.edu/ml/databases/synthetic_control/synthetic_control.data

数据包括600个数据点，每个数据点有60个属性。

数据格式

数据存储在ASCII文件中，600行，60列，每行一个图表。类别信息如下：

1-100正常；101-200循环；201-300增长趋势；301-400减少趋势；401-500向上移动；501-600向下移动



A screenshot of a terminal window titled 'Desktop'. The window contains the first 10 lines of a data file. The data consists of 60 numerical values per row, separated by spaces. The columns represent different features of the data points.

4.9255	27.4865	29.5363	39.5122	40.9042	45.146	33.9636	26.3889	19.4483	21.8745
8.82	24.1643	32.8481	37.1484	42.0557	38.9684	40.8135	31.4954	21.8887	20.5554
4.8372	22.4996	30.557	34.4191	33.456	35.0899	36.9469	34.3121	27.8942	18.838
5.871	13.3219	26.9516	25.106	32.9866	44.2686	43.8543	35.5043	34.9124	23.3873
4.3422	35.9905	38.3697	41.2425	43.0979	29.3166	24.7874	18.155	19.4387	17.4181
1.5329	30.2066	44.1624	50.1312	42.0688	38.7064	25.1866	17.7194	16.1348	14.9991
7.5425	33.7719	42.1944	47.5171	48.3573	35.6699	28.374	27.1421	18.7235	14.5728
1.0474	23.5288	34.8436	45.2607	47.1319	38.758	39.575	31.5292	21.4903	14.5456
2.6757	19.2534	31.849	40.1556	42.5452	47.1333	40.4108	25.6262	22.655	16.502
7.6411	25.2974	32.839	42.0852	45.8687	38.8255	46.1264	35.347	30.212	13.206
1.5071	30.3462	36.6107	38.6893	45.5604	42.8918	33.7633	32.1046	28.3337	22.8577

利用Mahout进行K-means聚类分析

1. 上传数据到HDFS

hadoop fs -put synthetic_control.data hdfs://localhost:9000/user/sparkxixi/testdata

/user/sparkxixi/testdata						Go!
Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	sparkxixi	supergroup	281.62 KB	1	128 MB	synthetic_control.data

2. 运行 Mahout 中的 kmeans 聚类算法，执行命令

hadoop jar mahout-examples-0.11.0-job.jar org.apache.mahout.clustering.syntheticcontrol.kmeans.Job

利用Mahout进行K-means聚类分析

```
18/11/20 04:03:06 INFO mapreduce.Job: map 0% reduce 0%
18/11/20 04:03:11 INFO mapreduce.Job: map 100% reduce 0%
18/11/20 04:03:11 INFO mapreduce.Job: Job job_1542705685059_0012 completed successfully
18/11/20 04:03:11 INFO mapreduce.Job: Counters: 30
  File System Counters
    FILE: Number of bytes read=0
    FILE: Number of bytes written=107753
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=343371
    HDFS: Number of bytes written=363295
    HDFS: Number of read operations=13
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
  Job Counters
    Launched map tasks=1
    Data-local map tasks=1
    Total time spent by all maps in occupied slots (ms)=2429
    Total time spent by all reduces in occupied slots (ms)=0
    Total time spent by all map tasks (ms)=2429
    Total vcore-milliseconds taken by all map tasks=2429
    Total megabyte-milliseconds taken by all map tasks=2487296
  Map-Reduce Framework
    Map input records=600
    Map output records=600
    Input split bytes=126
    Spilled Records=0
    Failed Shuffles=0
    Merged Map outputs=0
    GC time elapsed (ms)=47
    CPU time spent (ms)=1020
    Physical memory (bytes) snapshot=168841216
```

利用Mahout进行K-means聚类分析

聚类结果可在output cluster-10-final里查看：

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	sparkxixi	supergroup	194 B	1	128 MB	_policy
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusteredPoints
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-0
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-1
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-10-final
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-2
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-3
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-4
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-5
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-6
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-7
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-8
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	clusters-9
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	data
drwxr-xr-x	sparkxixi	supergroup	0 B	0	0 B	random-seeds

03

踩过的坑

1. 关于在pycharm中spark连接python的错误

如图所示：

```
rotocol.Py4JJavaError: An error occurred while calling z:org.apache.spark.api.python.PythonRDD.collectAndServe.  
apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 0.0 failed 1 times, most recent failure  
java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1128)
```

解决办法：在Run/Edit Configurations/Environment variables中添加环境变量
如图所示：

Name	Value
PYTHONUNBUFFERED	1
SPARK_HOME	/usr/local/spark
PYTHONPATH	/usr/local/spark/python
PYSPARK_PYTHON	/usr/bin/python3.6

2. KNN算法运行时内存爆满：
不要使用那么多数据，Spark是基于内存的
 3. 每次数据处理完之后一定要打开看一看，确保预处理之后的数据时正确的
第一次处理结果对，第二次数据预处理结果未必对
 4. Pycharm设置环境变量后可以本地连接spark进行简单的调试，
这样方便查找代码错误出在哪里
-

04

反思和总结

收获

收获：

- 1、了解了Hadoop生态环境以及应用场景
- 2、小组讨论可以活跃思维，可以向组内其他成员学习不同的看法以及理解
- 3、PySpark的编程案例，以及ML库的接口调用
- 4、基础机器学习算法的原理，以及算法准确度的度量
- 5、了解了Mahout等第三方数据挖掘工具
- 6、增强了动手能力，以及写Bug的能力

不足之处：

- 1、对于HADOOP系统使用不够深刻，平时只会运用简单的HDFS分布式文件系统
 - 2、对与MAP-REDUCE的编程模型理解不够，许多代码只能借鉴。
 - 3、Spark的理解不够深入，只实现了最简单的KNN算法，其他算法都是调库
 - 4、菜是原罪
 - 5、菜是原罪
 - 6、菜是原罪
-



thanks