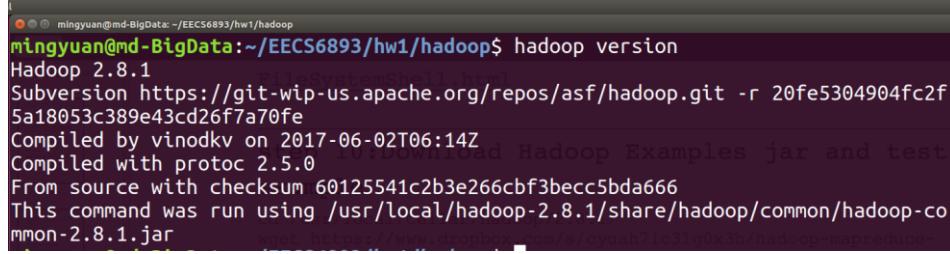


Hadoop Installation

Hadoop version check

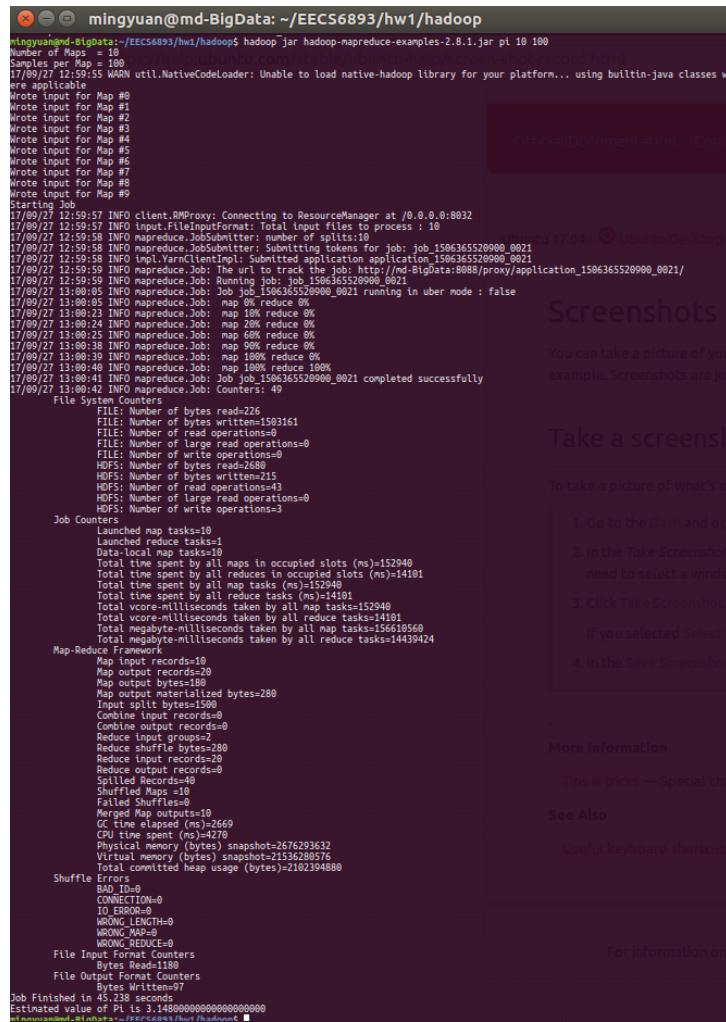
Run “\$ Hadoop version” to ensure Hadoop is installed on VM and check its version.



```
mingyuan@md-BigData:~/EECS6893/hw1/hadoop$ hadoop version
Hadoop 2.8.1
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 20fe5304904fc2f
5a18053c389e43cd26f7a70fe
Compiled by vinodkv on 2017-06-02T06:14Z
Compiled with protoc 2.5.0
From source with checksum 60125541c2b3e266cbf3becc5bda666
This command was run using /usr/local/hadoop-2.8.1/share/hadoop/common/hadoop-common-2.8.1.jar
```

Running Hadoop examples

1. Pi calculation



```
mingyuan@md-BigData:~/EECS6893/hw1/hadoop$ hadoop jar hadoop-mapreduce-examples-2.8.1.jar pi 10 100
Number of Maps = 100
Number of Reduces = 10
17/09/27 12:59:55 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Wrote input For Map #0
Wrote input For Map #1
Wrote input For Map #2
Wrote input For Map #3
Wrote input For Map #4
Wrote input For Map #5
Wrote input For Map #6
Wrote input For Map #7
Wrote input For Map #8
Wrote input For Map #9
executing job
17/09/27 12:59:57 INFO client.OMProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/09/27 12:59:57 INFO InputFormat: Total input files to process : 10
17/09/27 12:59:58 INFO mapreduce.JobSubmitter: number of splits:10
17/09/27 12:59:58 INFO mapreduce.Job: Job job_1506365520900_0021 running in uber mode : false
17/09/27 12:59:58 INFO impl.YarnClientImpl: Submitted application application_1506365520900_0021
17/09/27 12:59:58 INFO mapreduce.Job: The url to track the job: http://md-BigData:8088/proxy/application_1506365520900_0021/
17/09/27 12:59:59 INFO mapreduce.Job: Running job: job_1506365520900_0021
17/09/27 13:00:23 INFO mapreduce.Job: Job job_1506365520900_0021 running in uber mode : false
17/09/27 13:00:24 INFO mapreduce.Job: map 10% reduce 0%
17/09/27 13:00:25 INFO mapreduce.Job: map 20% reduce 0%
17/09/27 13:00:26 INFO mapreduce.Job: map 30% reduce 0%
17/09/27 13:00:39 INFO mapreduce.Job: map 100% reduce 0%
17/09/27 13:00:40 INFO mapreduce.Job: map 100% reduce 100%
17/09/27 13:00:41 INFO mapreduce.Job: Job job_1506365520900_0021 completed successfully
17/09/27 13:00:41 INFO mapreduce.Job: Counters: 49
File System Counters
FILE: Number of bytes read=226
FILE: Number of bytes written=1503161
FILE: Number of read operations=1
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=2680
HDFS: Number of bytes written=215
HDFS: Number of read operations=4
HDFS: Number of large read operations=0
HDFS: Number of write operations=3
Job Counters
Launched map tasks=10
Launched reduce tasks=1
Data-local map tasks=10
Total time spent by all maps in occupied slots (ms)=152940
Total time spent by all reduce in occupied slots (ms)=14101
Total failed map tasks=1
Total failed reduce tasks=1
Total time spent by all reduce tasks (ms)=14101
Total vcore-milliseconds taken by all map tasks=152940
Total vcore-milliseconds taken by all reduce tasks=14101
Total map-milliseconds taken by all map tasks=156610569
Total map-milliseconds taken by all reduce tasks=14439424
Map-Reduce Framework
Map input records=10
Map output records=0
Map output bytes=180
Map output materialized bytes=280
Input split bytes=1500
Combine input records=0
Combiner input records=0
Reduce input groups=2
Reduce shuffle bytes=280
Reduce input records=20
Reduce output records=0
Spilled Records=0
Shuffled Maps =10
Failed Shuffles=0
File Input Format Counters
Bytes Read=180
File Output Format Counters
Bytes Written=97
Job Finished in 45.238 seconds
Estimated value of PI is 3.14000000000000000000
informant@md-BigData:~/EECS6893/hw1/hadoop$
```

The result of Pi calculation given arguments “10 100” is 3.14000000000000000000

2. Sudoku puzzle solution

```

mingyuan@md-BigData:~/EECS6893/hw1/hadoop$ hadoop
    jar hadoop-mapreduce-examples-2.8.1.jar sudoku p
uzzle1.dta
          ? ? ? ? 4 5 ? 7 8
Solving puzzle1.dta
8 5 1 3 9 2 6 4 7
4 3 2 6 7 8 1 9 5
7 9 6 5 1 4 3 8 2
6 1 4 8 2 3 7 5 9      Execute: ./bin/hadoop jar hadoop-
.5 7 8 9 6 1 4 2 3
3 2 9 4 5 7 8 1 6
9 4 7 2 8 6 5 3 1
1 8 5 7 3 9 2 6 4
2 6 3 1 4 5 9 7 8
Create file romeo_juliet.txt
Found 1 solutions

```

● Example 3: Word counter

3. Wordcount

a. Running word count job

```

an@md-BigData: ~/EECS6893/hw1/hadoop
mingyuan@md-BigData:~/EECS6893/hw1/hadoop$ clear
mingyuan@md-BigData:~/EECS6893/hw1/hadoop$ hdfs dfs -ls input
17/09/27 13:07:17  WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 6 items
drwxr-xr-x  - mingyuan supergroup          0 2017-09-26 10:47 input/Bitcoin
drwxr-xr-x  - mingyuan supergroup          0 2017-09-26 10:20 input/Crone
drwxr-xr-x  - mingyuan supergroup          0 2017-09-26 10:16 input/HB
drwxr-xr-x  - mingyuan supergroup          0 2017-09-20 09:27 input/Starline
drwxr-xr-x  - mingyuan supergroup          0 2017-09-20 16:01 input/dota2
-rw-r--r--  1 mingyuan supergroup 153248 2017-09-27 13:06 input/romeo_juliet.txt
airline/
bitcoin/
hadoop-mapreduce-examples-2.8.1-jar puzzle1.dta
installation_check.sh  romeo_juliet.txt
mingyuan@md-BigData:~/EECS6893/hw1/hadoop$ hadoop jar hadoop-mapreduce-examples-2.8.1-jar wordcount input/romeo_juliet.txt output/wordcount
17/09/27 13:07:01  WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
17/09/27 13:07:02 INFO InputFileInputFormat: Total input files = 1
17/09/27 13:07:02 INFO MapSubmitter: number of splits=1
17/09/27 13:07:02 INFO MapSubmitter: Submitting tokens for job: job_1506365520900_0022
17/09/27 13:07:03 INFO Impl.YarnClientImpl: Submitted application application_1506365520900_0022
17/09/27 13:07:03 INFO MapSubmitter: The url to access the job: http://md-BigData:8088/proxy/application_1506365520900_0022/
17/09/27 13:07:03 INFO MapSubmitter: Job job_1506365520900_0022 running
17/09/27 13:07:09 INFO MapSubmitter: Job job_1506365520900_0022 running in uber mode : false
17/09/27 13:07:09 INFO MapSubmitter: map 0% reduce 0%
17/09/27 13:07:14 INFO MapSubmitter: map 100% reduce 100%
17/09/27 13:07:19 INFO MapSubmitter: Job job_1506365520900_0022 completed successfully
17/09/27 13:07:19 INFO MapSubmitter: Counters: 49
File System Counters
FILE: Number of bytes read=8424
FILE: Number of bytes written=441253
FILE: Number of read operations=1
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=153364
HDFS: Number of bytes written=59326
HDFS: Number of read operations=6
HDFS: Number of large read operations=0
HDFS: Number of write operations=2
Job Counters
Launched map tasks=1
Launched reduce tasks=1
Decompressed total=1
Total time spent by all maps in occupied slots (ms)=2408
Total time spent by all reduces in occupied slots (ms)=2263
Total time spent by all map tasks (ms)=2408
Total time spent by all reduce tasks (ms)=2263
Total vcore-milliseconds taken by all map tasks=2408
Total vcore-milliseconds taken by all reduce tasks=2263
Total megabyte-milliseconds taken by all map tasks=2465792
Total megabyte-milliseconds taken by all reduce tasks=2317312
Map-Reduce Framework
Map input records=6052
Map output records=25652
Map output bytes=20799
Map output materialized bytes=84324
Input split bytes=123
Combine input records=25652
Combine output bytes=153364
Reduce input groups=6342
Reduce shuffle bytes=84324
Reduce input records=6342
Reduce output records=6342
Split input Record=12684
Shuffles=1
Failed Shuffles=0
Merged Map outputs=1
GC time elapsed (ms)=125
CPU time spent (ms)=1250
Physical memory (bytes) snapshots=434724864
Virtual memory (bytes) snapshots=3916529664
Total committed heap usage (bytes)=331874380
Shuffle Errors
BAD_ID=0
UNKNOWN_PARTITION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=153241
File Output Format Counters
Bytes Written=59326

```

b. MapReduce job results

```
m@m-BigData:~/EECS6893/hw1/hadoop$ hdfs dfs -ls output/
17/09/27 13:07:59 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
drwxr-xr-x  - mingyuan supergroup          0 2017-09-20 16:28 output/airline
drwxr-xr-x  - mingyuan supergroup          0 2017-09-26 13:31 output/bitcoin
drwxr-xr-x  - mingyuan supergroup          0 2017-09-27 13:07 output/wordcount
m@m-BigData:~/EECS6893/hw1/hadoop$ hdfs dfs -ls output/wordcount
17/09/27 13:08:04 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--  1 mingyuan supergroup      0 2017-09-27 13:07 output/wordcount/_SUCCESS
-rw-r--r--  1 mingyuan supergroup  59326 2017-09-27 13:07 output/wordcount/part-r-00000
```

```
m@m-BigData:~/EECS6893/hw1/hadoop/wordcount
mingyuan@m-BigData:~/EECS6893/hw1/hadoop/wordcount$ head part-r-00000
&          2
'An        1
'Ay        1
'Ay,       1
'Ay.'      4
'By        1
'God       1
'Heart's    3
'Hold,     1
```

Datasets

1. Employee resign dataset

Column Metadata

satisfaction_level	Level of satisfaction (0-1)	Numeric
last_evaluation	Time since last performance evaluation (in Years)	Numeric
number_project	Number of projects completed while at work	Numeric
average_montly_hours	Average monthly hours at workplace	Numeric
time_spend_company	Number of years spent in the company	Numeric
Work_accident	Whether the employee had a workplace accident	Numeric
left	Whether the employee left the workplace or not (1 or 0) Factor	Numeric
promotion_last_5years	Whether the employee was promoted in the last five years	Numeric
sales	Department in which they work for	String
salary	Relative level of salary (high)	String

Preview (first 100 rows) Column Metadata

satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	sales	salary
0.38	0.53	2	157	3	0	1	0	sales	low
0.8	0.86	5	262	6	0	1	0	sales	medium
0.11	0.88	7	272	4	0	1	0	sales	medium
0.72	0.87	5	223	5	0	1	0	sales	low
0.37	0.52	2	159	3	0	1	0	sales	low
0.41	0.5	2	153	3	0	1	0	sales	low
0.1	0.77	6	247	4	0	1	0	sales	low
0.92	0.85	5	259	5	0	1	0	sales	low
0.89	1	5	224	5	0	1	0	sales	low
0.42	0.53	2	142	3	0	1	0	sales	low
0.45	0.54	2	135	3	0	1	0	sales	low
0.11	0.81	6	305	4	0	1	0	sales	low
0.84	0.92	4	234	5	0	1	0	sales	low
0.41	0.55	2	148	3	0	1	0	sales	low
0.36	0.56	2	137	3	0	1	0	sales	low
0.38	0.54	2	143	3	0	1	0	sales	low
0.45	0.47	2	160	3	0	1	0	sales	low
0.78	0.99	4	255	6	0	1	0	sales	low
0.45	0.51	2	160	3	1	1	1	sales	low
0.76	0.89	5	262	5	0	1	0	sales	low
0.11	0.83	6	282	4	0	1	0	sales	low
0.38	0.55	2	147	3	0	1	0	sales	low
0.09	0.95	6	304	4	0	1	0	sales	low

2. San Francisco crime dataset

This dataset contains incidents derived from SFPD Crime Incident Reporting system. The data ranges from 1/1/2003 to 5/13/2015. The training set and test set rotate every week, meaning week 1,3,5,7... belong to test set, week 2,4,6,8 belong to training set.

2003-01-07 07:52:00	WARRANTS	WARRANT ARREST	Tuesday	SOUTHERN	ARREST, BOOKED	5TH ST / SHIPLEY ST	-122.402843	37.779829
2003-01-07 04:49:00	WARRANTS	ENROUTE TO OUTSIDE JURISDICTION	Tuesday	TENDERLOIN	ARREST, BOOKED	CYRIL MAGNIN STORTH ST / EDDY ST	-122.408495	37.784452
2003-01-07 03:52:00	WARRANTS	WARRANT ARREST	Tuesday	NORTHERN	ARREST, BOOKED	OFARRELL ST / LARKIN ST	-122.417904	37.785167
2003-01-07 03:34:00	WARRANTS	WARRANT ARREST	Tuesday	NORTHERN	ARREST, BOOKED	DIVISADERO ST / LOMBARD ST	-122.442650	37.798999
2003-01-07 01:22:00	WARRANTS	WARRANT ARREST	Tuesday	SOUTHERN	ARREST, BOOKED	900 Block of MARKET ST	-122.409537	37.782691
2003-01-06 23:30:00	WARRANTS	ENROUTE TO OUTSIDE JURISDICTION	Monday	BAYVIEW	ARREST, BOOKED	REVERE AV / INGALLS ST	-122.384557	37.728487
2003-01-06 23:14:00	WARRANTS	WARRANT ARREST	Monday	CENTRAL	ARREST, BOOKED	BUSH ST / HYDE ST	-122.417019	37.789110
2003-01-06 22:45:00	WARRANTS	WARRANT ARREST	Monday	SOUTHERN	ARREST, BOOKED	800 Block of BRYANT ST	-122.403405	37.775421
2003-01-06 22:45:00	WARRANTS	ENROUTE TO OUTSIDE JURISDICTION	Monday	SOUTHERN	ARREST, BOOKED	800 Block of BRYANT ST	-122.403405	37.775421
2003-01-06 22:19:00	WARRANTS	ENROUTE TO OUTSIDE JURISDICTION	Monday	NORTHERN	ARREST, BOOKED	GEARY ST / POLK ST	-122.419740	37.785893
2003-01-06 21:54:00	WARRANTS	ENROUTE TO OUTSIDE JURISDICTION	Monday	NORTHERN	ARREST, BOOKED	SUTTER ST / POLK ST	-122.420120	37.787757



satisfaction_level Level of satisfaction (0-1)

Numeric

last_evaluation Time since last performance evaluation (in Years)

Numeric

number_project Number of projects completed while at work

Numeric

average_montly_hours Average monthly hours at workplace

Numeric

time_spend_company Number of years spent in the company

Numeric

Work_accident Whether the employee had a workplace accident

Numeric

left Whether the employee left the workplace or not (1 or 0) Factor

Numeric

promotion_last_5years Whether the employee was promoted in the last five years

Numeric

sales Department in which they work for

String

salary Relative level of salary (high)

String

3. Dota2 player ratings dataset

About this file

number of wins, number of matches, and TrueSkill rating computed from 900k matches. Each players rating is made up of a mean value and variance(mu, and sigma). The trueskill values were calculated using the python library `trueskill`, with default values. Kernel with details should be available soonish

account_id	total_wins	total_matches	trueskill_mu	trueskill_sigma
236579	14	24	27.86803545	5.212360755
-343	1	1	26.54416264	8.065475476
-1217	1	1	26.52110284	8.114989027
-1227	1	1	27.24802496	8.092216692
-1284	0	1	22.93101578	8.092224309
308663	1	1	26.76147599	8.108879564
79749	21	40	30.55341692	3.868734418
-1985	0	1	23.26340923	8.098020436
-2160	8	12	27.42601792	6.391300187
26500	26	50	27.94362078	4.04900453
-2776	0	1	23.05352155	8.110911377
137046	46	89	26.02599788	2.865184074
56881	15	23	32.85642397	5.132468772
-3110	1	2	25.49104138	7.846784863
-3126	0	1	23.28169403	8.119258685
221202	6	12	22.75076351	6.19852854
-3161	0	3	20.48337426	7.757888786
-3194	0	1	22.89152055	8.100930345
221204	3	8	21.29320658	6.560565351
-3421	2	4	23.66957648	7.481699509
-3477	2	2	28.53452152	7.728687255
-3596	6	9	27.43509115	6.591948751
-3602	0	2	22.56004551	7.951011533
-3682	7	11	26.49582281	6.474783609
120495	16	28	27.34034403	4.892810732
-3776	0	4	19.03829843	7.538703814
-3787	3	6	25.15808346	6.877262472
-3911	5	7	27.44244554	6.93831947
-3970	2	4	24.22086229	7.088620515

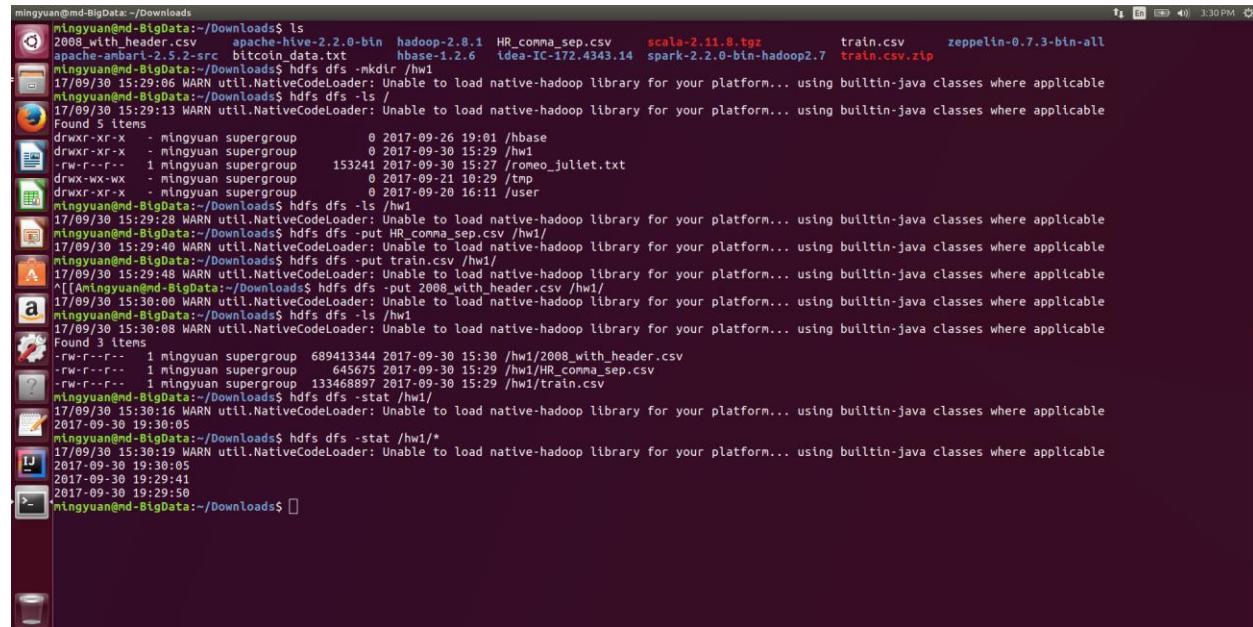
Hadoop Operation

1. -mkdir

2. -ls

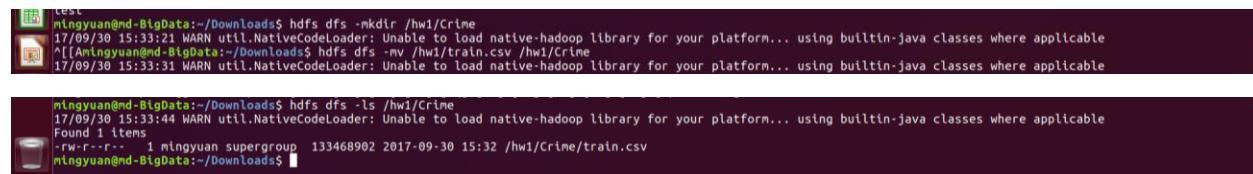
3. -put

4. -stat



```
mingyuan@md-BigData:~/Downloads$ hdfs dfs -mkdir /hw1
17/09/30 15:29:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mingyuan@md-BigData:~/Downloads$ hdfs dfs -ls /
17/09/30 15:29:13 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 5 items
drwxr-xr-x 1 mingyuan supergroup 0 2017-09-26 19:01 /base
drwxr-xr-x 1 mingyuan supergroup 0 2017-09-30 15:29 /hw1
-rw-r--r-- 1 mingyuan supergroup 153241 2017-09-30 15:27 /romeo_juliet.txt
drwxrwxr-x 1 mingyuan supergroup 0 2017-09-21 10:29 /tmp
drwxr-xr-x 1 mingyuan supergroup 0 2017-09-20 16:11 /user
mingyuan@md-BigData:~/Downloads$ hdfs dfs -ls /hw1
17/09/30 15:29:28 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mingyuan@md-BigData:~/Downloads$ hdfs dfs -put HR_comma_sep.csv /hw1/
17/09/30 15:29:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mingyuan@md-BigData:~/Downloads$ hdfs dfs -put train.csv /hw1/
17/09/30 15:29:48 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
^[[Amingyuan@md-BigData:~/Downloads$ hdfs dfs -put 2008_with_header.csv /hw1/
17/09/30 15:30:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mingyuan@md-BigData:~/Downloads$ hdfs dfs -ls /hw1
17/09/30 15:30:08 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 3 items
-rw-r--r-- 1 mingyuan supergroup 689413344 2017-09-30 15:30 /hw1/2008_with_header.csv
-rw-r--r-- 1 mingyuan supergroup 645675 2017-09-30 15:29 /hw1/HR_comma_sep.csv
-rw-r--r-- 1 mingyuan supergroup 133468897 2017-09-30 15:29 /hw1/train.csv
mingyuan@md-BigData:~/Downloads$ hdfs dfs -stat /hw1/
17/09/30 15:30:16 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2017-09-30 19:30:05
mingyuan@md-BigData:~/Downloads$ hdfs dfs -stat /hw1/*
17/09/30 15:30:19 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2017-09-30 19:30:05
2017-09-30 19:29:41
2017-09-30 19:29:50
mingyuan@md-BigData:~/Downloads$
```

5. -mv



```
mingyuan@md-BigData:~/Downloads$ hdfs dfs -mkdir /hw1/Crime
17/09/30 15:33:21 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
^[[Amingyuan@md-BigData:~/Downloads$ hdfs dfs -mv /hw1/train.csv /hw1/Crime
17/09/30 15:33:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

mingyuan@md-BigData:~/Downloads$ hdfs dfs -ls /hw1/Crime
17/09/30 15:33:44 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r-- 1 mingyuan supergroup 133468902 2017-09-30 15:32 /hw1/Crime/train.csv
mingyuan@md-BigData:~/Downloads$
```

6. -appendToFile

7. -tail



```
mingyuan@md-BigData:~/Downloads$ hdfs dfs -appendToFile test /hw1/train.csv
17/09/30 15:32:59 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
mingyuan@md-BigData:~/Downloads$ hdfs dfs -tail /hw1/train.csv
17/09/30 15:33:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
DE WITH A GUN,Monday,BAYVIEW,"ARREST, BOOKED",1500 Block of SHAFTER AV,-122.389768984951,37.7305636925712
878046,2003-01-06 00:20:00,OTHER OFFENSES,PAROLE VIOLATION,Monday,BAYVIEW,"ARREST, BOOKED",1500 Block of SHAFTER AV,-122.389768984951,37.7305636925712
878047,2003-01-06 00:01:00,LARCENY/THEFT,GRAND THEFT FROM LOCKED AUTO,Monday,INGLESIDE,NONE,600 Block of EDNA ST,-122.447363587104,37.7319475636101
878048,2003-01-06 00:01:00,LARCENY/THEFT,GRAND THEFT FROM LOCKED AUTO,Monday,SOUTHERN,NONE,5TH ST / FOLSOM ST,-122.403399364804,37.780265577696
878049,2003-01-06 00:01:00,VANDALISM,"MALICIOUS MISCHIEF, VANDALISM OF VEHICLES",Monday,SOUTHERN,NONE,TOWNSEND ST / 2ND ST,-122.39053140418699,37.78060707982429
878050,2003-01-06 00:01:00,FORGERY/COUNTERFEITING,"CHECKS, FORGERY (FELONY)",Monday,BAYVIEW,NONE,1800 Block of NEWCOMB AV,-122.394925721424,37.738211541051996
test
```

Hbase query

Dota2 Dataset

1. Dataset example

	A	B	C	D	E
1	account_id	total_wins	total_matches	trueskill_mu	trueskill_sigma
2	236579	14	24	27.86803545	5.212360755
3	-343	1	1	26.54416264	8.065475476
4	-1217	1	1	26.52110284	8.114989027
5	-1227	1	1	27.24802496	8.092216692
6	-1284	0	1	22.93101578	8.092224309
7	308663	1	1	26.76147599	8.108879564
8	79749	21	40	30.55341692	3.868734418
9	-1985	0	1	23.26340923	8.098020436
10	-2160	8	12	27.42601792	6.391300187
11	26500	26	50	27.94362078	4.04900453
12	-2776	0	1	23.05352155	8.110911377
13	137046	46	89	26.02599788	2.865184074
14	56881	15	23	32.85642397	5.132468772
15	-3110	1	2	25.49104138	7.846784863
16	-3126	0	1	23.28169403	8.119258685
17	221202	6	12	22.75076351	6.19852854
18	-3161	0	3	20.48337426	7.757888786
19	-3194	0	1	22.89152055	8.100930345
20	221204	3	8	21.29320658	6.560565351
21	-3421	2	4	23.66957648	7.481699509
22	-3477	2	2	28.53452152	7.728687255
23	-3596	6	9	27.43509115	6.591948751
24	-3602	0	2	22.56004551	7.951011533
25	-3682	7	11	26.49582281	6.474783609
26	120495	16	28	27.34034403	4.892810732
27	-3776	0	4	19.03829843	7.538703814
28	-3787	3	6	25.15808346	6.877262472
29	-3911	5	7	27.44244554	6.93831947
30	-3970	7	4	24.22086229	7.088620515

2. Hbase queries

a. Describe Hbase “PlayerRatings” table

```
-----  
1. describe 'PlayerRatings'  
-----  
hbase(main):007:0> describe 'PlayerRatings'  
  
Table PlayerRatings is ENABLED  
PlayerRatings  
COLUMN FAMILIES DESCRIPTION  
{NAME => 'total_matches', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
{NAME => 'total_wins', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
{NAME => 'trueskill_mu', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
{NAME => 'trueskill_sigma', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN VERSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
4 row(s) in 0.1390 seconds
```

b. Count total rows in “PlayerRatings” table and set INTERVAL to 100000

```
-----  
2.count 'PlayerRatings'  
-----  
  
hbase(main):009:0> count 'PlayerRatings', INTERVAL => 100000  
Current count: 100000, row: -120885589  
Current count: 200000, row: -149265190  
Current count: 300000, row: -190801670  
Current count: 400000, row: -255433468  
Current count: 500000, row: -70551592  
Current count: 600000, row: 118655  
Current count: 700000, row: 23721  
Current count: 800000, row: 62009  
834227 row(s) in 24.6220 seconds
```

c. Get 1 row from table

```
3. get 'PlayerRatings','1'                                1. Hadoop Installation  
-----  
hbase(main):016:0> get 'PlayerRatings','1'  
COLUMN          CELL  
total_matches:   timestamp=1506006301661, value=24  
total_wins:      timestamp=1506006301661, value=14  
trueskill_mu:    timestamp=1506006301661, value=26.232905478885414  
trueskill_sigma: timestamp=1506006301661, value=4.854237724893719  
4 row(s) in 0.0130 seconds
```

d. Get 1 element from table

```
-----  
4. get 'PlayerRating', '1', 'trueskill_mu'                1. Hadoop Installation  
-----  
hbase(main):008:0> get 'PlayerRatings', '1', 'trueskill_mu'  
COLUMN          CELL  
trueskill_mu:   timestamp=1506006301661, value=26.232905478885414  
1 row(s) in 0.0040 seconds
```

e. Put 1 row into table

```
5. put  
-----  
hbase(main):001:0> get 'PlayerRatings','1000000'  
COLUMN          CELL  
0 row(s) in 0.2190 seconds  
  
hbase(main):002:0> put 'PlayerRatings','1000000','total_matches', '0'  
0 row(s) in 0.0970 seconds  
  
hbase(main):003:0> get 'PlayerRatings','1000000'  
COLUMN          CELL  
total_matches:   timestamp=1506303169804, value=0  
1 row(s) in 0.0150 seconds  
  
hbase(main):006:0> put 'PlayerRatings','1000000','total_wins', '0'  
0 row(s) in 0.0310 seconds  
  
hbase(main):007:0> put 'PlayerRatings','1000000','trueskill_mu', '0'  
0 row(s) in 0.0110 seconds  
  
hbase(main):008:0> put 'PlayerRatings','1000000','trueskill_sigma', '0'  
0 row(s) in 0.0310 seconds  
  
hbase(main):009:0> get 'PlayerRatings','1000000'  
COLUMN          CELL  
total_matches:   timestamp=1506303169804, value=0  
total_wins:      timestamp=1506303229672, value=0  
trueskill_mu:    timestamp=1506303240168, value=0  
trueskill_sigma: timestamp=1506303243353, value=0  
4 row(s) in 0.0080 seconds
```

Hive Query

Dota2 Dataset

1. Create table and load dataset

```
-----  
1. Create table  
-----  
hive> CREATE TABLE IF NOT EXISTS data2.PlayerRatings (account_id BIGINT, total_matches INT, total_wins INT, trueskill_mu FLOAT, trueskill_sigma FLOAT)  
    > COMMENT 'Player Ratings' ROW FORMAT DELIMITED FIELDS TERMINATED BY ','  
    > LOCATION '/user/mingyuan/test';  
OK  
Time taken: 0.131 seconds
```

2. Select 5 players who have played more than 100 games

```
-----  
2. Select rows  
-----  
hive> Select * FROM PlayerRatings WHERE total_matches > 100 limit 5;  
  
OK  
37018    137    262    25.410934    2.001894  
1589     109    229    26.938852    2.0036588  
4880     119    238    25.396338    2.1643844  
168114   127    221    26.771683    2.2445276  
19609    139    251    30.792833    1.6721019  
Time taken: 2.552 seconds, Fetched: 5 row(s)
```

3. Row count

Let's see how many players out of 834227 have played more than 100 games.

```
-----  
3. Count rows  
-----  
hive> SELECT COUNT(*) FROM PlayerRatings WHERE total_matches > 100;  
  
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.  
Query ID = mingyuan_20170925103105_7a1fbff1-7e3b-479b-a397-a890cc1cf053  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks determined at compile time: 1  
In order to change the average load for a reducer (in bytes):  
set hive.exec.reducers.bytes.per.reducer=<number>  
In order to limit the maximum number of reducers:  
set hive.exec.reducers.max=<number>  
In order to set a constant number of reducers:  
set mapreduce.job.reduces=<number>  
Starting Job = job_1506276692849_0002, Tracking URL = http://md-BigData:8088/proxy/application_1506276692849_0002/  
Kill Command = /usr/local/hadoop-2.8.1/bin/hadoop job -kill job_1506276692849_0002  
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1  
2017-09-25 10:31:36,426 Stage-1 map = 0%, reduce = 0%  
2017-09-25 10:32:10,687 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 11.45 sec  
2017-09-25 10:32:35,563 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 19.82 sec  
MapReduce Total cumulative CPU time: 19 seconds 820 msec  
Ended Job = job_1506276692849_0002  
MapReduce Jobs Launched:  
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 19.82 sec HDFS Read: 41903928 HDFS Write: 103 SUCCESS  
Total MapReduce CPU Time Spent: 19 seconds 820 msec  
OK  
682  
Time taken: 92.607 seconds, Fetched: 1 row(s)
```

The answer is 682, which is a small number. It means only ~0.1% players in this dataset has played more than 100 games. Not as same as I expected.

4. More complex selection

```
4.
hive> SELECT * FROM PlayerRatings WHERE total_matches > 100 AND trueskill_mu > 35 ORDER BY trueskill_mu DESC;

WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or
using Hive 1.X releases.
Query ID = mingyuan_20170925103658_d58c33a2-a3d7-48ff-bf96-7d74495d5a57
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1506276692849_0003, Tracking URL = http://md-BigData:8088/proxy/application_1506276692849_0003/
Kill Command = /usr/local/hadoop-2.8.1/bin/hadoop job -kill job_1506276692849_0003
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-09-25 10:37:34,972 Stage-1 map = 0%,  reduce = 0%
2017-09-25 10:37:41,664 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 3.22 sec
2017-09-25 10:37:48,160 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.7 sec
MapReduce Total cumulative CPU time: 4 seconds 700 msec
Ended Job = job_1506276692849_0003
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1  Cumulative CPU: 4.7 sec  HDFS Read: 41904366 HDFS Write: 273 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 700 msec
Time taken: 51.361 seconds, Fetched: 4 row(s)
```

Select players who have played more than 100 games and have high trueskill score. This query only returned 4 players.

5. Group by number of games played

```
hive> SELECT ROUND(total_matches, -1), COUNT(*) FROM PlayerRatings GROUP BY ROUND(total_matches, -1);

WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or
using Hive 1.X releases.
Query ID = mingyuan_20170925104504_d527f2ca-bdff-4afc-a843-601bdaaa71c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks not specified. Estimated from input data size: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1506276692849_0008, Tracking URL = http://md-BigData:8088/proxy/application_1506276692849_0008/
Kill Command = /usr/local/hadoop-2.8.1/bin/hadoop job -kill job_1506276692849_0008
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2017-09-25 18:45:09,660 Stage-1 map = 0%,  reduce = 0%
2017-09-25 18:45:15,990 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 2.75 sec
2017-09-25 18:45:22,295 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 4.07 sec
MapReduce Total cumulative CPU time: 4 seconds 70 msec
Ended Job = job_1506276692849_0008
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 1  Cumulative CPU: 4.07 sec  HDFS Read: 41903461 HDFS Write: 738 SUCCESS
Total MapReduce CPU Time Spent: 4 seconds 70 msec
OK
0      677338
10     115818
20     22687
30     8393
40     3992
50     2229
60     1264
70     802
80     512
90     345
100    246
110    168
120    109
130    75
140    61
150    45
160    35
170    28
180    11
190    18
200    12
210    10
220    3
230    5
240    7
250    3
260    2
270    3
280    1
300    1
320    1
400    1
1608400 1
Time taken: 18.618 seconds, Fetched: 34 row(s)
```

6. Find the maximum trueskill score

```
-----  
6. Maximum trueskill_mu  
-----  
  
hive> SELECT MAX(trueskill_mu) AS label FROM PlayerRatings;  
  
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine  
(i.e. spark, tez) or using Hive 1.X releases.  
Query ID = mingyuan_20170925105042_e6231bf1-dcc5-4a25-b113-ac225fed7798  
Total jobs = 1  
Launching Job 1 out of 1  
Number of reduce tasks determined at compile time: 1  
In order to change the average load for a reducer (in bytes):  
set hive.exec.reducers.bytes.per.reducer=<number>  
In order to limit the maximum number of reducers:  
set hive.exec.reducers.max=<number>  
In order to set a constant number of reducers:  
set mapreduce.job.reduces=<number>  
Starting Job = job_1506276692849_0009, Tracking URL = http://md-BigData:8088/proxy/application_1506276692849_0009/  
Kill Command = /usr/local/Hadoop-2.8.1/bin/hadoop job -kill job_1506276692849_0009  
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1  
2017-09-25 10:50:46,933 Stage-1 map = 0%, reduce = 0%  
2017-09-25 10:50:53,234 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 3.05 sec  
2017-09-25 10:50:58,525 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 4.14 sec  
MapReduce Total cumulative CPU time: 4 seconds 140 msec  
Ended Job = job_1506276692849_0009  
MapReduce Jobs Launched:  
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 4.14 sec HDFS Read: 41903199 HDFS Write: 109 SUCCESS  
Total MapReduce CPU Time Spent: 4 seconds 140 msec  
OK  
48.825893  
Time taken: 18.539 seconds, Fetched: 1 row(s)
```

San Francisco Crime Dataset

1. Dataset Example

```
2 2015-05-13 23:53:00 WARRANTS WARRANT ARREST Wednesday NORTHERN "ARREST BOOKED" NULL -122.425891675136
3 2015-05-13 23:53:00 OTHER OFFENSES TRAFFIC VIOLATION ARREST Wednesday NORTHERN "ARREST BOOKED" NULL -122.425891675136
4 2015-05-13 23:53:00 OTHER OFFENSES TRAFFIC VIOLATION ARREST Wednesday NORTHERN "ARREST BOOKED" NULL -122.42436302145
5 2015-05-13 23:53:00 LARCENY/THEFT GRAND THEFT FROM LOCKED AUTO Wednesday NORTHERN NONE 1500 Block of LOMBARD ST -122.4269953267
```

3. Count crime numbers grouped by crime category

Query: SELECT Category, COUNT(*) AS cnt GROUP BY Category ORDER BY cnt DESC;

Category	cnt
LARCENY/THEFT	174900
OTHER OFFENSES	126182
NON-CRIMINAL	92304
ASSAULT	76876
DRUG/NARCOTIC	53971
VEHICLE THEFT	53781
VANDALISM	44725
WARRANTS	42214
BURGLARY	36755
SUSPICIOUS OCC	31414
MISSING PERSON	25989
ROBBERY	23000
FRAUD	16679
FORGERY/COUNTERFEITING	10609
SECONDARY CODES	9985
WEAPON LAWS	8555
PROSTITUTION	7484
TRESPASS	7326
STOLEN PROPERTY	4540
SEX OFFENSES FORCIBLE	4388
DISORDERLY CONDUCT	4320
DRUNKENNESS	4280
RECOVERED VEHICLE	3138
KIDNAPPING	2341
DRIVING UNDER THE INFLUENCE	
RUNAWAY	1946
LIQUOR LAWS	1903
ARSON	1513
LOITERING	1225
EMBEZZLEMENT	1166
SUICIDE	508
FAMILY OFFENSES	491
BAD CHECKS	406
BRIBERY	289
EXTORTION	256
SEX OFFENSES NON FORCIBLE	

Airline Dataset Analysis

Hadoop

1. Hadoop Job

a. Runner class

```
1 package EECS6893.airline;
2
3 import org.apache.hadoop.conf.Configuration;
4
5
6 public class AirlineAnalysisRunner extends Configuration implements Tool {
7     public int run(String args[]) throws Exception {
8         String inputDir = "input/airline/2008.csv";
9         String outputDir = "output/airline/day_of_week";
10        //String teamFight = "input/teamfights.csv";
11        Job job = Job.getInstance();
12        job.setJobName("test");
13        job.setJarByClass(AirlineAnalysisRunner.class);
14        //job.addCacheFile(new Path(teamFight).toUri());
15
16        job.setMapperClass(AirlineMapper.class);
17        job.setReducerClass(AirlineReducer.class);
18
19        job.setInputFormatClass(TextInputFormat.class);
20        TextInputFormat.addInputPath(job, new Path(inputDir));
21        //job.getConfiguration().setInt("mapreduce.input.lineinputformat.linespermap", 1);
22        FileOutputFormat.setOutputPath(job, new Path(outputDir));
23        job.setNumReduceTasks(1);
24
25        job.setMapOutputKeyClass(IntWritable.class);
26        job.setMapOutputValueClass(Text.class);
27        job.setOutputKeyClass(Text.class);
28        job.setOutputValueClass(Text.class);
29
30        return job.waitForCompletion(true) ? 0 : -1;
31    }
32
33    public static void main(String args[]) throws Exception {
34        AirlineAnalysisRunner Data2Driver = new AirlineAnalysisRunner();
35        int res = ToolRunner.run(Data2Driver, args);
36        System.exit(res);
37    }
38}
39
40}
```

b. Mapper class

```

Eclipse

AirlineAnalysisRunner.java AirlineMapper.java

1 package EECS6893.airline;
2
3+import java.io.IOException;
4
5
6 public class AirlineMapper extends Mapper <LongWritable, Text, IntWritable> {
7     public void map(LongWritable key, Text value, Context context) throws
8         IOException {
9             String[] airlines = value.toString().split(",");
10            int dayOfWeek = Integer.parseInt(airlines[3]);
11            context.write(new IntWritable(dayOfWeek), value);
12        }
13    }
14}

```

c. Reducer class

```

AirlineAnalysisRunner.java AirlineMapper.java *AirlineReducer.java

1 package EECS6893.airline;
2
3+import java.io.IOException;
4
5
6 public class AirlineReducer extends Reducer <IntWritable, Text, Text, Text> {
7     HashMap<Integer, String> res;
8
9     public void reduce(IntWritable key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
10        if (res == null) res = new HashMap<Integer, String>();
11        int sum = 0;
12        int delayedNum = 0;
13        int earlyNum = 0;
14        int depDelayedNum = 0;
15        int depEarlyNum = 0;
16        int cancelledNum = 0;
17        for (Text t: values) {
18            String[] info = t.toString().split(",");
19
20            if (!info[14].equals("NA")) {
21                int delay = Integer.parseInt(info[14]);
22                if (delay > 5) delayedNum++;
23                else if (delay < -5) earlyNum++;
24            }
25
26            if (!info[15].equals("NA")) {
27                int delay = Integer.parseInt(info[15]);
28                if (delay > 5) depDelayedNum++;
29                else if (delay < -5) depEarlyNum++;
30            }
31
32            if (info[21].equals("1")) cancelledNum++;
33
34            sum++;
35        }
36
37        String s = String.format("%-7d\t%-7d\t%2.1f%\t%-7d\t%2.1f%\t%-7d\t%2.1f%\t%-7d\t%2.1f%",
38                               sum,
39                               delayedNum, (double)delayedNum/sum * 100,
40                               depDelayedNum, (double)depDelayedNum/sum * 100,
41                               earlyNum, (double)earlyNum/sum * 100,
42                               depEarlyNum, (double)depEarlyNum/sum * 100);
43
44        //context.write(new Text(String.valueOf(day)), new Text(s));
45        res.put(key, s);
46    }
47}

```

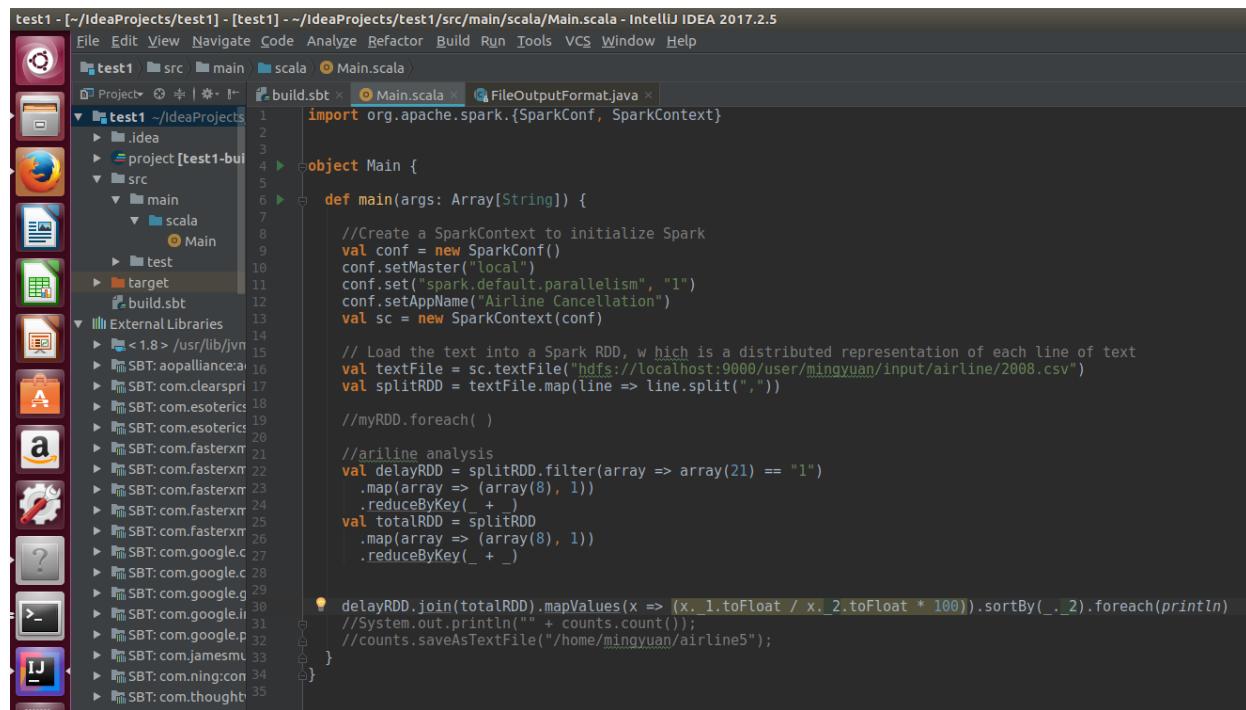
2. Result

In this MapReduce job, I am trying to find the Airline delay corresponding to Day_Of_Week information. And it is obvious that airlines in Friday have higher arrive and departure delay rates. Also, we can see that around 40% of airlines arrived 5 mins earlier than expected arriving time, but only about 17% of airlines take off 5 mins earlier than scheduled departure time! It could be possible that many of the airline flying time is less than expected flying time and needs further investigate.

Dow	Sum	ArrDly	pct	DepDly	pct	ArrEly	pct	DepEly	pct
1	1036201	338725	32.7%	290933	28.1%	393371	38.0%	163385	15.8%
2	1032049	320368	31.0%	260943	25.3%	409879	39.7%	187414	18.2%
3	1039665	319652	30.7%	262805	25.3%	413096	39.7%	186823	18.0%
4	1032224	347551	33.7%	289451	28.0%	379414	36.8%	164186	15.9%
5	1035166	376409	36.4%	323259	31.2%	355542	34.3%	148018	14.3%
6	857536	254964	29.7%	223256	26.0%	364271	42.5%	150516	17.6%
7	976887	325379	33.3%	286111	29.3%	374324	38.3%	157544	16.1%

Spark

1. Spark job



The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** The project is named "test1". It contains a "src" directory with "main" and "scala" sub-directories. "Main.scala" is the active file.
- Main.scala Content:**

```

import org.apache.spark.{SparkConf, SparkContext}

object Main {
  def main(args: Array[String]) {
    //Create a SparkContext to initialize Spark
    val conf = new SparkConf()
    conf.setMaster("local")
    conf.set("spark.default.parallelism", "1")
    conf.setAppName("Airline Cancellation")
    val sc = new SparkContext(conf)

    // Load the text into a Spark RDD, which is a distributed representation of each line of text
    val textFile = sc.textFile("hdfs://localhost:9000/user/mingyuan/input/airline/2008.csv")
    val splitRDD = textFile.map(line => line.split(","))
    //myRDD.foreach( )

    //airline analysis
    val delayRDD = splitRDD.filter(array => array(21) == "1")
      .map(array => (array(8), 1))
      .reduceByKey(_ + _)
    val totalRDD = splitRDD
      .map(array => (array(8), 1))
      .reduceByKey(_ + _)

    delayRDD.join(totalRDD).mapValues(x => (x._1.toFloat / x._2.toFloat * 100)).sortBy(_.2).foreach(println)
    //System.out.println(" " + counts.count());
    //counts.saveAsTextFile("/home/mingyuan/airline5");
  }
}

```

2. Result

17/09/27 15:49:46
(F9,0.31640944)
(AQ,0.5384615)
(NW,0.8358934)
(FL,0.8544657)
(HA,0.9219422)
(WN,1.0309098)
(CO,1.240388)
(AS,1.4156001)
(US,1.4510934)
(DL,1.507531)
(B6,1.6344452)
(EV,1.7913215)
(OO,2.1926832)
(UA,2.344972)
(XE,2.6680195)
(9E,2.707774)
(AA,2.8831928)
(OH,3.2701273)
(YV,3.6162868)
(MQ,3.735737)

We can see the highest cancelation rate is Envoy Air(MQ)'3.7% and the lowest cancelation rate is Frontier Airline(F9)'s 0.32%, which is only one tenth of the highest rate. Delta has the lowest cancelation rate 1.51% comparing to the other two big airline companies America Airline (2.88%) and United Airline (2.34). So, Delta might be a better choice if you don't want your flying to be canceled.

Bitcoin Dataset Analysis

Hadoop

1. Hadoop Jobs

```
BitCoinRunner.java - Eclipse
-----
BitcoinMapper.java BitcoinReducer.java BitCoinRunner.java
1 package EECS6893.bitcoin;
2
3
4@import org.apache.hadoop.conf.Configured;[]
5
6
7 public class BitCoinRunner extends Configured implements Tool {
8     public int run(String args[]) throws Exception {
9         String inputDir = "input/Bitcoin/bitcoin_data.txt";
10        String outputDir = "output/bitcoin";
11        Job job = Job.getInstance();
12        job.setJobName("bitcoin");
13        job.setJarByClass(BitCoinRunner.class);
14
15        job.setMapperClass(BitcoinMapper.class);
16        job.setReducerClass(BitcoinReducer.class);
17
18        job.setInputFormatClass(TextInputFormat.class);
19        TextInputFormat.addInputPath(job, new Path(inputDir));
20        //job.getConfiguration().setInt("mapreduce.input.lineinputformat.linespermap", 1);
21        FileOutputFormat.setOutputPath(job, new Path(outputDir));
22        job.setNumReduceTasks(1);
23
24        job.setMapOutputKeyClass(Text.class);
25        job.setMapOutputValueClass(Text.class);
26        job.setOutputKeyClass(Text.class);
27        job.setOutputValueClass(Text.class);
28
29        return job.waitForCompletion(true) ? 0 : -1;
30
31    }
32
33    public static void main(String args[]) throws Exception {
34        BitCoinRunner BitcoinDriver = new BitCoinRunner();
35        int res = ToolRunner.run(BitcoinDriver, args);
36        System.exit(res);
37    }
38
39 }
40 }
```

```
/BitcoinMapper.java - Eclipse
-----
BitcoinMapper.java BitcoinReducer.java BitCoinRunner.java
1 package EECS6893.bitcoin;
2
3@import java.io.IOException;[]
4
5 public class BitcoinMapper extends Mapper <LongWritable, Text, Text, Text> {
6     public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
7         String[] bitcoinTrans = value.toString().split(",");
8
9         String genderInfo = bitcoinTrans[5] + "->" + bitcoinTrans[6];
10        if (!bitcoinTrans[5].equals("gender_from"))
11            context.write(new Text(genderInfo), value);
12
13    }
14
15 }
16 }
```

/BitcoinReducer.java - Eclipse

```

BitcoinMapper.java *BitcoinReducer.java BitCoinRunner.java bitcoin_data.txt
1 package EECS6893.bitcoin;
2
3
4* import java.io.IOException;*
5
6
7 public class BitcoinReducer extends Reducer <Text, Text, Text, Text>{
8
9     public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
10         int num = 0;
11         long sum = 0;
12         for (Text t: values) {
13             num += 1;
14             String[] info = t.toString().split(",");
15             String amntS = info[info.length - 1];
16             double amnt = Double.parseDouble(amntS.substring(1, amntS.length()));
17             sum += amnt;
18
19         }
20         context.write(new Text(key),
21                     new Text(String.valueOf(num) + String.format("\t\t%.2f", (double)sum / num)));
22
23     }
24
25 }
26
27
28
29
30 }
```

2. Result

ECS6893/hw1/hadoop/bitcoin/bitcoin) - VIM

<u>Female->Female</u>	<u>12716</u>	<u>49425.42</u>
Female->Male	12708	50057.24
Male->Female	12737	50130.55
Male->Male	12839	50227.80

As we can see, the Hadoop MapReduce job output shows us the average amount of dollars in each transaction between genders. Female to female has the lowest average transaction amount 49425.42 and male to male has the highest average transaction amount 50227.80. Although the difference between genders is subtle (1.6% between F-F and M-M), this fact still shows that males tend to transact more amount of money in each transaction.

Spark

1. Spark job

[~/EECS6893/hw1/spark/bitcoin] - [bitcoin] - ~/EECS6893/hw1/spark/bitcoin/src/main/scala/Main.scala - IntelliJ IDEA 2017.2.5

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

bitcoin src main scala Main.scala

build.sbt Main.scala

```
1 import org.apache.spark.{SparkConf, SparkContext}
2
3
4 object Main {
5
6   def main(args: Array[String]) {
7
8     //Create a SparkContext to initialize Spark
9     val conf = new SparkConf()
10    conf.setMaster("local")
11    conf.set("spark.default.parallelism", "1")
12    conf.setAppName("Airline Cancellation")
13    val sc = new SparkContext(conf)
14
15    // Load the text into a Spark RDD, which is a distributed representation of each line of text
16    val textFile = sc.textFile("hdfs://localhost:9000/user/mingyuan/input/Bitcoin/bitcoin_data.txt")
17    val splitRDD = textFile.map(line => line.split(","))
18
19    //myRDD.foreach( )
20    // 0: index,
21    // 1: id-from,
22    // 2: id-to,
23    // 3: username_from,
24    // 4: username_to,
25    // 5: gender_from,
26    // 6: gender_to,
27    // 7: date,
28    // 8: amount
29
30    val amountRDD = splitRDD.filter(array => array(8) != "amount")
31      .map(array => ((array(8).substring(1).toFloat / 10000).toInt * 10000 + 10000, 1))
32      .reduceByKey(_ + _)
33
34    amountRDD.sortByKey().foreach(println)
35    //System.out.println("") + counts.count());
36    //counts.saveAsTextFile("/home/mingyuan/airline5");
37
38  }
39}
```

Main > main(args: Array[String])

Event Log

9/27/17 3:13 PM Auto build completed with errors

Auto build completed with errors (moments ago)

2. Result

(10000,5091)
(20000,5140)
(30000,5127)
(40000,5030)
(50000,5152)
(60000,5108)
(70000,5130)
(80000,4957)
(90000,5295)
(100000,4970)

Here I want to investigate the amount of money in each bitcoin transaction. It seems that the amount of money in each transaction are almost evenly distributed from 0 ~ 90000 dollars given an interval of 10000, which is different from what I expected. My previous thought is that people are willing to do large amount transaction using bitcoin. And we can see that many people are using bitcoin to send large amount of money (~\$100000).

Other analysis

Dota2 dataset linear regression

1. Load data

a. Define schema

```
-----  
1. Load data  
-----  
  
A. Define schema  
-----  
  
scala> val id = StructField("account_id", IntegerType)  
id: org.apache.spark.sql.types.StructField = StructField(account_id, IntegerType, true)  
  
scala> val total_matches = StructField("total_matches", IntegerType)  
total_matches: org.apache.spark.sql.types.StructField = StructField(total_matches, IntegerType, true)  
  
scala> val total_wins = StructField("total_wins", IntegerType)  
total_wins: org.apache.spark.sql.types.StructField = StructField(total_wins, IntegerType, true)  
  
scala> val trueskill_mu = StructField("trueskill_mu", DoubleType)  
trueskill_mu: org.apache.spark.sql.types.StructField = StructField(trueskill_mu, DoubleType, true)  
  
scala> val trueskill_sigma = StructField("trueskill_sigma", DoubleType)  
trueskill_sigma: org.apache.spark.sql.types.StructField = StructField(trueskill_sigma, DoubleType, true)  
  
scala> val fields = Array(id, total_matches, total_wins, trueskill_mu, trueskill_sigma)  
fields: Array[org.apache.spark.sql.types.StructField] = Array(StructField(account_id, IntegerType, true), StructField(total_matches, IntegerType, true), StructField(total_wins, IntegerType, true), StructField(trueskill_mu, DoubleType, true), StructField(trueskill_sigma, DoubleType, true))  
  
scala> val schema = StructType(fields)  
schema: org.apache.spark.sql.types.StructType = StructType(StructField(account_id, IntegerType, true), StructField(total_matches, IntegerType, true), StructField(total_wins, IntegerType, true), StructField(trueskill_mu, DoubleType, true), StructField(trueskill_sigma, DoubleType, true))
```

b. Load Dota2 player rating data

```
B. Load Dota2 player rating data.          1. Hadoop Installation  
-----  
  
scala> val df1 = spark.read.schema(schema).option("header", true).csv("hdfs://localhost:9000/user/mingyuan/pr/player_ratings.csv")  
df1: org.apache.spark.sql.DataFrame = [account_id: int, total_matches: int ... 3 more fields]
```

c. Show data

C. Show data

```
scala> df1.show
+-----+-----+-----+-----+
|account_id|total_matches|total_wins|trueskill_mu|trueskill_sigma|
+-----+-----+-----+-----+
| 236579|        14|       24| 27.86803544887669| 5.212360755154124|
| -343|         1|       1| 26.54416264494717| 8.06547547571326|
| -1217|         1|       1| 26.521102844769825| 8.114989027066821|
| -1227|         1|       1| 27.248024958648585| 8.092216692198111|
| -1284|         0|       1| 22.931015779479623| 8.092224309040315|
| 308663|         1|       1| 26.761475988284694| 8.108879563800533|
| 79749|        21|      40| 30.55341692111649| 3.8687344175280614|
| -1985|         0|       1| 23.263409234993325| 8.098020435825411|
| -2160|         8|      12| 27.4260179194793| 6.391300186726303|
| 26500|        26|      50| 27.94362077676565| 4.0490045303341855|
| -2776|         0|       1| 23.053521554831327| 8.1109113767047|
| 137046|        46|      89| 26.02599787812959| 2.865184074214484|
| 56881|        15|      23| 32.85642396786685| 5.132468772471912|
| -3110|         1|       2| 25.491041376532355| 7.846784862628624|
| -3126|         0|       1| 23.28169402641756| 8.11925868487977|
| 221202|         6|      12| 22.75076350501192| 6.198528539862464|
| -3161|         0|       3| 20.483374255793766| 7.757888786330731|
| -3194|         0|       1| 22.89152055167358| 8.100930344663169|
| 221204|         3|       8| 21.293206583904983| 6.560565350911063|
| -3421|         2|       4| 23.66957648440773| 7.481699509258572|
+-----+-----+-----+-----+
only showing top 20 rows
```

d. Print schema

D. Print schema

```
scala> df1.printSchema()
root
|-- account_id: integer (nullable = true)
|-- total_matches: integer (nullable = true)
|-- total_wins: integer (nullable = true)
|-- trueskill_mu: double (nullable = true)
|-- trueskill_sigma: double (nullable = true)
```

2. DataFrame count

2. DataFrame count

```
scala> df1.count()
res19: Long = 834226
```

3. Spark SQL

```
3. Spark SQL
scala> df1.createOrReplaceTempView("dota2Player")
scala> val dedicatedPlayerDF = spark.sql("SELECT account_id, total_matches FROM dota2Player WHERE total_matches > 100 ORDER BY total_matches DESC")
dedicatedPlayerDF: org.apache.spark.sql.DataFrame = [account_id: int, total_matches: int]
scala> dedicatedPlayerDF.show()
+-----+-----+
|account_id|total_matches|
+-----+-----+
|       0|      1608398|
|     6647|        396|
|   12357|        341|
|   77866|        302|
|  35770|        290|
|   8101|        274|
|  17194|        271|
|  55446|        269|
|  28485|        262|
|  24722|        255|
|   2765|        251|
| 138306|        247|
|  52594|        245|
|  33769|        243|
|   3627|        241|
| 10069|        240|
| 116900|        240|
|  96383|        237|
| 297414|        237|
| 282413|        235|
+-----+
only showing top 20 rows
```

Select player who has played more than 100 games. There is a player who has played 1608398 which is impossible. Might be a problem caused by csv header.

4. Naïve Linear Regression

a. Features assembler

```

4. Simple LR
-----+-----+-----+-----+
A. Features assembler
-----+-----+-----+-----+
scala> import org.apache.spark.ml.linalg.Vectors
import org.apache.spark.ml.linalg.Vectors
Hadoop Installation
scala> val assembler = new VectorAssembler().setInputCols(Array("total_matches", "total_wins")).setOutputCol("features")
assembler: org.apache.spark.ml.feature.VectorAssembler = vecAssembler_b78134c8668f
-----+-----+-----+-----+
scala> val featuresDF = assembler.transform(df)
featuresDF: org.apache.spark.sql.DataFrame = [account_id: int, total_matches: int ... 4 more fields]
-----+-----+-----+-----+
scala> featuresDF.show
+-----+-----+-----+-----+
|account_id|total_matches|total_wins|trueskill_mu|trueskill_sigma|features|
+-----+-----+-----+-----+
236579|14|24|27.86803544887669|5.212360755154124|[14.0,24.0]|
-343|1|1|26.54416264494717|8.06547547571326|[1.0,1.0]|
-1217|1|1|26.521102844769825|8.114989027066821|[1.0,1.0]|
-1227|1|1|27.248624958648585|8.092216692198111|[1.0,1.0]|
-1284|0|1|22.931015779479623|8.092224309040315|[0.0,1.0]|
308663|1|1|26.761475988284694|8.108879563800533|[1.0,1.0]|
79749|21|40|30.55341692111649|3.8687344175280614|[21.0,40.0]|
-1985|0|1|23.263409234993325|8.098020435825411|[0.0,1.0]|
-2160|8|12|27.4260179194793|6.391300186726303|[8.0,12.0]|
26500|26|50|27.94362077676565|4.0490045303341855|[26.0,50.0]|
-2776|0|1|23.053521554831327|8.1109113767047|[0.0,1.0]|
137046|46|89|26.02599787812959|2.865184074214484|[46.0,89.0]|
56881|15|23|32.85642396786685|5.132468772471912|[15.0,23.0]|
-3110|1|2|25.491041376532355|7.846784862628624|[1.0,2.0]|
-3126|0|1|23.28169402641756|8.11925868487977|[0.0,1.0]|
221202|6|12|22.75076350501192|6.198528539862464|[6.0,12.0]|
-3161|0|3|20.483374255793766|7.757888786330731|[0.0,3.0]|
-3194|0|1|22.89152055167358|8.100930344663169|[0.0,1.0]|
221204|3|8|21.293206583904983|6.560565350911063|[3.0,8.0]|
-3421|2|4|23.66957648446773|7.481699509258572|[2.0,4.0]|
-----+-----+-----+-----+
CS6893/hw1/spark/spark_shell_dota2/spark.txt [FORMAT=unix] [TYPE=TEXT] [POS=159,1][56%] 27/09/17 - 13:19

```

b. Change column names

ECS6893/hw1/spark/spark_shell_data2) - VIM

B. Change column name

```
scala> val newN = Seq("id", " ", " ", "label", " ", "features")
newN: Seq[String] = List(id, " ", " ", label, " ", features)

scala> val lrDF=featuresDF.toDF(newN:_*)
lrDF: org.apache.spark.sql.DataFrame = [id: int, : int ... 4 more fields]

scala> lrDF.show
+-----+-----+-----+
|   id|      |      label|           |    features|
+-----+-----+-----+
| 236579|  14|   24| 27.86803544887669| 5.212360755154124|[14.0,24.0]|
| -343|   1|   1| 26.54416264494717| 8.06547547571326|[1.0,1.0]|
| -1217|   1|   1| 26.521102844769825| 8.114989027066821|[1.0,1.0]|
| -1227|   1|   1| 27.248024958648585| 8.092216692198111|[1.0,1.0]|
| -1284|   0|   1| 22.931015779479623| 8.092224309040315|[0.0,1.0]|
| 308663|   1|   1| 26.761475988284694| 8.108879563800533|[1.0,1.0]|
| 79749|  21|   40| 30.55341692111649| 3.8687344175280614|[21.0,40.0]|
| -1985|   0|   1| 23.263409234993325| 8.098020435825411|[0.0,1.0]|
| -2160|   8|   12| 27.4260179194793| 6.391300186726303|[8.0,12.0]|
| 26500|  26|   50| 27.94362077676565| 4.0490045303341855|[26.0,50.0]|
| -2776|   0|   1| 23.053521554831327| 8.1109113767047|[0.0,1.0]|
| 137046|  46|   89| 26.02599787812959| 2.865184074214484|[46.0,89.0]|
| 56881|  15|   23| 32.85642396786685| 5.132468772471912|[15.0,23.0]|
| -3110|   1|   2| 25.491041376532355| 7.846784862628624|[1.0,2.0]|
| -3126|   0|   1| 23.28169402641756| 8.11925868487977|[0.0,1.0]|
| 221202|   6|   12| 22.75076350501192| 6.198528539862464|[6.0,12.0]|
| -3161|   0|   3| 20.483374255793766| 7.757888786330731|[0.0,3.0]|
| -3194|   0|   1| 22.89152055167358| 8.100930344663169|[0.0,1.0]|
| 221204|   3|   8| 21.293206583904983| 6.560565350911063|[3.0,8.0]|
| -3421|   2|   4| 23.66957648440773| 7.481699509258572|[2.0,4.0]|
+-----+-----+-----+
only showing top 20 rows
```

c. Split data into training set and validation set

```
C. Split data into training data and validation data
-----
scala> val split = lrDF.randomSplit(Array(0.9,0.1))
split: Array[org.apache.spark.sql.Dataset[org.apache.spark.sql.Row]] = Array([id: int, : int ... 4 more fields], [id: int, : int ... 4 more fields])
1 Hadoop Installation
scala> val train = split(0).cache()
train: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: int, : int ... 4 more fields]
scala> val validation = split(1)
validation: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [id: int, : int ... 4 more fields]
```

d. Train a simple LR model

```
D. Train a simple linear regression model
-----
scala> val model = lr.fit(train)
17/09/26 10:13:36 WARN optim.WeightedLeastSquares: regParam is zero, which might cause numerical instability and overfitting.
17/09/26 10:13:36 WARN netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
17/09/26 10:13:36 WARN netlib.BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
17/09/26 10:13:38 WARN netlib.LAPACK: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
17/09/26 10:13:38 WARN netlib.LAPACK: Failed to load implementation from: com.github.fommil.netlib.NativeRefLAPACK
model: org.apache.spark.ml.regression.LinearRegressionModel = linReg_b23de0832693
```

e. Show results

```
E. Show results
-----
scala> model.transform(validation).show()
+-----+-----+-----+-----+
| id | | label | | features | | prediction |
+-----+-----+-----+-----+
|-128396584| 1| 27.918812376134106| 8.053308860258714|[1.0,1.0]|25.636305865382393|
|-128396327| 2| 23.522165162753314| 7.258348927823103|[2.0,5.0]|24.238425636682162|
|-128391906| 1| 28.161518027242852| 8.019844081617187|[1.0,1.0]|25.636305865382393|
|-128385801| 3| 26.848128667469677| 7.370802079258343|[3.0,4.0]|26.445372362442924|
|-128384439| 0| 19.303048628447982| 7.534733942409549|[0.0,4.0]| 21.98742835783724|
|-128382256| 0| 22.554251369341607| 7.990255691229144|[0.0,2.0]|23.429359139621635|
|-128379426| 2| 31.09536931318958| 7.7954858068257025|[2.0,2.0]| 26.40132180935876|
|-128362765| 1| 23.38138537665235| 7.533235212298229|[1.0,3.0]|24.194375083597997|
|-128357274| 0| 23.54096127606779| 8.09166616855586|[0.0,1.0]|24.150324530513835|
|-128356857| 1| 27.07600480063651| 7.7138069904713555|[1.0,2.0]|24.915340474490197|
|-128356527| 1| 26.14823703813555| 8.143672786544798|[1.0,1.0]|25.636305865382393|
|-128355496| 2| 26.589489067080606| 7.668522592115432|[2.0,3.0]| 25.68035641846656|
|-128353485| 3| 31.989149994214028| 7.5491212159433845|[3.0,3.0]| 27.16633775333512|
|-128353222| 1| 28.025719800232352| 7.963141354806845|[1.0,1.0]|25.636305865382393|
|-128349146| 1| 21.07936493484888| 7.441836835256949|[1.0,4.0]| 23.4734096927058|
|-128348482| 0| 20.15261750242769| 7.784772142372692|[0.0,2.0]|23.429359139621635|
|-128344989| 2| 30.467669883864204| 7.754308395571972|[2.0,2.0]| 26.40132180935876|
|-128343728| 1| 27.75372310889385| 8.029805259888121|[1.0,1.0]|25.636305865382393|
|-128338893| 2| 27.912820097659736| 7.3226383507304025|[2.0,4.0]|24.959391027574362|
|-128336617| 2| 28.016630392213372| 7.967178037130402|[2.0,2.0]| 26.40132180935876|
+-----+
only showing top 20 rows
```

Validation Set

```

scala> model.transform(train).show()
+-----+-----+-----+-----+-----+
|     id|    |     label|          | features| prediction|
+-----+-----+-----+-----+-----+
|-128398173| 10| 27| 20.689088691544| 5.0831419702521385|[10.0,27.0]| 20.265037716002297|
|-128397829| 1| 3| 24.065572096781725| 7.497539429075076|[1.0,3.0]| 24.194375083597997|
|-128397797| 1| 1| 26.97496663166592| 8.052270122623218|[1.0,1.0]| 25.636305865382393|
|-128397047| 0| 1| 23.638990984594173| 8.106116057289787|[0.0,1.0]| 24.150324530513835|
|-128396750| 2| 3| 26.630882351026454| 7.660736520814687|[2.0,3.0]| 25.68035641846656|
|-128396552| 5| 7| 27.736702156235467| 6.863038043612407|[5.0,7.0]| 27.254438859503452|
|-128395862| 0| 1| 23.120203102063922| 8.02096736583166|[0.0,1.0]| 24.150324530513835|
|-128395592| 38| 59| 30.24520249380323| 3.69801583396422|[38.0,59.0]| 38.80162258377168|
|-128395541| 5| 11| 22.26825708994913| 6.428754519813465|[5.0,11.0]| 24.37057729593466|
|-128395390| 1| 1| 28.334513967790194| 8.019769727806871|[1.0,1.0]| 25.636305865382393|
|-128395244| 1| 3| 23.465269461896373| 7.612651710816195|[1.0,3.0]| 24.194375083597997|
|-128395197| 0| 5| 18.39246980228457| 7.365140373595669|[0.0,5.0]| 21.266462966945042|
|-128395082| 4| 4| 32.202466398183304| 7.466894460617119|[4.0,4.0]| 27.931353697311486|
|-128395051| 0| 2| 21.32127820261808| 7.877511119478758|[0.0,2.0]| 23.429359139621635|
|-128394880| 1| 1| 28.043870756278054| 7.97881206147458|[1.0,1.0]| 25.636305865382393|
|-128394278| 1| 1| 28.228062271998144| 7.9949827798162545|[1.0,1.0]| 25.636305865382393|
|-128393384| 2| 2| 28.841544567657408| 7.8666392457260095|[2.0,2.0]| 26.40132180935876|
|-128393117| 0| 1| 23.05333232104851| 8.057701028697512|[0.0,1.0]| 24.150324530513835|
|-128392938| 1| 1| 27.478545260286488| 7.962923344228324|[1.0,1.0]| 25.636305865382393|
|-128391577| 5| 13| 20.851925696264416| 6.010443876522932|[5.0,13.0]| 22.928646514150266|
+-----+-----+-----+-----+-----+
only showing top 20 rows

```

Training Set

Human Resource dataset decision tree prediction

1. Code

```
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.regression.LabeledPoint
import org.apache.spark.mllib.tree.DecisionTree
object Main {
  def main(args: Array[String]) {
    val conf = new SparkConf()
    conf.setMaster("local")
    conf.set("spark.default.parallelism", "1")
    conf.setAppName("HR")
    val sc = new SparkContext(conf)
    case class HR(id: Int,
                  sat: Double,
                  last_evl: Double,
                  num_p: Int,
                  amh: Int,
                  tsc: Int,
                  wa: Int,
                  left: Int,
                  pro: Int,
                  sales: String,
                  sala: String)
    def parseHR(str: String): HR = {
      val line = str.split(",")
      HR(line(0).toInt, line(1).toDouble, line(2).toDouble, line(3).toInt,
          line(4).toInt, line(5).toInt, line(6).toInt, line(7).toInt,
          line(8).toInt, line(9), line(10))
    }
    val textRDD =
      sc.textFile("hdfs://localhost:9000/user/mingyuan/input/HR/HR_comma_sep.csv")
    val header = textRDD.first()
    val dropHeaderRDD = textRDD.filter(row => row != header)
    val hrRDD = dropHeaderRDD.map(parseHR).cache()
    val salesRDD = hrRDD.map(line => (line.sales, 1)).reduceByKey(_ + _)
    salesRDD.foreach(println)
    var salesMap: Map[String, Int] = Map()
    var index: Int = 0
    hrRDD.map(hr => hr.sales).distinct.collect.foreach(x => {
      salesMap += (x -> index);
      index += 1
    })
  }
}
```

```

    })
    println(salesMap.toString())
    var salaryMap: Map[String, Int] = Map()
    var index1: Int = 0
    hrRDD.map(hr => hr.sala).distinct.collect.foreach(x => {
        salaryMap += (x -> index1);
        index1 += 1
    })
    println(salaryMap.toString())
    val mlprep = hrRDD.map(hr => {
        val sat = hr.sat
        val eval = hr.last_evl
        val proj = hr.num_p
        val amh = hr.amh
        val tsc = hr.tsc
        val wa = hr.wa //category
        val left = hr.left //category
        val pro = hr.pro //category
        val sales = salesMap(hr.sales) //category
        val sala = salaryMap(hr.sala) //category
        Array(left.toDouble, sat.toDouble, eval.toDouble, proj.toDouble,
            amh.toDouble, tsc.toDouble, wa.toDouble, pro.toDouble, sales.toDouble,
            sala.toDouble)
    })
    mlprep.take(10).foreach(println)
    val mldata = mlprep.map(x => LabeledPoint(x(0), Vectors.dense(x(1),
        x(2), x(3), x(4), x(5), x(6), x(7), x(8), x(9))))
    val mldata0 = mldata.filter(x => x.label == 0).randomSplit(Array(0.8, 0.2))(1)
    val mldata1 = mldata.filter(x => x.label != 0)
    val mldata2 = mldata0 ++ mldata1
    val splits = mldata2.randomSplit(Array(0.7, 0.3))
    val (trainingData, testData) = (splits(0), splits(1))
    var categoricalFeaturesInfo = Map[Int, Int]()
    categoricalFeaturesInfo += (5 -> 2)
    categoricalFeaturesInfo += (6 -> 2)
    categoricalFeaturesInfo += (7 -> salesMap.size)
    categoricalFeaturesInfo += (8 -> salaryMap.size)
    val numClasses = 2
    val impurity = "gini"
    val maxDepth = 9
    val maxBins = 5000
    val model = DecisionTree.trainClassifier(trainingData, numClasses,
        categoricalFeaturesInfo, impurity, maxDepth, maxBins)

```

```

    println(model.toDebugString)
    val labelAndPreds = testData.map { point =>
      val prediction = model.predict(point.features)
      (point.label, prediction)
    }
    val res = labelAndPreds.map(x => ((x._1, x._1.equals(x._2)), 1))
      .reduceByKey(_ + _)
    res.collect().foreach(println)
  }
}

```

2. Result

a. Jobs summary

```

17/09/30 13:08:33 INFO
(marketing,858)
(IT,1227)
(RandD,787)
(technical,2720)
(accounting,767)
(sales,4140)
(product_mng,902)
(management,630)
(hr,739)
(support,2229)
17/09/30 13:08:33 INFO

```

We can see that the job classes distribution is very similar to our intuition. The number of sales job is 4140, which is the most, comparing to the number management job 630.

b. Trained decision tree

Here I split the dataset into training data and test data, given the proportion 0.8,0.2. And I used the decision tree model from mllib to train a model to predict whether an employee quitted company or not given some data during his or her employment. The trained decision tree is like this:

```
chooseSplits: 3.055918095
DecisionTreeModel classifier of depth 9 with 187 nodes
If (feature 0 <= 0.46)
  If (feature 4 <= 4.0)
    If (feature 3 <= 121.0)
      Predict: 0.0
    Else (feature 3 > 121.0)
      If (feature 4 <= 2.0)
        If (feature 0 <= 0.14)
          If (feature 3 <= 214.0)
            Predict: 1.0
          Else (feature 3 > 214.0)
            Predict: 0.0
        Else (feature 0 > 0.14)
          If (feature 7 in {0.0,1.0,6.0,9.0,2.0,7.0})
            Predict: 0.0
          Else (feature 7 not in {0.0,1.0,6.0,9.0,2.0,7.0})
            If (feature 1 <= 0.46)
              Predict: 0.0
            Else (feature 1 > 0.46)
              If (feature 0 <= 0.36)
                If (feature 0 <= 0.21)
                  Predict: 0.0
                Else (feature 0 > 0.21)
                  Predict: 1.0
              Else (feature 0 > 0.36)
                If (feature 3 <= 171.0)
                  Predict: 0.0
                Else (feature 3 > 171.0)
                  Predict: 1.0
            Else (feature 4 > 2.0)
              If (feature 1 <= 0.42)
                Predict: 0.0
              Else (feature 1 > 0.42)
                If (feature 2 <= 2.0)
                  If (feature 1 <= 0.57)
                    If (feature 0 <= 0.32)
                      If (feature 0 <= 0.29)
                        Predict: 1.0
                      Else (feature 0 > 0.29)
                        Predict: 0.0
                    Else (feature 0 > 0.32)
                      If (feature 3 <= 161.0)
                        Predict: 1.0
                      Else (feature 3 > 161.0)
                        Predict: 0.0
                Else (feature 1 > 0.57)
                  If (feature 4 <= 3.0)
                    Predict: 0.0
```

And the prediction accuracy is 95.4%

```
((0.0, false), 31)
((1.0, true), 1031)
((1.0, false), 51)
((0.0, true), 646)
```