DATASCI W261: Machine Learning at Scale

- **Sayantan Satpati**
- **sayantan.satpati@ischool.berkeley.edu**

- **W261**
- **Week-2**
- **Assignment-3**
- **Date of Submission: 15-SEP-2015**

=== Week 2 ASSIGNMENTS using Hadoop Streaming and Python ===

HW2.0. What is a race condition in the context of parallel computation? Give an example.

In the context of parallel computation, a race condition signifies a programming fault producing undetermined program state and behavior due to un-synchronized parallel program executions. One example can be a shared variable in the memory - for example a HashMap - which is written to and read from by multiple threads.

What is MapReduce?

MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.

How does it differ from Hadoop?

Google MapReduce and Hadoop are two different implementations of the MapReduce framework/concept. Hadoop is open source while Google MapReduce is not, and actually there are not so many available details about it. However, the basic undelying principles of both are the same.

Which programming paradigm is Hadoop based on? Explain and give a simple example in code and show the code running.

Hadoop is based on the paradigm of divide and conquer - dividing a large problem into chunks which can be solved in parallel, by moving compute to data.

Example of a simple map reduce is show in the next cell. This example shows how we can use hadoop

In [24]:
```
'''
HW2.0. Which programming paradigm is Hadoop based on? Explain and give a simple example in cod
e and show the code running.

The following simple example demonstrates how a Hadoop Streaming program based on simple Unix
Commands can be used to
extract some useful information. In this case, it cuts the 2nd column of enronemail_1h.txt and
finds out the number of
categories (ham & spam) that the emails are categorized into. The mapper cuts the 2nd column u
sing the Unix cut command.
The reducer then finds the unique categories using the Unix uniq command.
'''


# Delete existing Output Dirs if available
!hadoop fs -rm -r -skipTrash /user/cloudera/w261/wk2/hw20/output

# Run the Hadoop Streaming Command
!hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh
4.7.0.jar \
-input /user/cloudera/w261/wk2/hw20/input/enronemail_1h.txt \
-output /user/cloudera/w261/wk2/hw20/output \
-mapper 'cut -f2' \
-reducer 'uniq'

# Show Output
!hadoop fs -cat /user/cloudera/w261/wk2/hw20/output/part-00000
```

```
Deleted /user/cloudera/w261/wk2/hw20/output
packageJobJar: [/tmp/hadoop-cloudera/hadoop-unjar6123782021518346907/] [] /tmp/streamjob1249
243011904010815.jar tmpDir=null
15/09/14 13:17:04 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
Applications should implement Tool for the same.
15/09/14 13:17:04 INFO mapred.FileInputFormat: Total input paths to process : 1
15/09/14 13:17:05 INFO streaming.StreamJob: getLocalDirs(): [/tmp/hadoop-cloudera/mapred/loc
al]
15/09/14 13:17:05 INFO streaming.StreamJob: Running job: job_201509131822_0052
15/09/14 13:17:05 INFO streaming.StreamJob: To kill this job, run:
15/09/14 13:17:05 INFO streaming.StreamJob: UNDEF/bin/hadoop job  -Dmapred.job.tracker=local
host.localdomain:8021 -kill job_201509131822_0052
15/09/14 13:17:05 INFO streaming.StreamJob: Tracking URL: http://0.0.0.0:50030/jobdetails.js
p?jobid=job_201509131822_0052
15/09/14 13:17:06 INFO streaming.StreamJob:  map 0%  reduce 0%
15/09/14 13:17:18 INFO streaming.StreamJob:  map 100%  reduce 0%
15/09/14 13:17:24 INFO streaming.StreamJob:  map 100%  reduce 100%
15/09/14 13:17:27 INFO streaming.StreamJob: Job complete: job_201509131822_0052
15/09/14 13:17:27 INFO streaming.StreamJob: Output: /user/cloudera/w261/wk2/hw20/output
0
1
```

In [1]:
```
%%writefile random_num_generation.py
#!/usr/bin/python

from random import randint
with open('random.txt', 'w') as f:
    for i in xrange(0,10000):
        r_number = randint(0,10000)
        f.write('{0},"NA"\n'.format(r_number))
```

```
Overwriting random_num_generation.py
```

In [2]:
```
!chmod a+x random_num_generation.py
```

In [3]:
```
%%writefile identity_map_red_hw21.py
#!/usr/bin/python
import sys
for line in sys.stdin:

    tokens = line.strip().split(",")
    print "%s" %(tokens[0])
```

```
Overwriting identity_map_red_hw21.py
```

In [4]:
```
!chmod a+x identity_map_red_hw21.py
```

In [5]:
```
'''
HW2.1. Sort in Hadoop MapReduce
'''

# Delete existing Output Dirs if available
!hadoop fs -rm -r -skipTrash /user/cloudera/w261/wk2/hw21/output

# Run the Hadoop Streaming Command
!hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh
4.7.0.jar \
-D mapred.output.key.comparator.class=org.apache.hadoop.mapred.lib.KeyFieldBasedComparator \
-D  mapred.text.key.comparator.options=-n \
-input /user/cloudera/w261/wk2/hw21/input/random.txt \
-output /user/cloudera/w261/wk2/hw21/output \
-file ./identity_map_red_hw21.py \
-mapper ./identity_map_red_hw21.py \
-file ./identity_map_red_hw21.py \
-reducer ./identity_map_red_hw21.py

# Show Output
!hadoop fs -cat /user/cloudera/w261/wk2/hw21/output/part-00000 | head -10
```

```
Deleted /user/cloudera/w261/wk2/hw21/output
packageJobJar: [./identity_map_red_hw21.py, ./identity_map_red_hw21.py, /tmp/hadoop-clouder
a/hadoop-unjar3193611839852604013/] [] /tmp/streamjob1586304532498584956.jar tmpDir=null
15/09/14 12:13:52 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
Applications should implement Tool for the same.
15/09/14 12:13:52 INFO mapred.FileInputFormat: Total input paths to process : 1
15/09/14 12:13:53 INFO streaming.StreamJob: getLocalDirs(): [/tmp/hadoop-cloudera/mapred/loc
al]
15/09/14 12:13:53 INFO streaming.StreamJob: Running job: job_201509131822_0041
15/09/14 12:13:53 INFO streaming.StreamJob: To kill this job, run:
15/09/14 12:13:53 INFO streaming.StreamJob: UNDEF/bin/hadoop job  -Dmapred.job.tracker=local
host.localdomain:8021 -kill job_201509131822_0041
15/09/14 12:13:53 INFO streaming.StreamJob: Tracking URL: http://0.0.0.0:50030/jobdetails.js
p?jobid=job_201509131822_0041
15/09/14 12:13:54 INFO streaming.StreamJob:  map 0%  reduce 0%
15/09/14 12:14:08 INFO streaming.StreamJob:  map 100%  reduce 0%
15/09/14 12:14:16 INFO streaming.StreamJob:  map 100%  reduce 100%
15/09/14 12:14:19 INFO streaming.StreamJob: Job complete: job_201509131822_0041
15/09/14 12:14:19 INFO streaming.StreamJob: Output: /user/cloudera/w261/wk2/hw21/output
0
2
2
3
3
3
4
6
7
9
```

In [6]:
```
%%writefile mapper_hw22.py

#!/usr/bin/env python

import sys
import re

def strip_special_chars(word):
    return re.sub('[^A-Za-z0-9]+', '', word)


for line in sys.stdin:
    try:
        # Remove leading & trailing chars
        line = line.strip()
        # Split the line by <TAB> delimeter
        email = re.split(r'\t+', line)

        # Check whether Content is present
        if len(email) < 4:
            continue

        # Get the content as a list of words
        content = email[len(email) - 1].split()

        for w in content:
            w = strip_special_chars(w)
            if w == 'assistance':
                print '%s\t%d' % (w, 1)
    except Exception as e:
        print line
        print e
```

```
Overwriting mapper_hw22.py
```

In [7]:
```
!chmod a+x mapper_hw22.py
```

In [8]:
```
%%writefile reducer_hw22.py


#!/usr/bin/env python

import sys
import re

word = None
count = 0

for line in sys.stdin:
    # Remove leading & trailing chars
    line = line.strip()
    # Split the line by <TAB> delimeter
    wc = re.split(r'\t+', line)

    word = wc[0]
    count += int(wc[1])

print '%s\t%d' % (word, count)
```

```
Overwriting reducer_hw22.py
```

In [9]:
```
!chmod a+x reducer_hw22.py
```

In [10]:
```
'''
HW2.2. Using the Enron data from HW1 and Hadoop MapReduce streaming,
write mapper/reducer pair that  will determine the number of occurrences of a single,
user-specified word. Examine the word “assistance” and report your results.
'''

# Delete existing Output Dirs if available
!hadoop fs -rm -r -skipTrash /user/cloudera/w261/wk2/hw22/output

!ls -l *hw22.py

# Run the Hadoop Streaming Command
!hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh

4.7.0.jar \
-D mapred.reduce.tasks = 1 \
-input /user/cloudera/w261/wk2/hw22/input/enronemail_1h.txt \
-output /user/cloudera/w261/wk2/hw22/output \
-file ./mapper_hw22.py \
-mapper 'python mapper_hw22.py' \
-file ./reducer_hw22.py \
-reducer 'python reducer_hw22.py'

# Show Output
!hadoop fs -cat /user/cloudera/w261/wk2/hw22/output/part-00000
```

```
Deleted /user/cloudera/w261/wk2/hw22/output
-rwxrwxr-x 1 cloudera cloudera 695 Sep 14 12:14 mapper_hw22.py
-rwxrwxr-x 1 cloudera cloudera 307 Sep 14 12:14 reducer_hw22.py
packageJobJar: [./mapper_hw22.py, ./reducer_hw22.py, /tmp/hadoop-cloudera/hadoop-unjar440427
0376137637736/] [] /tmp/streamjob6249198798003576990.jar tmpDir=null
15/09/14 12:14:26 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.
Applications should implement Tool for the same.
15/09/14 12:14:26 INFO mapred.FileInputFormat: Total input paths to process : 1
15/09/14 12:14:27 INFO streaming.StreamJob: getLocalDirs(): [/tmp/hadoop-cloudera/mapred/loc
al]
15/09/14 12:14:27 INFO streaming.StreamJob: Running job: job_201509131822_0042
15/09/14 12:14:27 INFO streaming.StreamJob: To kill this job, run:
15/09/14 12:14:27 INFO streaming.StreamJob: UNDEF/bin/hadoop job  -Dmapred.job.tracker=local
host.localdomain:8021 -kill job_201509131822_0042
15/09/14 12:14:27 INFO streaming.StreamJob: Tracking URL: http://0.0.0.0:50030/jobdetails.js
p?jobid=job_201509131822_0042
15/09/14 12:14:28 INFO streaming.StreamJob:  map 0%  reduce 0%
15/09/14 12:14:40 INFO streaming.StreamJob:  map 100%  reduce 0%
15/09/14 12:14:47 INFO streaming.StreamJob:  map 100%  reduce 100%
15/09/14 12:14:50 INFO streaming.StreamJob: Job complete: job_201509131822_0042
15/09/14 12:14:50 INFO streaming.StreamJob: Output: /user/cloudera/w261/wk2/hw22/output
assistance      9
```

HW2.2: The word assistance has 9 occurrences if we perform the following steps:

1. Tokenization
2. Remove Special Chars
3. Don't include emails which have a bad format (3 cols as opposed to 4 cols)
4. Include only Email Content

In [25]:
```
%%writefile mapper_hw23.py

#!/usr/bin/env python

import sys
import os
import re
```

```
# Output from mapp
vocab = set()
word_counts = {
    "1": {},
    "0": {}
}
total = 0
total_spam = 0
total_ham = 0

word_list = os.environ['WORDS'].split(",")

def strip_special_chars(word):
    word = word.strip().lower()
    return re.sub('[^A-Za-z0-9]+', '', word)

for line in sys.stdin:
    try:
        # Remove leading & trailing chars
        line = line.strip()
        # Split the line by <TAB> delimeter
        email = re.split(r'\t+', line)

        # Check whether Content is present
        if len(email) < 4:
            continue

        # Get the content as a list of words
        spam = email[1]
        content = email[len(email) - 1].split()

        # Totals
        total += 1
        if spam == '1':
            total_spam += 1
        else:
            total_ham += 1

        for w in content:
```

```
            w = strip_special_chars(w)

            # Add to category dict
            word_counts[spam][w] = word_counts[spam].get(w, 0) + 1

            # Vocab Unique
            vocab.add(w)
    except Exception as e:
        print line
        print e

print 'TOTAL_DOCUMENTS\t%d\t%d\t%d' % (total,total_spam,total_ham)
print 'TOTAL_WORDS\t%d\t%d\t%d' % (len(vocab), sum(word_counts['1'].values()), sum(word_count
s['0'].values()))
for w in word_list:
    print '%s\t%d\t%d' %(w, word_counts['1'].get(w, 0.0), word_counts['0'].get(w, 0.0))
```

```
Overwriting mapper_hw23.py
```

In [26]:
```
!chmod a+x mapper_hw23.py
```

In [29]:
```
%%writefile reducer_hw23.py

#!/usr/bin/env python

import sys
import os
import re
import math

# Totals from Mapper
total = 0
total_spam = 0
total_ham = 0

vocab = 0
vocab_spam = 0
vocab_ham = 0
word_count = {}
```

```
word_list = os.environ['WORDS'].split(",")

for line in sys.stdin:
    try:
         # Remove leading & trailing chars
        line = line.strip()
        # Split the line by <TAB> delimeter
        tokens = re.split(r'\t+', line)

        if tokens[0] == 'TOTAL_DOCUMENTS':
            total += int(tokens[1])
            total_spam += int(tokens[2])
            total_ham += int(tokens[3])
        elif tokens[0] == 'TOTAL_WORDS':
            vocab = int(tokens[1])
            vocab_spam = int(tokens[2])
            vocab_ham = int(tokens[3])
        else:
            word_count[tokens[0]] = (int(tokens[1]), int(tokens[2]))
    except Exception as e:
        sys.exit(1)

prior_spam = (total_spam * 1.0) / total
prior_ham = (total_ham * 1.0) / total

spam_lhood_denom = vocab_spam + vocab
ham_lhood_denom = vocab_ham + vocab
spam_lhood_log = 0.0
ham_lhood_log = 0.0
for w in word_list:
    spam_lhood_log += math.log( (word_count[w][0] + 1.0) * 1.0 / spam_lhood_denom )
    ham_lhood_log += math.log( (word_count[w][1] + 1.0) * 1.0 / ham_lhood_denom )
spam_score = spam_lhood_log + math.log(prior_spam)
ham_score = ham_lhood_log + math.log(prior_ham)

classification = 'HAM'
if spam_score > ham_score:
    classification = 'SPAM'

print '#<Feature>\t<Spam_Score>\t<Ham_Score>\t<Predicted_Class>'
```

```
print '%s\t%f\t%f\t%s' %(",".join(word_list), spam_score, ham_score, classification)
```

```
Overwriting reducer_hw23.py
```

In [30]:
```
!chmod a+x reducer_hw23.py
```

In [32]:
```
'''
HW2.3. Using the Enron data from HW1 and Hadoop MapReduce, write  a mapper/reducer pair that
   will classify the email messages by a single, user-specified word.
   Examine the word "assistance" and report your results.

RESULT: The document is classified as a SPAM.
'''

# Delete existing Output Dirs if available
!hadoop fs -rm -r -skipTrash /user/cloudera/w261/wk2/hw23/output

!ls -l *hw23.py

# Run the Hadoop Streaming Command
!hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh
4.7.0.jar \
-D mapred.reduce.tasks = 1 \
-input /user/cloudera/w261/wk2/hw23/input/enronemail_1h.txt \
-output /user/cloudera/w261/wk2/hw23/output \
-file ./mapper_hw23.py \
-mapper 'python mapper_hw23.py' \
-file ./reducer_hw23.py \
-reducer 'python reducer_hw23.py' \
-cmdenv WORDS='assistance' \

# Show Output
!hadoop fs -cat /user/cloudera/w261/wk2/hw23/output/part-00000
```

```
Deleted /user/cloudera/w261/wk2/hw23/output
-rwxrwxr-x 1 cloudera cloudera 1464 Sep 14 13:24 mapper_hw23.py
-rwxrwxr-x 1 cloudera cloudera 1582 Sep 14 13:26 reducer_hw23.py
packageJobJar: [./mapper_hw23.py, ./reducer_hw23.py, /tmp/hadoop-cloudera/hadoop-unjar267839
2748320022312/] [] /tmp/streamjob7567663044371259044.jar tmpDir=null
15/09/14 13:27:09 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments.

Applications should implement Tool for the same.
15/09/14 13:27:09 INFO mapred.FileInputFormat: Total input paths to process : 1
15/09/14 13:27:10 INFO streaming.StreamJob: getLocalDirs(): [/tmp/hadoop-cloudera/mapred/loc
al]
15/09/14 13:27:10 INFO streaming.StreamJob: Running job: job_201509131822_0054
15/09/14 13:27:10 INFO streaming.StreamJob: To kill this job, run:
15/09/14 13:27:10 INFO streaming.StreamJob: UNDEF/bin/hadoop job  -Dmapred.job.tracker=local
host.localdomain:8021 -kill job_201509131822_0054
15/09/14 13:27:10 INFO streaming.StreamJob: Tracking URL: http://0.0.0.0:50030/jobdetails.js
p?jobid=job_201509131822_0054
15/09/14 13:27:11 INFO streaming.StreamJob:  map 0%  reduce 0%
15/09/14 13:27:23 INFO streaming.StreamJob:  map 100%  reduce 0%
15/09/14 13:27:28 INFO streaming.StreamJob:  map 100%  reduce 100%
15/09/14 13:27:31 INFO streaming.StreamJob: Job complete: job_201509131822_0054
15/09/14 13:27:31 INFO streaming.StreamJob: Output: /user/cloudera/w261/wk2/hw23/output
#<Feature>      <Spam_Score>    <Ham_Score>     <Predicted_Class>
assistance      -8.631765       -10.055939      SPAM
```

HW2.3: Document is classified as a SPAM

In [ ]:
```
'''
HW2.4. Using the Enron data from HW1 and in the Hadoop MapReduce framework, write  a mapper/reducer pair that
   will classify the email messages using multinomial Naive Bayes Classifier using a list of one or more
   user-specified words.
   (SAME MAPPER AND REDUCER AS IN HW2.3 IS USED, BUT WITH DIFFERENT PARAMETERS PASSED IN -cmdenv)

RESULT: The document is classified as a SPAM.
'''


# Delete existing Output Dirs if available
!hadoop fs -rm -r -skipTrash /user/cloudera/w261/wk2/hw24/output

!ls -l *hw23.py

# Run the Hadoop Streaming Command
!hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop-streaming-2.0.0-mr1-cdh4.7.0.jar \
-D mapred.reduce.tasks = 1 \
-input /user/cloudera/w261/wk2/hw24/input/enronemail_1h.txt \
-output /user/cloudera/w261/wk2/hw24/output \
-file ./mapper_hw23.py \
-mapper 'python mapper_hw23.py' \
-file ./reducer_hw23.py \
-reducer 'python reducer_hw23.py' \
-cmdenv WORDS='assistance,valium,enlargementWithATypo' \

# Show Output
!hadoop fs -cat /user/cloudera/w261/wk2/hw24/output/part-00000
```

HW2.4: Document is classified as a SPAM

In [ ]:
```

```