

Running Map reduce in python

- 1) Python3 comes preinstalled in newer version of Ubuntu. However, if you want to check which version is installed on your machine:

- For python3 version check : **python3 --version**

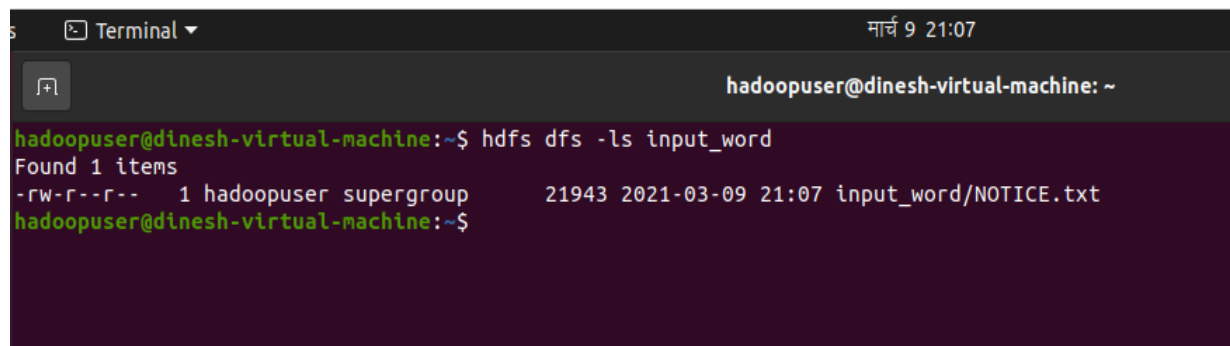
OR

- For python2 version check: **python2 --version**

- 2) If the python is not installed then you can install the python with the following code

```
sudo apt install python3
```

- 3) Next we have to create a folder and insert any local file in the HDFS.

A terminal window titled 'Terminal' with a timestamp 'मार्च 9 21:07'. The prompt is 'hadoopuser@dinesh-virtual-machine: ~'. The user enters the command 'hdfs dfs -ls input_word'. The output shows 'Found 1 items' followed by a table of file details: '-rw-r--r-- 1 hadoopuser supergroup 21943 2021-03-09 21:07 input_word/NOTICE.txt'. The prompt returns to 'hadoopuser@dinesh-virtual-machine:~\$'.

- 4) Now we have to create a mapper and reducer. To create a mapper class first we have to create a empty file and add the code related.
- 5) Type “ **touch mapper.py**” and press enter. Similarly for reducer “**touch reducer.py**” and press enter. You will have two file located in your **/home/hadoop/** directory.
- 6) Open the mapper.py by typing “**nano mapper.py**” and add the flowing code

```
#!/usr/bin/env python3
```

```
"""mapper.py"""
```

```
import sys
```

```
# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # split the line into words
    words = line.split()

    # increase counters
    for word in words:
        # write the results to STDOUT (standard output);
        # what we output here will be the input for the
        # Reduce step, i.e. the input for reducer.py
        #
        # tab-delimited; the trivial word count is 1
        print('%s\t%s' % (word, 1))
```

NOTE: I have python3 in my machine so I have changed that accordingly. In python 3 we have to add parenthesis in print statement and remove in python version 2

Example:

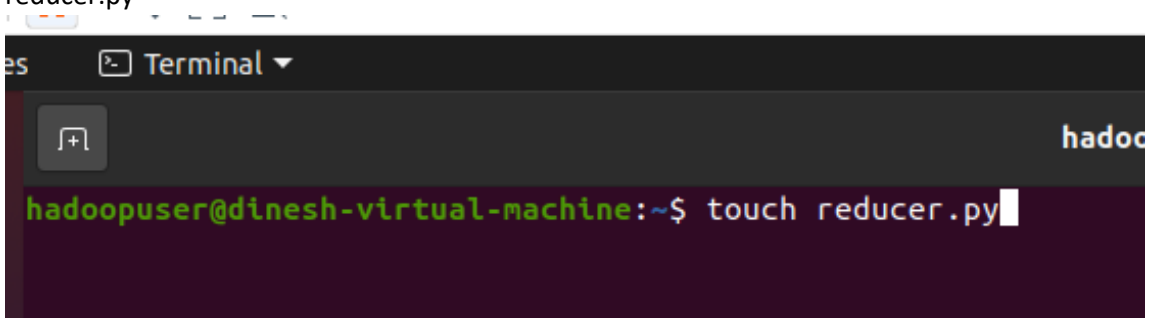
- For python version 2 = `print '%s\t%s' % (word, 1)`
- For python version 3 = `print('%s\t%s' % (word, 1))`

```
#!/usr/bin/env python3
"""mapper.py"""

import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split()
    # increase counters
    for word in words:
        # write the results to STDOUT (standard output);
        # what we output here will be the input for the
        # Reduce step, i.e. the input for reducer.py
        #
        # tab-delimited; the trivial word count is 1
        print('%s\t%s' % (word, 1))
```

- 7) Now open the reducer file by typing “reducer.py” and press enter. add the following code in reducer.py



```
hadoopuser@dinesh-virtual-machine:~$ touch reducer.py
```

```
#!/usr/bin/env python3

"""reducer.py"""

from operator import itemgetter
```

```
import sys

current_word = None
current_count = 0
word = None

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    word, count = line.split('\t', 1)

    # convert count (currently a string) to int
    try:
        count = int(count)
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        continue

    # this IF-switch only works because Hadoop sorts map output
    # by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
            # write result to STDOUT
```

```

        print('%s\t%s' % (current_word, current_count))

current_count = count

current_word = word

# do not forget to output the last word if needed!
if current_word == word:
    print('%s\t%s' % (current_word, current_count))

```

```

# remove leading and trailing whitespace
line = line.strip()

# parse the input we got from mapper.py
word, count = line.split('\t', 1)

# convert count (currently a string) to int
try:
    count = int(count)
except ValueError:
    # count was not a number, so silently
    # ignore/discard this line
    continue

# this IF-switch only works because Hadoop sorts map output
# by key (here: word) before it is passed to the reducer
if current_word == word:
    current_count += count
else:
    if current_word:
        # write result to STDOUT
        print('%s\t%s' % (current_word, current_count))
    current_count = count
    current_word = word

# do not forget to output the last word if needed!
if current_word == word:
    print('%s\t%s' % (current_word, current_count))

```

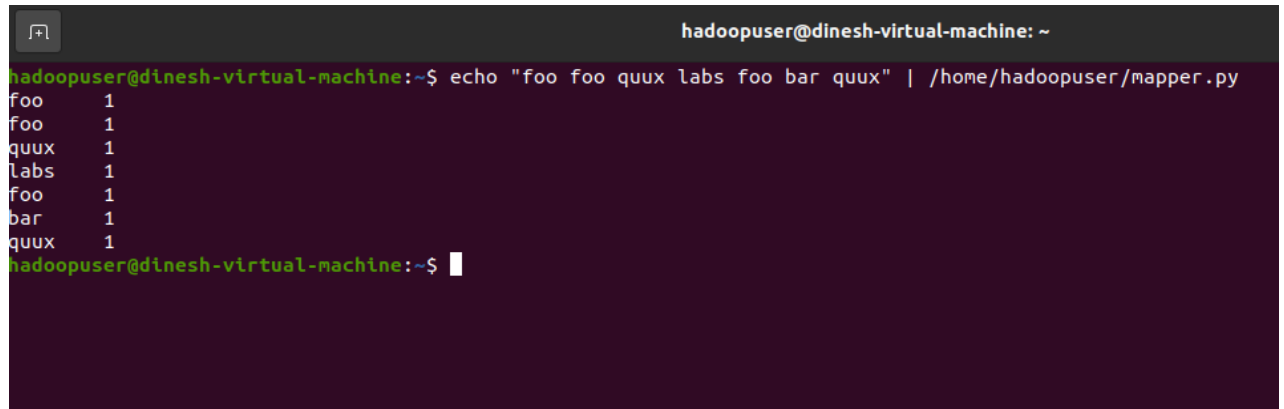
8) Now we have to give permission for the mapper and reducer. Type the following stepwise step.

- `chmod +x /home/hduser/mapper.py`
- `chmod +x /home/hduser/reducer.py`

Testing the mapper.py and reducer.py

9) To test the mapper type the following code and enter.

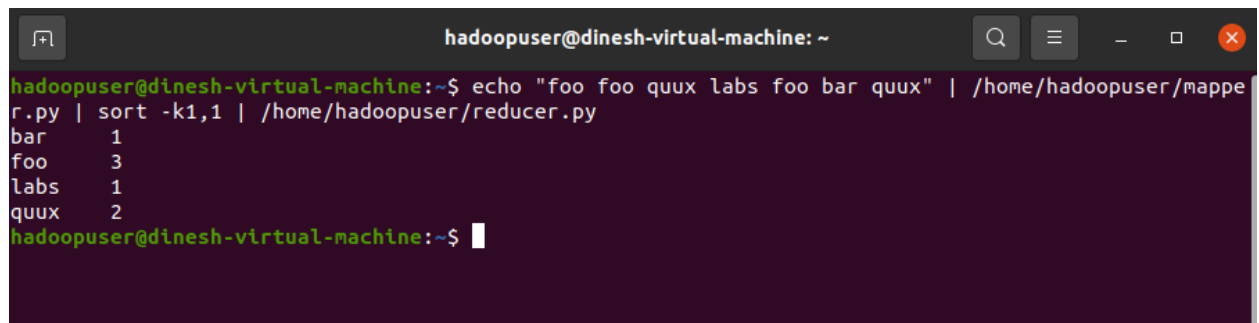
```
echo "foo foo quux labs foo bar quux" | /home/hadoopuser/mapper.py
```



```
hadoopuser@dinesh-virtual-machine: ~  
hadoopuser@dinesh-virtual-machine:~$ echo "foo foo quux labs foo bar quux" | /home/hadoopuser/mapper.py  
foo      1  
foo      1  
quux     1  
labs     1  
foo      1  
bar      1  
quux     1  
hadoopuser@dinesh-virtual-machine:~$
```

10) Now test the mapper and reducer code together

```
echo "foo foo quux labs foo bar quux" | /home/hadoopuser/mapper.py | sort -k1,1 |  
/home/hadoopuser/reducer.py
```



```
hadoopuser@dinesh-virtual-machine: ~  
hadoopuser@dinesh-virtual-machine:~$ echo "foo foo quux labs foo bar quux" | /home/hadoopuser/mapper.py | sort -k1,1 | /home/hadoopuser/reducer.py  
bar      1  
foo      3  
labs     1  
quux     2  
hadoopuser@dinesh-virtual-machine:~$
```

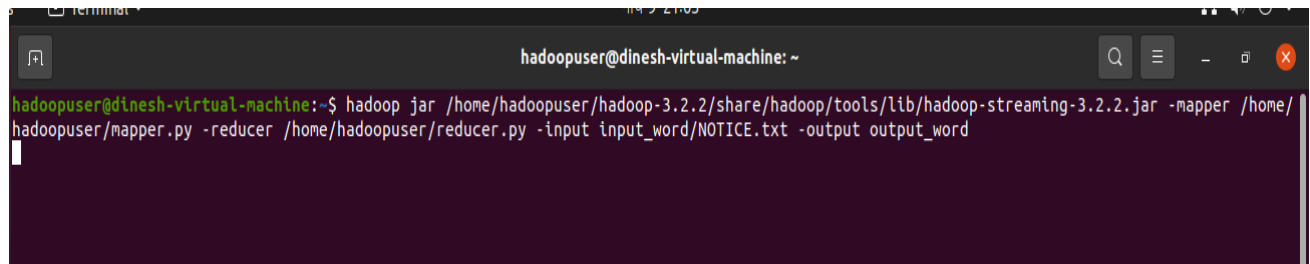
Running the Mapper.py and reducer.py in HDFS

11) Now to execute the code run the following code.

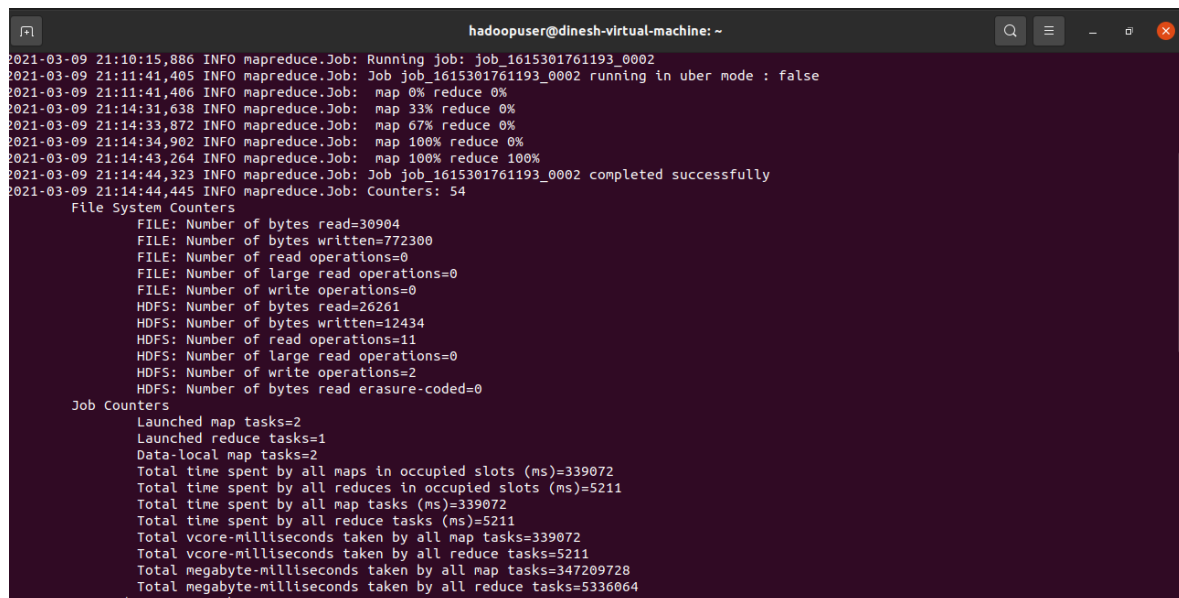
```
hadoop jar /home/hdoop/Hadoop-3.2.2/share/hadoop/tools/lib/hadoop-streaming-3.2.2.jar  
-mapper /home/hdoop/mapper.py -reducer /home/hdooppuser/reducer.py -input  
input_word/NOTICE.txt -output output_word
```

12) if you have output folder then you have to delete that

Note: It is just the syntax. Please enter proper credentials as your system has been configured.



```
hadoopuser@dinesh-virtual-machine: ~  
hadoopuser@dinesh-virtual-machine:~$ hadoop jar /home/hadoopuser/hadoop-3.2.2/share/hadoop/tools/lib/hadoop-streaming-3.2.2.jar -mapper /home/  
hadoopuser/mapper.py -reducer /home/hadoopuser/reducer.py -input input_word/NOTICE.txt -output output_word
```



```
hadoopuser@dinesh-virtual-machine: ~  
2021-03-09 21:10:15,886 INFO mapreduce.Job: Running job: job_1615301761193_0002  
2021-03-09 21:11:41,405 INFO mapreduce.Job: Job job_1615301761193_0002 running in uber mode : false  
2021-03-09 21:11:41,406 INFO mapreduce.Job: map 0% reduce 0%  
2021-03-09 21:14:31,638 INFO mapreduce.Job: map 33% reduce 0%  
2021-03-09 21:14:33,872 INFO mapreduce.Job: map 67% reduce 0%  
2021-03-09 21:14:34,902 INFO mapreduce.Job: map 100% reduce 0%  
2021-03-09 21:14:43,264 INFO mapreduce.Job: map 100% reduce 100%  
2021-03-09 21:14:44,323 INFO mapreduce.Job: Job job_1615301761193_0002 completed successfully  
2021-03-09 21:14:44,445 INFO mapreduce.Job: Counters: 54  
File System Counters  
FILE: Number of bytes read=30904  
FILE: Number of bytes written=772300  
FILE: Number of read operations=0  
FILE: Number of large read operations=0  
FILE: Number of write operations=0  
HDFS: Number of bytes read=26261  
HDFS: Number of bytes written=12434  
HDFS: Number of read operations=11  
HDFS: Number of large read operations=0  
HDFS: Number of write operations=2  
HDFS: Number of bytes read erasure-coded=0  
Job Counters  
Launched map tasks=2  
Launched reduce tasks=1  
Data-local map tasks=2  
Total time spent by all maps in occupied slots (ms)=339072  
Total time spent by all reduces in occupied slots (ms)=5211  
Total time spent by all map tasks (ms)=339072  
Total time spent by all reduce tasks (ms)=5211  
Total vcore-milliseconds taken by all map tasks=339072  
Total vcore-milliseconds taken by all reduce tasks=5211  
Total megabyte-milliseconds taken by all map tasks=347209728  
Total megabyte-milliseconds taken by all reduce tasks=5336064
```

13) Now you will see output folder and the output file. Output word in my case as I have given that specific name while executing map reduce code.

```
hadoopuser@dinesh-virtual-machine: ~  
hadoopuser@dinesh-virtual-machine:~$ hdfs dfs -ls  
Found 2 items  
drwxr-xr-x - hadoopuser supergroup          0 2021-03-09 21:07 input_word  
drwxr-xr-x - hadoopuser supergroup          0 2021-03-09 21:14 output_word  
hadoopuser@dinesh-virtual-machine:~$ hdfs dfs -ls output_word  
Found 2 items  
-rw-r--r-- 1 hadoopuser supergroup          0 2021-03-09 21:14 output_word/_SUCCESS  
-rw-r--r-- 1 hadoopuser supergroup    12434 2021-03-09 21:14 output_word/part-00000  
hadoopuser@dinesh-virtual-machine:~$
```

14) Now you can test if the map reduce program in the python worked correctly or not by following command.

```
hdfs dfs -cat output_word/part-00000
```

```
hadoopuser@dinesh-virtual-machine: ~  
hadoopuser@dinesh-virtual-machine:~$ hdfs dfs -cat output_word/part-00000  
'AS' 1  
'GCC' 1  
'Java' 1  
'License'); 1  
# 6  
& 1  
'Aalto' 1  
'Apache' 4  
'ArrayDeque', 1  
'Bouncy' 1  
'Calliper', 1  
'Compress-LZF', 1  
'HPACK', 1  
'Hadoop', 1  
'JBoss' 1  
'JCTools', 1  
'JDOM' 1  
'JZlib', 1  
'Protocol' 1  
'SLF4J', 1  
'Snappy', 1  
'Webbit', 1  
'hbase-common/src/main/java/org/apache/hadoop/hbase/io/LimitInputStream.java' 1  
'jzlib2', 1  
'jfastlz', 1  
'libdivsufsort', 1  
'license' 1  
'lz4', 1  
'lzma-java', 1  
'ASL2' 1  
'Apache' 12  
'BSD' 2
```

!!WELL DONE!!

15) References

- [Writing An Hadoop MapReduce Program In Python \(michael-noll.com\)](http://michael-noll.com)