## Owen Galvin

## HU Extension School E-63 Big Data Analytics

## Assignment 02

### Handed out: 02/06/2016 Due by 11:30 PM on Friday, 02/12/2016

Capture all steps of your implementation with comments indicating what are accomplishing with every step. Place those in this MS Word document bellow the problem statement. Please send comments and questions to the Discussion Forum on the class site.

### 

**Problem 1)** Please, download and install VMware Workstation 11 on your 64 bit Windows PC or VMWare Fusion 7, if you are on a MAC. Please download 64 bit CentOS6.7 and create a 64 bit VM. If you know what you are doing and you work with another flavor of Linux supported by CDH5.5.1, please be free to create a virtual machine based on your favorite Linux flavor. Provide your virtual machine with some 40GB of disk space, if you can spare it. For whatever reasons, Hadoop installation appears to prefer to have more than 20 GB of available space. Name the main user of your VM cloudera. Do not use name hadoop. “hadoop” is a bad name for a user, since Hadoop framework has an executable called hadoop and it creates many directories with that same name and those would not necessarily be owned by the VM user called hadoop. On that VM create yet another user called joe. Make both users sudo users. Once your CentOS is fully installed, please shut the VM down and make a copy of the entire directory containing that VM. Name the folder containing that copy differently. Two VMs are identical and you could even run them simultaneously if your machine has enough memory. In the folder of each VM add a text file describing OS on your VM, usernames and passwords of important users. This little file will make your VMs useful long into the future. The reason you are creating the backup VM is to save time, if you damage the one VM on which you are installing your software.

NEW: Please do not capture installation of Workstation 12 or Fusion. Please do not capture every step in creation of VM. Show addition of the second network adapter. Show steps in creation of user joe. Demonstrate that joe is a sudo user. Show results of your ifconfig command.

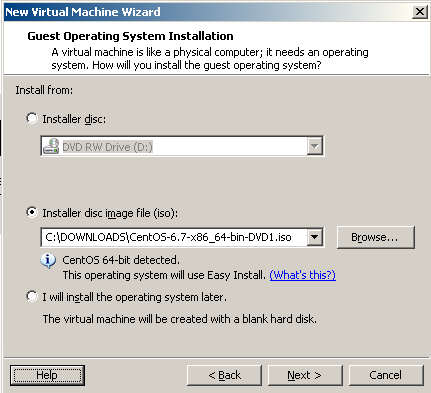
**Solution:**

1. **VMWare installation related**

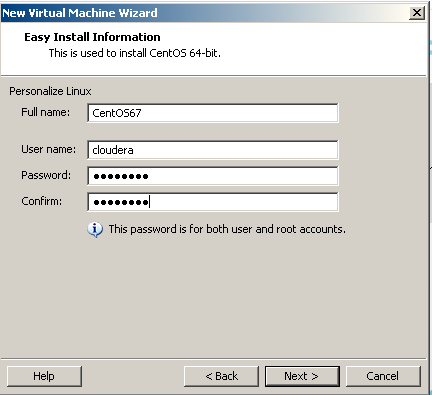
**Download VMWare Workstation 12 for windows since I am on Windows 7, file =VMware-workstation-full-12.1.0-3272444.exe & install.**

1. **Linux related**

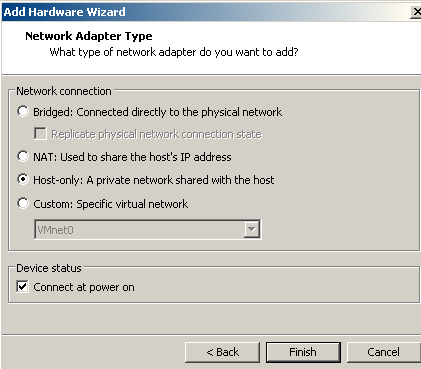
**Browse to** [**http://isoredirect.centos.org/centos/6.7/isos/x86\_64/**](http://isoredirect.centos.org/centos/6.7/isos/x86_64/) **and select mirror =** [**http://mirrors.mit.edu/centos/6.7/isos/x86\_64/**](http://mirrors.mit.edu/centos/6.7/isos/x86_64/) **since I live a few blocks away. Who knows where the mirror really is but shouldn’t be too far away. Download CentOS-6.7-x86\_64-bin-DVD1.iso and also CentOS-6.7-x86\_64-bin-DVD2.iso. As indicated in the professor’s email I’ll sprinkle a few screenshots during the VM creation but not every dialog. Begin the VM Creation by clicking on Create a New Virtual Machine. Select Typical, and then on the next screen browse to the .iso I downloaded.**



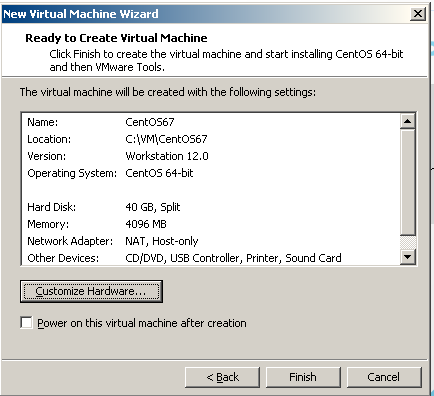
**Click Next and create cloudera user**



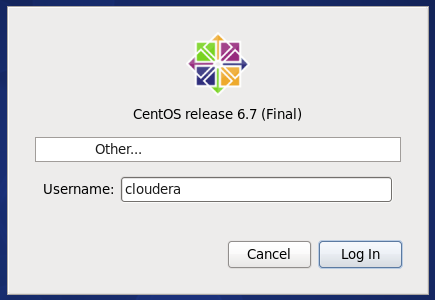
**On the next screen, provide machine name = CentOS67 and save into the local directory I use for VMs. After that set the size to 40Gb and choose to split the virtual disk into multiple files. On the final screen, click Customize Hardware button. From there set the RAM to 4GB and then create a new Host-only Network Adapter.**



**After clicking Close on the Customize Hardware screen, the Final config screen shows the result of both of the last steps.**



**Once the VM creation has finished, log on as cloudera per the professor’s email.**



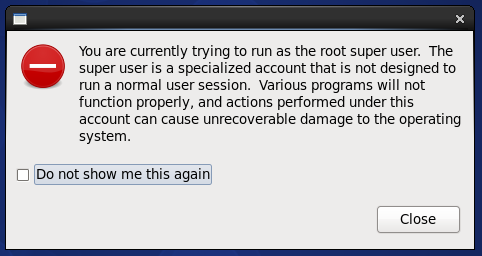
**From there open a terminal window and changed user to root and then ran the ‘yum update’ command, and followed through prompts until finished.**

|  |
| --- |
|  |

**The next step is to create a new user named ‘joe’. I would normally do this through Administration -> User and Groups but the pdf shows it in the shell environment. No indication that ‘joe’ needs to be in a group as yet though, so I’ll simply create him using ‘useradd’ and set a password to unlock the account. This is a continuation of previous Terminal session, so everything is being executed as user = root. Tried password = ‘joe’, which it really didn’t like, so I switched to ‘cloudera’, which it also didn’t like but worked.**

|  |
| --- |
|  |

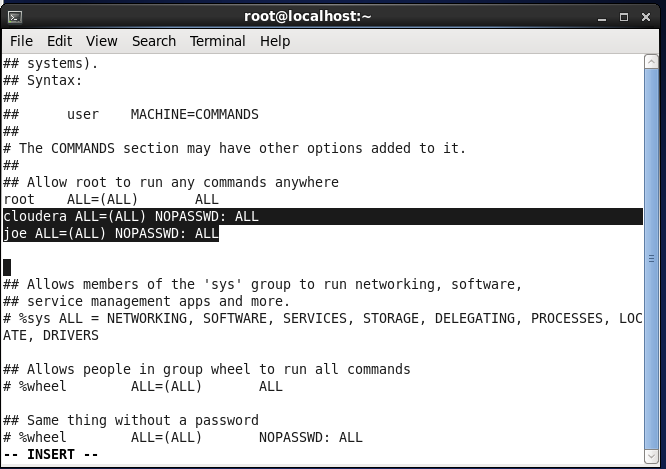
**Log out as user cloudera and login as user = root as indicated in the pdf. A warning message is displayed but in context it doesn’t seem unusual, so click Close and open a Terminal window.**



**Within the terminal window enter below commands, chmod on file = /etc/sudoers to change permissions and allow editing. Followed by calling visudo in order allow for the same file to be edited.**

|  |
| --- |
|  |

**Navigate to the section in /etc/sudoers that contains syntax similar to what we are supposed to enter. Add those two lines (highlighted in black on the screenshot)**



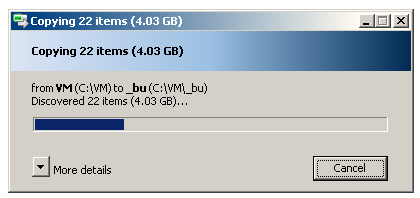
**Hit the escape to exit INSERT mode, followed by ":x" to save the file and exit. Demonstrate that joe is a sudo user by trying to run visudo standalone, which is denied, and then via sudo. That latter command led to /etc/sudoers file opening in visudo (not displayed), no permission error. Then I run the –v to validate and –l which apparently should list what joe can run as sudo. Hopefully that suffices, kind of reliant on googling an appropriate demonstration.**

|  |
| --- |
|  |

**Run the ifconfig file and show output.**

|  |
| --- |
|  |

**Shut down the VM and copy the entire folder in order to make a backup.**



**In case it should matter, here is the final copy along with a ReadMe\_CentOS67.txt that contains details on the two accounts and OS details.**

|  |
| --- |
|  |

**---- END OF PROBLEM 1 ----**

**Problem 2)** Use one of yourVMs and follow closely steps in the CDH5.5.1 Quick Start Guide, or my notes. PDF and PPT formats and characters on PC do not always map well into Unix (Linux) characters. If you want to copy commands from the guide you are better off doing it from the HTML version of the CDH Quick Start Guide, which you could find at:

[http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm\_qs\_quick\_start.html](http://www.cloudera.com/content/cloudera/en/documentation/core/latest/topics/cm_qs_quick_start.html%20) and open from with your VM.

Both my notes and the Quick Start Guide will lead you through a “semi-automated” process of installing Hadoop. Please install YARN version of Hadoop. My notes add a few explanations beyond what you can see in the Cloudera’s guide. Read the notes and the guide very carefully. Do not execute commands for flavors of Linux other than RedHat (CentOS) unless you are working with another flavor purposefully. You will know that you have successfully installed Hadoop if all of tests described in the guide work properly.

**Solution: Following steps in pdf from class, first download Java by navigating to Oracle download site and selecting Java SE Development Kit 8u60 link. From subsequent page download the .rpm highlighted below = dk-8u60-linux-x64.rpm, after creating Oracle account etc.**

|  |
| --- |
|  |

**Navigate to Downloads directory and install the Java package. Though cloudera account doesn’t have necessary privileges, so become root by entering “su” and root’s password. Then use the rpm command along with –ivh argument to install Java.**

|  |  |
| --- | --- |
|  |  |

**Next, confirm I continue to running as root via the whoami command. Navigate to root’s home directory and open .bash\_profile using the vi editor.**

|  |
| --- |
|  |

**Original contents of file are on left below. Replace the section under ‘# User specific…’ with full contents from pdf, below on the right.**

|  |  |
| --- | --- |
|  | **…** |

**After finishing edit, hit Escape key, followed by ‘:wq’ in order to save edits. Before using source command, check the $PATH variable and try to cd into $JAVA\_HOME, confirming no changes are in effect yet. Then use source on the .bash\_profile file and check those two variables and see that indeed they are working.**

|  |
| --- |
|  |

**Exit as user root and repeat same steps as above for cloudera, once I’ve used the whoami command to confirm my identity is back to cloudera. I’ll do a couple of quick overview screenshots only since the process is essentially the same as just above. Below shows that the .bash\_profile file displays with color syntax highlighting (perhaps because we are the logged in user?)**

|  |
| --- |
|  |

**Overview of command window steps for cloudera user are below.**

|  |
| --- |
|  |

**Now begin following the steps in pdf under ‘Manual Installation of CDH 5.5.x with Yarn (Mrv2)’. First browse to URL from pdf and download cloudera-cdh-5-0.x86\_64.rpm.**

|  |
| --- |
|  |

**Command prompt shows that I am in Downloads directory for cloudera, so I should be able to install straightaway, use yum to do so and answer yes to prompt toward end of process. Below only shows beginning and end of the installation.**

|  |
| --- |
| **…..** |

**Also run the command mentioned in pdf to add Cloudera Public key, though I won’t bother with a real screenshot since that is an optional step.**

**$ *sudo rpm --import http://archive.cloudera.com/cdh5/redhat/6/x86\_64/cdh/RPM-GPG-KEY-cloudera***

**Next install Hadoop with YARN in pseudo-distributed mode by running the command “sudo yum install hadoop-conf-pseudo”. Again I’ll just screenshot the beginning and end of the install process**

|  |
| --- |
| **….** |

**Confirm the YARN configuration files are present via rpm command with ‘-ql’ argument**

|  |
| --- |
|  |

**Format the HDFS file system by calling the hdfs script as user=hdfs, passing in ‘namenode –format’ argument.**

|  |
| --- |
| **…** |

**I have trouble getting the for loop re the Hadoop hdfs services to work correctly, even though it seems simple and I don’t appear to have a cut & paste issue. Either way just as easy at this point to use ls to list all the services after cd’ng to the correct directory. From there manually start each of the three services as sudo.**

|  |
| --- |
|  |

**Use** [**http://localhost:50070/dfshealth.html#tab-overview**](http://localhost:50070/dfshealth.html#tab-overview) **url to do a quick check on dfs, partial screenshot below**

|  |
| --- |
|  |

**Run init-hdfs.sh script to create standard directories in HDFS & transfer ownership as appropriate. Screen output is extensive, show abbreviated below.**

|  |
| --- |
| **…** |

**Verify HDFS file structure by passing in arguments to hdfs script such that Hadoop will execute file system shell to recursively list files. Head portion shown below looks like that in the pdf, so should be good to go.**

|  |
| --- |
|  |

**Now start the YARN services & the mapreduce history service using service/start on each.**

|  |
| --- |
|  |

**User ‘joe’ was created earlier before CDH was installed, so he wouldn’t be a member of mapred group + the pdf is all done in context of user = ‘chuck’, so I’ll create chuck as a new user and continue.**

|  |
| --- |
|  |

**Now that the new user exists, have hadoop create a user directory for him, change ownership on the directory to him. Follow that by doing same for user cloudera.**

|  |
| --- |
|  |

**Login as user chuck, make a directory named input and copy some XML files into that folder. List contents of directory to confirm transfer.**

|  |
| --- |
|  |

**Use the export command to set the HADOOP\_MAPRED\_HOME variable and confirm that worked.**

|  |
| --- |
|  |

**Run the example job from the pdf, to process files in combination with grep command. I’m guessing it will look for text starting with ‘dfs’, followed by alpha or dot characters. Long output has been truncated to begin/end.**

|  |
| --- |
| **….** |

**Pass the fs command on to hadoop script with –ls command in order to see contents of output23. Then do similar with -cat in order to read the beginning of the part-r-00000 file and show the results, presumably these are all the words across the .xml files that (more or less) begin with “dfs”.**

|  |
| --- |
|  |

**---- END OF PROBLEM 2 ----**

**Problem 3)** As your new Linux user joe fetch the .txt version of James Joyce's Ulysses by issuing the following command on the command prompt:

wget <http://www.gutenberg.org/files/4300/4300.zip>

Unzip the file. Open the resulting txt file with Vi and convince yourself that the life of Buck Mulligan is in front of you. Create a HDFS directory called ulysses and copy the .txt file into that HDFS directory. Do **not** create another HDFS directory called counted. The Map Reduce job you will run will create that directory for its output. Actually, if the directory preexists the job will raise an error. That same hadoop-mapreduce-examples.jar file mentioned in class notes and you used as the final proof that MapReduce works contains another program called wordcount. wordcount will tell you how many times a word appears in a provided text. Invoke wordcount by the following command:

$ hadoop jar /usr/lib/hadoop-mapreduce/hadoop-mapreduce-examples.jar wordcount ulysses counted

Once the job is finished visit site <http://localhost:19888>. You will see some statistics on MapReduce jobs executed on your cluster. There will not be much for your short job. In general that is a very useful site.

Copy results of word count analysis to the local file system. Write a small program in any language (or scripting tool) of your choice and order the counting results by the decreasing count. Present the portion of your final result which does not contain so called stop words (the, a, and, or, …) in your report. Submit top 200 words in separate .txt file with your report.

**Solution: My user joe was added before CDH was installed and so needs to be added to mapred group. Do a su as root for permissions reasons and issue usermod command to add joe to mapred.**

|  |
| --- |
|  |

**Switch login to joe and download the 4300.zip file.**

|  |
| --- |
|  |

**Unzip the file and confirm 4300.txt was extracted .**

|  |
| --- |
|  |

**Open the .txt file with vi, enough to see contents and confirm we have the expected text, then exit.**

|  |
| --- |
|  |

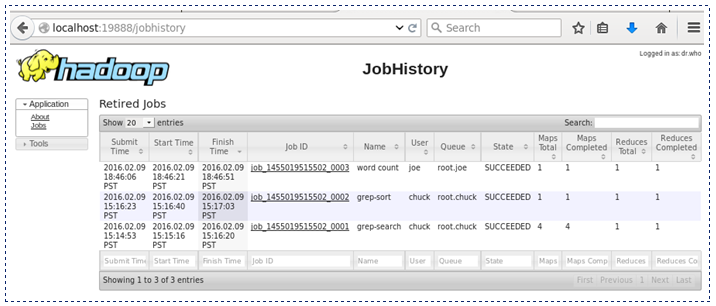
**First need to create the HDFS home directory for joe, using same commands as for chuck in Problem 2. Create the Ulysses directory, copy the 4300.txt into it and confirm the contents of HDFS folder named Ulysses.**

|  |
| --- |
|  |

**Run the wordcount program from hadoop-mapreduce-examples.jar and write the output to a new directory named counted. Truncated command window output is below.**

|  |
| --- |
| **…** |

**Navigate to** [**http://localhost:19888**](http://localhost:19888) **to show some statistics on the just-executed job, as well as earlier grep tests as chuck.**



**Place a copy of the output file (part-r-00000) into joe’s home directory, confirm it made it there**

|  |
| --- |
|  |

**Now moving on to the programming part, which will be done in Python 2.7. First note that I’ve renamed part-r-00000 to something more user friendly = counter\_contents.txt and placed it in a folder named data. With the file ready, the first task was to obtain a list of stop words. Since I have the Anaconda package installed I figured there would be something built-in and indeed there was, though my system did need to download the requested nltk corpus files first. With that in place I wrote a little snippet so that the 127 stop words would be self-contained within the .py file.**

|  |
| --- |
| from nltk.corpus import stopwords  wds = stopwords.words('english')  stop\_words = ''  stop\_words = [stop\_words + str(wd) for wd in wds] |

**I simply output the resulting stop\_words list, pasted into the .py file and assigned it to a variable of the same name. (I also import os module for path manipulation and operator for help in sorting, below is about a good place as any to insert.)**

|  |
| --- |
| import os  import operator  stop\_words = ['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your',... |

**Bulk of the file processing is below. Create a list to hold each word & count pairing found in the file. Assemble the file\_path and then open it and process line by line, where each line is split on the tab character. Since stop\_words are all lower-case, makes sense to lower-case the Ulysses word before checking if it should be rejected – when that happens just go on to next line in the file. The word\_counts will wind up as a list of tuples in format of (wordString, intCount).**

|  |
| --- |
| word\_counts = []  # found\_stop\_words as a sanity check, inspect contents while debugging to make sure  # nothing unusual - looks like 126 of 127 stop words were found in the file  found\_stop\_words = set()  file\_path = os.path.join(os.getcwd(), 'data/counter\_contents.txt')  for line in open(file\_path):  #file is tab separated, 50094 lines with 50094 tabs, split into two  word, count = line.strip().split('\t')  if word.lower() in stop\_words:  found\_stop\_words.add(word)  continue  word\_counts.append((word, int(count))) |

**Now sort the non-stopwords list, using key argument to sort by the 2nd item in each tuple, i.e. the word-count (and to sort descending). List comprehension to create a list of tab-separated word-TAB-count lines, though only grabbing the first 200 items from the sorted list. Finally open the output file = sorted\_word\_counts and write to that, which will be uploaded as part of the assignment.**

|  |
| --- |
| sorted\_word\_counts = sorted(word\_counts, key=operator.itemgetter(1), reverse = True)  output\_word\_counts = ['{0}\t{1}\r\n'.format(word, count) for (word, count) in sorted\_word\_counts[:200]]  with open('sorted\_word\_counts.txt', 'w') as f:  f.writelines(output\_word\_counts) |

**Problem 4).** Consider a symmetric matrix

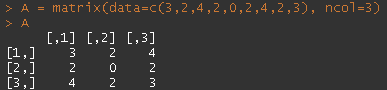
A =

Using R demonstrate that all three eigenvectors of that matrix are mutually orthogonal. Let be the matrix of eigenvectors of matrix A. Calculate product of tree matrices:

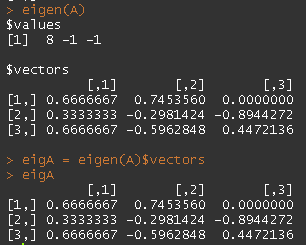
A Λ

Symbol T indicates the transpose matrix. Google around for properties of eigenvectors and eigenvalues of real symmetric matrices. What is the general statement you can make about the observation on the value of the above product.

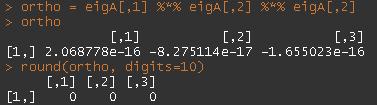
**Solution: Create matrix A that resembles the example. Use matrix function along with ncol=3 so that we wind up with 3x3 matrix.**



**Assuming that is supposed read “Calculate product of three matrices”, need to come up with the eigen vectors matrix and the transpose of said matrix. First examine the eigen values/vectors of matrix A and then create variable named eigA and assign that to the eigen vectors of A, confirm contents of that variable are as expected.**



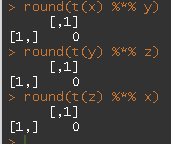
**Demonstrate the mutually orthogonal nature of A’s eigen vectors by multiplying each vector by one another. After rounding, clear that each dot product is zero, i.e. each vector is perpendicular to each other.**



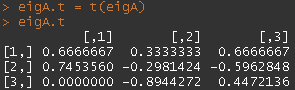
**Maybe the above wasn’t full proof, still googling around here and reading discussion board, try an alternate demonstration. First assign x,y,z to each of A’s eigen vectors**



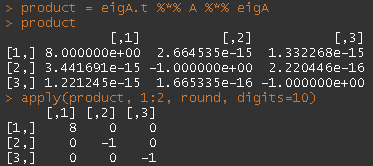
**Then multiply the transpose of each by the subsequent vector. Note that each product = zero.**



**Create the third matrix = the transpose of the eigen vectors of matrix A, by using t() function (aka transpose) on eigA.**



**Create product variable to hold result of multiplying the three matrixes like** A Λ **and display. Most values appear to be very small, so then use apply() function to use the round function on both columns and rows.**



**As to a general observation re the final product… I’ve googled around for some time, watched a couple of videos, but without a background in linear algebra it is rather difficult to understand examples as they use mathematical notations with which I am not familiar. The closest I got is “*Every symmetric matrix is orthogonally diagonalizable*”, where I can see that we started off with a symmetric matrix and that the final product is a diagonal matrix.**