Predictive Analytics
Large Scale Data Analytics Frameworks

Learning outcomes

Introduction to Hadoop, Mahout, MapReduce Paradigm and Targeted Marketing.

Understanding Hadoop, Yarn and MapReduce

Context

As a data scientist, you need have exposure to major infrastructure frameworks for large scale data science. Hadoop is a data science large scale framework that servers for two main data purposes: storage, and computation engine. It has two main components: HDFS and MapReduce.

In this homework, you will be introduced to the fundamentals of Hadoop and writing MapReduce programs. You will have to conduct a preliminary research on the differences between Hadoop 1.0 and Hadoop 2.0 as we discussed in class and report it in this homework.

Installing Hadoop

In this assignment, you are required to install Hadoop. For this homework, Hadoop 1.0 suffice but it is recommended to install the latest version of Hadoop. For a very straightforward installation of Hadoop, we will be adopting Cloudera' distribution.

Cloudera's Hadoop Distribution (CDH) is an integrated Apache Hadoop that contains pre-installed and configured components for a Hadoop production environment. CDH is free and is available on different formats Linux packages, virtual machine images and in tar files. You can also run CDH on the cloud. Everything in CDH is installed and ready to go. You can download the image type that corresponds to you preferred virtualization platform.

Pre-requisites for this Homework

Review the chapter of Large Scale Frameworks covered in class. You may need to read the following papers:

- Dean, Jeffrey, and Sanjay Ghemawat. MapReduce: simplified data processing on large clusters. Communications of the ACM 51.1 (2008): 107-113.
- Ghemawat, Sanjay, Howard Gobioff, and Shun-Tak Leung. The Google file system. ACM SIGOPS operating systems review. Vol. 37. No. 5. ACM, 2003.

Understanding the three modes of Running Hadoop

Generally, Hadoop can run in three modes:

1. **Fully-distributed mode.** Normal mode with namenodes and datanodes.
2. **Pseudo-distributed mode.** In this mode there still can be several mappers and reducers but all daemons run one machine (your local machine) using HDFS protocol. This mode is very useful for developers to mimic a Hadoop environment for testing purposes.
3. **Hadoop in Standalone mode.** All jobs will run as one mapper and one reducer in your local file system (not HDFS). This is very practical model for developers to write and debug map reduce applications.

Installing Hadoop

You can install Hadoop using the Cloudera Manager or the Hadoop plain distribution. For using Cloudera's distribution please follow the instructions shown below:

0. Preparation

0.0 Download and install Oracle *Virtual Box* from the following website

- https://www.virtualbox.org/wiki/Downloads
- Follow the instructions to install

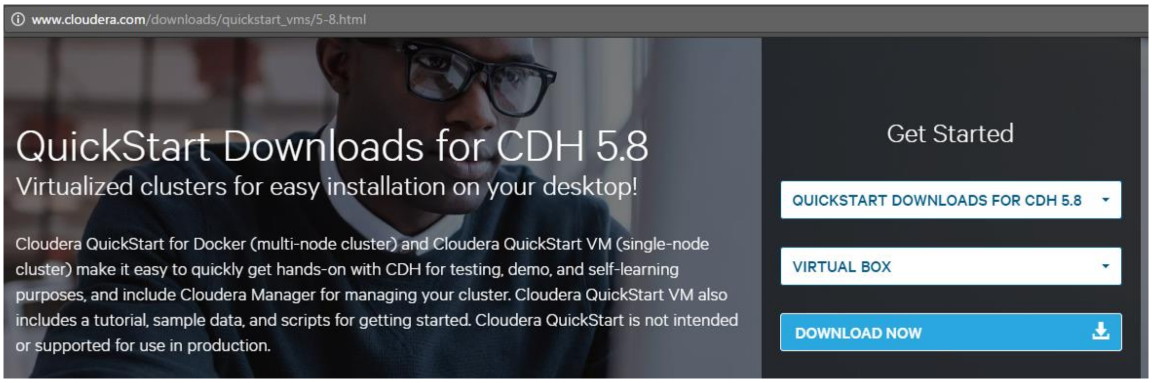
0.1 Download and install Cloudera *QuickStart* VM

- Go to the Cloudera QuickStart download page for the most updated Virtual Box virtual machine image

(As this instruction is prepared, the most recent release is CDH 5.8)

http://www.cloudera.com/downloads/quickstart_vms/5-8.html

- In this page, choose from the second dorp down box "VIRTUAL BOX"


www.cloudera.com/downloads/quickstart_vms/5-8.html

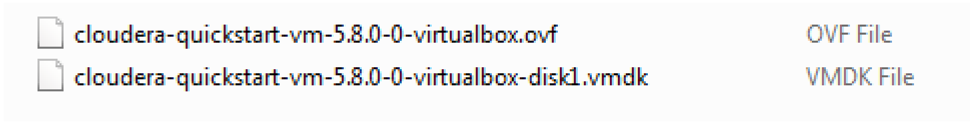
QuickStart Downloads for CDH 5.8

Virtualized clusters for easy installation on your desktop!

Cloudera QuickStart for Docker (multi-node cluster) and Cloudera QuickStart VM (single-node cluster) make it easy to quickly get hands-on with CDH for testing, demo, and self-learning purposes, and include Cloudera Manager for managing your cluster. Cloudera QuickStart VM also includes a tutorial, sample data, and scripts for getting started. Cloudera QuickStart is not intended or supported for use in production.

- Then click download

- After download, unzip the file, then you will see



cloudera-quickstart-vm-5.8.0-0-virtualbox.ovf	OVF File
cloudera-quickstart-vm-5.8.0-0-virtualbox-disk1.vmdk	VMDK File

0.2 Load your VM into *Virtual Box*

- Run Virtual Box, then from menu bar:

"Machine" -> "Import Appliance ..." -> choose the .ovf file shown above -> "Next" -> "Import"

- Then wait for the VM to load
- It will take a while to load the VM image at the first time.

0.3 After the VM loaded, double click it on the left side panel to start a Linux OS which is a Hadoop Environment pre-configured

- In the VM, you can find an Eclipse in which a stub project has been set up with all MapReduce dependencies for you.

0. Word Count in Eclipse under Hadoop (20pts)

For more details on how to set up Eclipse' java projects with Hadoop libraries view this video.

Using Eclipse and WordCount.java from the homework folder, create a project on Eclipse on **pseudo-distributed** mode for Hadoop under your Virtual Machine. You must use Eclipse.

1) Run WordCount on the attached file **document.txt** from the homework folder.

Attach at least the following snapshots:

1. Snapshot of the WordCount.java in Eclipse project
2. Snapshot of the mappers and reducers on http://localhost:8088
3. Snapshot of the full output of the WordCount (count of words) on the file and attach the output file to your folder submission under ***WordCountOUTPUT2.txt***

2) Repeat (1) or Article1.txt, Aricle2.txt and Article3.txt

Explain all your steps (use snapshots or list all your steps)

1. Targeted Advertising using MapReduce (60pts)

To increase the sales volume on the next season, you company that has similar business like Amazon is planning to target the most valuable customers and make the marketing plan accordingly within the budget limit. One of the possible strategy is to find out, for all combinations of categories of products, the customers with the largest purchases histories.

This will allow the marketers to advertise and promote to the most potential buyers for each combination of categories, rather than targeting everyone.

You will need to follow the instructions below to implement a simple MapReduce program using Hadoop framework to figure out who are the targeted customers.

Given N possible categories, there are *N^2 - N* combinations of categories (we deducted N to not include the combination of categories that have one element).

The purpose is to find the top-k customers who have the highest number of purchase history for each possible combination of categories

Input: <UID> <Categories>

where UID and Categories are separated by a tab ("\t"), and Categories is a string which is a sequence of Category IDs separated by only a comma (","). The input data is attached to the homework link.

For Example:
UID Categories of Goods in This Transaction (this is the log of each customer and their purchase history of product from each category)

```
...    ....
U0  C6,C13,C14
U0  C3,C6
U1  C0,C9
U2  C13,C16
U3  C2
U3  C0,C10,C11,C12,C14
....   .....
```

Expected Output:

Your output should be in this format: for each possible combination of categories, generate a list of UIDs with the number of this combination appeared as a sub set in the UID's historical transactions. For example, if user 13 (UID13), has 2 transactions:

```
    U13 C1,C2
    U13 C0,C1,C2
```

The output will be:

```
    C0   ...,U13(1), ....            // since 1 transactions has (C1,C2) as subset
    C1   ...,U13(2), ....            // since 2 transactions has (C1,C2) as subset
    C2   ...,U13(2), ....            // since 2 transactions has (C1,C2) as subset
    C0,C1      ...,U13(1), ...       // since 1 transactions has (C1,C2) as subset
    C0,C2      ...,U13(1),...        // since 1 transactions has (C1,C2) as subset
    C1,C2      ...,U13(2), ....      // since 2 transactions has (C1,C2) as subset
    C0,C1,C2 ...,U13(1), ...     // since 1 transactions has (C1,C2) as subset
    ....
```

You should use tab ("\t") to separate combination of categories from its corresponding UID list. In the UID list, delimit each UID using a comma (",").

For Example:

```
<Combination of Categories> <List of UIDs (Counts) >
C0	U1395(12),U1395(12),U1395(12),U1395(12),U1395(12)
C0,C1	U5088(6),U5088(6),U5088(6),U5088(6),U5088(6)
C0,C1,C10	U1850(3),U1850(3),U1850(3),U1850(3),U1850(3)
C0,C1,C10,C11	U7869(2),U7869(2),U7869(2),U7869(2),U7869(2)
C0,C1,C10,C11,C12	U907(1),U907(1),U907(1),U907(1),U907(1)
C0,C1,C10,C11,C12,C14	U3253(1),U3253(1)
```

3. Recommender System using Apache Mahout (20pts)

Follow the steps under this link to build a recommender system using Apache Mahout.
Attach snapshots of your program and attached the code.