Hands-on with Apache Mahout

A beginner's guide on how to get started using Apache Mahout

JOSEPH PONTANI

CS4604: SPRING 2014

VIRGINIA TECH, BLACKSBURG VA

MARCH 14, 2014

Contents

Introduction

This tutorial will provide an introductory glance at how to get up and running using the machine learning capabilities of Apache Mahout. It will go over how to use the command line interface to run different algorithms on a data set.

This tutorial will assume the reader has a working knowledge of using a command line to issue commands to a Unix operating system, as well as knowledge of installing software on a Unix system or variant of, and knowledge of modifying environment variables ($PATH and $CLASS_PATH). This tutorial also assumes a working installation of Mahout (and Hadoop, if running on a cluster) and any prerequisite software installs. This tutorial will not cover how to install any required software.

Any images or screenshots pertained in this tutorial are for reference, and may not be exactly the same that you may see on your system. Screenshots of the shell display for Mahout commands are based on Mahout running off of a single-node Hadoop cluster.

Background

Before getting started, it will benefit the reader to understand a bit about Apache Hadoop and Apache Mahout. Both Hadoop and Mahout were developed by the Apache group to help analyze large data sets. Hadoop provides the framework to manage a node cluster that does the actual computing of the algorithms that Mahout designates on a given data set.

Hadoop

Hadoop is a framework that provides a foundation for processing large data sets across a cluster of computers. It can scale from a single computer, to as many computers as the user can provide to it. It distributes the processing requirements across the network to reduce the time it takes to return a result. Hadoop includes four main features: Hadoop Common, Hadoop Distributed File System, Hadoop YARN, and Hadoop MapReduce.

Hadoop Common is the core library of features that other modules utilize in order to distribute and process data.

Hadoop Distributed File System (HDFS) is a distributed file system that guarantees high availability for all nodes to the data set.

Hadoop YARN is a framework that handles job scheduling and manages the resources of the cluster.

Hadoop MapReduce is a YARN-based approach that allows for parallel processing of data.

Mahout

Apache Mahout is a machine-learning and data mining library. It provides three core features for processing large data sets. They are: clustering, classification, and collaborative filtering.

Clustering is the ability to identify related documents to each other based on the content of each document.

Classification is the ability to categorize a document into an existing category based on the content of the document.

Collaborative filtering is the ability to provide filters that are based on the similarities and differences of tastes between users.

Getting Started with Mahout

With a working install of Mahout, we first want to get ourselves familiar with some of the basic options and approaches that we will be using to interact with Mahout.

Creating Sequence Files

Sequence a Directory

To start, any raw text files must be converted to sequence files, and then to vector files, that Mahout will then analyze for any given approach. So to convert to sequence files, you will simply run:

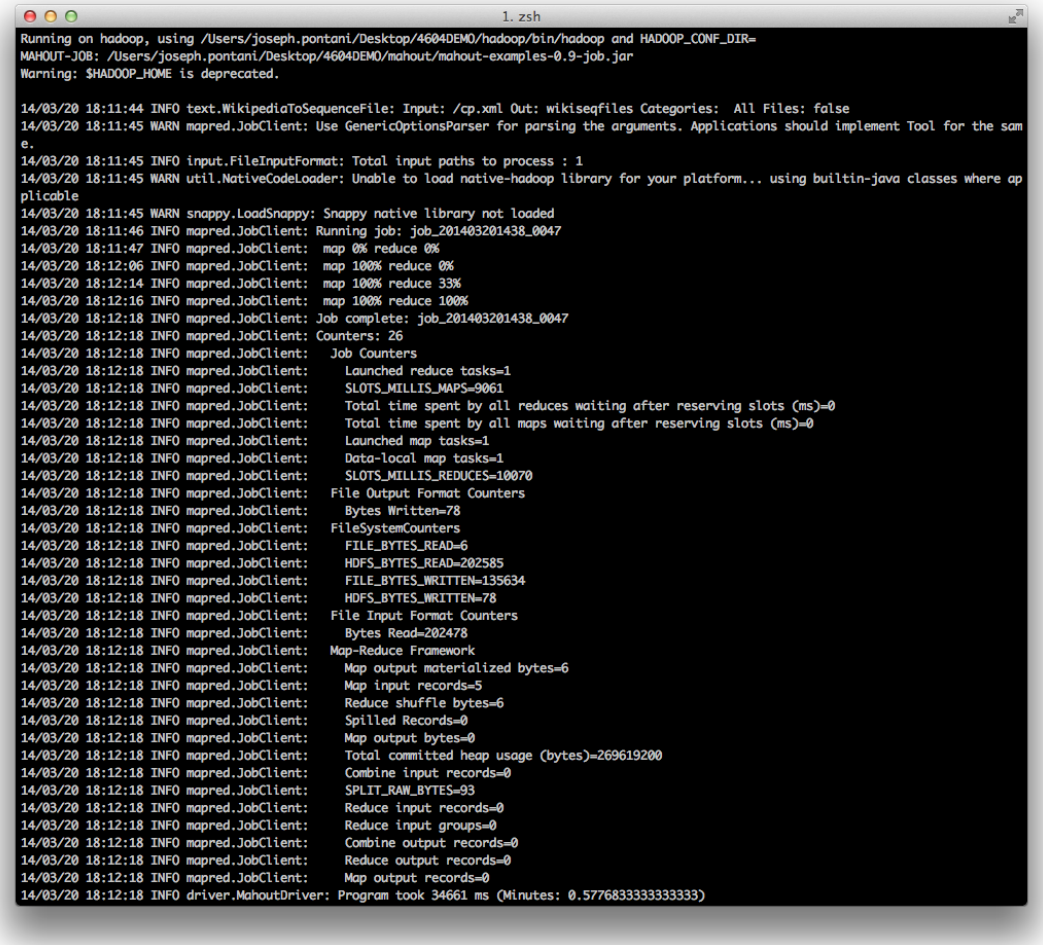
```
mahout seqdirectory -c UTF-8 -i plaintext/ -o pt_sequences/
```

Sequence a Wikipedia Export

Another way to create sequence files (especially useful when testing) is to simply use a Wikipedia XML export. To create a sequence from this XML file, we would run the command:

```
mahout seqwiki -i Wikipedia_export.xml -o wiki_sequence/
```

Running on a test export, you may see something similar to:



Creating Vector Files

This will take any plain text files in the directory “inputdirectory” and convert them to sequence files. We specify UTF8 character set to make sure each file inside the input directory is processed using the same set rather than being able to specify individual character sets. These sequence files must next be converted to vector files that Mahout can then run any number of algorithms on. To create these vector files, you will need to run the command:

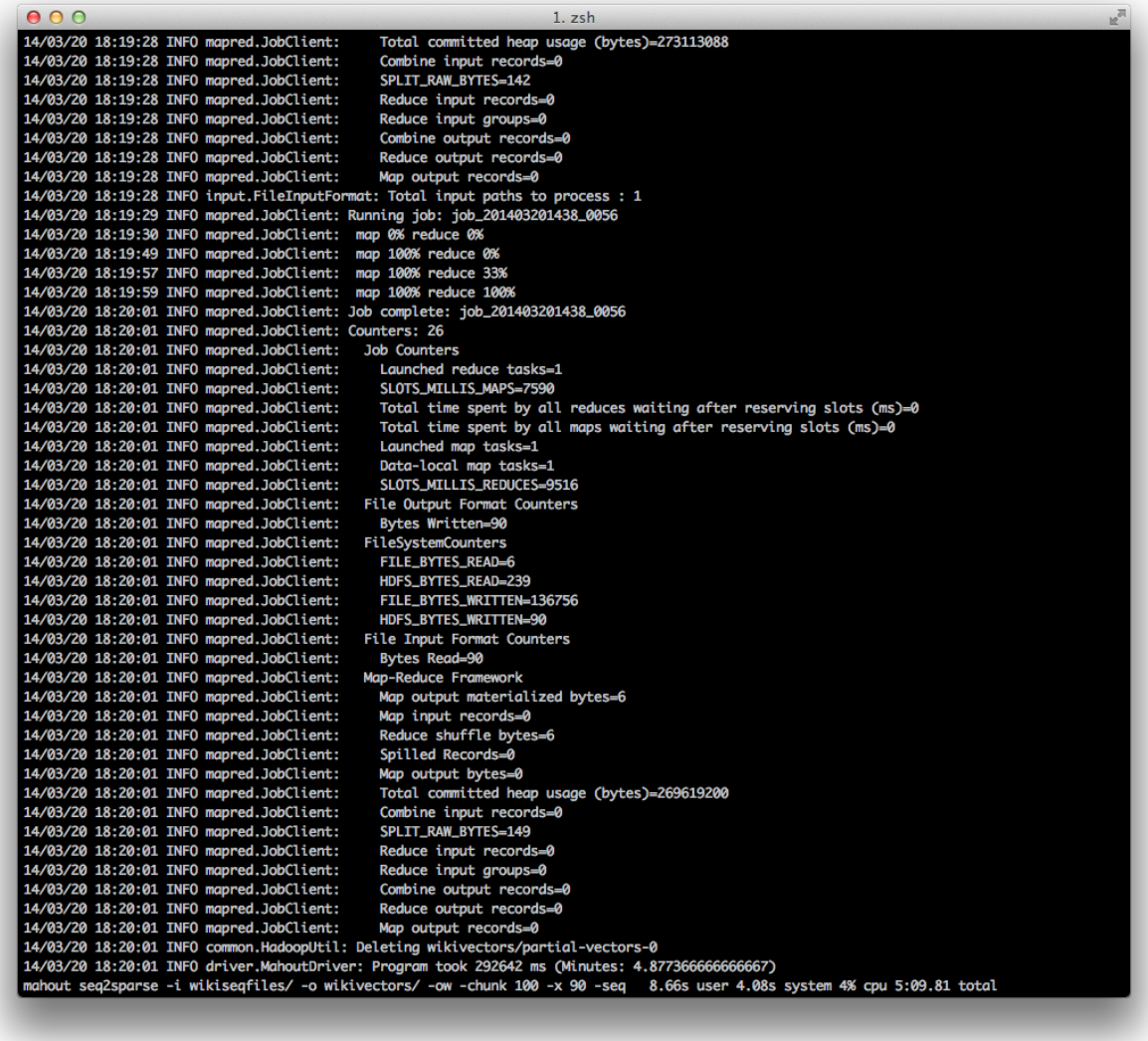
```
mahout seq2sparse -i sequences -o outputdirectory –ow –chunk 100 –x
                    90 –seq –ml 50 –n 2 -nv
```

The ‘-nv’ flag means that we use named vectors, so that the data files will be easier to inspect by hand. The ‘-x 90’ flag means that any word that appears in more than 90% of the files is a stop word, or a word is too common and thus is filtered out (i.e.: articles such as ‘the’, ‘an’, ‘a’, etc.).

From our Wikipedia export from above, I ran the command:

```
mahout seq2sparse -i wikiseqfiles/ -o wikivectors/ –ow –chunk 100 –
                    x 90 –seq –ml 50 –n 2 -nv
```

And the result was:



Analyzing Vectors with Mahout

Now that the vector files have been created, we can run analytics on these files. Let's say we wanted to see the top terms included in our original document (in this case, a Wikipedia export of 5 pages about computer programming). One simple algorithm Mahout offers is the k-means clustering algorithm. It will give us a list of top terms in the documents, or essentially a list of terms that we could cluster these documents around.

To run the k-means algorithm, we need to run the command:

```
mahout kmeans -i vectors/tfidf/ -c clusterseeds –cl –o clusters –k 20
–ow –x 10 –dm org.apache.mahout.common.distance.CosineDistanceMeasure
```

This command will run the k-means algorithm on the vector data, and output the cluster data to the 'clusters' directory. We specify a distance measuring technique via the '-dm' flag and pass it a fully-formed name to distance measuring class. The '-k 20' flag means we want to simply limit our cluster to the top 20 terms.

Other algorithms that can cluster data include Canopy, Fuzzy k-means and Spectral. Each algorithm clusters data together via differing techniques and approaches.

The resulting data in the 'clusters' directory, however, isn't really viewable just yet.

Viewing Cluster Data

We've now ran an algorithm to cluster our information together, now we need to see what the results are. Mahout offers an option for dumping a cluster to a plaintext file, which becomes quite a bit more readable than before.

To dump cluster data to a text file, we need to run the clusterdump command, via:

```
mahout clusterdump –d vectors/dictionary.file-0 –dt sequencefile –i
   clusters/cluster-#-final –n 20 –b 100 –o plaintextdump.txt –p
                        clusters/clusteredPoints
```

Before this command can be executed properly, you need to change the 'cluster-#-final' portion to the appropriate name, replacing the # with the largest number available that was created via the clustering algorithm. This command takes the dictionary file, the cluster data files, and the cluster points, and creates a plaintext dump file where you can view the top 20 terms for the data.

Running other algorithms (such as classification and filtering) utilize a similar approach, by simply creating sequence files from the raw data, analyzing those sequence files to vector files, and then by converting the vector files to cluster/filter/classification files.

Frequent Pattern Mining with Mahout

Another important aspect of Mahout is the ability to do frequent pattern mining via the FPGrowth algorithm. FPGrowth is a way to determine the most frequent groupings of items, be it transactional data with products, or words. FPGrowth can run off a simple dataset, such as a CSV file. No sequence file generation is required.

Running FPGrowth on a CSV

To run the FPGrowth algorithm, you need to start with a dataset. This dataset can be anything, words, transaction data, etc. However, row needs to be a separate instance of pairings between items. So, for example, if you're running FPGrowth on text (as I will detail later), each row simply needs to be a phrase or sentence. If you're using text, you can either process the dataset to have the words separated by a comma, or keep it as a space. You can specify to Mahout what the word separator will be.

For the sake of this tutorial, I will assume that the dataset is of words separated by spaces.

Running Mahout with FPGrowth is easier than the previous algorithms. We simply need to tell Mahout where our input file is, where to output the results, and then what our data is separated by.

To do this, simply run:

```
mahout fpg –i input_data.csv –o results –regex '[\ ]' –method
mapreduce –k 50 –s 2
```

In this command, -i specifies the input file, -o specifies the output directory, -regex gives Mahout a regular expression describing what the items are separated by (in this case the pattern matches a single space), -method tells Mahout to run using MapReduce, -k specifies that we want to keep the top 50 matches for each word, and –s tells Mahout that we want at least 2 pairings of items per word before it counts.

Viewing the Results

The output that gets placed in the results directory can be viewed by running the seqdumper command. This command takes an input of sequence files generated by FPGrowth, and converts it to a text file that can be viewed or parsed by custom software.

To run the seqdumper command to see the results, we run the command:

```
mahout seqdumper –i results/frequentpatterns/part-r-XXXXX –o
outputfilename
```

We need to specify a file for input into the seqdumper command, although directories should also work if there is more than a single result file. The result file for input is usually named part-r-XXXXX, where the X's are numbers (typically 00000). It then runs and creates an output text file that you can view with any text editor.

Mahout and Big Data

After learning the basics of how to use Mahout on a small-scale, single node cluster on Hadoop, you can then move on to your big datasets. In the examples above, a small pothole dataset was used. In this sense, the term small refers to the initial CSV file being only 18MB. Moving on to a large dataset, like a tweet dataset about shootings, can be quite daunting. The shooting dataset utilized on our 12-node cluster that the DLRL uses encompasses 8GB worth of text in a CSV file. This substantial increase in size adds some new challenges into the mix.

Data Processing

One challenge faced when attempting to analyze a dataset of this magnitude is how do you do any preprocessing before running Mahout? Attempting to organize and format data in an 8GB flat file is an extreme challenge. Running a script to process each the data line by line may take hours, if not days (depending on the number of lines). The most appropriate way to do data formatting would be to process it as the data is collected, rather than after the data is exported and about to be processed.

Another approach towards processing data would be to split your dataset into multiple, smaller datasets. You would then be able to process and format the data post-export faster, as well as analyze the data much faster. You would need to run Mahout either multiple times, once for each dataset, or combine the datasets into a single directory, and then use the directory sequencer that Mahout provides to create sequence files based on the data stored in a given directory.

Conclusion

You have now gone through a really basic tutorial on how to use Apache Mahout to do some basic data clustering. There are many more features that Mahout offers, including classification and collaborative filtering, and other algorithms that developers have created utilizing Mahout's core capabilities. Utilizing the power of Hadoop, data mining, pattern recognition, and information analysis should become that much easier when using Mahout as a foundation. Expanding Mahout onto a Hadoop cluster allows you to further the power be leveraging the computational power of many nodes.