Article from

Predictive Analytics and Futurism

July 2016
Issue 13

Using Hadoop and Spark for Distributed Predictive Modeling

By Dihui Lai and Richard Xu

In the age of big data, the physical world we live in is dynamically mapped to the digital world in the form of data: news, messages, pictures, videos, health records, stock market data, you name it. Cloud computing and various sensors have made this process simpler than ever before. The ability to process enormous amounts of data in a timely and insightful manner is becoming the key to business success.

Computational power is essential in speeding up our data processing, and distributed computing systems (e.g., Hadoop, Spark) seem to be good candidates compared to many others (e.g., graphics processing units (GPUs), better central processing units (CPUs), quantum computers, etc.). On the other hand, predictive modeling (PM) has shown its importance in sophisticated data analysis (e.g., spam filters, product recommendations). A recent breakthrough in machine learning has also been the key to the success of Google's AlphaGo.[1]

However, the two components do not naturally proceed together. Modeling algorithms are focused on accuracy more than speed. Making them compatible with a distributed system requires a deep understanding of computer hardware, data structures and modeling mathematics. To an organization/company, this is simply translated into "cost." There may be less expensive ways to do it. In this article, we are going to review the ways to do scalable predictive analytics with an emphasis on open-source packages that support the Hadoop ecosystem.

IS YOUR JOB PARALLELIZABLE?

Perhaps one of the most important steps in moving a computing task to a distributed system is to determine if it can be parallelized and what the best parallelizing strategy could be. When building predictive models, there are mainly two computational intensive jobs: optimization and hyper-parameter search. Planning them well is critical to creating an efficient program.

In general, a machine learning algorithm or statistical model has an error function (sum of squared residuals, cross-entropy, etc.) it needs to minimize. The optimization algorithm updates the model parameters iteratively until the error function is minimized, considering some value (derivatives, predicting errors) estimated at each data point. A simple parallel strategy has two

steps: mapping each data point to the value-needs-estimate and adjusting the model parameters accordingly. This description might remind you of a map-reduce job and, indeed, the strategy can be easily implemented in Hadoop/Spark since map-reduce is well supported there.

Besides the optimization, statistical models normally have a list of hyper-parameters associated with them (e.g., distribution prior, sampling ratio, variable selection ratio, etc.) Determining the best hyper-parameters is critical to model accuracy, and searching through the hyper-parameter space is a common practice. The search process is computationally expensive, and speeding it up will allow searching a larger space. An intuitive solution here is to create a pool of models with promising hyper-parameters and distribute them to worker nodes for concurrent evaluation.

Writing parallel code is nontrivial. It is tricky to balance efficiency with the overhead the code will introduce. It is not uncommon for a developer to find that after days or weeks of diligent work, the map-reduce job he wrote helps little to none on a program's execution time. In the following sections, we will introduce some active open-source projects that aim to make scalable machine learning easy.

SCALABLE MACHINE LEARNING PACKAGES

MLLIB/SPARKNET

More people are now accepting Spark as the new process engine for the Hadoop ecosystem.[2] Spark's in-memory support has made it ideal for developing scalable machine learning algorithms. MLlib is a product of such efforts from the Spark

community. The library covers a wide range of common algorithms: linear regression, naïve Bayes, decision trees, k-mean, etc. (see Table 1). SparkNet, the all-star deep-learning algorithm, is not included in MLlib but was developed in a separate Spark package.[3]

The library conveniently provides APIs to languages like Python, Java and Scala. As the library is built on top of built-in data structures like RDD or data frames, Spark's data processing tools (e.g., Spark SQL) come in handy to the user. Data manipulations like merging or subsetting can be handled smoothly without much painstaking work.

However, there is one piece missing in the MLlib that is important for actuarial use—the generalized linear model (GLM). Although linear regression and logistical regression are supported, MLlib is missing two important members of the GLM family: the Poisson distribution and the Tweedie distribution. These distributions are responsible for frequency models and loss-cost models.

Table 1: Comparisons of the machine learning algorithms supported by H2O, MLlib/SparkNet and Mahout

	H2O	MLlib/SparkNet	Mahout
Generalized Linear Model	X		
Random Forest	X	X	X
Naïve Bayes	X	X	X
Gradient Boosting Machine	X	X	
K-Mean Clustering	X	X	X
Cox Proportional Hazards	X		
SVM		X	

H2O

Compared to MLlib, which might seem like a direct application of the Spark engine, H2O was aiming to solve scalable statistical problems with its creation. As a key to fast machine learning algorithms, H2O supports in-memory processing as well. To actuaries' delight, H2O does support GLM and includes distributions like Poisson, gamma and Tweedie. Moreover, H2O also supports survival analysis like Cox-model (Table 1). However, H2O is slightly weak in data manipulation. For example, to add a derived variable from an existing column, users have to write a map-reduce job for the H2O-frame.

H2O can be plugged into Hadoop or Spark (with sparkling-water) clusters easily and leverages the capabilities of the distributed system: resource management, HDFS storage, data manipulation, etc. The current version of sparkling-water supports Scala and Python. R users can install H2O as a library and use H2O cluster by connecting to the service.

MAHOUT

Apache Mahout has a slightly longer history than the two packages described above. Most of its algorithms are designed within the framework of map-reduce. The initial project has been focused on algorithms like clustering and classification. In light of the Spark success, the Mahout project has recently shifted its focus from writing map-reduce algorithms to providing a platform supporting H2O, Spark and Apache Flink.

OTHER

Besides the open source projects listed above, commercial software like SAS, Revolution R (Microsoft) and Big R (IBM) all provide scalable predictive modeling on Hadoop/Spark with nontrivial cost—as the size of the cluster goes up, the cost will increase proportionally.

DISCUSSION

As the era of big data approaches, the need for fast big data analytics is becoming greater than ever. The open source projects we reviewed here provide us ways to gain power at relatively low cost. However, the packages are created with their own flavors and each has features others do not. Depending on the application, users need to choose the one that best fits their need. If your PM application requires lots of data manipulation, MLlib could be the best option. If the application requires using a model like GLM, H2O is your best friend. And, if your organization has plenty in its budget, it is hard to say no to the commercial software! ■

Dihui Lai, Ph.D., is assistant data scientist, global research and development, at RGA Reinsurance Co. in Chesterfield, Mo. He can be reached at *dlai@rgare.com.*

Richard Xu is vice president and actuary, head of data science, at RGA Reinsurance Co. in Chesterfield, Mo. He can be reached at *rxu@rgare.com.*

ENDNOTES

1 David Silver, et al., "Mastering the Game of Go With Deep Neural Networks and Tree Search," Nature 529, no. 7587 (2016), 484-89.

2 Phillipp Moritz, Robert Nishihara, Ion Stoica and Michael I. Jordan, "SparkNet: Training Deep Networks in Spark," (conference paper, ICLR, 2016).

3 Tom White, Hadoop: The Definitive Guide, 3rd ed. (Sebastaopol, CA: O'Reilly Media/Yahoo Press, 2012).