Sparkling Infrastructure

H2O

H2O is a big-data machine learning and predictive analytics platform, reputable for its fast and scalable deep learning capabilities. These include supervised and unsupervised learning algorithms like neural networks, tree ensembling, generalized linear regression models and k-means clustering. Additionally H2O provides deep learning capabilities through algorithms like perceptrons. In its essence, H2O is a Java Virtual Machine (JVM), which is an abstract computing environment for running a Java instance. JVM provides a close and secure environment for running Java applications thus preventing encroachment from malicious users [@JVM]. H2O's contained environment also makes it optimal for performing distributed and parallel machine learning computations simultaneously on clusters.

H2O installation process: H2O installation in R entails installation of Spark 1.6 and H2O 3.10.06 version since this version of H2O is integrated with rsparkling. H2O installation first involves removal of any previous versions of H2O followed by download of H2O dependency packages and lastly installation and initialization of H2O packages for R. Package download link can be obtained at (http://h2o-release.s3.amazonaws.com/h2o/rel-turing/6/index.html#R)[1]. Rsparkling package is downloaded using devtools.

Before using H2O's machine learning algorithms, the same version of Spark and Sparkling water needs to be installed in R. Desktop version of Spark can be downloaded at (http://spark.apache.org/downloads.html)[2]. Desktop version of Sparkling Water can be downloaded at (http://h2o-release.s3.amazonaws.com/sparkling-water/rel-1.6/8/index.html)[3]. H2O also necessitates initialization of a local Spark connection, which can be instantiated with the spark_connect function. In R, H2O was connected to the local cluster. In comparison, for Hadoop platform, H2O was connected to yarn-client which is discussed further below.

Apache Spark

Spark is a big-data platform that provides fast in-memory distributed processing. This contrasts with Hadoop, which employs the MapReduce processing platform and necessitates the need to write data to an external disk through the Hadoop Distributed File System (HDFS) [@HDFS]. Due to multiple replication, serialization, and disk IO, about 90% of time is spent on HDFS read-write operations, which slows the computation and processing time [@tutorialApache]. In comparison, Spark uses the Resilient Distributed Datasets (RDD) data structure, which divides the dataset in logical partitions each of which can be processed on separate nodes within clusters. This RDD structure obviates the need to write data to an external storage system thus providing faster in-memory processing [@RDD]. Moreover Apache Spark is an in-memory data processing tool that hosts the Spark platform on Apache Hadoop YARN providing a collective and shared access to a dataset.

In R, the sparklyr package provides an R interface for Apache Spark. Besides a dplyr background, this facilitates the use of Spark's distributed machine learning library.

Sparkling Water

Sparkling water combines the machine learning capabilities of H2O with the in-memory distributed, fast computation of the Spark platform. Tachyon, which is an in-memory distributed file system, facilitates exchange of data between Spark and H2O [@Tachyon]. The rsparkling package in R provides access to H2O's machine learning routines within the Spark platform accessible over R. Spark data frames can be converted to H2O frames thus facilitating machine learning algorithms.

[1] @H2OInstall
[2] @DownloadSpark
[3] @DownloadSparklingWater

RSparkling

The rsparkling package provides the ability to transfer data between Spark and H2O dataframes. Rsparkling also allows access to Sparkling Water spark package's machine learning algorithms [@SparklingWaterML].

Hadoop Yarn

Apache Hadoop Yarn includes separate resource management and job scheduling infrastructures in collective resource manager and individual node manager through a master-slave hierarchy (as shown by Figure 1). The per-application based application masters negotiate resources with the resource manager and execute and monitor tasks through a collaboration with the node managers. The resource managers additionally contain a scheduler that schedules jobs based on resource requirements. Individual node managers are responsible for launching application containers and examining resource usage (like memory and disk consumption). These updates are then reported back to the resource manager. Per-node application master negotiate resource containers with scheduler [@hortonApache].

Figure 1: Hadoop YARN architecture

Additional Resources

- H2O Installation guide - http://h2o-release.s3.amazonaws.com/h2o/rel-turing/6/index.html#R
- H2O Documentation - http://h2o-release.s3.amazonaws.com/h2o/rel-lambert/5/docs-website/index.html
- Sparkling Water Installation - http://www.h2o.ai/download/sparkling-water/
- Sparkling Water Overview - http://spark.rstudio.com/h2o.html
- Sparklyr Overview and Installation - http://spark.rstudio.com