6.883 Final Project Proposal: Incremental Random Forest Classifiers in Spark ML

Katie Siegel (ksiegel@mit.edu)
John Holliman (holliman@mit.edu)

Random forests have become one of the most popular machine learning classifiers due to their robustness to noisy data, accuracy, and ability to handle big data workloads. At a high level, random forests are collections of decision trees used for classification. Once grown, each decision tree classifies an unlabeled point by casting a vote, and the random forest reports the label with the most votes.

The downside to using random forests is that, given additional training data, the random forest must be regrown from scratch. This project will explore several techniques for updating a random forest incrementally without fully regrowing the classifier, and will provide insight into the performance and robustness of these techniques on different workloads. Specifically, we will develop a custom incremental random forest classifier in Spark and provide a Scala API through which users can call incremental random forest methods.

1 Batched Workflows

A common pattern in data science workflows involves training the same model on increasing amounts of data. Such workloads are common in analytics, where observations are continuously collected in log entries, as well as in Internet of Things (IoT) networks, where extensive data collection takes place offline and data is transmitted periodically. Existing machine learning tools retrain a model on the entire dataset when new data is added. This involves reiterating over every single data point in the dataset, even when the added batch has a minimal effect on the resulting classifier. As such, implementing incremental training in Spark random forests should drastically improve the performance of these classifiers on batched workloads.

2 Spark Random Forest Classifiers

This project involves the Spark Machine Learning (ML) implementation of random forest classifiers. Apache Spark is a scalable system developed at Berkeley that provides an engine for processing big data workloads. At its core is a structure called the resilient distributed dataset (RDD), which can be distributed over a cluster of machines and is fault-tolerant. A series of powerful libraries run on Spark and take advantage of its powerful cluster-computing capabilities. One such library is Spark MLlib, which contains a wide array of machine learning tools. In this project, we will model my incremental random forest classifier on the Spark random forest classifier, for the purposes of maintaining optimizations within the codebase that take advantage of Spark's strengths. Since each batch in a batched workload may be large, implementing an incremental classifier in Spark allows us to take advantage of Spark's distributed computing capabilities for every batch.

The existing Spark ML random forest classifier performs well given a large dataset. However, like all implementations of random forests, a change in the training dataset would mean retraining the random forest from scratch. An incremental implementation would allow random forest classifiers to be updated with each new batch of training data more efficiently, therefore saving time for data scientists and other users of Spark MLlib.