SQML: Large-scale in-database machine learning with pure SQL

Umar Syed
Google
usyed@google.com

Sergei Vassilvitskii
Google
sergeiv@google.com

Many enterprises have migrated their data from an on-site database to a cloud-based *database-as-a-service* that handles all database-related administrative tasks while providing a simple SQL interface to the end user. Businesses are also increasingly relying on machine learning to understand their customers and develop new products. Given these converging trends, there is a pressing need for database-as-a-service providers to add support for sophisticated machine learning algorithms to the core functionality of their products.

In this paper we present SQML (*i.e.*, SQL + ML): a general-purpose, large-scale, in-database machine learning system implemented entirely with pure SQL and client-side driver scripts. SQML has several advantages over existing in-database machine learning methods, especially for a database-as-a-service:

- Unlike the most popular in-database machine learning systems, such as Bismarck [1], SQML does not require user-defined aggregates (UDAs). This is a key advantage for database-as-a-service providers who have tailored their products to prioritize simplicity over extensibility and do not support UDAs.
- Even when UDAs are available, UDA-based approaches assume that the learned model can fit in memory on a single machine. SQML represents models as disk-backed tables that are partitioned across many machines, so it can scale to arbitrary model sizes.
- SQML estimates *generalized linear models*, a large class of models for supervised machine learning that includes linear regression, logistic regression, and support vector machines as special cases. Existing pure SQL implementations of in-database machine learning support only a single type of relatively primitive model, such as a Naive Bayes classifier [2].

The SQML learning algorithm is based on *preconditioned proximal gradient descent*, a state-of-the-art method for convex optimization. We implemented both SQML and Bismarck's UDA-based algorithm using Dremel, a database-as-a-service available internally at Google. We trained models on two datasets collected from Google's online display advertising business: Clicks on mobile display ad impressions (300M examples, 10M features), and bids on display ad impressions sold via auction in a real-time ad exchange (1.8B examples, 200M features). Each impression was described by features such as userid, publisher, vertical, and so on. For the click data, we trained a logistic regression model to predict clickthrough rate, and for the auction data we trained a linear regression model to predict log(highest bid).

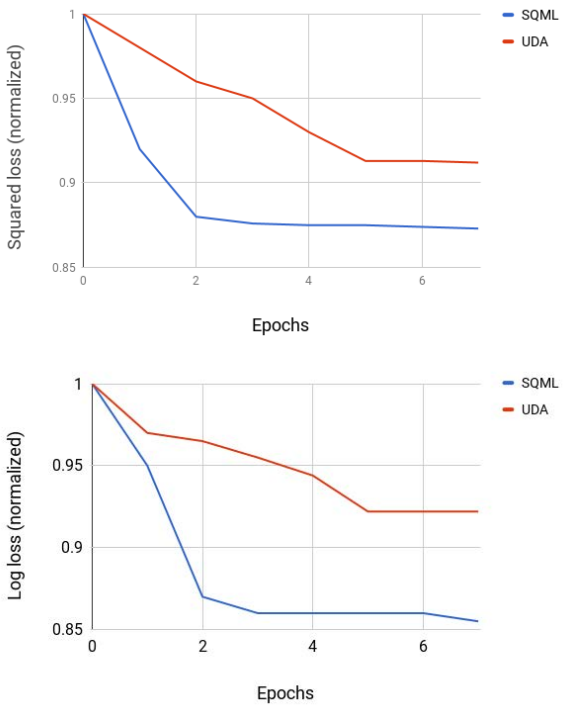

Figure 1: Normalized performance of the SQML and UDA-based algorithms on the bid prediction problem (top) and clickthrough rate prediction problem (bottom).

The results of the experiment are shown in Figure 1. SQML outperformed the UDA-based algorithm for two main reasons. First, Dremel imposes relatively tight memory limits on each user, and since UDAs must hold the entire model in memory, we had to use model compression techniques that invariably degrade model quality. SQML does not require model compression, since the models are stored in disk-backed tables. Second, on large training sets UDA-based algorithms learn several independent models on disjoint subsets of the training data and then average them together, an approach that slows convergence. SQML always learns a single model on the entire dataset.

REFERENCES

[1] X. Feng, A. Kumar, B. Recht, and C. Ré. Towards a unified architecture for in-RDBMS analytics. SIGMOD, 2012.
[2] S. K. Pitchaimalai and C. Ordonez. Bayesian classifiers programmed in SQL. *IEEE Trans. Knowl. Data Eng.*, 22:139–144, 2009.

SoCC '17, Santa Clara, CA, USA
 978-1-4503-5028-0/17/09…$15.00
DOI: 10.1145/3127479.3132746