MODELDB: A System for Machine Learning Model Management

Manasi Vartak
MIT
mvartak@csail.mit.edu

Introduction. Building a machine learning model is an iterative process. A data scientist will build many tens to hundreds of models before arriving at one that meets some acceptance criteria (e.g. AUC cutoff, accuracy threshold). However, the current style of model building is ad-hoc and there is no practical way for a data scientist to manage models that are built over time. This gives rise to three types of problems: (1) *Reproducing models and results is excessively time-consuming or at times infeasible.* E.g., one of the companies we interviewed described how lack of documentation regarding a previous experiment forced them to spend a week recreating and running the same experiment; (2) *The data scientist must "remember" results and parameters of previous versions of a model.* E.g., the data scientist must remember what combinations of parameters or features have been tested as well as their results in order to draw insights and inform the next experiment; (3) *Data scientists have no means of answering aggregate or example-level questions regarding different versions of a model.* E.g., a data scientist often discovers a discrepancy in the code or data at some point and must then re-run all analyses downstream from it. However, in the absence of model versioning, identifying experiments that must be re-run is an uphill battle.

The above challenges which are a result of the iterative nature of modeling highlight an important and little-studied problem for machine learning tools: *model management*. Model management is the problem of tracking, storing, and indexing large numbers of machine learning models so that they may subsequently be reproduced, shared, queried and analyzed. In this talk, we will present ongoing work on ModelDB, a system to manage machine learning models. ModelDB automatically tracks models in their native environments, indexes them intelligently, and allows flexible exploration of models via SQL as well as a visual interface.

Related Work. Scientific workflow management is a rich area of research that has produced systems including Kepler [3], Taverna [4], and VisTrails [1]. Commercial software vendors have also introduced tools such as the AzureML suite and the SeaHorse suite that allow the (graphical) construction of machine learning workflows. While these systems provide good workflow management, they suffer from several drawbacks: (a) almost all of these systems require workflows to be pre-specified via GUIs, a mode of interaction unacceptable to most data scientists; (b) these systems track some subset of – but not all – ingredients required for reproducibility and versioning: code, data, models and results; (c) they have limited ability to build workflows across multiple environments (e.g. hadoop + torch); (d) they do not support interesting queries on models as described above.

Architecture. Figure 1 shows the high-level architecture of our system. ModelDB consists of three key components: native client libraries for different machine learning environments, a backend that defines key abstractions and brokers access to the storage layer, and a web-based visualization interface. Using the native client libraries for ModelDB (currently available for spark.ml and scikit-learn), data scientists can perform experimentation and model building in their favorite ML environment as usual while, in the background, the library automatically extracts relevant information and passes it to the ModelDB backend. The backend exposes a thrift interface to allow clients in different languages to communicate with it. ModelDB stores models and pipelines as a sequence of actions (as opposed to states) and uses a branching model of history to track the changes in models over time [2]. The backend uses a relational database to store modeling workflows while a custom storage engine is used to store and index models. Logging model-specific metadata and results allows ModelDB to support aggregate as well as example-level queries as described in the introduction. The third component of ModelDB, the visual interface, provides an easy-to-navigate layer on top of the database that permits visual exploration and analyses of models and workflows.

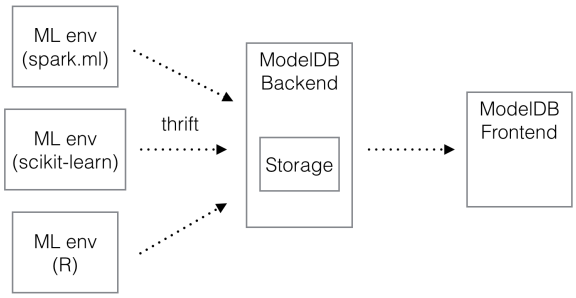

Figure 1: ModelDB Architecture

In this abstract, we described ongoing work on ModelDB, a novel system to manage machine learning models. The talk will describe details of the system design and results from early adopters of ModelDB.

1. REFERENCES

[1] L. Bavoil, S. P. Callahan, P. J. Crossno, J. Freire, C. E. Scheidegger, C. T. Silva, and H. T. Vo. Vistrails: Enabling interactive multiple-view visualizations. In *Visualization, 2005. VIS 05. IEEE*, pages 135–142. IEEE, 2005.

[2] M. Derthick and S. F. Roth. Enhancing data exploration with a branching history of user operations, 2001.

[3] B. Ludäscher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. Scientific workflow management and the kepler system. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2006.

[4] K. Wolstencroft, R. Haines, D. Fellows, A. Williams, D. Withers, S. Owen, S. Soiland-Reyes, I. Dunlop, A. Nenadic, P. Fisher, et al. The taverna workflow suite: designing and executing workflows of web services on the desktop, web or in the cloud. *Nucleic acids research*, page gkt328, 2013.