Scalable forecasts for IoT in the cloud

November 26 2018, by Brad Eck

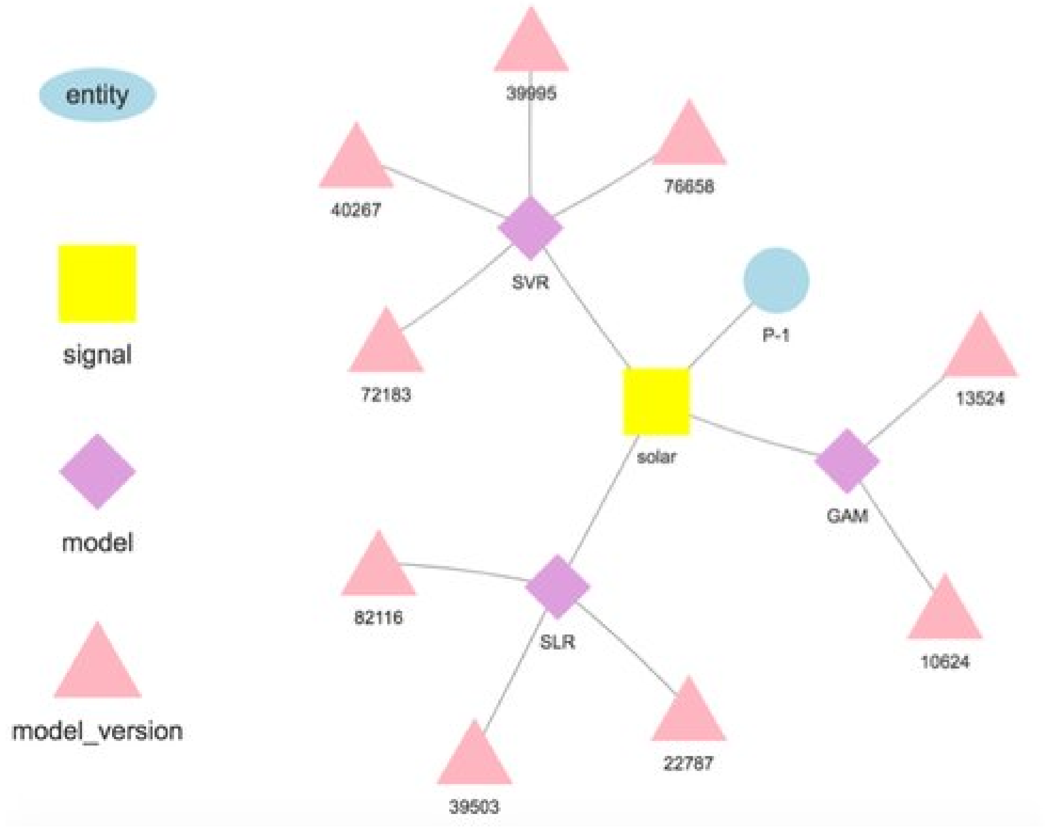

Figure 1. Model hierarchy for a selected entity and signal. Credit: IBM

This week at the International Conference on Data Mining, IBM Research-Ireland scientist Francesco Fusco demonstrated IBM Research Castor, a system for managing time series data and models at scale and on the cloud. Businesses of today run on forecasts. Whether a hunch of what we think is going to happen or the product of carefully honed analysis, we have a picture of what's going to happen and we act accordingly. IBM Research Castor is for IoT-driven businesses needing hundreds or thousands of different forecasts for time series. Although the model for an individual forecast may be small, keeping up with the provenance and performance of this number of models can be a challenge. In contrast to AI-driven cases using a small number of big models for image processing or natural language, this work aims at the IoT applications needing a large number of smaller models.

Our system provides a rich but selective set of capabilities for time series data and models. It ingests data from IoT devices or other sources. It provides access to the data using semantics, allowing users to retrieve data like this: getTimeseries(myServer, "Store1234", "hourly revenue").

It stores models written in R or Python for training and scoring. Every model is associated with an entity describing where the data originates, like "Store1234" above, and a signal describing what is measured, like "hourly revenue". Models are trained and scored at user-defined frequencies, and in contrast to many other offerings, the forecasts are stored automatically.

Data scientists deploy models by implementing a four-step workflow:

1. Load the data for training or scoring from relevant data sources;
2. Transform that data into a data frame for model training or scoring;
3. Train the model to obtain a version suitable for making forecasts; and
4. Score the model to forecast quantities of interest.

Once the model is deployed, the system carries out the training and scoring, automatically storing the trained model and forecast results. Data used in training and scoring need not originate on the platform, allowing models to use data from multiple sources. In fact, this is a key motivation for our work—making value-added forecasts based on multiple data sources. For example, a business can combine some of its own data with data purchased from a third party, such as weather forecasts, to predict a quantity of interest.

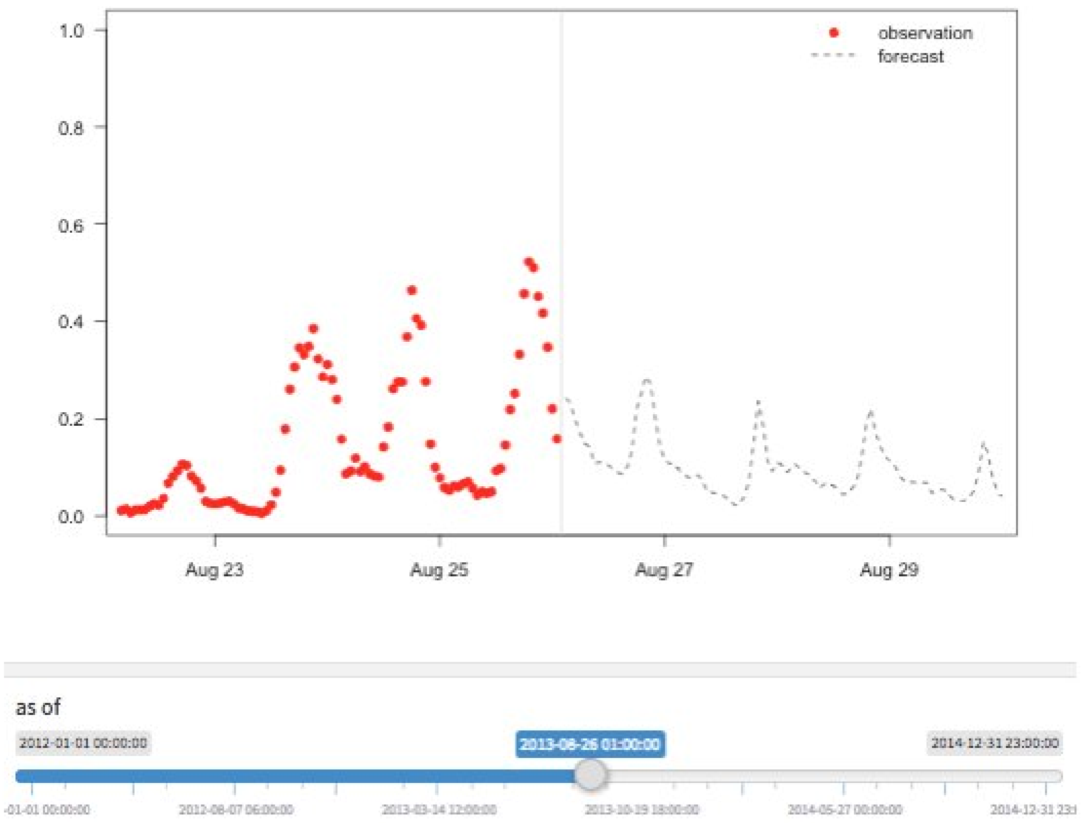

Figure 2. “Time machine” view showing available observations and forecasts for different points in history. Credit: IBM

Our system stores models separate from configuration and runtime parameters. This separation allows the changing of some details of a model, such as the API key for accessing third-party data or the scoring frequency, without redeployment. Several models for the same target variable are supported and encouraged to enable comparisons of forecasts from different algorithms. Models can be chained together so that output of one model forms the input to another as in an ensemble. A model trained on a specific dataset represents a model version, which is also tracked. Thus it is possible to establish the provenance of models and forecasts (Figure 1).

Several views are available to explore forecast values. Of course values themselves can be retrieved and visualized. We also support a "time machine" view showing the latest forecasts and latest observations (Figure 2). In this interactive view, the user can select different points in history and see what information was available at the time. We also support a view of forecast evolution showing successive forecasts for the same point in time (Figure 3). In this way users can see how forecasts changed as the target time became closer.

Under the hood, IBM Research Castor makes heavy use of serverless computing to provide resource elasticity and cost control. Typical deployments see models trained every week or every month and scored every hour. At training or scoring time, a serverless function is created for each model, allowing hundreds of models to train or score in parallel at the desired time. After this work is over, the computing resource disappears until it's needed again. In a more conventional workflow, virtual machines or cloud containers are idle when not in use but still attracting cost.

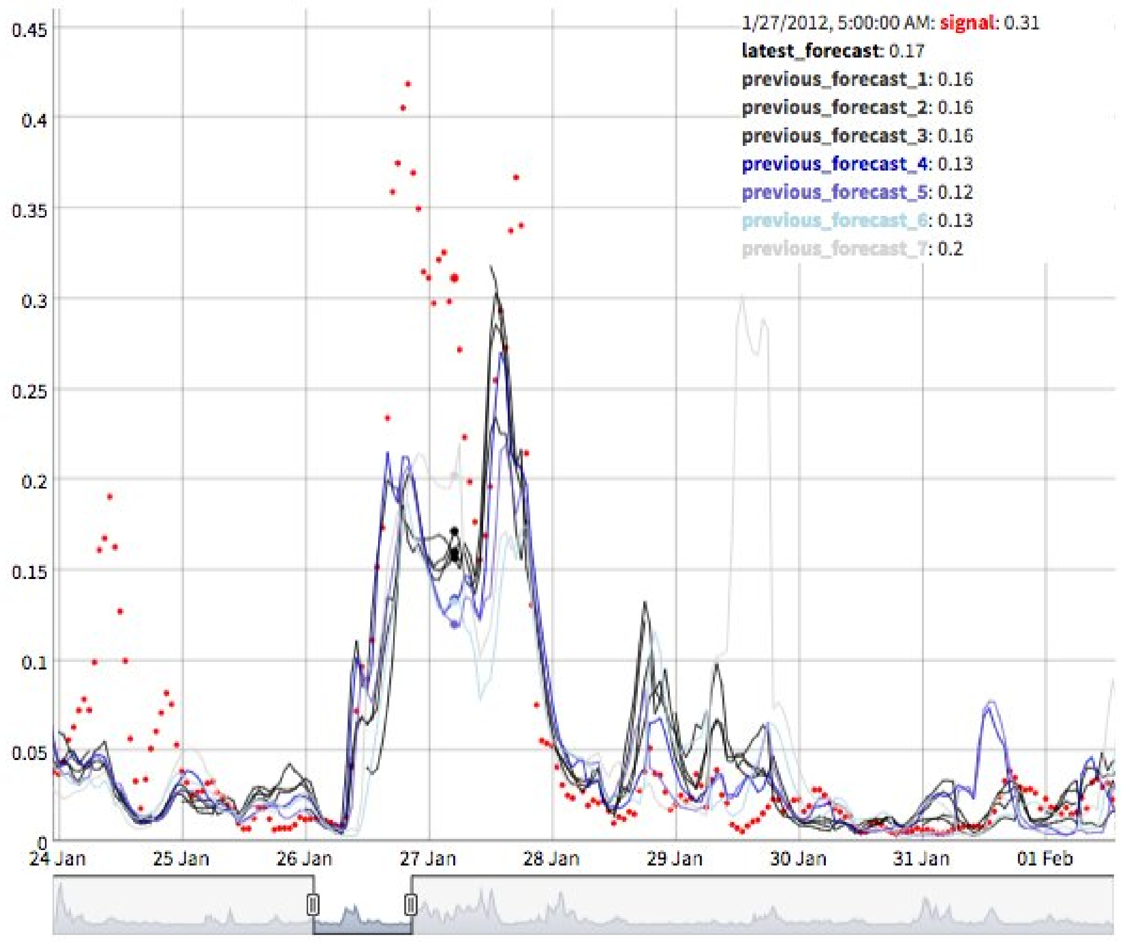

Figure 3. Forecast evolution. Credit: IBM

IBM Research Castor deploys natively on IBM Cloud using the latest services such as IBM's DashDB, Compose, Cloud Functions, and Kubernetes to provide a robust and reliable system. With an entitled account on IBM Cloud, IBM Research Castor deploys in a matter of minutes, making it ideal for proof-of-concept as well as longer running projects. Client packages / SDKs for Python and R are provided so that data scientists can get up and running quickly in a familiar environment and visualization teams can leverage familiar frameworks such Django and Shiny. If those don't suit your application, the JSON-based messaging API is also available.

More information: Castor: Contextual IoT Time Series Data and Model Management at Scale. Bei Chen, Bradley Eck, Francesco Fusco, Robert Gormally, Mark Purcell, Mathieu Sinn, Seshu Tirupathi. 2018 IEEE International Conference on Data Mining (ICDM)

This story is republished courtesy of IBM Research. Read the original story here.

Provided by IBM

Citation: Scalable forecasts for IoT in the cloud (2018, November 26) retrieved 26 April 2024 from https://techxplore.com/news/2018-11-scalable-iot-cloud.html