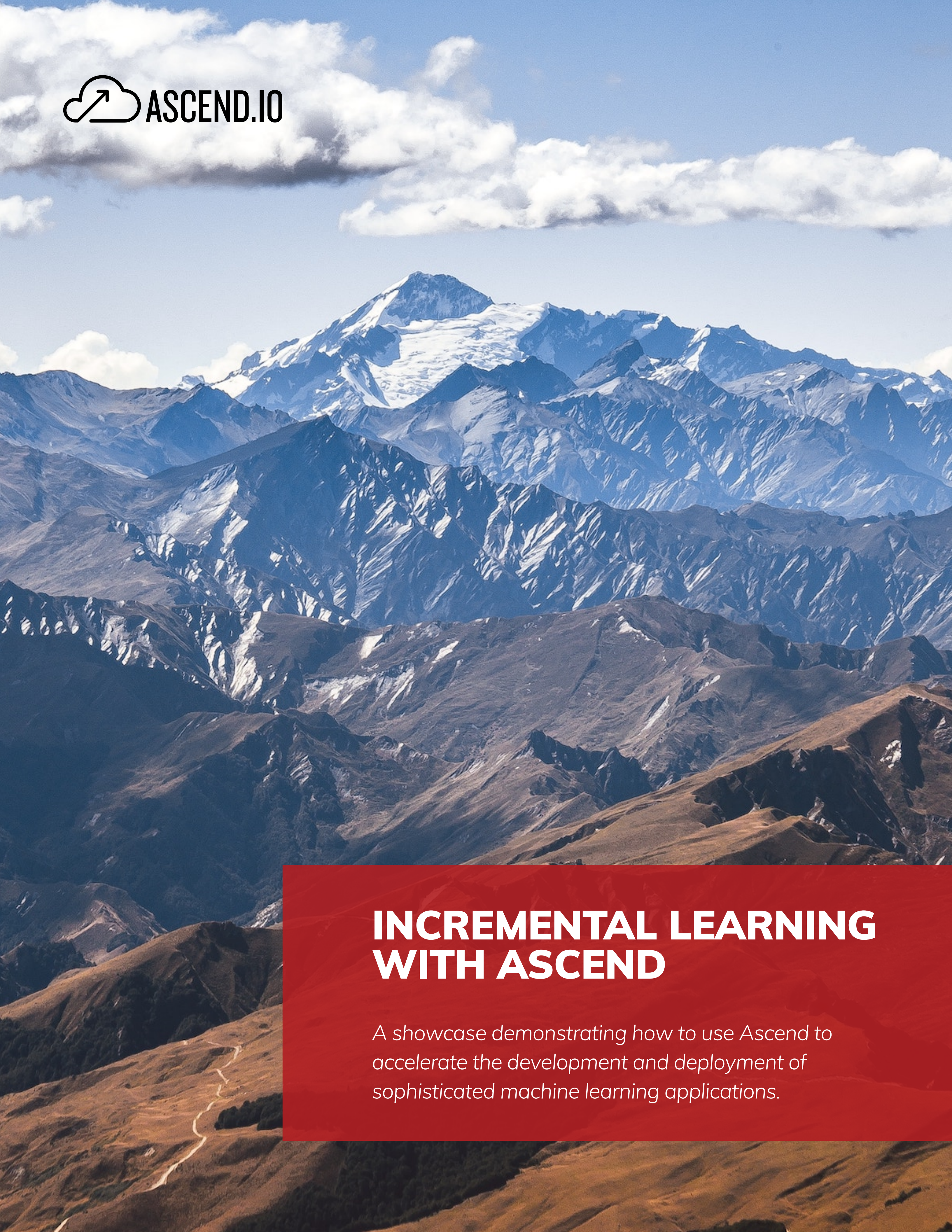

ASCEND.IO

INCREMENTAL LEARNING WITH ASCEND

A showcase demonstrating how to use Ascend to accelerate the development and deployment of sophisticated machine learning applications.

What is new here?

The traditional machine learning methods widely used today generally assume that a comprehensive “good” training dataset in the domain of interest is available a priori. “Good” means making assumptions that the data contains sufficient information to find and validate a single best model, and that applying that model to any new data in the domain generates reliable forecasts. This static approach emphasizes learning as much as possible from a fixed training and testing set.

Unfortunately, many real-world applications cannot match such assumptions. Datasets can be huge in volume and still not suffice to properly quantify the relevance of many features, arrive sequentially, as well as reflect phenomena that change significantly over time (like consumer behavior or traffic patterns).

> *With its unprecedented autonomous data orchestration platform, Ascend unlocks the next generation of dynamic machine learning techniques that until now have been hidden within large digital natives.*

In recent years, advanced machine learning practices have begun adopting more dynamic approaches that have long been understood in mathematics, computer science and academia. At commercial scale, such approaches require massively scaled data management techniques that until recently have been limited to the large digital native enterprises. These capabilities are now becoming accessible to the world’s enterprises through the unprecedented sophistication of Ascend’s autonomous data platform, which builds end-to-end data flows while automating the orchestration of cloud services to bring them to life.

What is incremental learning and why does it matter?

To demonstrate how such dynamic machine learning techniques work in real life, we focus on incremental learning, an important class of techniques which simulates learning by "forgetting" old data, retaining "learned" insights, and continuously "retraining" the operational forecasting model. The technique is valuable in that it:

- Prevents performance and throughput degradation under massive data conditions
- Significantly reduces storage bloat and compute costs
- Mitigates historical bias introduced by data from specific training periods
- Constantly adjusts for fundamental changes in the environment and / or behavior
- Unifying model training and model deployment

Incremental learning is increasingly useful in data-heavy, non-static scenarios like:

- Autonomous robotics
- Human feedback on websites and apps
- Very large, streaming datasets driving short-term forecasts
- Profile building and maintenance over years

> *The automation of incremental learning methods is poised to improve predictions in today’s rapidly changing world.*

Success in business depends on the accuracy of forecasts, so as enterprises become more digital and the world around us changes at an increasing pace, dynamic techniques like incremental learning become more relevant.

How does incremental learning work?

Incremental learning essentially works by accumulating mini-batches or streaming of new data, generating a compact representation of the signals in the data in the form of an ML model, discarding the data mini-batch, and iterating to accumulate the next mini-batch. With each additional mini-batch, the technique updates the model over time. Variants of the technique may retain a sliding window of a set number of mini-batches before discarding them.

> *Incremental learning techniques have long been well understood in mathematics, computer science and academia, but difficult to implement in practice.*

Incremental learning has to deal with a host of technical challenges in the core algorithms that we are not addressing here, such as concept drift, balancing effects of the stability-plasticity dilemma, adaptive model complexity, and more (Thapliyal, 2018). For our purpose, these challenges are addressed in the actual model design and training techniques, the implementation of which occurs on the machine learning platforms of Ascend partners.

In this paper, we focus on the commercial operationalization of data pipelines, with the high level of automation demanded by these dynamic techniques.

The following diagram demonstrates the iterative nature of the incremental learning technique.

1. An initial batch of data is collected, and used to generate an initial model (F0).
2. From then on, new mini-batches of sample are accumulated, bounded by time, volume, or other parameter that triggers a “closing” of the batch and “opening” of the next batch to continue the accumulation.
3. When a batch is “closed”, the system takes that batch to generate an update of the current model. The actual generation and management of the model is the domain of third-party machine learning platforms or library like skLearn in Python and mllib in PySpark that are partners in this approach.
4. The ML platform or library also performs an evaluation of the updated model. Depending on this fitness test, the system either stores the updated model for the next iteration, or discards it and revert to the previous one.

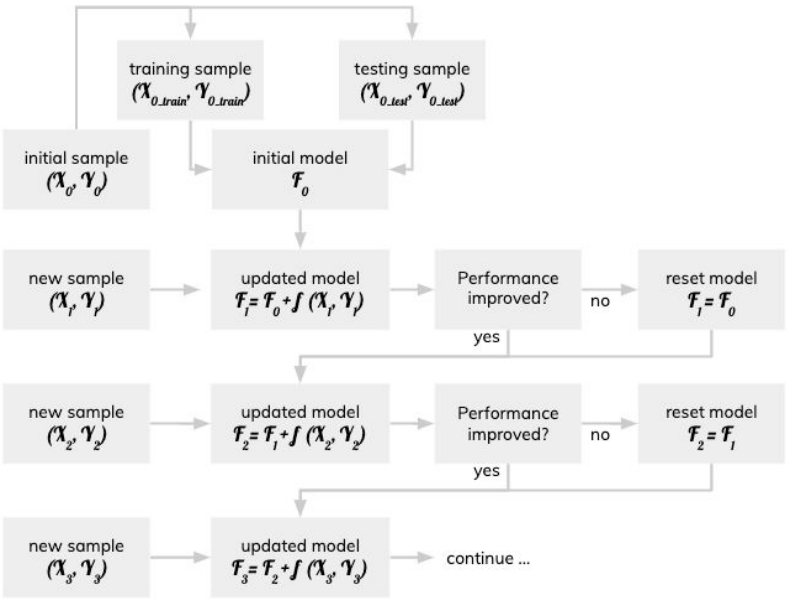

Figure 1

> *It is finally possible to operationalize sophisticated incremental learning techniques for commercial benefit without building a custom data management platform.*

How do I use Ascend to implement incremental learning?

The operationalization of incremental learning and similar dynamic techniques fundamentally demands sophisticated pipelines. The Ascend platform unlocks commercial adoption by enabling data experts to intuitively design and test the needed data operations, and then automating the deployment and ongoing operation of all the necessary data management operations, including the interactions with the ML partners and libraries.

Continuous modeling

There are two sides to the operation of incremental learning techniques. The first is in creating and updating the model, which initializes and runs continuously:

- Connect to various data sources,
- Clean and join the raw data,
- Generate the data batches,
- Automate and optimize all compute and storage resources,
- Interoperate with the model libraries and third-party platforms,
- Handle the data inputs to the models,
- Persist the resulting models and parameters,
- Manage the timing and ongoing iterations of the technique,
- Monitor and alert all operations in real-time.

The following diagram illustrates the part of the application that feeds the model construction as well as the iterative updating cycles for the model:

- A set of built-in connectors access the data sources, which can vary from highly dynamic (weather feeds, vehicles in motion, click streams) to more static metadata (vehicle fleet catalog, location listings, street name listings);
- A series of cleansing, formatting, and joining operations that prepare the data for model development, training, and validation;
- A series of operations invoking and managing third-party ML platforms and libraries:
    - ○ A core set of feature engineering and model building operations;
    - ○ A set of initial model training operations;
    - ○ A set of model updating operations;
- A method to store and retrieve models for each iteration of model updates.

The Ascend platform provides constructors for ML practitioners and data engineers to design and deploy these operations, and automatically runs all the data orchestration.

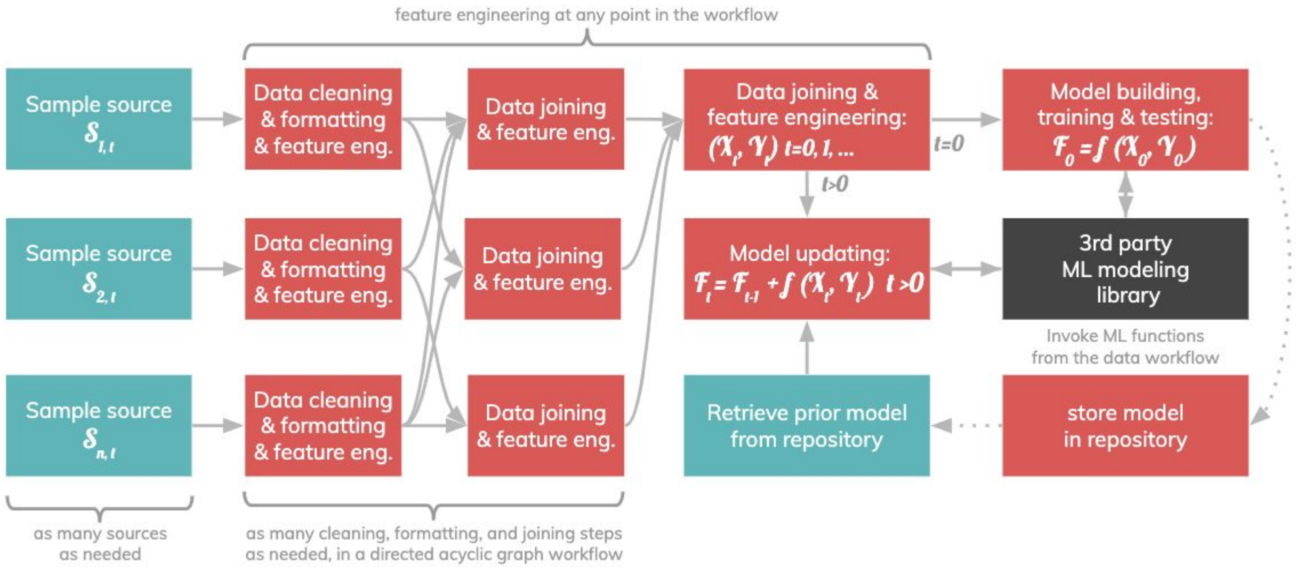
feature engineering at any point in the workflow
Sample source $S_{1,t}$
Sample source $S_{2,t}$
Sample source $S_{n,t}$
Data cleaning & formatting & feature eng.
Data cleaning & formatting & feature eng.
Data cleaning & formatting & feature eng.
Data joining & feature eng.
Data joining & feature eng.
Data joining & feature eng.
Data joining & feature engineering: (X_t, Y_t) $t=0, 1, \dots$
$t=0$
$t>0$
Model building, training & testing: $F_0 = f(X_0, Y_0)$
Model updating: $F_t = F_{t-1} + f(X_t, Y_t)$ $t>0$
3rd party ML modeling library
Invoke ML functions from the data workflow
Retrieve prior model from repository
store model in repository
as many sources as needed
as many cleaning, formatting, and joining steps as needed, in a directed acyclic graph workflow

Figure 2

Continuous forecasting

The second side of these incremental learning techniques is the prediction of the unlabeled data. Unlike common static model deployment, incremental learning enables continuous deployment of updated models, linking to existing data sources, and routing the resulting forecasts to consuming applications without their awareness. The level of automation and continuous operation available with Ascend is a game changer for operationalizing these techniques:

- Connect to the sources of unlabeled data,
- Clean and join the raw unlabeled data,
- Monitor the model repository for updated models,
- Handle the updated models to the model libraries and third-party platforms,
- Handle the unlabeled data inputs to the models,
- Receive and persist the resulting predictions,
- Manage the timing and ongoing iterations of the predictions,
- Feed the predictions to the consuming application(s),
- Automate and optimize all compute and storage resources,
- Monitor and alert all operations in real-time.

The following diagram illustrates the part of the application that feeds the unlabeled data, the working model, and the resulting predictions:

- A set of built-in connectors access the unlabeled data inputs,
- A series of cleansing, formatting, and joining operations that prepare the data for prediction;
- A series of operations invoking and managing third-party ML platforms and libraries:
- Operating the model to generate the predictions;
- A set of model updating operations;
- Operations to store the predictions;
- Connectors to make the predictions available to application(s).

As on the modeling side, the Ascend platform provides all the constructors for ML practitioners and data engineers to design and deploy these operations, and automates every aspect of actually running them.

> *The Ascend Platform easily connects with modern data feeds, ML platforms, data warehouses and data lakes, and APIs.*

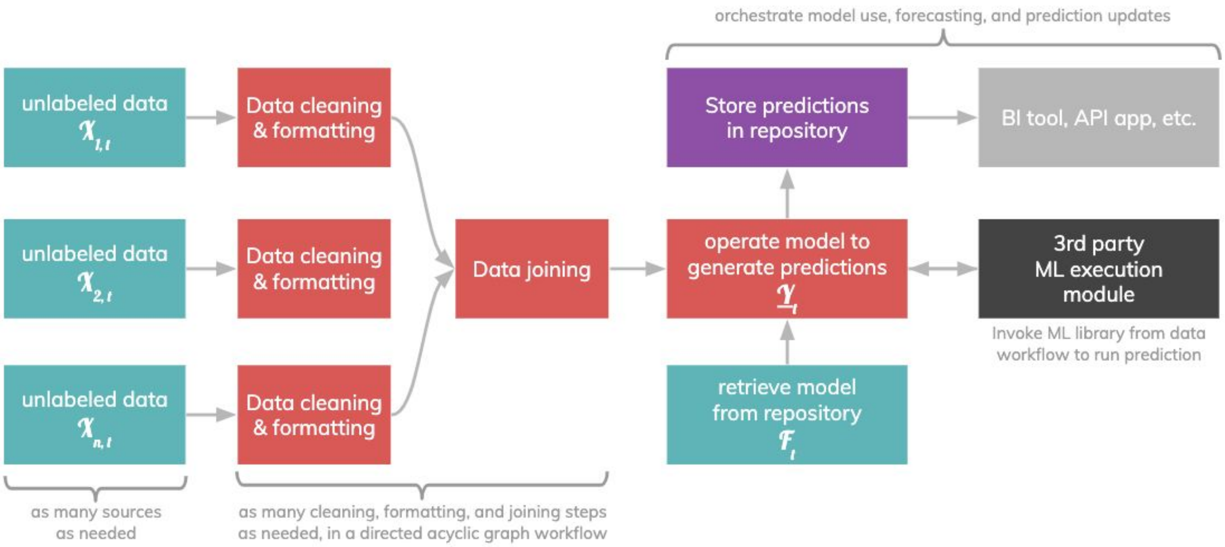
orchestrate model use, forecasting, and prediction updates
unlabeled data $X_{1,t}$
unlabeled data $X_{2,t}$
unlabeled data $X_{n,t}$
Data cleaning & formatting
Data cleaning & formatting
Data cleaning & formatting
Data joining
Store predictions in repository
BI tool, API app, etc.
operate model to generate predictions Y_t
3rd party ML execution module
Invoke ML library from data workflow to run prediction
retrieve model from repository F_t
as many sources as needed
as many cleaning, formatting, and joining steps as needed, in a directed acyclic graph workflow

Figure 3

What do you recommend next?

The emergence of dynamic machine learning techniques like incremental learning demands a new level of sophistication in the management of the data life cycle, process automation, and resource orchestration that is uniquely available from Ascend.

> *Enabling the end-to-end orchestration of data flows allows data teams to tap into sources, manage interactions with third parties, and deliver reliable, live data products to application teams across the enterprise.*

Consider examples like Twitter's 70 thousand API calls per second, or Google's 63 thousand searches per second. Adjusting user preference profiles from each interaction is critical for the business model, yet actually storing each and every interaction for months or years is not feasible or useful. Such scenarios are becoming increasingly commonplace, making autonomous data platforms like Ascend a critical staple for every data team.

We encourage data science and engineering teams that are working with such techniques to free up their time to focus on the actual data science of these sophisticated models. With Ascend, they can reduce the implementation time of the data life cycle to mere hours or days. While the platform is entirely self-service SaaS, we recommend working with our team for a trial:

1. Choose a current problem in your business to do a trial on the platform;
2. Identify the class of dynamic ML model you will use, based on a static data set;
3. Identify the sources of your modeling data, and run-time unlabeled data;
4. Contact us for next steps:
    - Identify the platform or library to tie into Ascend;
    - Identify the specific method to connect to the data sources;
    - Model each of the steps in the pipelines;
    - Stage the data for consumption of the predictions by an application or BI platform.

You can get started with a trial at www.ascend.io/get-started. We look forward to working with you.

Authors

This project was born out of an internal hackathon at Ascend.io, where we break new ground on the art of the possible and test solutions for problems brought up by our customers, with a focus on data flows, pipelines, autonomous data platforms and operationalizing smart applications and machine learning scenarios.

Written by Michael Leppitsch and Jin Huang with contributions and reviews by Andy Wang, Siddharth Panicker, and Tom Weeks.

Please contact us to apply these techniques and productionize your machine learning solution.

Thanks to the entire Ascend.io team for inspiration and guidance, and building the most amazing data automation platform ever!

About Ascend

Ascend provides the world's first Autonomous Dataflow Service, enabling data engineers to build, scale, and operate continuously optimized, Apache Spark-based pipelines with 85% less code. Running natively in Microsoft Azure, Amazon Web Services, and Google Cloud Platform, Ascend combines declarative configurations and automation to manage the underlying cloud infrastructure, optimize pipelines, and eliminate maintenance across the entire data lifecycle. For more information about Ascend, visit www.ascend.io.

Additional Reading

Alexander Gepperth and Barbara Hammer: Incremental learning algorithms and applications, ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (27-29 April 2016)

Manikandan Jeeva: Data Streams and Online Machine Learning in Python, self-published on Medium (19 Jan 2019)

Manish Thapliyal: Incremental learning Challenges and applications, self-published on Medium (05 August 2018)

Hafidz Zulkifli: Multivariate Time Series Forecasting Using Random Forest, self-published on Medium, Towards Data Science (31 March 2019)

Ryan Elwell, Member and Robi Polikar: Incremental Learning of Concept Drift in Nonstationary Environments, IEEE Transactions on Neural Networks (October 2011)

Bartosz Krawczyk, Michał Woźniak: One-class classifiers with incremental learning and forgetting for data streams with concept drift, Springerlink Soft Compute (21 October 2014)

Javier J. Sánchez-Medina, Juan Antonio Guerra-Montenegro, David Sánchez-Rodríguez, Itziar G. Alonso-González, and Juan L. Navarro-Mesa: Data Stream Mining Applied to Maximum Wind Forecasting in the Canary Islands, Multidisciplinary Digital Publishing Institute, Sensors (24 May 2019)

Julien Kervizic: Overview of the different approaches to putting Machine Learning (ML) models in production, self-published on Medium (29 April 2019)

Losing, V., Hammer, B., Wersing, H.: Incremental on-line learning: a review and comparison of state of the art algorithms. Neurocomputing 275, 1261–1274 (2017).

Eric Broda: Rethinking the AI / Machine Learning Lifecycle for the Enterprise, self-published on Medium (11 June 2019)