MLOps: Emerging Trends in Data, Code, and Infrastructure

Venture Capital and Startup Perspective

SEQUOIA

MADRONA VENTURE GROUP

AIFUND

Contents

The state of ML Ops With Vin Sharma, GM, ML Edge and Engines

What is MLOps and why is it important? How do top enterprises operationalize Machine Learning in production and are they using all in one platforms or building it themselves? In this white paper, we begin with a Q&A with Vin Sharma, General Manager of Deep Learning Engines, about the evolution of Machine Learning Operations, or MLOps. Then we examine MLOps through the lens of three of the industry's prominent venture capital investors – Sequoia Capital, Madrona Venture Group, and the AI Fund – and take an in-depth look at the portfolio impact MLOps has had on the adoption of machine learning. We examine in detail how MLOps is evolving, the role of Open Source Software, AWS contributions to the world of MLOps, and finally, how startups and AWS work together to enable business-critical deliverables and outcomes.

What are the paths to MLOps Success?

At AWS, we have seen the growing popularity of machine learning platforms through Amazon SageMaker and variety of self-hosted solutions on EC2. This trend suggests that many organizations want to build, train, and deploy ML models in the Cloud, either using unified DevOps environments or a combination of best-of-breed open source and partner-provided tools that achieve their business goals. At one end of the range are organizations experimenting with ML models or just starting to scale their first few deployments. At the other end are businesses that need powerful capabilities to develop and operate sophisticated ML models in production that drive mission-critical business applications and extract unique value from potentially massive amounts of data.

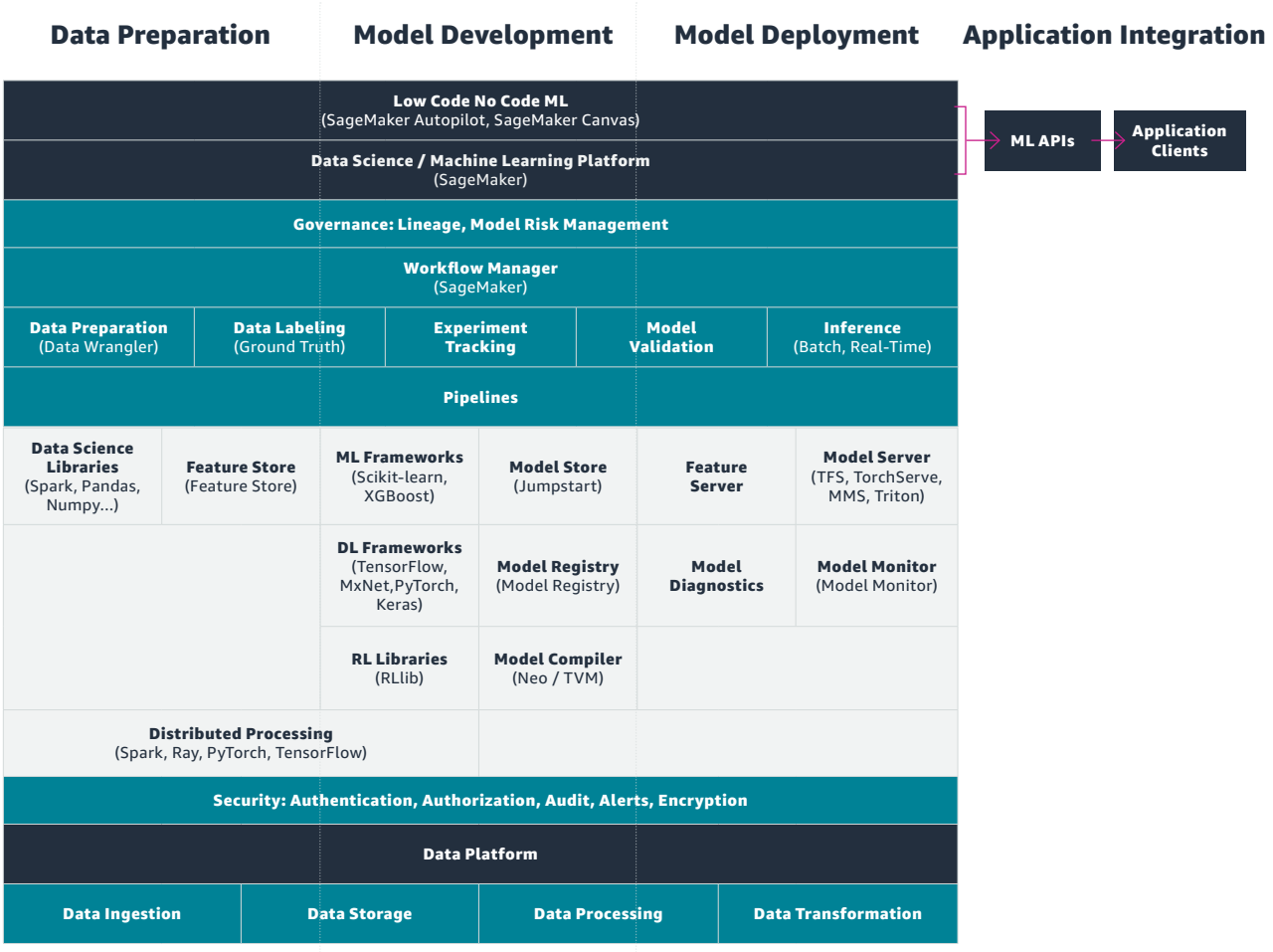
For any organization that wants to generate value from domain-specific business applications or curated data sets, the underlying ML platform and the effort of managing its infrastructure are just a means to an end and potentially represent a heavy burden with no differentiation. Cloud ML platforms provide managed services that are easy to use at a lower cost for such organizations. Even developers without ML expertise can build, train, and deploy ML models quickly to save time and reduce operational expenses. However, these end-to-end services do not meet the needs of many customers with more specialized requirements. Thus, they choose to implement an ML platform composed of best-of-breed products from vendors, open source, homegrown code, and native AWS services.

Ultimately, these organizations want fine-grained control over the full stack and willingly take on the additional complexity and cost of operating a DIY. These businesses want to generate value not only from proprietary applications, models, and data sets but also from the customization of their ML platform. They hope to deliver unique value based on their ML platform software, underlying infrastructure, and ML engineering expertise.

For example, an autonomous vehicle ML platform must:

- Ingest terabytes of vehicle sensor data per hour
- Automatically filter and enrich the data for processing
- Label a representative subset
- Train models on the labeled data
- Test models integrated into the vehicle simulation
- Deploy onboard a test fleet of vehicles
- Continuously monitor the performance and health of models
- Generate alerts, dashboards, and feedback reports which aid model improvement

The unique challenges of operating an ML platform for autonomous vehicles can drive customization of the entire platform software stack, such as this stack using SageMaker:



Building and operating a bespoke ML services stack at scale can seem a like daunting task for an ML engineering team, but many are finding success with open-source frameworks and best-of-breed vendor tools for MLOps.

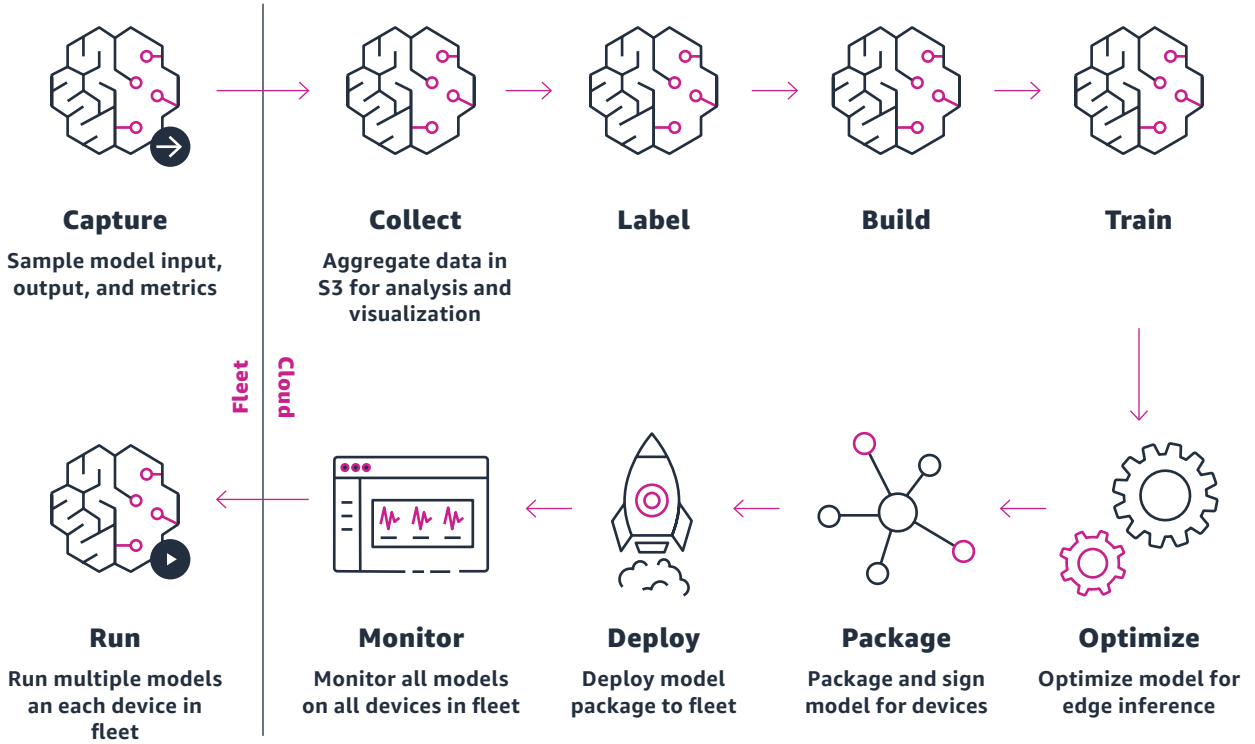
What is MLOps?

Developing ML applications today can be like developing enterprise applications 20 years ago. Multiple personas – data scientists, ML engineers, application developers, and system operators – contend for attention and resources. They use different processes and tools, with researchers and data scientists using multiple frameworks to build and train models.

ML engineers use other tools to optimize and tune production models. Application developers use wildly different processes to develop and deploy applications, and system operators try, often vainly, to manage a common platform infrastructure for all others. The researcher's model with 95% confidence against the training data may fail to predict using production data – neither research nor production know where the mismatch occurred.

The DevOps movement emerged as software engineering practices to enable tighter communication and collaboration between software development and IT operations. Under the DevOps approach, software engineers focus on delivering incremental changes to the customer continuously. Central to the DevOps approach was the notion of near-automatic pipelines for Continuous Integration and Continuous Delivery (CI/CD) that picked up the latest changes, ran unit tests, built the software artifacts, ran integration tests, and finally deployed the change to production in tightly-controlled stages with rapid rollback if needed.

Like DevOps, the Machine Learning Operations (MLOps) approach attempts to formalize near-automatic pipelines for the model lifecycle, from data collection, data preparation, and data wrangling, to model training, model evaluation, and model deployment to model monitoring and then back again for model updates.



Today, data scientists and ML engineers use craftsman tools that treat models like pets – with individualized attention, considerable care, and significant expense. With the MLOps approach, models become like cattle – managed at a scale of tens of millions of models thanks to instrumentation and automation that handles routine operations while surfacing anomalies to human attention. For ML to become ubiquitous and successful in production, an all-new stack is emerging to support robust development, testing, and automated operation of machine learning models at scale.

At AWS we developed a maturity model to help organizations measure their own MLOps readiness:

MLOps Defined:
MOps Maturity Model

	People	Data	Train	Deploy
Initial	• Disconnected data science & IT teams • Limited cross-training	• Ad-hoc data collection and preparation	• Manual training & retraining • No clear path to deployment	• Manual deployment
Repeatable	• Improved collaboration with stakeholders • Shared project goals	• Automated data pipelines	• Defined path for experimentation • Automated training pipelines • Manual Model Validation	• Automated deployment pipelines • Limited monitoring measuring
Reliable	• Cross-functional project teams • Cross-training	• Automated ML Pipelines • Data Governance	• Experiment Management • Automated ML Pipelines • Model Governance • Automated Model Validation	• Automated MLPipelines • Monitoring & Logging (Model, Workload, Pipeline)
Scalable	• Cross-functional project teams • Cross-training	• CI/CD • Policy-as-Code • Configuration-as Code • Automated Validation	• CI/CD • Policy/Config-as-Code • Automated Model Validation • Automated Integration Validation	• CI/CD • Policy/Infra/ Config-as Code • Model Monitoring • Dashboard & Transparency

Many organizations find themselves at the initial phase in at least one category. Most discover miscommunication and confusion around the management of data.

What is the data-centric approach to MLOps?

A data-centric approach to building ML systems emphasizes data management over incremental modeling improvements. ML Systems are always a combination of data and code, but ML teams are increasingly spending their time iterating over data improvements over model improvements.

Synthetic data, data labeling, validation, auditing, and ML monitoring become important parts of the MLOps process. It is becoming more common for a company to start with pre-trained models such as those offered through Amazon SageMaker or those offered through Hugging Face and implement ML monitoring before diving deeper into algorithmic improvements.

The data-centric approach emphasizes data quality and monitoring as an essential part of building production systems and seeks to understand this data with code. Organizations taking on these techniques are more likely to engage in collaboration between teams, are less likely to over-engineer their machine learning code, and are likely to have better outcomes in production.

Will MLOps converge to a modular architecture?

If an organization is committed to a DIY approach to their MLOps stack, the next big strategic question that kicks off hours of debate in the executive conference room is often “build in-house or buy best-of-breed?”. The answer typically ends up as a bit of both.

There are many reasons for this convergence. To build an entire MLOps platform in-house, the organization must possess, acquire, and retain exceptional and expensive talent and then differentiate their in-house implementation from all the goodness available in open-source projects or vendor-provided tools. Inevitably, the resident CTO arrives at a fruit bowl of cherries picked from open-source projects, vendor-supplied proprietary tools, and services from a cloud provider.

This design philosophy depends on the ability to compose the platform architecture with interoperating components. If not thoughtfully constructed, developers may find that these platform architectures fit better in slideware than in software. However, the approach is rational given the vast array of components and open-source or vendor-specific implementations.

What is critical is to ensure these components interoperate. In the abstract, it is easy to train a model with one framework on one ML platform and deploy it for inference in another framework on a different platform. In reality, models are dependent on the model framework.

Developers need tools and model format standards like ONNX to convert them from one framework to another. And if you want to run the model on an edge device, you will need tools to cross-compile the model for the runtime supported on that particular edge hardware. Further out, as you look to monitor the model, you will want a tool to detect model drift that can interoperate with your training and inference platform. Ultimately, the interoperability between the modules becomes the secret key to MLOps success or frustration.

Through the advent of more sophisticated AI use-cases and more advanced data scientists, enterprises do not just want integrated black-box systems anymore. They will want the flexibility to tweak all the components of their model and model-building workflow to produce the most optimal analyses and systems for their specific business needs.

Today, customizing different components of the ML lifecycle can be highly manual, with practitioners primarily taking a DIY approach. Businesses prefer some degree of abstraction from the underlying functions and hardware in each step of the modeling process. They don’t have or want to invest in the in-house resources to custom-build core capabilities like data labeling, experiment tracking, model monitoring, etc. We see organizations opting for best-of-breed solutions from focused vendors to gain additional control over their modeling workflow without needing to be too concerned with what’s going on “under-the-hood.”

Why does MLOps need open-source software development?

A critical element of MLOps is being open about the technology and processes. Openness builds transparency and reproducibility into the machine learning workflow. It is an essential component of building and scaling production machine learning models. It also helps build a community around the technology, enabling other companies and organizations to build, scale, and improve their machine learning models. The best MLOps solutions are also highly involved in the open-source community, where they actively contribute back to the community and help others achieve their goals.

"A critical element of MLOps is being open about the technology and processes."

At AWS, we recognize teams need this flexibility. Amazon Sagemaker is intentionally designed to be both modular and flexible to support teams who wish to work with open-source frameworks. For example, with SageMaker Operators for Kubernetes you manage your Kubernetes cluster to create Amazon SageMaker jobs natively using the Kubernetes API and command line Kubernetes tools, such as kubectl. We support both end-to-end SageMaker and DIY workflows.

How does AWS support open-source machine learning?

AWS is committed to building and investing in open-source MLOps capabilities. We're actively investing in the most critical open-source machine learning tools and frameworks, and we're excited to see the community grow around these projects. We are leading contributors to open-source communities such as Apache MXNet, PyTorch TorchServe, TensorFlow, Deep Java Library, Treelite, and Apache TVM.

AWS aims to be the best place to use these and other open-source machine learning tools. AWS builds, optimizes, tests, and delivers open-source machine learning frameworks (Apache MXNet, PyTorch, TensorFlow, TorchServe, etc KubeFlow), containers, and images (DLC, DLAMI). The AWS distributions of open-source software include versions of ML frameworks optimized for EC2 compute, S3 storage, and ENA/EFA networking, scanned and patched for security vulnerabilities, and built and tested on accelerated instances. As a result,thousands of customers train, run and orchestrate models using the latest features of machine learning frameworks on the newest AWS accelerated instances.

> ***"AWS aims to be the best place to use these and other open-source machine learning tools."***

AWS supports customers who want to operate machine learning models on AWS infrastructure while maintaining control over integration via open-source APIs, control over cost through DIY management, and control over the stack through customization. AWS customers can integrate their model development and deployment workflows with open-source APIs implemented in frameworks such as TensorFlow or PyTorch, orchestrate their workflows using home-grown control systems or open-source toolkits such as KubeFlow. They can customize the combination of frameworks, tools, device drivers with proprietary components of their own without being locked into any platform services. Customers need only to rely on AWS to deliver open-source machine learning software version as a matched set that runs well on AWS infrastructure, packaged as easy-to-consume AMIs or containers.

Today, the AWS distributions of open-source machine learning software include frameworks such as TensorFlow, PyTorch, Apache MXNet, JAX, DGL; tools and libraries integrated into frameworks including compilers such as TVM and XLA, model servers such as PyTorch TorchServe and TensorFlow Serving; and ML workflow orchestration toolkits such as Kubeflow and Ray. AWS customers get current versions of open-source frameworks and can use the latest features and fixes, optimized, patched, built, and tested with relevant EC2 instances. That lets the customers get the latest hardware features and price-performance benefits. This is packaged and distributed as images and containers for easy deployment through AWS services such as EC2, EKS, SageMaker, and ECS, and for optional customization as "Bring Your Own Containers."

Why does MLOps need vendors as well as open-source software?

Not surprisingly, we see a proverbial Cambrian explosion in MLOps today as nearly every application is utilizing machine learning somehow. Many startups develop specialized tools that implement highly efficient methods to perform specific tasks in the model lifecycle. Focuses range from training large-scale language models, optimizing models for inference on edge devices, detecting bias and model explainability, and some on observability and monitoring. As businesses from startup to enterprises are moving their intelligent application and related ML models into production, the AI/ML requires a diverse ecology of tool-makers with a significant opportunity to provide mission-critical functionality core to MLOps success.

SEQUOIA

By Sonya Huang, Partner, Sequoia Capital

Sequoia's Investment Thesis

We believe the rise of data and autonomous software is one of the most important paradigm shifts that will shape the technology landscape and broader enterprise market over the next decade. Just as every great company is a software company today, in ten years we believe every great company will be machine learning driven. Proprietary data is a known super power and the organizations who best harness it with ML will capture disproportionate economic value over the next decade. As a result, there is a race in every enterprise to invest in data science and machine learning.

> ***"Just as every great company is a software company today, in ten years we believe every great company will be machine learning driven."***

While machine learning methods are increasingly popularized and well understood, most ML development is really only accessible in "toy" or local environments. Operationalizing and deploying these models to production for broader impact is an entirely different beast. The most sophisticated organizations have hand-built their own ML frameworks to make the process of managing ML systems easier, as we've seen at Doordash or Uber, but the vast majority of organizations haven't invested in operationalizing their ML systems. The end result is projects take very long to reach production, or never see the light of day.

We believe the ML ecosystem resembles the software ecosystem in the 1990s and early 2000s, before DevOps as a discipline took over and the software development stack professionalized and coalesced around systems like GitHub, Atlassian, and Datadog. ML is now having its "DevOps" moment, where technology companies realize they need to operationalize their ML stacks so that their Data and ML teams can develop with maximum impact. Everybody has seen the case studies out of Uber for example about how Michelangelo democratized and drastically inflected ML development across all of Uber. Our investment thesis is that data/machine learning development is the new software development, and we should expect the DS/ML development stack to mature and spawn interesting new projects and investment opportunities over the next decade.

How are you viewing companies that build modular ML systems vs all in one solutions?

Modularity is key. Production ML systems serve models that are as “core” to an enterprise as it gets. Having the most important models locked into a proprietary vendor that doesn’t integrate well with other solutions simply does not work for ambitious ML organizations.

If you believe that ML development is the new software development, then the egalitarian Linux philosophy very much applies to ML systems. Each component needs to perform its role well - be that a feature store or monitoring solution. As such, we should expect to see modular ML systems win by integrating well into the rest of the stack.

That said, modularity doesn’t preclude bundling if you can do multiple things well. We have seen companies serve multiple parts of the ML stack very well. Hugging Face is an example of a company that has excelled at multiple parts of the ML stack - from finding models, to training, to deployment. But what’s key is that a customer has the option to integrate Hugging Face with their own stack if they choose, or just use it for a single component. That requires Hugging Face to stay on its toes and win on product in every sub-category where it competes.

Do these modular systems become all in one in the long term?

Again, I would look at how the software development stack evolved, which I believe is an apt analogy for how the machine learning landscape will develop. As vendors like Atlassian and Datadog and GitHub grew, they took on more and more adjacent functionality. But there is no single “all in one” software development platform – customers still want to pick and choose a mix of vendors that are best suited for each purpose – it’s simply too large of a scope for a single vendor to deliver well.

I believe a similar dynamic will play out in machine learning. Ambitious vendors will extend their functionality out beyond their initial wedge – the ML serving companies may get into ML monitoring, for example – but we should see a handful of winners. The best companies will strike a balance. They execute incredibly well on their wedge, have the ambition to extend into adjacencies, and they aren’t overly naive about the scope of their project.

Why is open source important/interesting as part of that thesis?

First, we believe that open source is the right product "packaging" for developer audiences. Open source allows developers to try the product before they buy, to peel back the code to see what's inside if desired, and to build for core production use cases without fear about getting locked into a vendor that could disappear. That is why open source has worked so well for software development, and the same principles apply to machine learning systems.

Second, we believe that open source is necessary because of the pace of development in the ML field. The feats you can accomplish with machine learning are developing at a breakneck pace – and ML systems need to keep pace. Open source can speed up the pace of innovation when multiple organizations are involved, and we see many organizations' ML teams contributing back to open source projects. Every company is in a race to make the most out of data and ML, but developers are willing to share the tooling they are using to make these efforts successful and mutually benefit from each other's contributions, which is delightful to see.

"All of this means that open-source ML systems is a particularly fertile ground for new investment opportunities."

Finally, we are interested in open source companies because the business model around open source is really starting to work as a business model. Companies like MongoDB, Confluent, Hashicorp, and GitLab have proven that you can build large and interesting commercial businesses while providing a lot of value for free in the open source. We have also seen the formalization of product led growth as the predominant GTM model in developer-led businesses, and the proliferation of PLG best practices in helping these companies go to market, meaning that companies are getting increasingly good at turning bottoms-up product usage into revenue through data science and new organizational structures (such as developer relations and "growth" and self-serve teams).

All of this means that open-source ML systems is a particularly fertile ground for new investment opportunities.

Who are you investing in?

We've been fortunate to partner with a number of young companies that are quickly becoming category leaders and forging new standards in the "operational ML stack," such as:

Tecton. Tecton offers a key piece of the ML stack, a feature platform, designed to operate and manage the data flows for operational ML applications. Tecton was built by the creators of Uber's ML platform. At Uber, Mike del Balson and Kevin Stumpf saw firsthand the impact that Michelangelo had on democratizing machine learning. They founded Tecton with the mission of bringing machine intelligence to every production application. Tecton's product allows data scientists and engineers to manage feature definitions as code, orchestrate pipelines to continually calculate fresh feature values, build high-quality training data sets, and serve features in production for real-time inference. Tecton enables teams to deploy ML applications in days instead of months, and currently serves customers ranging from leading technology and Fortune 50 companies to a wide range of start-ups. Tecton's open source offering, Feast, is the leading open source feature store.

dbt Labs. dbt Labs allows companies to transform data in-warehouse, using best practices from software engineering (modularity, testing, etc). dbt Labs was founded in 2016 by Tristan Handy, a former analytics practitioner who experienced firsthand just how bad data tooling was for analysts. Since 2016, dbt Labs has been on a mission to help analysts create and disseminate organizational knowledge. dbt Labs pioneered the practice of analytics engineering, built the primary tool in the analytics engineering toolbox, and has been fortunate enough to see a fantastic community coalesce to help push the boundaries of the analytics engineering workflow. Today there are 9,000 companies using open-source dbt every week, 25,000 practitioners in the dbt Community Slack, and 1,800 companies paying for dbt Cloud.

Hugging Face is the central hub where AI practitioners go to download the latest transformer models, and where AI research institutions and open source contributors upload their models for maximum distribution and impact. There are currently 47K models and 5K datasets in Hugging Face's open source repository, including GPT-2 and various iterations of BERT. Hugging Face has cultivated a vibrant community of developers and data scientists in the Natural Language Processing space – and they are now seeing their models and community move into different modalities like speech and vision.

MADRONA VENTURE GROUP

By Tim Porter, Managing Director & Jon Turow, Partner

What is your investment thesis for operational ML systems?

> ***"Applications will evolve (or be replaced) by ML-powered 'intelligent applications' that learn continuously from historical and real-time data."***

At Madrona Venture Group, the single biggest theme we have been investing in over the last 5+ years is the evolution of nearly every application into an intelligent application. We define application intelligence as the process of using machine learning models embedded in applications that use both historical and real-time data to build a continuous learning system. These learning systems solve a business problem in a contextually relevant way – better than before, and they typically deliver rich information and insights that are either applied automatically or leveraged by end-users to make superior decisions. To build and scale intelligent applications into production, organizations need operational ML systems – MLOps products and tools. As such, we have not only been investing in "finished" intelligent applications, like Highspot in sales enablement or SeekOut in recruiting, but also in "intelligent application enablers" that provide the products and platforms that allow organizations and engineers to successfully build, deploy and manage their ML models and intelligent apps. We began investing in this wave well before this category became known as MLOps. Previous investments in companies like Turi (acquired by Apple), Lattice Data (acquired by Apple) and Algorithmia (acquired by DataRobot) helped pave the way and inform our decisions for future investments.

There are a number of core tenets of our MLOps investment thesis. Certainly, the category has proliferated and now there are defined subcategories with multiple vendors in each. We believe that the hyperscale clouds, with AWS the clear leader, will continue to offer excellent services that will provide great alternatives for many organizations, particularly those that either want (a) quick and easy tools for rapid prototyping and deployment or (b) an end-to-end platform backed by a single vendor.

However, we also think potentially even more organizations – particularly those with the most mission critical workloads that often drive the most usage and spend – will want to take a composable approach of choosing best-of-breed products from a variety of vendors, selectively use open-source software, build on hyperscaler infrastructure, and combine these with their own code that best addresses their business and application needs. Sometimes this is because their data or customers are in multiple places and their application needs to run on multiple clouds and even on-prem. Almost always it's because customers view certain products across the MLOps pipeline as being easier to use, providing deeper features or functionality, and perhaps even more cost effective.

Madrona is excited to continue our investment thesis in MLOps. We feel we are still in the early innings of this massive wave of every application becoming an intelligent application and have only begun to see the potential positive impacts of machine learning. We will continue to look for companies with visionary, customer-focused founders with differentiated approaches for providing key functionality across the MLOps spectrum, who are also savvy about partnering with the hyperscale clouds to best serve customers.

Why is open source important/interesting as part of that thesis?

In general, we feel that having an open source component has become essential to a successful MLOps offering. First, given the customer persona is typically an ML engineer, software engineer and/or data scientist, they have come to expect open source as a means to trial and experiment, and then maintain a degree of control and extensibility once implementing into production. They also value the community that develops around open source projects for validation, trouble shooting and best practices. The best open source offerings provide value to a single engineer and makes her job easier or solves a problem that they personally are facing. From there, open source can spread "bottoms up" through an organization. This often then creates an opportunity for a commercial company to provide the additional team and enterprise features and support that organizations need once broader open source option has occurred, as well as the option of providing a "we'll run it for you managed service." Management teams appreciate that open source is a hedge against vendor lock-in. Investors and customers alike recognize the power of continuing to ride the wave of community innovation, which can often be more rapid and powerful than any single vendor. As investors, we also look to early open source adoption metrics as a key leading indicator as to what project are taking off and solving important customer problems. For all these reasons, we view a robust open source offering as key to any successful company in MLOps, as is true for almost every area of software infrastructure today.

How are you viewing companies that build modular ML systems vs all in one solutions?

We like to invest in companies that provide a modular ML system, with a focused offering that solves an important customer problem in a differentiated way, and can work multi-cloud, meeting customers wherever they are and their data resides. A trend in MLOps has been that companies who become successful in an MLOps subcategory seem to evolve or expand into offering end-to-end solutions. Sometimes we question, however, whether this is in response to customer pull or just broader startup imperialism and ambition. We have seen some of both – if a vendor is providing a strong solution in the "front end" of the ML pipeline (say, labeling, training or model creation), customers might also want deployment and management that can close the loop back to training. However, in many more cases, we see customers want a modular system where they can knit together best-of-breed solutions that best fit their environments and business needs.

Who are you investing in?

WHYLABS

A successful investment we've made in this space is WhyLabs, the leader in ML observability across both model performance and data quality. Our investment started with a belief in the ability and vision of the incredible founding team led by ex-Amazonian Alessya Visnjic. They founded WhyLabs with the goal of equipping every ML team with the tools necessary to operate ML applications responsibly. Once in production, the ML application is prone to failure due to issues like data drift and data quality. WhyLabs equips ML teams with an observability platform that is purpose-built to proactively identify and fix issues that arise across the ML pipeline. With WhyLabs, teams from Fortune 100 enterprises to AI first start-ups are able to eliminate model failures and significantly reduce manual operations, shifting focus to building more and better models. Just like best-of-breed winners like DataDog and New Relic emerged in previous years for application performance management, we believed that ML Observability would produce multiple large winners that provide this functionality cross-platform and integrate with whichever existing pipelines and tools customers choose. We also deeply believed in WhyLabs commitment to open source. They established the open standard for data logging, whylogs, which enables a single ML builder to enable basic monitoring and create value immediately, and then "graduate" to a self-serve SaaS, and ultimately to a much deeper product suite with all of the features needed for enterprise ML teams. We also believed in the company's conviction that to provide true observability, builders need to be able to monitor both model health and data health from a single pane, to truly understand root causes and the interactions between the two sides of this coin.

OctoML

Another successful investment that highlights our MLOps thesis has been OctoML. The company was founded out of the University of Washington by Prof. Luis Ceze and an amazing group of co-founders. It is based on an open-source project called Apache TVM that the OctoML founding team created. TVM takes models built on any leading framework and, by using ML-based optimizations, compiles, runs, and accelerates them on any hardware.

While OctoML continues to be core contributors of TVM, the project has blossomed widely and grown to include a broad consortium of industry leaders including AWS, Microsoft, Intel, AMD, Arm, Nvidia, Samsung and many more. OctoML's core mission is to bring DevOps agility to ML deployment, automating the process of turning models into highly optimized code and packaging them for the cloud or edge. This reduces ML operating expense, obviates the need for specialized teams or infrastructure, accelerates time to value for models, and offers hardware choice flexibility to meet SLAs. Here again, a core tenet of the investment thesis started with open source and the community innovation it leverages. Further, it illustrates the core belief that intelligent applications will be deployed everywhere – across clouds and end point types – from beefy cloud GPUs to performance constrained devices on the edge.

Do these modular systems become all in one in the long term?

In short, we do not view the MLOps world ultimately converging into monolithic end-to-end platforms. There is large demand, and a rich opportunity, for new companies to provide these critical cross-platform products and services. An interesting related question is whether MLOps continues to exist as its own category, or whether it converges simply into DevOps, as nearly every application becomes a data-centric intelligent application. Madrona's belief is these two spaces will begin to largely converge, but certain unique customer needs specific to MLOps will continue to persist.

AIFUND

By Ryan Cunningham, Senior Associate Builder

AI Fund's investment thesis

Artificial intelligence is the new electricity. As electricity revolutionized the world and transformed nearly every industry, so AI is poised to radically reshape the global economy and our place in it. This emerging technology creates unparalleled opportunities to create and augment businesses solving difficult problems in a variety of domains. AI Fund operates as a venture studio that partners with founders to build such companies from scratch. We are particularly excited about companies working in human capital, health and wellness, next-generation enterprise, and MLOps.

Human capital: Developing and enabling talent, ensuring access to jobs, and providing training and education in all sectors and levels.

Health and wellness: Building infrastructure and tools that enable people to live healthier and longer lives.

Next-generation enterprise: Improving the ways people work and automating enterprise systems and processes.

MLOps: Making ML systems easier to build, deploy and maintain in a variety of industries and applications.

When it comes to investing in MLOps systems, we have a particular interest in companies that take advantage of data-centric AI development principles. For many applications, an off-the-shelf model will be sufficient, and the developers' time is best spent tuning the data rather than the model. We have seen taking a data-centric approach resulting in much faster progress in multiple industry verticals. However, there still remains a significant market gap in tools to help developers collect, manage, track, and systematically engineer the data fed to an AI system.

"We have seen taking a data-centric approach resulting in much faster progress in multiple industry verticals."

While having more data can be very helpful, huge amounts of data aren't always necessary. What's more important is having good data, data that represents the real world your model will be operating in. This is especially important in industries where large datasets simply don't exist. The best progress will be made through tools that can bring a system to required performance thresholds by making sure the dataset—even as small as 50 examples—is carefully chosen to show the AI system exactly what we want it to learn, while looping in new examples from the real world to keep the system current.

Why is open source important/interesting as part of that thesis?

Open source accelerates adoption of new technology by virtuously improving it with tighter feedback loops from customers and developers alike. The faster the technology matures, the more market use cases it will find, and the faster mass adoption can occur. I've worked with many companies that have built fantastic software for internal use cases that have gone on to open-source it with great results.

While some companies might worry they risk defensibility by contributing to open-source products instead of an internal 'secret sauce,' the benefits usually outweigh this:

1. **Vetting new vendors**
   It's difficult for customers to evaluate closed source products. Customers have to take vendors at their word and, in the end, they usually make a choice based on some combination of brand name and price. With open source, customers have full visibility into the code, contributors, and ecosystem supporting the product. This increases their trust in the vendor, especially for startups that typically don't have the brand recognition.

2. **Security**
   In the event of a security breach, customers of closed source products have to rely on the vendor's in-house cybersecurity team, which will likely result in a delayed response to the breach. Unless you have a preternaturally proactive security team, you have a higher likelihood of spotting a zero-day threat with a community of contributors who alert each other to potential danger. More often than not, open source is even more secure than closed source for this reason.

3. **Talent preference**
   The best productivity software to use is the one that the whole team buys into. Typically speaking, engineers prefer to use and contribute to open-sourced products rather than rely on closed products, thanks to the transferability of that skill across different organizations. Thus, a team using open-source software has a higher probability of attracting quality talent already skilled in its use. Creating open source software caters to the preferences of the engineers and developers who will use it, and can spur further usage.

4. **Halo effect**
   There is a halo effect for organizations that contribute actively to open source development. Those companies are seen as value additive to the broader community and that builds goodwill. This is important for both sales and recruiting top talent.

How are you viewing companies that build modular ML systems vs all in one solutions?

One approach is not necessarily better than another, and we work with companies in both camps. It depends more on the target customer and the product's value proposition.

Larger companies can afford to create more customized solutions. These companies benefit from modular systems where they can combine off-the-shelf tools with internal solutions purpose-built to their needs. Of course, doing that means some extra work - they have to spend some time up front integrating these systems, and consistently testing for compatibility as new updates are pushed overtime. But chances are a large enterprise will have distinct needs that a complete external solution simply cannot provide.

For MLOps specifically, a team may only require solutions for parts of the development cycle. A team with a massive backlog of data, for instance, may have a strong need for production-grade labeling, auditing, splits, and integration with new external data as it comes up. But that does not guarantee that they'd make use of the model versioning functionality from an all-in-one tool; they could have a proprietary model architecture that isn't compatible with existing solutions on the market, so a bespoke solution works internally for now.

For smaller teams just starting out, an all-in-one option gives you the ability to get up and running quickly without a large investment of time and resources as you build out your ML stack. In this stage, your number one priority should be finding product-market fit, so unless you have strong reasons otherwise, you might consider this route up front to optimize for speed.

Do these modular systems become all in one in the long term?

No single company can own all the modular systems right now, because everyone is still actively developing the best frameworks to conduct these operations. The MLOps field is nascent, which is exciting, because the more people contribute to these open source dialogues, the faster we will converge on standards for ML development, deployment, and management.

We are still several years away from a single dominant company emerging as the central hub for MLOps product to exist within. In the long run, as MLOps standards formalize, we expect larger players will begin to acquire adjacent products and scale horizontally. From there we may begin to see complete, Atlassian-like platforms take shape.

Who are you investing in?

Landing AI is building an end-to-end platform for computer vision, with applications spanning manufacturing, pharmaceuticals, analysis of aerial imagery, and other sectors.

It used to be that companies needed massive data sets for ML to be effective. This effectively locked out companies that lacked the large datasets common to big tech and internet companies. With data-centric AI, Landing AI is bringing AI to less data-rich companies and making AI more accessible to organizations, particularly in manufacturing, where tiny datasets are the norm.

Manufacturing companies often do not have machine learning engineering expertise in-house, yet these companies stand to benefit enormously from AI in their plants, particularly with computer vision for defect detection. With Landing AI's LandingLens enterprise MLOps platform, companies can easily build, iterate, and operationalize AI-powered visual inspection solutions.

ValidMind is a stealth-mode startup that is facilitating trust, transparency, and visibility around ML models for financial services institutions. The company helps clients better understand the risks in their ML models, streamlines ML model validation processes, and accelerates production deployment.

The financial services industry operates within a complex regulatory environment that requires institutions to have independent review and validation of models before they are deployed to production. A consequence of these regulations is that data science teams and model validation teams operate in silos, using different tools and reporting to different branches of the organization.

Compliance teams need to review ML models prior to deployment for bias and anything else that could negatively impact customers. Currently, validation at most institutions is a manual and tedious process, where data science teams will wait until after they are done building a model to draft the (often lengthy) documentation they need to validate the model. While the model may take only a few weeks to build, this validation process can slow deployment to production by months. The silos block effective collaboration and slow ML model deployment to a crawl.

ValidMind's solution is a central platform for data scientists and compliance teams to collaborate as the model is developed to ensure ML compliance by design, rather than as an afterthought. Using the ValidMind SDK, data scientists can run tests and capture documentation about ML models with a few lines of code; the compliance team can review this information in real-time on the ValidMind platform to flag potential risks and findings that need to be addressed. By the time the ML model is built, validation is a simple sign-off, as opposed to a several months-long process after the fact.

Conclusion

Top organizations are adopting a DevOps-like process for Machine Learning called MLOps to create a durable and competitive advantage. These systems continue to evolve at a breakneck speed where the most essential components are obvious: modularity and interoperability. With its commitment to machine learning innovation, enterprise grade security, and deep collaboration with Open Source, AWS aims to hyper scale MLOps adoption and the top MLOps startups.

Contributors

Rob Ferguson, Artificial Intelligence, AWS

Natalie Heard, Venture Capital Business Development, AWS

Sonya Huang, Partner, Sequoia Capital

Tim Porter, Managing Director, Madrona Venture Group

Jon Turow, Partner, Madrona Venture Group