The Complete Guide to Machine Learning Operations (MLOps)

This guide covers four important areas of Machine Learning Operations

What is it?	ML Infrastructure	ML Automation	Considerations for ML Workflows

This Guide Covers:

We begin with an explanation of how machine learning operations came to be a discipline inside many companies and then cover some of the details around how to best implement MLOps in your organization.

Machine Learning Operations:

What is it? Why do we need it?

Machine learning (ML) is a subset of artificial intelligence in which computer systems autonomously learn a task over time. Based on pattern analyses and inference models, ML algorithms allow a computer system to adapt in real time as it is exposed to data and real-world interactions.

For many people, ML was, until recently, considered science fiction. But advances in computational power, frictionless access to scalable cloud resources, and the exponential growth of data have fueled an increase in ML-based applications. Today, ML has a profound impact on a wide range of verticals such as financial services, telecommunications, healthcare, retail, education, and manufacturing. Within all of these sectors, ML is driving faster and better decisions in business-critical use cases, from marketing and sales to business intelligence, R&D, production, executive management, IT, and finance.

MLOps: Getting from Science to Production

Machine learning is rooted in the realm of data science. For the ML inference model used during runtime to identify a pattern or predict an outcome, data scientists must make sure it is "asking the right questions," i.e., looking for the features that are most relevant to the task at hand. Once data scientists have defined an initial set of features, their next task is to identify, aggregate, clean, and annotate a known data set that can be used to train the model to recognize those features.

Here, the larger the training data set, the better. The data scientists then continue to optimize the model under development through a highly iterative process of training, testing, and tuning.

In addition to data science expertise, developing an ML model also involves considerable IT and infrastructure skills. Huge data sets have to be aggregated, stored, moved, protected, and managed. Training and testing the models requires very high levels of compute capacity and performance.

Thus, one of the first challenges in accelerating the ML development lifecycle is to abstract the infrastructure layer from the data science. In much the same way that DevOps freed developers from infrastructure issues, allowing them to concentrate on application development, a simple and easy-to-use research environment is necessary to allow data scientists to focus on model development rather than infrastructure provisioning and monitoring.

Other challenges are related to an inherent disconnect between data scientists and the engineers who must operationalize the models in production-ready applications. Each group works in its own silo, with its own unique mindset, concepts, processes, and tool stacks. In many cases, the engineering team may have difficulty simply understanding the model handed off to them by the data scientists. And once the model is in production, it is tough for the operations team to understand which metrics and parameters need to be tracked in order to effectively monitor accuracy and performance. It's also not easy to establish the critical feedback loop for the data science team to be able to continue improving the inference model while ensuring that the updated models don't have a negative impact on application performance.

Why is MLOps Important? Closing the Loop with Machine Learning Operations

MLOps (Machine Learning Operations) is a relatively new discipline that seeks to systematize the entire ML lifecycle, from science to production. It started as a set of best practices to improve the communications between data scientists and DevOps teams—promoting workflows and processes that could accelerate the time to market for ML applications. Soon, open source MLOps frameworks began to emerge, such as MLflow and Kubeflow.

Today, MLOps capabilities are considered a key requirement for Data Science and Machine Learning (DSML) platforms. Gartner's "2020 Magic Quadrant for Data Science and Machine Learning Platforms" cites MLOps as a key inclusion criterion, noting that "...[a]s DSML moves out of the lab and into the mainstream, it must be operationalized with seamless integration and carefully designed architecture and processes.

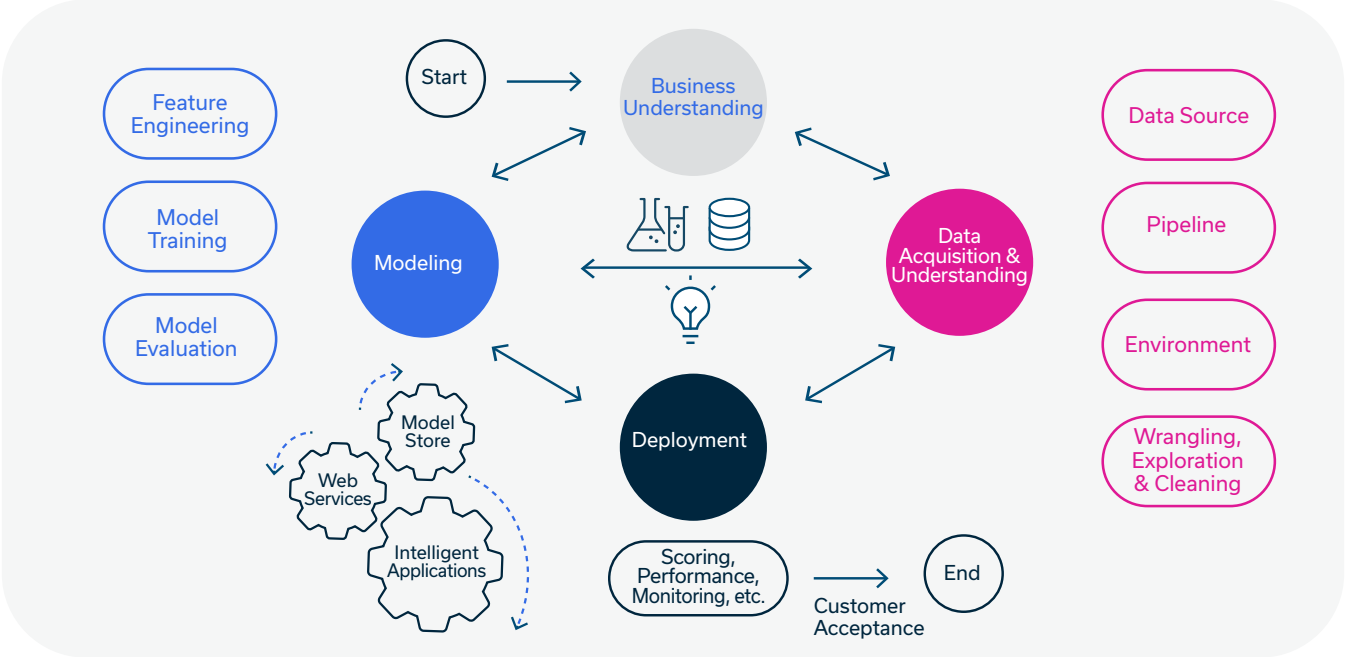
Machine learning operations capabilities should also include explainability, versioning of models and business impact analysis, among others."

As shown in Figure 1 below, the next-generation data science lifecycle breaks down the silos among all the different stakeholders that need to be involved for ML projects to capture business value.

This starts with the modeling and data acquisition activities of the data science team being informed by a clear understanding of the business objectives for the ML application—as well as of the governance and compliance issues that should be taken into account. The MLOps model then ensures that the data science, production, and operations teams work seamlessly together across ML workflows that are as automated as possible, ensuring smooth deployments and effective ongoing monitoring. Performance issues, as well as new production data, are reflected back to the data science team so that they can tune and improve the model, which is then thoroughly tested by the operations team before being put into production. (Source: A report reprint, available to Gartner subscribers only.)

Data Science Lifecycle

Figure 1: MLOps Drives Data Science Success and Value. (Source: Azure)



In short, machine learning operations is the critical missing link that allows IT to support the highly specialized infrastructure requirements of ML infrastructure. The cyclical, highly automated MLOps approach:

- Reduces the time and complexity of moving models into production.
- Enhances communications and collaboration across teams that are often siloed: data science, development, operations.
- Streamlines the interface between R&D processes and infrastructure, in general, and operationalizes the use of specialized hardware accelerators (such as GPUs), in particular.
- Operationalizes model issues critical to long-term application health, such as versioning, tracking, and monitoring.
- Makes it easier to monitor and understand ML infrastructure and compute costs at all stages, from development to production.
- Standardizes the ML process and makes it more auditable for regulation and governance purposes.

Machine Learning Infrastructure

Components of Effective Pipelines

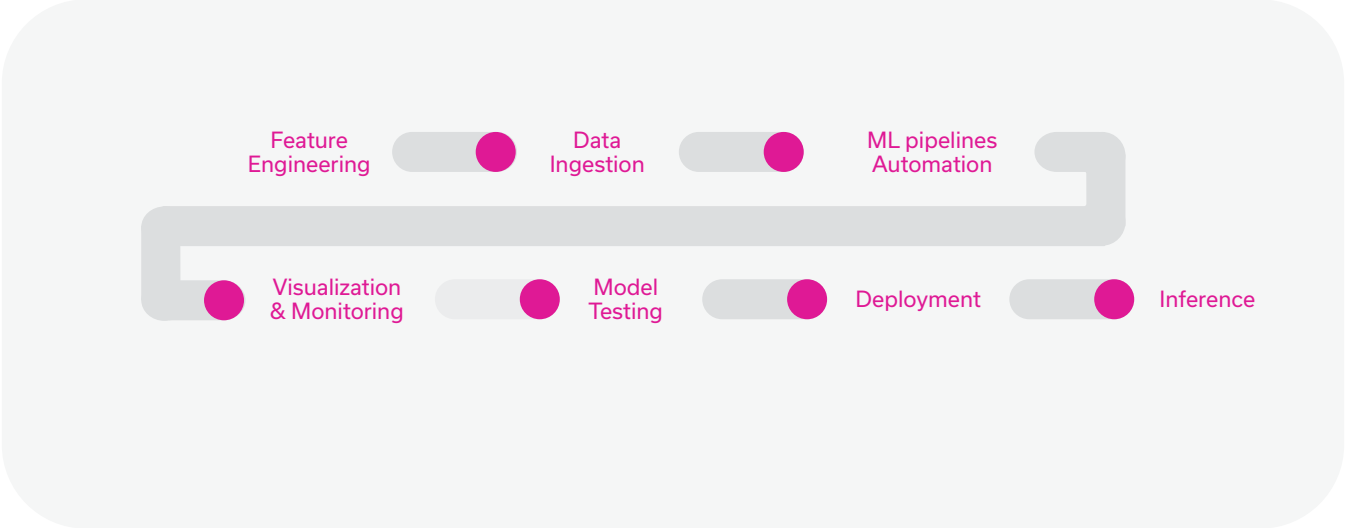
Machine learning (ML) infrastructure is the foundation on which machine learning models are developed and deployed. Because models differ between projects, machine learning infrastructure implementations also vary. However, there are core components any machine learning infrastructure needs to be fully functional.

This article explains these components, and reviews important aspects you should consider when creating your machine learning infrastructure.

What Is Machine Learning Infrastructure?

Machine learning infrastructure includes the resources, processes, and tooling needed to develop, train, and operate machine learning models. It is sometimes referred to as AI infrastructure or a component of MLOps.

ML infrastructure supports every stage of machine learning workflows. It enables data scientists, engineers, and DevOps teams to manage and operate the various resources and processes required to train and deploy neural network models.



Machine Learning Infrastructure Development: The Building Blocks

To understand machine learning infrastructure it helps to first understand its components.

Model Selection

Machine learning model selection is the process of selecting a well-fitting model. It determines what data is ingested, what tools are used, which components are required, and how components are interlinked.

Data Ingestion

Data ingestion capabilities are at the core of any machine learning infrastructure. These capabilities are needed to collect data for model training, application, and refinement.
Data ingestion tools enable data from a wide range of sources to be aggregated and stored without requiring significant upfront processing. This allows teams to leverage real-time data and to effectively collaborate on the creation of datasets.

ML Pipelines Automation

There are numerous tools available that can automate machine learning workflows according to scripts and event triggers. Pipelines are used to process data, train models, perform monitoring tasks, and deploy results. These tools enable teams to focus on higher-level tasks while helping to increase efficiency and ensure the standardization of processes.

When developing your infrastructure, you can create toolchains from scratch by individually integrating and orchestrating tools. You can also adopt pre-built or self-contained pipelines, such as MLflow Pipelines or Apache Airflow. Learn more in our guide about machine learning automation.

Visualization and Monitoring

Machine learning visualization and monitoring are used to gain perspective on how smoothly workflows are moving, how accurate model training is, and to derive insights from model results.

Visualizations can be integrated at any point in machine learning workflows to enable teams to quickly interpret system metrics. Monitoring should be integrated throughout.

When incorporating visualization and monitoring into your machine learning infrastructure, you need to ensure that tools ingest data consistently. If solutions do not integrate with all relevant data sources you will not get meaningful insights. Additionally, you need to keep in mind the resources that these tools require. Make sure that you are choosing solutions that work efficiently and do not create resource conflicts with your training or deployment tools.

Model Testing

Testing machine learning models requires integrating tooling between training and deployment phases. This tooling is used to run models against manually labeled datasets to ensure that the results are as expected. Thorough testing requires:

- Collection and analysis of both qualitative and quantitative data
- Multiple training runs in identical environments
- The ability to identify where errors occurred To set up machine learning testing, you need to add monitoring, data analysis, and visualization tools to your infrastructure. You also need to set up automated creation and management of environments. During set up you should perform integration tests to ensure that components are not causing errors in other components or negatively affecting your test results.

Deployment

Deployment is the final step that you need to account for in your architecture. This step packages your model and makes it available to development teams for integration into services or applications.

If you are offering machine learning as a Service (MLaaS), it may also mean deploying the model to a production environment. This deployment enables you to take data from and return results to users. Typically, MLaaS involves containerizing models. When models are hosted in containers, you can deliver them as scalable, distributed services regardless of end environment.

Inference

In the deployment stage, it is important to evaluate deep learning frameworks and select those that best fit your needs for ongoing inference of new data. You will need to select and optimize the framework that meets your performance requirements in production without exhausting your hardware resources. For example, a computer vision model running in a self driving car must perform inference at millisecond speeds, while taking into account the hardware available on board the car.

The process of moving models between frameworks, according to production needs, has been made easier in recent years with the development of universal model file formats. These formats enable you to more easily port models between libraries, such as the Open Neural Network eXchange (ONNX).

Key Considerations for Infrastructure that Supports ML

When creating your machine learning infrastructure there are several considerations that you should keep in mind.

Location

Pay attention to where your machine learning workflows are being conducted. The requirements for on-premises operations vs cloud operations can differ significantly. Additionally, your location of choice should support the purpose of your model.

In the training stage, you should primarily focus on cost considerations and operational convenience. Security and regulations relating to data are also important considerations when deciding where to store training data. Will it be cheaper and/or easier to perform training on premises or in the cloud? The answer may vary depending on the number of models, the size and nature of data being ingested, and your ability to automate the infrastructure.

In the inference stage, the focus should be on balancing between performance and latency requirements vs available hardware in the target location.

Models that need a fast response or very low latency should prioritize local or edge infrastructures, and be optimized to run on low-powered local hardware. Models that can tolerate some latency can leverage cloud infrastructure, which can scale up if needed to run "heavier" inference workflows.

Compute Requirements

The hardware used for machine learning can have a huge impact on performance and cost. Typically, GPUs are used to run deep learning models, and CPUs are used to run classical machine learning models. In some cases, the traditional ML uses large volumes of data, it can also be accelerated by GPUs using frameworks like Nvidia's RAPIDS.

In both cases, the efficiency of the GPU or CPU for the algorithms being used will affect operating and cloud costs, hours spent waiting for processes to complete, and by extension, time to market..

When building your machine learning infrastructure you should find the balance between underpowering and overpowering your resources. Underpowering may save you upfront costs but requires extra time and reduces efficiency. Overpowering ensures that you aren't restricted by hardware but means you're paying for unused resources.

Network Infrastructure

The right network infrastructure is vital to ensuring efficient machine learning operations. You need all of your various tools to communicate smoothly and reliably. You also need to ingest and deliver data to and from outside sources without bottlenecks. To ensure that networking resources meet your needs, you should consider the overall environment you are working in. You should also carefully gauge how well networking capabilities match your processing and storage capabilities. Lightning fast network speeds aren't helpful if your processing or data retrieval speeds lag.

Storage Infrastructure

An automated ML pipeline should have access to an appropriate volume of storage, according to the data requirements of the models. Data-hungry models may require Petabytes of storage. You need to consider in advance where to locate this storage – on-premises or on the cloud.

It is always preferred to colocate storage with training. For example, you can run training using TPUs on Google Cloud, and have data stored in Google Cloud Storage, which is infinitely scalable. Or you could run training on local NVIDIA GPUs and use a large-volume, high performance, fast distributed file system to store data locally. If you create a hybrid infrastructure, plan data ingestion carefully to prevent delays and complexity in training

Data Center Extension

If you are incorporating machine learning into existing business operations you should work to extend your current infrastructure. While it may seem easier to start from scratch, this often isn't cost-efficient and can negatively affect productivity.

A better option is to evaluate the existing infrastructure resources and tooling you have. Any assets that are suited to your machine learning needs should be integrated. The exception is if you are planning to retire those assets soon. Then, you are better off adopting new resources and tools.

Security

Training and applying models requires extensive amounts of data, which is often valuable or sensitive. For example, financial data or medical images. Big data is a big lure for threat actors interested in using data for malicious purposes, like ransoming or stealing data in black markets.

Additionally, depending on the purpose of the model, illegitimate manipulation of data could lead to serious damages. For example, if models used for object detection in autonomous vehicles are manipulated to cause intentional crashes.

When creating your machine learning infrastructure you should take care to build in monitoring, encryption, and access controls to properly secure your data. You should also verify which compliance standards apply to your data. Depending on the results, you may need to limit the physical location of data storage or process data to remove sensitive information before use.

Machine Learning Automation

Speeding Up the Data Science Pipeline

Machine learning automation enables data scientists to automate the creation of machine learning processes. Without machine learning automation, the ML process can take months, from data preparation, through training, until actual deployment.

Machine learning automation tools were created to help speed up the machine learning pipeline. In some cases, this means automating only specific tasks, like model selection. In other cases, it means automating your entire machine learning operations process.

In this article we discuss the potential and possibilities of automating machine learning pipelines.

Furthermore, these big investments in data and AI projects are successful only 15% of the time. As a result, for many readers, delivering an effective AI app in one day sounds like an impossible pipe dream.

Apart from math, data analysis is the essential skill for machine learning. The ability to crunch data to derive useful insights and patterns form the foundation of ML. Like math, not every developer has the knack to play with data. Loading a large dataset, cleansing it to fill missing data, slicing and dicing the dataset to find patterns and correlation are the critical steps in data analysis.

Learn more in our article about the machine learning workflow.

What Is AutoML?

Automated machine learning (AutoML) is a process that automatically performs many of the time-consuming and repetitive tasks involved in model development. It was developed to increase the productivity of data scientists, analysts, and developers and to make machine learning more accessible to those with less data expertise.

Challenges of Machine Learning Pipelines: The Need for AutoML

In ML, data scientists first start with a problem statement and a dataset. The data is analysed and cleaned, a metric of performance is decided on and then a few models which might work on the dataset, according to the human intuition, are experimented with. There is a lot of feature engineering and fine tuning involved before we finally reach an acceptable model.

A recent Gartner survey reported that it takes on average four years to get an AI project live. For 58% of businesses it takes two years to get to the piloting stage.

Apart from math, data analysis is the essential skill for machine learning. The ability to crunch data to derive useful insights and patterns form the foundation of ML. Like math, not every developer has the knack to play with data. Loading a large dataset, cleansing it to fill missing data, slicing and dicing the dataset to find patterns and correlation are the critical steps in data analysis.

Why is Automated Machine Learning Important?

Machine learning automation is important because it enables organizations to significantly reduce the knowledge-based resources required to train and implement machine learning models. It can be used effectively by organizations with less domain knowledge, fewer computer science skills, and less mathematical expertise. This reduces the pressure on individual data scientists as well as on organizations to find and retain those scientists.

AutoML can also help organizations improve model accuracy and insights by reducing opportunities for bias or error. This is because machine learning automation is developed with best practices determined by expert data scientists. AutoML models do not rely on organizations or developers to individually implement best practices.

Machine learning automation lowers the requirements for entry to model development, allowing industries that were previously unable to leverage machine learning to do so. This creates opportunities for innovation and strengthens the competitiveness of markets, driving advancement.

Learn more in our article about machine learning infrastructure.

What Tasks Should You Automate?

While not everything in machine learning can be automated, many processes and steps are iterative, especially in model training. These iterative steps are ideal for automation.

Hyperparameter Optimization

Hyperparameters are values that are defined before a model is trained. These values govern model training and impact the end accuracy of the model. Example hyperparameters include learning rate, activations functions, number of hidden units and layers, and the number of epochs.

To improve models, you need to optimize your hyperparameters. This is typically done through the application of search algorithms, such as random search, grid search, or Bayesian optimization. This application is what can be automated. There are multiple individual tools available for this, including SigOpt, Katib, Eclipse Arbiter, Tensorflow Vizier, and Spearmint.

Model Selection

In machine learning, model selection is the process of selecting the right candidate model for your machine learning implementations. It is based on model performance, complexity and maintainability, as well as what resources you have available. The model selection process is what determines the structure of your model development pipeline.

Automating model selection is done in much the same way as hyperparameter optimization. This is because both are essentially seeking the same end goal. The difference is that model selection may also include more extensive filtering through methods like Akaike Information Criterion (AIC) or Bayesian Information Criterion (BIC).

Feature Selection

Machine learning feature selection is a process that refines how many predictor variables are used in a machine learning model. The number of features that your model includes directly affects how difficult it is to train, understand, and run.

When automating feature selection testing is scripted to use one or more of a variety of algorithmic methods, such as wrapper, filter, or embedded. After performing your feature selection tests, the one with the lowest error rate or proxy measure is selected.

Data Preprocessing

Data preprocessing involves cleaning, encoding, and verifying data before use. Automated tasks can perform basic data preprocessing before performing hyperparameter and model optimization steps. This type of machine learning automation typically includes the detection of column types, transformation into numerical data, and handling missing values.

Advanced preprocessing can also be performed. This includes automation of feature selection, target encoding, data compression, text content processing, feature generation or creation, and data cleaning.

Transfer Learning and Pre-Trained Models

In machine learning, transfer learning involves taking models that have already been trained on a similar data set and using it for your machine learning initiative. Generally, this model is used as a base and then further trained to match your exact needs.

In terms of machine learning automation, this initial model can be trained in the same way as your end model while you are collecting or preparing datasets for the final model. This can save significant time, especially if you do not need a highly accurate model.

Search for Network Architecture

You can also move beyond preparation and model selection processes, extending to the dynamic development of machine learning algorithms. New developments have allowed some automation of network architectures searches.

In particular, the neural architecture search (NAS) method is being explored and applied to problems based on gradient descent, reinforcement learning, and evolutionary algorithms. This method has already been integrated into several tools including AutoKeras, an open-source library, and the results integrated into several projects, including autonomous vehicles.

Machine Learning Workflow

Streamlining Your ML Pipeline

Machine learning workflows define which phases are implemented during a machine learning project. The typical phases include data collection, data pre-processing, building datasets, model training and refinement, evaluation, and deployment to production. You can automate some aspects of the machine learning operations workflow, such as model and feature selection phases, but not all.

While these steps are generally accepted as a standard, there is also room for change. When creating a machine learning workflow, you first need to define the project, and then find an approach that works. Don't try to fit the model into a rigid workflow. Rather, build a flexible workflow that allows you to start small and scale up to a production-grade solution.

Understanding the Machine Learning Workflow

Machine learning workflows define the steps initiated during a particular machine learning implementation. Machine learning workflows vary by project, but four basic phases are typically included.

Gathering Machine Learning Data

Gathering data is one of the most important stages of machine learning workflows. During data collection, you are defining the potential usefulness and accuracy of your project with the quality of the data you collect.

To collect data, you need to identify your sources and aggregate data from those sources into a single dataset. This could mean streaming data from Internet of Things sensors, downloading open source data sets, or constructing a data lake from assorted files, logs, or media.

Data Pre-Processing

Once your data is collected, you need to pre-process it. Pre-processing involves cleaning, verifying, and formatting data into a usable dataset. If you are collecting data from a single source, this may be a relatively straightforward process. However, if you are aggregating several sources you need to make sure that data formats match, that data is equally reliable, and remove any potential duplicates.

Building Datasets

This phase involves breaking processed data into three datasets—training, validating, and testing:

- Training set—used to initially train the algorithm and teach it how to process information. This set defines model classifications through parameters.
- Validation set—used to estimate the accuracy of the model. This dataset is used to finetune model parameters.
- Test set—used to assess the accuracy and performance of the models. This set is meant to expose any issues or mistrainings in the model.

Training and Refinement

Once you have datasets, you are ready to train your model. This involves feeding your training set to your algorithm so that it can learn appropriate parameters and features used in classification.

Once training is complete, you can then refine the model using your validation dataset. This may involve modifying or discarding variables and includes a process of tweaking model-specific settings (hyperparameters) until an acceptable accuracy level is reached.

Machine Learning Evaluation

Finally, after an acceptable set of hyperparameters is found and your model accuracy is optimized you can test your model. Testing uses your test dataset and is meant to verify that your models are using accurate features. Based on the feedback you receive you may return to training the model to improve accuracy, adjust output settings, or deploy the model as needed.

Machine Learning Best Practices for Efficient Workflows

When defining the workflow for your machine learning project, there are several best practices you can apply. Below are a few to start with.

Define the Project

Carefully define your project goals before starting to ensure your models add value to a process rather than redundancy. When defining your project, consider the following aspects:

- What is your current process? Typically models are designed to replace an existing process. Understanding how the existing process works, what its goals are, who performs it, and what counts as success are all important. Understanding these aspects lets you know what roles your model needs to fill, what restrictions might exist in implementation, and what criteria the model needs to meet or exceed.

- What do you want to predict? Carefully defining what you want to predict is key to understanding what data you need to collect and how models should be trained. You want to be as detailed as possible with this step and make sure to quantify results. If your goals aren't measurable you'll have a hard time ensuring that each is met.

- What are your data sources? Evaluate what data your current process relies on, how it's collected and in what volume. From those sources, you should determine what specific data types and points you need to form predictions.

Find an Approach that Works

The goal of implementing machine learning workflows is to improve the efficiency and/or accuracy of your current process. To find an approach that achieves this goal you need to:

- Research—before implementing an approach, you should spend time researching how other teams have implemented similar projects. You may be able to borrow methods they used or learn from their mistakes, saving yourself time and money.

- Experiment—whether you have found an existing approach to start from or created your own, you need to experiment with it. This is essentially the training and testing phases of your model training.

Build a Full-Scale Solution

When developing your approach, your end result is typically a proof-of-concept. However, you need to be able to translate this proof into a functional product to meet your end goal. To transition from proof to deployable solution, you need the following:

- A/B testing—enables you to compare your current model with the existing process. This can confirm or deny whether your model is effective and able to add value to your teams and users.

- Machine learning API—creating an API for your model implementation is what enables it to communicate with data sources and services. This accessibility is especially important if you plan to offer your model as a machine learning service.

- User-friendly documentation—includes documentation of code, methods, and how to use the model. If you want to create a marketable product it needs to be clear to users how they can leverage the model, how to access its results, and what kind of results they can expect.

Automating Machine Learning Workflows

Automating machine learning workflows enables teams to more efficiently perform some of the repetitive tasks involved in model development. There are many modules and an increasing number of platforms for this, sometimes referred to as autoML.

What is Automated Machine Learning?

AutoML essentially applies existing machine learning algorithms to the development of new models. Its purpose is not to automate the entire process of model development. Instead, it is to reduce the number of interventions that humans must make to ensure successful development. AutoML helps developers get started with and complete projects significantly faster. It also has potential to improve deep learning and unsupervised machine learning training processes, potentially enabling self correction in developed models.

What Can You Automate?

While it would be great to be able to automate all aspects of machine learning operations, this currently isn't possible. What can be reliably automated includes:

- Hyperparameter optimization—uses algorithms like grid search, random search, and Bayesian methods to test combinations of pre-defined parameters and find the optimal combination.
- Model selection—the same dataset is run through multiple models with default hyperparameters to determine which is best suited to learn from your data.
- Feature selection—tools select the most relevant features from pre-determined sets of features.

MLOps With Run:ai

While the tools and tips mentioned above help with automating some parts of the ML lifecycle, such as data preparation, they are not built to automate resource allocation and job scheduling. If resource allocation is not properly configured and optimized, you can quickly hit compute or memory bottlenecks.

You can avoid these issues by replacing static allocation and provisioning with automated and dynamic resource management. This capability is enabled by virtualization and orchestration software from Run:ai, which automates resource management for machine learning and deep learning. With Run:ai, you can automatically run as many compute intensive experiments as needed.

Here are some of the capabilities you gain when using Run:ai:

- Advanced queueing and fair scheduling to allow users to easily and automatically share clusters of GPUs
- Distributed training on multiple GPU nodes to accelerate model training times
- Fractional GPUs to seamlessly run multiple workloads on a single GPU of any type
- Visibility into workloads and resource utilization to improve user productivity

Run:ai simplifies machine learning workflows, helping data scientists accelerate their productivity and the quality of their models. Learn more about the Run:ai platform.