Machine Learning Best Practices for Public Sector Organizations

September 29, 2021

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Introduction

In 2019, the White House issued an executive order promoting the use of trustworthy artificial intelligence (AI) in the federal government.[1] This order launched the American AI Initiative, a concerted effort to promote and protect AI technology and innovation in the United States. This executive order also laid the foundation, with broad guidelines and policies, for agencies on the design, development, acquisition, and the use of AI in government.

Machine learning (ML) and deep learning (DL) are computer science fields derived from the discipline of AI. Collectively called ML in this whitepaper, these fields help modernize the government and ensure federal agencies are effectively delivering on their mission objectives on behalf of the American people. AI & ML can help government agencies solve complex problems with citizen services, public safety, healthcare, transportation, and other service verticals. To enable these capabilities, agencies are investing in AI & ML solutions, especially to improve mission effectiveness, make evidence-based decisions, and automate repetitive tasks. As an example, in 2018 the Defense Advanced Research Project Agency (DARPA) announced a multi-year investment of more than $2 billion in new and existing programs and called it the “AI Next” campaign.[2] The National Science Foundation (NSF) invests more than $500 million in AI research annually.[3]

However, several challenges remain within the US public sector regarding the broader adoption of ML initiatives. Organizations have stringent federal, state, and local security and compliance mandates including the Federal Risk and Authorization Management Program (FedRAMP), Department of Defense (DOD) Cloud Computing Security Requirements Guide (CC SRG), and the Health Insurance Portability and Accountability Act (HIPAA), among others. These requirements include protecting sensitive citizen data, isolating environments from internet access, and the principles of least-privilege-access controls. Additionally, the ML lifecycle presents its own challenges in terms of data and model lifecycle management, including the bias within ML models that needs to be addressed to improve the trust with public.

This whitepaper outlines some of the challenges for US public sector agencies in adoption and implementation of ML, and provides best practices to address these challenges. The target audience for this whitepaper includes executive leaders and agency IT Directors. You can get started on AI and ML by visiting Machine Learning on AWS, AWS Machine Learning Embark Program, or the Amazon Machine Learning Solutions Lab

Challenges for public sector

Government, education, and nonprofit organizations face several challenges in implementing ML programs to accomplish their mission objectives. This section outlines some of the challenges in seven critical areas of an ML implementation. These are outlined as follows:

1. **Data Ingestion and Preparation**. Identifying, collecting, and transforming data is the foundation for ML. The ability to extract data from different types of data sources (ranging from flat files to databases, structured and unstructured, real time and batch) can be challenging given the range of technologies found in public sector organizations. Once the data is extracted, it needs to be cataloged and organized so that it is available for consumption with the necessary approvals in compliance with public sector guidelines.
2. **Model Training and Tuning.** There are hundreds of algorithms available for ML model training and tuning that solve various types of problems. One of the major challenges facing public sector organizations is the ability to create a common platform that provides these algorithms and the structure required for visibility and maintenance. Challenges also exist in optimizing model training performance with minimal resources without compromising on the quality of ML models.
3. **ML Operations (MLOps).** Integrating ML into business operations, referred to as MLOps, requires significant planning and preparation. One of the major hurdles facing government organizations is the ability to create a repeatable process for deployment that is consistent with their organizational best practices. Mechanisms need to be put in place to ensure scalability and availability, as well as recovery of the models in case of disasters. Another challenge is to effectively monitor the model in production to ensure that ML models do not lose their effectiveness due to introduction of new variables, changes in source data, or issues with source data.
4. **Management & Governance.** Public sector organizations face increased scrutiny to ensure that public funds are being properly utilized to serve mission needs. As such, they need to provide increased visibility into monitoring and auditing ML workloads. Changes need to be tracked in several places, including data sources, data models, data transfer and transformation mechanisms, deployments and inference endpoints. A clear separation needs to be put in place between development and production workloads while enforcing separation of duties with appropriate approval mechanisms. In addition, any underlying infrastructure, software, and licenses need to be maintained and managed.

5. **Security & Compliance.** Security and compliance of ML workloads is one of the biggest challenges facing public sector organizations. The sensitive nature of the work done by these organizations results in increased security requirements at all levels of an ML platform. This can be very challenging as data is spread across a large number of data sources, is constantly evolving, and is constantly sent across the network between data storage and compute platforms. Data is also transmitted between compute instances in the case of distributed learning. Last but not least is the alignment with the principles of least privilege and application of a consistent user authentication and authorization mechanism.
6. **Cost Optimization.** Given the complexity of ML projects, and the amount of data, compute, and other software required to successfully manage a project, costs can quickly spiral out of control. The challenge facing public sector agencies is the need to account for the resources used, and to monitor the usage against specified cost centers and task orders. Not only do they need to track usage of resources, but they also need to be able to effectively manage the costs.
7. **Bias & Explainability.** Given the impact of public sector organizations on the citizens, the ability to understand why an ML model makes a specific prediction becomes paramount – this is also known as ML explainability. Organizations are under pressure from policymakers and regulators to ensure that ML and data-driven systems do not violate ethics and policies, and do not result in potentially discriminatory behavior. In January 2020, the U.S. government published draft rules for the regulation of Artificial Intelligence (AI) in the United States. These rules state that any government regulation of public sector AI must encourage “reliable, robust, and trustworthy AI” and these standards should be the overarching guiding theme. Demonstrating explainability is a significant challenge because complex ML models are hard to understand and even harder to interpret and debug. Public sector organizations need to invest significant time with appropriate tools, techniques, and mechanisms to demonstrate explainability and lack of bias in their ML models, which could be a deterrent to adoption.

Best Practices

AWS Cloud provides several fully-managed services that supply developers and data scientists with the ability to prepare, build, train, and deploy ML models. This section provides the best practices for using these services to address the challenges outlined earlier. The best practices are organized by the seven critical areas of an ML implementation described in the previous section.

Data Ingestion and Preparation

Data ingestion and preparation involves processes in collecting, curating, and preparing the data for ML. Data ingestion involves collecting batch or streaming data in unstructured or structured format. Data preparation takes the ingested data and processes to a format that can be used with ML.

Identifying, collecting, and transforming data is the foundation for ML. There is widespread consensus[4] among ML practitioners that data preparation accounts for approximately 80% of the time spent in developing a viable ML model. There are several challenges that public sector organizations face in this phase: First is the ability to connect to and extract data from different types of data sources. Once the data is extracted, it needs to be cataloged and organized so that it is available for consumption, and there needs to be a mechanism in place to ensure that only authorized resources have access to the data. Mechanisms are also needed to ensure that source data transformed for ML is reviewed and approved for compliance with federal government guidelines.

The AWS Cloud provides services that enable public sector customers to overcome challenges in data ingestion, data preparation, and data quality. These are further described as follows:

Data Ingestion

The AWS Cloud enables public sector customers to overcome the challenge of connecting to and extracting data from both streaming and batch data, as described in the following:

- **Streaming Data.** For streaming data, Amazon Kinesis and Amazon Managed Streaming for Apache Kafka (Amazon MSK) enable the collection, processing, and analysis of data in real time. Amazon Kinesis provides a suite of capabilities to collect, process, and analyze real-time, streaming data. Amazon Kinesis Data Streams (KDS) is a service that enables ingestion of streaming data. Producers of data push data directly into a stream, which consists of a group of stored data units called records. The stored data is available for further processing or storage as part of the data pipeline. Ingestion of streaming videos can be done using Amazon Kinesis Video Streams.

This service can capture streams from millions of devices, and durably store, encrypt, and index video data for use in ML models. If data does not need to be stored for real-time processing, Amazon Kinesis Data Firehose is a service that can be used to deliver real-time streaming data to a chosen destination. For example, a data source could be a custom producer application and a destination could be Amazon Simple Storage Service (Amazon S3) or Amazon RedShift. If you already use Apache Kafka, you can use Amazon MSK, a fully managed service, to build and run applications that use Apache Kafka to process streaming data without needing Apache Kafka infrastructure management expertise.

- **Batch Data.** There are a number of mechanisms available for data ingestion in batch format. With AWS Database Migration Services (AWS DMS), you can replicate and ingest existing databases while the source databases remain fully operational. The service supports multiple database sources and targets, including writing data directly to Amazon S3. AWS DataSync is a data transfer service that simplifies, automates, and accelerates moving and replicating data between on-premises storage systems such as network file system (NFS) and AWS storage services such as Amazon Elastic File System (EFS) and Amazon S3. You can use AWS Transfer Family for ingestion of data from flat files using secure protocols such as Secure File Transfer Protocol (SFTP), File Transfer Protocol over SSL (FTPS), and File Transfer Protocol (FTP). For large amounts of data, you can use the AWS Snow Family for transferring data in bulk using secure physical appliances.

Data Preparation

Once the data is extracted, it needs to be transformed and loaded into a data store for feeding into an ML model. It also needs to be cataloged and organized so that it is available for consumption, and also needs to enable data lineage for compliance with federal government guidelines. AWS Cloud provides three services that provide these mechanisms. They are:

- AWS Glue is a fully managed ETL (extract, transform and load) service that makes it simple and cost-effective to categorize, clean, enrich, and migrate data from a source system to a data store for ML. The AWS Glue Data Catalog provides the location and schema of ETL jobs as well as metadata tables (where each table specifies a single source data store). A crawler can be set to automatically take inventory of the data in your data stores..

  ETL jobs in AWS Glue consist of scripts that contain the programming logic that performs the transformation. Triggers are used to initiate jobs either on a schedule or as a result of a specified event. AWS Glue Studio provides a graphical interface that enables visual composition of data transformation workflows on AWS Glue's Apache Spark-based serverless ETL engine. AWS Glue generates the code that's required to transform the data from source to target based on the source and target information provided. Custom scripts can also be provided in the AWS Glue console or API to transform and process the data.

  In addition, AWS Glue DataBrew, a visual data preparation tool, can be used to simplify the process of cleaning and normalizing the data. It comes with hundreds of data transformations that can be used quickly to prepare data for ML without having to write your own transformation scripts.

  AWS Glue also features the ability to integrate with Amazon SageMaker. Amazon SageMaker is a comprehensive service that provides purpose-built tools for every step of ML development and implementation. In AWS Glue, you can create a development endpoint and then create a SageMaker notebook to help develop your ETL and ML scripts. A development endpoint allows you to iteratively develop and test your ETL scripts using the AWS Glue console or API.

- Amazon SageMaker Data Wrangler is a service that enables the aggregation and preparation of data for ML and is directly integrated into Amazon SageMaker Studio. Both Amazon Data Wrangler and Amazon SageMaker Studio are features of the Amazon SageMaker service. Data Wrangler contains hundreds of built-in transformations to quickly normalize, transform, and combine features without having to write any code. Using the Data Wrangler user interface, you can view table summaries, histograms, and scatter plots.

- Amazon EMR: Many organizations use Spark for data processing and other purposes such as for a data warehouse. These organizations already have a complete end-to-end pipeline in Spark and also the skillset and inclination to run a persistent Spark cluster for the long term. In these situations, Amazon EMR, a managed service for Hadoop-ecosystem clusters, can be used to process data. Amazon EMR reduces the need to set up, tune, and maintain clusters. Amazon EMR also features other integrations with Amazon SageMaker, for example, to start a SageMaker model training job from a Spark pipeline in Amazon EMR.

Data quality

Data that is obsolete or inaccurate not only causes issues in developing accurate ML models, but can significantly erode stakeholder and public trust. Public sector organizations need to ensure that data ingested and prepared for ML is of the highest quality by establishing a well-defined data quality framework. See How to Architect Data Quality on the AWS Cloud for an example on how you can set up a data quality framework on the AWS Cloud.

Model Training and Tuning

Model Training and Tuning involves the selection of a ML model that is appropriate for the use case, followed by training and tuning of the ML model.

One of the major challenges facing the public sector is the ability for team members to apply a consistent pattern or framework for working with multitudes of options that exist in this space. Different teams use different technologies and it is challenging to bring these into a uniform environment for increased visibility and tracking. For example, some teams may be using Python, while some other teams use R. Some teams may have standardized on TensorFlow, whereas other teams may have standardized on Pytorch. Challenges also exist in optimizing model training performance, input data formats, and distributed training. A significant amount of time is spent on fine tuning a model to achieve the expected performance.

The AWS Cloud enables public sector customers to overcome challenges in model selection, training, and tuning as described in the following.

Model Selection

Amazon SageMaker provides the flexibility to select from a wide number of options using a consistent underlying platform.

- **Programming Language.** Amazon SageMaker notebook kernels provide the ability to use both Python, as well as R, natively. The Amazon SageMaker Python SDK provides open-source Python APIs and containers to train and deploy models in SageMaker. To use coding languages such as Stan or Julia, a Docker image can be created and brought into SageMaker for model training and inference (see Figure 3 below for more details on this option). To use programming languages like C++ or Java, custom images on Amazon ECS/EKS can be used to perform model training.
- **Built-in algorithms:** **Amazon SageMaker Built-in Algorithms** provides several built-in algorithms covering different types of ML problems. These algorithms are already optimized for speed, scale, and accuracy. Additionally, for classification or regression with tabular data, SageMaker Autopilot can be used to automatically explore data, select algorithms relevant to the problem type, and prepare the data to facilitate model training and tuning. AutoML ranks all of the optimized models tested by their performance and finds out the best performing model. The AutoML approach is especially useful for application programmers who are new to ML.
- **Script Mode:** For experienced ML programmers who are comfortable with using their own algorithms, Amazon SageMaker provides the option to write your custom code (script) in a text file with a .py extension (see Figure 2).

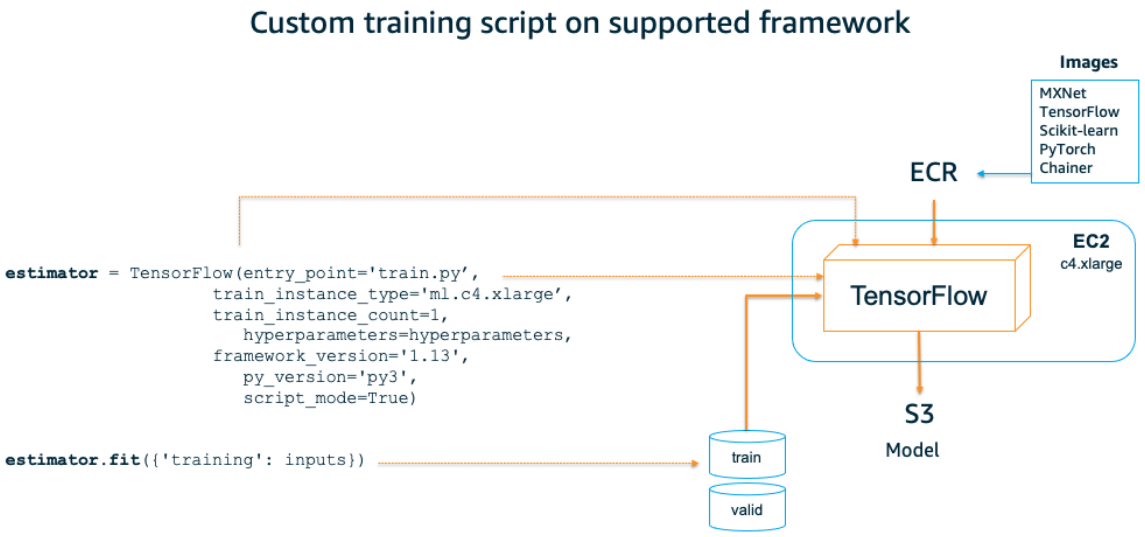

Figure 1: Script Mode

This option is known as “script mode” and the custom code can be written using any SageMaker supported framework. Code needs to be prepared and packaged in a Python file (.py extension), adding in some training environment variables as input arguments. Code that requires Python packages hosted on PyPi can be listed in a requirement.txt file and included in the code directory.

- **Use a custom Docker image**: ML programmers may be using algorithms that are not included in a SageMaker supported framework, not hosted on PyPi, or written in a language like Stan and Julia. In these cases, the training of the algorithm and serving of the model can be done using a custom Docker image (see Figure 2 below).

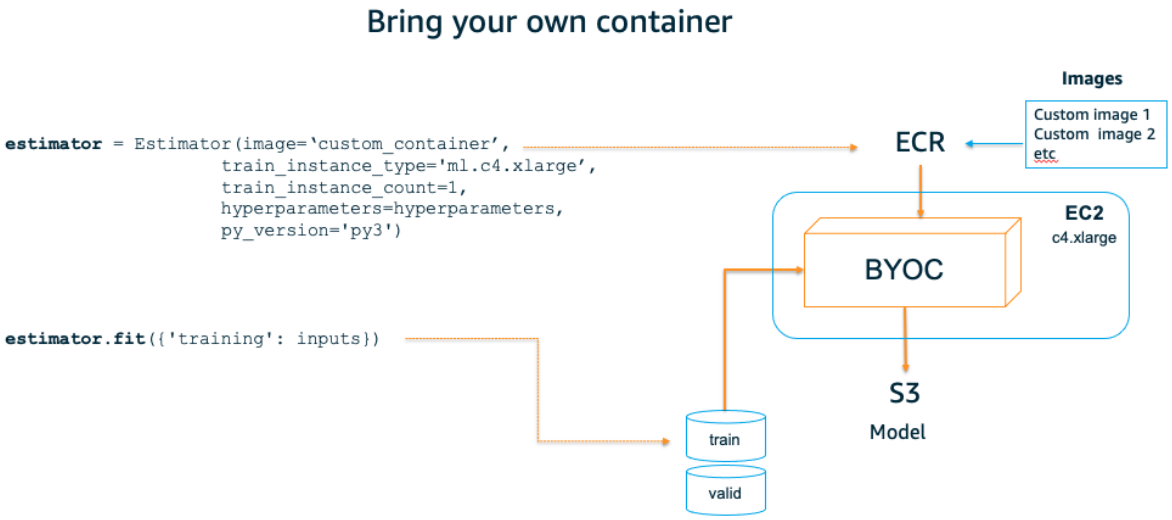

Figure 2: Bring your own container

For more information on custom Docker images in SageMaker, see Using Docker containers with SageMaker

Model Training

Amazon SageMaker provides a number of built-in options for optimizing model training performance, input data formats, and distributed training.

- **Data parallel:** ML training processes go through an entire dataset in one training cycle called an epoch. It is common to have multiple training iterations per epoch. When the training dataset is big, each epoch becomes time consuming. In these situations, SageMaker's distributed data parallel library can be considered for running training jobs in parallel. The library optimizes the training job for AWS network infrastructure and Amazon EC2 instance topology, and takes advantage of gradient updates to communicate between nodes with a custom algorithm.
- **Pipe mode:** Pipe mode accelerates the ML training process: instead of downloading data to the local Amazon EBS volume prior to starting the model training, Pipe mode streams data directly from S3 to the training algorithm while it is running. This enables the training job to start sooner, finish quicker, and need less disk space.
- **Incremental training:** Amazon SageMaker supports incremental training to train a new model from an existing model artifact, to save both training time and resources. Incremental training may be considered when there are publicly available pre-trained models related to the ML use case. It can also be considered if an expanded dataset contains an underlying pattern that was not accounted in previous models, or to resume a stopped training job.
- **Model Parallel training:** Sometimes ML models are too large to fit into GPU memory in a training process. In these situations, Amazon SageMaker's distributed model parallel library can be used to automatically and efficiently split a model across multiple GPUs and instances and coordinate model training.

Model Tuning

Amazon SageMaker provides automatic hyperparameter tuning to find the best version of a model in an efficient manner, enabling public sector organizations to judiciously use their resource on other activities. SageMaker hyperparameter tuning runs many training jobs on a dataset using specified ranges of hyperparameters. It then chooses the hyperparameter values that result in a model that performs the best, as measured by a selected metric. The following best practices ensure a better tuning result:

- **Limit the number of hyperparameters:** Up to 20 hyperparameters can be simultaneously specified to optimize a tuning job. However, limiting the search to a much smaller number is likely to give better results, as this can reduce the computational complexity of a hyperparameter tuning job. Moreover, a smaller number of hyperparameters provides better understanding of how a specific hyperparameter would affect the model performance.

- **Choose hyperparameter ranges appropriately:** The range of values for hyperparameters can significantly affect the success of hyperparameter optimization. Better results are obtained by limiting the search to a small range of values. If the best metric values within a subset of the possible range are already known, consider limiting the range to that subset.
- **Pay attention to scales for hyperparameters:** During hyperparameter tuning, SageMaker attempts to figure out if hyperparameters are log-scaled or linear-scaled. Initially, it assumes that hyperparameters are linear-scaled. If they are in fact log-scaled, it might take some time for SageMaker to discover that fact. Directly setting hyperparameters as log-scaled when they're already known could improve hyperparameter optimization.
- **Set the best number of concurrent training jobs:** Running more hyperparameter tuning jobs concurrently gets more work done quickly, but a tuning job improves only through successive rounds of experiments. Typically, running one training job at a time achieves the best results with the least amount of compute time.
- **Report the wanted objective metric for tuning when the training job runs on multiple instances:** When a training job runs on multiple instances, hyperparameter tuning uses the last-reported objective metric value from all instances of that training job as the value of the objective metric for that training job. Therefore, distributed training jobs should be designed such that the objective metric reported is the one that is needed.
- **Enable early stopping for hypermeter tuning job:** Early stopping helps reduce compute time and helps avoid overfitting the model. It stops the training jobs that a hyperparameter tuning job launches early when they are not improving significantly as measured by the objective metric.
- **Run a warm start using previous tuning jobs:** Use a warm start for fine-tuning previous hyperparameter tuning jobs. A warm start uses information from the previous hyperparameter tuning jobs to increase the performance of the new hyperparameter tuning job by making the search for the best combination of hyperparameters more efficient.

MLOps

MLOps is the discipline of integrating ML workloads into release management, Continuous Integration / Continuous Delivery (CI/CD), and operations.

One of the major hurdles facing government organizations is the ability to create a repeatable process for deployment that is consistent with their organizational best practices. Using ML models in software development makes it difficult to achieve versioning, quality control, reliability, reproducibility, explainability, and audibility in that process. This is due to the number of changing artifacts to be managed in addition to the software code, such as the datasets, the ML models, the parameters and hyperparameters used by such models, and the size and portability of such artifacts can be orders of magnitude higher than the software code. In addition, different teams might own different parts of the process; data engineers might be building pipelines to make data accessible, while data scientists can be researching and exploring better models. ML engineers or developers have to work on integrating the models and releasing them to production. When these groups work independently, there is a high risk of creating friction in the process and delivering suboptimal results.

AWS Cloud provides a number of different options that solve these challenges, either by building an MLOps pipeline from scratch or by using managed services.

Amazon SageMaker Projects

A SageMaker project is an AWS Service Catalog provisioned product that enables creation of an end-to-end ML solution. By using a SageMaker project, teams of data scientists and developers can work together on ML business problems. SageMaker projects use MLOps templates that automate the model building and deployment pipelines using CI/CD. SageMaker-provided templates can be used to provision the initial setup required for a complete end-to-end MLOps system including model building, training, and deployment. Custom templates can also be used to customize the provisioning of resources.

Amazon SageMaker Pipelines

SageMaker Pipelines is a purpose-built, CI/CD service for ML. SageMaker Pipelines brings CI/CD practices to ML, such as maintaining parity between development and production environments, version control, on-demand testing, and end-to-end automation, helping scale ML throughout the organization. Pipelines is integrated with SageMaker Python SDK as well as SageMaker Studio for visualization and management of workflows. With the SageMaker Pipelines model registry, model versions can be stored in a central repository for easy browsing, discovery, and selection of the right model for deployment based on business requirements. Pipelines provide the ability to log each step within the ML workflow for a complete audit trail of model components such as training data, platform configurations, model parameters,

and learning gradients. Audit trails can be used to recreate models and help support compliance requirements.

AWS CodePipeline and AWS Lambda

For AWS programmers and teams that are already working with CodePipeline for deployment of other workloads, the option exists to utilize the same workflows for ML. Figure 3 below represents a reference pipeline for deployment on AWS.

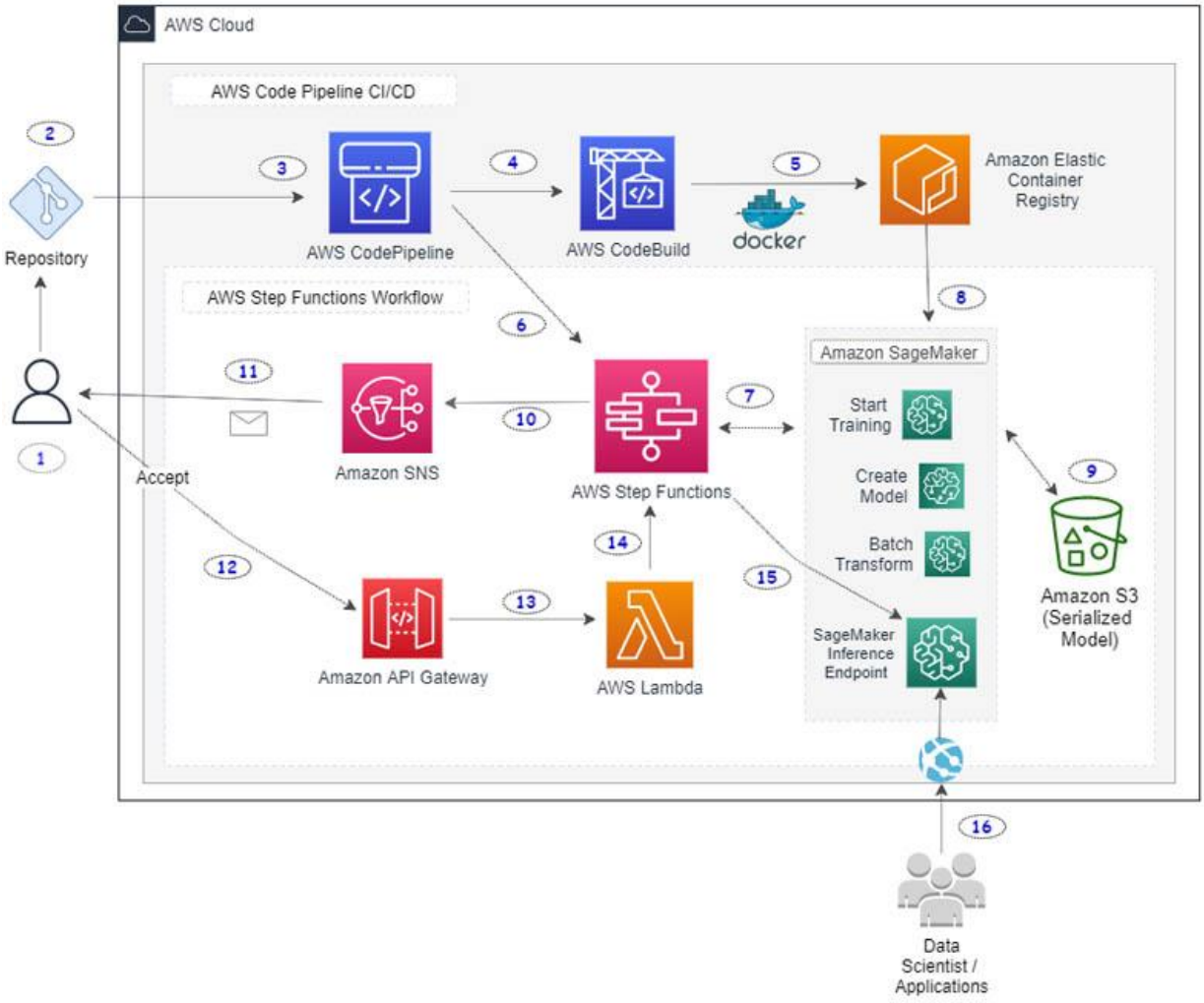

Figure 3: Reference Architecture CI/CD Pipeline for ML on AWS

See [Build a CI/CD pipeline for deploying custom machine learning models using AWS services] for details on the reference architecture and implementation.

AWS StepFunctions Data Science Software Development Kit (SDK)

The AWS Step Functions Data Science SDK is an open-source Python library that allows data scientists to create workflows that process and publish ML models using SageMaker and Step Functions. This can be used by teams that are already comfortable using Python and AWS Step Functions. The SDK provides the ability to copy workflows, experiment with new options, and then put the refined workflow in production. The SDK can also be used to create and visualize end-to-end data science workflows that perform tasks such as data pre-processing on AWS Glue and model training, hyperparameter tuning, and endpoint creation on Amazon SageMaker. Workflows can be reused in production by exporting AWS CloudFormation (infrastructure as code) templates.

AWS MLOps Framework

Figure 4 below illustrates an AWS solution that provides an extendable framework with a standard interface for managing ML pipelines.

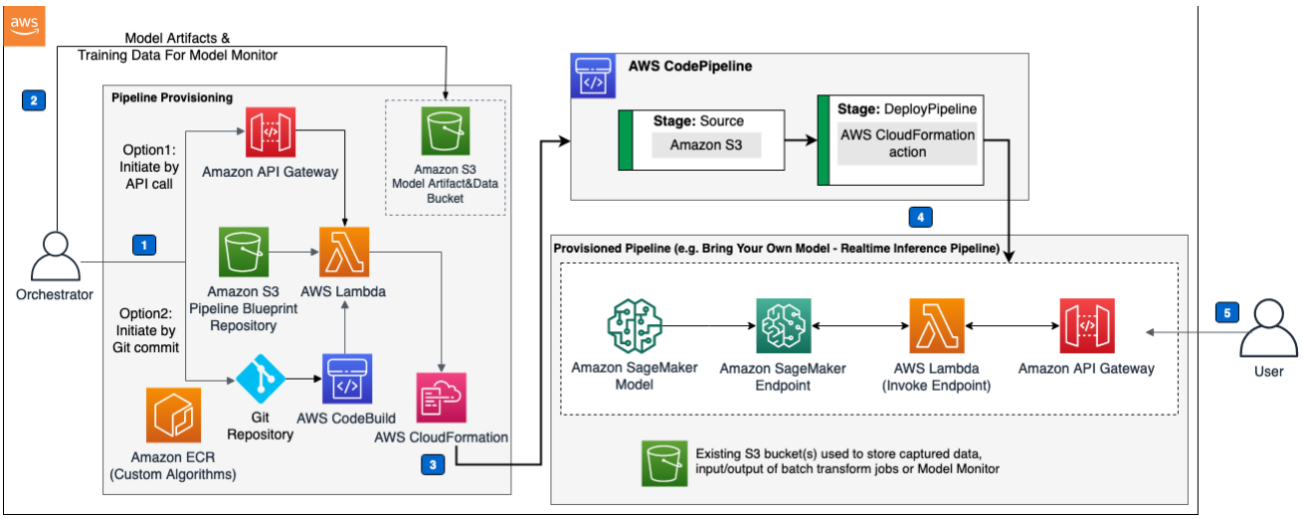

Figure 4: AWS MLOps Framework

The solution provides a ready-made template to upload trained models (also referred to as a *bring your own model*), configure the orchestration of the pipeline, and monitor the pipeline's operations.

Deploy Custom Deep Learning Models

In addition to Amazon SageMaker, AWS also provides the option to deploy custom code on virtual machines using Amazon EC2, and containers using self-managed Kubernetes on Amazon EC2, Amazon Elastic Container Service (Amazon ECS) and Amazon Elastic Kubernetes Service (Amazon EKS). AWS Deep Learning AMIs can be used to accelerate deep learning by quickly launching Amazon EC2 instances that are pre-installed with popular deep learning frameworks. AWS Deep Learning Containers are Docker images pre-installed with deep learning frameworks to deploy optimized ML environments. For an example of how to deploy custom deep learning models, see Deploy Deep Learning Models on Amazon ECS.

Deploy ML at the edge

Training your ML models requires powerful compute infrastructure available in the cloud. However, making inferences against these models typically requires far less computational power. In some cases, such as with edge devices, inferencing needs to occur even when there is limited or no connectivity to the cloud. Mining fields are an example of this type of use case. To make sure that an edge device can respond quickly to local events, it is critical that you can get inference results with low latency.

AWS IoT Greengrass enables ML inference locally using models that are created, trained, and optimized in the cloud using Amazon SageMaker, AWS Deep Learning AMI, or AWS Deep Learning Containers, and deployed on the edge devices.

Performing inference locally on connected devices running AWS IoT Greengrass reduces latency and cost. Instead of sending all device data to the cloud to perform ML inference and make a prediction, you can run inference directly on the device. As predictions are made on these edge devices, you can capture the results and analyze them to detect outliers. Analyzed data can then be sent back to the cloud, where it can be reclassified and tagged to improve the ML model. For example, you can build a predictive model in Amazon SageMaker for scene detection analysis, optimize it to run on any camera, and then deploy it to send an alert when suspicious activity occurs. Data gathered from the inference running on AWS IoT Greengrass can be sent back to Amazon SageMaker, where it can be tagged and used to continuously improve the quality of the ML models. See Machine Learning at the Edge: Using and Retraining Image Classification Models with AWS IoT Greengrass (Part 1) for more details.

Management and Governance

Public sector organizations face increased scrutiny to ensure that funds are properly utilized to serve mission needs. As such, ML workloads need to provide increased visibility for monitoring and auditing. Changes need to be tracked in several places, including data sources, data models, data transfer processes and transformation processes, and deployment endpoints and inference endpoints. A clear separation needs to be put in place between development and production workloads, while enforcing separation of duties with appropriate approval mechanisms. In addition, any underlying infrastructure, software, and licenses need to be maintained and managed. This section highlights several AWS services and associated best practices to address these management and governance challenges.

Enable governance and control

AWS Cloud provides several services that enable governance and control. These include:

- **AWS Control Tower.** Setup and governance can be complex and time consuming for organizations with multiple AWS accounts and teams. AWS Control Tower creates a landing zone that consists of a predefined structure of accounts using AWS Organizations, the ability to create accounts using AWS Service Catalog, enforcement of compliance rules called guardrails using Service Control Policies, and detection of policy violations using AWS Config. (See the Cross-account deployments in an AWS Control Tower environment blog for details on how to set up Control Tower)
- **AWS License Manager.** Public sector organizations may have existing software with their own licenses being used for various tasks in ML such as ETL. AWS License Manager can be used to track this software obtained from the AWS Marketplace and keep a consolidated view of all licenses. AWS License Manager enables sharing of licenses with other accounts in the organization.

- **Resource Tagging.** Organizing AI/ML resources can be done using tags. Each tag is a simple label consisting of a customer-defined key and an optional value that can make it easier to manage, search for, and filter resources by purpose, owner, environment, or other criteria. Automated tools such as AWS Resource Groups and the Resource Groups Tagging API enable programmatic control of tags, making it easier to automatically manage, search, and filter tags and resources. To make the most effective use of tags, organizations should create business-relevant tag groupings to organize their resources along technical, business, and security dimensions.

Provision ML resources that meet policies

AWS Cloud provides several services that enable consistent and repeatable provisioning of ML resources per organization policies.

- **AWS CloudFormation.** A successful AI/ML solution may involve resources from multiple services. Deploying and managing these resources one by one can be time-consuming and inconvenient. AWS CloudFormation provides a mechanism to model a collection of related AWS and third-party resources, provision them quickly and consistently, and manage them throughout their lifecycles, by treating infrastructure as code.
- **AWS Cloud Development Kit (CDK).** Many team members prefer to work in their own language to define the infrastructure, as opposed to using JSON and YAML. The AWS CDK, an open-source software development framework, allows teams to define cloud infrastructure in code directly in supported programming languages (i.e., TypeScript, JavaScript, Python, Java, and C#). CDK defines reusable cloud components known as Constructs, and composes them together into Stacks and Apps. The constructs are synthesized into CloudFormation at the time of deployment.

- **AWS Service Catalog.** Deploying and setting up ML workspaces for a group or different groups of people is always a big challenge for public sector organizations. AWS Service Catalog provides a solution for this problem. It enables the central management of commonly deployed IT services, and achieves consistent governance and meets compliance requirements. End users can quickly deploy only the approved IT services they need, following the constraints set by the organization. For example, AWS Service Catalog can be used with Amazon SageMaker notebooks to provide end users a template to quickly deploy and set up their ML Workspace. The following diagram shows how AWS Service Catalog ensures two separate workflows for cloud system administrators and data scientists or developers who work with Amazon SageMaker.

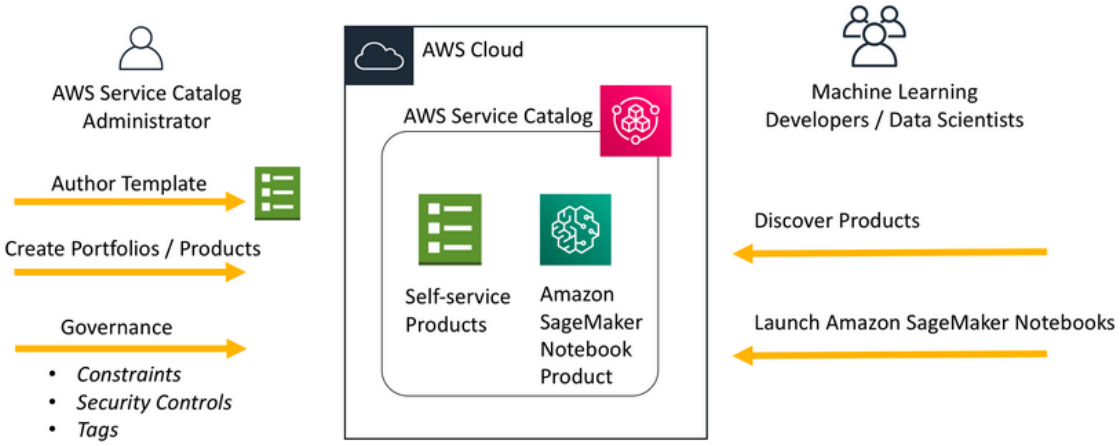

Figure 5: Setting up ML workspace using Service Catalog

By leveraging AWS Service Catalog, cloud administrators are able to define the right level of controls and enforce data encryption along with centrally-mandated tags for any AWS service used by various groups. At the same time, data scientists can achieve self-service and a better security posture by simply launching an Amazon SageMaker notebook instance through AWS Service Catalog.

Operate environment with governance

AWS Cloud provides several services that enable the reliable operation of the ML environment.

- Amazon CloudWatch is a monitoring and observability service used to monitor resources and applications run on AWS in real time. Amazon SageMaker has built-in Amazon CloudWatch monitoring and logging to manage production compute infrastructure and perform health checks, apply security patches, and conduct other routine maintenance. For a complete list of metrics that can be monitored, refer to the Monitor Amazon SageMaker with Amazon CloudWatch section of the SageMaker user guide.
- Amazon EventBridge is a serverless event bus service that can monitor status change events in Amazon SageMaker. EventBridge enables automatic responses to events such as a training job status change or endpoint status change. Events from SageMaker are delivered to EventBridge in near real time. Simple rules can be written to indicate which events are of interest, and what automated actions to take when an event matches a rule.
- SageMaker Model Monitor can be used to continuously monitor the quality of ML models in production. Model Monitor can notify team members when there are deviations in the model quality. Early and proactive detection of these deviations enables corrective actions, such as retraining models, auditing upstream systems, or fixing quality issues without having to monitor models manually or build additional tooling. The model monitor provides various types of monitoring, including data quality drift, model quality drift, bias drift, and feature attribution drift. For a sample notebook with the full end-to-end workflow for Model Monitor, see the Introduction to Amazon SageMaker Model Monitor or see Monitoring in-production ML models at large scale using Amazon SageMaker Model Monitor, which outlines how to monitor ML models in production at scale.
- **AWS CloudTrail.** Amazon SageMaker is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in SageMaker. CloudTrail captures all API calls for SageMaker. The calls captured include actions from the SageMaker console and code calls to the SageMaker API operations. Continuous delivery of CloudTrail events can be delivered to an Amazon S3 bucket, including events for SageMaker. Every event or log entry contains information about who generated the request.

Security and compliance

Public sector organizations have a number of security challenges and concerns with hosting ML workloads in the cloud as these applications can contain sensitive customer data – this includes personal information or proprietary information that must be protected over the entire data lifecycle. The specific concerns also include protecting

the network and underlying resources such as compute, storage and databases; user authentication and authorization; logging, monitoring and auditing. These objectives are summarized in Figure 6 below.

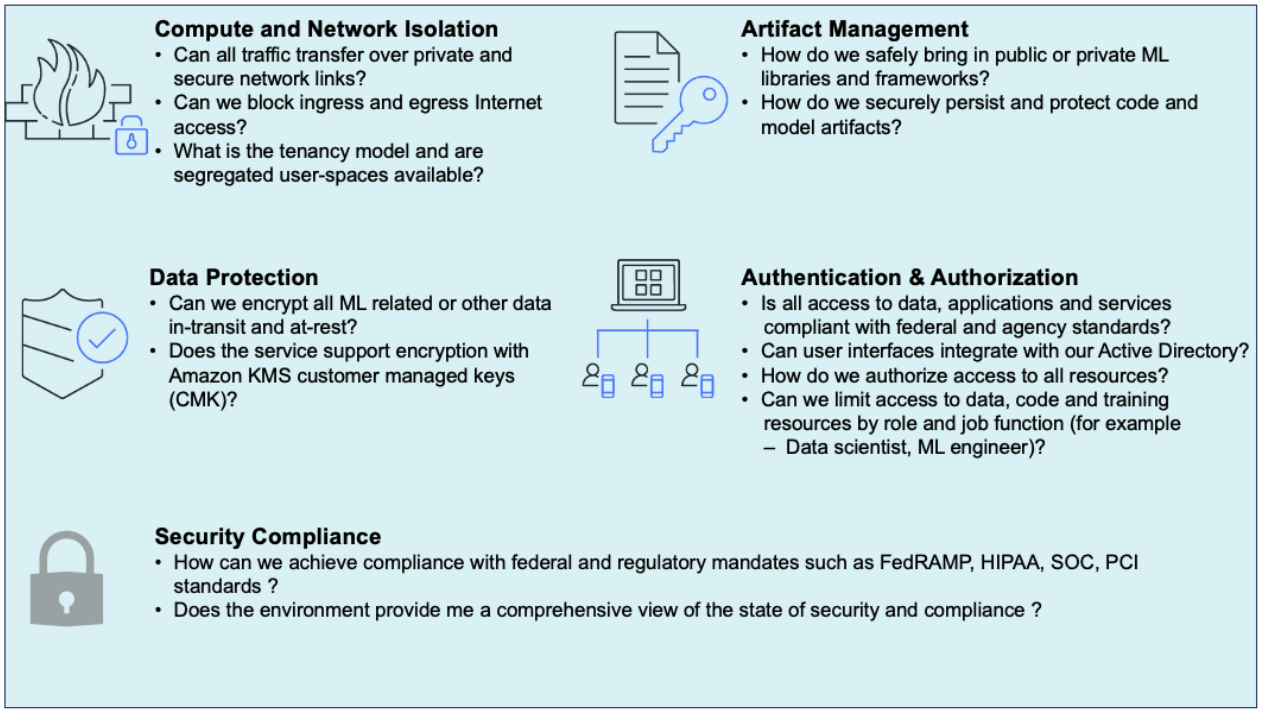

Figure 6 – Security and Compliance objectives for hosting public sector ML workloads

This subsection provides best practices and guidelines to address some of these security and compliance challenges.

Compute and network isolation

One of the major requirements with many public sector ML projects is the ability to keep the environments, data and workloads secure and isolated from internet access. These can be achieved using the following methods:

- **Provision ML components in an isolated VPC with no internet access**: SageMaker components including the studio, notebooks, training jobs and hosting instances can be provisioned in an isolated VPC with no internet access. Traffic can be restricted from accessing the internet by launching SageMaker Studio in a Virtual Private Cloud (VPC) of choice. This allows fine-grained control of the network access and internet connectivity of SageMaker Studio notebooks. Direct internet access can be disabled to add an additional layer of security.

  To disable direct internet access, specify the VPC only network access type when onboarding to Studio. The same concept can be applied to SageMaker notebooks by choosing to launch the notebook instance in a VPC to restrict which traffic can go through the public Internet. When launched with the VPC attached, the notebook instance can be configured either with or without direct internet access. Traffic to public endpoints such as S3 or SageMaker APIs can be configured to traverse over VPC endpoints to ensure that the traffic stays within the AWS network. Please refer to Building secure ML environments with Amazon SageMaker for further details.

- **Use VPC end-point and end-point policies to further limit access:** AWS resources can be directly connected with public endpoints such as S3, CloudWatch, and SageMaker API / SageMaker Runtime through an interface endpoint in the VPC instead of connecting over the internet. When a VPC interface endpoint is used, communication between the VPC and the SageMaker API or Runtime is entirely and securely within the AWS network. VPC endpoint policies can be configured to further limit access based on who can perform actions, what actions can be performed, and the resources on which these actions can be performed. As an example, access to an S3 bucket can be restricted only to a specific SageMaker studio domain or set of users, and each studio domain can be restricted to have access only to a specific S3 bucket (see Securing Amazon SageMaker Studio connectivity using a private VPC, which outlines how to secure SageMaker studio connectivity using a private VPC). Figure 7 below outlines an architecture diagram that represents how to set up SageMaker studio using a private VPC.

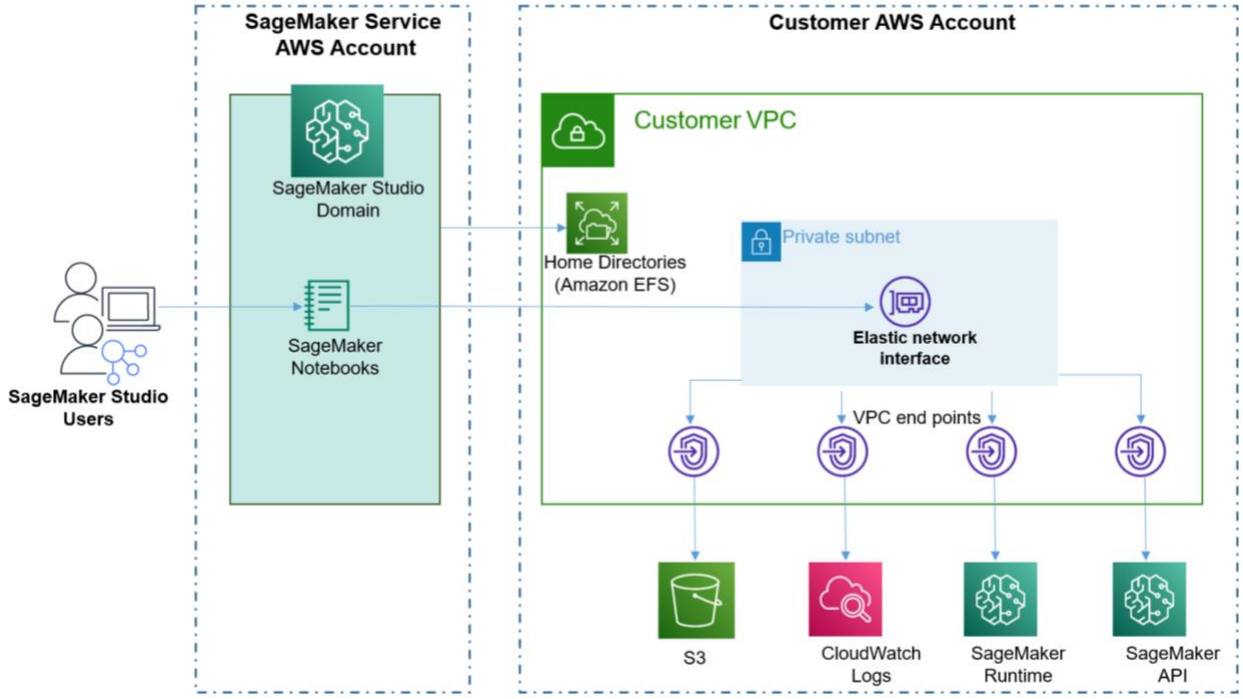

Figure 7: SageMaker Studio in a private VPC

- **Allow access from only within the VPC:** An IAM policy can be created to prevent users outside the VPC from accessing SageMaker Studio or SageMaker notebooks over the internet. This ensures access to only connections made from within the VPC. As an example, this policy can help restrict connections made only through specific VPC endpoints or a specific set of source IP addresses. This policy can be added to every IAM user, group, or role used to access Studio or Jupyter notebooks.
- **Intrusion detection and prevention:** AWS Gateway Load Balancer (GWLB) can be used to deploy, scale, and manage the availability of third-party virtual appliances such as firewalls, intrusion detection and prevention systems, and deep packet inspection systems in the cloud. GWLB allows custom logic or third party offering into any networking path for AWS where inspection is needed and the corresponding action is taken on packets. For example, a simple application can be developed to check if there is any unencrypted traffic or TLS1.0/TLS1.1 traffic between VPCs. Additionally, AWS Partner Network and AWS Marketplace partners can offer their virtual appliances as a service to AWS customers without having to solve the complex problems of scale, availability, and service delivery. Please refer to Introducing AWS Gateway Load Balancer – Easy Deployment, Scalability, and High Availability for Partner Appliances for further details on GWLB.

- **Additional security to allow access to resources outside your VPC:** If access is needed to an AWS service that does not support interface VPC endpoints, or to a resource outside of AWS, a NAT gateway needs to be created and security groups need to be configured to allow outbound connections. Additionally, AWS Network Firewall can be used to filter outbound traffic, for example, to specific GitHub repositories. AWS Network Firewall supports inbound and outbound web filtering for unencrypted web traffic. For encrypted web traffic, Server Name Indication (SNI) is used for blocking access to specific sites. In addition, AWS Network Firewall can filter fully qualified domain names (FQDN).

Data Protection

- **Protect data at rest:** AWS Key Management service (KMS) can be used to encrypt ML data, studio notebooks and SageMaker notebook instances. SageMaker uses AWS managed customer master keys (CMKs) by default. Customer managed CMKs can be used to get more control on encryption and key management. For studio notebooks, the ML-related data is primarily stored in multiple locations. An S3 bucket hosts notebook snapshots and metadata, EFS volumes contain studio notebook and data files, and EBS volumes are attached to the instance that the notebook runs on. KMS can be used for encrypting all these storage locations. Encryption keys can be specified to encrypt the volumes of all Amazon EC2-based SageMaker resources, such as processing jobs, notebooks, training jobs, and model endpoints. FIPS endpoints can be used if FIPS 140-2 validated cryptographic modules are required to access AWS through a command line interface or an API.
- **Protect data in transit:** To protect data in transit, AWS makes extensive use of HTTPS communication for its APIs. Requests to the SageMaker API and console are made over a secure (SSL) connection. In addition to passing all API calls through a TLS-encrypted channel, AWS APIs also require that requests are signed using the Signature Version 4 signing process. This process uses client access keys to sign every API request, adding authentication information as well as preventing tampering of the request in flight.
  Additionally, communication between instances in a distributed training job can be further protected and another level of security can be added to protect your training containers and data by configuring a private VPC. SageMaker can be instructed to encrypt inter-node communication automatically for the training job. The data passed between nodes is then passed over an encrypted tunnel without the algorithm having to take on responsibility for encrypting and decrypting the data.

- **Secure shared notebook instances:** SageMaker notebook instances are designed to work best for individual users. They give data scientists and other users the most power for managing their development environment. A notebook instance user has root access for installing packages and other pertinent software. The recommended best practice is to use IAM policies when granting individuals access to notebook instances that are attached to a VPC that contains sensitive information. For example, allow only specific users access to a notebook instance with an IAM policy.

Authentication and Authorization

AWS IAM enables control of access to AWS resources. IAM administrators control who can be authenticated (signed in) and authorized (have permissions) to use SageMaker resources. IAM can help create preventive controls for many aspects of your ML environment, including access to Amazon SageMaker resources, data in Amazon S3, and API endpoints. AWS services can be accessed using a RESTful API, and every API call is authorized by IAM. Explicit permissions can be granted through IAM policy documents, which specify the principal (who), the actions (API calls), and the resources (such as Amazon S3 objects) that are allowed, as well as the conditions under which the access is granted. Access can be controlled by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. Two common ways to implement least privilege access to the SageMaker environments are identity-based policies and resource-based policies:

- Identity-based policies are attached to an IAM user, group, or role. These policies specify what that identity can do. For example, by attaching the AmazonSageMakerFullAccess managed policy to an IAM role for data scientists, they are granted full access to the SageMaker service for model development work.
- Resource-based policies are attached to a resource. These policies specify who has access to the resource, and what actions can be performed on it. For example, a policy can be attached to an Amazon Simple Storage Service (Amazon S3) bucket, granting read-only permissions to data scientists accessing the bucket from a specific VPC endpoint. Another typical policy configuration for S3 buckets is to deny public access, to prevent unauthorized access to data.

Please refer to Configuring Amazon SageMaker Studio for teams and groups with complete resource isolation, which outlines how to configure access control for teams or groups within Amazon SageMaker Studio using attribute-based access control (ABAC). ABAC is a powerful approach that can be utilized to configure Studio so that different ML and data science teams have complete isolation of team resources. AWS Single

Sign-On (AWS SSO) can also be used for user authentication with an external identity provider such as Ping identity or Okta. Please refer to Onboarding Amazon SageMaker Studio with AWS SSO and Okta Universal Directory, which outlines how to onboard SageMaker Studio with SSO and Okta universal directory.

Artifact and model management

The recommended best practice is to use version control to track code or other model artifacts. If model artifacts are modified or deleted, either accidentally or deliberately, version control allows you to roll back to a previous stable release. This can be used in cases where an unauthorized user gains access to the environment and makes changes to the model. If model artifacts are stored in Amazon S3, versioning should be enabled. S3 versioning should also be paired with multi-factor authentication (MFA) delete, to help ensure that only users authenticated with MFA can permanently delete an object version, or change the versioning state of the bucket. Another way of enabling version control is to associate Git repositories with new or existing SageMaker notebook instances. SageMaker supports AWS CodeCommit, GitHub, and other Git-based repositories. Using CodeCommit, repository can be further secured by rotating credentials and enabling MFA.

Additionally, the SageMaker Model registry can also be used to register, deploy, and manage models as discussed in SageMaker Pipelines in the MLOps section earlier.

Security compliance

Third-party auditors assess the security and compliance of Amazon SageMaker as part of multiple AWS compliance programs including FedRAMP, HIPAA, and others. For a list of AWS services in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. Third-party audit reports can be downloaded using AWS Artifact. The customer's compliance responsibility when using Amazon SageMaker is determined by the sensitivity of the Organization's data, its compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying security- and compliance-focused baseline environments on AWS.

- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how organizations can use AWS to help create HIPAA-compliant applications.
- AWS Compliance Resources – This collection of workbooks and guides might apply to the Organization's industry and location.
- AWS Config – This AWS service assesses how well resource configurations comply with internal practices, industry guidelines, and regulations. As an example, AWS Config can be used to create compliance rules that can scan AWS Key Management Service (AWS KMS) key policies to determine whether these policies align with the principle of granting least privilege to users. Please refer to the How to use AWS Config to determine compliance of AWS KMS key policies to your specifications, which outlines this process.
- AWS Security Hub – This AWS service provides a comprehensive view of the security state within AWS that helps check compliance with security industry standards and best practices.

Cost optimization

Cost management is a primary concern for public sector organizations projects to ensure the best use of public funds while enabling agency missions. AWS provides several mechanisms to manage costs in each phase of the ML lifecycle (Prepare, Build, Train & Tune, Deploy, and Manage) as described in this section.

Prepare

This step of the ML lifecycle includes storing the data, labeling the data, and processing the data. Cost control in this phase can be accomplished using the following techniques:

- **Data Storage:** ML requires extensive data exploration and transformation. Multiple redundant copies of data are quickly generated, which can lead to exponential growth in storage costs. Therefore, it is essential to establish a cost control strategy at the storage level. Processes can be established to regularly analyze source data and either remove duplicative data or archive data to lower cost storage based on compliance policies. For example, for data stored in S3, S3 storage class analysis can be enabled on any group of objects (based on prefix or object tagging) to automatically analyze storage access patterns. This enables identification and transition of rarely-accessed data to S3 glacier, lowering costs. S3 intelligent storage can also be used to lower costs of data that has unpredictable usage patterns. It works by monitoring and moving data between a data tier that is optimized for frequent access and another lower-cost tier that is optimized for infrequent access.
- **Data Labeling.** Data labeling is a key process of identifying raw data (such as images, text files, and videos) and adding one or more meaningful and informative labels to provide context so that an ML model can learn from it. This process can be very time consuming and can quickly increase costs of a project.

  Amazon SageMaker Ground Truth can be used to reduce these costs. Ground Truth's automated data labeling utilizes the Active Learning ML technique to reduce the number of labels required for models, thereby lowering these costs. Ground Truth also provides additional mechanisms such as crowdsourcing with Amazon Mechanical Turk or another vendor company, that can be chosen to lower the costs of labeling.
- **Data Wrangling.** In ML, a lot of time is spent in identifying, converting, transforming, and validating raw source data into features that can be used to train models and make predictions. Amazon SageMaker Data Wrangler can be used to reduce this time spent, lowering the costs of the project. With Data Wrangler, data can be imported from various data sources, and transformed without requiring coding. Once data is prepared, fully automated ML workflows can be built with Amazon SageMaker Pipelines and saved for reuse in the Amazon SageMaker Feature Store, eliminating the costs incurred in preparing this data again.

Build

This step of the ML lifecycle involves building ML models. Cost control in this phase can be accomplished using the following techniques:

- **Notebook Utilization.** An Amazon SageMaker notebook instance is a ML compute instance running the Jupyter Notebook. It helps prepare and process data, write code to train models, deploy models to SageMaker hosting, and test or validate models. Costs incurred can be reduced significantly by optimizing notebook utilization. One way is to stop the notebook instance when it's not being used and starting it up only when needed. Another option is to use a lifecycle configuration script that automatically shuts down the instance when not being worked on. (See Right-sizing resources and avoiding unnecessary costs in Amazon SageMaker for details.)
- **Test code locally.** The SageMaker Python SDK supports local mode, which allows creation of estimators and deployment to the local environment. Before a training job is submitted, running the fit function in local mode enables early feedback prior to running in SageMaker's managed training or hosting environments. Issues with code and data can be resolved early to reduce costs incurred in failed training jobs. This also saves time spent in initializing the training cluster.
- **Use Pipe mode (where applicable) to reduce training time.** Certain algorithms in Amazon SageMaker, such as Blazing text, work on a large corpus of data. When these jobs are launched, significant time goes into downloading the data from Amazon S3 into Amazon EBS. Training jobs don't start until this download finishes.

  These algorithms can take advantage of Pipe mode, in which training data is streamed from Amazon S3 into Amazon EBS to start training jobs immediately.
- **Find the right balance: Performance vs. accuracy.** 32-bit (single precision or FP32) and even 64-bit (double precision or FP64) floating point variables are popular for many applications that require high precision. These are workloads such as engineering simulations that simulate real-world behavior and need the mathematical model to be as exact as possible. In many cases, however, moving to half or mixed precision (16-bit or FP16) reduces training time and consequently costs less, and is worth the minor tradeoffs in accuracy. See this Accelerating GPU computation through mixed-precision methods for details. A similar trade-off also applies when deciding on the number of layers in a neural network for classification algorithms, such as image classification. Throughput of 16-bit floating point and 32-bit floating point calculations need to be compared to determine an appropriate approach for the model in question.

- **Jumpstart:** Developers who are new to ML often learn that importing an ML model from a third-party source and getting an API endpoint up and running to deploy the model can be time-consuming. The end-to-end process of building a solution, including building, training, and deploying a model, and assembling different components, can take months for users new to ML. SageMaker JumpStart accelerates time-to-deploy over 150 open-source models and provides pre-built solutions, preconfigured with all necessary AWS services required to launch the solution into production, including CloudFormation templates and reference architecture.
- **AWS Marketplace:** AWS Marketplace is a digital catalog with listings from independent software vendors to find, test, buy, and deploy software that runs on AWS. AWS Marketplace provides many pre-trained, deployable ML models for SageMaker. Pre-training the models enables the delivery of ML-powered features faster and at a lower cost.

Train and Tune

This step of the ML lifecycle involves providing the algorithm selected in the build phase with the training data to learn from, and setting the model parameters to optimize the training process. Cost control in this phase can be accomplished using the following techniques:

- **Use Spot Instances.** If the training job can be interrupted, Amazon SageMaker Managed spot training can be used to optimize the cost of training models up to 90% over On-Demand Instances. Training jobs can be configured to use Spot Instances and a stopping condition can be used to specify how long Amazon SageMaker waits for a job to run using EC2 Spot Instances. See this Managed Spot Training: Save Up to 90% On Your Amazon SageMaker Training Jobs for details.
- **Hyperparameter optimization (HPO).** Amazon SageMaker's built-in HPO automatically adjusts hundreds of different combinations of parameters to quickly arrive at the best solution for your ML problem. When combined with high-performance algorithms, distributed computing, and managed infrastructure, built-in HPO drastically decreases the training time and overall cost of building production-grade systems. Built-in HPO works best with a reduced search space.

- **CPU vs GPU.** CPUs are best at handling single, more complex calculations sequentially, whereas GPUs are better at handling multiple but simple calculations in parallel. GPUs provide a great price/performance ratio if effectively used. However, GPUs also cost more, and should be chosen only when really needed. For many use cases, a standard current generation instance type from an instance family such as ml.m* provides enough computing power, memory, and network performance for many Jupyter notebooks to perform well. A best practice is to start with the minimum requirement in terms of ML instance specification and work up to identifying the best instance type and family for the model in question.
- **Distributed Training.** When using massive datasets for training, the process can be sped up by distributing training on multiple machines or processes in a cluster as described earlier. Another option is to use a small subset of data for development, and use the full dataset for a training job that is distributed across optimized instances such as P2 or P3 GPU instances or an instance with powerful CPU, such as c5.
- **Monitor the performance of your training jobs to identify waste.** Amazon SageMaker is integrated with CloudWatch out of the box and publishes instance metrics of the training cluster in CloudWatch. These metrics enable adjustments to the cluster, such as CPUs, memory, number of instances, and more. Also, Amazon SageMaker Debugger provides full visibility into model training by monitoring, recording, analyzing, and visualizing training process tensors. Debugger can reduce the time, resources, and cost needed to train models.

Deploy and Manage

This step of the ML lifecycle involves deployment of the model to get predictions, and managing the model to ensure it meets functional and non-functional requirements of the application. Cost control in this phase can be accomplished using the following techniques:

- **Endpoint deployment:** Amazon SageMaker enables testing of new models using A/B testing. Endpoints need to be deleted when testing is completed to reduce costs. These can be recreated from S3 if and when needed. Endpoints that are not deleted can be automatically detected by using EventBridge / CloudWatch Events and Lambda functions. For example, you can detect if endpoints have been idle (with no invocations over a certain period, such as 24 hours), and send an email or text message with the list of detected idle endpoints using SNS. See this Right-sizing resources and avoiding unnecessary costs in Amazon SageMaker for details.

- **Multi-model endpoints.** SageMaker endpoints provide the capability to host multiple models. Multi-model endpoints reduce hosting costs by improving endpoint utilization, and provide a scalable and cost-effective solution to deploying a large number of models. Multi-model endpoints enable time-sharing of memory resources across models. It also reduces deployment overhead because Amazon SageMaker loads models in memory and scales them based on traffic patterns.
- **Auto Scaling.** Amazon SageMaker Auto Scaling optimizes the cost of model endpoints. Auto Scaling automatically increases the number of instances to handle increase in load (scale out) and decreases the number of instances when not needed (scale in), thereby reducing operational costs. The endpoint can be monitored to adjust the scaling policy based on the CloudWatch metrics. (See Load test and optimize an Amazon SageMaker endpoint using automatic scaling for details).
- **Amazon Elastic Inference for deep learning.** For inferences, a deep learning application may not fully utilize the capacity offered by a GPU. Using Amazon Elastic Inference allows the attachment of low-cost GPU-powered acceleration to Amazon EC2 and Amazon SageMaker instances to reduce the cost of running deep learning inference by up to 75%.
- **Analyzing costs with Cost Explorer.** Cost Explorer is a tool that enables viewing and analyzing AWS service-related costs and usage including SageMaker. Cost allocation tags can be used to get views of costs aggregated across specific views, such as a project. To accomplish this, all Amazon SageMaker project-related resources, including notebook instances and the hosting endpoint, can be tagged with user-defined tags. For example, tags can be the name of the project, business unit, or environment (such as development, testing, or production). After user-defined tags have been defined and created, they will need to be activated in the Billing and Cost Management console for cost allocation tracking. These tags can then be used to get different views of costs using Cost Explorer as well as Cost and Usage Reports (Cost Allocation Tags appear on the console after Cost Explorer, Budgets, and AWS Cost and Usage Reports have been enabled).

- **AWS Budgets.** AWS Budgets help you manage Amazon SageMaker costs, including development, training, and hosting, by setting alerts and notifications when cost or usage exceeds (or is forecasted to exceed) the budgeted amount. After a budget is created, progress can be tracked on the AWS Budgets console. AWS Service Catalog can be integrated with AWS Budgets to create and associate budgets with portfolios and products, and keep developers informed on resource costs for running cost-aware workloads. See Cost Control Blog Series #2: Automate Cost Control using AWS Service Catalog and AWS Budgets for details.

Bias and Explainability

Demonstrating explainability is a significant challenge because complex ML models are hard to understand and even harder to interpret and debug. There is an inherent tension between ML performance (predictive accuracy) and explainability; often the highest performing methods are the least explainable, and the most explainable are less accurate. Hence, public sector organizations need to invest significant time with appropriate tools, techniques, and mechanisms to demonstrate explainability and lack of bias in their ML models, which could be a deterrent to adoption.

AWS Cloud provides the following capabilities and services to assist public sector organizations in resolving these challenges.

Amazon SageMaker Debugger

Amazon SageMaker Debugger provides visibility into the model training process for real-time and offline analysis. In the existing training code for TensorFlow, Keras, Apache MXNet, PyTorch, and XGBoost, the new SageMaker Debugger SDK can be used to save the internal model state at periodic intervals in S3. This state is composed of a number of components: The parameters being learned by the model (for example, weights and biases for neural networks), the changes applied to these parameters by the optimizer (gradients), optimization parameters, scalar values such as accuracies and losses, and outputs of each layer of a neural network.

SageMaker Debugger provides three built-in tensor collections called feature importance, average_shap, and full_shap, to visualize and analyze captured tensors specifically for model explanation. Feature importance is a technique that explains the features that make up the training data using a score (importance). It indicates how useful or valuable the feature is, relative to other features.

SHAP (SHapley Additive exPlanations) is an open-source technique based on

coalitional game theory. It explains an ML prediction by assuming that each feature value of training data instance is a player in a game in which the prediction is the payout. Shapley values indicate how to distribute the payout fairly among the features. The values consider all possible predictions for an instance and use all possible combinations of inputs. Because of this exhaustive approach, SHAP can guarantee consistency and local accuracy. For more information, see the SHAP website.

SHAP values can be used for global explanatory methods to understand the model and its feature contributions in aggregate over multiple data points. SHAP values can also be used for local explanations that focus on explaining each individual prediction. See ML Explainability with Amazon SageMaker Debugger for details.

Amazon SageMaker Clarify

Amazon SageMaker Clarify is a service that is integrated into SageMaker Studio and detects potential bias during data preparation, model training, and in deployed models, by examining specified attributes. For instance, bias in attributes related to age can be examined in the initial dataset, in the trained as well as the deployed model, and quantified in a detailed report. Clarify provides a range of metrics to measure bias such as Difference in positive proportions in labels (DPL), Difference in positive proportions in predicted labels (DPPL), Accuracy difference (AD), and Counterfactuals – Fliptest (FT). In addition, SageMaker Clarify also enables explainability by including feature importance graphs using SHAP to help explain model predictions. It produces reports and visualizations that can be used to support internal presentations on a model's predictions. See New – Amazon SageMaker Clarify Detects Bias and Increases the Transparency of Machine Learning Models for details. Clarify has been designed to work without burdening the inference operations – assessment of a model can be spun off as a separate activity in SageMaker. This capability is very helpful to automate monitoring drift.

SHAP and LIME (Local Interpretable Model-Agnostic Explanations) libraries:

In case team members are unable to use Amazon SageMaker Debugger or Amazon SageMaker Clarify for explainability and bias, their libraries can directly be installed on SageMaker Jupyter instances or Studio Notebooks and incorporated into the training code See Explaining Amazon SageMaker Autopilot models with SHAP for details on using SHAP. LIME provides a model-agnostic approach for setting up explanations; LIME builds sparse linear models around each prediction to explain how the black box model works in that local vicinity. SHAP is a more cost-intensive process as it requires

more compute time calculating all the probable combinations and permutations of features for explaining predictions compared to LIME.

Conclusion

US Public sector organizations have complex mission objectives and are increasingly adopting ML services to help with their initiatives. ML can transform the way government agencies operate, and enable them to provide improved citizen services. However, several barriers remain for these organizations to implement ML. This whitepaper outlined some of the challenges and provided best practices that can help address these challenges using AWS Cloud.

Next Steps

Adopting the AWS Cloud can provide you with sustainable advantages for telehealth systems. Your AWS account team can work together with your team and/or your chosen member of the AWS Partner Network (APN) to implement your enterprise cloud computing initiatives. You can reach out to an AWS partner through the AWS Partner Network. Get started on AI and ML by visiting AWS ML, AWS ML Embark Program, or the ML Solutions Lab.

References to Public Sector Use Cases

The following list provides some examples of public sector use cases for AI/ML in AWS. For a more comprehensive list, refer to the AWS Blog.

1) https://www.amazon.science/how-nasa-uses-aws-to-protect-life-and-infrastructure-on-earth
2) https://www.amazon.science/blog/paper-on-forecasting-spread-of-covid-19-wins-best-paper-award
3) https://www.amazon.science/blog/amazon-supports-nsf-research-in-human-ai-interaction-collaboration
4) https://aws.amazon.com/blogs/machine-learning/fine-tune-and-deploy-the-protbert-model-for-protein-classification-using-amazon-SageMaker/
5) https://aws.amazon.com/blogs/publicsector/using-ai-rethink-document-automation-extract-insights/
6) https://aws.amazon.com/blogs/publicsector/chesterfield-county-public-schools-uses-machine-learning-predict-countys-chronic-absenteeism/

7) https://aws.amazon.com/blogs/publicsector/using-advanced-analytics-accelerate-problem-resolution-public-sector/
8) https://aws.amazon.com/blogs/publicsector/how-ai-and-ml-are-helping-tackle-the-global-teacher-shortage/
9) https://aws.amazon.com/blogs/publicsector/improving-school-safety-how-cloud-helping-k12-students-wake-violence/
10) https://aws.amazon.com/blogs/publicsector/heading-into-hurricane-season/
11) https://aws.amazon.com/blogs/publicsector/helping-to-end-future-famines-with-machine-learning/

Contributors

Contributors to this document include:

- Mona Mona, Sr. AI/ML Specialist Solutions Architect
- Sam Palani, Sr. AI/ML Specialist Solutions Architect
- Sanjeev Pulapaka, Sr. Solutions Architect, AWS WWPS
- Sherry Ding, Sr. AI/ML Specialist Solutions Architect
- Srinath Godavarthi, Sr. Solutions Architect, AWS WWPS

Further Reading

For additional information, see:

- Machine Learning on AWS
- AWS ML Blog

Document Revisions

Date	Description
September 2021	First publication

Notes

[1] https://www.nitrd.gov/pubs/National-AI-RD-Strategy-2019.pdf

[2] https://www.darpa.mil/work-with-us/ai-next-campaign

[3] https://www.nsf.gov/cise/ai.jsp

[4] https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/?sh=2fb540636f63