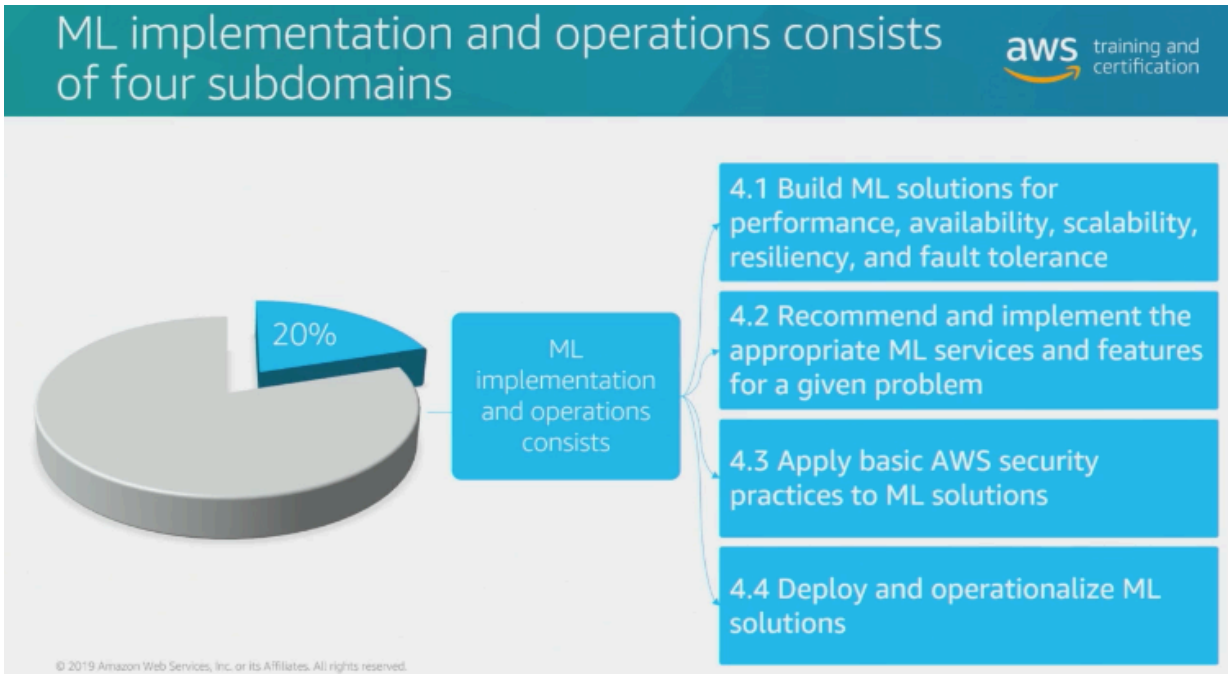
AWS Exam Readiness - Domain 4 - ML Implementation and Operations

Module 6 of 9

Domain 4: ML Implementation and Operations



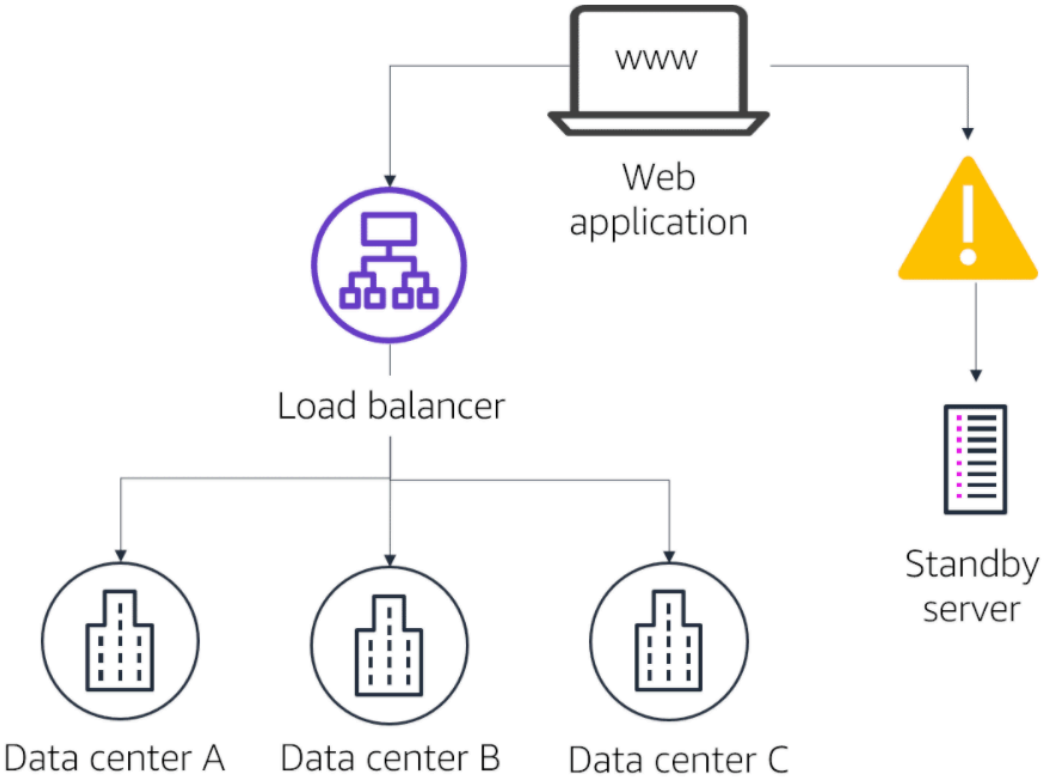
Domain 4.1: Build ML solutions for performance, availability, scalability, resiliency, and fault tolerance

Part of this domain is focused on deploying your ML solution into production. But before you walk through the steps of model deployment, you have to ensure your solution is designed to effectively deal with operational failure. This subdomain focuses on best practices for how to do this in the context of machine learning.

High availability and fault tolerance

At the heart of designing for failure are two concepts known as high availability and fault tolerance.

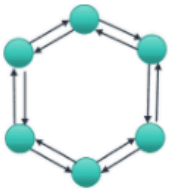
In a highly available solution, the system will continue to function even when any component of the architecture stops working. A key aspect of high availability is fault tolerance, which, when built into an architecture, ensures that applications will continue to function without degradation in performance, despite the complete failure of any component of the architecture.



One method of achieving high availability and fault tolerance is loose coupling

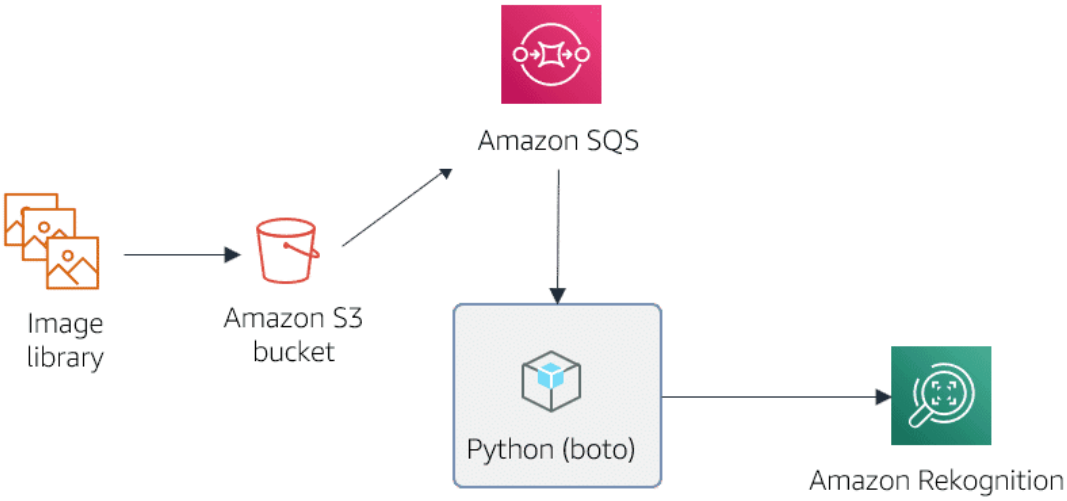
With a loosely coupled, distributed system, the failure of one component can be managed in between your application tiers so that the faults do not spread beyond that single point. Loose coupling is often achieved by making sure application components are independent of each other. For example, you should always decouple your storage layer with your compute layer because a training job only requires minimal time, but storing data is permanent. Decoupling helps turn off the compute resources when they are not needed.

Click on the image below to learn more about the differences between loosely and tightly coupled systems.



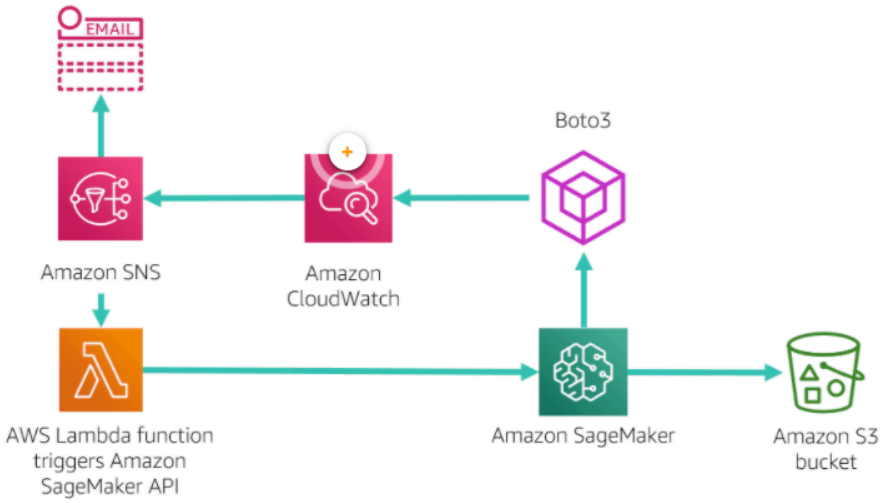
Queues are used in loose coupling to pass messages between components

In a general architecture, you can use a queue service like Amazon SQS or workflow service like AWS Step Functions to create a workflow between various components.



Amazon CloudWatch helps you monitor your system

Services like Amazon CloudWatch help you monitor your system while storing all the logs and operational metrics separately from the actual implementation and code for training and testing your ML models. In this example, Amazon CloudWatch is used to keep a history of the model metrics for a specific amount of time, visualize model performance metrics, and create a CloudWatch dashboard. Amazon SageMaker provides out-of-the-box integration with Amazon CloudWatch, which collects near-real-time utilization metrics for the training job instance, such as CPU, memory, and GPU utilization of the training job container.



Use CloudWatch to monitor your AWS resources and applications

Amazon CloudWatch Logs enables you to monitor, store, and access your log files from Amazon EC2 instances

Amazon CloudWatch Events delivers a near-real-time stream of system events that describe changes in AWS resources

Amazon CloudWatch alarms allows you to watch CloudWatch metrics and to receive notifications when the metrics fall outside of the levels (high or low thresholds) that you configure

AWS CloudTrail captures API calls and related events

AWS CloudTrail

AWS CloudTrail captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred.

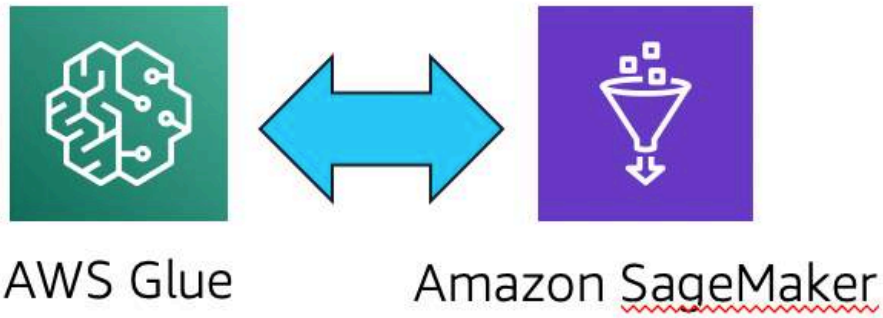
AWS CloudTrail stores event up to 90d by default

You can design for failure of any individual component by leveraging key AWS services and features

AWS GLUE AND AMAZON EMR	AMAZON SAGEMAKER ENDPOINTS	AMAZON SAGEMAKER	AWS AUTO SCALING

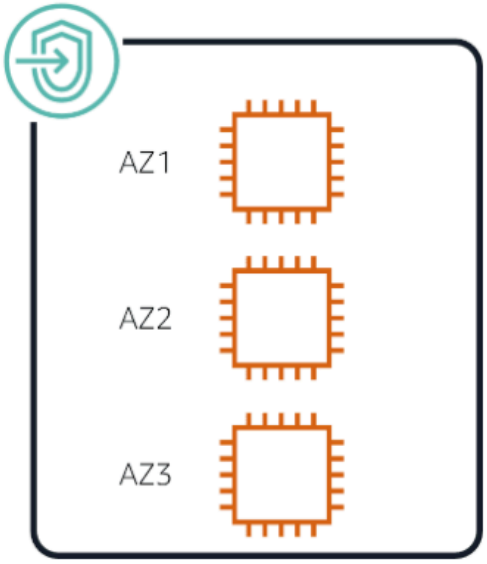
You should decouple your ETL process from the ML pipeline. The compute power needed for ML isn't the same as what you'd need for an ETL process—they have very different requirements.

- An ETL process needs to read in files from multiple formats, transform them as needed, and then write them back to a persistent storage. Keep in mind that reading and writing takes a lot of memory and disk I/O, so when you decouple your ETL process, use a framework like Apache Spark, which can handle large amounts of data easily for ETL.
- Training, on the other hand, may require GPUs which are much more suited to handle the training requirements than CPUs. However, GPUs are less cost-effective to keep running when a model is not being trained. So you can make use of this decoupled architecture by simply using an ETL service like AWS Glue or Amazon EMR, which use Apache Spark for your ETL jobs and Amazon SageMaker to train, test, and deploy your models.



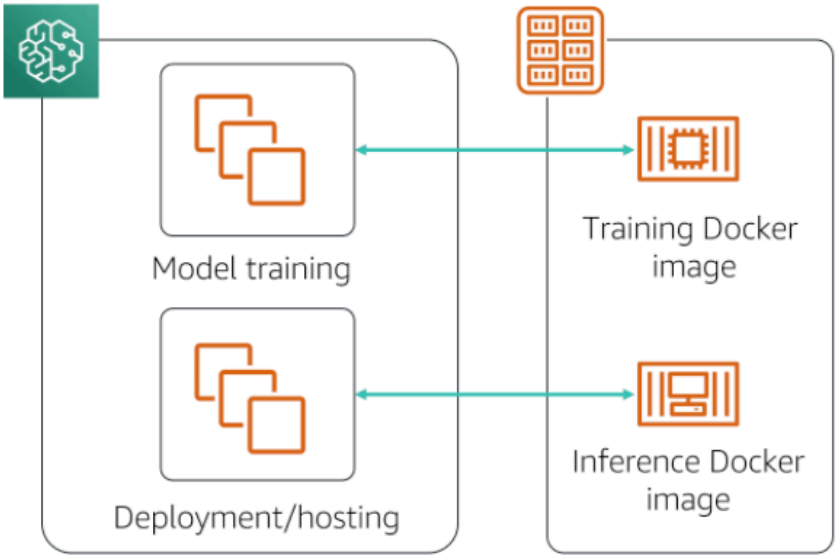
AWS GLUE AND AMAZON EMR | AMAZON SAGEMAKER ENDPOINTS | AMAZON SAGEMAKER | AWS AUTO SCALING

To ensure a highly available ML serving endpoint, deploy Amazon SageMaker endpoints backed by multiple instances across Availability Zones.



AWS GLUE AND AMAZON EMR | AMAZON SAGEMAKER ENDPOINTS | AMAZON SAGEMAKER | AWS AUTO SCALING

Amazon SageMaker makes it easy to containerize ML models for both training and inference. In doing so, you can create ML models made up of loosely coupled, distributed services that can be placed on any number of platforms, or close to the data that the applications are analyzing.



Use AWS Auto Scaling to build scalable solutions by configuring automatic scaling for the AWS resources such as Amazon SageMaker endpoints that are part of your application in response to the changes in traffic to your application.

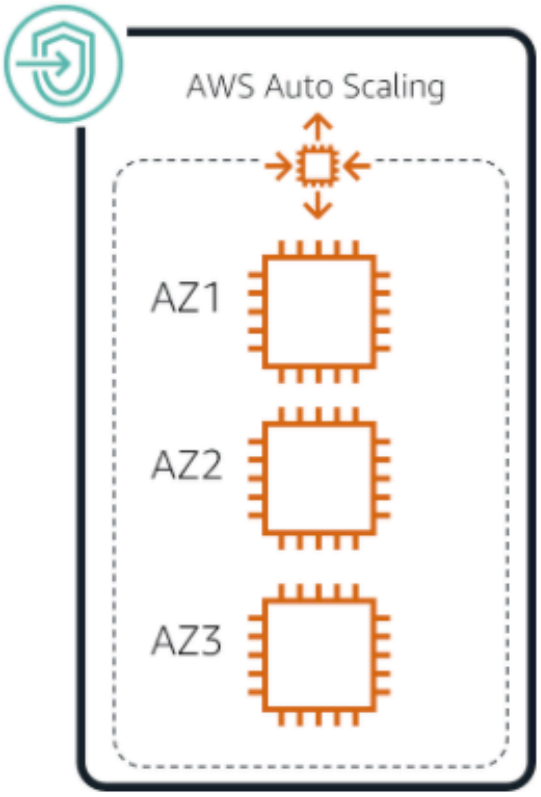
With AWS Auto Scaling, you configure and manage scaling for your resources through a scaling plan. The scaling plan uses dynamic scaling and predictive scaling to automatically scale your application's resources.

This ensures that you add the required computing power to handle the load on your application, and then remove it when it's no longer required. The scaling plan lets you choose scaling strategies to define how to optimize your resource utilization. You can optimize for availability, for cost, or a balance of both.

As you increase the number of concurrent prediction requests, at some point the endpoint responds more slowly and eventually errors out for some requests. Automatically scaling the endpoint avoids these problems and improves prediction throughput. When the endpoint scales out, Amazon SageMaker automatically spreads instances across multiple Availability Zones. This provides Availability Zone-level fault tolerance and protects from an individual instance failure.

If the endpoint has only a moderate load, you can run it on a single instance and still get good performance. Use automatic scaling to ensure high availability during traffic fluctuations without having to constantly provision for peak traffic. For production workloads, use at least two instances. Because Amazon SageMaker automatically spreads the endpoint instances across multiple Availability Zones, a minimum of two instances ensures high availability and provides individual fault tolerance.

To determine the scaling policy for automatic scaling in Amazon SageMaker, test for how much load (RPS) the endpoint can sustain. Then configure automatic scaling and observe how the model behaves when it scales out. Expected behavior is lower latency and fewer or no errors with automatic scaling.

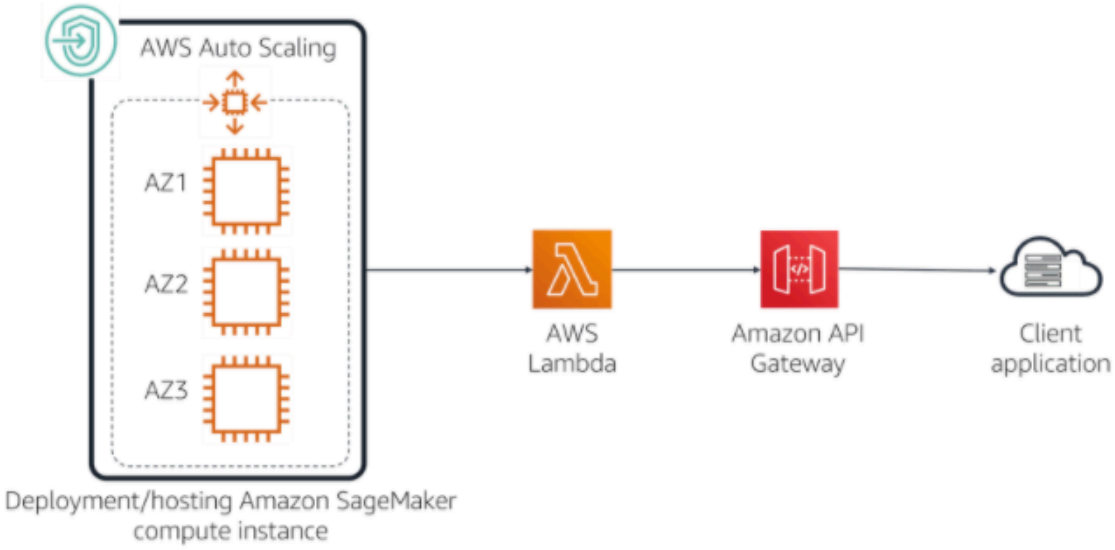


Because Amazon SageMaker automatically spreads the endpoint instances across multiple Availability Zones, a minimum of two instances ensures high availability and provides individual fault tolerance.

Designing highly available and fault-tolerant ML architectures

The example below brings together these and some other best practices related to designing highly available and fault-tolerant ML architectures.

As previously mentioned, you can deploy your Amazon SageMaker-built models to an Amazon SageMaker endpoint. Once created, you need to invoke the endpoint outside the Amazon SageMaker notebook instance with appropriate input (the model signature). These input parameters can be in a file format such as CSV and LIBSVM, as well as an audio, image, or video file. You can use AWS Lambda and Amazon API Gateway to format the input request and invoke the endpoint from the web. The diagram on the slide shows the infrastructure architecture at this point.



Topics related to this subdomain

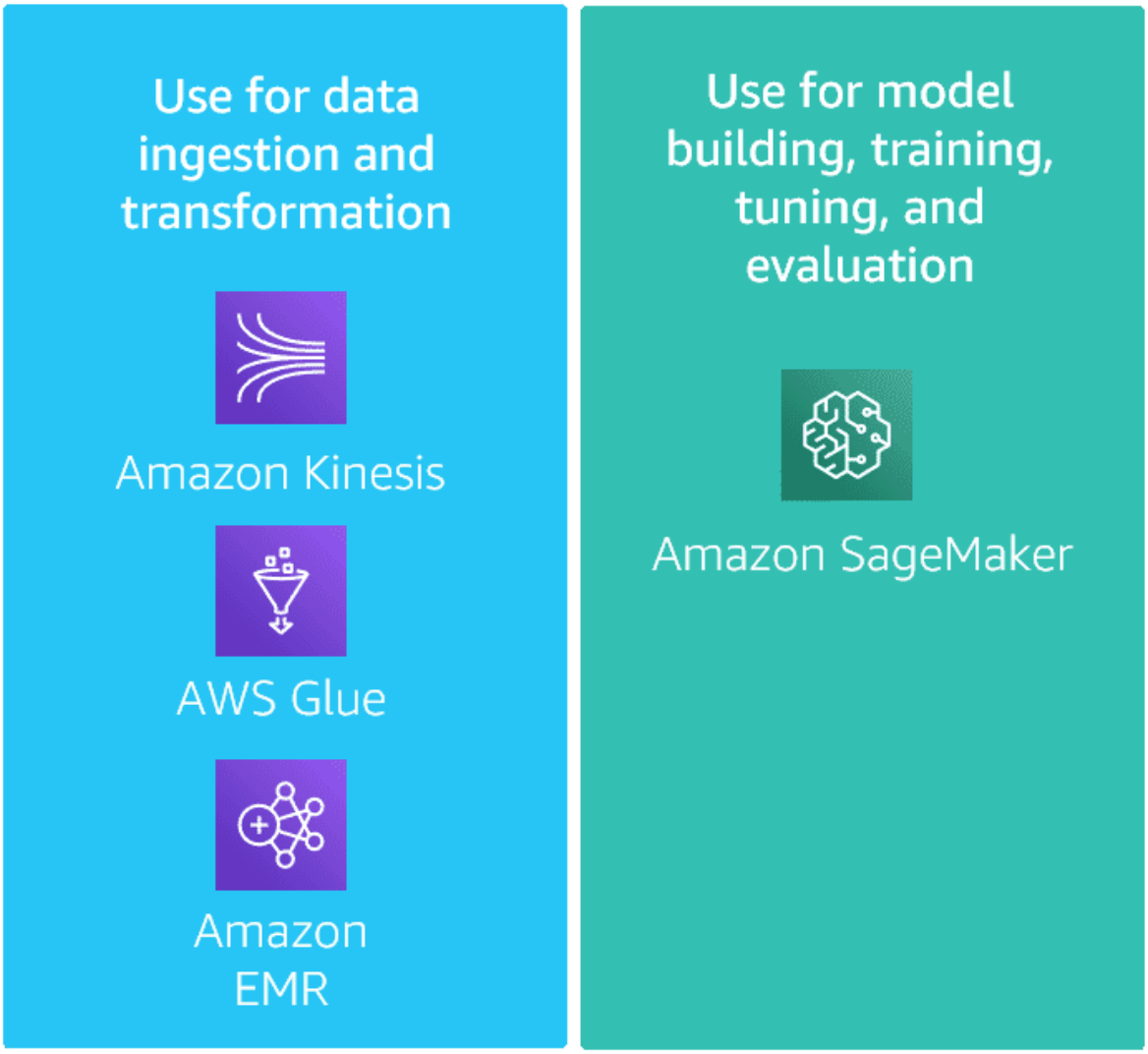
Here are some topics you may want to study for more in-depth information related to this subdomain:

- [ ] Amazon Deep Learning containers
- [ ] AWS Deep Learning AMI (Amazon Machine Image)
- [ ] AWS Auto Scaling
- [ ] AWS GPU (P2 and P3) and CPU instances
- [ ] Amazon CloudWatch
- [ ] AWS CloudTrail

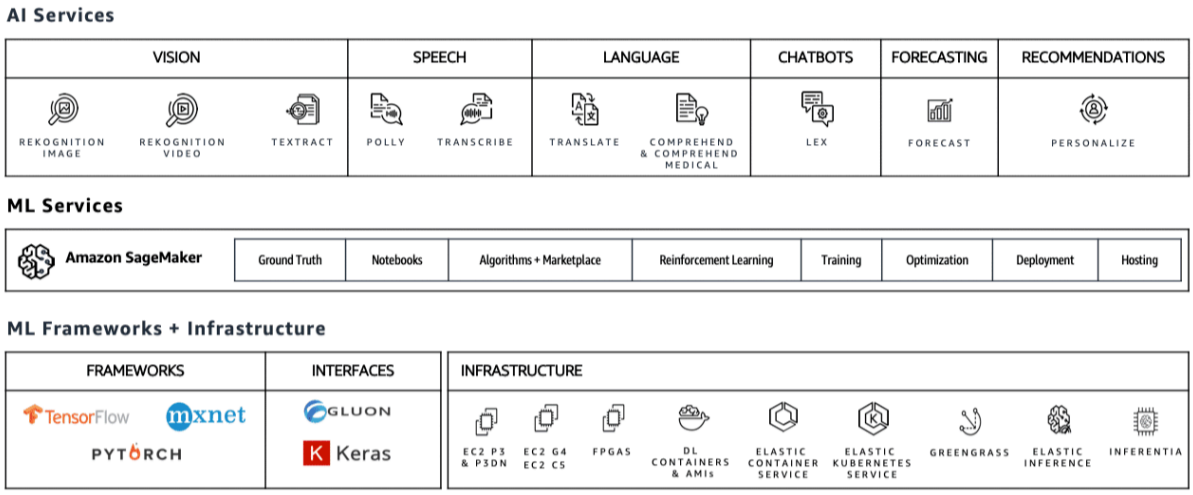
Domain 4.2: Recommend and implement the appropriate ML services and features for a given problem

Most of the exam domains have focused on at least one phase of the ML pipeline and, where applicable, the AWS services needed to perform related tasks and create relevant solutions. For instance, AWS Glue, Amazon EMR, and Amazon Kinesis were discussed as services used for data ingestion and transformation, while Amazon SageMaker was discussed as the service to use for model building, training, tuning, and evaluation.

This particular subdomain, however, pivots away from the ML pipeline and focuses more generally on the entire ecosystem of AWS ML services and their use cases.



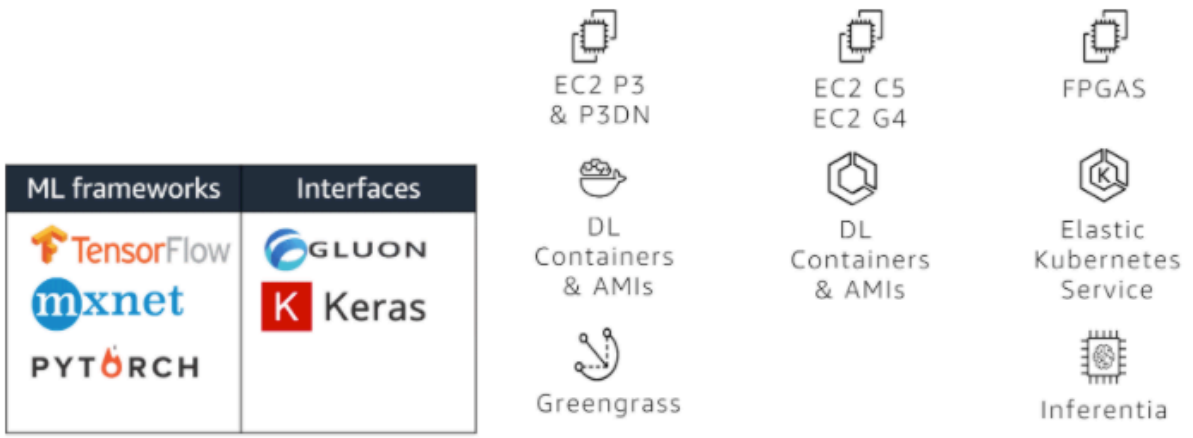
The stack for Amazon machine learning has three tiers



ML frameworks + infrastructure

The bottom tier of the stack is for expert ML practitioners who work at the framework level. To work with these frameworks, you are comfortable building, training, tuning, and deploying ML models on the metal, so to speak. ML frameworks are the foundation from which innovation in ML is designed. The focus here is on making it easier for you to connect more broadly to the AWS ecosystem, whether that's about pulling in IoT data from AWS IOT Greengrass, accessing state-of-the art chips (P3), or leveraging elastic inference.

The vast majority of deep learning and ML in the cloud is done on P3 instances in AWS. You can use whichever ML deep learning framework you like, but some popular options are TensorFlow, MXNet, and PyTorch, which are all supported on AWS.



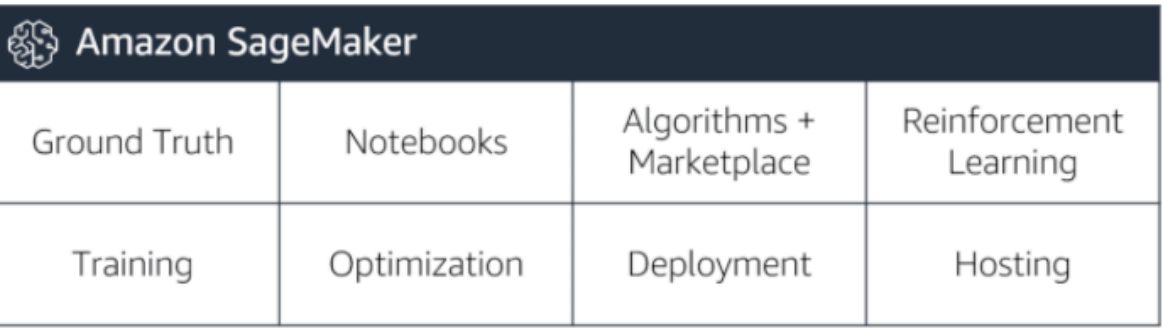
ML services —

While there is a lot of activity at this bottom layer, the reality is that there just aren't that many expert ML practitioners out there.

That's why the second tier on the stack, platform services, was created. At the heart of this tier is Amazon SageMaker, which we've discussed. While ML can provide tremendous business value, today the process for authoring, training, and deploying ML models has many challenges. Why? Because collecting, cleaning, and formatting the training data can be time-consuming, particularly if you're not using the latest tools.

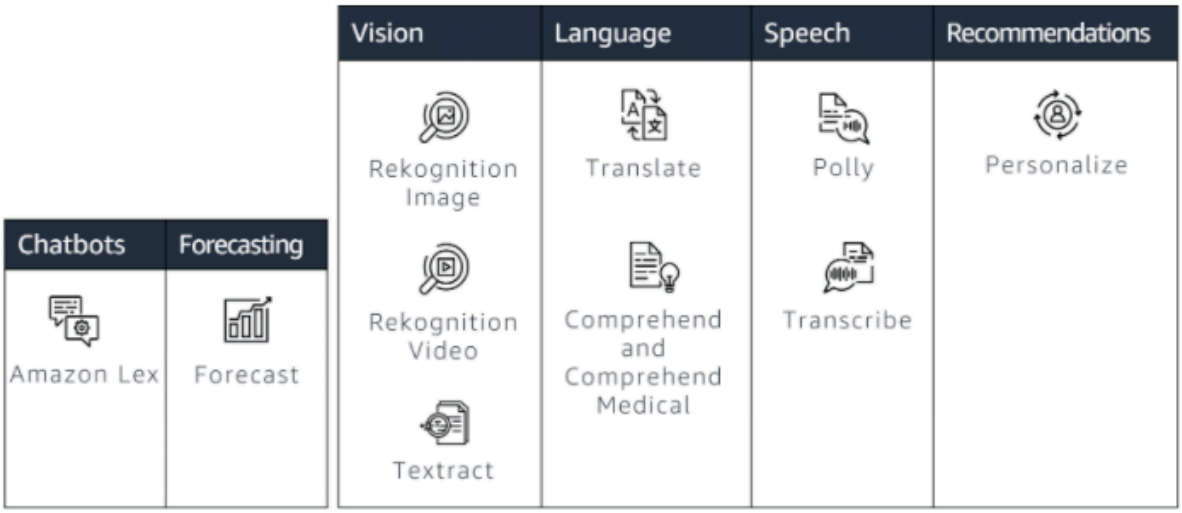
Once the training data set is created, you need to ensure that your algorithms and compute environments can quickly handle the scale of the data needed. Simply figuring out how to train increasingly complex models on increasingly larger data sets can be a blocker, and companies that frequently train models often have dedicated teams just to manage the training environments.

Once you're ready to move to production, a new set of challenges can come up. Often, the person developing the model hands it off to another team to begin the tedious process of figuring out how to run the model at scale. This involves an entirely different set of computer science challenges related to efficiently operating high-scale distributed systems. If you don't do this routinely, it can be a slow and cumbersome process. Amazon SageMaker was designed to address many of these fundamental challenges.



AI services

AWS services in the top tier are for customers who really don't want to deal with building and training their ML models. All of that has been abstracted away, leaving you with easy-to-use services designed to help you deal with common ML problems in various domains, like computer vision, NLP, and time series forecasting.



Topics related to this subdomain

Here are some topics you may want to study for more in-depth information related to this subdomain:

- [ ] Amazon SageMaker Spark containers
- [ ] Amazon SageMaker build your own containers
- [ ] Amazon AI services

  - Amazon Translate
  - Amazon Lex
  - Amazon Polly
  - Amazon Transcribe
  - Amazon Rekognition
  - Amazon Comprehend

Domain 4.3: Apply Basic AWS security practices to ML solutions

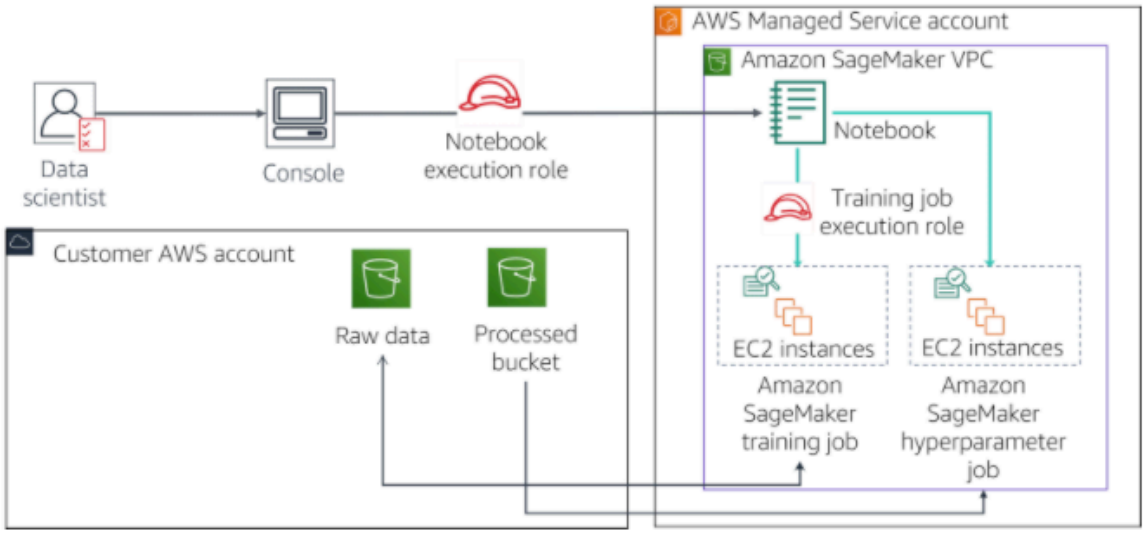
For any solution you build, including an ML solution, you need to make sure it and your data are secure. Applying security practices to your ML solutions in the cloud is what this subdomain is centered around.

This section will provide you with an overview of those security practices by walking you through an example involving Amazon SageMaker.

Security is intrinsically built into Amazon SageMaker

Security is intrinsically built in to Amazon SageMaker. With training data, Amazon SageMaker gets data from Amazon S3, passes that data to the training job environment, and then passes the generated model back to Amazon S3. This is done in the customer's account and is not saved in the Amazon SageMaker managed account. If you want to deploy the model, it is loaded into instances that are serving the model so that you can call the endpoint for prediction.

To keep this process secure, Amazon SageMaker supports IAM role-based access to secure your artifacts in Amazon S3, where you can set different roles for different parts of the process. For instance, a certain data scientist can have access to PII information in the raw data bucket, but the DevOps engineer only has access to the trained model itself. This approach helps you restrict access to the user(s) who need it. For the data scientist, you can use a notebook execution role for creating and deleting notebooks, and a training job execution role to run the training jobs.

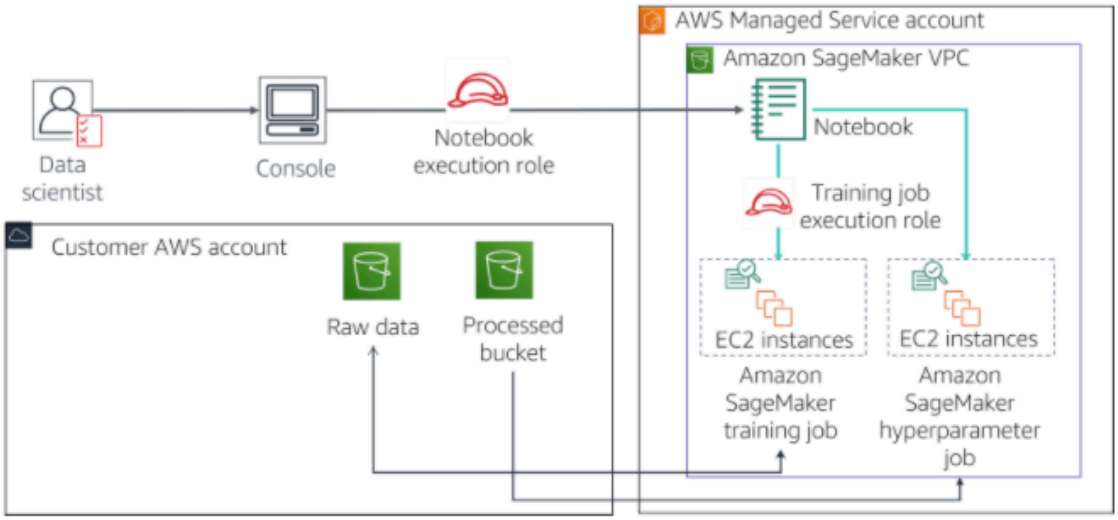


You can launch an Amazon SageMaker instance in a customer-managed VPC

When you create an Amazon SageMaker notebook instance, you can launch the instance with or without your Virtual Private Cloud (VPC) attached. When launched with your VPC attached, the notebook can either be configured with optional direct internet access.

You specify your private VPC configuration when you create a model by specifying subnets and security groups. When you specify the subnets and security groups, Amazon SageMaker creates elastic network interfaces that are associated with your security groups in one of the subnets.

Network interfaces allow your model containers to connect to resources in your VPC. For instances with direct internet access, Amazon SageMaker provides a network interface that allows for the notebook to talk to the internet through a VPC managed by the service. If you disable direct internet access, the notebook instance won't be able to train or host models unless your VPC has an interface endpoint (PrivateLink) or a NAT gateway and your security groups allow outbound connections.



Amazon SageMaker also encrypts data at rest

Along with IAM roles to prevent unwanted access, Amazon SageMaker also encrypts data at rest with either AWS Key Management Service (AWS KMS) or a transient key if the key isn't provided and in transit with TLS 1.2 encryption for the all other communication. Users can connect to the notebook instances using an AWS SigV4 authentication so that any connection remains secure. Any API call you make is executed over an SSL connection.

You can use encrypted Amazon S3 buckets for model artifacts and data

Similar to encrypting data in Amazon SageMaker, you can use encrypted Amazon S3 buckets for model artifacts and data, and pass an AWS KMS key to Amazon SageMaker notebooks, training jobs, hyperparameter tuning jobs, batch transform jobs, and endpoints, to encrypt the attached ML storage volume. If you do not specify an AWS KMS key, Amazon SageMaker encrypts storage volumes with a transient key. A transient key is discarded immediately after it is used to encrypt the storage volume.

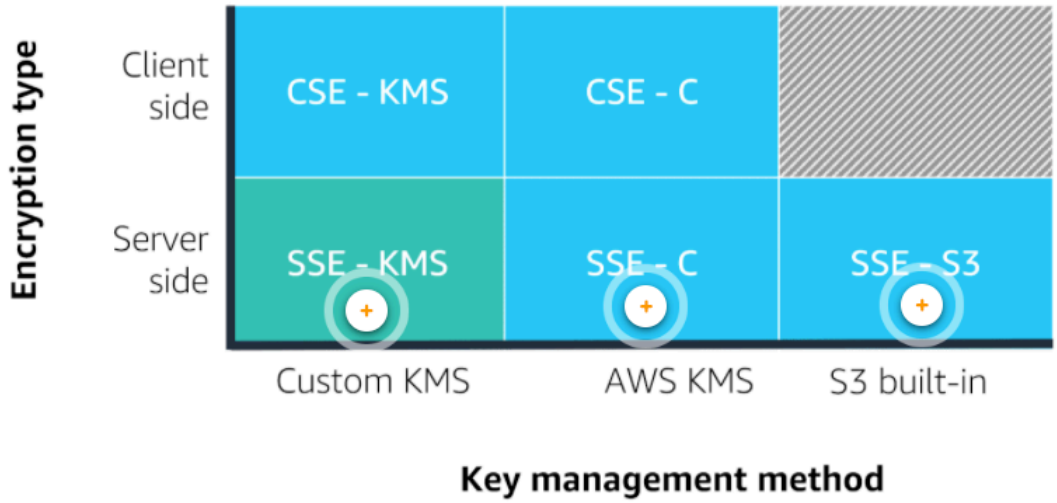
AWS KMS gives you centralized control over the encryption keys used to protect your data. You can create, import, rotate, disable, delete, define usage policies for, and audit the use of encryption keys used to encrypt your data. You specify a KMS key ID when you create Amazon SageMaker notebook instances, training jobs, or endpoints.

The attached ML storage volumes are encrypted with the specified key. You can specify an output Amazon S3 bucket for training jobs that is also encrypted with a key managed with AWS KMS, and pass in the KMS key ID for storing the model artifacts in that output S3 bucket.

If you do not specify an AWS KMS key, Amazon SageMaker encrypts storage volumes with a transient key. A transient key is discarded immediately after it is used to encrypt the storage volume.

There are two ways to use AWS KMS with Amazon S3

Click each marker below to review how you can protect data at rest in Amazon S3 by using three different modes of server-side encryption.



SSE-KMS

SSE-KMS requires that AWS manage the data key, but you manage the customer master key in AWS KMS.

SSE-S3

SSE-S3 requires that Amazon S3 manage the data and master encryption keys.

SSE-C

SSE-C requires that you manage the encryption key.

Below is a summary of security features integrated with Amazon SageMaker

Authentication

IAM federation

Gaining insight

Restrict access by IAM policy and condition keys

Audit

API logs to AWS CloudTrail - exception of InvokeEndpoint

Data protection at rest

AWS KMS-based encryption for:

- Notebooks
- Training jobs
- Amazon S3 location to store modelsEndpoint

Data protection at motion

HTTPS for:

- API/console
- Notebooks
- VPC-enabled
- Interface endpoint
- Limit by IPTraining jobs/endpoints

Compliance programs

- PCI DSS
- HIPAA-eligible with BAA
- ISO

Topics related to this subdomain

Here are some topics you may want to study for more in-depth information related to this subdomain:

- [ ] Security on Amazon SageMaker
- [ ] Infrastructure security on Amazon SageMaker
- [ ] What is a:
    - VPC
    - Security group
    - NAT gateway
    - Internet gateway
- [ ] AWS Key Management Service (AWS KMS)
- [ ] AWS Identity and Access Management (IAM)

Domain 4.4: Deploy and operationalize ML solutions

The ML model you develop is one component in a larger software ecosystem. All the usual software engineering and management practices must still be applied, including security, logging and monitoring, task management, API versioning, and so on. This ecosystem must be managed using cloud and software engineering practices, including:

- End-to-end and A/B testing
- API versioning, if multiple versions of the model are used
- Reliability and failover
- Ongoing maintenance
- Cloud infrastructure best practices, such as continuous integration/continuous deployment (CI/CD)

Deploying a model using Amazon SageMaker hosting services is a three-step process

Create a model in Amazon SageMaker

You need:

- The Amazon S3 path where the model artifacts are stored
- The Docker registry path for the image that contains the inference code
- A name that you can use for subsequent deployment steps

Create an endpoint configuration for an HTTPS endpoint

You need:

- The name of one or more models in production variants
- The ML compute instances that you want Amazon SageMaker to launch to host each production variant. When hosting models in production, you can configure the endpoint to elastically scale the deployed ML compute instances. For each production variant, you specify the number of ML compute instances that you want to deploy. When you specify two or more instances, Amazon SageMaker launches them in multiple Availability Zones. This ensures continuous availability. Amazon SageMaker manages deploying the instances.

Create an HTTPS endpoint

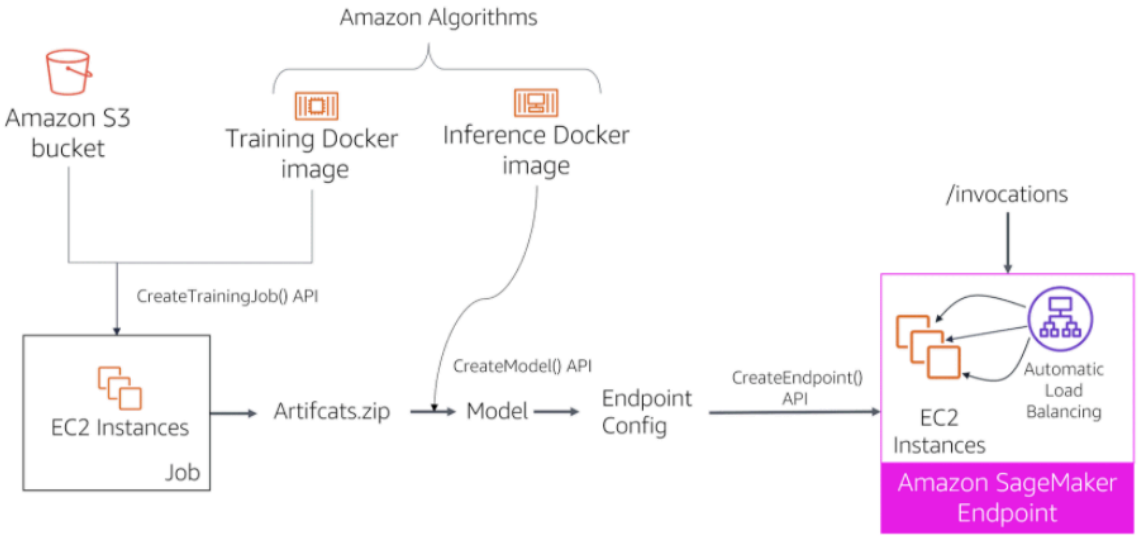
You need to provide the endpoint configuration to Amazon SageMaker. The service launches the ML compute instances and deploys the model or models as specified in the configuration.

You may want to pay extra attention to the following points when you're delivering an ML model into a production environment:

- Apply software engineering disciplines. Add error recovery code and make sure that tests for unexpected data inputs exist. Perform the same kind of unit testing, quality assurance, and user acceptance testing that is performed for other systems. If the ML system has moved from the research stage to development, some of these expected software engineering practices might have been inconsistently applied. Automate this system using common DevOps tools like AWS CodeBuild and AWS CodeCommit.
- Track, identify, and account for changes in data sources. The data might change over time. One change in data type in one source can break the whole pipeline. Changes in software that produces a data source can have flow-on effects.
- Perform ongoing monitoring and evaluation of results. Evaluate the expectations versus the results of the ML system. Build methods to check the error rate and the classes of errors being made against project expectations. If the overall error rate is the same, are the same proportions of the different classes of errors still the same? Is model drift occurring?
- Create methods to collect data from production inferences that can be used to improve future models.

Amazon SageMaker supports automatic scaling for production variants

Amazon SageMaker supports automatic scaling for production variants. Automatic scaling dynamically adjusts the number of instances provisioned for a production variant in response to changes in your workload. When the workload increases, automatic scaling brings more instances online. When the workload decreases, automatic scaling removes unnecessary instances so that you don't pay for provisioned variant instances that you aren't using.



Define and apply a scaling policy that uses Amazon CloudWatch metrics

- Automatic scaling uses the policy to adjust the number of instances up or down in response to actual workloads
- You can use the AWS Management Console to apply a scaling policy based on a predefined metric
- A predefined metric is defined in an enumeration so you can specify it by name in code or use it in the console
- Always load-test your automatic scaling configuration to ensure that it works correctly before using it to manage production traffic

Topics related to this subdomain

Here are some topics you may want to study for more in-depth information related to this subdomain:

- [ ] A/B testing with Amazon SageMaker
- [ ] Amazon SageMaker endpoints
  - Production variants
  - Endpoint configuration
- [ ] Using Lambda with Amazon SageMaker

Walk-through of sample questions

In this section, you'll have a chance to answer and walk through the solution to two different sample questions. The questions reflect some of the design and technical content you may see on the exam. The videos below will give you the answers to each question by walking you through the test-taking strategies presented to you earlier in this course.

Question 1

Answer the question below before watching the corresponding solution video.

A sports and betting company uses machine learning to predict the odds of winning during sporting events. It uses the Amazon SageMaker endpoint to serve its production model. The endpoint is on an m5.8xlarge instance.

What can the company do to ensure that this endpoint is highly available, while using the most cost-effective and easily managed solution?

- ✗ Create another endpoint. Put the two endpoints behind an Application Load Balancer.
- ✓ Increase the number of instances associated with the endpoint to more than one.
- ✗ Increase the instance size to m5.16 x-large.
- ✗ Add an elastic inference to the endpoint.

A healthcare company wants to deploy an ensemble of models behind a single endpoint with minimal management. The models include an XGBoost model trained on one of its structured datasets and a CNN model trained on an image dataset.

Which solution can the company use to reach this objective?

✓ Create an AWS Lambda function with Amazon API Gateway that preprocesses the incoming data. The function then creates the prediction from all the ensemble models. And finally, it returns the prediction.

✗ Create an AWS Deep Learning container that preprocesses the incoming data. The container then creates the prediction from all the ensemble models. And finally, it returns the prediction.

✗ Create an Amazon EC2 instance with the AWS Deep Learning AMI that preprocesses the incoming data. The instance then creates the prediction from all the ensemble models. And finally, it returns the prediction.

✗ Create an Amazon SageMaker endpoint that preprocesses the incoming data. The endpoint then creates the prediction from all the ensemble models. And finally, it returns the prediction.

A healthcare company wants to deploy an **ensemble of models** behind **a single endpoint** with **minimal management**. The models include an **XGBoost model** trained on one of its structured datasets and a **CNN model** trained on an image dataset.

A single sagemaker endpoint cannot serve 2 different models
=> leaves us with option A - lambda

Domain quiz

Take this short quiz to test your understanding of some of the topics related to this domain and experience the types of questions that will be on the exam. We recommend you use the test-taking strategies presented earlier in this course when completing the questions. Click the link below to begin.

aws training and certification

A Machine Learning Engineer created a pipeline for training an ML model using an Amazon SageMaker training job. The training job began successfully, but then failed after running for five minutes.

How should the Engineer begin to debug this issue? (Select TWO.)

- [x] Check the error in the given training job directly in the Amazon SageMaker console
- [ ] Call the DescribeJob API to check the FailureReason option
- [ ] Log into the Amazon SageMaker training job instance and check the job history
- [x] Go to Amazon CloudWatch logs and check the logs for the given training job
- [ ] Check AWS CloudTrail logs to check the error that caused the training to fail

Partially correct

✕ **Check the error in the given training job directly in the Amazon SageMaker console**

Call the DescribeJob API to check the FailureReason option

Log into the Amazon SageMaker training job instance and check the job history

✓ **Go to Amazon CloudWatch logs and check the logs for the given training job**

Check AWS CloudTrail logs to check the error that caused the training to fail

A news organization wants to extract metadata from its articles and blogs and index that metadata in Amazon Elasticsearch Service (Amazon ES) to enable faster searches.

What AWS service can the organization use to achieve this goal?

- ○ Amazon Rekognition Image
- ○ Amazon Personalize
- ◉ Amazon Comprehend
- ○ Amazon Textract

A Machine Learning Specialist is evaluating an ML model using a custom Deep Learning Amazon Machine Image (AMI) with Anaconda installed to run workloads through the terminal. Unfortunately, the ML Specialist does not have any experience with the Deep Learning AMI and wants to log into the instance and create an ipython notebook (*.ipynb), but cannot access the notebook interface.

After creating the AMI instance, what steps should the ML Specialist take to create a notebook?

- SSH into the Deep Learning AMI instance, start a new Flask interface application, and create a new ipython notebook
- SSH into the Deep Learning AMI instance with port forwarding at port 8080 and start a Zeppelin application to create a new ipython notebook
- (selected) SSH into the Deep Learning AMI instance with port forwarding at port 8888, start a Jupyter notebook application, and create a new ipython notebook
- SSH into the Deep Learning AMI instance with port forwarding at port 8888 and start a python3.6 application to create a new ipython notebook

A machine translation company is deploying its language translation models behind an Amazon SageMaker endpoint. The company wants to deploy a solution directly on its website so that users can input text in one language and have it translated into a second language. The company wants to reach a solution with minimal maintenance and latency for spiky traffic times.

How should the company architect this solution?

- (selected) Use Lambda to call InvokeEndpoint. Use the Amazon API Gateway URL to call the AWS Lambda function.
- Use Amazon SageMaker InvokeEndpoint with API Gateway
- Install the sagemaker-runtime library on the web server. Call InvokeEndpoint from the webserver.
- Create a function on an Amazon EC2 instance that uses CURL to call the InvokeEndpoint API. Call the Amazon EC2 instance from the website.

4/5 **80.0%**