Content outline

This exam guide includes weightings, content domains, and task statements for the exam. This guide does not provide a comprehensive list of the content on the exam. However, additional context for each task statement is available to help you prepare for the exam.

The exam has the following content domains and weightings:

• Domain 1: Data Preparation for Machine Learning (ML) (28% of scored content)

• Domain 2: ML Model Development (26% of scored content)

• Domain 3: Deployment and Orchestration of ML Workflows (22% of scored content)

• Domain 4: ML Solution Monitoring, Maintenance, and Security (24% of scored content)

Domain 1: Data Preparation for Machine Learning (ML)

Task Statement 1.1: Ingest and store data.

Knowledge of:

• Data formats and ingestion mechanisms (for example, validated and non-validated formats, Apache Parquet, JSON, CSV, Apache ORC, Apache Avro, RecordIO)

• How to use the core AWS data sources (for example, Amazon S3, Amazon Elastic File System [Amazon EFS], Amazon FSx for NetApp ONTAP)

• How to use AWS streaming data sources to ingest data (for example, Amazon Kinesis, Apache Flink, Apache Kafka)

• AWS storage options, including use cases and tradeoffs

Skills in:

• Extracting data from storage (for example, Amazon S3, Amazon Elastic Block Store [Amazon EBS], Amazon EFS, Amazon RDS, Amazon DynamoDB) by using relevant AWS service options (for example, Amazon S3 Transfer Acceleration, Amazon EBS Provisioned IOPS)

• Choosing appropriate data formats (for example, Parquet, JSON, CSV, ORC) based on data access patterns

• Ingesting data into Amazon SageMaker Data Wrangler and SageMaker Feature Store

• Merging data from multiple sources (for example, by using programming techniques, AWS Glue, Apache Spark)

• Troubleshooting and debugging data ingestion and storage issues that involve capacity and scalability

• Making initial storage decisions based on cost, performance, and data structure

Task Statement 1.2: Transform data and perform feature engineering.

Knowledge of:

• Data cleaning and transformation techniques (for example, detecting and treating outliers, imputing missing data, combining, deduplication)

• Feature engineering techniques (for example, data scaling and standardization, feature splitting, binning, log transformation, normalization)

• Encoding techniques (for example, one-hot encoding, binary encoding, label encoding, tokenization)

• Tools to explore, visualize, or transform data and features (for example, SageMaker Data Wrangler, AWS Glue, AWS Glue DataBrew)

• Services that transform streaming data (for example, AWS Lambda, Spark)

• Data annotation and labeling services that create high-quality labeled datasets

Skills in:

• Transforming data by using AWS tools (for example, AWS Glue, AWS Glue DataBrew, Spark running on Amazon EMR, SageMaker Data Wrangler)

• Creating and managing features by using AWS tools (for example, SageMaker Feature Store)

• Validating and labeling data by using AWS services (for example, SageMaker Ground Truth, Amazon Mechanical Turk)

Task Statement 1.3: Ensure data integrity and prepare data for modeling.

Knowledge of:

• Pre-training bias metrics for numeric, text, and image data (for example, class imbalance [CI], difference in proportions of labels [DPL])

• Strategies to address CI in numeric, text, and image datasets (for example, synthetic data generation, resampling)

• Techniques to encrypt data

• Data classification, anonymization, and masking

• Implications of compliance requirements (for example, personally identifiable information [PII], protected health information [PHI], data residency)

Skills in:

• Validating data quality (for example, by using AWS Glue DataBrew and AWS Glue Data Quality)

• Identifying and mitigating sources of bias in data (for example, selection bias, measurement bias) by using AWS tools (for example, SageMaker Clarify)

• Preparing data to reduce prediction bias (for example, by using dataset splitting, shuffling, and augmentation)

• Configuring data to load into the model training resource (for example, Amazon EFS, Amazon FSx)

Domain 2: ML Model Development

Task Statement 2.1: Choose a modeling approach.

Knowledge of:

• Capabilities and appropriate uses of ML algorithms to solve business problems

• How to use AWS artificial intelligence (AI) services (for example, Amazon Translate, Amazon Transcribe, Amazon Rekognition, Amazon Bedrock) to solve specific business problems

• How to consider interpretability during model selection or algorithm selection

• SageMaker built-in algorithms and when to apply them

Skills in:

• Assessing available data and problem complexity to determine the feasibility of an ML solution

• Comparing and selecting appropriate ML models or algorithms to solve specific problems

• Choosing built-in algorithms, foundation models, and solution templates (for example, in SageMaker JumpStart and Amazon Bedrock)

• Selecting models or algorithms based on costs

• Selecting AI services to solve common business needs

Task Statement 2.2: Train and refine models.

Knowledge of:

• Elements in the training process (for example, epoch, steps, batch size)

• Methods to reduce model training time (for example, early stopping, distributed training)

• Factors that influence model size

• Methods to improve model performance

• Benefits of regularization techniques (for example, dropout, weight decay, L1 and L2)

• Hyperparameter tuning techniques (for example, random search, Bayesian optimization)

• Model hyperparameters and their effects on model performance (for example, number of trees in a tree-based model, number of layers in a neural network)

• Methods to integrate models that were built outside SageMaker into SageMaker

Skills in:

• Using SageMaker built-in algorithms and common ML libraries to develop ML models

• Using SageMaker script mode with SageMaker supported frameworks to train models (for example, TensorFlow, PyTorch)

• Using custom datasets to fine-tune pre-trained models (for example, Amazon Bedrock, SageMaker JumpStart)

• Performing hyperparameter tuning (for example, by using SageMaker automatic model tuning [AMT])

• Integrating automated hyperparameter optimization capabilities

• Preventing model overfitting, underfitting, and catastrophic forgetting (for example, by using regularization techniques, feature selection)

• Combining multiple training models to improve performance (for example, ensembling, stacking, boosting)

• Reducing model size (for example, by altering data types, pruning, updating feature selection, compression)

• Managing model versions for repeatability and audits (for example, by using the SageMaker Model Registry)

Task Statement 2.3: Analyze model performance.

Knowledge of:

• Model evaluation techniques and metrics (for example, confusion matrix, heat maps, F1 score, accuracy, precision, recall, Root Mean Square Error [RMSE], receiver operating characteristic [ROC], Area Under the ROC Curve [AUC])

• Methods to create performance baselines

• Methods to identify model overfitting and underfitting

• Metrics available in SageMaker Clarify to gain insights into ML training data and models

• Convergence issues

Skills in:

• Selecting and interpreting evaluation metrics and detecting model bias

• Assessing tradeoffs between model performance, training time, and cost

• Performing reproducible experiments by using AWS services

• Comparing the performance of a shadow variant to the performance of a production variant

• Using SageMaker Clarify to interpret model outputs

• Using SageMaker Model Debugger to debug model convergence

Domain 3: Deployment and Orchestration of ML Workflows

Task Statement 3.1: Select deployment infrastructure based on existing architecture and requirements.

Knowledge of:

• Deployment best practices (for example, versioning, rollback strategies)

• AWS deployment services (for example, SageMaker)

• Methods to serve ML models in real time and in batches

• How to provision compute resources in production environments and test environments (for example, CPU, GPU)

• Model and endpoint requirements for deployment endpoints (for example, serverless endpoints, real-time endpoints, asynchronous endpoints, batch inference)

• How to choose appropriate containers (for example, provided or customized)

• Methods to optimize models on edge devices (for example, SageMaker Neo)

Skills in:

• Evaluating performance, cost, and latency tradeoffs

• Choosing the appropriate compute environment for training and inference based on requirements (for example, GPU or CPU specifications, processor family, networking bandwidth)

• Selecting the correct deployment orchestrator (for example, Apache Airflow, SageMaker Pipelines)

• Selecting multi-model or multi-container deployments

• Selecting the correct deployment target (for example, SageMaker endpoints, Kubernetes, Amazon Elastic Container Service [Amazon ECS], Amazon Elastic Kubernetes Service [Amazon EKS], Lambda)

• Choosing model deployment strategies (for example, real time, batch)

Task Statement 3.2: Create and script infrastructure based on existing architecture and requirements.

Knowledge of:

• Difference between on-demand and provisioned resources

• How to compare scaling policies

• Tradeoffs and use cases of infrastructure as code (IaC) options (for example, AWS CloudFormation, AWS Cloud Development Kit [AWS CDK])

• Containerization concepts and AWS container services

• How to use SageMaker endpoint auto scaling policies to meet scalability requirements (for example, based on demand, time)

Skills in:

• Applying best practices to enable maintainable, scalable, and cost-effective ML solutions (for example, automatic scaling on SageMaker endpoints, dynamically adding Spot Instances, by using Amazon EC2 instances, by using Lambda behind the endpoints)

• Automating the provisioning of compute resources, including communication between stacks (for example, by using CloudFormation, AWS CDK)

• Building and maintaining containers (for example, Amazon Elastic Container Registry [Amazon ECR], Amazon EKS, Amazon ECS, by using bring your own container [BYOC] with SageMaker)

• Configuring SageMaker endpoints within the VPC network

• Deploying and hosting models by using the SageMaker SDK

• Choosing specific metrics for auto scaling (for example, model latency, CPU utilization, invocations per instance)

Task Statement 3.3: Use automated orchestration tools to set up continuous integration and continuous delivery (CI/CD) pipelines.

Knowledge of:

• Capabilities and quotas for AWS CodePipeline, AWS CodeBuild, and AWS CodeDeploy

• Automation and integration of data ingestion with orchestration services

• Version control systems and basic usage (for example, Git)

• CI/CD principles and how they fit into ML workflows

• Deployment strategies and rollback actions (for example, blue/green, canary, linear)

• How code repositories and pipelines work together

Skills in:

• Configuring and troubleshooting CodeBuild, CodeDeploy, and CodePipeline, including stages

• Applying continuous deployment flow structures to invoke pipelines (for example, Gitflow, GitHub Flow)

• Using AWS services to automate orchestration (for example, to deploy ML models, automate model building)

• Configuring training and inference jobs (for example, by using Amazon EventBridge rules, SageMaker Pipelines, CodePipeline)

• Creating automated tests in CI/CD pipelines (for example, integration tests, unit tests, end-to-end tests)

• Building and integrating mechanisms to retrain models

Domain 4: ML Solution Monitoring, Maintenance, and Security

Task Statement 4.1: Monitor model inference.

Knowledge of:

• Drift in ML models

• Techniques to monitor data quality and model performance

• Design principles for ML lenses relevant to monitoring

Skills in:

• Monitoring models in production (for example, by using SageMaker Model Monitor)

• Monitoring workflows to detect anomalies or errors in data processing or model inference

• Detecting changes in the distribution of data that can affect model performance (for example, by using SageMaker Clarify)

• Monitoring model performance in production by using A/B testing

Task Statement 4.2: Monitor and optimize infrastructure and costs.

Knowledge of:

• Key performance metrics for ML infrastructure (for example, utilization, throughput, availability, scalability, fault tolerance)

• Monitoring and observability tools to troubleshoot latency and performance issues (for example, AWS X-Ray, Amazon CloudWatch Lambda Insights, Amazon CloudWatch Logs Insights)

• How to use AWS CloudTrail to log, monitor, and invoke re-training activities

• Differences between instance types and how they affect performance (for example, memory optimized, compute optimized, general purpose, inference optimized)

• Capabilities of cost analysis tools (for example, AWS Cost Explorer, AWS Billing and Cost Management, AWS Trusted Advisor)

• Cost tracking and allocation techniques (for example, resource tagging)

Skills in:

• Configuring and using tools to troubleshoot and analyze resources (for example, CloudWatch Logs, CloudWatch alarms)

• Creating CloudTrail trails

• Setting up dashboards to monitor performance metrics (for example, by using Amazon QuickSight, CloudWatch dashboards)

• Monitoring infrastructure (for example, by using EventBridge events)

• Rightsizing instance families and sizes (for example, by using SageMaker Inference Recommender and AWS Compute Optimizer)

• Monitoring and resolving latency and scaling issues

• Preparing infrastructure for cost monitoring (for example, by applying a tagging strategy)

• Troubleshooting capacity concerns that involve cost and performance (for example, provisioned concurrency, service quotas, auto scaling)

• Optimizing costs and setting cost quotas by using appropriate cost management tools (for example, AWS Cost Explorer, AWS Trusted Advisor, AWS Budgets)

• Optimizing infrastructure costs by selecting purchasing options (for example, Spot Instances, On-Demand Instances, Reserved Instances, SageMaker Savings Plans)

Task Statement 4.3: Secure AWS resources.

Knowledge of:

• IAM roles, policies, and groups that control access to AWS services (for example, AWS Identity and Access Management [IAM], bucket policies, SageMaker Role Manager)

• SageMaker security and compliance features

• Controls for network access to ML resources

• Security best practices for CI/CD pipelines

Skills in:

• Configuring least privilege access to ML artifacts

• Configuring IAM policies and roles for users and applications that interact with ML systems

• Monitoring, auditing, and logging ML systems to ensure continued security and compliance

• Troubleshooting and debugging security issues

• Building VPCs, subnets, and security groups to securely isolate ML systems