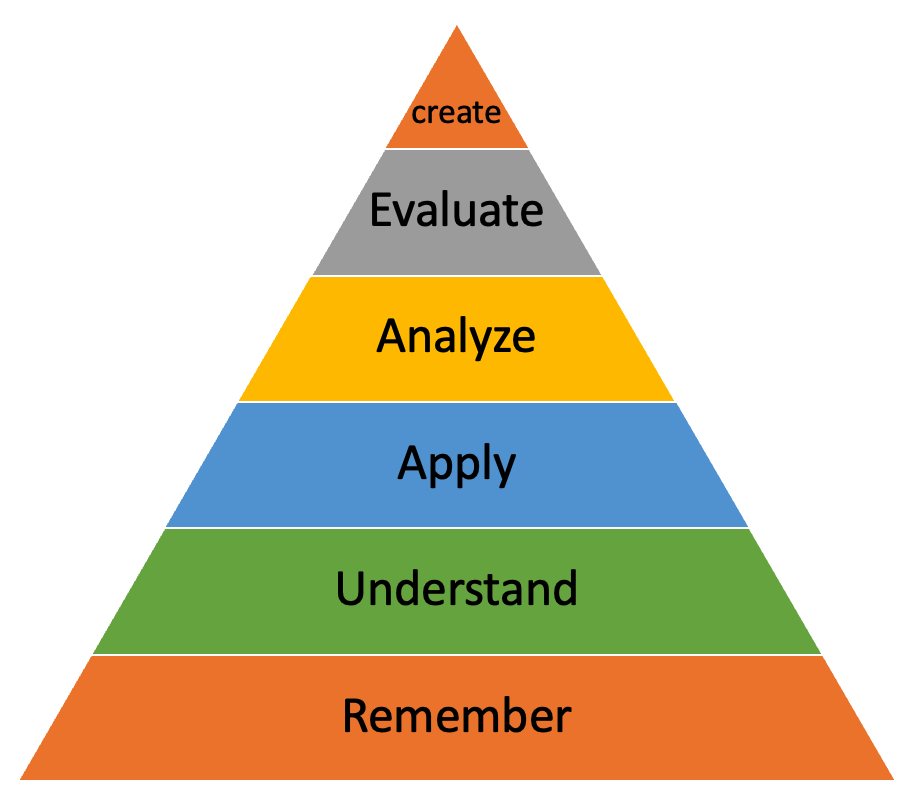
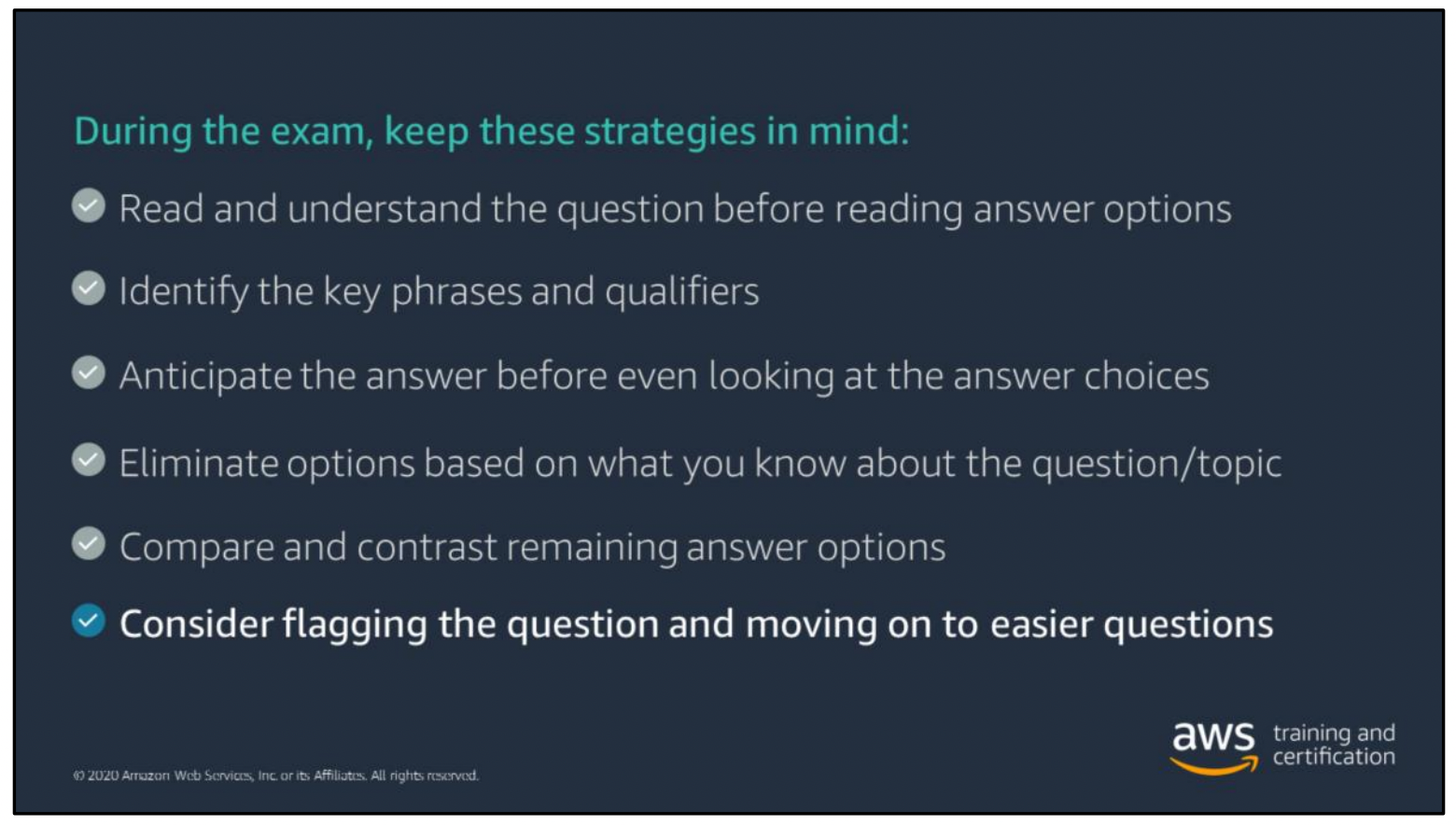
ML Specialty level exam challenge you to apply higher orders of knowledge



For instance, Specialty-level exam questions are not designed to solely test your recall of an AWS service. Instead, they challenge you to apply your knowledge of AWS services, features, and related concepts to a given scenario.



**Domain 1 (Data Engineering)**

They include:

1. Create data repositories for ML

2. Identify and implement a data-ingestion solution

3. Identify and implement a data-transformation solution

1. Create data repositories for ML

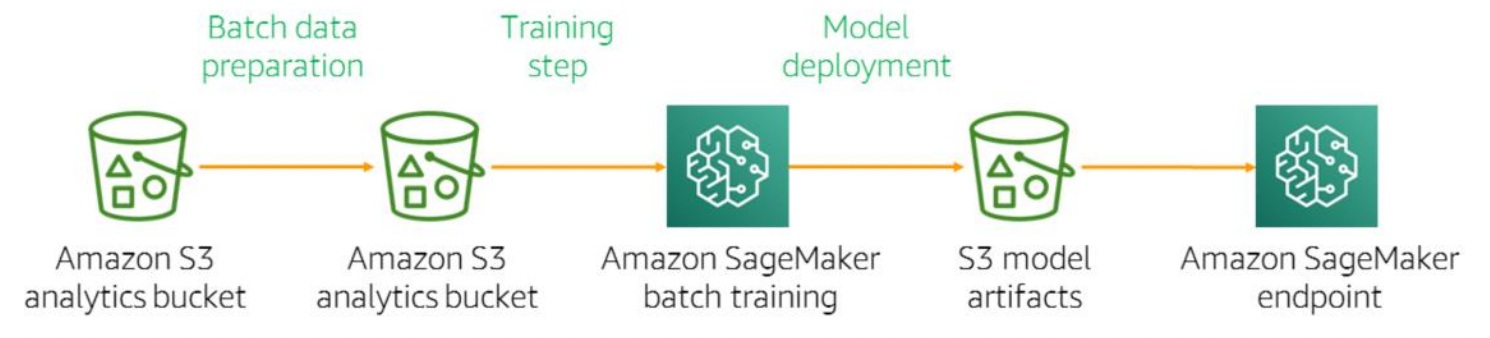
* The whole field of ML revolves around data. With clean data, you can gain important business insights.
* You need a way to store your data in a centralized repository (data lake)
* This data can be structured and unstructured
* Amazon S3 is the preferred storage option for data science processing on AWS (it can be used as “one source of truth” storage for most AWS ML services.)

1. Identify and implement a data-ingestion solution

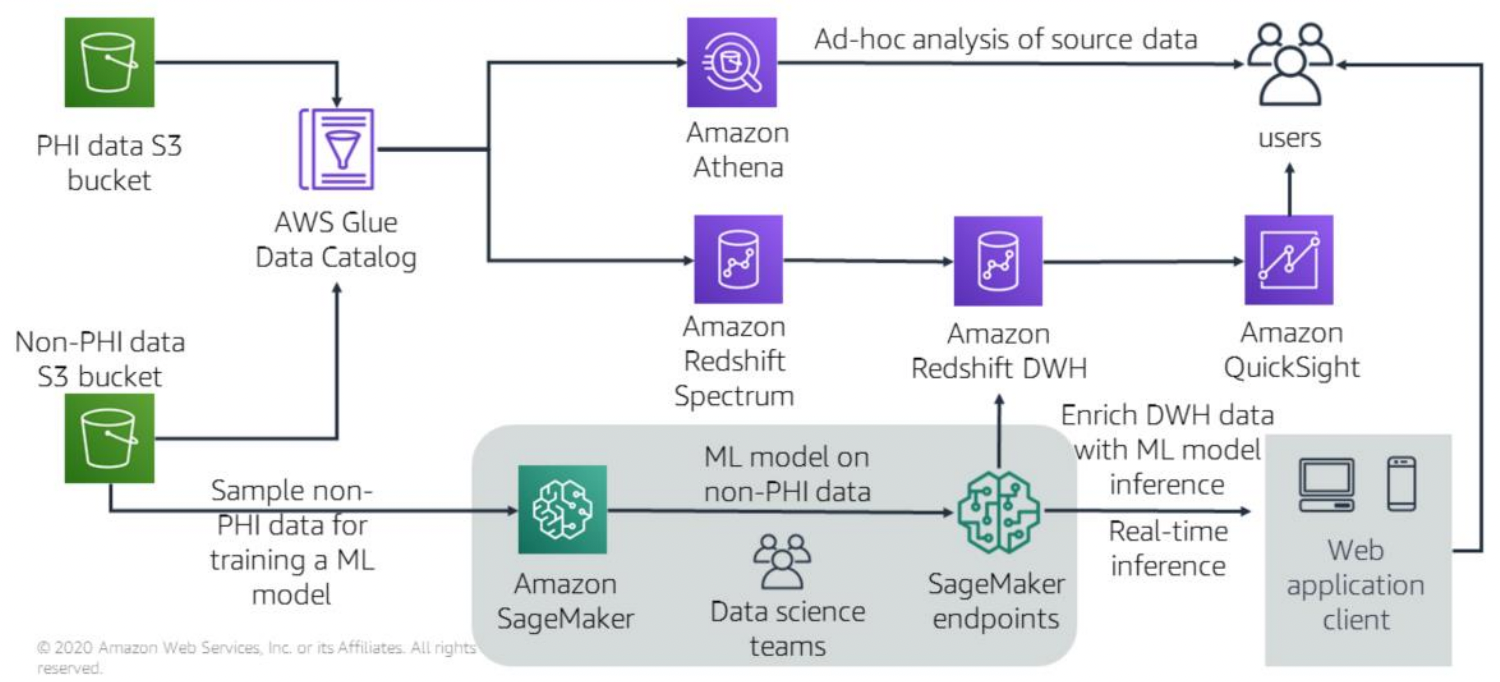
* To use this data for ML, you need to ingest it into a service like Amazon S3. There are different types of data ingestion methods. The one you choose depends on your business requirements and constraints.
* Batch and stream processing are two kinds of data ingestion.
* For batch ingestion to AWS, you can use services like AWS Glue (an ETL service), AWS DMS, … and automate it via AWS Step Functions.
* For Streaming ingestion to AWS, you can use Kinesis services

1. Identify and implement a data-transformation solution

* The raw data ingested into a service like Amazon S3 is usually not ML-ready. The data needs to be transformed and cleaned, which includes de-duplication, incomplete data management, and attribute standardization.



* Transforming your data for ML by using Apache Spark on Amazon EMR
* You can store a single source of data in Amazon S3 and perform ad-hoc analysis



**Domain 2 (Exploratory Data Analysis)**

1. Sanitize and prepare data for modeling

* Data is ingested and transformed, but it’s still missy and misunderstood. You still have to better understand and sanitize your data before you start training your model.
* The first thing you should do, before cleaning the data, is to use descriptive statistics to better understand your data. Descriptive statistics help you gain valuable insights into your data so that you can more effectively preprocess the data and prepare it for your ML model.
* Identifying correlations is important, because they can impact model performance. Correlations metrics measure the linear dependence between features. They can be visualized with heat maps.
* Also scatter plots visualize relationships between numerical variables.
* Now you understand your data, it’s time to clean it
* Make sure your data is on the same scale
* Outliers are points in your dataset that lie at an abnormal distance from other values. They are not always something you want to clean up, because they can add richness to your dataset.
* Two techniques to handle missing data;

1. Remove rows or columns
2. Impute missing values with mean, zero, ML, or DL
3. Perform feature engineering

* Feature engineering gives your model stronger prediction power. One of the first rules of feature engineering is to use your intuition
* Dimensionality reduction techniques include;

1. PCA
2. t-SNE

* For numerical features, you can do what is referred to as transformation (a multi-nominal or polynomial transformation)
* You will often handle categorical data that needs to be converted into numerical data before it can be read by your ML algorithm.
* One-hot encoding
* Common techniques for scaling such as, Mean/variance standardization, MinMax Scaling, Maxabs Scaling, Robust Scaling or Normalizer

1. Analyze and visualize data for ML

* Part of the exploratory data analysis phase of the ML pipeline is analyzing and visualizing your data, which helps you better understand your features and the relationships among features.

**Domain 3 (Modeling)**

1. Frame business problems as ML problems

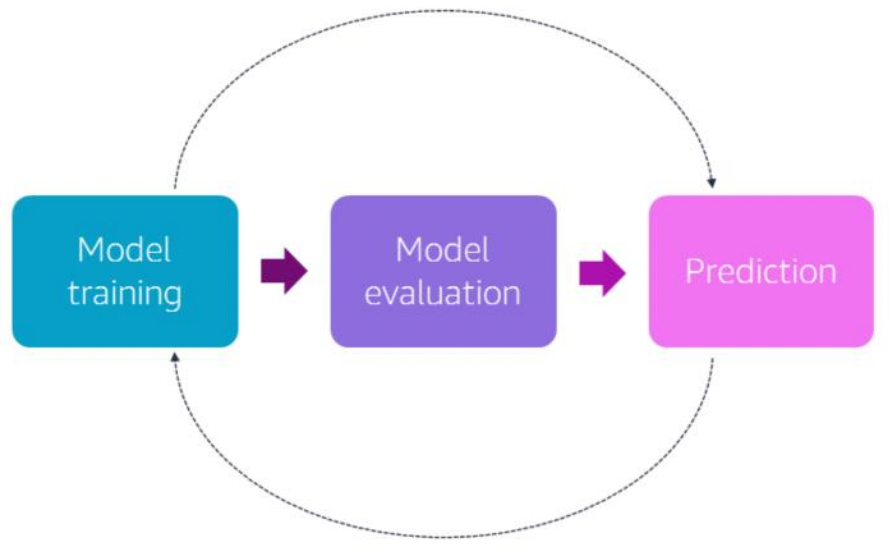
* First consider whether ML is an appropriate solution to this problem. By other words, is ML an appropriate solution
* There’re three categories of ML algorithms (supervised, unsupervised, and reinforcement)
* But within supervised learning, you have different types of problems. These can be broadly categorized into two categories, classification (binary or multiclass) and regression.

1. Select the appropriate models for an ML problem

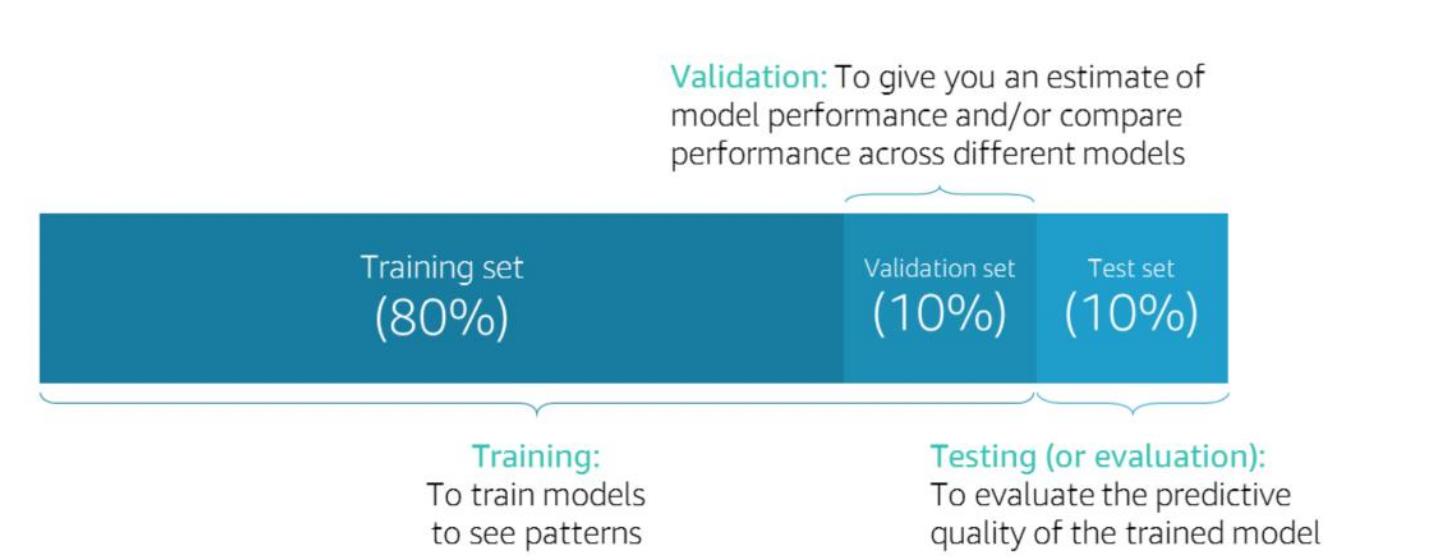
* Amazon SageMaker built-in algorithms

1. Train ML models

* Now that you have chosen an algorithm, you are ready to train, tune, and evaluate your ML model. This is an iterative process



* A big part of preparing for that training process is to first split your data to ensure a proper division between your training and evaluation efforts.
* Think about it this way. The fundamental goal of ML is to generalize beyond the data instances used to train models. You want to evaluate your model to estimate the quality of its predictions for data the model has not been trained on.



* There’re are many types of cross-validation;

1. K-fold cross-validation
2. Leave-one-out cross validation for small datasets
3. Stratified K-fold cross validation when you have imbalanced data

* Creating a training job in Amazon SageMaker

1. Perform Hyperparameter optimization

* You tune hyperparameters to get the best tuned model that generalizes or predicts both the training data and the future production data well.
* what are hyperparameters, exactly? Hyperparameters are the knobs or settings that can be tuned before running a training job to control the behavior of an ML algorithm. They can have a big impact on model training as it relates to training time, model convergence, and model accuracy.
* There’re different methods for tuning hyperparameters;

1. Manually
2. Grid Search or Random Search
3. Amazon SageMaker lets you perform automated hyperparameter tuning.
4. Evaluate ML model

* Once you have trained and tuned your models and decided which model is the best for your business problem, it’s time to evaluate that model to determine if it will do a good job predicting the target on new and future data.
* For classification problems, a confusion matrix is the building block for your model evaluation.

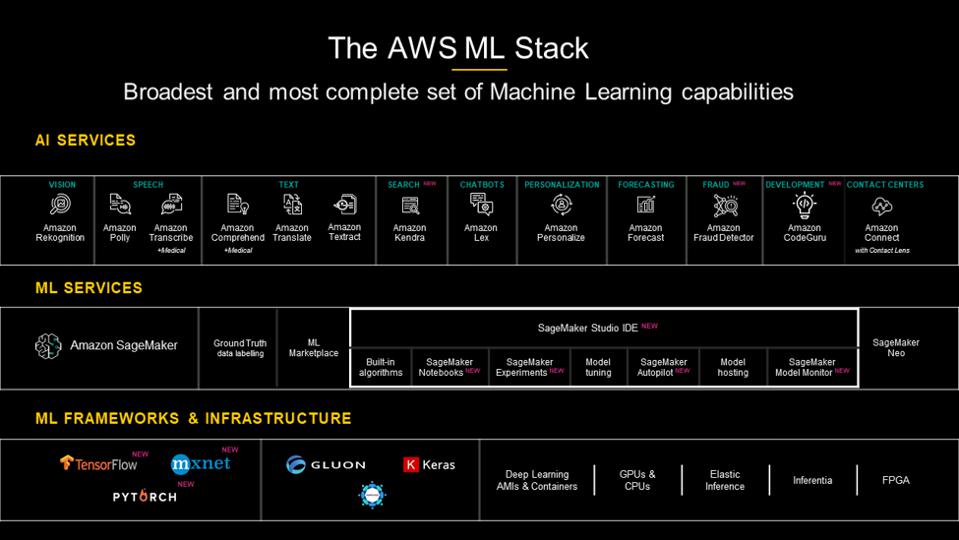
**Domain 4 (ML Implementation and Operations)**

1. Build ML solutions for performance, availability, scalability, resiliency, and fault tolerance

* At the heart of designing for failure are two concepts known as high availability and fault tolerance.
* One common method of achieving high availability and fault tolerance in your architectures is loose coupling
* Queues are also used when loosely coupling your system in order to pass messages between the various system 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.
* Additionally, 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.
* 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.
* 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.
* To ensure a highly available ML serving endpoint, deploy Amazon SageMaker endpoints backed by multiple instances across Availability Zones.
* Use Amazon Elastic Container Service (Amazon ECS) to containerize your application. Amazon ECS makes it easy to containerize ML models for both training and inference.
* 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.

1. Recommend and implement the appropriate ML services and features for a given problem

* The stack for Amazon ML has three tiers;



1. Apply basic AWS security practices to ML solutions

* When you create an Amazon SageMaker notebook instance, you can launch the instance with or without your Virtual Private Cloud (VPC) attached.
* 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 and in transit with TLSv1.2
* 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

1. Deploy and operationalize ML solutions

* This ecosystem must be managed using cloud and software engineering practices. For example:

• 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)

* You may want to focus some of your study on this subdomain around some of these, and other related, topics.

• A/B testing with Amazon SageMaker

• Production variants

• Using AWS Lambda with Amazon SageMaker