Amazon
MLA-C02
AWS Certified Machine Learning Engineer - Associate

For More Information – Visit link below:
https://www.examsempire.com/

Product Version

1. **Up to Date products, reliable and verified.**
2. **Questions and Answers in PDF Format.**



Latest Version: 6.0

Question: 1

Case Study
A company is building a web-based AI application by using Amazon SageMaker. The application will provide the following capabilities and features: ML experimentation, training, a central model registry, model deployment, and model monitoring.
The application must ensure secure and isolated use of training data during the ML lifecycle. The trainingdata is stored in Amazon S3.
The company needs to use the central model registry to manage different versions of models in the application.
Which action will meet this requirement with the LEAST operational overhead?

A. Create a separate Amazon Elastic Container Registry (Amazon ECR) repository for each model.
B. Use Amazon Elastic Container Registry (Amazon ECR) and unique tags for each model version.
C. Use the SageMaker Model Registry and model groups to catalog the models.
D. Use the SageMaker Model Registry and unique tags for each model version.

Answer: C

Explanation:
Amazon SageMaker Model Registry is a feature designed to manage machine learning (ML) models throughout their lifecycle. It allows users to catalog, version, and deploy models systematically, ensuring efficient model governance and management.
Key Features of SageMaker Model Registry:
Centralized Cataloging: Organizes models into Model Groups, each containing multiple versions.
Version Control: Maintains a history of model iterations, making it easier to track changes.
Metadata Association: Attach metadata such as training metrics and performance evaluations to models.
Approval Status Management: Allows setting statuses like PendingManualApproval or Approved to ensure only vetted models are deployed.
Seamless Deployment: Direct integration with SageMaker deployment capabilities for real-time inference or batch processing.
Implementation Steps:
Create a Model Group: Organize related models into groups to simplify management and versioning.
Register Model Versions: Each model iteration is registered as a version within a specific Model Group.
Set Approval Status: Assign approval statuses to models before deploying them to ensure quality control.
Deploy the Model: Use SageMaker endpoints for deployment once the model is approved.
Benefits:
Centralized Management: Provides a unified platform to manage models efficiently.
Streamlined Deployment: Facilitates smooth transitions from development to production.
Governance and Compliance: Supports metadata association and approval processes.

By leveraging the SageMaker Model Registry, the company can ensure organized management of models, version control, and efficient deployment workflows with minimal operational overhead.
AWS Documentation: SageMaker Model Registry
AWS Blog: Model Registry Features and Usage

Question: 2

Case Study
A company is building a web-based AI application by using Amazon SageMaker. The application will provide the following capabilities and features: ML experimentation, training, a central model registry, model deployment, and model monitoring.
The application must ensure secure and isolated use of training data during the ML lifecycle. The training data is stored in Amazon S3.
The company is experimenting with consecutive training jobs.
How can the company MINIMIZE infrastructure startup times for these jobs?

A. Use Managed Spot Training.
B. Use SageMaker managed warm pools.
C. Use SageMaker Training Compiler.
D. Use the SageMaker distributed data parallelism (SMDDP) library.

Answer: B

Explanation:
When running consecutive training jobs in Amazon SageMaker, infrastructure provisioning can introduce latency, as each job typically requires the allocation and setup of compute resources. To minimize this startup time and enhance efficiency, Amazon SageMaker offers Managed Warm Pools.
Key Features of Managed Warm Pools:
Reduced Latency: Reusing existing infrastructure significantly reduces startup time for training jobs.
Configurable Retention Period: Allows retention of resources after training jobs complete, defined by the
KeepAlivePeriodInSeconds parameter.
Automatic Matching: Subsequent jobs with matching configurations (e.g., instance type) can reuse retained infrastructure.
Implementation Steps:
Request Warm Pool Quota Increase: Increase the default resource quota for warm pools through AWS Service Quotas.
Configure Training Jobs:
Set KeepAlivePeriodInSeconds for the first training job to retain resources.
Ensure subsequent jobs match the retained pool's configuration to enable reuse.
Monitor Warm Pool Usage: Track warm pool status through the SageMaker console or API to confirm resource reuse.
Considerations:
Billing: Resources in warm pools are billable during the retention period.
Matching Requirements: Jobs must have consistent configurations to use warm pools effectively.
Alternative Options:
Managed Spot Training: Reduces costs by using spare capacity but doesn’t address startup latency.

SageMaker Training Compiler: Optimizes training time but not infrastructure setup.
SageMaker Distributed Data Parallelism Library: Enhances training efficiency but doesn't reduce setup time.
By using Managed Warm Pools, the company can significantly reduce startup latency for consecutive training jobs, ensuring faster experimentation cycles with minimal operational overhead.
AWS Documentation: Managed Warm Pools
AWS Blog: Reduce ML Model Training Job Startup Time

Question: 3

Case Study
A company is building a web-based AI application by using Amazon SageMaker. The application will provide the following capabilities and features: ML experimentation, training, a central model registry, model deployment, and model monitoring.
The application must ensure secure and isolated use of training data during the ML lifecycle. The training data is stored in Amazon S3.
The company must implement a manual approval-based workflow to ensure that only approved models can be deployed to production endpoints.
Which solution will meet this requirement?

A. Use SageMaker Experiments to facilitate the approval process during model registration.
B. Use SageMaker ML Lineage Tracking on the central model registry. Create tracking entities for the approval process.
C. Use SageMaker Model Monitor to evaluate the performance of the model and to manage the approval.
D. Use SageMaker Pipelines. When a model version is registered, use the AWS SDK to change the approval status to "Approved."

Answer: D

Explanation:
To implement a manual approval-based workflow ensuring that only approved models are deployed to production endpoints, Amazon SageMaker provides integrated tools such as SageMaker Pipelines and the SageMaker Model Registry.
SageMaker Pipelines is a robust service for building, automating, and managing end-to-end machine learning workflows. It facilitates the orchestration of various steps in the ML lifecycle, including data preprocessing, model training, evaluation, and deployment. By integrating with the SageMaker Model Registry, it enables seamless tracking and management of model versions and their approval statuses.
Implementation Steps:
Define the Pipeline:
Create a SageMaker Pipeline encompassing steps for data preprocessing, model training, evaluation, and registration of the model in the Model Registry.
Incorporate a Condition Step to assess model performance metrics. If the model meets predefined criteria, proceed to the next step; otherwise, halt the process.
Register the Model:
Utilize the RegisterModel step to add the trained model to the Model Registry.

Set the ModelApprovalStatus parameter to PendingManualApproval during registration. This status indicates that the model awaits manual review before deployment.
Manual Approval Process:
Notify the designated approver upon model registration. This can be achieved by integrating Amazon EventBridge to monitor registration events and trigger notifications via AWS Lambda functions.
The approver reviews the model's performance and, if satisfactory, updates the model's status to Approved using the AWS SDK or through the SageMaker Studio interface.
Deploy the Approved Model:
Configure the pipeline to automatically deploy models with an Approved status to the production endpoint. This can be managed by adding deployment steps conditioned on the model's approval status.
Advantages of This Approach:
Automated Workflow: SageMaker Pipelines streamline the ML workflow, reducing manual interventions and potential errors.
Governance and Compliance: The manual approval step ensures that only thoroughly evaluated models are deployed, aligning with organizational standards.
Scalability: The solution supports complex ML workflows, making it adaptable to various project requirements.
By implementing this solution, the company can establish a controlled and efficient process for deploying models, ensuring that only approved versions reach production environments.
Reference: Automate the machine learning model approval process with Amazon SageMaker Model Registry and Amazon SageMaker Pipelines
Update the Approval Status of a Model - Amazon SageMaker

Question: 4

Case Study
A company is building a web-based AI application by using Amazon SageMaker. The application will provide the following capabilities and features: ML experimentation, training, a central model registry, model deployment, and model monitoring.
The application must ensure secure and isolated use of training data during the ML lifecycle. The training data is stored in Amazon S3.
The company needs to run an on-demand workflow to monitor bias drift for models that are deployed to real-time endpoints from the application.
Which action will meet this requirement?

A. Configure the application to invoke an AWS Lambda function that runs a SageMaker Clarify job.
B. Invoke an AWS Lambda function to pull the sagemaker-model-monitor-analyzer built-in SageMaker image.
C. Use AWS Glue Data Quality to monitor bias.
D. Use SageMaker notebooks to compare the bias.

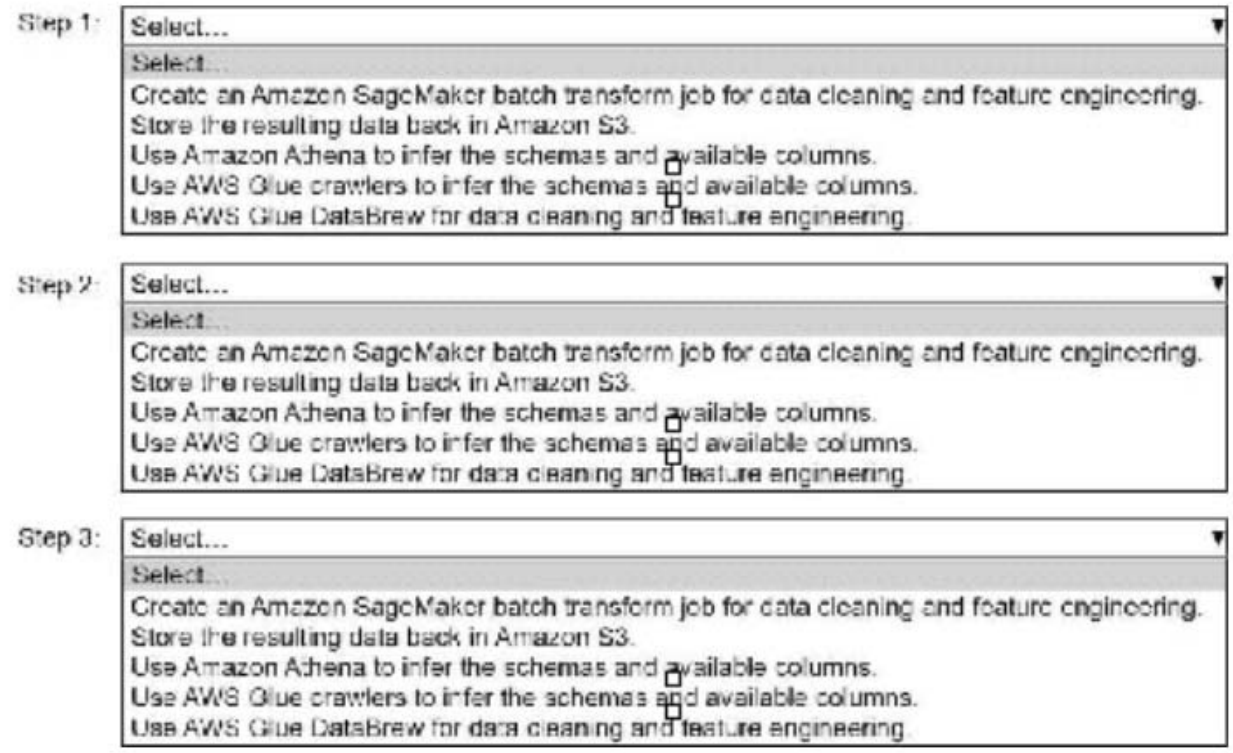
Answer: A

Explanation:
Monitoring bias drift in deployed machine learning models is crucial to ensure fairness and accuracy over time. Amazon SageMaker Clarify provides tools to detect bias in ML models, both during training

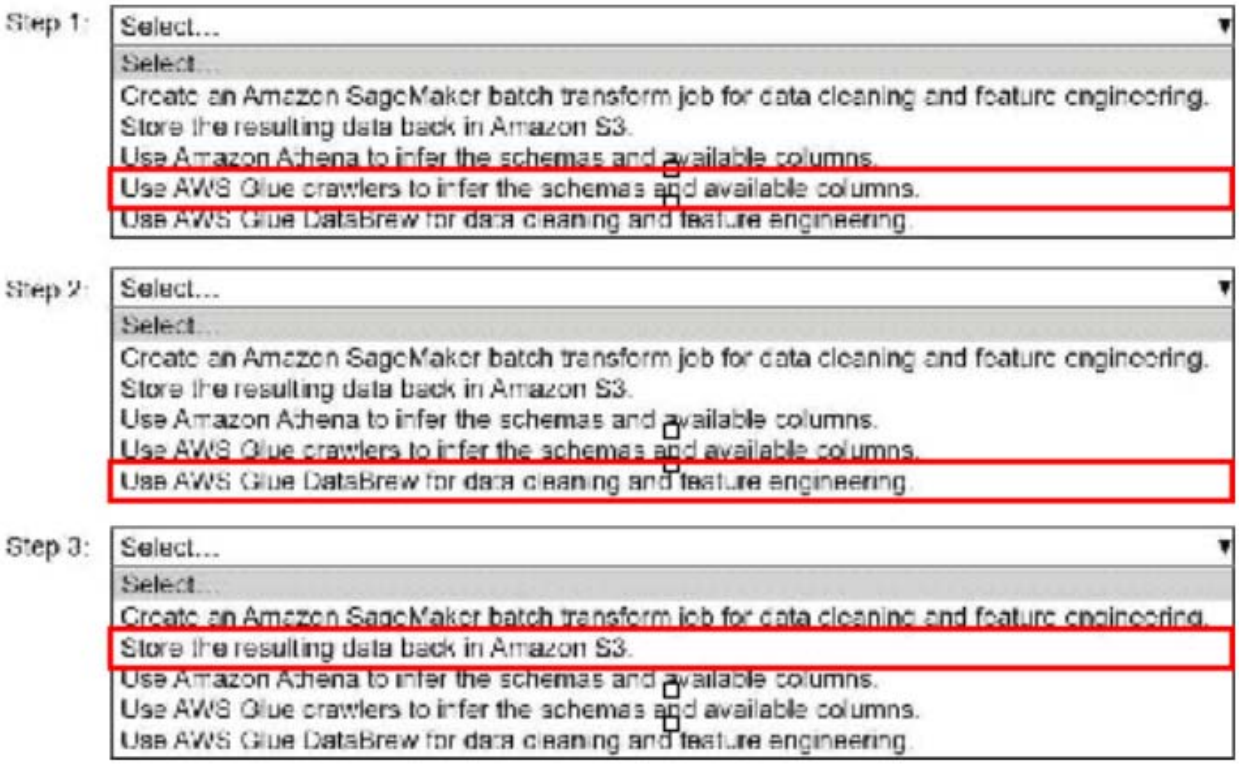
and after deployment. To monitor bias drift for models deployed to real-time endpoints, an effective approach involves orchestrating SageMaker Clarify jobs using AWS Lambda functions.
Implementation Steps:
Set Up Data Capture:
Enable data capture on the SageMaker endpoint to record input data and model predictions. This captured data serves as the basis for bias analysis.
Develop a Lambda Function:
Create an AWS Lambda function configured to initiate a SageMaker Clarify job. This function will process the captured data to assess bias metrics.
Schedule or Trigger the Lambda Function:
Configure the Lambda function to run on-demand or at scheduled intervals using Amazon CloudWatch Events or EventBridge. This setup allows for regular bias monitoring as per the application's requirements.
Analyze and Respond to Results:
After each Clarify job completes, review the generated bias reports. If bias drift is detected, take appropriate actions, such as retraining the model or adjusting data preprocessing steps.
Advantages of This Approach:
Automation: Utilizing AWS Lambda for orchestrating Clarify jobs enables automated and scalable bias monitoring without manual intervention.
Cost-Effectiveness: AWS Lambda's serverless nature ensures that you only pay for the compute time consumed during the execution of the function, optimizing resource usage.
Flexibility: The solution can be tailored to specific monitoring needs, allowing for adjustments in monitoring frequency and analysis parameters.
By implementing this solution, the company can effectively monitor bias drift in real-time, ensuring that the AI application maintains fairness and accuracy throughout its lifecycle.
Reference: Bias drift for models in production - Amazon SageMaker Schedule Bias Drift Monitoring Jobs - Amazon SageMaker

Question: 5

HOTSPOT:
A company stores historical data in .csv files in Amazon S3. Only some of the rows and columns in the .csv files are populated. The columns are not labeled. An ML engineer needs to prepare and store the data so that the company can use the data to train ML models.
Select and order the correct steps from the following list to perform this task. Each step should be selected one time or not at all. (Select and order three.)
• Create an Amazon SageMaker batch transform job for data cleaning and feature engineering.
• Store the resulting data back in Amazon S3.
• Use Amazon Athena to infer the schemas and available columns.
• Use AWS Glue crawlers to infer the schemas and available columns.
• Use AWS Glue DataBrew for data cleaning and feature engineering.



Answer:



Explanation:

Step 1: Use AWS Glue crawlers to infer the schemas and available columns.
Step 2: Use AWS Glue DataBrew for data cleaning and feature engineering.
Step 3: Store the resulting data back in Amazon S3.
Step 1: Use AWS Glue Crawlers to Infer Schemas and Available Columns
Why? The data is stored in .csv files with unlabeled columns, and Glue Crawlers can scan the raw data in Amazon S3 to automatically infer the schema, including available columns, data types, and any missing or incomplete entries.
How? Configure AWS Glue Crawlers to point to the S3 bucket containing the .csv files, and run the crawler to extract metadata. The crawler creates a schema in the AWS Glue Data Catalog, which can then be used for subsequent transformations.
Step 2: Use AWS Glue DataBrew for Data Cleaning and Feature Engineering
Why? Glue DataBrew is a visual data preparation tool that allows for comprehensive cleaning and transformation of data. It supports imputation of missing values, renaming columns, feature engineering, and more without requiring extensive coding.
How? Use Glue DataBrew to connect to the inferred schema from Step 1 and perform data cleaning and feature engineering tasks like filling in missing rows/columns, renaming unlabeled columns, and creating derived features.
Step 3: Store the Resulting Data Back in Amazon S3
Why? After cleaning and preparing the data, it needs to be saved back to Amazon S3 so that it can be used for training machine learning models.
How? Configure Glue DataBrew to export the cleaned data to a specific S3 bucket location. This ensures the processed data is readily accessible for ML workflows.
Order Summary:
Use AWS Glue crawlers to infer schemas and available columns.
Use AWS Glue DataBrew for data cleaning and feature engineering.
Store the resulting data back in Amazon S3.
This workflow ensures that the data is prepared efficiently for ML model training while leveraging AWS services for automation and scalability.

Thank You for Trying Our Product

Special 16 USD Discount Coupon: NSZUBG3X

Email: support@examsempire.com

Check our Customer Testimonials and ratings available on every product page.

Visit our website.

https://examsempire.com/