Free Questions for MLS-C01 by braindumpscollection

Shared by Hinton on 24-05-2024

Question 1

Question Type: MultipleChoice

A company is creating an application to identify, count, and classify animal images that are uploaded to the company's website. The company is using the Amazon SageMaker image classification algorithm with an ImageNetV2 convolutional neural network (CNN). The solution works well for most animal images but does not recognize many animal species that are less common.

The company obtains 10,000 labeled images of less common animal species and stores the images in Amazon S3. A machine learning (ML) engineer needs to incorporate the images into the model by using Pipe mode in SageMaker.

Which combination of steps should the ML engineer take to train the model? (Choose two.)

Options:

A- Use a ResNet model. Initiate full training mode by initializing the network with random weights.

B- Use an Inception model that is available with the SageMaker image classification algorithm.

C- Create a .lst file that contains a list of image files and corresponding class labels. Upload the .lst file to Amazon S3.

D- Initiate transfer learning. Train the model by using the images of less common species.

E- Use an augmented manifest file in JSON Lines format.

Answer:

C, D

Explanation:

The combination of steps that the ML engineer should take to train the model are to create a .lst file that contains a list of image files and corresponding class labels, upload the .lst file to Amazon S3, and initiate transfer learning by training the model using the images of less common species. This approach will allow the ML engineer to leverage the existing ImageNetV2 CNN model and fine-tune it with the new data using Pipe mode in SageMaker.

A .lst file is a text file that contains a list of image files and corresponding class labels, separated by tabs. The .lst file format is required for using the SageMaker image classification algorithm with Pipe mode. Pipe mode is a feature of SageMaker that enables streaming data directly from Amazon S3 to the training instances, without downloading the data first. Pipe mode can reduce the startup time, improve the I/O throughput, and enable training on large datasets that exceed the disk size limit.To use Pipe mode, the ML engineer needs to upload the .lst file to Amazon S3 and specify the S3 path as the input data channel for the training job1.

Transfer learning is a technique that enables reusing a pre-trained model for a new task by fine-tuning the model parameters with new data. Transfer learning can save time and computational resources, as well as improve the performance of the model, especially when the new task is similar to the original task. The SageMaker image classification algorithm supports transfer learning by allowing the ML engineer to specify the number of output classes and the number of layers to be retrained.The ML engineer can use the existing ImageNetV2 CNN model, which is trained on 1,000 classes of common objects, and fine-tune it with the new data of less common animal species, which is a similar task2.

The other options are either less effective or not supported by the SageMaker image classification algorithm. Using a ResNet model and initiating full training mode would require training the model from scratch, which would take more time and resources than transfer learning. Using an Inception model is not possible, as the SageMaker image classification algorithm only supports ResNet and ImageNetV2 models.Using an augmented manifest file in JSON Lines format is not compatible with Pipe mode, as Pipe mode only supports .lst files for image classification1.

References:

1: Using Pipe input mode for Amazon SageMaker algorithms | AWS Machine Learning Blog

2: Image Classification Algorithm - Amazon SageMaker

Question 2

Question Type: MultipleChoice

A manufacturing company wants to create a machine learning (ML) model to predict when equipment is likely to fail. A data science team already constructed a deep learning model by using TensorFlow and a custom Python script in a local environment. The company wants to use Amazon SageMaker to train the model.

Which TensorFlow estimator configuration will train the model MOST cost-effectively?

Options:

A- Turn on SageMaker Training Compiler by adding compiler_config=TrainingCompilerConfig() as a parameter. Pass the script to the estimator in the call to the TensorFlow fit() method.

B- Turn on SageMaker Training Compiler by adding compiler_config=TrainingCompilerConfig() as a parameter. Turn on managed spot training by setting the use_spot_instances parameter to True. Pass the script to the estimator in the call to the TensorFlow fit() method.

C- Adjust the training script to use distributed data parallelism. Specify appropriate values for the distribution parameter. Pass the script to the estimator in the call to the TensorFlow fit() method.

D- Turn on SageMaker Training Compiler by adding compiler_config=TrainingCompilerConfig() as a parameter. Set the MaxWaitTimeInSeconds parameter to be equal to the MaxRuntimeInSeconds parameter. Pass the script to the estimator in the call to the TensorFlow fit() method.

Answer:

B

Explanation:

The TensorFlow estimator configuration that will train the model most cost-effectively is to turn on SageMaker Training Compiler by adding compiler_config=TrainingCompilerConfig() as a parameter, turn on managed spot training by setting the use_spot_instances parameter to True, and pass the script to the estimator in the call to the TensorFlow fit() method. This configuration will optimize the model for the target hardware platform, reduce the training cost by using Amazon EC2 Spot Instances, and use the custom Python script without any modification.

SageMaker Training Compiler is a feature of Amazon SageMaker that enables you to optimize your TensorFlow, PyTorch, and MXNet models for inference on a variety of target hardware platforms. SageMaker Training Compiler can improve the inference performance and reduce the inference cost of your models by applying various compilation techniques, such as operator fusion, quantization, pruning, and graph optimization.You can enable SageMaker Training Compiler by adding compiler_config=TrainingCompilerConfig() as a parameter to the TensorFlow estimator constructor1.

Managed spot training is another feature of Amazon SageMaker that enables you to use Amazon EC2 Spot Instances for training your machine learning models. Amazon EC2 Spot Instances let you take advantage of unused EC2 capacity in the AWS Cloud. Spot Instances are available at up to a 90% discount compared to On-Demand prices. You can use Spot Instances for various fault-tolerant and flexible applications.You can enable managed spot training by setting the use_spot_instances parameter to True and specifying the max_wait and max_run parameters in the TensorFlow estimator constructor2.

The TensorFlow estimator is a class in the SageMaker Python SDK that allows you to train and deploy TensorFlow models on SageMaker. You can use the TensorFlow estimator to run your own Python script on SageMaker, without any modification. You can pass the script to the estimator in the call to the TensorFlow fit() method, along with the location of your input data.The fit() method starts a SageMaker training job and runs your script as the entry point in the training containers3.

The other options are either less cost-effective or more complex to implement. Adjusting the training script to use distributed data parallelism would require modifying the script and specifying appropriate values for the distribution parameter, which could increase the development time and complexity. Setting the MaxWaitTimeInSeconds parameter to be equal to the MaxRuntimeInSeconds parameter would not reduce the cost, as it would only specify the maximum duration of the training job, regardless of the instance type.

References:

1: Optimize TensorFlow, PyTorch, and MXNet models for deployment using Amazon SageMaker Training Compiler | AWS Machine Learning Blog

2: Managed Spot Training: Save Up to 90% On Your Amazon SageMaker Training Jobs | AWS Machine Learning Blog

3: sagemaker.tensorflow --- sagemaker 2.66.0 documentation

Question 3

Question Type: MultipleChoice

A medical device company is building a machine learning (ML) model to predict the likelihood of device recall based on customer data that the company collects from a plain text survey. One of the survey questions asks which medications the customer is taking. The data for this field contains the names of medications that customers enter manually. Customers misspell some of the medication names. The column that contains the medication name data gives a categorical feature with high cardinality but redundancy.

What is the MOST effective way to encode this categorical feature into a numeric feature?

Options:

A- Spell check the column. Use Amazon SageMaker one-hot encoding on the column to transform a categorical feature to a numerical feature.

C- Fix the spelling in the column by using char-RNN. Use Amazon SageMaker Data Wrangler one-hot encoding to transform a categorical feature to a numerical feature.

D- Use Amazon SageMaker Data Wrangler similarity encoding on the column to create embeddings Of vectors Of real numbers.

E- Use Amazon SageMaker Data Wrangler ordinal encoding on the column to encode categories into an integer between O and the total number Of categories in the column.

Answer:

C

Explanation:

The most effective way to encode this categorical feature into a numeric feature is to use Amazon SageMaker Data Wrangler similarity encoding on the column to create embeddings of vectors of real numbers. Similarity encoding is a technique that transforms categorical features into numerical features by computing the similarity between the categories. Similarity encoding can handle high cardinality and redundancy in categorical features, as it can group similar categories together based on their string similarity. For example, if the column contains the values "aspirin", "asprin", and "ibuprofen", similarity encoding will assign a high similarity score to "aspirin" and "asprin", and a low similarity score to "ibuprofen". Similarity encoding can also create embeddings of vectors of real numbers, which can be used as input for machine learning models. Amazon SageMaker Data Wrangler is a feature of Amazon SageMaker that enables you to prepare data for machine learning quickly and easily.You can use SageMaker Data Wrangler to apply similarity encoding to a column of categorical data, and generate embeddings of vectors of real numbers that capture the similarity between the categories1. The other options are either less effective or more complex to implement. Spell checking the column and using one-hot encoding would require additional steps and resources, and may not capture all the misspellings or redundancies. One-hot encoding would also create a large number of features, which could increase the dimensionality and sparsity of the dat

a. Ordinal encoding would assign an arbitrary order to the categories, which could introduce bias or noise in the data.References:

1: Amazon SageMaker Data Wrangler -- Amazon Web Services

Question 4

Question Type: MultipleChoice

A machine learning (ML) specialist uploads 5 TB of data to an Amazon SageMaker Studio environment. The ML specialist performs initial data cleansing. Before the ML specialist begins to train a model, the ML specialist needs to create and view an analysis report that details potential bias in the uploaded data.

Which combination of actions will meet these requirements with the LEAST operational overhead? (Choose two.)

Options:

A- Use SageMaker Clarify to automatically detect data bias

B- Turn on the bias detection option in SageMaker Ground Truth to automatically analyze data features.

C- Use SageMaker Model Monitor to generate a bias drift report.

D- Configure SageMaker Data Wrangler to generate a bias report.

E- Use SageMaker Experiments to perform a data check

Answer:

A, D

Explanation:

The combination of actions that will meet the requirements with the least operational overhead is to use SageMaker Clarify to automatically detect data bias and to configure SageMaker Data Wrangler to generate a bias report. SageMaker Clarify is a feature of Amazon SageMaker that provides machine learning (ML) developers with tools to gain greater insights into their ML training data and models. SageMaker Clarify can detect potential bias during data preparation, after model training, and in your deployed model.For instance, you can check for bias related to age in your dataset or in your trained model and receive a detailed report that quantifies different types of potential bias1. SageMaker Data Wrangler is another feature of Amazon SageMaker that enables you to prepare data for machine learning (ML) quickly and easily. You can use SageMaker Data Wrangler to identify potential bias during data preparation without having to write your own code. You specify input features, such as gender or age, and SageMaker Data Wrangler runs an analysis job to detect potential bias in those features.SageMaker Data Wrangler then provides a visual report with a description of the metrics and measurements of potential bias so that you can identify steps to remediate the bias2. The other actions either require more customization (such as using SageMaker Model Monitor or SageMaker Experiments) or do not meet the requirement of detecting data bias (such as using SageMaker Ground Truth).References:

1: Bias Detection and Model Explainability -- Amazon Web Services

2: Amazon SageMaker Data Wrangler -- Amazon Web Services

Question 5

Question Type: **MultipleChoice**

A network security vendor needs to ingest telemetry data from thousands of endpoints that run all over the world. The data is transmitted every 30 seconds in the form of records that contain 50 fields. Each record is up to 1 KB in size. The security vendor uses Amazon Kinesis Data Streams to ingest the dat

a. The vendor requires hourly summaries of the records that Kinesis Data Streams ingests. The vendor will use Amazon Athena to query the records and to generate the summaries. The Athena queries will target 7 to 12 of the available data fields.

Which solution will meet these requirements with the LEAST amount of customization to transform and store the ingested data?

Options:

A- Use AWS Lambda to read and aggregate the data hourly. Transform the data and store it in Amazon S3 by using Amazon Kinesis Data Firehose.

B- Use Amazon Kinesis Data Firehose to read and aggregate the data hourly. Transform the data and store it in Amazon S3 by using a short-lived Amazon EMR cluster.

C- Use Amazon Kinesis Data Analytics to read and aggregate the data hourly. Transform the data and store it in Amazon S3 by using Amazon Kinesis Data Firehose.

D- Use Amazon Kinesis Data Firehose to read and aggregate the data hourly. Transform the data and store it in Amazon S3 by using AWS Lambda.

Answer:

C

Explanation:

The solution that will meet the requirements with the least amount of customization to transform and store the ingested data is to use Amazon Kinesis Data Analytics to read and aggregate the data hourly, transform the data and store it in Amazon S3 by using Amazon Kinesis Data Firehose. This solution leverages the built-in features of Kinesis Data Analytics to perform SQL queries on streaming data and generate hourly summaries. Kinesis Data Analytics can also output the transformed data to Kinesis Data Firehose, which can then deliver the data to S3 in a specified format and partitioning scheme. This solution does not require any custom code or additional infrastructure to process the data. The other solutions either require more customization (such as using Lambda or EMR) or do not meet the requirement of aggregating the data hourly (such as using Lambda to read the data from Kinesis Data Streams).References:

1: Boosting Resiliency with an ML-based Telemetry Analytics Architecture | AWS Architecture Blog

2: AWS Cloud Data Ingestion Patterns and Practices

3: IoT ingestion and Machine Learning analytics pipeline with AWS IoT ...

4: AWS IoT Data Ingestion Simplified 101: The Complete Guide - Hevo Data

Question 6

Question Type: MultipleChoice

Each morning, a data scientist at a rental car company creates insights about the previous day's rental car reservation demands. The company needs to automate this process by streaming the data to Amazon S3 in near real time. The solution must detect high-demand rental cars at each of the company's locations. The solution also must create a visualization dashboard that automatically refreshes with the most recent data.

Which solution will meet these requirements with the LEAST development time?

Options:

A- Use Amazon Kinesis Data Firehose to stream the reservation data directly to Amazon S3. Detect high-demand outliers by using Amazon QuickSight ML Insights. Visualize the data in QuickSight.

B- Use Amazon Kinesis Data Streams to stream the reservation data directly to Amazon S3. Detect high-demand outliers by using the Random Cut Forest (RCF) trained model in Amazon SageMaker. Visualize the data in Amazon QuickSight.

C- Use Amazon Kinesis Data Firehose to stream the reservation data directly to Amazon S3. Detect high-demand outliers by using the Random Cut Forest (RCF) trained model in Amazon SageMaker. Visualize the data in Amazon QuickSight.

D- Use Amazon Kinesis Data Streams to stream the reservation data directly to Amazon S3. Detect high-demand outliers by using Amazon QuickSight ML Insights. Visualize the data in QuickSight.

Answer:

A

Explanation:

The solution that will meet the requirements with the least development time is to use Amazon Kinesis Data Firehose to stream the reservation data directly to Amazon S3, detect high-demand outliers by using Amazon QuickSight ML Insights, and visualize the data in QuickSight. This solution does not require any custom development or ML domain expertise, as it leverages the built-in features of QuickSight ML Insights to automatically run anomaly detection and generate insights on the streaming data. QuickSight ML Insights can also create a visualization dashboard that automatically refreshes with the most recent data, and allows the data scientist to explore the outliers and their key drivers.References:

1: Simplify and automate anomaly detection in streaming data with Amazon Lookout for Metrics | AWS Machine Learning Blog

2: Detecting outliers with ML-powered anomaly detection - Amazon QuickSight

3: Real-time Outlier Detection Over Streaming Data - IEEE Xplore

4: Towards a deep learning-based outlier detection ... - Journal of Big Data

Question 7

Question Type: **MultipleChoice**

A company wants to detect credit card fraud. The company has observed that an average of 2% of credit card transactions are fraudulent. A data scientist trains a classifier on a year's worth of credit card transaction dat

a. The classifier needs to identify the fraudulent transactions. The company wants to accurately capture as many fraudulent transactions as possible.

Which metrics should the data scientist use to optimize the classifier? (Select TWO.)

Options:

A- Specificity

B- False positive rate

C- Accuracy

D- Fl score

E- True positive rate

Answer:

D, E

Explanation:

The F1 score is a measure of the harmonic mean of precision and recall, which are both important for fraud detection. Precision is the ratio of true positives to all predicted positives, and recall is the ratio of true positives to all actual positives. A high F1 score indicates that the classifier can correctly identify fraudulent transactions and avoid false negatives. The true positive rate is another name for recall, and it measures the proportion of fraudulent transactions that are correctly detected by the classifier. A high true positive rate means that the classifier can capture as many fraudulent transactions as possible.

References:

Fraud Detection Using Machine Learning | Implementations | AWS Solutions

Detect fraudulent transactions using machine learning with Amazon SageMaker | AWS Machine Learning Blog

1. Introduction --- Reproducible Machine Learning for Credit Card Fraud Detection

Question 8

Question Type: **MultipleChoice**

A company wants to predict the classification of documents that are created from an application. New documents are saved to an Amazon S3 bucket every 3 seconds. The company has developed three versions of a machine learning (ML) model within Amazon SageMaker to classify document text. The company wants to deploy these three versions to predict the classification of each document.

Which approach will meet these requirements with the LEAST operational overhead?

Options:

A- Configure an S3 event notification that invokes an AWS Lambda function when new documents are created. Configure the Lambda function to create three SageMaker batch transform jobs, one batch transform job for each model for each document.

B- Deploy all the models to a single SageMaker endpoint. Treat each model as a production variant. Configure an S3 event notification that invokes an AWS Lambda function when new documents are created. Configure the Lambda function to call each production variant and return the results of each model.

C- Deploy each model to its own SageMaker endpoint Configure an S3 event notification that invokes an AWS Lambda function when new documents are created. Configure the Lambda function to call each endpoint and return the results of each model.

D- Deploy each model to its own SageMaker endpoint. Create three AWS Lambda functions. Configure each Lambda function to call a different endpoint and return the results. Configure three S3 event notifications to invoke the Lambda functions when new documents are created.

Answer:

B

Explanation:

The approach that will meet the requirements with the least operational overhead is to deploy all the models to a single SageMaker endpoint, treat each model as a production variant, configure an S3 event notification that invokes an AWS Lambda function when new documents are created, and configure the Lambda function to call each production variant and return the results of each model. This approach involves the following steps:

Deploy all the models to a single SageMaker endpoint. Amazon SageMaker is a service that can build, train, and deploy machine learning models. Amazon SageMaker can deploy multiple models to a single endpoint, which is a web service that can serve predictions from the models. Each model can be treated as a production variant, which is a version of the model that runs on one or more instances.Amazon SageMaker can distribute the traffic among the production variants according to the specified weights1.

Treat each model as a production variant. Amazon SageMaker can deploy multiple models to a single endpoint, which is a web service that can serve predictions from the models. Each model can be treated as a production variant, which is a version of the model that runs on one or more instances.Amazon SageMaker can distribute the traffic among the production variants according to the specified weights1.

Configure an S3 event notification that invokes an AWS Lambda function when new documents are created. Amazon S3 is a service that can store and retrieve any amount of data. Amazon S3 can send event notifications when certain actions occur on the objects in a bucket, such as object creation, deletion, or modification. Amazon S3 can invoke an AWS Lambda function as a destination for the event notifications.AWS Lambda is a service that can run code without provisioning or managing servers2.

Configure the Lambda function to call each production variant and return the results of each model. AWS Lambda can execute the code that can call the SageMaker endpoint and specify the production variant to invoke. AWS Lambda can use the AWS SDK or the SageMaker Runtime API to send requests to the endpoint and receive the predictions from the models.AWS Lambda can return the results of each model as a response to the event notification3.

The other options are not suitable because:

Option A: Configuring an S3 event notification that invokes an AWS Lambda function when new documents are created, configuring the Lambda function to create three SageMaker batch transform jobs, one batch transform job for each model for each document, will incur more operational overhead than using a single SageMaker endpoint. Amazon SageMaker batch transform is a service that can process large datasets in batches and store the predictions in Amazon S3. Amazon SageMaker batch transform is not suitable for real-time inference, as it introduces a delay between the request and the response.Moreover, creating three batch transform jobs for each document will increase the complexity and cost of the solution4.

Option C: Deploying each model to its own SageMaker endpoint, configuring an S3 event notification that invokes an AWS Lambda function when new documents are created, configuring the Lambda function to call each endpoint and return the results of each model, will incur more operational overhead than using a single SageMaker endpoint. Deploying each model to its own endpoint will increase the number of resources and endpoints to manage and monitor.Moreover, calling each endpoint separately will increase the latency and network traffic of the solution5.

Option D: Deploying each model to its own SageMaker endpoint, creating three AWS Lambda functions, configuring each Lambda function to call a different endpoint and return the results, configuring three S3 event notifications to invoke the Lambda functions when new documents are created, will incur more operational overhead than using a single SageMaker endpoint and a single Lambda function. Deploying each model to its own endpoint will increase the number of resources and endpoints to manage and monitor. Creating three Lambda functions will increase the complexity and cost of the solution.Configuring three S3 event notifications will increase the number of triggers and destinations to manage and monitor6.

References:

1: Deploying Multiple Models to a Single Endpoint - Amazon SageMaker

2: Configuring Amazon S3 Event Notifications - Amazon Simple Storage Service

3: Invoke an Endpoint - Amazon SageMaker

4: Get Inferences for an Entire Dataset with Batch Transform - Amazon SageMaker

5: Deploy a Model - Amazon SageMaker

6: AWS Lambda