

Efficient Test Data Prediction Using Amazon S3 Triggers, Lambda Functions, and Sagemaker Endpoint



Owned by [Ganesamanian Kolappan](#) ...
Yesterday at 11:25 AM

This document provides a comprehensive guide on using a serverless AWS Lambda function designed to seamlessly integrate with Amazon SageMaker, a machine learning service. Its main purpose is to make predictions on data stored in a CSV file located in an Amazon S3 bucket. This Lambda function downloads the input data, sends it to a pre-configured SageMaker endpoint, and receives the model's predictions. It then manually calculates the accuracy of these predictions using the labels from the same CSV file, providing a detailed evaluation of the model's performance. Finally, the Lambda function saves the predictions and accuracy results in a JSON file and uploads it to another specified Amazon S3 bucket. This approach ensures efficient and automated machine learning model evaluation while leveraging AWS serverless computing for enhanced scalability.

Steps Involved

To implement this solution, complete the following high-level steps:

1. Creating S3 buckets
2. Creating Endpoint
3. Creating read-and-write policy
4. Creating lambda-specific roles and attaching the policy
5. Creating lambda function and setting trigger

Prerequisite

No expertise level is required at least first-hand experience so that the following steps are ease to understand.

1. Python & pandas library
2. AWS
3. Sagemaker and ML knowledge

Code

The code files and the current PDF can be found in the Bitbucket repository [here](#)

Step 1: Creating S3 buckets

1. Open the [Amazon S3 console](#) and select the **Buckets** page

The screenshot shows the AWS S3 Buckets page. At the top, it displays summary statistics: Total storage 29.7 GB, Object count 166.2 k, and Average object size 187.5 KB. A note says you can enable advanced metrics in the "default-account-dashboard" configuration. Below this is a table of existing buckets:

Name	AWS Region	Access	Creation date
advosense-elb-virginia-logs	US East (N. Virginia) us-east-1	Objects can be public	September 7, 2023, 10:39:14 (UTC+02:00)
elasticbeanstalk-us-east-2-977473325382	US East (Ohio) us-east-2	Objects can be public	February 8, 2022, 19:39:41 (UTC+01:00)
prod-advosense-cert	Canada (Central) ca-central-1	Objects can be public	June 13, 2022, 15:26:17 (UTC+02:00)
advosense-prod-elb-logs	Canada (Central) ca-central-1	Bucket and objects not public	June 14, 2022, 14:45:15 (UTC+02:00)
aws-athena-query-results-977473325382-us-east-1	US East (N. Virginia) us-east-1	Bucket and objects not public	October 13, 2023, 14:15:58 (UTC+02:00)
aws-cloudtrail-logs-977473325382-48ccf3f2	US East (N. Virginia) us-east-1	Bucket and objects not public	October 13, 2023, 13:47:09 (UTC+02:00)
bucket-ganesh	Europe (Frankfurt) eu-central-1	Bucket and objects not public	August 30, 2023, 16:29:27 (UTC+02:00)
capecod.advosense	US East (N. Virginia) us-east-1	Bucket and objects not public	September 28, 2023, 04:57:08 (UTC+02:00)
bucket-indata		Not found	October 18, 2023, 16:43:48 (UTC+02:00)

A large orange "Create bucket" button is located at the top right of the table.

2. Choose **Create bucket** to load the input data.

3. Under **General configuration**, do the following:

- For the **Bucket name**, enter a globally unique name that meets the Amazon S3 **Bucket naming rules**. Bucket names can contain only lowercase letters, numbers, dots (.), and hyphens (-).
- For **AWS Region**, choose a Region. Later in the tutorial, you must create your Lambda function in the same Region.

The screenshot shows the "Create bucket" configuration page. The "General configuration" section is highlighted with a red box. It contains fields for "Bucket name" (containing "Bucket input") and "AWS Region" (set to "US East (N. Virginia) us-east-1"). Other sections like "Object Ownership", "Block Public Access settings for this bucket", and "Encryption" are also visible but not highlighted.

Bucket Versioning
Versioning is a feature of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning
 Disable
 Enable

Tags - optional (0)
You can use bucket tags to track storage costs and organize buckets. [Learn more](#)

No tags associated with this bucket.
[Add tag](#)

Default encryption [Info](#)
Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type [Info](#)
 Server-side encryption with Amazon S3 managed keys (SSE-S3)
 Server-side encryption with AWS Key Management Service keys (SSE-KMS)
 Dual-layer server-side encryption with AWS Key Management Service keys (DSS-E-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see [DSS-E-KMS pricing](#) on the Storage tab of the [Amazon S3 Pricing page](#).

Bucket Key
Using a bucket key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSS-E-KMS. [Learn more](#)
 Disable
 Enable

Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) [Create bucket](#)

4. Leave all other options set to their default values and choose **Create bucket**.
5. Create **another bucket** to store the processed data as per the above procedure
6. Should have **two buckets** to store one for input data and one for processed data

Name	AWS Region	Access	Creation date
advosme-ebs-virginia-logs	US East (N. Virginia) us-east-1	Objects can be public	September 7, 2023, 10:39:14 UTC+02:00
elastichemical-us-east-2-977473325382	US East (Ohio) us-east-2	Objects can be public	February 8, 2022, 19:39:41 UTC+01:00
grod-advosme	Canada (Central) ca-central-1	Objects can be public	June 12, 2023, 14:47:17 UTC+02:00
advosme-prd-us-east-1	Canada (Central) ca-central-1	Bucket and objects not public	June 12, 2023, 14:47:17 UTC+02:00
advosme-prd-us-east-1-977473325382-us-east-1	US East (US) us-east-1	Bucket and objects not public	October 11, 2023, 18:15:08 UTC+02:00
env-clusterv2-logs-977473325382-4kutPf2	US East (US) us-east-1	Bucket and objects not public	October 11, 2023, 18:15:08 UTC+02:00
Bucket-spam	Europe (Frankfurt) eu-central-1	Bucket and objects not public	August 30, 2023, 16:29:27 UTC+02:00
bucket-importdata	US East (N. Virginia) us-east-1	Bucket and objects not public	October 16, 2023, 16:48:15 UTC+02:00
bucket-promodata	US East (N. Virginia) us-east-1	Bucket and objects not public	October 16, 2023, 16:54:15 UTC+02:00
captured-advertisement	US East (N. Virginia) us-east-1	Bucket and objects not public	September 28, 2023, 04:37:08 UTC+02:00

Step 2: Creating Endpoint

1. To do this one should go through my [sagemaker file](#) and implement it till the end to get the endpoint

Name	ARN	Creation time	Status	Last updated
tuned-svm-endpoint	arn:aws:sagemaker:us-east-1:977473325382:endpoint:tuned-svm-endpoint	11/2/2023, 12:00:17 PM	InService	11/2/2023, 12:02:46 PM

2. Note down the ARN value of the endpoint, which should be added to the policy for access in the next step.

Step 3: Creating a read-and-write policy

1. Open the **Management Console** and select the **IAM** page

2. On the **IAM** page select **policies**

The screenshot shows the AWS Management Console with the IAM service selected. The left sidebar has a tree view with 'Policies' expanded, and its sub-item 'Create new policy' is highlighted with a red box. The main content area is the 'IAM Dashboard', which includes a summary table of resources and a 'What's new' section with recent updates.

User groups	Users	Roles	Policies	Identity providers
3	5	23	19	1

What's new (4 items):

- IAM Roles Anywhere is now available in the AWS GovCloud (US) Regions. 4 weeks ago
- AWS Identity and Access Management provides action last accessed information for more than 140 services. 1 month ago
- IAM roles last used and last accessed information available in AWS GovCloud (US) Regions. 1 month ago
- IAM Roles Anywhere credential helper adds support for OS certificate stores. 3 months ago

AWS Account details:

- Account ID: 977473325382
- Account Alias: sensa-development [Edit](#) | [Delete](#)
- Sign-in URL for IAM users in this account: <https://sensa-development.siginin.aws.amazon.com/console>

Tools

3. Choose to **create a policy**

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

- User groups
- Users
- Roles
- Policies**
- Identity providers
- Account settings

Access reports

- Access analyzer
- Archive rules
- Analyzers
- Settings
- Credential report
- Organization activity
- Service control policies (SCPs)

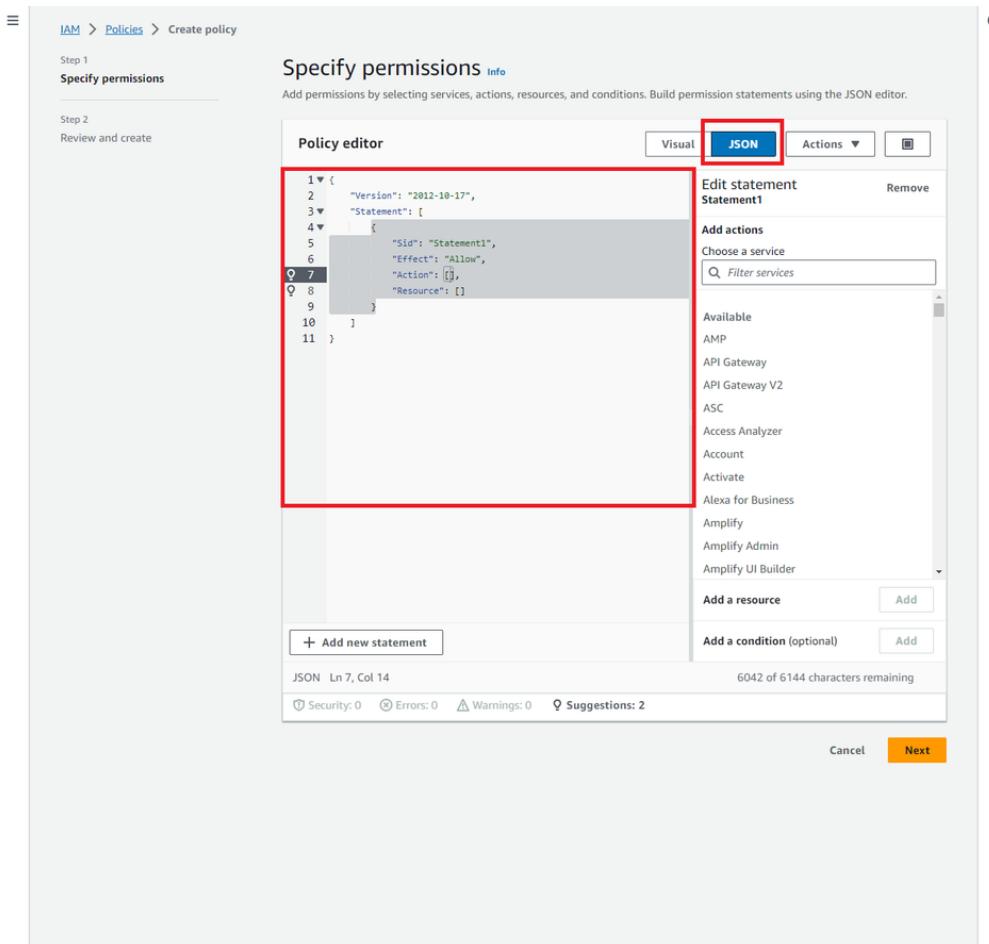
Related consoles

- IAM Identity Center
- AWS Organizations

Create policy

Policy name	Type	Used as	Description
AccessAnalyzerSer...	AWS managed	None	Allow Access Analyzer to analyze res...
AdministratorAccess	AWS managed - j...	Permissions polic...	Provides full access to AWS services.
AdministratorAcc...	AWS managed	None	Grants account administrative perm...
AdministratorAcc...	AWS managed	None	Grants account administrative perm...
AlexaForBusinessD...	AWS managed	None	Provide device setup access to Alexa...
AlexaForBusinessF...	AWS managed	None	Grants full access to AlexaForBusiness...
AlexaForBusinessG...	AWS managed	None	Provide gateway execution access to Alexa...
AlexaForBusinessL...	AWS managed	None	Provide access to Lifesize AVS device...
AlexaForBusinessN...	AWS managed	None	This policy enables Alexa for Busines...
AlexaForBusinessP...	AWS managed	None	Provide access to Poly AVS devices...
AlexaForBusinessR...	AWS managed	None	Provide read only access to AlexaFor...
AmazonAPIGatewa...	AWS managed	None	Provides full access to create/edit/de...
AmazonAPIGatewa...	AWS managed	None	Provides full access to invoke APIs in...
AmazonAPIGatewa...	AWS managed	None	Allows API Gateway to push logs to CloudWatch...
AmazonAppFlowF...	AWS managed	None	Provides full access to Amazon AppFlow...
AmazonAppFlowR...	AWS managed	None	Provides read only access to Amazon AppFlow...
AmazonAppStrea...	AWS managed	None	Provides full access to Amazon AppStream 2.0...
AmazonAppStrea...	AWS managed	None	Provides read only access to Amazon AppStream 2.0...
AmazonAppStrea...	AWS managed	None	Default policy for Amazon AppStream 2.0...

4. Select **JSON**



5. Paste the following code in the **policy editor field** giving access to S3 buckets and Sagemaker endpoint

```

1  {
2      "Version": "2012-10-17",
3      "Statement": [
4          {
5              "Sid": "putObject",
6              "Effect": "Allow",
7              "Action": [
8                  "s3:PutObject",
9                  "s3:GetObject"
10             ],
11             "Resource": [
12                 "arn:aws:s3:::bucket-processeddata/*"
13             ]
14         },
15         {
16             "Sid": "getObject",
17             "Effect": "Allow",
18             "Action": [
19                 "s3:GetObject"
20             ],
21             "Resource": [
22                 "arn:aws:s3:::bucket-inputdata/*"
23             ]
24         },
25         {
26             "Effect": "Allow",

```

```

27     "Action": [
28         "sagemaker:InvokeEndpoint"
29     ],
30     "Resource": "arn:aws:sagemaker:us-east-1:977473325382:endpoint/tuned-svm-endpoint"
31 }
32 ]
33 }
```

6. Select **Next**

7. In the next window, give the **policy name and description**, then select **Create policy**. In our case the name of the policy is **S3writebucket**

IAM > Policies > Create policy

Step 1
Specify permissions

Step 2
Review and create

Review and create Info

Review the permissions, specify details, and tags.

Policy details

Policy name
Enter a meaningful name to identify this policy.
S3writebucket
Maximum 128 characters. Use alphanumeric and '+-=,@-_.' characters.

Description - optional
Add a short explanation for this policy.
This policy is used to give access to store the processed data from lambda to the concern bucket
Maximum 1,000 characters. Use alphanumeric and '+-=,@-_.' characters.

Permissions defined in this policy Info

Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it

Edit

Search

Allow (1 of 384 services) Show remaining 383 services

Service	Access level	Resource	Request count
S3	Limited: Read, Write	Multiple	None

Add tags - optional Info
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag
You can add up to 50 more tags.

Create policy

8. Once created, verify by searching the name in the field as shown below

The screenshot shows the AWS Identity and Access Management (IAM) Policies page. The left sidebar includes sections for Dashboard, Access management (User groups, Users, Roles, Policies - which is selected and highlighted with a red box), Access reports (Access analyzer, Archive rules, Analyzers, Settings), Credential report, Organization activity, and Service control policies (SCPs). The main content area displays a table of policies. A search bar at the top has 's3' typed into it, and a red box highlights this search term. The table columns are Policy name, Type, Used as, and Description. The 'Used as' column shows 'None' for most policies, except for 'S3writebucket' which is 'Customer managed'. The 'Description' column for 'S3writebucket' states: 'This policy is used to give access to...'. Other policies listed include AmazonDMSRedsh..., AmazonS3FullAccess, AmazonS3Object... (with a red box around the '...' part), AmazonS3Outpost..., AmazonS3ReadOn..., AWSBackupService..., AWSS3OnOutpost..., IVSRecordToS3, QuickSightAccess..., S3StorageLensSer..., and SageMakerS3BucketP... (with a red box around the '...' part).

Step 4: Creating lambda-specific roles and attaching the policy

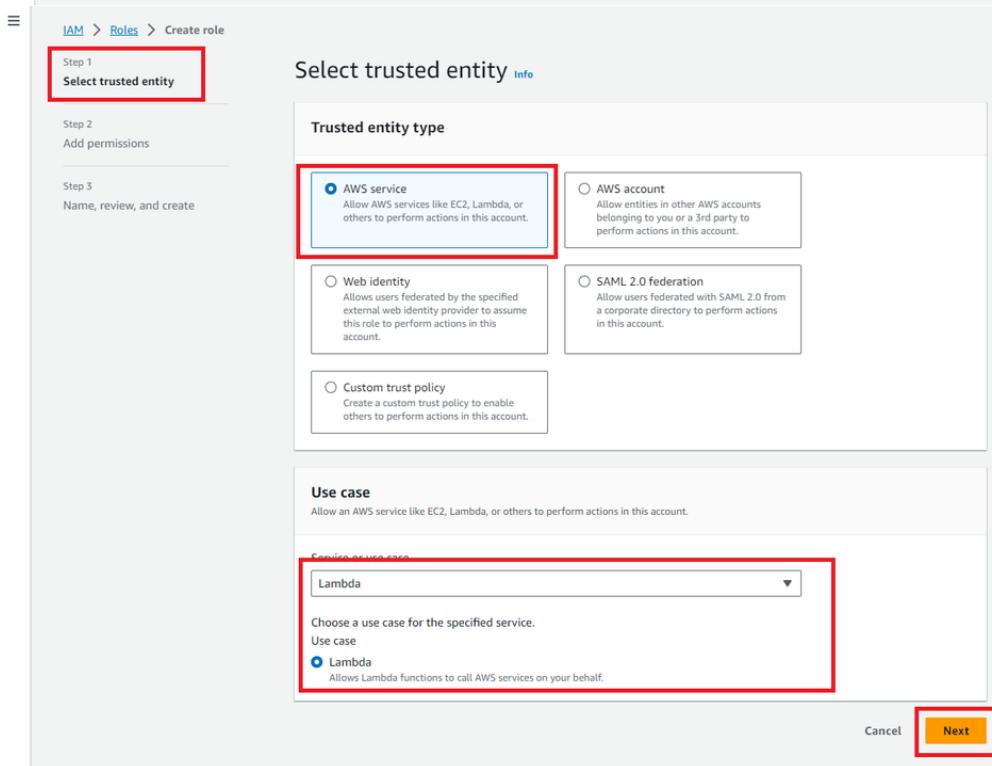
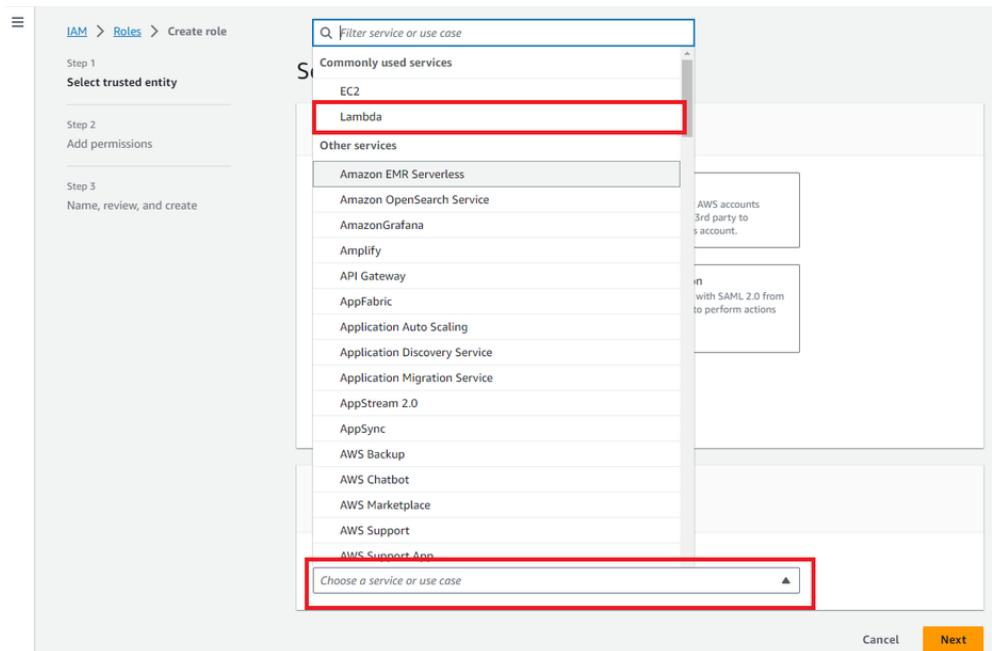
1. On the same **IAM** page, select **Roles**
2. Select **Create Role**

The screenshot shows the AWS IAM Roles page. On the left, there's a navigation sidebar with sections like 'Access management', 'Access reports', and 'Related consoles'. The 'Roles' section is highlighted with a red box. In the main content area, there's a table listing 23 roles. The first role listed is 'advocate_role'. The 'Create role' button is highlighted with a yellow box.

Role name	Trusted entities
advocate_role	AWS Service: iot
aws-elasticbeanstalk-ec2-role	AWS Service: ec2
aws-elasticbeanstalk-service-role	AWS Service: elasticbeanstalk
AWSCloud9SSMAccessRole	AWS Service: cloud9, and 1 more
AWSReservedSSO_AdministratorAccess_2d47c36323139d93	Identity Provider: arn:aws:iam::9774
AWSServiceRoleForAmazonSSM	AWS Service: ssm (Service-Linked Role)
AWSServiceRoleForAutoScaling	AWS Service: autoscaling (Service-Linked Role)
AWSServiceRoleForAWSCloud9	AWS Service: cloud9 (Service-Linked Role)
AWSServiceRoleForElasticLoadBalancing	AWS Service: elasticloadbalancing (Service-Linked Role)
AWSServiceRoleForGlobalAccelerator	AWS Service: globalaccelerator (Service-Linked Role)
AWSServiceRoleForOrganizations	AWS Service: organizations (Service-Linked Role)
AWSServiceRoleForRDS	AWS Service: rds (Service-Linked Role)
AWSServiceRoleForSSO	AWS Service: sso (Service-Linked Role)
AWSServiceRoleForSupport	AWS Service: support (Service-Linked Role)
AWSServiceRoleForTrustedAdvisor	AWS Service: trustedadvisor (Service-Linked Role)
copybucket-role-qbfj8r6	AWS Service: lambda
ec2-cloud-watch-logging	AWS Service: ec2
ec2-development	AWS Service: ec2
marco-role-ads0hlcg	AWS Service: lambda
polo-role-gag7e7d6	AWS Service: lambda

3. In the create role page, select **AWS service** and **choose service or use case as lambda**, then select **next**

The screenshot shows the 'Select trusted entity' step in the 'Create role' wizard. The 'AWS service' option is selected and highlighted with a red box. The 'service or use case' dropdown is also highlighted with a red box, showing 'Choose a service or use case'. The 'Next' button is highlighted with a yellow box.



4. In the next add permission page, select three policies namely **S3ReadonlyAcess**, **S3writebucket**, and **AWSLambdaExecute**, then click **Next**

IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Add permissions Info

Permissions policies (2/902) Info

Choose one or more policies to attach to your new role.

Policy name	Type	Description
<input type="checkbox"/>  AmazonDMSRedsh...	AWS managed	Provides access to manage S3 settings fo...
<input type="checkbox"/>  AmazonS3FullAccess	AWS managed	Provides full access to all buckets via the ...
<input type="checkbox"/>  AmazonS3ObjectL...	AWS managed	Provides AWS Lambda functions permissi...
<input type="checkbox"/>  AmazonS3Outpost...	AWS managed	Provides full access to Amazon S3 on Ou...
<input type="checkbox"/>  AmazonS3Outpost...	AWS managed	Provides read only access to Amazon S3 ...
<input checked="" type="checkbox"/>  AmazonS3ReadOn...	AWS managed	Provides read only access to all buckets v...
<input type="checkbox"/>  AWSBackupService...	AWS managed	Policy containing permissions necessary f...
<input type="checkbox"/>  AWSBackupService...	AWS managed	Policy containing permissions necessary f...
<input type="checkbox"/>  QuickSightAccessF...	AWS managed	Policy used by QuickSight team to access...
<input checked="" type="checkbox"/>  S3writebucket	Customer managed	This policy is used to give access to store ...
<input type="checkbox"/>  SageMakerS3BucketP...	Customer managed	-

▶ Set permissions boundary - optional

Cancel Previous Next

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Add permissions Info

Permissions policies (3/902) Info

Choose one or more policies to attach to your new role.

Policy name	Type	Description
<input type="checkbox"/>  AmazonS3ObjectL...	AWS managed	Provides AWS Lambda functions permis...
<input type="checkbox"/>  AmazonSageMaker...	AWS managed	Service role policy used by the AWS La...
<input type="checkbox"/>  AmazonSageMaker...	AWS managed	Service role policy used by the AWS La...
<input type="checkbox"/>  AWSCodeDeploy...	AWS managed	Provides CodeDeploy service access to ...
<input type="checkbox"/>  AWSCodeDeployR...	AWS managed	Provides CodeDeploy service limited a...
<input type="checkbox"/>  AWSDeepLensLam...	AWS managed	This policy specifies permissions requir...
<input type="checkbox"/>  AWSLambda_FullA...	AWS managed	Grants full access to AWS Lambda serv...
<input type="checkbox"/>  AWSLambda_ReadOnly...	AWS managed	Grants read-only access to AWS Lamb...
<input type="checkbox"/>  AWSLambdaBasicE...	AWS managed	Provides write permissions to CloudW...
<input type="checkbox"/>  AWSLambdaBasicExc...	Customer managed	-
<input type="checkbox"/>  AWSLambdaBasicExc...	Customer managed	-
<input type="checkbox"/>  AWSLambdaBasicExc...	Customer managed	-
<input checked="" type="checkbox"/>  AWSLambdaDyna...	AWS managed	Provides list and read access to Dynam...
<input type="checkbox"/>  AWSLambdaENIMan...	AWS managed	Provides minimum permissions for a L...
<input checked="" type="checkbox"/>  AWSLambdaExecute	AWS managed	Provides Put, Get access to S3 and full ...
<input type="checkbox"/>  AWSLambdaInvoke...	AWS managed	Provides read access to DynamoDB Str...
<input type="checkbox"/>  AWSLambdaKinesi...	AWS managed	Provides list and read access to Kinesis...
<input type="checkbox"/>  AWSLambdaMSKE...	AWS managed	Provides permissions required to acces...
<input type="checkbox"/>  AWSLambdaRole	AWS managed	Default policy for AWS Lambda service...
<input type="checkbox"/>  AWSLambdaSQS...	AWS managed	Provides receive message, delete mess...

▶ Set permissions boundary - optional

Cancel Previous Next

5. On the next page, give the role name, and description, then select **Create role**

IAM > Roles > Create role

Step 1
Select trusted entity

Step 2
Add permissions

Step 3
Name, review, and create

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.
preprocessingwithlambda

Maximum 64 characters. Use alphanumeric and '+-=._-' characters.

Description
Add a short explanation for this role.
Allows Lambda functions to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+-=._-' characters.

Step 1: Select trusted entities

Trust policy

```

1 [ { "Version": "2012-10-17", "Statement": [ "Effect": "Allow", "Action": [ "sts:AssumeRole" ], "Principal": { "Service": [ "lambda.amazonaws.com" ] } ] }

```

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AmazonS3ReadOnlyAccess	AWS managed	Permissions policy
AWSLambdaExecute	AWS managed	Permissions policy
S3writebucket	Customer managed	Permissions policy

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AmazonS3ReadOnlyAccess	AWS managed	Permissions policy
AWSLambdaExecute	AWS managed	Permissions policy
S3writebucket	Customer managed	Permissions policy

Step 3: Add tags

Add tags - optional Info
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel Previous **Create role**

6. Once created, verify by searching the name in the field as shown below

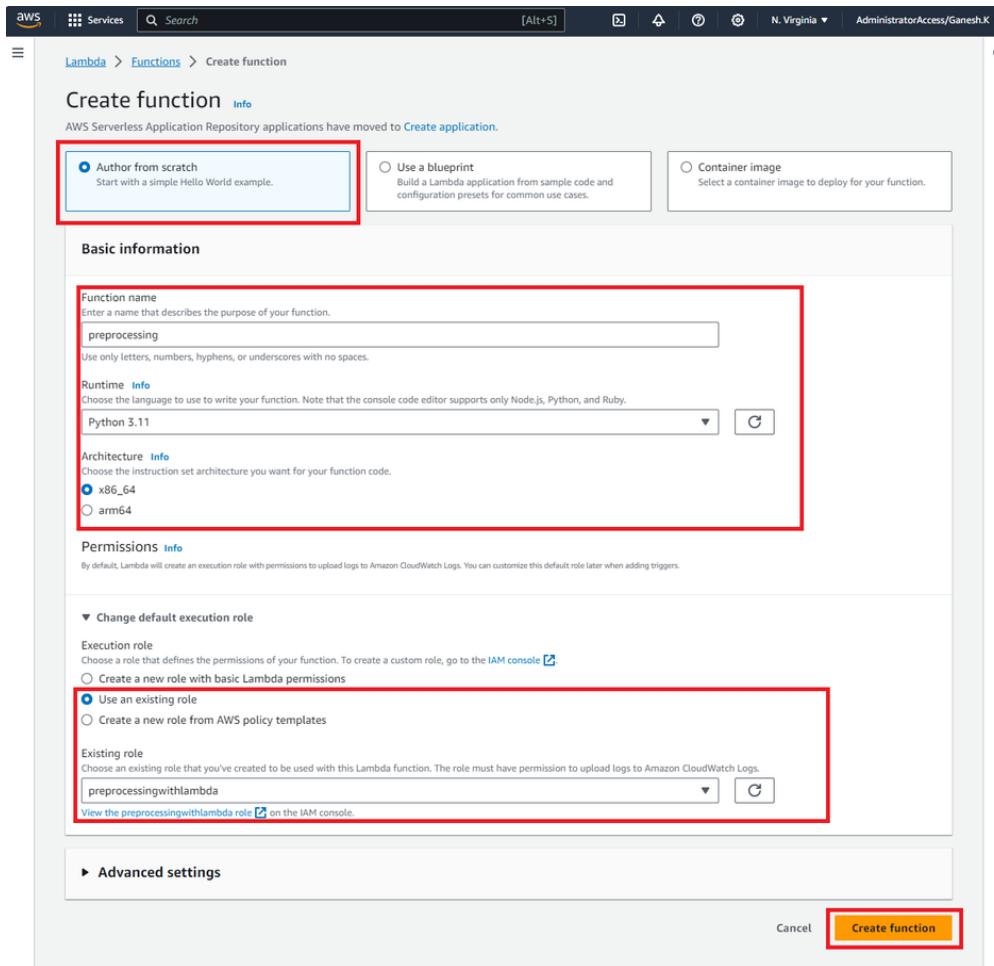
The screenshot shows the AWS IAM Roles page. A success message at the top says "Role preprocessingwithlambda created." Below it, the "Roles (24) Info" section is visible. A search bar shows "Search IAM" and a search result for "lamb" with one match: "preprocessingwithlambda". The "Trusted entities" section shows "AWS Service: lambda". Below this, there's a "Roles Anywhere" section with three options: "Access AWS from your non AWS workloads", "X.509 Standard", and "Temporary credentials".

Step 5: Creating lambda function and setting trigger

1. Open the **Management Console** and select the **lambda** page
2. Select **Create Function**

The screenshot shows the AWS Lambda Functions page. The left sidebar has "AWS Lambda" selected, with "Functions" highlighted. The main area shows a table titled "Functions (0)" with a single row: "There is no data to display." A yellow box highlights the "Create function" button at the top right of the table.

3. In the creation function, choose an **author from scratch**, give a **function name**, select **runtime as python**, **architecture x86_64**, choose a role as **use an existing role** as created, then select **create function**



4. Paste the below code in the **code source** on the next page

```
1 import json
2 import boto3
3 import csv
4 import tempfile
5 import os
6 import pandas as pd
7
8 def calculate_accuracy(predictions, labels):
9     correct_predictions = 0
10    total_predictions = len(predictions)
11
12    for pred, label in zip(predictions, labels):
13        if pred == label:
14            correct_predictions += 1
15
16    return correct_predictions / total_predictions
17
18 def lambda_handler(event, context):
19    # Initialize SageMaker runtime and S3 client
20    sm_runtime = boto3.client('sagemaker-runtime')
21    s3_client = boto3.client('s3')
22
23    # Specify the SageMaker endpoint name
24    endpoint_name = 'tuned-svm-endpoint'
```

```

25     # Define S3 bucket and file paths
26     input_bucket = 'bucket-inputdata'
27     input_file_key = 'test.csv'
28     output_bucket = 'bucket-processeddata'
29     output_file_key = 'predictions.json'
30
31
32     # Download the input CSV file from S3 to a temporary directory
33     temp_dir = tempfile.mkdtemp()
34     input_file_path = os.path.join(temp_dir, 'input.csv')
35     s3_client.download_file(input_bucket, input_file_key, input_file_path)
36
37     # Read the CSV file into a pandas DataFrame
38     df = pd.read_csv(input_file_path)
39
40     # Extract the label column for accuracy calculation
41     labels = df['label'] # Assuming 'label' is the name of your label column
42
43     # Remove the label column before making predictions
44     data = df.drop(columns=['label'])
45
46     # Make a prediction request to the SageMaker endpoint
47     response = sm_runtime.invoke_endpoint(
48         EndpointName=endpoint_name,
49         ContentType='text/csv',
50         Body=data.to_csv(index=False, header=False)
51     )
52
53     # Extract the prediction result
54     prediction_result = json.loads(response['Body'].read())
55
56     # Calculate accuracy manually
57     accuracy = calculate_accuracy(prediction_result, labels)
58
59     # Save the predictions and accuracy to a JSON file
60     output_data = {
61         "predictions": prediction_result,
62         "accuracy": accuracy
63     }
64     output_json = json.dumps(output_data)
65
66     # Upload the JSON file to the output S3 bucket
67     s3_client.put_object(
68         Bucket=output_bucket,
69         Key=output_file_key,
70         Body=output_json,
71         ContentType='application/json'
72     )
73
74     return {
75         'statusCode': 200,
76         'body': json.dumps("Predictions and accuracy saved to S3")
77     }
78
79

```

5. Once the code is pasted, click **Deploy**

The screenshot shows the AWS Lambda function code editor. At the top, there are tabs for Code, Test, Monitor, Configuration, Aliases, and Versions. The Code tab is selected. Below the tabs is a toolbar with File, Edit, Find, View, Go, Tools, Window, Test, Deploy, and Changes not deployed. The Deploy button is highlighted with a red box. The main area shows the code for a lambda_function.py file:

```
df = json2dataframe(json_data)
# Save the DataFrame as a CSV file
csv_bytes = df.to_csv(index=False).encode('utf-8')
s3.put_object(Bucket=destination_bucket_name, Key=destination_key, Body=csv_bytes)

# s3.copy_object(CopySource={'Bucket': source_bucket_name, 'Key': source_key},
#                 Bucket=destination_bucket_name, Key=destination_key)
copied_files.add(destination_key) # Add the copied file to the record

# Update the record of copied files in S3
s3.put_object(Bucket=destination_bucket_name, Key=copied_file_record_key, Body=json.dumps(list(copied_files)))

return {
    'statusCode': 200,
    'body': 'Recent uncopied files copied successfully!'
}
```

6. Now scroll down to the layers, click **add layer**

The screenshot shows the Lambda function configuration page. At the top, there is a code editor window with the same Python code as above. Below it are sections for Code properties, Runtime settings, and Layers.

Code properties

- Package size: 1.3 kB
- SHA256 hash: 8p10eHnu0axZ6w1MjDErQWQOXz9C4Rp0Dz7OE=
- Last modified: October 18, 2023 at 05:52 PM GMT+2

Runtime settings

- Runtime: Python 3.11
- Handler: lambda_function.lambda_handler
- Architecture: x86_64

Layers

There is no data to display.

The 'Add a layer' button in the Layers section is highlighted with a red box.

7. In the layer page, select **pandas** and version, then select **add**

Add layer

Function runtime settings

Runtime Python 3.11	Architecture x86_64
------------------------	------------------------

Choose a layer

Layer source [Info](#)
Choose from layers with a compatible runtime and instruction set architecture or specify the Amazon Resource Name (ARN) of a layer version. You can also [create a new layer](#).

AWS layers Choose a layer from a list of layers provided by AWS.

Custom layers Choose a layer from a list of layers created by your AWS account or organization.

Specify an ARN Specify a layer by providing the ARN.

AWS layers
Layers provided by AWS that are compatible with your function's runtime.

AWSSDKPandas-Python311

Version
2

Cancel Add

8. Now in the **function overview** click **Add trigger**

Lambda > Functions > preprocessing

preprocessing

Throttle Copy ARN Actions ▾

+ Add trigger

Layers (1)

+ Add destination

Description -

Last modified 2 minutes ago

Function ARN arn:aws:lambda:us-east-1:977473325582:function:preprocessing

Function URL [Info](#)

Code Test Monitor Configuration Aliases Versions

Code source [Info](#)

File Edit Find View Go Tools Window Test Deploy

Upload from ▾

Environment Var

lambda_function

```
1 import json
2 import boto3
3 import pandas as pd
4
5 def jsonDataFrame(data):
6     tags = []
7
8     for message in data:
9         if 'messages' in message:
10             for nested_message in message['messages']:
11                 if 'payload' in nested_message:
12                     payload = json.loads(nested_message['payload'])
13                     if 'tags' in payload:
14                         for fields in payload['tags']:
15                             tag = {
16                                 'reader_name': payload.get('rfidReaderName', ''),
17                                 'reader_serial': payload.get('rfidReaderSerialNumber', ''),
18                                 'reader_internal_serial': payload.get('rfidReaderInternalSerialNumber', ''),
19                                 'reader_timezone': payload.get('timeZone', ''),
20                                 'tag_pc': fields.get('pc', ''),
21                                 'tag_epc': fields.get('epc', ''),
22                                 'tag_rssi': fields.get('rssi', ''),
23                                 'tag_timestamp': fields.get('timeStampOfRead', ''),
24                                 'tag_phase': fields.get('phase', ''),
25                                 'tag_antennaport': fields.get('antennaPort', '')
26                             }
27                             tags.append(tag)
28
29             df = pd.DataFrame(tags)
30
31
```

9. Select the source **S3** on the next page then click **next**

The screenshot shows the 'Add trigger' dialog for AWS Lambda. At the top, there's a 'Trigger configuration' section with a 'Info' link. Below it is a search bar with the placeholder 'Select a source' and a magnifying glass icon. To the right of the search bar are 'Cancel' and 'Add' buttons. The main area is divided into three sections: 'APIs/interactive/web', 'Batch/bulk data processing', and 'Real-time/streaming data'. Under 'Batch/bulk data processing', the 'S3' trigger is highlighted with a red box. The 'S3' trigger has a green icon, the label 'S3', and the tag 'aws asynchronous storage'. Other triggers listed include Alexa, API Gateway, Application Load Balancer, CloudFront, CodeCommit, Cognito Sync Trigger, VPC Lattice, AWS IoT, CloudWatch Logs, EventBridge (CloudWatch Events), SNS, SQS, Apache Kafka, and DocumentDB.

10. On the next page, select the **input bucket**, check the **acknowledge** then select **Add**

Lambda > Add trigger

Add trigger

Trigger configuration [Info](#)

S3 aws asynchronous storage

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

[X](#) [C](#)

Bucket region: us-east-1

Event types
Select the events that you want to trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

[X](#)

Prefix - optional
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

Suffix - optional
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.

Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. [Learn more](#)

I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

Lambda will add the necessary permissions for AWS S3 to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

[Cancel](#) [Add](#)

11. Goto configuration tab on the lambda function page and choose edit in General configuration as shown below

Lambda > Functions > preprocessing

preprocessing

Function overview [Info](#)

preprocessing [Layers](#) (1)

S3 [+ Add destination](#)

[+ Add trigger](#)

Description
-

Last modified
8 minutes ago

Function ARN
[arn:aws:lambda:us-east-1:977473325382:function:preprocessing](#)

Function URL [Info](#)
-

[Throttle](#) [Copy ARN](#) [Actions ▾](#)

[Code](#) | [Test](#) | [Monitor](#) | **Configuration** [Edit](#) | [Aliases](#) | [Versions](#)

General configuration [Info](#)

Description -	Memory 128 MB	Ephemeral storage 512 MB
Timeout 1 min 3 sec	SnapStart Info None	

Triggers

Permissions

Destinations

Function URL

Environment variables

Tags

VPC

Monitoring and operations tools

Concurrency

Edit basic settings

Basic settings [Info](#)

Description - optional

Memory [Info](#)
 Your function is allocated CPU proportional to the memory configured.
 MB
 Set memory to between 128 MB and 10240 MB.

Ephemeral storage [Info](#)
 You can configure up to 10 GB of ephemeral storage (/tmp) for your function. [View pricing](#)
 MB
 Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

SnapStart [Info](#)
 Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the [SnapStart compatibility considerations](#).

 Supported runtimes: Java 11, Java 17.

Timeout
 min sec

Execution role
 Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).
 Use an existing role
 Create a new role from AWS policy templates

Existing role
 Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
 [View the preprocessingwithlambda role](#) on the IAM console.

[Cancel](#) [Save](#)

12. After changing the time click save

If we don't change in case the execution will take more time than specified the function will stop due to timeout

Checking

1. Now upload a **test.csv** file in the bucket-inputdata

aws Services Search [Alt+S]

Amazon S3 > Buckets > bucket-inputdata

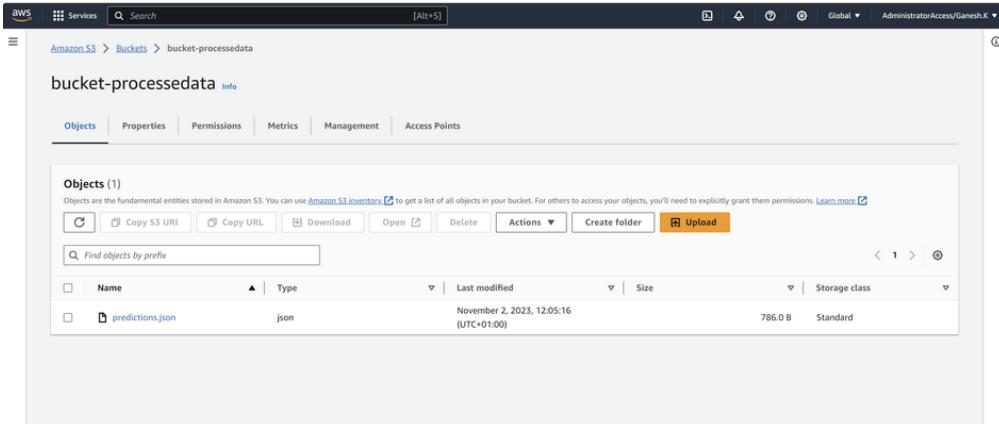
bucket-inputdata [Info](#)

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	test.csv	csv	November 2, 2023, 12:05:06 (UTC+01:00)	24.2 KB	Standard

2. You should be able to see the **predictions.json** that stores the predictions and accuracy that have been processed using the sagemaker endpoint.



3. When you open the file, you could able to see the output predictions and accuracy

4. In case it didn't give the required output see the error in Cloudwatch (the code given in the document works fine)