**March 2018**



**Intelligent Data Lake Workshop**

*Lab 3 – SageMaker*

Table of Contents

[Overview 3](#_Toc508307393)

[Prepare Training Data with AWS Glue 4](#_Toc508307394)

[Train, Create and Host a Machine Learning Model with Amazon SageMaker 8](#_Toc508307395)

[Invoke Amazon SageMaker Endpoint for Prediction/Inference 11](#_Toc508307396)

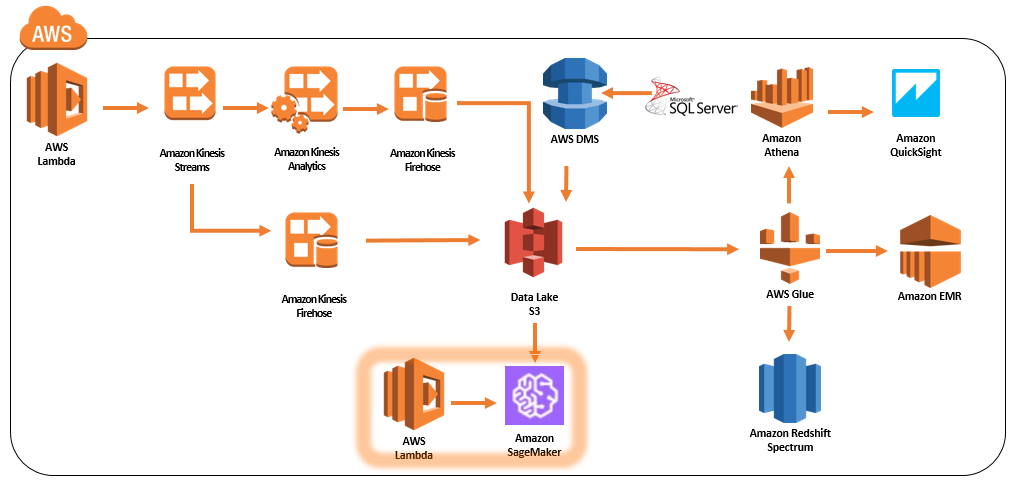
[Conclusion 15](#_Toc508307397)

# Overview

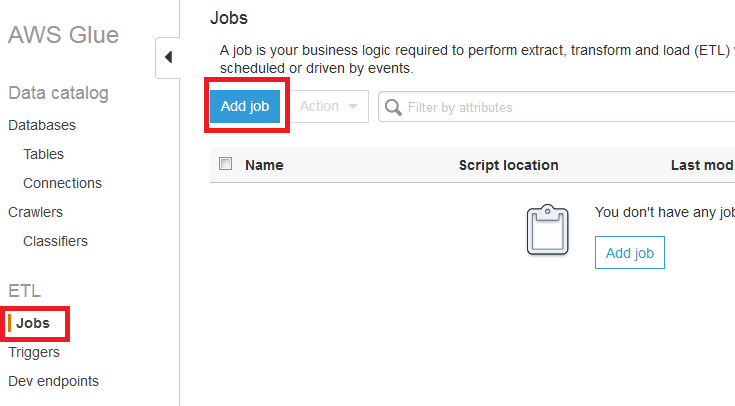
In previous two labs, you used Kinesis Streams to collect and store the streaming IoT sensor data, then used Kinesis Analytics to process and analyze the streaming data continuously. You also used Amazon Kinesis Firehose to export both the raw and processed data into S3 for further processing and analysis. You then used Glue Data Catalog to build a small data lake and shared catalog with disparate systems/services such as Amazon Athena, Amazon EMR and Amazon Redshift Spectrum. You also explored how to use Glue ETL tools to transform raw data to Parquet format. Finally, you used QuickSight to visualize the data stored on S3 with Athena.

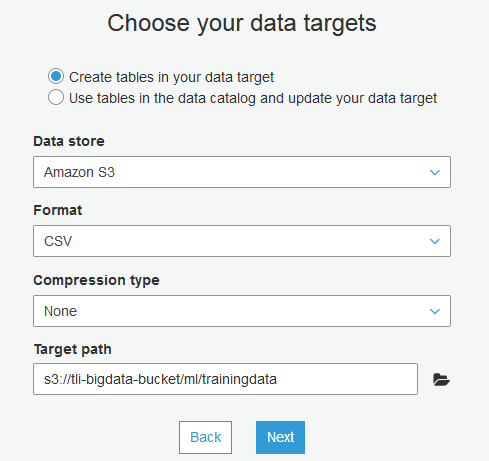
In this lab, you will continue to explore on how to use the new Amazon SageMaker service to train, create and host a machine learning model to perform anomaly detection similar to Amazon Kinesis Analytics in lab 1. You will use one of built-in SageMaker machine learning algorithms called [K-Means algorithm](https://docs.aws.amazon.com/sagemaker/latest/dg/k-means.html) to analyze the data for anomaly. You will also leverage AWS Glue to perform ETL and prepare the data we need for training the model.

Diagram below with highlighted area depicts what you will be building in this lab.

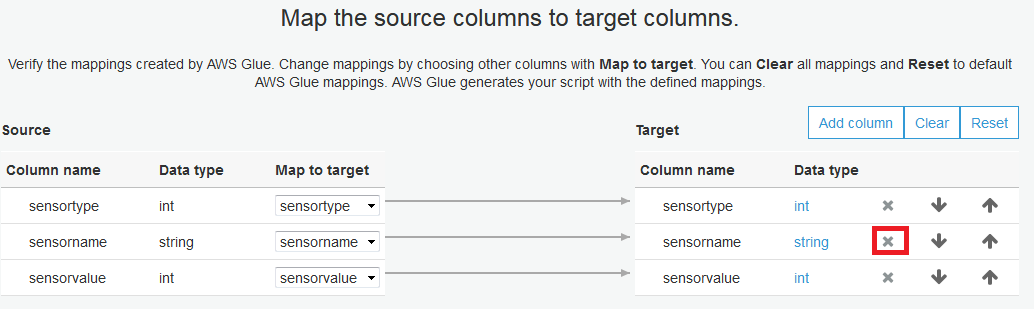


# Prepare Training Data with AWS Glue

1. Recall in the first lab, raw data generated by the Lambda sensor simulator is in JSON format. Amazon SageMaker [K-Means algorithm](https://docs.aws.amazon.com/sagemaker/latest/dg/k-means.html) supports training data in CSV format. The algorithm also only supports numeric fields, since raw data contains sensor name as text, we will need to remove that field. We will leverage AWS Glue to transform the raw data as discussed.
2. Please use Chrome or Firefox browser to ensure smooth lab experience.
3. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
4. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
5. Click on **Glue** from the list of all services. This will bring you to the AWS Glue dashboard page.
6. Click on **Jobs** on the left panel and then click **Add job** 
7. In Job properties page, enter the following
   * Name: **YourInitials\_bigdata\_ml**
   * IAM role: **YouInitialsGlueServiceRole**
   * The job runs: **A proposed script generated by AWS Glue**
   * Script file name: **YourInitials\_bigdata\_ml**
   * Temporary directory: Copy and paste the S3 bucket path from S3 path above and append **/temp** behind the path. For example, s3://aws-glue-scripts-1234567890-us-east-1/**YourInitials(or root)/**temp
   * Expand **Script libraries and job** **parameters** section and change Concurrent DPUs per job run from **10** to **100**. This will help speed up the transformation process.
   * Leave everything else default
8. Click **Next**
9. Select **raw2018** or your raw table and click **Next**
10. In Data target page, choose **Create tables in your data target**. Select **Amazon S3** as the Data store and **CSV** as the Format. For Target path, use the same big data S3 bucket for the lab and append **ml**. For example, s3://**YourInitials**-bigdata-bucket/ml/trainingdata



1. Click **Next**
2. In field mapping page, remove sensorname field in the Target section. This will remove the text field that would cause issue for traning with [K-Means algorithm](https://docs.aws.amazon.com/sagemaker/latest/dg/k-means.html)



1. Review the configuration and click **Finish**
2. In the Glue ETL editor page, copy and replace the following code in the code window, change the code highlighted in yellow to point to your S3 bucket and table/database names, then click **Save**. The code will remove the CSV column headers in the first row, this is needed as [K-Means algorithm](https://docs.aws.amazon.com/sagemaker/latest/dg/k-means.html) will not need it and it contains texts which would cause issue.

import sys

from awsglue.transforms import \*

from awsglue.utils import getResolvedOptions

from pyspark.context import SparkContext

from awsglue.context import GlueContext

from awsglue.job import Job

## @params: [JOB\_NAME]

args = getResolvedOptions(sys.argv, ['JOB\_NAME'])

sc = SparkContext()

glueContext = GlueContext(sc)

spark = glueContext.spark\_session

job = Job(glueContext)

job.init(args['JOB\_NAME'], args)

## @type: DataSource

## @args: [database = "**YourInitial**\_bigdata", table\_name = "**raw2018**", transformation\_ctx = "datasource0"]

## @return: datasource0

## @inputs: []

datasource0 = glueContext.create\_dynamic\_frame.from\_catalog(database = "**YourInitial**\_bigdata", table\_name = "**raw2018**", transformation\_ctx = "datasource0")

## @type: ApplyMapping

## @args: [mapping = [("sensortype", "int", "sensortype", "int"), ("sensorvalue", "int", "sensorvalue", "int")], transformation\_ctx = "applymapping1"]

## @return: applymapping1

## @inputs: [frame = datasource0]

applymapping1 = ApplyMapping.apply(frame = datasource0, mappings = [("sensortype", "int", "sensortype", "int"), ("sensortype", "int", "sensortype1", "int"), ("sensorvalue", "int", "sensorvalue", "int")], transformation\_ctx = "applymapping1")

## @type: DataSink

## @args: [connection\_type = "s3", connection\_options = {"path": "s3://**YourInitial**-bigdata-bucket/ml/trainingdata"}, format = "csv", transformation\_ctx = "datasink2"]

## @return: datasink2

## @inputs: [frame = applymapping1]

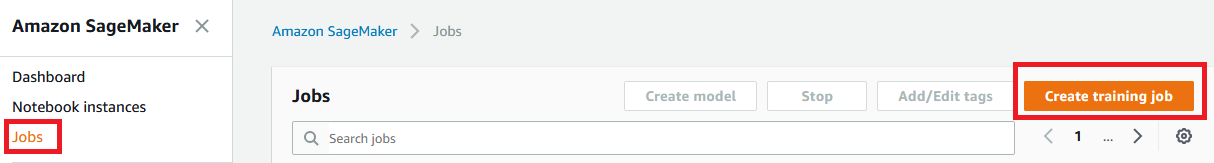
datasink2 = glueContext.write\_dynamic\_frame.from\_options(frame = applymapping1, connection\_type = "s3", connection\_options = {"path": "s3://**YourInitial**-bigdata-bucket/ml/trainingdata"}, format = "csv", format\_options = {"writeHeader": False}, transformation\_ctx = "datasink2")

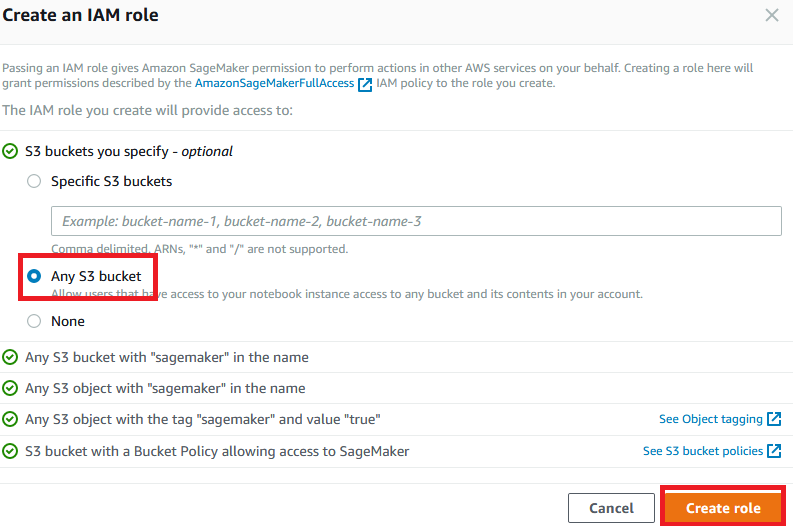
job.commit()

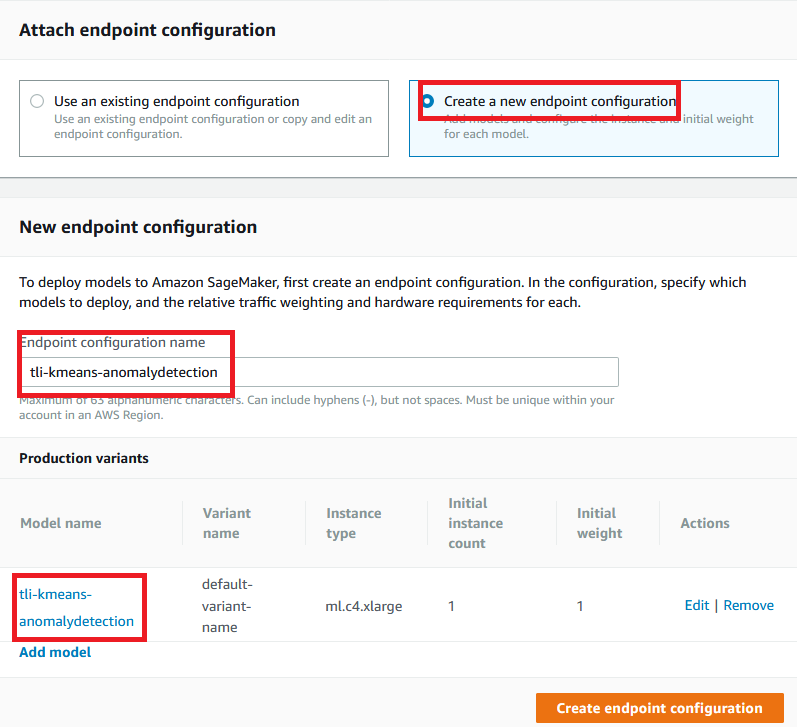
1. Click on the **X** button on the top right hand corner to exit the code editor page.
2. Back in Jobs page, tick the job checkbox and click Action then select Run Job.
3. Take a break, this could take over 10 minutes to complete.

# Train, Create and Host a Machine Learning Model with Amazon SageMaker

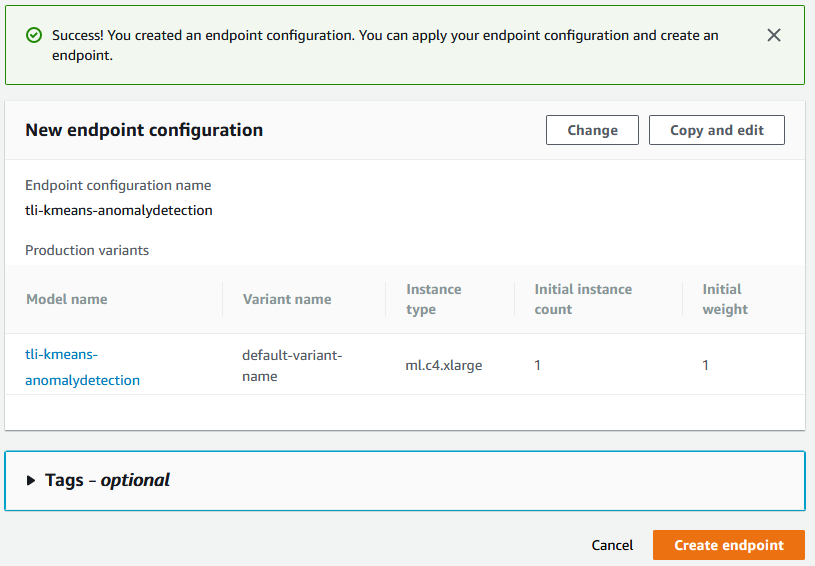
1. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
2. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
3. Click on **Amazon SageMaker** from the list of all services. This will bring you to the IAM dashboard page.
4. Click **Jobs** on the left hand panel and then click **Create training job**



1. Name it **YourInitials-kmeans-anomalydetection**
2. In IAM role, select **Create a new role** and select **Any S3 bucket** in the dialog, then click **Create role**
3. Continue and enter the following in **Job Settings** section
   * Algorithm: **K-Means**
   * Leave everything else default
4. Enter the following in **Hyperparameters** section
   * k: **4** (we are creating 4 clusters as there are 4 sensors from the simulator)
   * feature\_dim: **2** (we have 2 features – sensortype and sensorvalue in the training data)
   * Leave everything else default
5. Enter the following in **Input data configuration** section
   * Content type - optional: **text/csv**
   * S3 location: s3://**YourInitials**-bigdata-bucket/ml/trainingdata
   * Leave everything else default
6. Enter the following in Output data configuration section
   * S3 output path: s3://**YourInitials**-bigdata-bucket/ml/trainingoutput
   * Leave everything else default
7. Click **Create training**. It will take about 7 - 9 minutes to complete
8. Once the status displays **Completed**, click **YourInitials-kmeans-anomalydetection** link, then click **Create model** from top right hand corner.
9. Name the model as **YourInitials-kmeans-anomalydetection**
10. In Location of model artifacts – optional, enter the location of the gzip artifact, for example, s3://**YourInitials**-bigdata-bucket/ml/trainingoutput/**YourInitials**-kmeans-anomalydetection/output/model.tar.gz
11. Leave everything else default and click **Create model**
12. In the Models page, click **YourInitials-kmeans-anomalydetection** link, then click **Create endpoint**
13. Name the endpoint as **YourInitials-kmeans-anomalydetection**
14. Select create a new endpoint configuration, in Endpoint configuration name, enter **YourInitials-kmeans-anomalydetection** and make sure correct model is selected.



1. Click **Create endpoint configuration**
2. The screen will display endpoint configuration successfully created, click **Create endpoint**

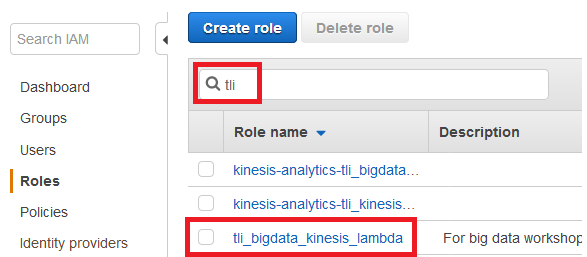


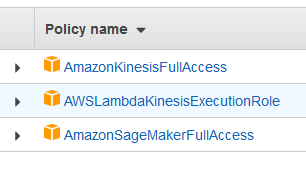
1. The model has now been trained, deployed and hosted on Amazon SageMaker.
2. Click the newly created endpoint, the URL is the endpoint you’re your application code to call for prediction, for example, [https://runtime.sagemaker.**YourRegion**.amazonaws.com/endpoints/**YourInitials**-kmeans-anomalydetection/invocations](https://runtime.sagemaker.YourRegion.amazonaws.com/endpoints/YourInitials-kmeans-anomalydetection/invocations)

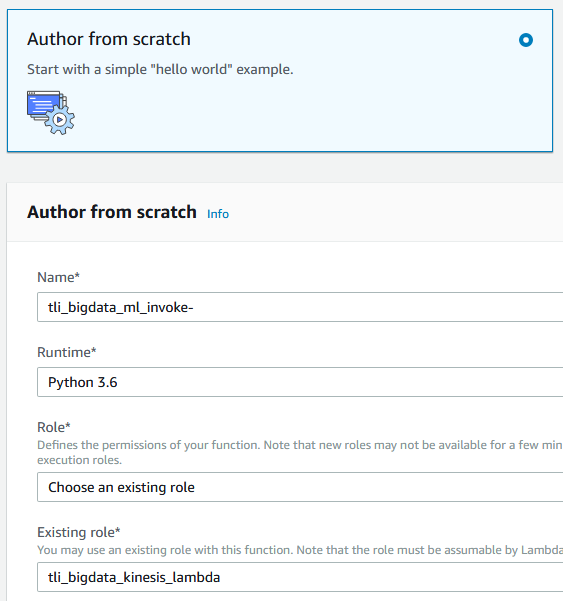
Note: Do not click on the link as SageMaker endpoint do not accept Get requests, only Post requests. If you do click on the link, you will get an error, but it causes no harm. In the next section, we will create a simple Lambda function acting as a client to invoke the endpoint for a prediction.

# Invoke Amazon SageMaker Endpoint for Prediction/Inference

1. Sign into the AWS Management Console <https://console.aws.amazon.com/>.
2. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
3. Click on **IAM** from the list of all services. This will bring you to the IAM dashboard page.
4. Click **Roles** on the left hand panel and then enter **YourInitials** in the search box and look for **YourInitials\_bigdata\_kinesis\_lambda** role



1. Click on **YourInitials\_bigdata\_kinesis\_lambda** role
2. Click **Attach Policy**, search for **AmazonSageMakerFullAccess** then click **Attach Policy.** **YourInitials\_bigdata\_kinesis\_lambda** role should now have the following 3 policies. 
3. Go back to AWS Management Console <https://console.aws.amazon.com/>.
4. In the upper-right corner of the AWS Management Console, confirm you are in the desired AWS region (e.g., N. Virginia).
5. Click on **Lambda** from the list of all services. This will bring you to the AWS Lambda dashboard page.
6. On the Lambda Dashboard, click **Create Function**
7. Select **Author from scratch** and enter the following
   * Name: **YourInitials\_bigdata\_ml\_invoke**
   * Runtime: Python 3.6
   * Role: Choose an existing role
   * Exiting Role: **YourInitials\_bigdata\_kinesis\_lambda**



1. Click **Create function**
2. In code editor, copy and paste the following code, be sure to change variable **endpoint\_name** to the endpoint created with your initials. Note the variable **payload** contains 4 sensors, each with a normal and an abnormal reading.

import boto3

import json

def lambda\_handler(event, context):

payload = '1,13000 \n 1,20000 \n 2,3500 \n 2,5000 \n 3,3000 \n 3,3300 \n 4,2 \n 4,10'

endpoint\_name = **'YourInitials-kmeans-anomalydetection'**

runtime = boto3.client('runtime.sagemaker')

response = runtime.invoke\_endpoint(EndpointName=endpoint\_name,

ContentType='text/csv',

Body=payload)

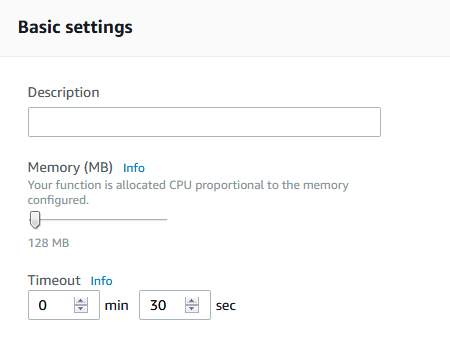
result = json.loads(response['Body'].read().decode())

for p in result['predictions']:

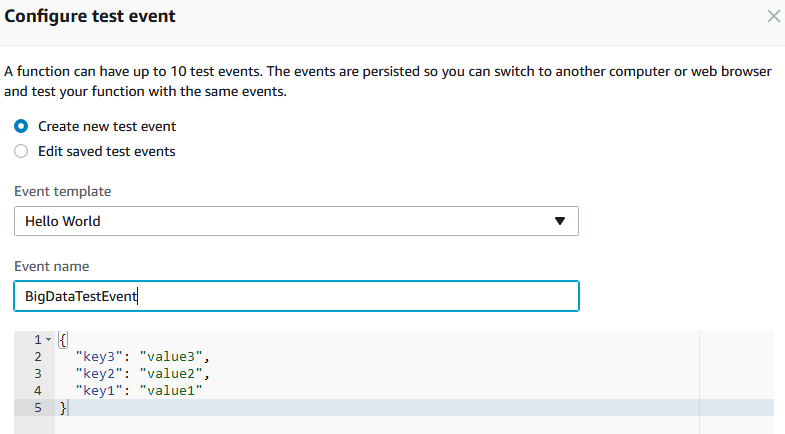
print (p)

return 'Complete'

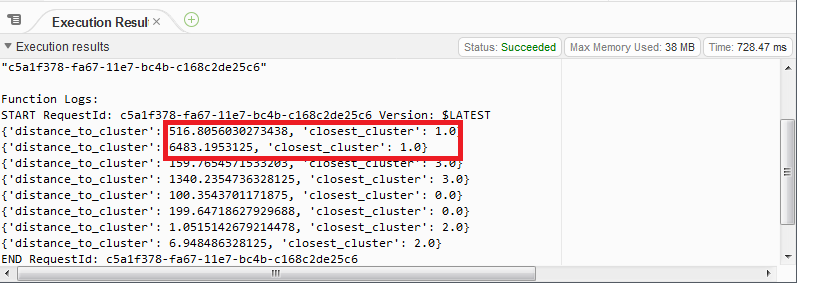
1. Leave everything on the page default except the Timeout value, change it from 3 seconds to 30 seconds



1. Click **Save** on the top right hand corner of the screen and then click **Test**. Since we are not providing any parameter or input values, leave everything default, give it a name, and click **Create**.



1. The function will invoke the Amazon SageMaker endpoint for a prediction. Looking at the output from the function, we can see there are 4 clusters (0-3), one for each sensor. Also note that abnormal readings have a higher distance to cluster than normal readings. You can program this function or your client application to react to this anomaly based on business requirements.



# Conclusion

In this lab, you have learned how to use the new Amazon SageMaker service to train, create and host a machine learning model to perform anomaly detection. You also used the built-in SageMaker machine learning algorithms called [K-Means algorithm](https://docs.aws.amazon.com/sagemaker/latest/dg/k-means.html) to analyze the data store on S3 for anomaly. In addition, you have learned how to AWS Glue to perform ETL and prepare the data needed for training the model. Finally, you have learned how to invoke the endpoint for prediction.