

High Integrity Systems Euro Currency Note Identification using AWS Sagemaker

Summer Semester 2022 (Professor Dr. Christian Baun)

Group Members:

Muhamad Haseeb Anwar	1381906
Moeez Ur Rehman	1378333
Sahrish Kanwal	1382303
Harmain Haider	1359598



But... It's Just a talk....:(Oops... sorry...

Agenda

- What is Amazon SageMaker?
- What are the main capability of Amazon SageMaker?
- Components of Amazon SageMaker
- How Amazon SageMaker works with other AWS AI Services
- About SageMaker inference endpoint, Aws lambda functions, Aws Api gateway output, AWS Amplify
- Machine Learning Model Workflow
- About S3
- How do I get started using Amazon SageMaker?
 - Create, Train, Deploy your Model
- Q&A

Amazon SageMaker

Amazon SageMaker is a cloud machine-learning platform that was launched in November 2017. SageMaker enables developers to create, train, and deploy machine-learning models in the cloud. SageMaker also enables developers to deploy ML models on embedded systems.

OR

A managed service that provides the quickest and easiest way for your **data scientists** and **developers** to get **Machine Learning** models from idea to production.

What are the main capability of Amazon SageMaker?

Amazon SageMaker enables you to deploy inference pipelines so you can pass raw input data and execute pre-processing, predictions, and post-processing on real-time and batch inference. You can build feature data processing and feature engineering pipelines, and deploy these as part of the inference pipelines.

Amazon SageMaker's Components



- Hyperparameter tuning job. ...
- Selecting the best hyperparameters. ...
- Training job with the best hyperparameters. ...
- Creating a model for deployment. ...
- Deploying the inference endpoint.

AWS machine learning infrastructure

You can choose a broad set of Machine Learning Services

AWS offers below mentioned services

- Workflow services, make it easier for you to manage and scale your underlying ML infrastructure.
- **Framework**, The next layer highlights that AWS ML infrastructure supports all of the leading ML frameworks.
- **Compute Networking and storage**, The bottom layer shows examples of compute, networking, and storage services that constitute the foundational blocks of ML infrastructure.



Ref: https://aws.amazon.com/machine-learning/infrastructure/

AWS lambda functions

• AWS Lambda is a serverless compute service that runs your code in response to events and automatically manages the underlying compute resources for you.

AWS Api gateway endpoint

• API Gateway is an AWS service that supports the following: Creating, deploying, and managing a RESTful application programming interface (API) to expose backend HTTP endpoints, AWS Lambda functions, or other AWS services.

lacksquare

AWS Amplify

• AWS Amplify is a set of purpose-built tools and features that lets frontend web and mobile developers quickly and easily build full-stack applications on AWS, with the flexibility to leverage the breadth of AWS services as your use cases evolve.

Machine Learning Model Workflow



Ref: https://docs.aws.amazon.com/sagemaker/latest/dg/how-it-works-mlconcepts.html

Creation of S3 Bucket

Services Q Search for services, features, blogs, docs, and more [Alt+S]
Amazon S3 > Buckets > Create bucket
Create bucket Info Buckets are containers for data stored in S3. Learn more
General configuration
Bucket name
cloud-computing-dataset
Bucket name must be unique and must not contain spaces or uppercase letters. See rules for bucket naming 🔀
AWS Region
EU (Frankfurt) eu-central-1
Copy settings from existing bucket - optional Only the bucket settings in the following configuration are copied. Choose bucket

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Setting up Security

Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

ACLs disabled (recommended)

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

Block all public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- Block public access to buckets and objects granted through new access control lists (ACLs) S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- Block public access to buckets and objects granted through any access control lists (ACLs)
 S3 will ignore all ACLs that grant public access to buckets and objects.
- So will block new buckets and access to buckets and objects granted through *new* public bucket or access point policies So will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- Block public and cross-account access to buckets and objects through any public bucket or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Advanced settings

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

	Cancel	Create bucket
Amazon S3 > Buckets > cloud-computing-dataset		
cloud-computing-dataset Info		
Objects Properties Permissions Metrics Management Access Points Objects (1) Image: Comparison of the second		
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory 2 to get a list of all objects in your bucket. For o grant them permissions. Learn more 2 C C Copy S3 URI C Copy S3 URI Copy URL Download Open 2 Delete Actions	Thers to access your ob	jects, you'll need to explicitly
Q Find objects by prefix		< 1 > 💿
□ Name ▲ Type マ Last modified マ Size	⊽ Sto	rage class 🛛 🗸
□ CC-Dataset/ Folder -		

CC-Dataset/



Objects	Properties
---------	------------

Objects them p	cts (8) are the fundamental entities sermissions. Learn more	stored in Amazon S3. You can	use Amazon <mark>S3 inve</mark>	entory <mark>7</mark> to get a list	of all objects in y	our bucket. For others to	o access your objects, you	u'll need to explicitly grant
C	Copy S3 URI	Copy URL	☑ Download	Open 🛽	Delete	Actions v	Create folder	🗗 Upload
Q F	ind objects by prefix]				< 1 > ③
	Name	▲ Туре	\bigtriangledown	Last modified		▼ Size		class 🗢
	L EURO-10/	Folder		-				
	EURO-20/	Folder		121				
	EURO-5/	Folder		~			-	
	EURO-50/	Folder						

aws Services Q S	Search for ser	vices, features, bl	logs, docs, and more	[Alt+	S]				\$	0	Global 🔻	MuhammadH	laseebAnwar 🤻
Amazon S3	×	Amazon S3	> Buckets > clou	d-computing-dataset	> CC-Dataset/ >	EURO-10/							
Buckets Access Points Object Lambda Access Points Multi-Region Access Points Batch Operations	S	EURO Objects	-10/ Properties									Copy S	S3 URI
Access analyzer for S3 Block Public Access settings this account	for	Objects ar them perm C	e the fundamental entities nissions. Learn more Copy S3 URI d objects by prefix	stored in Amazon S3. You	can use Amazon S3 inver	otory 🕻 to get a list	of all objects in y Delete	our bucket. For othe	ers to access	your obje ate folc	cts, you'll r	need to explicitly gra	ant O
 Storage Lens 													
Dashboards			Name	4	Type ♥	Last modified			▼	Size	▼	Storage class	▽
AWS Organizations settings			MG_20220618_14	0650.jpg	jpg	June 18, 2022,	19:01:47 (UTC	+02:00)		8.9	КВ	Standard	
			MG_20220618_14	0652.jpg	jpg	June 18, 2022,	19:0 <mark>1</mark> :47 (UTC	+02:00)		8.5	KB	Standard	
Feature spotlight 3			MG_20220618_14	0653.jpg	jpg	June 18, 2022,	19:0 <mark>1</mark> :47 (UTC	+02:00)		8.9	KB	Standard	
			MG_20220618_14	0654.jpg	jpg	June 18, 2022,	19:02:22 (UTC	+02:00)		9.6	KB	Standard	
AWS Marketplace for S3			MG_20220618_14	0655.jpg	jpg	June 18, 2022,	19:02:22 (UTC	+02:00)		9.6	КВ	Standard	
8 8 8 8 9 9 11 04781 20 82409			MG_20220618_14	0656.jpg	jpg	June 18, 2022,	19:02:2 <mark>3 (</mark> UTC	+02:00)		8.0	KB	Standard	
			MG_20220618_14	0657.jpg	jpg	June 18, 2022,	19:02:23 (UTC	+02:00)		8.3	KB	Standard	
			MG_20220618_14	0659.jpg	jpg	June 18, 2022,	19:02:23 (UTC	+02:00)		8.3	KB	Standard	

How do I get started using Amazon SageMaker?

- Create Notebook instance within sagemaker, choose your desired region, we have chosen eu-central-1
- Provide name of your notebook instance, and choose your instance type
- create a new IAM role or use already created IAM role
- After our notebook instance is created, we need to start this notebook instance after that we are going to write our model



, feature:	s, blogs, docs, and more [Alt+S]			۶ A	0	Frankfurt 🔻	MuhammadHaseebAnw
Amazon	SageMaker 〉 Notebook instances						
Note	ebook instances			C Act	ions 🔻	Create no	tebook instance
Q	Search notebook instances						< 1 > ⊚
	Name	⊽	Instance	Creation time	•	Status	▼ Actions
0	Cloud-Computing-Project-Image-Recognition-NB-Instance		ml.t2.medium	May 29, 2022 13:35 U	тс	Stopped	Start

Create your model

- Stored the name of S3 bucket and primary folder into variables so you can access them later.
- Importing the sagemaker library and setting up environment.
- We have used Sagemaker's built in Image classification Algorithm which is based on Supervised Learning and uses Conventional Neural Networks (CNN).

```
######### Team Members:
######### Muhammad Haseeb Anwar
######## Moeez Ur Rehman
######### Sehrish Kanwal
######## Harmain Haidar
# 53 Bucket Name
bucket_name='cloud-computing-dataset'
# Our Main Folder inside the S3 bucket Which has subfolders of our classes
# One Sub-Folder will be considered as One Class
dataset_name = 'CC-Dataset'
print('Name of the bucket is: '+dataset name)
print('Name of dataset folder is: '+bucket name)
Name of the bucket is: CC-Dataset
Name of dataset folder is: cloud-computing-dataset
```

```
[13]: #Setting Up Our Environment
```

```
# Importing Sagemaker
# getting execution role of notebook
# defining algorithm type in Image_Uri method
import sagemaker
from sagemaker import get_execution_role
from sagemaker.amazon.amazon_estimator import get_image_uri
role = get_execution_role()
session = sagemaker.Session()
#sagemaker.image_uris.retrieve
#get_image_uri
image_uris = sagemaker.image_uris.retrieve(region=session.boto_region_name, framework ='image-classification')
roit/incide news_incide bath environment)
```

```
print('Region name: '+session.boto_region_name)
print('Algorithm Used: image-classification ')
```

```
Region name: eu-central-1
Algorithm Used: image-classification
```

- For data loading here, we are using **RecordIO** format for training. (RecordIO format converts images into binary data exchange formats)
- Recommendation: Sagemaker recommended storing images as records and packing them together, the major benefit is that Stored images in RecordIO format greatly reduce the size of the dataset on the disk.

[7]: # The SageMaker Image Classification algorithm supports both RecordIO and Image (jpg, Jpeg) formats # to train the images, in our project we are going to use the RecordIO format for training.

Why RecordIO?

Sagemaker recommend storing images as records and packing them together, the major benifit is # Storing images in Recordio format greatly reduces the size of the dataset on the disk.

#here we specifiy the path of the script which converts images into RecordIO files

BASE_DIRECTORY='/tmp'

%env BASE_DIRECTORY=\$BASE_DIRECTORY %env S3_BUCKET_NAME = \$bucket_name %env DATASET_NAME = \$dataset_name

import sys,os

suffix='/mxnet/tools/im2rec.py'
im2rec = list(filter((lambda x: os.path.isfile(x + suffix)), sys.path))[0] + suffix
%env IM2REC=\$im2rec

env: BASE_DIR=/tmp

env: S3_DATA_BUCKET_NAME=cloud-computing-dataset

env: DATASET_NAME=CC-Dataset

 $env: IM2REC=/home/ec2-user/anaconda3/envs/mxnet_p36/lib/python3.6/site-packages/mxnet/tools/im2rec.python3.6/$

[8]: # The script below Pulls our images from S3 bucket

!aws s3 sync s3://\$S3_BUCKET_NAME/\$DATASET_NAME \$BASE_DIRECTORY/\$DATASET_NAME --quiet

```
print('Images have been Pulled!!! ')
```

Images have been Pulled!!!

- Transform the fetched images into RecordIO file, we have kept the training ratio to 70%, while Testing ratio to 30%. The RecordIO files will be created in this step with the above ratio.
- Upload created RecordIO files back into the S3 bucket, which then be used as an input for training of our model.

```
%%bash
# Now here we use the IM2REC script to convert our images which we fetched from S3 bucket into RecordIO files
# Delete if there are already created Recio files in our working directory
cd $BASE_DIR
rm *.rec
rm *.1st
# We want to create 2 LST files first, One for training and One for testing, along with saving the class of each image
# The output of the LST files command includes a list of all of our label classes
# We are specifying here the training and testing ration, 70% Training Ratio and 30% Testing Ratio
echo "Creating LST files"
python $IM2REC --list --recursive --pass-through --test-ratio=0.3 --train-ratio=0.7 $DATASET NAME $DATASET NAME > ${DATASET NAME } classes
echo "Label classes:"
cat ${DATASET_NAME}_classes
# Then we create RecordIO files from the LST files
echo "Creating RecordIO files"
python $IM2REC -- num-thread=4 ${DATASET_NAME}_train.lst $DATASET_NAME
python $IM2REC -- num-thread=4 ${DATASET_NAME}_test.lst $DATASET_NAME
ls -lh *.rec
Creating LST files
Label classes
EUR0-10 0
EUR0-20 1
EURO-5 2
EURO-58 3
Creating RecordIO files
Creating .rec file from /tmp/CC-Dataset train.lst in /tmp
time: 0.39785265922546387 count: 0
Creating .rec file from /tmp/CC-Dataset_test.lst in /tmp
```

```
[15]:
       # We are now Uploading our train and test RecordIO files to S3 bucket
      bucket = bucket name
      print (bucket)
       training_path_s3 = 's3://{}/{}/train/'.format(bucket, dataset_name)
       validation_path_S3 = 's3://{}/{}/validation/'.format(bucket, dataset_name)
       print(training path s3)
       print(validation path S3)
       # Delete any existing data
       !aws s3 rm s3://{bucket}/{dataset name}/train --recursive
       !aws s3 rm s3://{bucket}/{dataset name}/validation --recursive
       # Upload the rec files to the train and validation folders
       !aws s3 cp /tmp/{dataset name} train.rec $training path s3
       !aws s3 cp /tmp/{dataset_name}_test.rec $validation_path_S3
      cloud-computing-dataset
       s3://cloud-computing-dataset/CC-Dataset/train/
       s3://cloud-computing-dataset/CC-Dataset/validation/
      upload: ../../tmp/CC-Dataset train.rec to s3://cloud-computing-dataset/CC-Dataset/train/CC-Dataset train.rec
      upload: ../../tmp/CC-Dataset test.rec to s3://cloud-computing-dataset/CC-Dataset/validation/CC-Dataset test.rec
```

Model looks like..

- The uploaded RecordIO files in our S3 bucket will look like this.
- We have now done our preprocessing; the data is ready to be trained. Now are going towards the process of training our model using the created RecordIO files.

ain/					🗇 Copy S
Objects Properties					
Objects (1) Objects are the fundamental entities s your objects, you'll need to explicitly o	stored in Amazon 53. You	can use Amazon S3 invent	tory 🛃 to get a list	of all objects in y	your bucket. For others to acc
C D Copy S3 URI	🗇 Copy URL	Download	Open 🖄	Delete	Actions 🔻
Create folder	oad				
Q Find objects by prefix					< 1 >
Name	▲ Type ⊽	Last modified		⊽ Siz	e 🔻 Storage class
Name	▲ Type ⊽ c rec	Last modified July 1, 2022, 00:34	4:46 (UTC+02:00)	⊽ Siz 8.	e V Storage class
Name CC-Dataset_train.red Amazon S3 > Buckets > clou validation/ Objects Properties Objects (1) Objects are the fundemental participants	▲ Type ▼ c rec d-computing-dataset	Last modified July 1, 2022, 00:34	4:46 (UTC+02:00)	♥ Siz 8.	e ♥ Storage class
Name CC-Dataset_train.red Amazon S3 > Buckets > clou validation/ Objects Properties Objects (1) Objects you'll need to explicitly	▲ Type ▼ c rec ud-computing-dataset	Last modified July 1, 2022, 00:34	4:46 (UTC+02:00)	▼ Siz 8. all objects in your	se v Storage class
Name CC-Dataset_train.res Amazon S3 > Buckets > clou validation/ Objects Properties Objects (1) Objects are the fundamental entities your objects, you'll need to explicitly C Copy S3 URI	Type ▼ rec dc-computing-dataset sstored in Amazon S3. You grant them permissions. Lu C Copy URL	Last modified July 1, 2022, 00:34	4:46 (UTC+02:00)	▼ Siz 8. 8. all objects in your Delete	e V Storage class
Name CC-Dataset_train.red Amazon S3 > Buckets > clou Validation/ Objects Properties Objects (1) Objects validation Citer of the fundamental entities Validation Citer of the fundamental entities Validation Objects (1) Objects you'll need to explicitly C Copy S3 URI Create folder C UP	▲ Type ▼ c rec d-computing-dataset stored in Amazon S3. You grant them permissions. Lu © Copy URL sload	Last modified July 1, 2022, 00:34	4:46 (UTC+02:00) alidation/	▼ Siz 8. all objects in your Delete	e V Storage class

Train your model

- Define the RecordIO paths to the training and validation functions by using inputs.TrainingInput() method.
- Define the output location of the model and initialize the estimator function in parallel.

```
[20]:
       # Documentation of the function sagemaker.inputs.TrainingInput is available here
       # https://sagemaker.readthedocs.io/en/stable/api/utility/inputs.html#sagemaker.inputs.TrainingInput
       # Create a definition for input data used by an SageMaker training job.
       train data = sagemaker.inputs.TrainingInput(
           training_path_s3,
           distribution='FullyReplicated',
           content type='application/x-recordio',
           s3_data_type='S3Prefix'
        validation_data = sagemaker.inputs.TrainingInput(
           validation path S3,
           distribution='FullyReplicated',
           content type='application/x-recordio',
           s3 data type='S3Prefix'
       data channels = { 'train ': train data, 'validation': validation data }
       print(train_data)
       print(validation_data)
```

<sagemaker.inputs.TrainingInput object at 0x7f813d49e2e8> <sagemaker.inputs.TrainingInput object at 0x7f813d49e320>

```
3]: # The are defining the output location for model
s3_output_location = 's3://{}/{output'.format(bucket, dataset_name)
# we have used ml.p3.2xlarge isntance for traning
image_classifier = sagemaker.estimator.Estimator(
    role=role,
    image_uri=image_uris,
    instance_count=1,
    instance_type='ml.p3.2xlarge',
    output_path=s3_output_location,
    sagemaker_session=session
)
print('done')
done
```

- Image classification Hyperparameters,
 - image shape as 3,244,244 which is same as the image shape of our RecordIO files.
 - number of classes which in our case are 4,
 - Augmentation type here is important as we are taking the color into account, so we have chosen 'crop color'.
- The training job is started and will provide the path of the model where it will be stored.

```
[58]: num_classes=! ls -l {base_dir}/{dataset_name} | wc -l
      num_classes=int(num_classes[0]) - 3
      num_training_samples=! cat {base_dir}/{dataset_name}_train.lst | wc -1
      num_training_samples = int(num_training_samples[0])
      # Details on Sagemaker built-in Image Classifier hyperparameters
      # available here: https://docs.aws.amazon.com/sagemaker/latest/dg/IC-Hyperparameter.html
      base_hyperparameters=dict(
          use_pretrained_model=1,
          image shape='3,224,224',
          num classes=num classes.
          augmentation_type='crop_color', #taking corresponding Hue-Saturation-Lightness into account
          num_training_samples=num_training_samples,
      # These are hyperparameters are important which can affect the model training success:
      hyperparameters={
          **base_hyperparameters,
          **dict(
              learning_rate=0.001,
              mini_batch_size=5,
      image_classifier.set_hyperparameters(**hyperparameters)
      hyperparameters
      print('No of tranining Samples: '+str(num_training_samples))
      print ('No of Classes: '+str(num_classes))
      No of tranining Samples: 464
```

print(f"\n\n Finished training! The model is available for download at: {image_classifier.output_path}/{job.name}/output/model.tar.gz")



import time

now = str(int(time.time()))

job = image_classifier.latest_training_job model_path = f"{BASE_DIR}/{job.name}"

No of Classes: 4

training_job_name = 'IC-' + dataset_name.replace('_', '-') + '-' + now

2022-06-30 23:09:40 Starting - Preparing the instances for training......

2022-06-30 23:10:43 Downloading - Downloading input data..

image_classifier.fit(inputs=data_channels, job_name=training_job_name, logs=True)

2022-06-30 23:08:48 Starting - Starting the training job...ProfilerReport-1656630528: InProgress

2022-06-30 23:13:04 Training - Training image download completed. Training in progress.[23:13:11] /opt/brazil-pkg-cache/packages/AIAlgorithm sMXNet/AIAlgorithmsMXNet-1.3.x ecl Cuda 10.1.x.11282.0/AL2 x86 64/generic-flavor/src/src/operator/nn/./cudnn/./cudnn algoreg-inl.h:97: Runni ng performance tests to find the best convolution algorithm, this can take a while... (setting env variable MXNET_CUDNN_AUTOTUNE_DEFAULT to 0 to disable) [06/30/2022 23:13:15 INFO 140493106341696] Epoch[0] Batch [20]#011Speed: 23.524 samples/sec#011accuracy=0.428571 [06/30/2022 23:13:16 INFO 140493106341696] Epoch[0] Batch [40]#011Speed: 32.955 samples/sec#011accuracy=0.609756 [06/30/2022 23:13:18 INFO 140493106341696] Epoch[0] Batch [60]#011Speed: 38.072 samples/sec#011accuracy=0.708197 [06/30/2022 23:13:20 INFO 140493106341696] Epoch[0] Batch [80]#011Speed: 41.102 samples/sec#011accuracy=0.767901 [06/30/2022 23:13:21 INFO 140493106341696] Epoch[0] Train-accuracy=0.784783 [06/30/2022 23:13:21 INFO 140493106341696] Epoch[0] Time cost=10.725 [06/30/2022 23:13:22 INFO 140493106341696] Epoch[0] Validation-accuracy=0.995000 [06/30/2022 23:13:23 INFO 140493106341696] Storing the best model with validation accuracy: 0.995000 [06/30/2022 23:13:23 INFO 140493106341696] Saved checkpoint to "/opt/ml/model/image-classification-0001.params" [06/30/2022 23:13:25 INFO 140493106341696] Epoch[1] Batch [20]#011Speed: 55.178 samples/sec#011accuracy=0.895238 [06/30/2022 23:13:27 INFO 140493106341696] Epoch[1] Batch [40]#011Speed: 55.481 samples/sec#011accuracy=0.902439 [06/30/2022 23:13:28 INFO 1404931063416961 Epoch[1] Batch [60]#011Speed: 55.671 samples/sec#011accuracy=0.927869 [06/30/2022 23:13:30 INFO 140493106341696] Epoch[1] Batch [80]#011Speed: 55.449 samples/sec#011accuracy=0.938272 [06/30/2022 23:13:31 INFO 140493106341696] Epoch[1] Train-accuracy=0.936957 [06/30/2022 23:13:31 INFO 140493106341696] Epoch[1] Time cost=8.199 [06/30/2022 23:13:32 INFO 140493106341696] Epoch[1] Validation-accuracy=1.000000 [06/30/2022 23:13:33 INFO 140493106341696] Storing the best model with validation accuracy: 1.000000 [06/30/2022 23:13:33 INFO 140493106341696] Saved checkpoint to "/opt/ml/model/image-classification-0002.params" [06/30/2022 23:13:35 INFO 140493106341696] Epoch[2] Batch [20]#011Speed: 54.774 samples/sec#011accuracy=0.914286 [06/30/2022 23:13:37 INFO 140493106341696] Epoch[2] Batch [40]#011Speed: 55.316 samples/sec#011accuracy=0.951220 [06/30/2022 23:13:39 INFO 140493106341696] Epoch[2] Batch [60]#011Speed: 55.483 samples/sec#011accuracy=0.963934 [06/30/2022 23:13:40 INFO 140493106341696] Epoch[2] Batch [80]#011Speed: 55.151 samples/sec#011accuracy=0.967901 [06/30/2022 23:13:41 INFO 140493106341696] Epoch[2] Train-accuracy=0.971739 [06/30/2022 23:13:41 INFO 140493106341696] Epoch[2] Time cost=8.231 [06/30/2022 23:13:42 INFO 140493106341696] Epoch[2] Validation-accuracy=1.000000 [06/30/2022 23:13:45 INFO 140493106341696] Epoch[3] Batch [20]#011Speed: 53.828 samples/sec#011accuracy=0.990476 [06/30/2022 23:13:47 INFO 140493106341696] Epoch[3] Batch [40]#011Speed: 55.007 samples/sec#011accuracy=0.990244 [06/30/2022 23:13:48 INFO 140493106341696] Epoch[3] Batch [60]#0115peed: 55.255 samples/sec#011accuracy=0.977049 [06/30/2022 23:13:50 INFO 140493106341696] Epoch[3] Batch [80]#011Speed: 55,150 samples/sec#011accuracy=0.980247 [06/30/2022 23:13:51 INFO 140493106341696] Epoch[3] Train-accuracy=0.976087 [06/30/2022 23:13:51 INFO 140493106341696] Epoch[3] Time cost=8.235

print(f"\n\n Finished training! The model is available for download at: {image_classifier.output_path}/{job.name}/output/model.tar.gz")

Finished training! The model is available for download at: s3://cloud-computing-dataset/CC-Dataset/output/IC-CC-Dataset-1656630528/output/m odel.tar.gz

Deploy your model

- Deployed endpoint is available and in service now.
- Create a function which invokes the endpoint and returns the result of prediction.
- Upload a sample image into S3 Bucket, We have used a 50 euro image and uploaded in into the folder test_images on S3.
- we are getting our image from S3 and saving them into a variable called Euro50.
- We are calling our classify_deployed function to predict our image, and the result is shown in below image.

+ → C O A	## Migrative central Leonois ava anatoricon/Leon	nalan bene hegion vez- sedital. 16 indpoints		Ŷ	0 ± =
😬 III Services 🔍 Search for Se	rvices, Reitures, Nogs, docs, and more	(Att+S)	• •	🔿 Frankfurt •	Halannadiaasiidnay •
Git repositories	We want to hear from you. Take our short 5-10 m	sinute survey and help us make SageMaker better for customers like	you.		Take Survey 🗙
 Processing 	Amazon SageMaker) Endpoints				
+ Training	Endpoints	C	Update endpoint	Actions w	Smalle undpoint
♥ Inference Compilation jobs	Q banh indparts				(1) 0
Marketplace model packages	Name v	AIN	Creation time	• Status v	Last updated
Models Endpoint configurations Endpoints	O image-classification 2022-06-30-23-23-29-838	amawsagenakmau centul- 1540180785457.exdpoint/inage-classification- 2022-06-30-23-23-29-838	Jun 30, 2022 23:23 UTC	Øintervice	Am 30, 2022 25:27 UTC
Batch brandform jobe					
Edge Manager Get started					
Edge packaging jobs Edge device fleets					

```
[66]: # If we want to check out model' prediction through this notebook instance
      # we will create a function which will call our endpoint here and return the model prediction
      # we will have to upload some test images to our s3 bucket which this method will use
       import json
       import numpy as np
       import os
       def classify_deployed(file_name, classes):
          payload = None
           with open(file_name, 'rb') as f:
              payload = f.read()
              payload = bytearray(payload)
          result = deployed_endpoint.predict(payload, initial_args={'ContentType': 'image/jpeg'})
           #result = json.loads(deployed_endpoint.predict(payload))
           #result = deployed endpoint.predict(payload)
           #best_prob_index = np.argmax(result)
           #return (classes[best_prob_index], result[best_prob_index])
           resultarray = (result.decode('UTF-8')[1:len(result)-1]).split(",")
           for i in range(len(classes)):
              print(classes[i] + ":" + str(resultarray[i]))
           return result
       print("Function created")
```

carees an earcest bejeet at an isreeboar

Function created

• The EURO-50 image has the highest prediction value, our model is 99.9% confident that the provided image in of 50 Euro note.



100%[========>] 21.99K --.-KB/s

in 0.001s

[68]: # but for this we have to make our image public !wget -O test.jpg https://cloud-computing-dataset.s3.eu-central-1.amazonaws.com/CC-Dataset/Test_images/50-euro.JPG Euro50 = "test.ipg" # test image from IPython.display import Image Image(Euro50) --2022-06-30 23:39:43-- https://cloud-computing-dataset.s3.eu-central-1.amazonaws.com/CC-Dataset/Test_images/50-euro.JPG Resolving cloud-computing-dataset.s3.eu-central-1.amazonaws.com (cloud-computing-dataset.s3.eu-central-1.amazonaws.com)... 52.219.47.144 Connecting to cloud-computing-dataset.s3.eu-central-1.amazonaws.com (cloud-computing-dataset.s3.eu-central-1.amazonaws.com) 52.219.47.144 :4 43... connected. HTTP request sent, awaiting response... 200 OK Length: 22513 (22K) [image/jpeg] Saving to: 'test.jpg' test.jpg 100%[========>] 21.99K --.-KB/s in 0.001s 2022-06-30 23:39:44 (21.8 MB/s) - 'test.jpg' saved [22513/22513]

[70]: object_categories = [
 "EURO-10",
 "EURO-5",
 "EURO-5",
 "EURO-5",
 "EURO-50",
 "E

2022-06-30 23:39:44 (21.8 MB/s) - 'test.jpg' saved [22513/22513]

43... connected.

test.jpg

Saving to: 'test.jpg'

HTTP request sent, awaiting response... 200 OK

Length: 22513 (22K) [image/jpeg]

Alert...

• **Important:** Important is to stop the notebook after you have done creating the model, otherwise AWS will keep charging you for the time the notebook is in service.

Note	ebook instances		C	Actions 🔺 C	reate notebook instance
Q	Search notebook instances			Open Jupyter Open JupyterLab	< 1 > @
	Name 🗢	Instance	Creation time	Stop	Actions
0	Cloud-Computing-Project-Image- Recognition-NB-Instance	ml.t2.medium	May 29, 2022 13:35 UTC	Start Update settings	Open Jupyter Open JupyterLab
				Add/Edit tags	
				Add/Edit tags Delete	

Flow Diagram



Web Application

The web application is created for demo purposes of the model deployed on AWS Sagemaker. It is created using React libraries in addition to using amplify library which streamlines the connection of the web application with the AWS cloud setup. For setting up amplify in the project we need to install amplify and with in the project directory execute the command: amplify init Now we need to setup an API endpoint on the AWS API Gateway. We use the command: amplify add api We further follow the steps in the process executed by the command to make a POST Rest api. When the api end point is created locally we push the setup on the AWS using: amplify push The web application consists of 2 main components: Image Capture: This component uses the camera of the device to capture the image to be identified

<mark>is</mark> Image	copture.js u ×
src > coi	nponents > 🥦 ImageCapture.js > 😋 ImageCapture > 🕁 render
26	// Moeez comments: This component opens and captures or screenshots
27	<pre>// the image in the camera session which is then used for image recognition</pre>
28	return (
29	<div></div>
30	<pre><div></div></pre>
31	<webcam< td=""></webcam<>
32	audio={false}
33	height={IMAGE_HEIGHT}
34	width={IMAGE_WIDTH}
35	<pre>ref={this.setRef}</pre>
36	<pre>screenshotFormat="image/jpeg"</pre>
37	<pre>screenshotWidth={IMAGE_WIDTH}</pre>
38	<pre>videoConstraints={videoConstraints}</pre>
39	/>
40	
41	
42	<pre><form.button onclick="{this.handleCapture}">Classify</form.button></pre>
43	
44	
45	}
46	}

IS ClassifiedImage.js ∪ ×

ic / components / 13 classificultilage.js / .	ssifiedImage.js >	classified	> JS	ponents	rc > com
---	-------------------	------------	------	---------	----------

30	
37	// Moeez comments: This component creates a React UI Card which is consists of a
38	<pre>// Header, Meta and a Description element.</pre>
39	render() {
40	return (
41	<card '224px'}}="" style="{{width:"></card>
42	<image src="{this.props.imageSrc}"/>
43	<card.content></card.content>
44	<card.header></card.header>
45	<pre>{ this.state.bestLabel ? this.state.bestLabel : "Loading" }</pre>
46	
47	<card.meta></card.meta>
48	<pre>{ this.state.bestLabelScore ? this.state.bestLabelScore : "" }</pre>
49	
50	<card.description></card.description>
51	<pre><accordion defaultactiveindex="{-1}" panels="{this.accordionPanels()}"></accordion></pre>
52	
53	
54	
55	
56	}
57	}
58	

App.js	
src > 25 Ap	p.js > 😋 App > 🔗 render
140	1/* Morez comments: This is form to take inputs which are used to send the
148	post request to and the labels are used to present the result of specific
149	categories. */}
150	<form></form>
151	<form.group widths="equal"></form.group>
152	<pre><form.input <="" label="SM Endpoint Name" placeholder="Please enter Sagemaker endpoint name" pre=""></form.input></pre>
153	<pre>name='endpointName' onChange={this.changeHandler} value={this.state.endpointName} /></pre>
154	
155	<form.input endpointregion'="" label="SM Endpoint Region" onchange="{this.changeHandler}" placeholder="Please enter sagemaker endpoint region</td></tr><tr><td>156</td><td><pre>name=" value="{this.state.endpointRegion}"></form.input>
157	
158	<pre>Form.Input label='Labels' placeholder='Please enter space delimited list of labels'</pre>
159	<pre>name='labels' onChange={this.changeHandler} value={this.state.labels} /></pre>
160	
161	
162	<form.group widths="equal"></form.group>
163	<imagecapture oncapture="{this.classity}/"></imagecapture>
164	
165	

JS App.js M 🗙	
src > /s App.js ;	😫 App > 🕲 render
187	
188	{/* Moeez comments: Finally the results are grouped together and displayed */}
189	<cardgroup></cardgroup>
190	{ this.state.imageSources.map((src, index) =>
191	<classifiedimage classifier="{this.classifier}" imagesrc="{src}" key='{"img"+index}'></classifiedimage> > }
192	

²⁵ App.js M ×

SEC > 15	App.js > 😋 App > 🔗 render
83	
84	// Moeez comments: This function calls the AWS API Gateway endpointName
85	<pre>// and returns the categories with their predictions as the result</pre>
86	<pre>classifier = async (imageSrc) => {</pre>
87	<pre>const base64Image = new Buffer(imageSrc.replace(/^data:image\/\w+;base64,/, ""), 'base64')</pre>
88	<pre>const { predictions } = await API.post(</pre>
89	aws exports.aws cloud logic custom[0].name,
90	'/classify',
91	
92	body: {
93	base64Image,
94	endpointName: this.state.endpointName,
95	endpointRegion: this.state.endpointRegion,
96).
97	3
98);
99	<pre>const topProbIndex = argMax(predictions);</pre>
100	<pre>const labels = [].concat(this.state.labels.split(' '));</pre>
101	labels.sort();
102	return {
103	labels: labels, predictions, topProbIndex: topProbIndex
104	
105	}
106	

35 App.js	5 M ×
SIC > 15	App.js > 😋 App > 🚱 render
83	
84	// Moeez comments: This function calls the AWS API Gateway endpointName
85	<pre>// and returns the categories with their predictions as the result</pre>
86	<pre>classifier = async (imageSrc) => {</pre>
87	<pre>const base64Image = new Buffer(imageSrc.replace(/^data:image\/\w+;base64,/, ""), 'base64')</pre>
88	<pre>const { predictions } = await API.post(</pre>
89	aws_exports.aws_cloud_logic_custom[0].name,
90	'/classify',
91	Constant and the second
92	body: {
93	base64Image,
94	endpointName: this.state.endpointName,
95	endpointRegion: this.state.endpointRegion,
96	Y.
97	}
98);
99	<pre>const topProbIndex = argMax(predictions);</pre>
100	<pre>const labels = [].concat(this.state.labels.split(' '));</pre>
101	labels.sort();
102	return {
103	labels: labels, predictions, topProbIndex: topProbIndex
104	}
105	}
106	



← → C O localhost3000

Clear Images 5 10 20 50 EURO-5 EURO-10 EURO-20 EURO-50 EURO-5 EURO-5 0.9995838615562439 0.9951698184013367 0.883262038230896 0.9913842678070068 0.9993664622306824 0.999525558329773 Show Score Details Show Score Details



EURO-5 0.9973311424255371

Show Score Details



EURO-5 0.9894456267356873

Show Score Details



EURO-5 0.094642972946167

Show Score Details



EURO-5 0.8661041855812073

Show Score Details



EURO-5 0.9968541264533997

Show Score Details



EURO-5 0.9250454306602478

Show Score Details



