# Adapted from An Introduction to SageMaker Random Cut Forests

***Unsupervised anomaly detection on timeseries data a Random Cut Forest algorithm.***

---

# Introduction

---

Amazon SageMaker Random Cut Forest (RCF) is an algorithm designed to detect anomalous data points within a dataset. Examples of when anomalies are important to detect include when website activity uncharactersitically spikes, when temperature data diverges from a periodic behavior, or when changes to public transit ridership reflect the occurrence of a special event.

The main goals of this notebook are,

- to learn how to obtain, transform, and store data for use in Amazon SageMaker;
- to create an AWS SageMaker training job on a data set to produce an RCF model,
- use the RCF model to perform inference with an Amazon SageMaker endpoint.

The following are ***not*** goals of this notebook:

- deeply understand the RCF model,
- understand how the Amazon SageMaker RCF algorithm works.

If you would like to know more please check out the SageMaker RCF Documentation.

# Setup

---

*This notebook was tested in Amazon SageMaker Studio on a ml.t3.medium instance with Python 3 (Data Science) kernel.*

Our first step is to setup our AWS credentials so that AWS SageMaker can store and access training data and model artifacts. We also need some data to inspect and to train upon.

# Select Amazon S3 Bucket

We first need to specify the locations where the original data is stored and where we will store our training data and trained model artifacts. ***This is the only cell of this notebook that you will need to edit.*** In particular, we need the following data:

- `bucket` - An S3 bucket accessible by this account.
- `prefix` - The location in the bucket where this notebook's input and output data will be stored. (The default value is sufficient.)
- `downloaded_data_bucket` - An S3 bucket where data is downloaded from this link and stored.
- `downloaded_data_prefix` - The location in the bucket where the data is stored.

In [176]:

```
import boto3
import botocore
import sagemaker
import sys
from time import gmtime, strftime


bucket = sagemaker.Session().default_bucket()
prefix = "sagemaker/rcf-benchmarks"
execution_role = sagemaker.get_execution_role()
region = boto3.Session().region_name

# S3 bucket where the original data is downloaded and stored.
downloaded_data_bucket = f"s3-datalake-iot-curated"
downloaded_data_prefix = "testdata/current/modbus-conveyer-current-databrew-job_24Sep2022_1664007637220"
#downloaded_data_prefix = "datastore/output_1665222858/"


#def check_bucket_permission(bucket):
    # check if the bucket exists
#    permission = False
#    try:
#        boto3.Session().client("s3").head_bucket(Bucket=bucket)
#    except botocore.exceptions.ParamValidationError as e:
#        print(
#            "Hey! You either forgot to specify your S3 bucket"
#            " or you gave your bucket an invalid name!"
#        )
#    except botocore.exceptions.ClientError as e:
#        if e.response["Error"]["Code"] == "403":
```

```
#            print(f"Hey! You don't have permission to access the bucket,
{bucket}.")
#        elif e.response["Error"]["Code"] == "404":
#            print(f"Hey! Your bucket, {bucket}, doesn't exist!")
#        else:
#            raise
#    else:
#        permission = True
#    return permission


#if check_bucket_permission(bucket):
print(f"Training input/output will be stored in: s3://{bucket}/{prefix}")
#if check_bucket_permission(downloaded_data_bucket):
print(f"Downloaded training data will be read from
s3://{downloaded_data_bucket}/{downloaded_data_prefix}")
Training input/output will be stored in: s3://sagemaker-us-west-2-
305723022616/sagemaker/rcf-benchmarks
Downloaded training data will be read from s3://s3-datalake-iot-
curated/testdata/current/modbus-conveyer-current-databrew-
job_24Sep2022_1664007637220
```

# Obtain and Inspect Example Data

```
%%time

import pandas as pd
import urllib.request

data_filename = "modbus-conveyer-current-databrew-
job_24Sep2022_1664007637220_part00000.csv.gz"
s3 = boto3.client("s3")
s3.download_file(downloaded_data_bucket,
f"{downloaded_data_prefix}/{data_filename}", data_filename)
current_data = pd.read_csv(data_filename, delimiter=",")
CPU times: user 42.8 ms, sys: 4.02 ms, total: 46.8 ms
Wall time: 220 ms
```

We already know what our data looks like, but a quick look to be sure we have loaded the right data could be good here. We can use the head() function to view the first 5 rows of data just to check. Our data should look like the following:

```
current_data.head()
```

|   | current | datetime | id |
|---|---|---:|---:|
| **0** | 17 | 2022-09-23T07:22:04Z | 824502 |
| **1** | 14 | 2022-09-23T11:12:15Z | 824502 |
| **2** | 13 | 2022-09-23T09:56:12Z | 824502 |
| **3** | 12 | 2022-09-23T13:20:22Z | 824502 |
| **4** | 16 | 2022-09-24T04:35:06Z | 824502 |

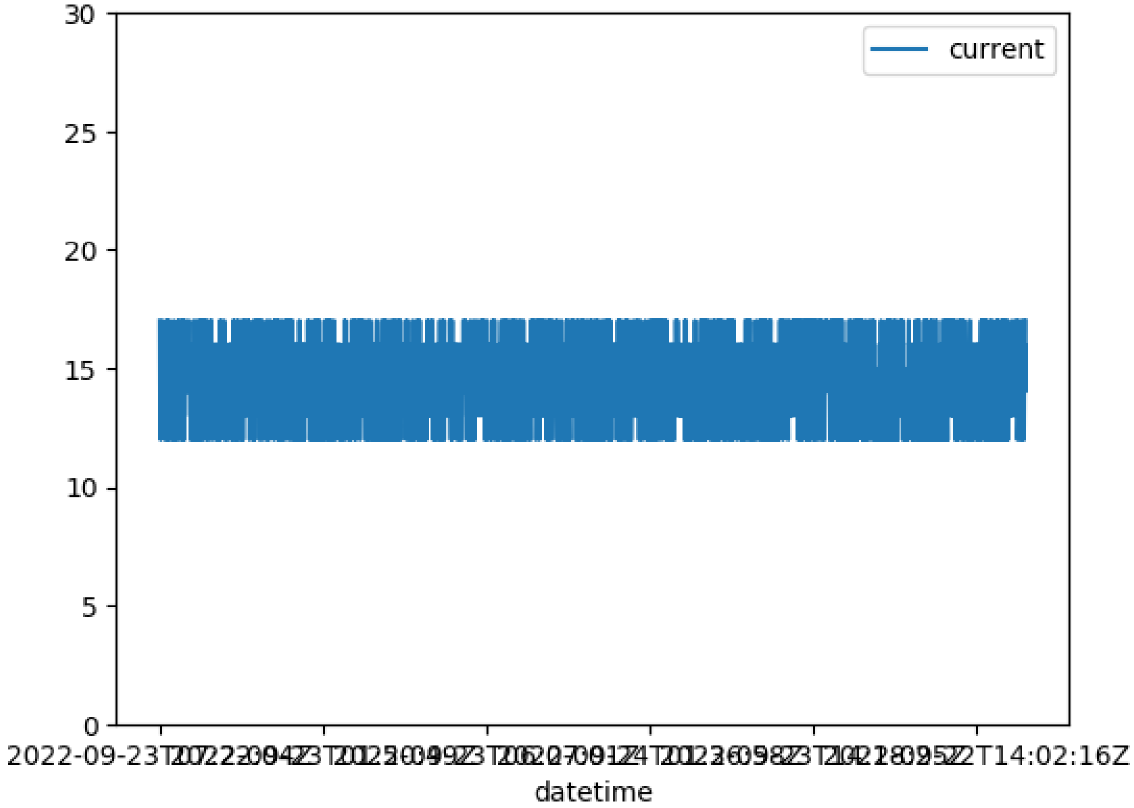
let's look at the data elements

```
current_data.info
```

```
<bound method DataFrame.info of       current              datetime      id
0          17  2022-09-23T07:22:04Z  824502
1          14  2022-09-23T11:12:15Z  824502
2          13  2022-09-23T09:56:12Z  824502
3          12  2022-09-23T13:20:22Z  824502
4          16  2022-09-24T04:35:06Z  824502
...       ...                   ...     ...
2646       14  2022-09-22T10:53:08Z  824502
2647       16  2022-09-22T11:22:09Z  824502
2648       14  2022-09-22T11:12:09Z  824502
2649       14  2022-09-22T11:42:10Z  824502
2650       17  2022-09-22T11:20:09Z  824502

[2651 rows x 3 columns]>
```

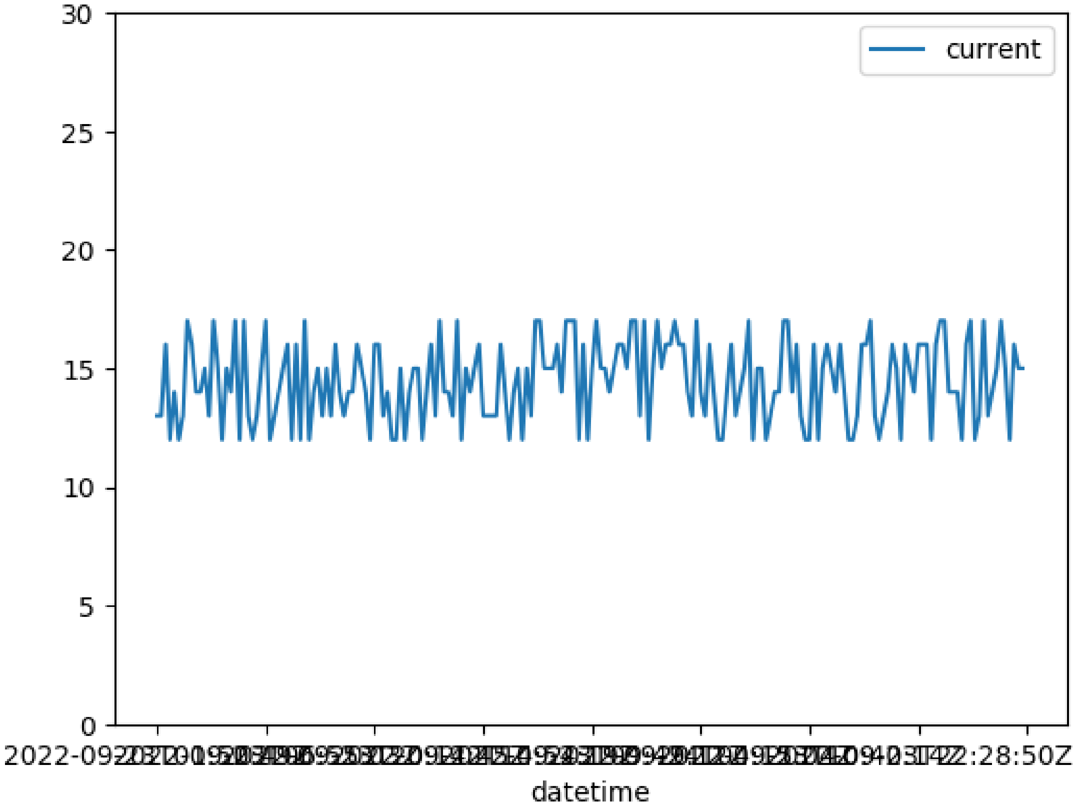
Human beings are visual creatures so let's take a look at a plot of the data.

```
%matplotlib inline

import matplotlib
import matplotlib.pyplot as plt

matplotlib.rcParams["figure.dpi"] = 100

current_data.plot("datetime", "current")
plt.ylim(0, 30)
(0, 30)
```



Human beings are also extraordinarily good at perceiving patterns. Note, for example, that something uncharacteristic occurs at around datapoint number 6000. Additionally, as we might expect with taxi ridership, the passenger count appears more or less periodic. Let's zoom in to not only examine this anomaly but also to get a better picture of what the "normal" data looks like.

```
current_data[500:700].plot("datetime", "current")
plt.ylim(0, 30)
(0, 30)
```



# Training

Next, we configure a SageMaker training job to train the Random Cut Forest (RCF) algorithm on the current data.

## Hyperparameters

Particular to a SageMaker RCF training job are the following hyperparameters:

- **`num_samples_per_tree`** - the number randomly sampled data points sent to each tree. As a general rule, `1/num_samples_per_tree` should approximate the the estimated ratio of anomalies to normal points in the dataset.
- **`num_trees`** - the number of trees to create in the forest. Each tree learns a separate model from different samples of data. The full forest model uses the mean predicted anomaly score from each constituent tree.
- **`feature_dim`** - the dimension of each data point.

In addition to these RCF model hyperparameters, we provide additional parameters defining things like the EC2 instance type on which training will run, the S3 bucket containing the data, and the AWS access role. Note that,

- Recommended instance type: `ml.m4`,`ml.c4`, or `ml.c5`
- Current limitations:
    - The RCF algorithm does not take advantage of GPU hardware.

```
from sagemaker import RandomCutForest

session = sagemaker.Session()

# specify general training job information
rcf = RandomCutForest(
    role=execution_role,
    instance_count=1,
    instance_type="ml.m4.xlarge",
    data_location=f"s3://{bucket}/{prefix}/",
    output_path=f"s3://{bucket}/{prefix}/output",
    num_samples_per_tree=512,
    num_trees=50,
    #base_job_name = f"modbus-current-randomforest-{strftime('%Y-%m-%d-%H-%M-
%S', gmtime())}"
    base_job_name = f"modbus-current-randomforest-{strftime('%Y-%m-%d-%H-%M',
gmtime())}"
)

# automatically upload the training data to S3 and run the training job
rcf.fit(rcf.record_set(current_data.current.to_numpy().reshape(-1, 1)))
Defaulting to the only supported framework/algorithm version: 1. Ignoring
framework/algorithm version: 1.
2022-09-26 06:33:56 Starting - Starting the training job...
2022-09-26 06:34:24 Starting - Preparing the instances for
trainingProfilerReport-1664174036: InProgress
.........
2022-09-26 06:35:48 Downloading - Downloading input data...
2022-09-26 06:36:08 Training - Downloading the training image...............
2022-09-26 06:38:50 Training - Training image download completed. Training in
progress..Docker entrypoint called with argument(s): train
Running default environment configuration script
[09/26/2022 06:39:02 INFO 140703060981568] Reading default configuration from
/opt/amazon/lib/python3.7/site-packages/algorithm/resources/default-
conf.json: {'num_samples_per_tree': 256, 'num_trees': 100, 'force_dense':
'true', 'eval_metrics': ['accuracy', 'precision_recall_fscore'], 'epochs': 1,
'mini_batch_size': 1000, '_log_level': 'info', '_kvstore': 'dist_async',
'_num_kv_servers': 'auto', '_num_gpus': 'auto', '_tuning_objective_metric':
'', '_ftp_port': 8999}
[09/26/2022 06:39:02 INFO 140703060981568] Merging with provided
configuration from /opt/ml/input/config/hyperparameters.json: {'feature_dim':
'1', 'mini_batch_size': '1000', 'num_samples_per_tree': '512', 'num_trees':
'50'}
[09/26/2022 06:39:02 INFO 140703060981568] Final configuration:
{'num_samples_per_tree': '512', 'num_trees': '50', 'force_dense': 'true',
'eval_metrics': ['accuracy', 'precision_recall_fscore'], 'epochs': 1,
'mini_batch_size': '1000', '_log_level': 'info', '_kvstore': 'dist_async',
'_num_kv_servers': 'auto', '_num_gpus': 'auto', '_tuning_objective_metric':
'', '_ftp_port': 8999, 'feature_dim': '1'}
[09/26/2022 06:39:02 WARNING 140703060981568] Loggers have already been
setup.
[09/26/2022 06:39:02 INFO 140703060981568] Launching parameter server for
role scheduler
[09/26/2022 06:39:02 INFO 140703060981568] {'ENVROOT': '/opt/amazon',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION': 'cpp', 'HOSTNAME': 'ip-10-0-89-
145.us-west-2.compute.internal', 'TRAINING_JOB_NAME': 'modbus-current-
randomforest-2022-09-26--2022-09-26-06-33-56-690', 'NVIDIA_REQUIRE_CUDA':
'cuda>=9.0', 'TRAINING_JOB_ARN': 'arn:aws:sagemaker:us-west-
2:305723022616:training-job/modbus-current-randomforest-2022-09-26--2022-09-
26-06-33-56-690', 'AWS_CONTAINER_CREDENTIALS_RELATIVE_URI':
'/v2/credentials/proxy-
d3162d092105bdf76d013b7d8888921aa72e6b530249e8e1f942ef03b18808be-customer',
'CANONICAL_ENVROOT': '/opt/amazon', 'PYTHONUNBUFFERED': 'TRUE',
'NVIDIA_VISIBLE_DEVICES': 'all', 'LD_LIBRARY_PATH':
'/opt/amazon/lib/python3.7/site-
packages/cv2/../../../../lib:/usr/local/nvidia/lib64:/opt/amazon/lib',
'MXNET_KVSTORE_BIGARRAY_BOUND': '400000000', 'NVIDIA_DRIVER_CAPABILITIES':
'compute,utility', 'PATH':
'/opt/amazon/bin:/usr/local/nvidia/bin:/usr/local/sbin:/usr/local/bin:/usr/sb
in:/usr/bin:/sbin:/bin', 'PWD': '/', 'LANG': 'en_US.utf8', 'AWS_REGION': 'us-
west-2', 'SAGEMAKER_METRICS_DIRECTORY': '/opt/ml/output/metrics/sagemaker',
'HOME': '/root', 'SHLVL': '1',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION_VERSION': '2', 'OMP_NUM_THREADS':
'2', 'DMLC_INTERFACE': 'eth0', 'SAGEMAKER_HTTP_PORT': '8080',
'SAGEMAKER_DATA_PATH': '/opt/ml', 'KMP_DUPLICATE_LIB_OK': 'True',
'KMP_INIT_AT_FORK': 'FALSE'}
[09/26/2022 06:39:02 INFO 140703060981568] envs={'ENVROOT': '/opt/amazon',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION': 'cpp', 'HOSTNAME': 'ip-10-0-89-
145.us-west-2.compute.internal', 'TRAINING_JOB_NAME': 'modbus-current-
randomforest-2022-09-26--2022-09-26-06-33-56-690', 'NVIDIA_REQUIRE_CUDA':
'cuda>=9.0', 'TRAINING_JOB_ARN': 'arn:aws:sagemaker:us-west-
2:305723022616:training-job/modbus-current-randomforest-2022-09-26--2022-09-
```

```
26-06-33-56-690', 'AWS_CONTAINER_CREDENTIALS_RELATIVE_URI':
'/v2/credentials/proxy-
d3162d092105bdf76d013b7d8888921aa72e6b530249e8e1f942ef03b18808be-customer',
'CANONICAL_ENVROOT': '/opt/amazon', 'PYTHONUNBUFFERED': 'TRUE',
'NVIDIA_VISIBLE_DEVICES': 'all', 'LD_LIBRARY_PATH':
'/opt/amazon/lib/python3.7/site-
packages/cv2/../../../../lib:/usr/local/nvidia/lib64:/opt/amazon/lib',
'MXNET_KVSTORE_BIGARRAY_BOUND': '400000000', 'NVIDIA_DRIVER_CAPABILITIES':
'compute,utility', 'PATH':
'/opt/amazon/bin:/usr/local/nvidia/bin:/usr/local/sbin:/usr/local/bin:/usr/sb
in:/usr/bin:/sbin:/bin', 'PWD': '/', 'LANG': 'en_US.utf8', 'AWS_REGION': 'us-
west-2', 'SAGEMAKER_METRICS_DIRECTORY': '/opt/ml/output/metrics/sagemaker',
'HOME': '/root', 'SHLVL': '1',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION_VERSION': '2', 'OMP_NUM_THREADS':
'2', 'DMLC_INTERFACE': 'eth0', 'SAGEMAKER_HTTP_PORT': '8080',
'SAGEMAKER_DATA_PATH': '/opt/ml', 'KMP_DUPLICATE_LIB_OK': 'True',
'KMP_INIT_AT_FORK': 'FALSE', 'DMLC_ROLE': 'scheduler', 'DMLC_PS_ROOT_URI':
'10.0.89.145', 'DMLC_PS_ROOT_PORT': '9000', 'DMLC_NUM_SERVER': '1',
'DMLC_NUM_WORKER': '1'}
[09/26/2022 06:39:02 INFO 140703060981568] Launching parameter server for
role server
[09/26/2022 06:39:02 INFO 140703060981568] {'ENVROOT': '/opt/amazon',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION': 'cpp', 'HOSTNAME': 'ip-10-0-89-
145.us-west-2.compute.internal', 'TRAINING_JOB_NAME': 'modbus-current-
randomforest-2022-09-26--2022-09-26-06-33-56-690', 'NVIDIA_REQUIRE_CUDA':
'cuda>=9.0', 'TRAINING_JOB_ARN': 'arn:aws:sagemaker:us-west-
2:305723022616:training-job/modbus-current-randomforest-2022-09-26--2022-09-
26-06-33-56-690', 'AWS_CONTAINER_CREDENTIALS_RELATIVE_URI':
'/v2/credentials/proxy-
d3162d092105bdf76d013b7d8888921aa72e6b530249e8e1f942ef03b18808be-customer',
'CANONICAL_ENVROOT': '/opt/amazon', 'PYTHONUNBUFFERED': 'TRUE',
'NVIDIA_VISIBLE_DEVICES': 'all', 'LD_LIBRARY_PATH':
'/opt/amazon/lib/python3.7/site-
packages/cv2/../../../../lib:/usr/local/nvidia/lib64:/opt/amazon/lib',
'MXNET_KVSTORE_BIGARRAY_BOUND': '400000000', 'NVIDIA_DRIVER_CAPABILITIES':
'compute,utility', 'PATH':
'/opt/amazon/bin:/usr/local/nvidia/bin:/usr/local/sbin:/usr/local/bin:/usr/sb
in:/usr/bin:/sbin:/bin', 'PWD': '/', 'LANG': 'en_US.utf8', 'AWS_REGION': 'us-
west-2', 'SAGEMAKER_METRICS_DIRECTORY': '/opt/ml/output/metrics/sagemaker',
'HOME': '/root', 'SHLVL': '1',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION_VERSION': '2', 'OMP_NUM_THREADS':
'2', 'DMLC_INTERFACE': 'eth0', 'SAGEMAKER_HTTP_PORT': '8080',
'SAGEMAKER_DATA_PATH': '/opt/ml', 'KMP_DUPLICATE_LIB_OK': 'True',
'KMP_INIT_AT_FORK': 'FALSE'}
```

```
[09/26/2022 06:39:02 INFO 140703060981568] envs={'ENVROOT': '/opt/amazon',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION': 'cpp', 'HOSTNAME': 'ip-10-0-89-
145.us-west-2.compute.internal', 'TRAINING_JOB_NAME': 'modbus-current-
randomforest-2022-09-26--2022-09-26-06-33-56-690', 'NVIDIA_REQUIRE_CUDA':
'cuda>=9.0', 'TRAINING_JOB_ARN': 'arn:aws:sagemaker:us-west-
2:305723022616:training-job/modbus-current-randomforest-2022-09-26--2022-09-
26-06-33-56-690', 'AWS_CONTAINER_CREDENTIALS_RELATIVE_URI':
'/v2/credentials/proxy-
d3162d092105bdf76d013b7d8888921aa72e6b530249e8e1f942ef03b18808be-customer',
'CANONICAL_ENVROOT': '/opt/amazon', 'PYTHONUNBUFFERED': 'TRUE',
'NVIDIA_VISIBLE_DEVICES': 'all', 'LD_LIBRARY_PATH':
'/opt/amazon/lib/python3.7/site-
packages/cv2/../../../../lib:/usr/local/nvidia/lib64:/opt/amazon/lib',
'MXNET_KVSTORE_BIGARRAY_BOUND': '400000000', 'NVIDIA_DRIVER_CAPABILITIES':
'compute,utility', 'PATH':
'/opt/amazon/bin:/usr/local/nvidia/bin:/usr/local/sbin:/usr/local/bin:/usr/sb
in:/usr/bin:/sbin:/bin', 'PWD': '/', 'LANG': 'en_US.utf8', 'AWS_REGION': 'us-
west-2', 'SAGEMAKER_METRICS_DIRECTORY': '/opt/ml/output/metrics/sagemaker',
'HOME': '/root', 'SHLVL': '1',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION_VERSION': '2', 'OMP_NUM_THREADS':
'2', 'DMLC_INTERFACE': 'eth0', 'SAGEMAKER_HTTP_PORT': '8080',
'SAGEMAKER_DATA_PATH': '/opt/ml', 'KMP_DUPLICATE_LIB_OK': 'True',
'KMP_INIT_AT_FORK': 'FALSE', 'DMLC_ROLE': 'server', 'DMLC_PS_ROOT_URI':
'10.0.89.145', 'DMLC_PS_ROOT_PORT': '9000', 'DMLC_NUM_SERVER': '1',
'DMLC_NUM_WORKER': '1'}
[09/26/2022 06:39:02 INFO 140703060981568] Environment: {'ENVROOT':
'/opt/amazon', 'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION': 'cpp', 'HOSTNAME':
'ip-10-0-89-145.us-west-2.compute.internal', 'TRAINING_JOB_NAME': 'modbus-
current-randomforest-2022-09-26--2022-09-26-06-33-56-690',
'NVIDIA_REQUIRE_CUDA': 'cuda>=9.0', 'TRAINING_JOB_ARN':
'arn:aws:sagemaker:us-west-2:305723022616:training-job/modbus-current-
randomforest-2022-09-26--2022-09-26-06-33-56-690',
'AWS_CONTAINER_CREDENTIALS_RELATIVE_URI': '/v2/credentials/proxy-
d3162d092105bdf76d013b7d8888921aa72e6b530249e8e1f942ef03b18808be-customer',
'CANONICAL_ENVROOT': '/opt/amazon', 'PYTHONUNBUFFERED': 'TRUE',
'NVIDIA_VISIBLE_DEVICES': 'all', 'LD_LIBRARY_PATH':
'/opt/amazon/lib/python3.7/site-
packages/cv2/../../../../lib:/usr/local/nvidia/lib64:/opt/amazon/lib',
'MXNET_KVSTORE_BIGARRAY_BOUND': '400000000', 'NVIDIA_DRIVER_CAPABILITIES':
'compute,utility', 'PATH':
'/opt/amazon/bin:/usr/local/nvidia/bin:/usr/local/sbin:/usr/local/bin:/usr/sb
in:/usr/bin:/sbin:/bin', 'PWD': '/', 'LANG': 'en_US.utf8', 'AWS_REGION': 'us-
west-2', 'SAGEMAKER_METRICS_DIRECTORY': '/opt/ml/output/metrics/sagemaker',
'HOME': '/root', 'SHLVL': '1',
'PROTOCOL_BUFFERS_PYTHON_IMPLEMENTATION_VERSION': '2', 'OMP_NUM_THREADS':
```

```
'2', 'DMLC_INTERFACE': 'eth0', 'SAGEMAKER_HTTP_PORT': '8080',
'SAGEMAKER_DATA_PATH': '/opt/ml', 'KMP_DUPLICATE_LIB_OK': 'True',
'KMP_INIT_AT_FORK': 'FALSE', 'DMLC_ROLE': 'worker', 'DMLC_PS_ROOT_URI':
'10.0.89.145', 'DMLC_PS_ROOT_PORT': '9000', 'DMLC_NUM_SERVER': '1',
'DMLC_NUM_WORKER': '1'}
Process 36 is a shell:scheduler.
Process 48 is a shell:server.
Process 1 is a worker.
[09/26/2022 06:39:02 INFO 140703060981568] Using default worker.
[09/26/2022 06:39:02 INFO 140703060981568] Loaded iterator creator
application/x-recordio-protobuf for content type ('application/x-recordio-
protobuf', '1.0')
[09/26/2022 06:39:02 INFO 140703060981568] Checkpoint loading and saving are
disabled.
[09/26/2022 06:39:02 INFO 140703060981568] Verifying hyperparamemters...
[09/26/2022 06:39:02 INFO 140703060981568] Hyperparameters are correct.
[09/26/2022 06:39:02 INFO 140703060981568] Validating that feature_dim agrees
with dimensions in training data...
[09/26/2022 06:39:02 INFO 140703060981568] feature_dim is correct.
[09/26/2022 06:39:02 INFO 140703060981568] Validating memory limits...
[09/26/2022 06:39:02 INFO 140703060981568] Available memory in bytes:
15615365120
[09/26/2022 06:39:02 INFO 140703060981568] Estimated sample size in bytes:
204800
[09/26/2022 06:39:02 INFO 140703060981568] Estimated memory needed to build
the forest in bytes: 1024000
[09/26/2022 06:39:02 INFO 140703060981568] Memory limits validated.
[09/26/2022 06:39:02 INFO 140703060981568] Starting cluster sharing
facilities...
[09/26/2022 06:39:02 INFO 140700494583552] concurrency model: async
[09/26/2022 06:39:02 INFO 140703060981568] Create Store: dist_async
[09/26/2022 06:39:02 INFO 140700494583552] masquerade (NAT) address: None
[09/26/2022 06:39:02 INFO 140700494583552] passive ports: None
[09/26/2022 06:39:02 INFO 140700494583552] >>> starting FTP server on
0.0.0.0:8999, pid=1 <<<
[09/26/2022 06:39:04 INFO 140703060981568] Cluster sharing facilities
started.
[09/26/2022 06:39:04 INFO 140703060981568] Verifying all workers are
accessible...
[09/26/2022 06:39:04 INFO 140703060981568] All workers accessible.
[09/26/2022 06:39:04 INFO 140703060981568] Initializing Sampler...
[09/26/2022 06:39:04 INFO 140703060981568] Sampler correctly initialized.
#metrics {"StartTime": 1664174342.2113159, "EndTime": 1664174344.2464464,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
```

```
"training"}, "Metrics": {"initialize.time": {"sum": 2020.9953784942627,
"count": 1, "min": 2020.9953784942627, "max": 2020.9953784942627}}}
#metrics {"StartTime": 1664174344.2466617, "EndTime": 1664174344.2467191,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
"training", "Meta": "init_train_data_iter"}, "Metrics": {"Total Records
Seen": {"sum": 0.0, "count": 1, "min": 0, "max": 0}, "Total Batches Seen":
{"sum": 0.0, "count": 1, "min": 0, "max": 0}, "Max Records Seen Between
Resets": {"sum": 0.0, "count": 1, "min": 0, "max": 0}, "Max Batches Seen
Between Resets": {"sum": 0.0, "count": 1, "min": 0, "max": 0}, "Reset Count":
{"sum": 0.0, "count": 1, "min": 0, "max": 0}, "Number of Records Since Last
Reset": {"sum": 0.0, "count": 1, "min": 0, "max": 0}, "Number of Batches
Since Last Reset": {"sum": 0.0, "count": 1, "min": 0, "max": 0}}}
[2022-09-26 06:39:04.247] [tensorio] [info] epoch_stats={"data_pipeline":
"/opt/ml/input/data/train", "epoch": 0, "duration": 2035, "num_examples": 1,
"num_bytes": 28000}
[09/26/2022 06:39:04 INFO 140703060981568] Sampling training data...
[2022-09-26 06:39:04.277] [tensorio] [info] epoch_stats={"data_pipeline":
"/opt/ml/input/data/train", "epoch": 1, "duration": 30, "num_examples": 3,
"num_bytes": 74228}
[09/26/2022 06:39:04 INFO 140703060981568] Sampling training data completed.
#metrics {"StartTime": 1664174344.2465985, "EndTime": 1664174344.281782,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
"training"}, "Metrics": {"epochs": {"sum": 1.0, "count": 1, "min": 1, "max":
1}, "update.time": {"sum": 34.612417221069336, "count": 1, "min":
34.612417221069336, "max": 34.612417221069336}}}
[09/26/2022 06:39:04 INFO 140703060981568] Early stop condition met. Stopping
training.
[09/26/2022 06:39:04 INFO 140703060981568] #progress_metric: host=algo-1,
completed 100 % epochs
#metrics {"StartTime": 1664174344.247132, "EndTime": 1664174344.2821517,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
"training", "epoch": 0, "Meta": "training_data_iter"}, "Metrics": {"Total
Records Seen": {"sum": 2651.0, "count": 1, "min": 2651, "max": 2651}, "Total
Batches Seen": {"sum": 3.0, "count": 1, "min": 3, "max": 3}, "Max Records
Seen Between Resets": {"sum": 2651.0, "count": 1, "min": 2651, "max": 2651},
"Max Batches Seen Between Resets": {"sum": 3.0, "count": 1, "min": 3, "max":
3}, "Reset Count": {"sum": 1.0, "count": 1, "min": 1, "max": 1}, "Number of
Records Since Last Reset": {"sum": 2651.0, "count": 1, "min": 2651, "max":
2651}, "Number of Batches Since Last Reset": {"sum": 3.0, "count": 1, "min":
3, "max": 3}}}
[09/26/2022 06:39:04 INFO 140703060981568] #throughput_metric: host=algo-1,
train throughput=75407.24499844019 records/second
[09/26/2022 06:39:04 INFO 140703060981568] Master node: building Random Cut
Forest...
[09/26/2022 06:39:04 INFO 140703060981568] Gathering samples...
```

```
[09/26/2022 06:39:04 INFO 140703060981568] 2651 samples gathered
[09/26/2022 06:39:04 INFO 140703060981568] Building Random Cut Forest...
[09/26/2022 06:39:04 INFO 140703060981568] Random Cut Forest built:
ForestInfo{num_trees: 50, num_samples_in_forest: 2650, num_samples_per_tree:
53, sample_dim: 1, shingle_size: 1, trees_num_nodes: [11, 11, 11, 11, 11, 11,
11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11,
11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11,
11, 11, 11, 11, 11, 11, ], trees_depth: [6, 6, 5, 5, 5, 5, 6, 4, 6, 6, 5, 6,
6, 6, 5, 5, 6, 6, 5, 6, 6, 6, 6, 5, 5, 6, 5, 6, 6, 5, 6, 6, 6, 6, 6, 5, 6, 6,
6, 6, 6, 6, 6, 4, 5, 5, 5, 6, 6, 5, ], max_num_nodes: 11, min_num_nodes: 11,
avg_num_nodes: 11, max_tree_depth: 6, min_tree_depth: 4, avg_tree_depth: 5,
mem_size: 57648}
#metrics {"StartTime": 1664174344.2818837, "EndTime": 1664174344.2844095,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
"training"}, "Metrics": {"fit_model.time": {"sum": 0.8094310760498047,
"count": 1, "min": 0.8094310760498047, "max": 0.8094310760498047},
"model.bytes": {"sum": 57648.0, "count": 1, "min": 57648, "max": 57648},
"finalize.time": {"sum": 1.9054412841796875, "count": 1, "min":
1.9054412841796875, "max": 1.9054412841796875}}}
[09/26/2022 06:39:04 INFO 140703060981568] Master node: Serializing the
RandomCutForest model
#metrics {"StartTime": 1664174344.2844946, "EndTime": 1664174344.2860587,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
"training"}, "Metrics": {"serialize_model.time": {"sum": 1.5196800231933594,
"count": 1, "min": 1.5196800231933594, "max": 1.5196800231933594}}}
[09/26/2022 06:39:04 INFO 140703060981568] Test data is not provided.
#metrics {"StartTime": 1664174344.2861342, "EndTime": 1664174344.2862887,
"Dimensions": {"Algorithm": "RandomCutForest", "Host": "algo-1", "Operation":
"training"}, "Metrics": {"setuptime": {"sum": 59.601783752441406, "count": 1,
"min": 59.601783752441406, "max": 59.601783752441406}, "totaltime": {"sum":
2159.339666366577, "count": 1, "min": 2159.339666366577, "max":
2159.339666366577}}}

2022-09-26 06:39:29 Uploading - Uploading generated training model
2022-09-26 06:39:29 Completed - Training job completed
Training seconds: 227
Billable seconds: 227
```

If you see the message

```
      ===== Job Complete =====
```

at the bottom of the output logs then that means training successfully completed and the output RCF model was stored in the specified output path. You can also view information about and the status of a training job using the AWS SageMaker console. Just click on the "Jobs" tab and select training job matching the training job name, below:

```
print(f"Training job name: {rcf.latest_training_job.job_name}")
```

```
Training job name: modbus-current-randomforest-2022-09-26--2022-09-26-06-33-
56-690
```

# Inference

A trained Random Cut Forest model does nothing on its own. We now want to use the model we computed to perform inference on data. In this case, it means computing anomaly scores from input time series data points.

We create an inference endpoint using the SageMaker Python SDK `deploy()` function from the job we defined above. We specify the instance type where inference is computed as well as an initial number of instances to spin up. We recommend using the `ml.c5` instance type as it provides the fastest inference time at the lowest cost.

```
#endpoint_name = f"current-rcfodbus-current-randomforest-{strftime('%Y-%m-%d-
%H-%M', gmtime())}"
endpoint_name = f"current-rcf-{strftime('%Y-%m-%d-%H-%M', gmtime())}"

rcf_inference = rcf.deploy(initial_instance_count=1,
instance_type="ml.m4.xlarge", endpoint_name = endpoint_name)
```

```
Defaulting to the only supported framework/algorithm version: 1. Ignoring
framework/algorithm version: 1.
---------!
```

Congratulations! You now have a functioning SageMaker RCF inference endpoint. You can confirm the endpoint configuration and status by navigating to the "Endpoints" tab in the AWS SageMaker console and selecting the endpoint matching the endpoint name, below:

```
print(f"Endpoint name: {rcf_inference.endpoint}")
```

```
The endpoint attribute has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Endpoint name: current-rcf-2022-09-26-06-56
```

## Data Serialization/Deserialization

We can pass data in a variety of formats to our inference endpoint. In this example we will demonstrate passing CSV-formatted data. Other available formats are JSON-formatted and RecordIO Protobuf. We make use of the SageMaker Python SDK utilities `csv_serializer` and `json_deserializer` when configuring the inference endpoint.

```
from sagemaker.serializers import CSVSerializer
from sagemaker.deserializers import JSONDeserializer
```

```
rcf_inference.serializer = CSVSerializer()
rcf_inference.deserializer = JSONDeserializer()
```

Let's pass the training dataset, in CSV format, to the inference endpoint so we can automatically detect the anomalies we saw with our eyes in the plots, above. Note that the serializer and deserializer will automatically take care of the datatype conversion from Numpy NDArrays.

For starters, let's only pass in the first six datapoints so we can see what the output looks like.

```
current_data_numpy = current_data.current.to_numpy().reshape(-1, 1)
print(current_data_numpy[:6])
[[24]
 [14]
 [13]
 [ 8]
 [16]
 [14]]

current_data_numpy[current_data_numpy == 17] = 24
current_data_numpy[current_data_numpy == 12] = 8
print(current_data_numpy[:6])
[[24]
 [14]
 [13]
 [ 8]
 [16]
 [14]]

list=([50])
array = np.asarray(list);
print(type(array))
print(array)


results = rcf_inference.predict(
    array, initial_args={"ContentType": "text/csv", "Accept":
"application/json"}
)

import pprint
pp = pprint.PrettyPrinter(indent=4)
pp.pprint(results)
<class 'numpy.ndarray'>
[50]
{'scores': [{'score': 4.5821495021}]}
```

```
#print(current_data_numpy[:6])
results = rcf_inference.predict(
    current_data_numpy[:1], initial_args={"ContentType": "text/csv",
"Accept": "application/json"}
)

import pprint
pp = pprint.PrettyPrinter(indent=4)
pp.pprint(results)
{'scores': [{'score': 4.451536592}]}
```

# Computing Anomaly Scores

Now, let's compute and plot the anomaly scores from the entire taxi dataset.

```
results = rcf_inference.predict(current_data_numpy)
scores = [datum["score"] for datum in results["scores"]]

# add scores to data frame and print first few values
current_data["score"] = pd.Series(scores, index=current_data.index)
current_data.head()
```

| | current | datetime | id | score |
|---|---|---|---|---|
| **0** | 24 | 2022-09-23T07:22:04Z | 824502 | 4.451537 |
| **1** | 14 | 2022-09-23T11:12:15Z | 824502 | 0.908935 |
| **2** | 13 | 2022-09-23T09:56:12Z | 824502 | 0.961220 |
| **3** | 8 | 2022-09-23T13:20:22Z | 824502 | 3.968004 |
| **4** | 16 | 2022-09-24T04:35:06Z | 824502 | 0.945301 |

```
results = rcf_inference.predict(current_data_numpy)
scores = [datum["score"] for datum in results["scores"]]

# add scores to data frame and print first few values
current_data["score"] = pd.Series(scores, index=current_data.index)
current_data.head()
```

| | current | datetime | id | score |
|---|---|---|---|---|
| **0** | 24 | 2022-09-23T07:22:04Z | 824502 | 4.451537 |
| **1** | 14 | 2022-09-23T11:12:15Z | 824502 | 0.908935 |
| **2** | 13 | 2022-09-23T09:56:12Z | 824502 | 0.961220 |
| **3** | 8 | 2022-09-23T13:20:22Z | 824502 | 3.968004 |
| **4** | 16 | 2022-09-24T04:35:06Z | 824502 | 0.945301 |

# Stop and Delete the Endpoint

Finally, we should delete the endpoint before we close the notebook.

To do so execute the cell below. Alternately, you can navigate to the "Endpoints" tab in the SageMaker console, select the endpoint with the name stored in the variable `endpoint_name`, and select "Delete" from the "Actions" dropdown menu.

```
sagemaker.Session().delete_endpoint(rcf_inference.endpoint)
```

```
The endpoint attribute has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
```