SageMaker

Overview, Pricing, Data Format

Content Prepared By: Chandra Lingam, Cloud Wave LLC

Introduction to SageMaker

Fully managed Cloud based machine learning service

Build – Jupyter Notebook development environment

Train – Managed Training infrastructure

Deploy – Scalable Hosting infrastructure

AWS SageMaker - Build

Managed Jupyter Notebook Environment

Extensive collection of popular Machine Learning Algorithms

Pre-configured to run TensorFlow and Apache MxNet

Bring-Your-Own Algorithm

AWS SageMaker - Train

Distribute training across one or many instances

Managed model training infrastructure

Scales to Petabyte datasets

Compute instances for training are automatically launched and released – Stores artifacts in S3

AWS SageMaker - Deploy

Realtime prediction

Batch Transform

Deploy for Realtime Predictions

Realtime Endpoint for interactive and low-latency use-cases

AutoScaling

- Maintain adequate capacity
- Replace unhealthy instances
- Dynamically scale-out and scale-in based on workload

Deploy for Batch Transforms

Batch Transform for non-interactive use-cases

Suitable for these scenarios:

- Inference for your entire dataset
- Don’t need a persistent real-time endpoint
- Don’t need sub-second latency performance

SageMaker manages resources for batch transform

SageMaker Instance Family

Instance Family	Strength/Uses
Standard	Balanced CPU performance
Compute Optimized	Highest CPU performance
Accelerated Computing	Graphics/GPU Compute
Inference Acceleration	Fractional GPUs (add-on)

Standard Instance Family

Balanced CPU, Memory and Network Performance

Example: T2, T3, M5

T type instances – Suitable for occasional burst. Perfect for Notebook and Development Systems

M type instances – Suitable for sustained load. Perfect for CPU intensive model training and hosting

Compute Optimized Family

Latest Generation CPUs. Higher Performance Systems

Example: C4, C5

Suitable for sustained load

Perfect for CPU intensive model training and hosting

Accelerated Computing Family

Powerful GPUs

Speed-up Algorithms optimized for GPUs

Example: P2, P3

Reduce time needed for training using GPUs. Perfect for GPU intensive model training and hosting

Inference Acceleration

Add-on Fractional GPUs

Some Algorithms are GPU intensive during Training but need only fractional GPU during Inference

Add GPU to lower cost Standard and Compute Optimized Instances

Perfect for speeding up inference using GPUs

Suggested Instance Types

Standard, Compute Optimized – Good for algorithms optimized for CPUs

Accelerated Computing – Good for algorithms optimized for GPUs

Choose a family first and then experiment various instance sizes – Simple AWS configuration change

Instance Type and Size

c5.2xlarge

Instance type generation size

ml.c5.2xlarge = Compute Optimized, 5th generation, 2xlarge (8 vCPUs, 16 GB Memory)

SageMaker Pricing Components

Instance Type and Size

Fractional GPUs

Storage

Data Transfer

AWS Region

SageMaker Free Tier

Two months free tier – starts from the first month you create a SageMaker resource

Development – 250 Hours/Month t2.medium or t3.medium

Train – 50 Hours/Month m4.xlarge or m5.xlarge

Deploy – 125 Hours/Month m4.xlarge or m5.xlarge

Development – On Demand Pricing

Instance + Fractional GPU Hourly Cost (pro-rated to the nearest second with a 1 minute minimum)

Storage – USD 0.14 per GB/Month

Data Transfer IN, OUT – USD 0.016 per GB

Training – On Demand Pricing

Instance Hourly Cost (pro-rated to the nearest second with a 1 minute minimum)

Storage – USD 0.14 per GB/Month

Instances are automatically launched and terminated

You are charged only for the duration the training job ran

Realtime Inference – On Demand Pricing

Instance + Fractional GPU Hourly Cost (pro-rated to the nearest second with a 1 minute minimum)

Storage – USD 0.14 per GB/Month

Data Transfer IN, OUT – USD 0.016 per GB

Batch Transform – On Demand Pricing

Instance + Fractional GPU Hourly Cost (pro-rated to the nearest second with a 1 minute minimum)

Storage – USD 0.14 per GB/Month

Data Transfer IN, OUT – USD 0.016 per GB

You are charged only for the duration batch transform job ran

SageMaker Data Formats

Training Data Format

- CSV
- RecordIO
- Algorithm specific formats (LibSVM, JSON, Parquet)
- Data needs to be stored in S3

Inference Format

- CSV
- JSON
- RecordIO

Data

Entire Dataset in a single file

Split across several files in a folder

Data Copy from S3 to Training Instance

File Mode:

- Training job copies entire dataset from S3 to training instance
- Space Needed: Entire data set + Final model artifacts

Pipe Mode:

- Training job streams data from S3 to training instance
- Faster start time and Better Throughput
- Space Needed: Final model artifacts

Built-in Algorithms

Variety of [Built-in Algorithms]

- XGBoost (Competition winner!)
- Linear Learners
- Factorization Machines
- K-Means
- PCA
- [And more]

ML Terminology

Training Data – Used for training a model

Validation Data – Used for verifying training accuracy and for optimizing parameters

Test Data – Used for verifying accuracy of a built-up model (last step)

Data needs to be stored in S3

Algorithms Overview

Algorithm	Description
Linear Models	+ Simple + Performs surprisingly well for a variety of problems - Single equation trying to capture interaction of all variables - Categorical data needs to be encoded using one hot encoding
Decision Tree	+ Can Handle Complex non-linear relationship + Easily handles categorical data, missing data - Prone to overfitting - Poor predictive accuracy
Ensemble Methods	+ Combines multiple simple decision trees + Addresses Decision Tree overfitting problem + Much better predictive performance - More complex

amazon web services

Must Watch Videos

Gradient Boosting Machine Learning by Trevor Hastie

Learning Decision Tree by Alexander Ihler

Ensembles (Bagging) by Alexander Ihler

Demo 1 : Create S3 Bucket for SageMaker

- Create dedicated S3 bucket for the course
  - Data Store for training models
  - Model Artifact storage
  - Bucket Name: *<prefix>-ml-sagemaker*
- Sign-in with *my_admin* account

IAM Account Sign-in Link

https://<AccountId>.signin.aws.amazon.com/console

https://<Alias>.signin.aws.amazon.com/console

Demo 2 : Launch a Notebook Instance

Launch Notebook Instance – use *my_admin* account

- Define Permissions
- Select Instance Type
- Launch Notebook instance

Use ml_user account from this point onwards!

- Starter Samples
- Storing your custom code
    - Create a folder *SageMakerCourse*

Demo 3: Data Setup

- Download the following files from resources for this lecture: *XGBoostExamples.zip*, *extract_zip_file_content.ipynb*
- Upload the files to *SageMakerCourse* folder on the notebook instance
- Open *extract_zip_file_content.ipynb* to unzip the contents
- SageMakerSteps.xlsx – Contains file naming convention used

Demo 4: Data Formats and Interacting with S3

- Create sample file in CSV, RecordIO Formats
- Upload Files to S3
- Download Files from S3

Notebook: DataFormats\data_format_exploration.ipynb

Demo 5: XGBoost Regression

- Install XGBoost on Notebook Instance
- Regression Examples
    - Linear Model
    - Quadratic Model

Notebook: LinearAndQuadraticFunctionRegression\...

Demo 6: Kaggle Bike Sharing

- When used with AWS Machine Learning, RMSLE score was around 0.9
- Let’s run the same example with XGBoost!

Notebook Folder: BikeSharingRegression

Demo 7: Kaggle Bike Sharing

- Optimization – Adding relevant features

Demo 8: Kaggle Bike Sharing

- Response variable as log1p

Demo 9: Train XGBoost Model on SageMaker

- SageMaker SDK Overview
- Prepare dataset for training, validation
- Train model
- Verify performance
- Deploy for Real-time prediction
- Query end point

Notebook: xgboost_cloud_training_template.ipynb

xgboost_cloud_prediction_template.ipynb

SageMaker Training Steps

1. Store data files in S3
2. Specify algorithm and hyper parameters
3. Configure and run the training job
4. Deploy the trained model

S3 Data Source Configuration

Attribute	Values/Purpose
S3DataDistributionType	FullyReplicated – entire dataset is replicated on each training instance ShardedByS3Key – Subset of data is replicated on each training instance. If dataset is split across multiple S3 objects, then SageMaker will distribute equal number of S3 objects to each training node.
S3DataType	ManifestFile – S3Uri points to a file that in-turn contains a list of files to be used for training S3Prefix – S3Uri points to a prefix. SageMaker uses all the objects with the specified prefix
S3Uri	Identifies a Key name prefix or a manifest file

Demo 10 – Invoke Prediction from outside

BikeSharingRegression

invoke_sagemaker_runtime_from_outside.ipynb

Pre-requisites:

1. Anaconda Python
2. Boto3 Library
3. SageMaker Library
4. ml_user_predict account as specified in house keeping lecture

Hyper Parameter Tuning

n_estimators (in XGBRegressor) is same as **num_round** (in XGBoost and SageMaker documentation)

This parameter controls number of rounds of boosting i.e. total number of trees.

Make sure you use correct parameter depending on the library. *XGBRegressor silently ignores parameters it does not understand* ☹

Hyper Parameter Tuning

[XGBoost Tuning Suggestions]

[SageMaker XGBoost Hyper Parameter Documentation]

Demo 11: XGBoost Classification

- Iris Model (categorical response, multi-class)
- Diabetes Dataset Model (binary classification)
- Mushroom Classification (categorical variables and response)
- Requires encoding categorical data to numeric for training and prediction