

Machine Learning with Containers on Amazon SageMaker

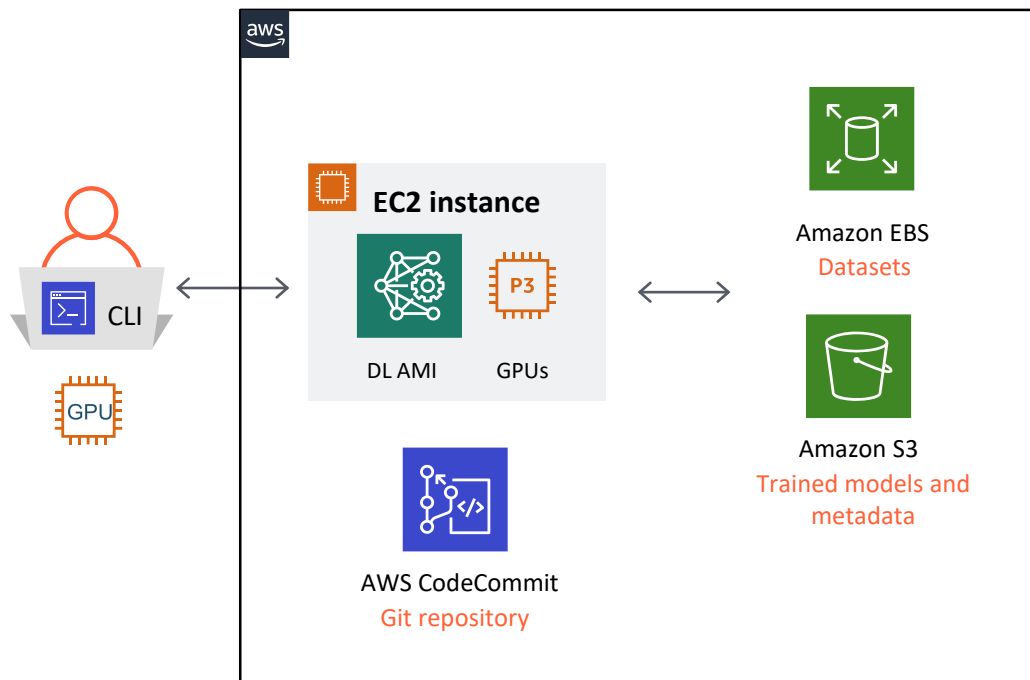
Shashank Prasanna,
Sr. Technical Evangelist, AI/ML

26th August 2019

Agenda

- Challenges with machine learning infrastructure
- Containers and machine learning
 - AWS Deep Learning Containers
- Scaling deep learning containers with Amazon SageMaker
- Demos using Deep Learning Containers and Amazon SageMaker
 1. Getting started with TensorFlow deep learning container
 2. Running hyperparameter search experiments on Amazon SageMaker
 3. Running distributed training with TensorFlow and Horovod on Amazon SageMaker
- Summary and Q&A

Machine learning setups on AWS today



- ✓ Compute (CPUs, GPUs)
- ✓ Storage
- ✓ Source control
- ✓ ML Frameworks

Challenges with existing machine learning setups

**Software
management**

**Performance
optimizations**

**Collaborative
development**

Scalability

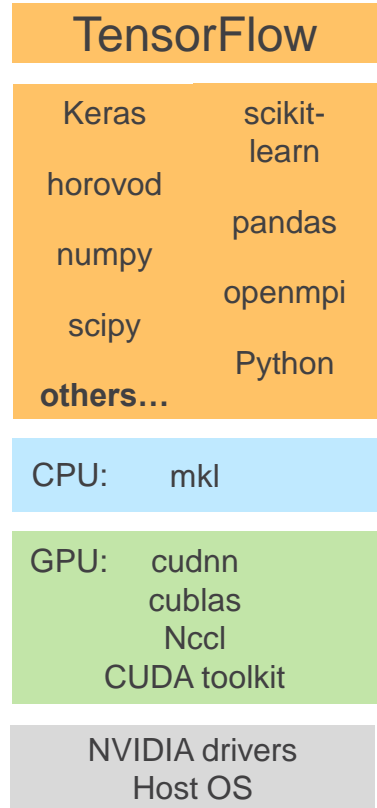
**Infrastructure
management**

...

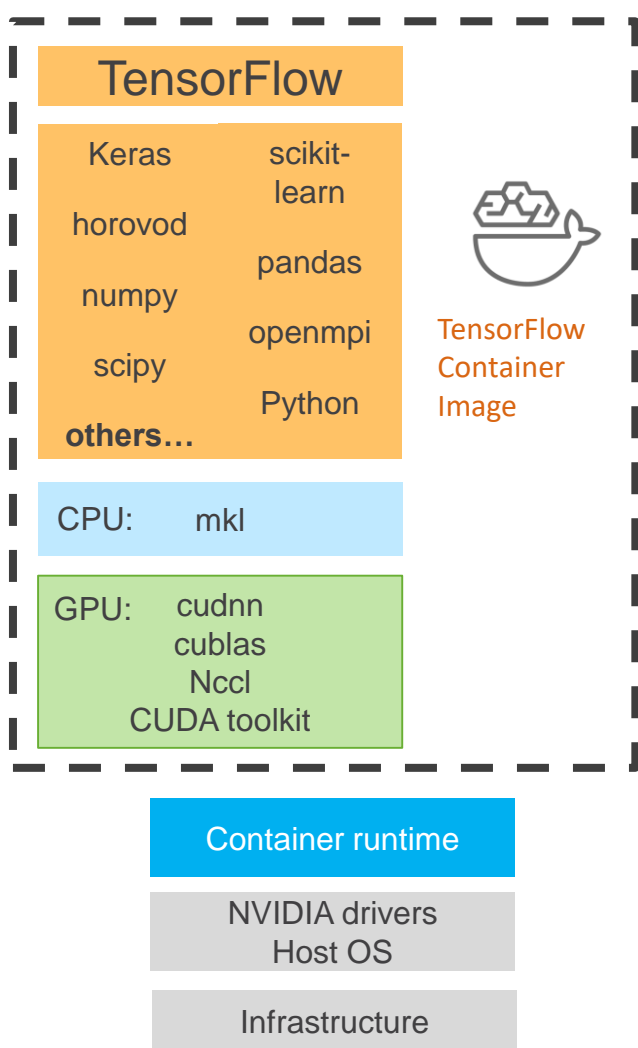
Machine learning stack is complex

- “My code requires building several dependencies from source”
- “My code isn’t taking advantage the GPU/GPUs”
 - “is cudnn, nccl installed, is it the right version?”
- “My code is running slow on CPUs”
 - “oh wait, is it taking advantage of AVX instruction set ?!?”
- “I updated my drivers and training is now slower/errors out”
- “My production cluster runs a different version of framework/linux distro”

Makes portability, collaboration, scaling workloads really really hard!



Using containers for Machine Learning workloads



ML environments that are:

- Lightweight
- Portable
- Scalable
- Consistent

Packages:

- Training code
- Dependencies
- Configurations

AWS Deep Learning Containers

- Prepackaged Docker container images fully configured and validated
- Optimized for performance with latest NVIDIA driver, CUDA libraries, and Intel libraries
- Consistent and reproducible deployment and lightweight
- Optimized for distributed machine learning
- Runs on Amazon EKS, Amazon ECS and **Amazon SageMaker**

Training (4)

GPU with py36
GPU with py27
CPU with py36
CPU with py27

Inference (4)

GPU with py36
GPU with py27
CPU with py36
CPU with py27

TensorFlow (8)

MXNet (8)

(PyTorch coming soon)



Amazon
EKS



Amazon
ECS



Amazon
SageMaker

AWS Deep Learning Containers

Deep Learning AMI Developer Guide

Documentation - This

Search

- What is the AWS Deep Learning AMI?
- Getting Started
- Launching a DLAMI
- Using a DLAMI
- Deep Learning Containers
 - Deep Learning Containers on Amazon EC2
 - Deep Learning Containers on ECS
 - Deep Learning Containers

Framework	Training/Inference	Horovod	CPU/GPU	Python Version	URL
TensorFlow 1.14.0	Training	Yes	CPU	2.7	763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.14.0-cpu-py27-ubuntu16.04
TensorFlow 1.14.0	Training	Yes	CPU	3.6	763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.14.0-cpu-py36-ubuntu16.04
TensorFlow 1.14.0	Training	Yes	GPU	2.7	763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.14.0-gpu-py27-cu100-ubuntu16.04
TensorFlow 1.14.0	Training	Yes	GPU	3.6	763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training:1.14.0-gpu-py36-cu100-ubuntu16.04
TensorFlow 1.14.0	Inference	No	CPU	2.7	763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference:1.14.0-cpu-py27-ubuntu18.04
TensorFlow 1.14.0	Inference	No	CPU	3.6	763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference:1.14.0-cpu-py36-ubuntu18.04

```
docker pull <container image>
```

Bring your own training script (BYOTS)

16 Images in ECR repositories in 15 AWS regions

<https://docs.aws.amazon.com/dlami/latest/devguide/deep-learning-containers-images.html>

Using deep learning containers at scale on Amazon SageMaker



Amazon SageMaker
Managed notebook
instance

OR




Local laptop or
desktop with
SageMaker SDK

```
from sagemaker.tensorflow import TensorFlow

tf_estimator = TensorFlow(entry_point='cifar10.py',
                          source_dir='code',
                          role=role,
                          train_instance_count=1,
                          train_instance_type='ml.p3.2xlarge',
                          framework_version='1.13',
                          py_version='py3',
                          hyperparameters=hyperparams)

tf_estimator.fit(dataset_in_s3)
```

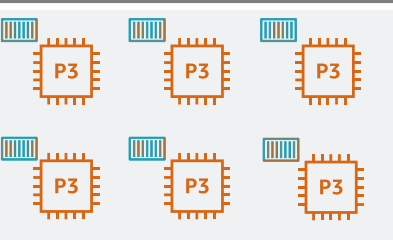


cifar10.py

```
import tensorflow as tf
import argparse
import os
from tensorflow import keras
from tensorflow.keras.layers import Input
from tensorflow.keras.models import Model
from tensorflow.keras.utils import multi
from tensorflow.keras.optimizers import

HEIGHT = 32
WIDTH = 32
DEPTH = 3
NUM_CLASSES = 10
```

Deep Learning
Container



Fully-managed
SageMaker cluster



Amazon S3

DEMO: Getting started with TensorFlow deep learning container

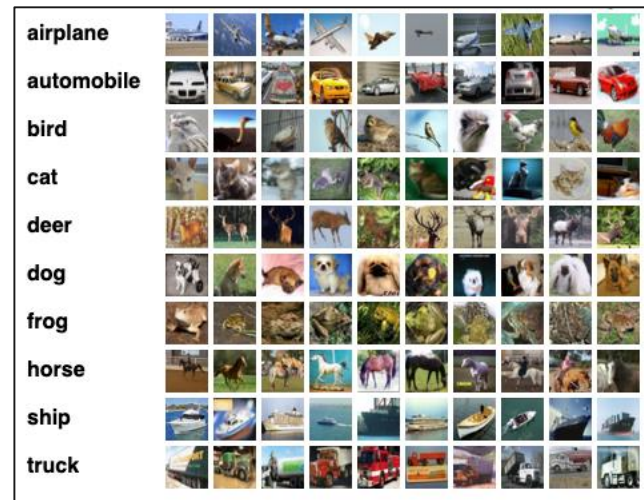
Dataset: CIFAR10

Problem: Image classification

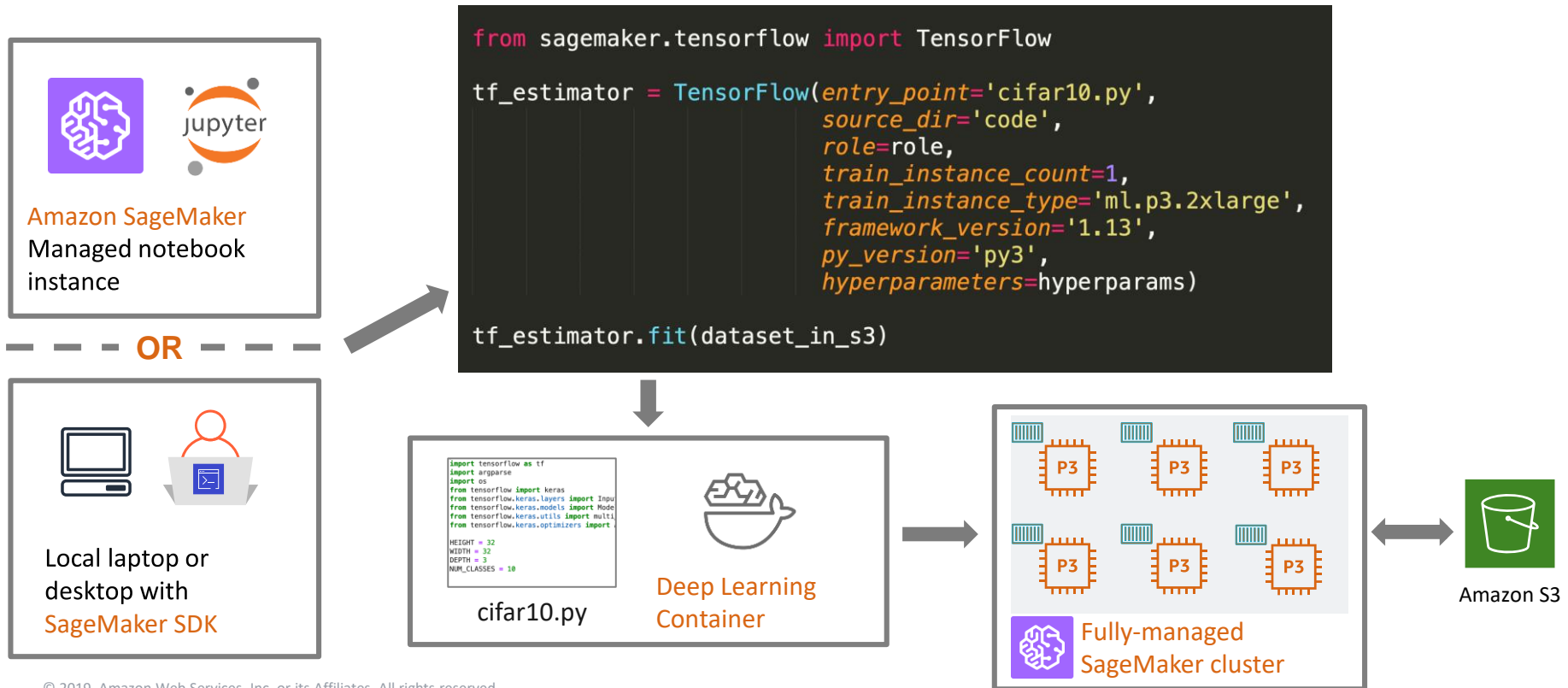
Framework: TensorFlow and Keras

Approach:

- Updating training script for Amazon SageMaker
- Test training on DL container locally
- Run training on GPU instance on SageMaker cluster



RECAP: Deep learning containers on SageMaker

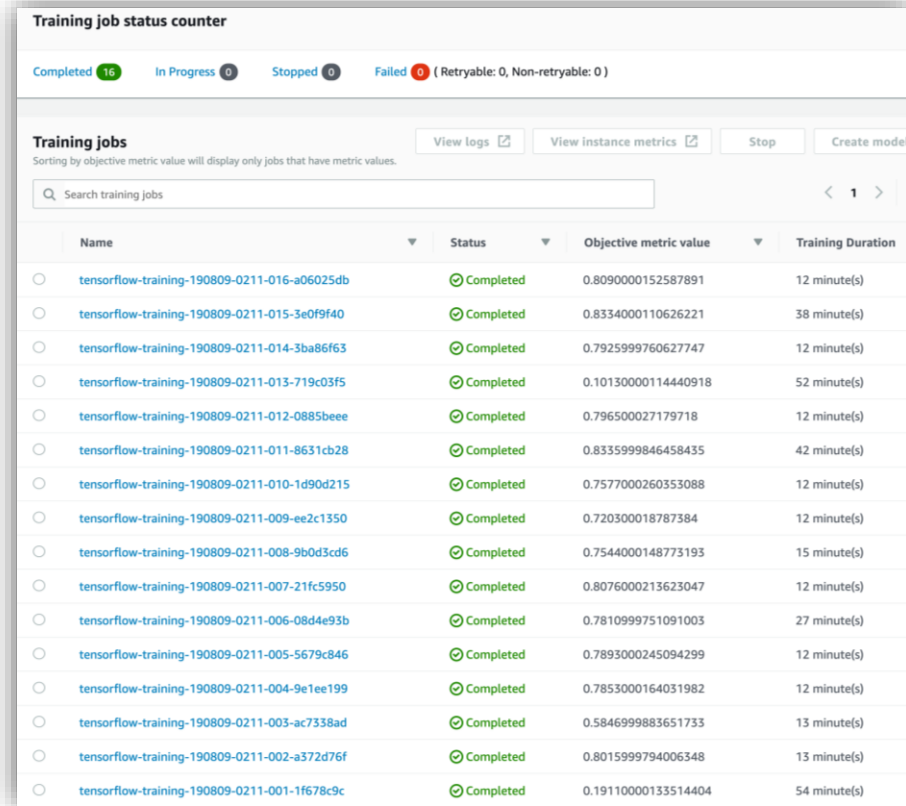


DEMO: Hyperparameter optimization

Approach:

- Specify hyperparameters
- Choose hyperparameter search strategy (Bayesian, random, custom)
- Launch a tuning job

Learning rate	Between 0.0001 and 0.1 on log scale
Batch size	32, 128, 512, 1024
Momentum	Between 0.9 and 0.99
Optimizer	SGD, Adam



Training job status counter

Completed **16** In Progress **0** Stopped **0** Failed **0** (Retriable: 0, Non-retriable: 0)

Training jobs View logs View instance metrics Stop Create model

Sorting by objective metric value will display only jobs that have metric values.

Search training jobs

Name	Status	Objective metric value	Training Duration
tensorflow-training-190809-0211-016-a06025db	Completed	0.8090000152587891	12 minute(s)
tensorflow-training-190809-0211-015-3e0f9f40	Completed	0.8334000110626221	38 minute(s)
tensorflow-training-190809-0211-014-3ba86f63	Completed	0.7925999760627747	12 minute(s)
tensorflow-training-190809-0211-013-719c03f5	Completed	0.10130000114440918	52 minute(s)
tensorflow-training-190809-0211-012-0885beee	Completed	0.796500027179718	12 minute(s)
tensorflow-training-190809-0211-011-8631cb28	Completed	0.8335999846458435	42 minute(s)
tensorflow-training-190809-0211-010-1d90d215	Completed	0.7577000260353088	12 minute(s)
tensorflow-training-190809-0211-009-ee2c1350	Completed	0.720300018787384	12 minute(s)
tensorflow-training-190809-0211-008-9b0d3cd6	Completed	0.7544000148773193	15 minute(s)
tensorflow-training-190809-0211-007-21fc5950	Completed	0.8076000213623047	12 minute(s)
tensorflow-training-190809-0211-006-08d4e93b	Completed	0.7810999751091003	27 minute(s)
tensorflow-training-190809-0211-005-5679c846	Completed	0.7893000245094299	12 minute(s)
tensorflow-training-190809-0211-004-9e1ee199	Completed	0.7853000164031982	12 minute(s)
tensorflow-training-190809-0211-003-ac7338ad	Completed	0.5846999883651733	13 minute(s)
tensorflow-training-190809-0211-002-a372d76f	Completed	0.8015999794006348	13 minute(s)
tensorflow-training-190809-0211-001-1f678c9c	Completed	0.19110000133514404	54 minute(s)

DEMO: Distributed training

Approach:

- Update training script with horovod api
- Specify number instances and number of GPUs per instance
- Launch a distributed training job

```
from sagemaker.tensorflow import TensorFlow  
  
hvd_instance_type = 'ml.p3.8xlarge'  
hvd_instance_count = 2  
hvd_processes_per_host = 4
```

8 GPU distributed training

- 2 x p3.8xlarge
- 4 GPUs/p3.8xlarge

Can I bring my own container?

The screenshot shows the AWS SageMaker documentation page. The main heading is "Use Your Own Algorithms or Models with Amazon SageMaker". The text explains that SageMaker uses Docker containers for build and runtime tasks. It states: "Amazon SageMaker makes extensive use of Docker containers for build and runtime tasks. Before using your own algorithm or model with Amazon SageMaker, you need to understand how Amazon SageMaker manages and runs them. Amazon SageMaker provides pre-built Docker images for its built-in algorithms and the deep learning frameworks supported used for training and inference. By using containers, you can train machine learning algorithms and deploy models quickly and reliably at any scale. Docker is a program that performs operating-system-level virtualization for installing, distributing, and managing software. It packages applications and their dependencies into virtual containers that provide isolation, portability, and security." It also mentions that you can put scripts, algorithms, and inference code for your machine learning models into containers. The page includes sections for "Scenarios for Running Scripts, Training Algorithms, or Deploying Models with Amazon SageMaker" and "Use Your Own Inference Code".

docs.aws.amazon.com/sagemaker/latest/dg/your-algorithms.html

The screenshot shows a GitHub repository page for "aws/aws-sagemaker". The specific file being viewed is a notebook titled "Building your own TensorFlow container". The notebook content includes an introduction: "With Amazon SageMaker, you can package your own algorithms that can then be trained and deployed in the SageMaker environment. This notebook guides you through an example using TensorFlow that shows you how to build a Docker container for SageMaker and use it for training and inference." It then lists the topics covered in the notebook, such as "Building your own TensorFlow container", "Packaging and uploading your algorithm for use with Amazon SageMaker", and "Training and testing your algorithm in Amazon SageMaker".

github.com/aws/aws-sagemaker/blob/master/advanced_functionality/tensorflow_bring_your_own/

AWS DL Containers and Amazon SageMaker

Software management	<ul style="list-style-type: none">• AWS Deep Learning Containers are lightweight, portable and fully configured and validated with latest deep learning frameworks
Performance optimizations	<ul style="list-style-type: none">• DL containers include frameworks optimized by experts to deliver the best training and inference performance on CPUs and GPUs.
Collaborative development	<ul style="list-style-type: none">• With DL containers, Amazon ECR collaborative development across different environments is easy.
Infrastructure management	<ul style="list-style-type: none">• Amazon SageMaker simplifies infrastructure management, and container orchestration for single and distributed training
Scalability	<ul style="list-style-type: none">• Amazon SageMaker lets you easily scale-out in bursts for large-scale training experiments or for long-running distributed training

Resources

Documentation

The screenshot shows the AWS documentation page for Amazon SageMaker. The main heading is "What Is Amazon SageMaker?". Below the heading, there is a paragraph explaining that SageMaker is a fully managed machine learning service that allows scientists and developers to quickly and easily build and train machine learning models. A sidebar on the left contains a navigation menu with items like "What is Amazon SageMaker?", "How it Works", "Set Up Amazon SageMaker", "Get Started", "Using Notebook Instances", "Build a Model", "Train a Model", "Deploy a Model", "Use Your Own Algorithms or Models", "Amazon SageMaker in AWS Marketplace", "Manage H1, Experiments with Amazon SageMaker Model Tracking Capability", "Use Machine Learning Frameworks with Amazon SageMaker", "Reinforcement Learning with Amazon SageMaker", "Authentication and Access Control", "Security", "Amazon SageMaker Ground Truth", "Limits and Supported Regions", "API Reference", "Document History", and "AWS Glossary".

docs.aws.amazon.com/sagemaker/latest/dg/whatis.html

Examples on GitHub

The screenshot shows the GitHub repository page for "Amazon SageMaker Examples". The repository description states: "This repository contains example notebooks that show how to apply machine learning and deep learning in Amazon SageMaker". Under the "Examples" section, there are three sub-sections: "Introduction to Ground Truth Labeling Jobs", "Introduction to Applying Machine Learning", and "Introduction to Deploying Machine Learning Models". Each sub-section includes a brief description and a list of example notebooks.

github.com/aws-labs/amazon-sagemaker-examples

AWS ML Blog

The screenshot shows the AWS Machine Learning Blog page. The main heading is "Category: SageMaker". Below the heading, there are three blog posts: "Kinect Energy uses Amazon SageMaker to Forecast energy prices with Machine Learning", "Harvesting success using Amazon SageMaker to power Bayer's digital farming unit", and "Git integration now available for the Amazon SageMaker Python SDK". Each post includes a thumbnail image, the title, author, date, and a brief description of the article.

aws.amazon.com/blogs/machine-learning/category/artificial-intelligence/sagemaker/



Thank you!

Shashank Prasanna,
Sr. Technical Evangelist, AI/ML

Questions? Happy to help:

Twitter: @shshnkp

LinkedIn: [linkedin.com/in/shashankprasanna](https://www.linkedin.com/in/shashankprasanna)