aws INNOVATE

AI/ML EDITION

24 February 2022

Train ML models quickly and cost-effectively with Amazon SageMaker

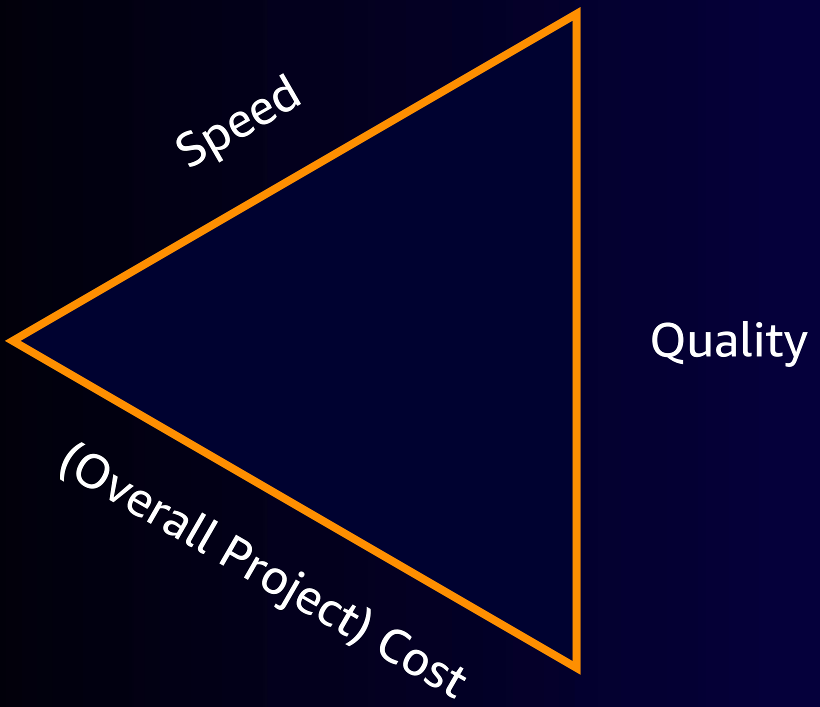
Alex Thewsey

AI/ML Specialist Solutions Architect, AWS

Agenda

1. **Why** optimize model training, and what to optimize for?
2. **How** model training works on Amazon SageMaker
3. **Tools and tips** for efficient training
4. **Demo**
5. **Recap** and resources

Priorities first: What are we optimizing for?



Accuracy
Robustness?
Explainability?
Fairness?
Privacy?
Other qualities?

Priorities first: Training is only part of the story

Inference may drive
up to

90%

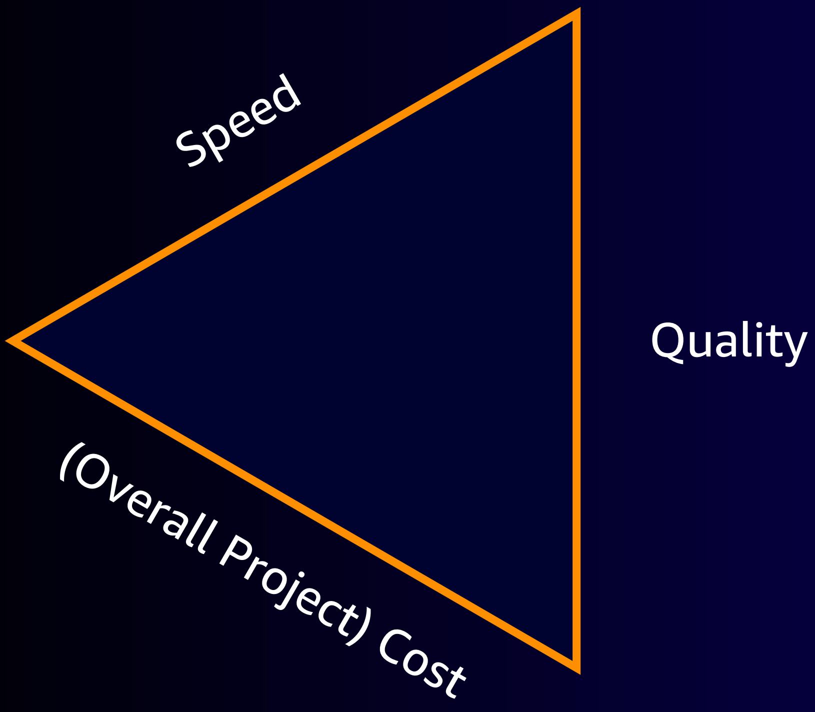
of ML project
infrastructure costs

Focus on training can be a

symptom

of organizational blockers to production deployment

Priorities first: What are we optimizing for?



Accuracy
Robustness?
Explainability?
Fairness?
Privacy?
Other qualities?

We should expect some **trade-offs,** but hopefully can find some **easy wins too**

Make use of pre-built algorithms and services

AWS AI Services

Fully-managed services for both pre-trained and bring-your-own-data AI applications

SageMaker Built-In Algos & Marketplace

17+ families of SageMaker-native algorithms, plus more via AWS Marketplace

SageMaker Autopilot

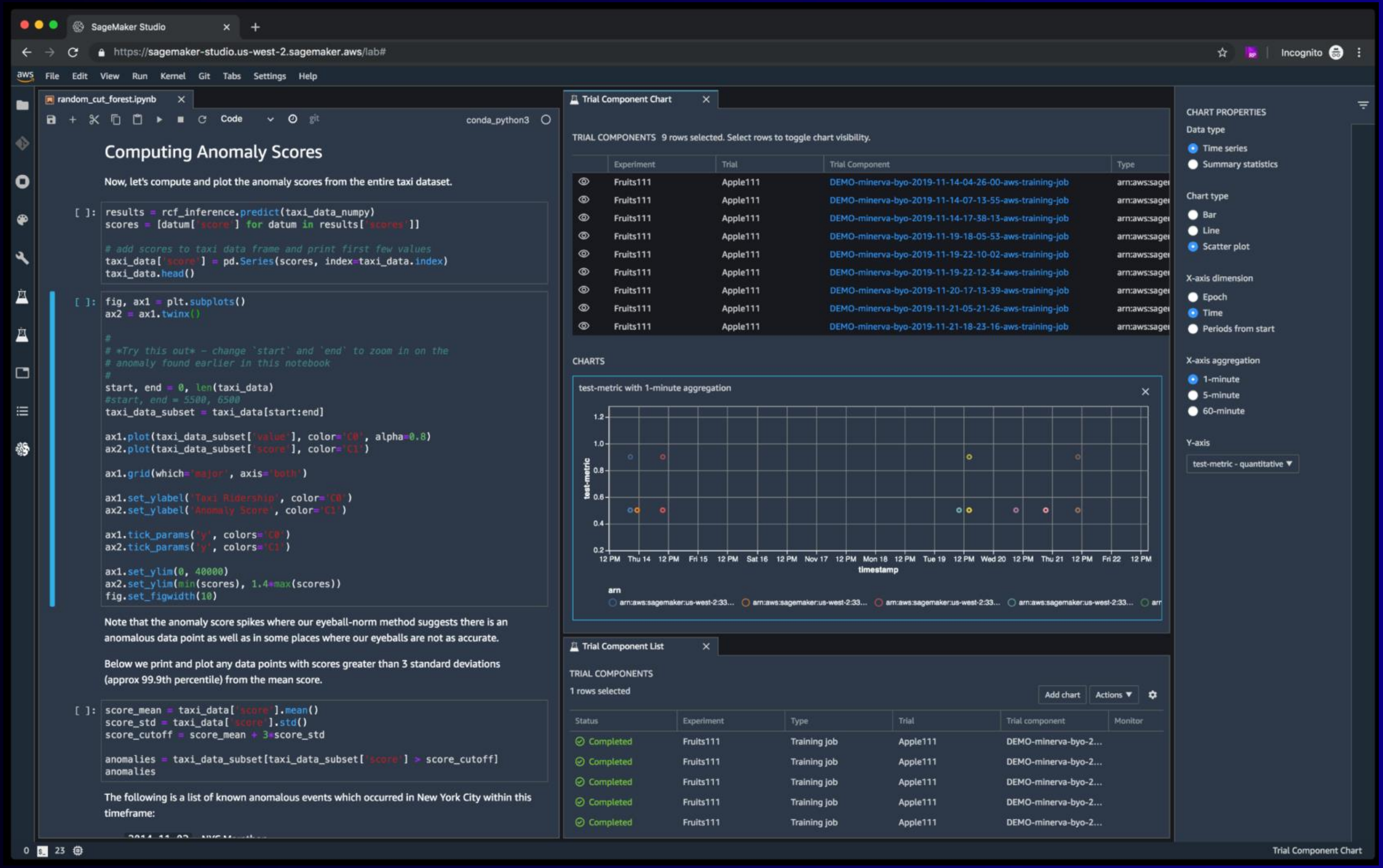
Automated but transparent, end-to-end learning for tabular data

SageMaker JumpStart

Deployable ML solution templates, additional algorithms and pre-trained public models

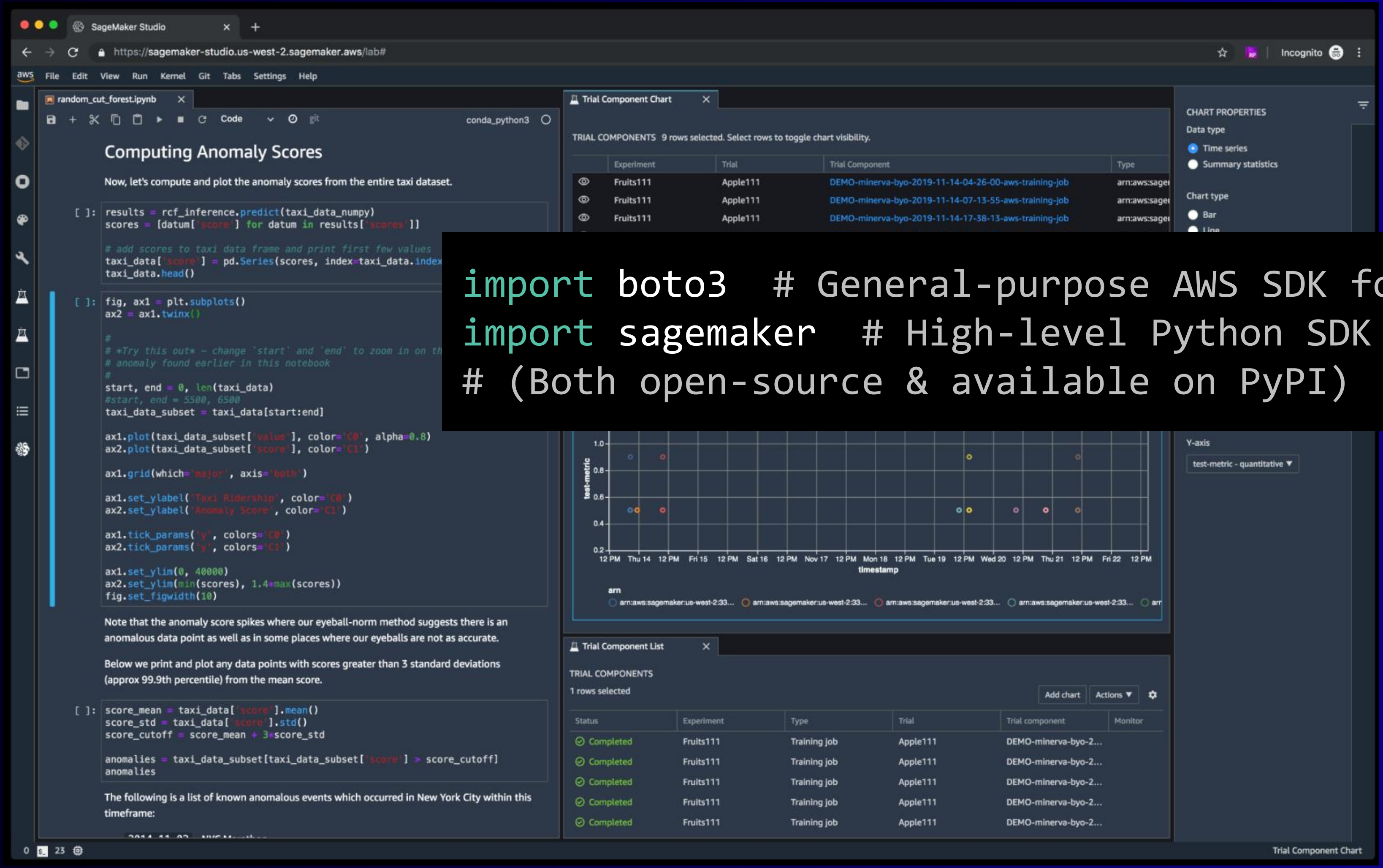
Understanding Amazon SageMaker infrastructure

"Work ***from*** the notebook, not ***on*** the notebook"



Understanding Amazon SageMaker infrastructure

"Work ***from*** the notebook, not ***on*** the notebook"



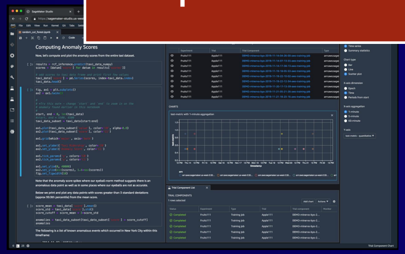

```
import boto3  # General-purpose AWS SDK for Python
import sagemaker  # High-level Python SDK for SageMaker
# (Both open-source & available on PyPI)
```

Understanding Amazon SageMaker infrastructure

“Work ***from*** the notebook, not ***on*** the notebook”

Script bundle (with optional requirements.txt?)

Development environment:
e.g. Amazon SageMaker Studio
Amazon SageMaker Notebook Instances



Understanding Amazon SageMaker infrastructure

"Work ***from*** the notebook, not ***on*** the notebook"

Amazon Elastic Container Registry (Amazon ECR)

AWS Deep Learning Container images
(for TensorFlow, PyTorch, MXNet, etc)

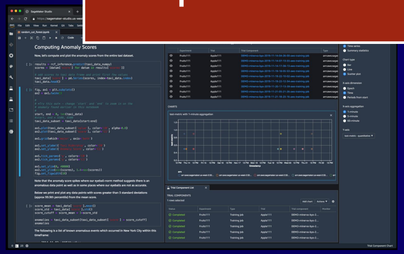
Amazon Simple Storage Service (Amazon S3)

Input B

Input A

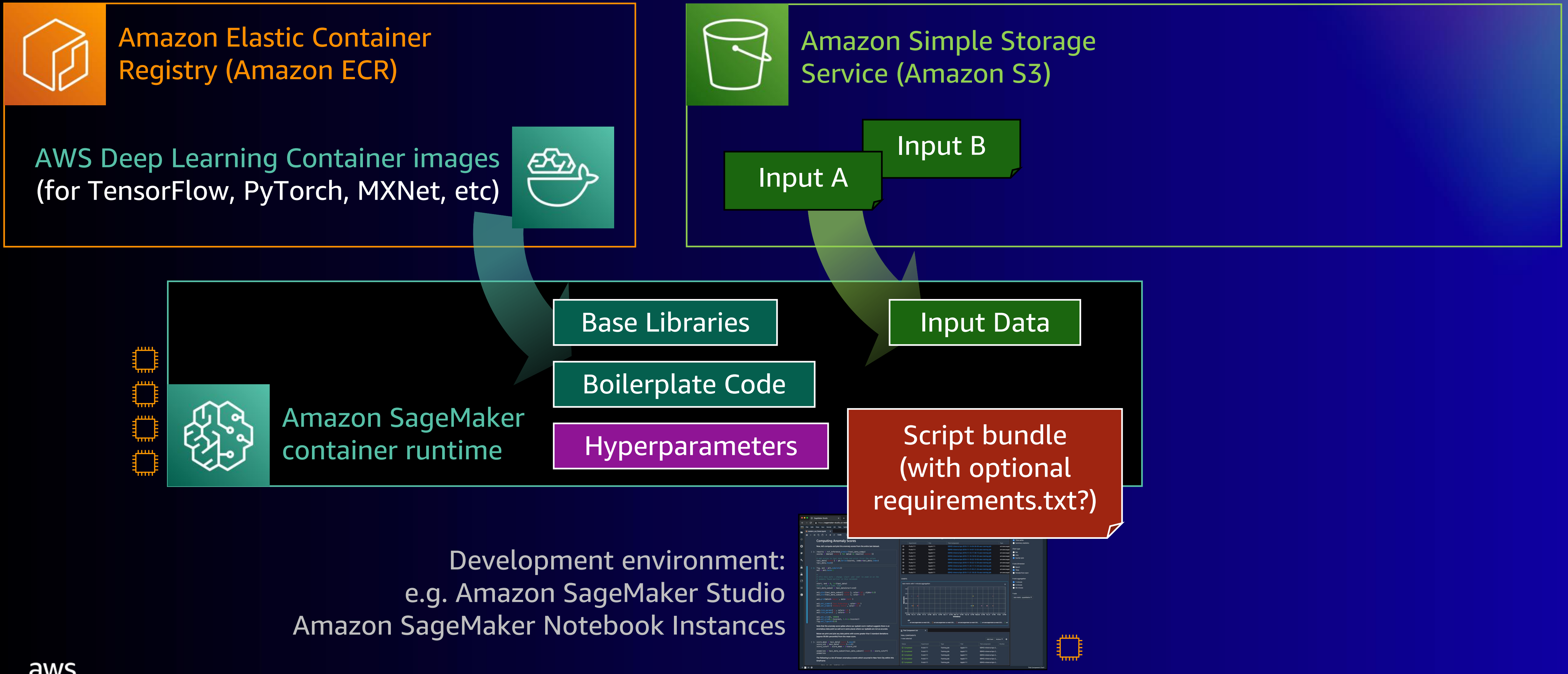
Script bundle
(with optional requirements.txt?)

Development environment:
e.g. Amazon SageMaker Studio
Amazon SageMaker Notebook Instances



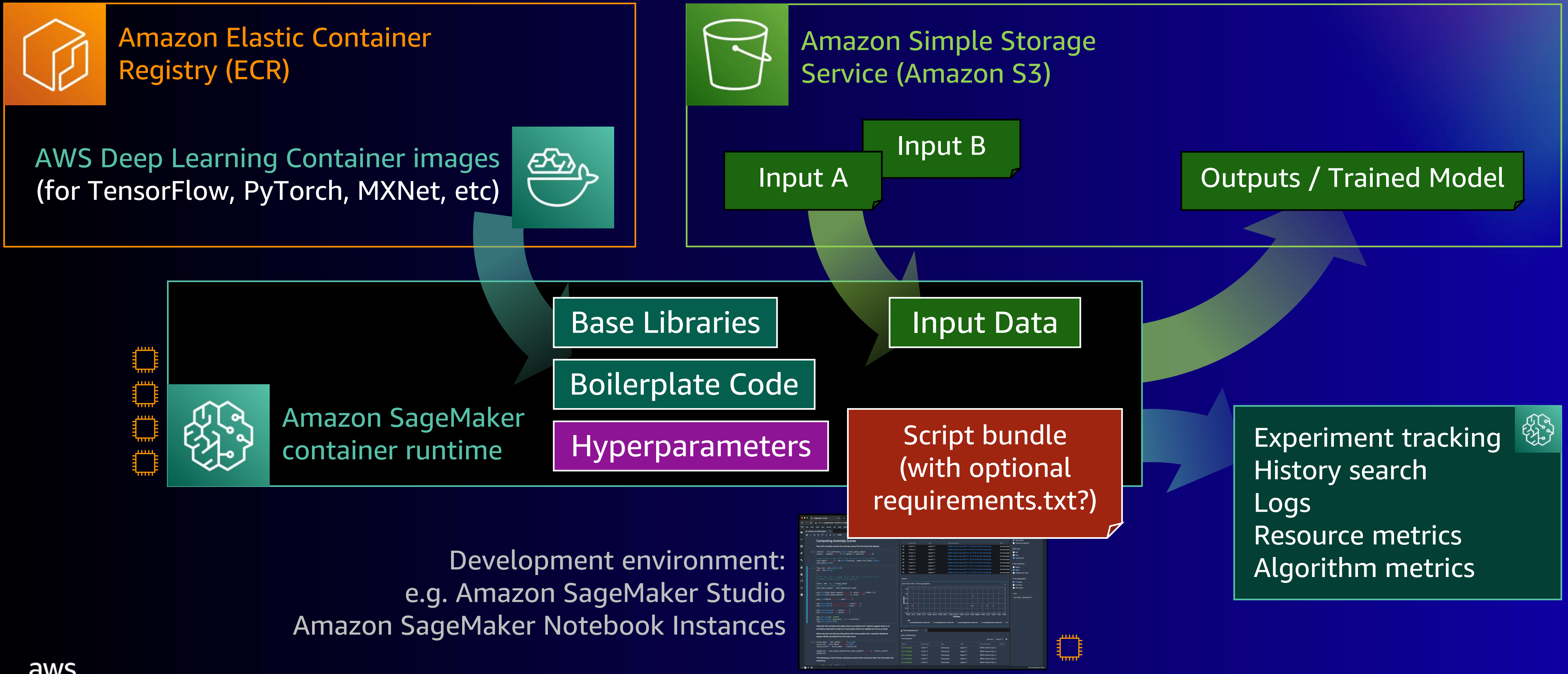
Understanding Amazon SageMaker infrastructure

“Work ***from*** the notebook, not ***on*** the notebook”



Understanding Amazon SageMaker infrastructure

"Work *from* the notebook, not *on* the notebook"



Training compute setup and managed spot

(In your notebook)

```
estimator = Estimator(
    ...,
    use_spot_instances=True,
    max_run=60*60*4,  # 4hrs training
    max_wait=60*60*8,  # 8hrs total
    checkpoint_s3_uri="s3://...",

    instance_count=4,
    instance_type="ml.p3.8xlarge",
    ...,
)
```

Enabling Amazon SageMaker Managed Spot is simple

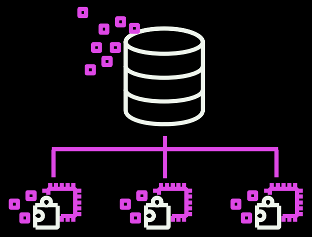
- Save up to 90% on compute costs!
- But as good practice, you should implement checkpointing

Before scaling out, check:

- Your framework and script are set up for multi-node training (not just duplicating!)
- Input channel distributions (shard or replicate?)

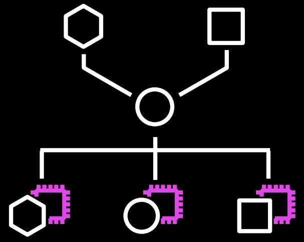
Select appropriate instance type

Amazon SageMaker distributed training libraries



Data Parallelism

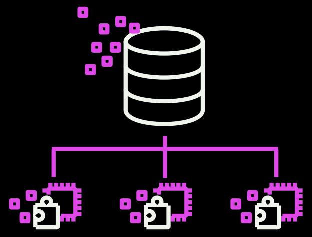
Scale out training clusters with near-linear efficiency, optimized for AWS networking and instance topology



Model Parallelism

Train models too large to fit within GPU memory, with automated model splitting and sophisticated pipeline scheduling

Optimize training with Amazon SageMaker Data Parallel



Data Parallelism

Scale out training clusters with near-linear efficiency, optimized for AWS networking and instance topology

Support for popular ML framework APIs

Re-use existing APIs such as Horovod and PyTorch DistributedDataParallel

Reduced training time

~25% faster with synchronization across GPUs (as tested with BERT)

Minimal code change

See SM Developer Guide for PyTorch & TensorFlow instructions – or use Hugging Face Trainer API scripts with no code changes at all!

Tune (hyper)-parameters automatically

(In your training script)

Read in parameters via
`/opt/ml/input/config/hyperparameters.json`
or CLI arguments

↓

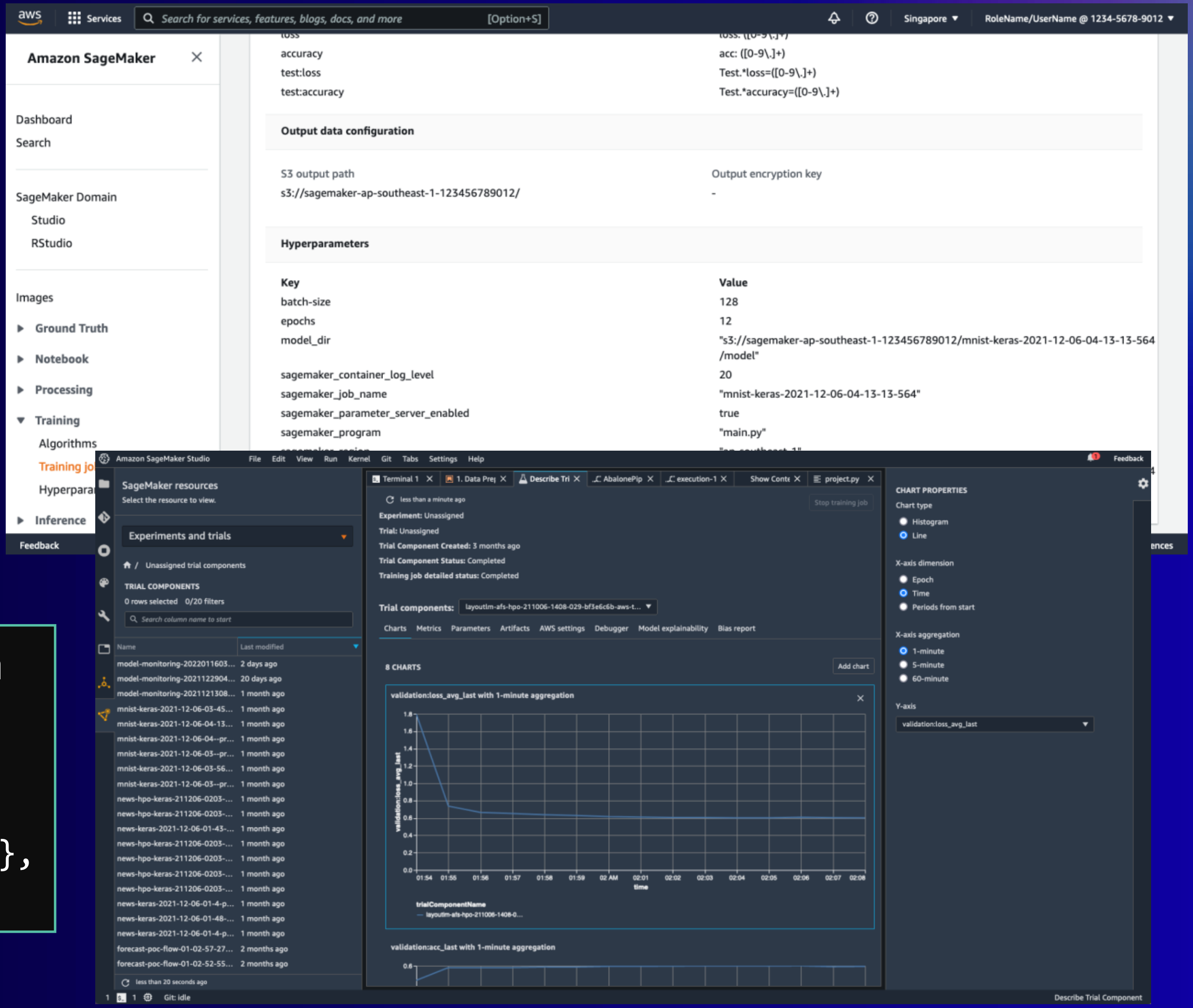
`print()` / log metrics to the console

↓

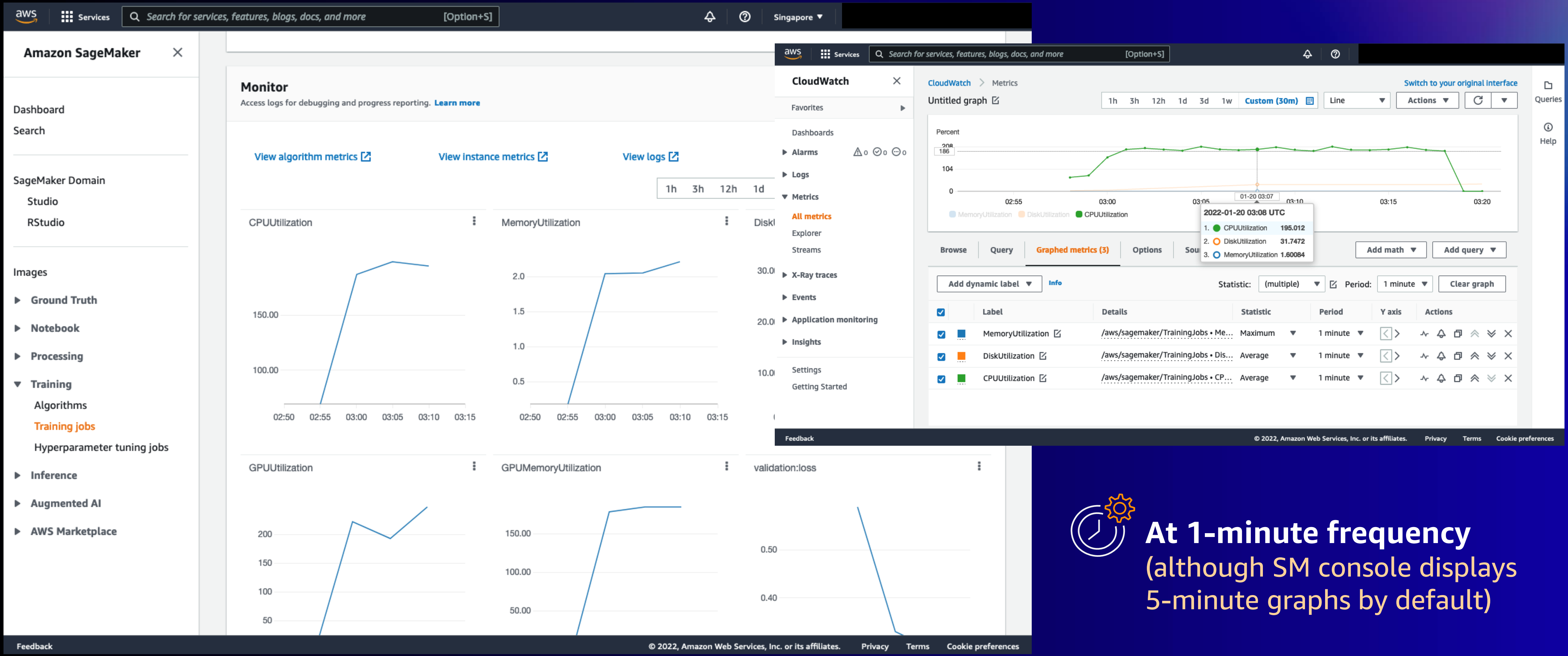
(In your notebook)

Define how Amazon SageMaker should scrape metrics from your training job logs via RegEx:

```
metric_definitions = [
  {"Name": "loss", "Regex": r"'loss': (-?[0-9\.\-e]+[,}]"},
]
```



Default Amazon SageMaker metrics via Amazon CloudWatch



At 1-minute frequency
(although SM console displays 5-minute graphs by default)

Tune (hyper)-parameters automatically

(In your training script)

Read in parameters via
`/opt/ml/input/config/hyperparameters.json`
or CLI arguments

`print()` / log metrics to the console

(In your notebook)

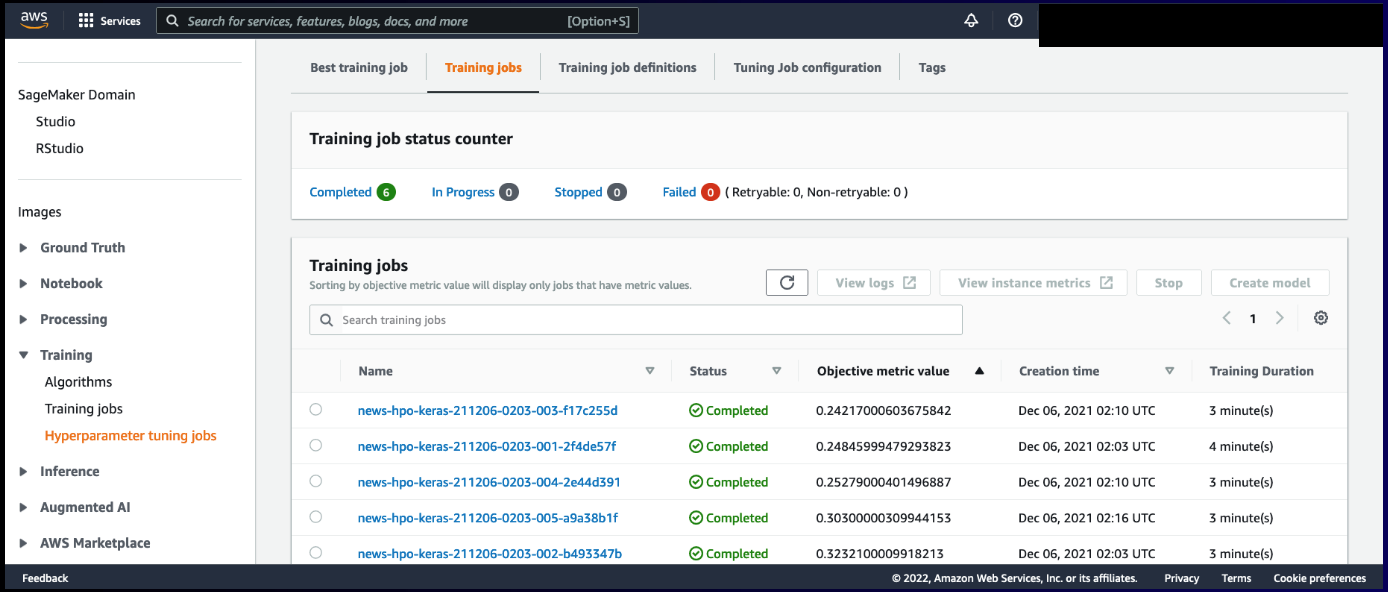
Define how Amazon SageMaker should scrape metrics from your training job logs via RegEx:

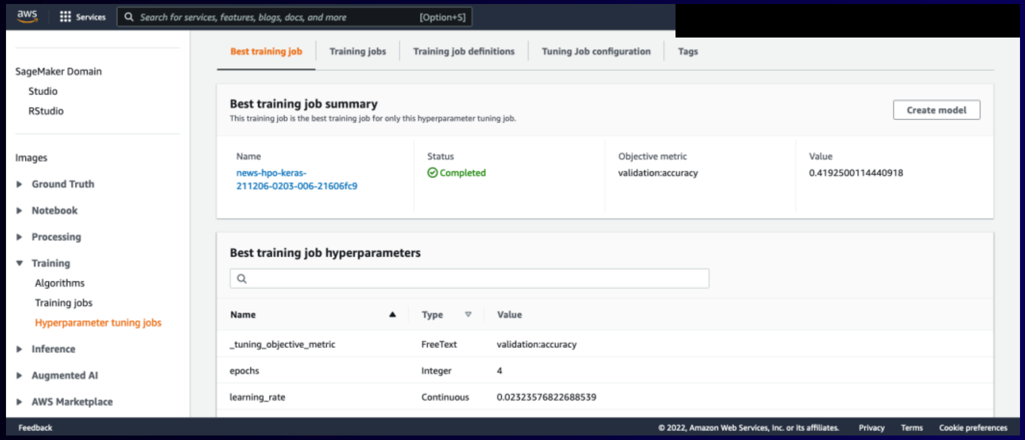
```
metric_definitions = [
  {"Name": "loss", "Regex": r"'loss': (-?[0-9\.\-e]+[,}]"},
]
```

Wrap your training job definition (or 'estimator') with automatic hyperparameter tuning!

```
sagemaker.tuner.HyperparameterTuner(
    estimator=...,
    metric_definitions=[...],
    objective_metric_name="loss",
    objective_type="Minimize",
    hyperparameter_ranges={
        "learning_rate":
sagemaker.parameter.ContinuousParameter(
            min_value=1e-8,
            max_value=1e-3,
            scaling_type="Logarithmic",
        ),
        ...
    },
    strategy="Bayesian",
    max_jobs=50,
    max_parallel_jobs=5,
    ...
)
```

Tune (hyper)-parameters automatically



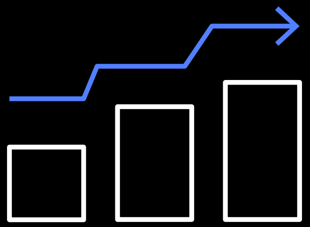


Wrap your training job definition (or 'estimator') with automatic hyperparameter tuning!

```
sagemaker.tuner.HyperparameterTuner(
    estimator=...,
    metric_definitions=[...],
    objective_metric_name="loss",
    objective_type="Minimize",
    hyperparameter_ranges={
        "learning_rate":
sagemaker.parameter.ContinuousParameter(
            min_value=1e-8,
            max_value=1e-3,
            scaling_type="Logarithmic",
        ),
        ...
    },
    strategy="Bayesian",
    max_jobs=50,
    max_parallel_jobs=5,
    ...
)
```

aws

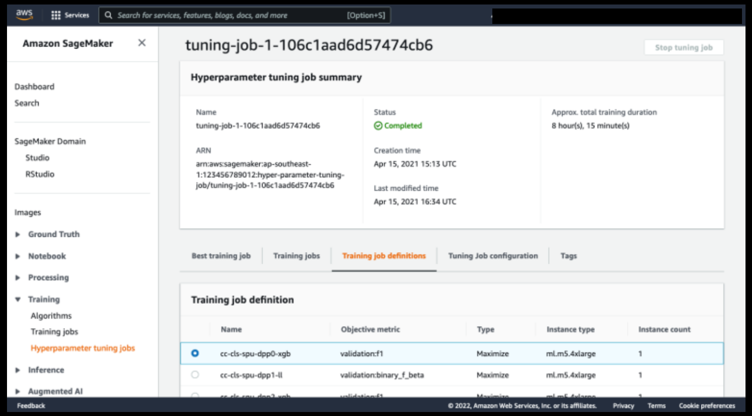
Advanced HPO tips



Continue a previous search

Warm-start the Bayesian optimizer to intelligently continue from previous HPO jobs

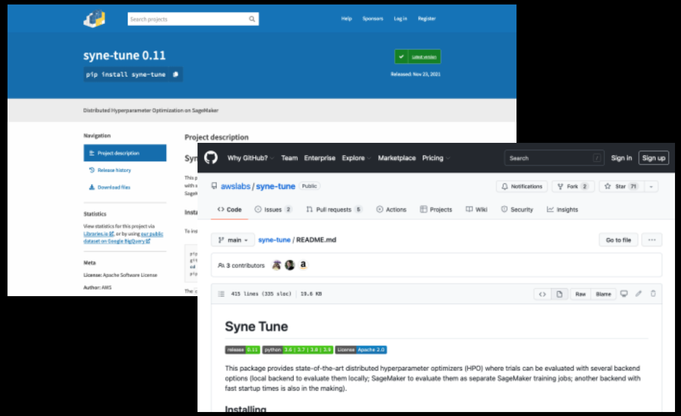
`warm_start_config` (sagemaker SDK) or `WarmStartConfig` (in API CreateHyperParameterTuningJob)



Multiple algorithms per HPO run

As used by Amazon SageMaker Autopilot! Searches may span multiple algorithms and container images

See `TrainingJobDefinitions` instead of `TrainingJobDefinition` in API!

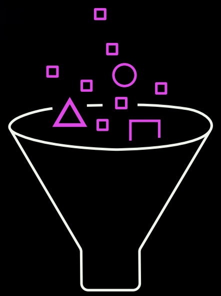


Customize further: Syne Tune

For advanced practitioners wanting more control than Amazon SageMaker's built-in HPO – see AWS Labs' open source Syne Tune library!

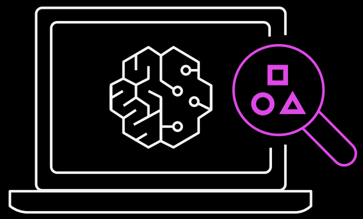
https://github.com/awslabs/syne-tune

Monitor and profile with Amazon SageMaker Debugger



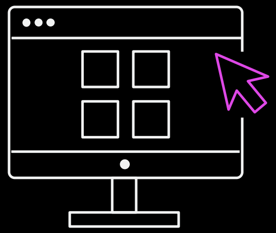
Capture data from training jobs

Apache MXNet
PyTorch
TensorFlow
XGBBoost



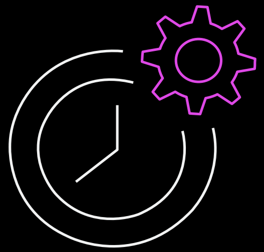
Real-time monitoring

Get deeper visibility into the training process as it runs



Automatic issue detection

Receive alerts to find and fix issues early, and accelerate prototyping

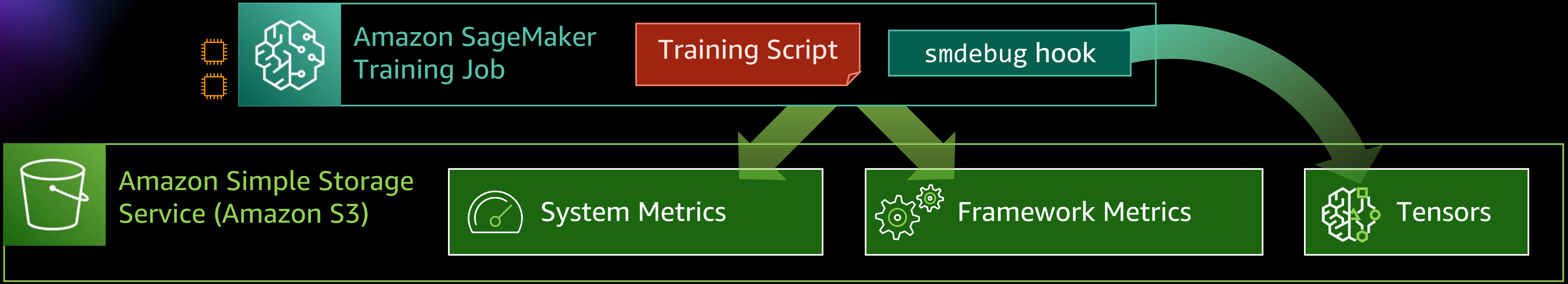


Resource profiling recommendations

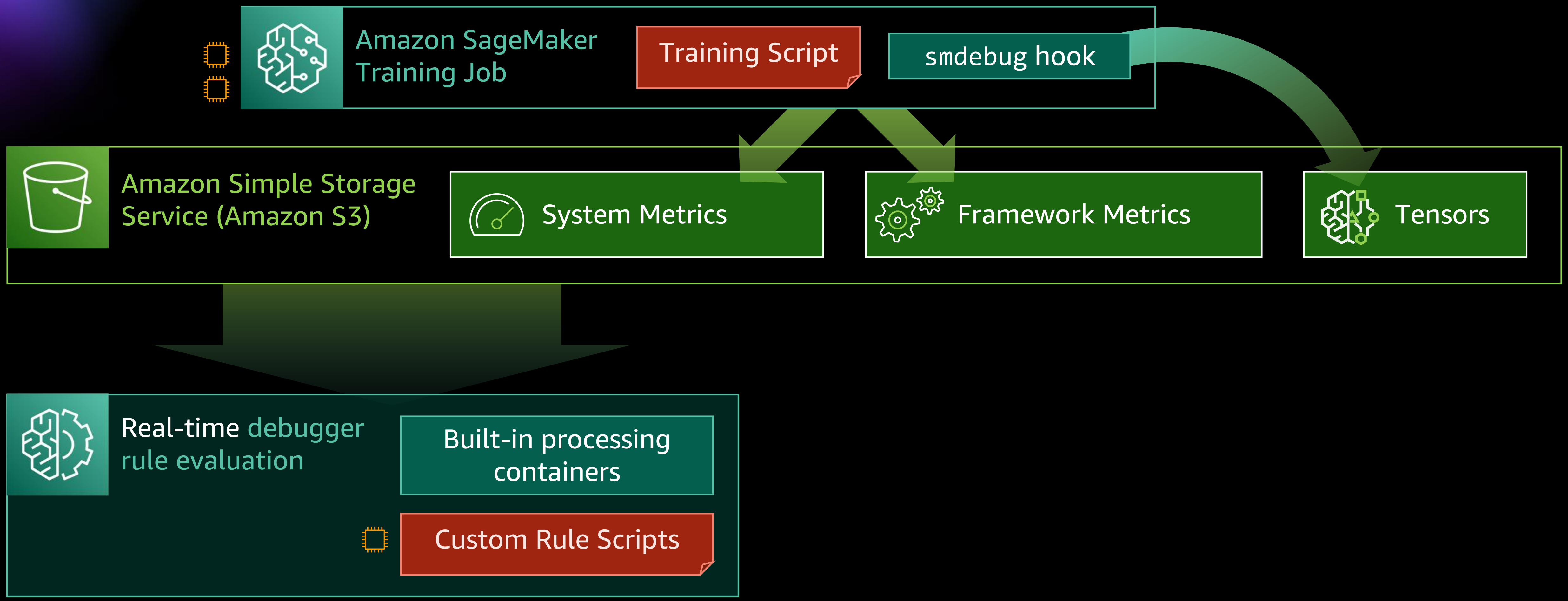
Find bottlenecks and optimize compute resources

Amazon SageMaker Debugger real-time monitoring

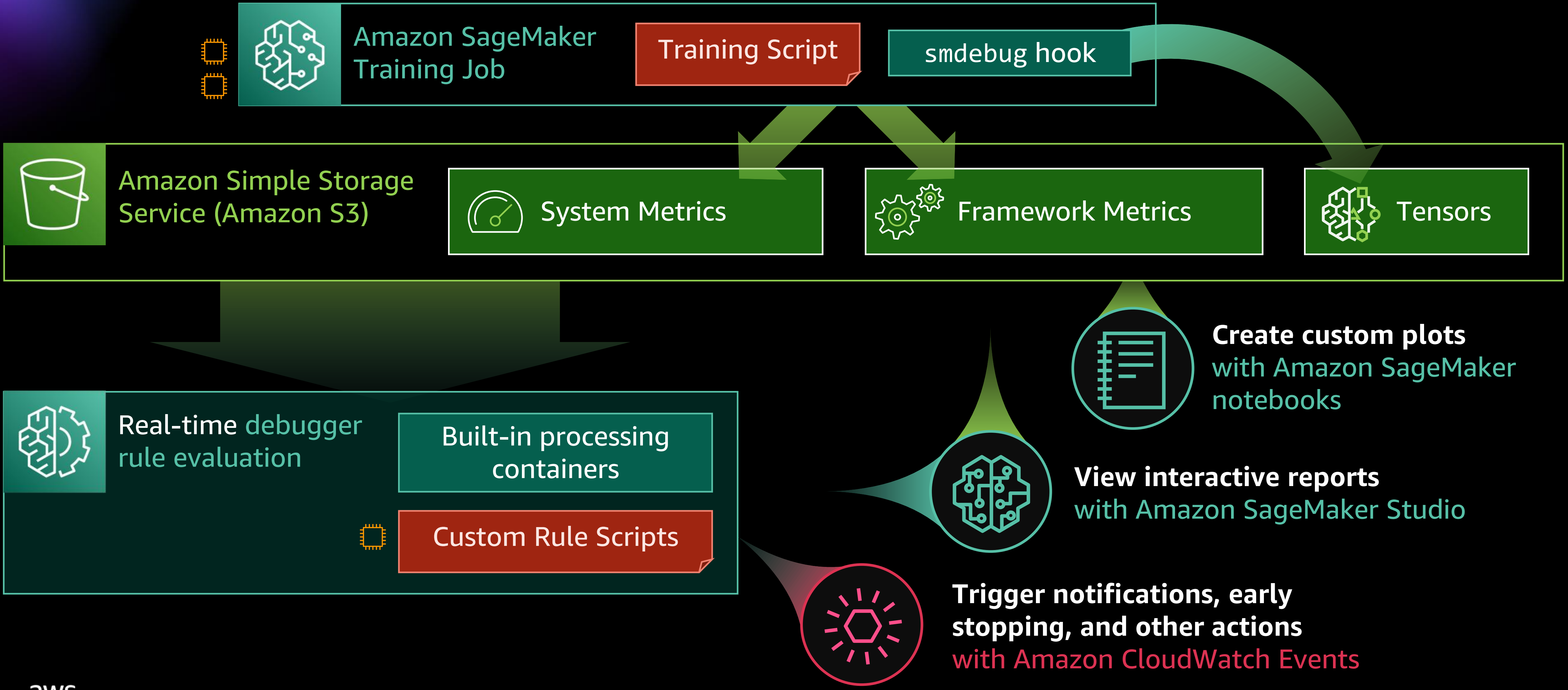
Amazon SageMaker Debugger real-time monitoring



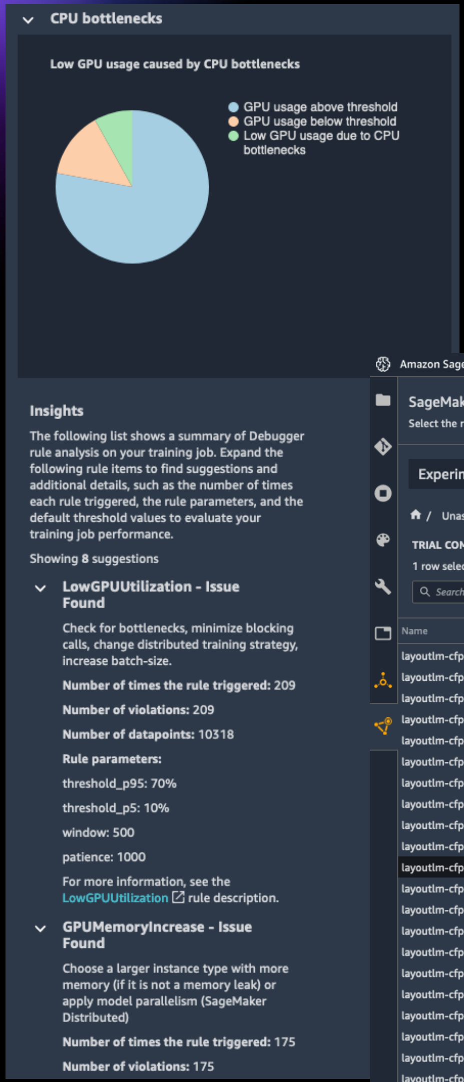
Amazon SageMaker Debugger real-time monitoring



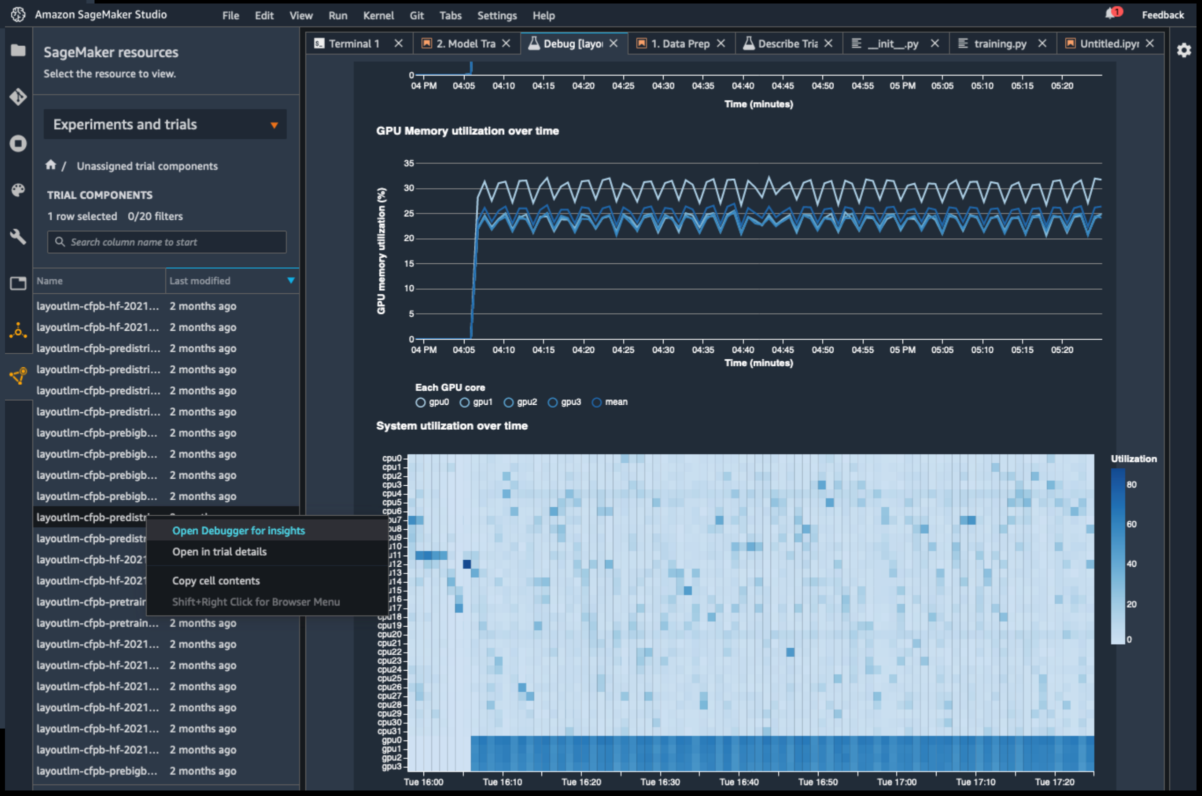
Amazon SageMaker Debugger real-time monitoring



Performance: Amazon SageMaker Debugger Profiler




```
estimator = PyTorch(  # Or TensorFlow, MXNet, etc
  ...,
  profiler_config=sagemaker.debugger.ProfilerConfig(
    system_monitor_interval_millis=100,
    framework_profile_params=sagemaker.debugger.FrameworkProfile(),
  ),
)
```



Granular system metrics down to 100ms interval

Interactive reports with downloadable options

Automatic recommendations to resolve bottlenecks

Extra options for operator, data-loader, and Python function profiling

aws

Optimize training data input

Amazon Simple Storage Service (Amazon S3)

Most common, feature-rich option, with advanced data lake capabilities

Amazon FSx for Lustre

High-performance file system optimized for ML and HPC workloads

Amazon Elastic File System (Amazon EFS)

Consider mainly just if your source data is already on Amazon EFS today

Optimize training data input

Amazon Simple Storage Service (Amazon S3)
Most common, feature-rich option, with advanced data lake capabilities

Amazon FSx for Lustre
High-performance file system optimized for ML and HPC workloads

Amazon Elastic File System (Amazon EFS)
Consider mainly just if your source data is already on Amazon EFS today

File Mode (Default)
Up-front download to local filesystem before your job starts

Pipe Mode
Stream data for serial access (usually needing code changes)

Fast File Mode (NEW 2021)
File-like access backed by on-demand streaming!

Optimize training data input

Amazon Simple Storage Service (Amazon S3)
Most common, feature-rich option, with advanced data lake capabilities

Amazon FSx for Lustre
High-performance file system optimized for ML and HPC workloads

Amazon Elastic File System (Amazon EFS)
Consider mainly just if your source data is already on Amazon EFS today

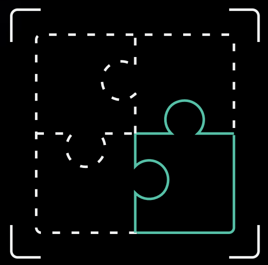
File Mode (Default)
Up-front download to local filesystem before your job starts

Pipe Mode
Stream data for serial access (usually needing code changes)

Fast File Mode (NEW 2021)
File-like access backed by on-demand streaming!

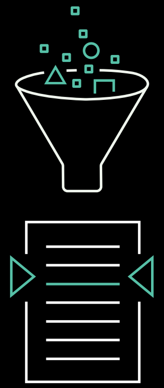
```
estimator.fit({
  "train": sagemaker.inputs.TrainingInput(
    "s3://doc-example-bucket/a-folder/",
    input_mode="FastFile",
  ),
  ...,
})
```

Other start-up optimization tips



Don’t use the whole dataset until needed

Initial, basic, functional tests should use small subsets of the data



Preprocess your data for performance

Avoid huge numbers of tiny files

Explore optimized formats like RecordIO & TFRecord

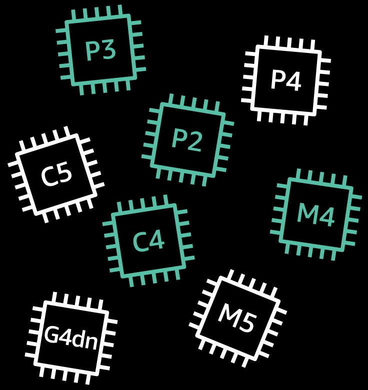
Amazon SageMaker Processing can help you scale out!

Build-in common dependencies

`requirements.txt` is convenient but repetitive

Deriving images ‘`FROM`’ AWS DLCs may be easier!

Smaller containers can start jobs faster



Instance type can affect start-up time

If working in an environment that supports docker, consider Amazon SageMaker Local Mode for initial functional tests

Demo

Recap and resources

Recap

1. Amazon SageMaker Managed Spot Instances
2. Amazon SageMaker Distributed Libraries
3. Amazon Metrics and Automatic Hyperparameter Tuning
4. Amazon SageMaker Debugger and Profiler
5. Amazon Input Optimization including Fast File Mode and Amazon FSx for Lustre

Resources

Amazon SageMaker developer guide: https://docs.aws.amazon.com/sagemaker/latest/dg/

Amazon SageMaker Python SDK documentation: https://sagemaker.readthedocs.io/en/stable/

Amazon SageMaker repository: https://github.com/aws/amazon-sagemaker-examples

AWS ML blog: https://aws.amazon.com/blogs/machine-learning/

Visit the AI & Machine Learning resource hub for more resources

Dive deeper into these resources, get inspired and learn how you can use AI and machine learning to accelerate your business outcomes.

- The machine learning journey e-book
- 7 leading machine learning use cases e-book
- A strategic playbook for data, analytics, and machine learning e-book
  Accelerate machine learning innovation with the right cloud services & infrastructure e-book
- Choosing the right compute infrastructure for machine learning e-book
- Improving service and reducing costs in contact centers e-book
- Why ML is essential in your fight against online fraud e-book
- … and more!

https://bit.ly/3mwi59V

Visit resource hub

AWS Machine Learning (ML) Training and Certification

AWS is how you build machine learning skills

Courses built on the curriculum leveraged by Amazon's own teams. Learn from the experts at AWS.

aws.training/machinelearning

Flexibility to learn your way

Learn online with on-demand digital courses or live with virtual instructor-led training, plus hands-on labs and opportunities for practical application.

explore.skillbuilder.aws/learn

Validate your expertise

Demonstrate expertise in building, training, tuning, and deploying machine learning models with an industry-recognized credential.

aws.amazon.com/certification

Thank you for attending AWS Innovate – AI/ML Edition

We hope you found it interesting! A kind reminder to **complete the survey.**
Let us know what you thought of today's event and how we can improve the event experience for you in the future.

- aws-apj-marketing@amazon.com
- twitter.com/AWSCloud
- facebook.com/AmazonWebServices
- youtube.com/user/AmazonWebServices
- slideshare.net/AmazonWebServices
- twitch.tv/aws

Thank you!

Alex Thewsey