Mastering Amazon SageMaker

Distributed Training & Deployment using Amazon SageMaker

Name: Mani Khanuja

Role: Sr. AI/ML Specialist SA

Email: mankhanu@amazon.com

Agenda

- SageMaker Training Deep Dive
    - Distributed Training
    - SageMaker Inference
    - Autoscaling
    - SageMaker Batch Transform
    - Async Inference
    - SageMaker Inference Pipelines
    - SageMaker Multi-model Endpoints
    - SageMaker Model Monitoring
- Q&A
- Survey

Distributed training

The fastest and easiest way to train large deep learning models

Reduced training time
Reduce training time by 25% with synchronization across GPUs

Optimized for AWS
Achieve near-linear scaling efficiency with data parallelism designed for AWS

Support for popular ML framework APIs
Re-use existing APIs such as Horovod without custom training code

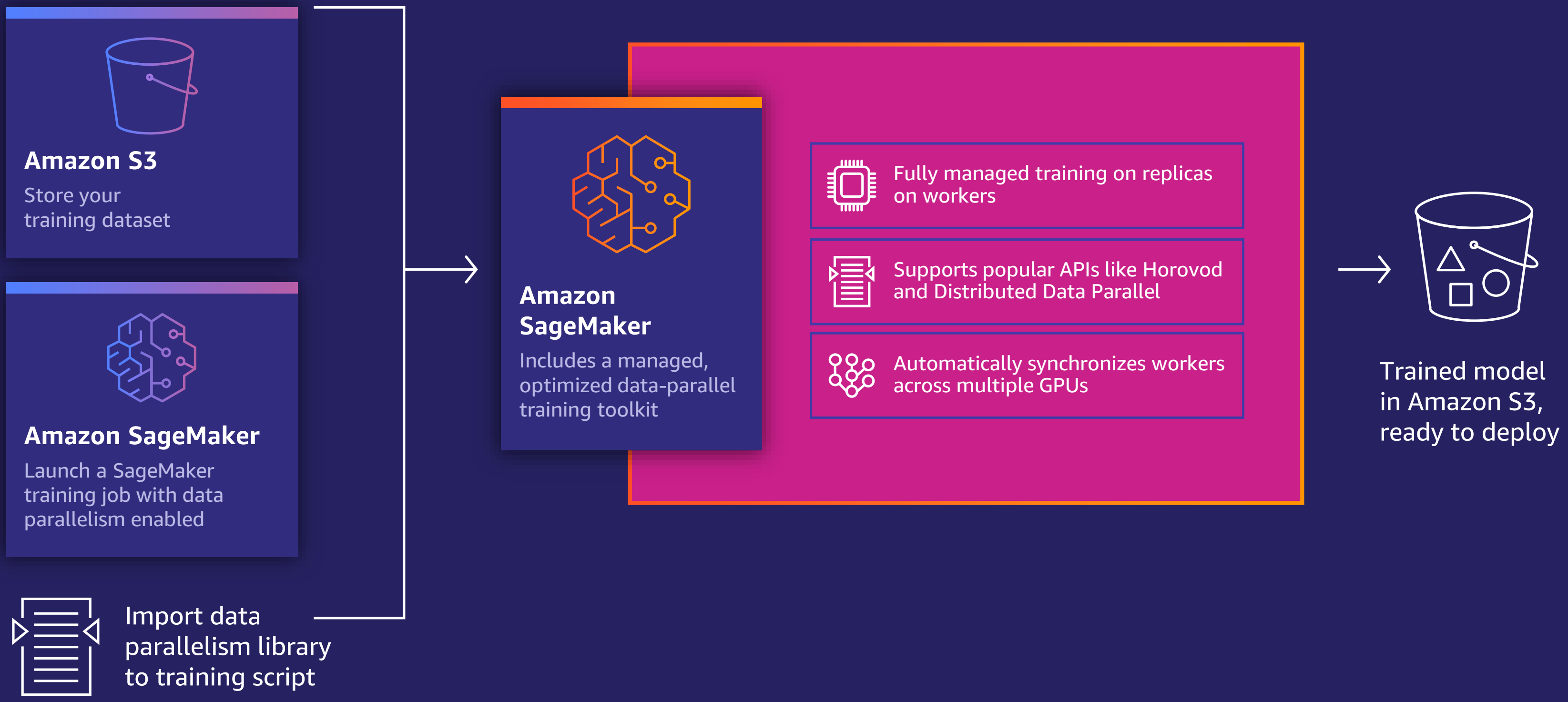
Automatic and efficient model partitioning
Avoid experimentation with automated model profiling and partitioning

Minimal code change
Implement model parallelism with fewer than 10 lines of code change

Efficient pipelining
Maximize resource usage with pipelining of micro-batches that keeps all GPUs active

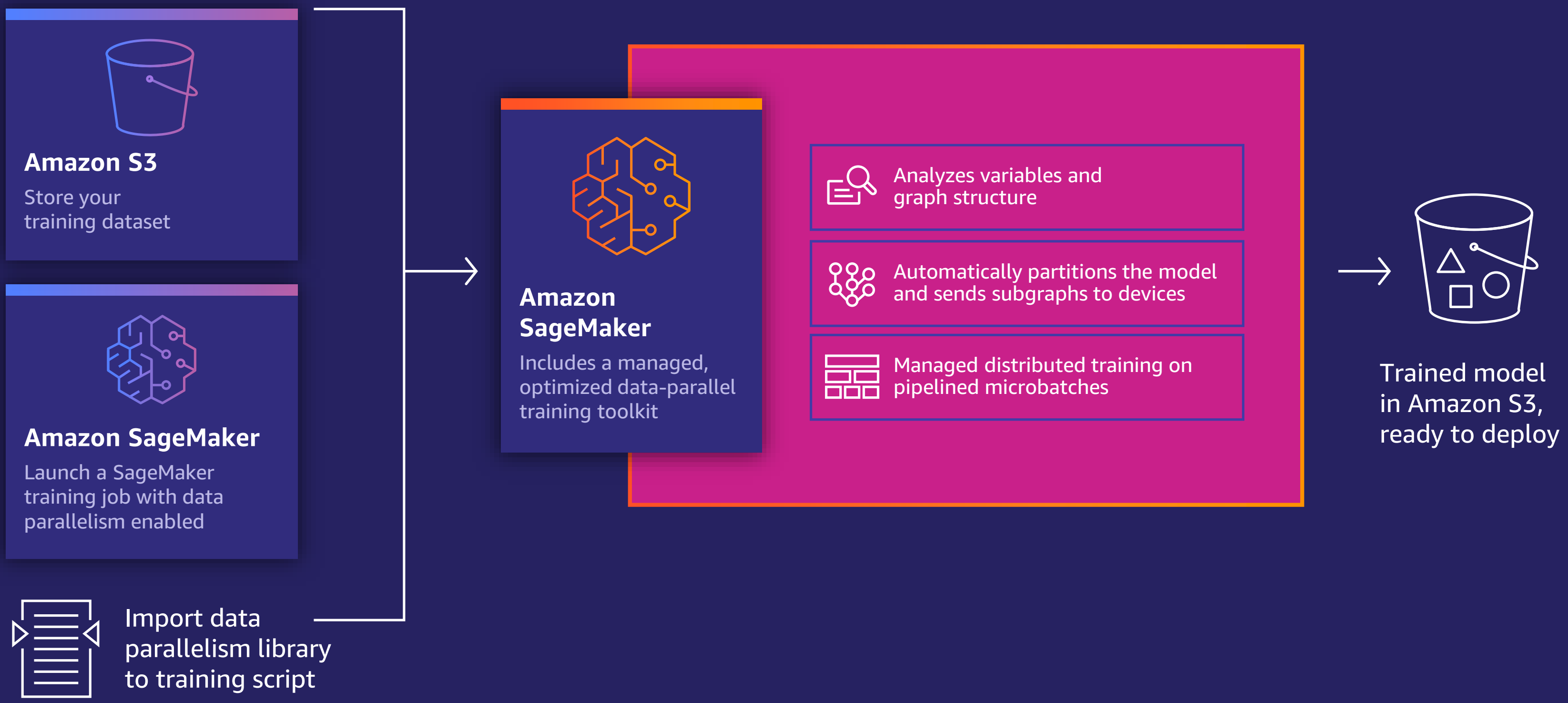
aws machine learning

How it works: Data parallelism library



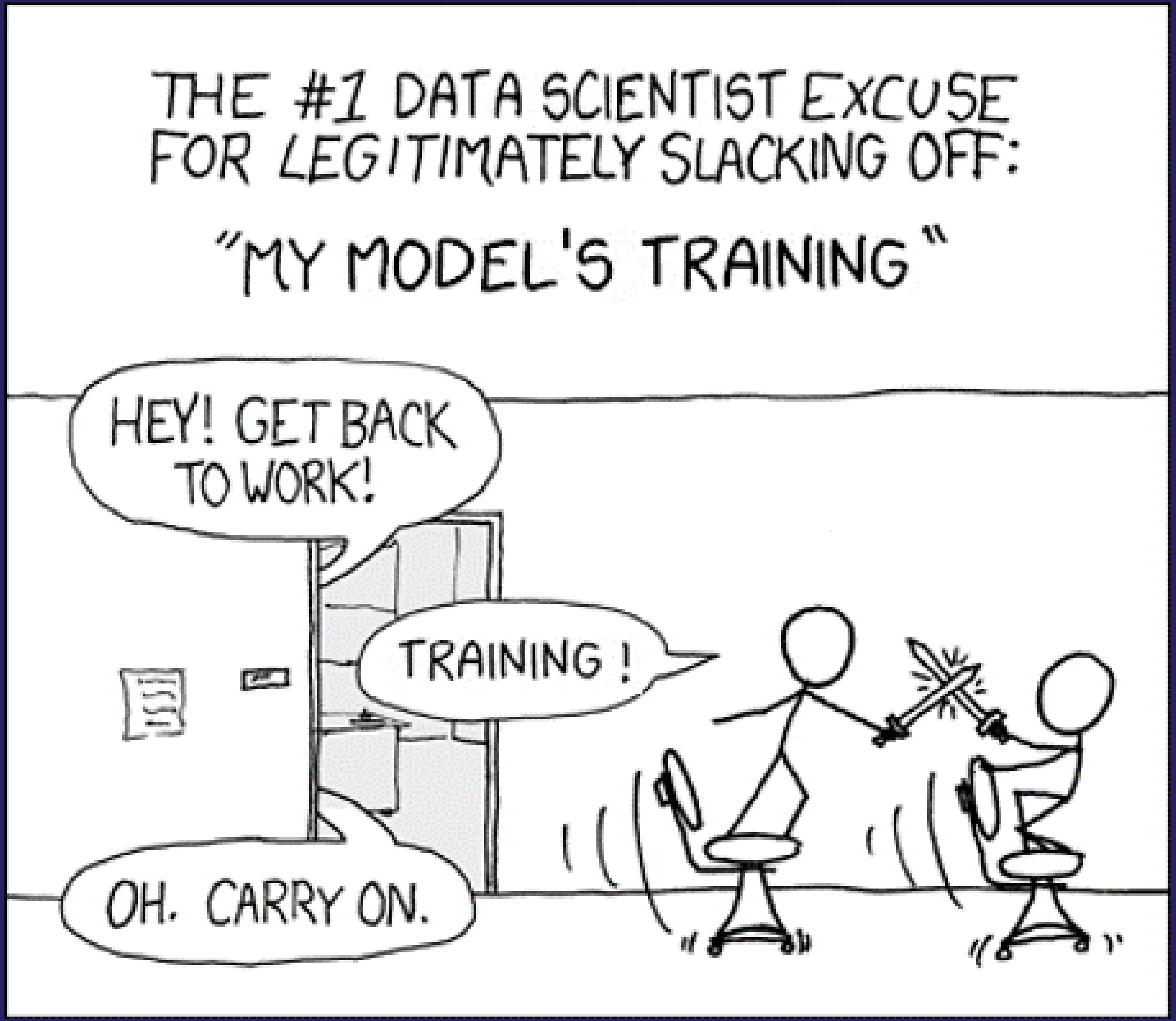
aws machine learning

How it works: Model parallelism library



aws machine learning

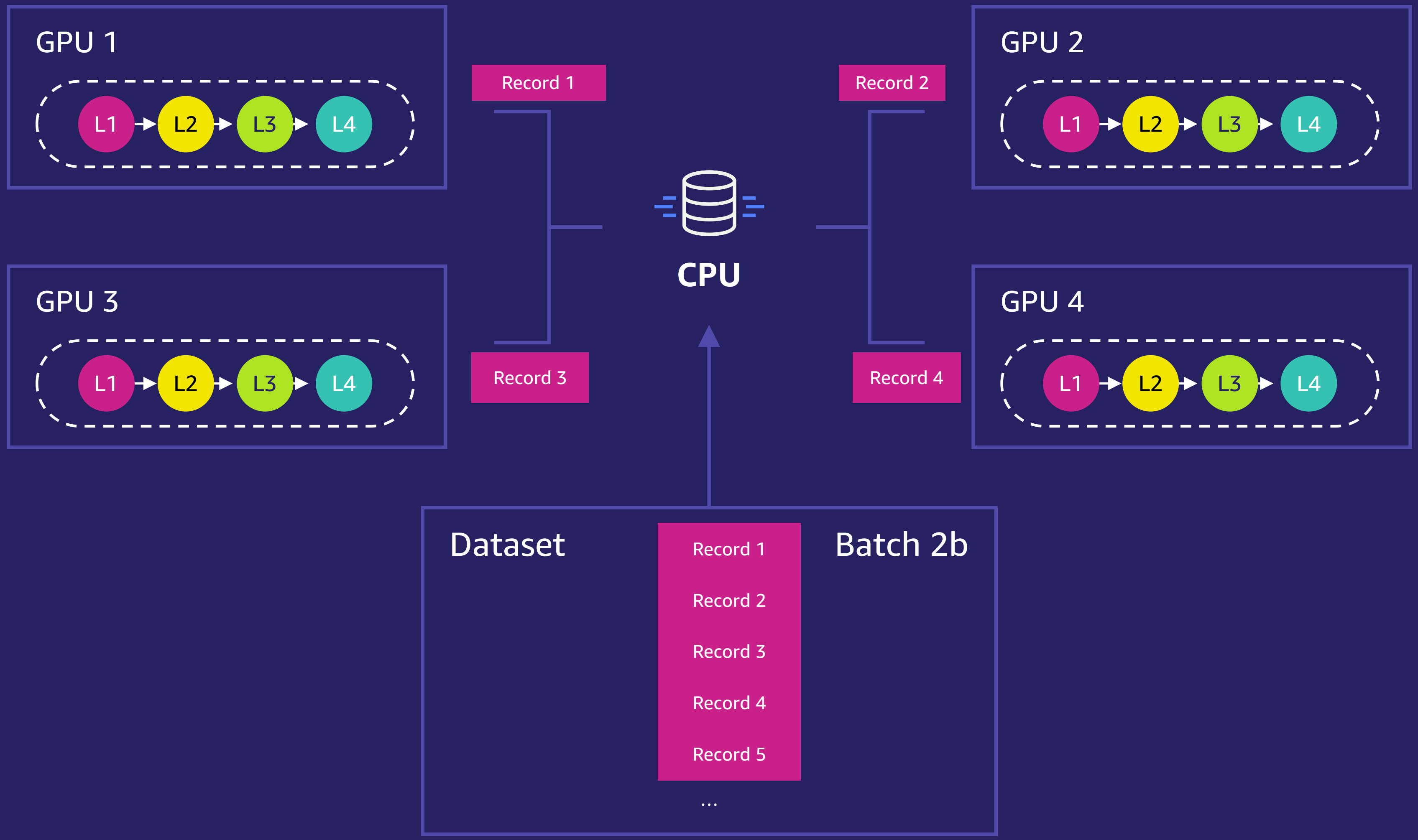
Training time slows down development



Credit: https://xkcd.com/303/

Model	RoBERTa
Dataset	300+ GB
Cluster	64 p3dn.24xl
Training time	Several days

Data parallelism in a nutshell



aws machine learning

Model parallel – think "massive models"

aws machine learning

Deep learning models are growing in size

MODEL	RELEASED	# PARAMETERS
BERT	Oct 2018	340 M
GPT-2	Feb 2019	1.5 B
T5	Oct 2019	11 B
GPT-3	Jul 2020	175 B

But hardware improvements are not keeping up

- HARDWARE CAPACITY GROWS TOO - BUT NOT AS FAST

INSTANCE TYPE	AVAILABLE	GPU	GPU MEMORY
P2.16xlarge	Sep 2016	NVIDIA K80	12 GB
P3.16xlarge	Oct 2017	NVIDIA V100	16 GB
P3dn.24xlarge	Dec 2018	NVIDIA V100	32 GB
P4d.24xlarge	Nov 2020	NVIDIA A100	40 GB

The speed of model size growth is outpacing hardware improvements, leading to memory bottlenecks.

Memory bottlenecks in training large models

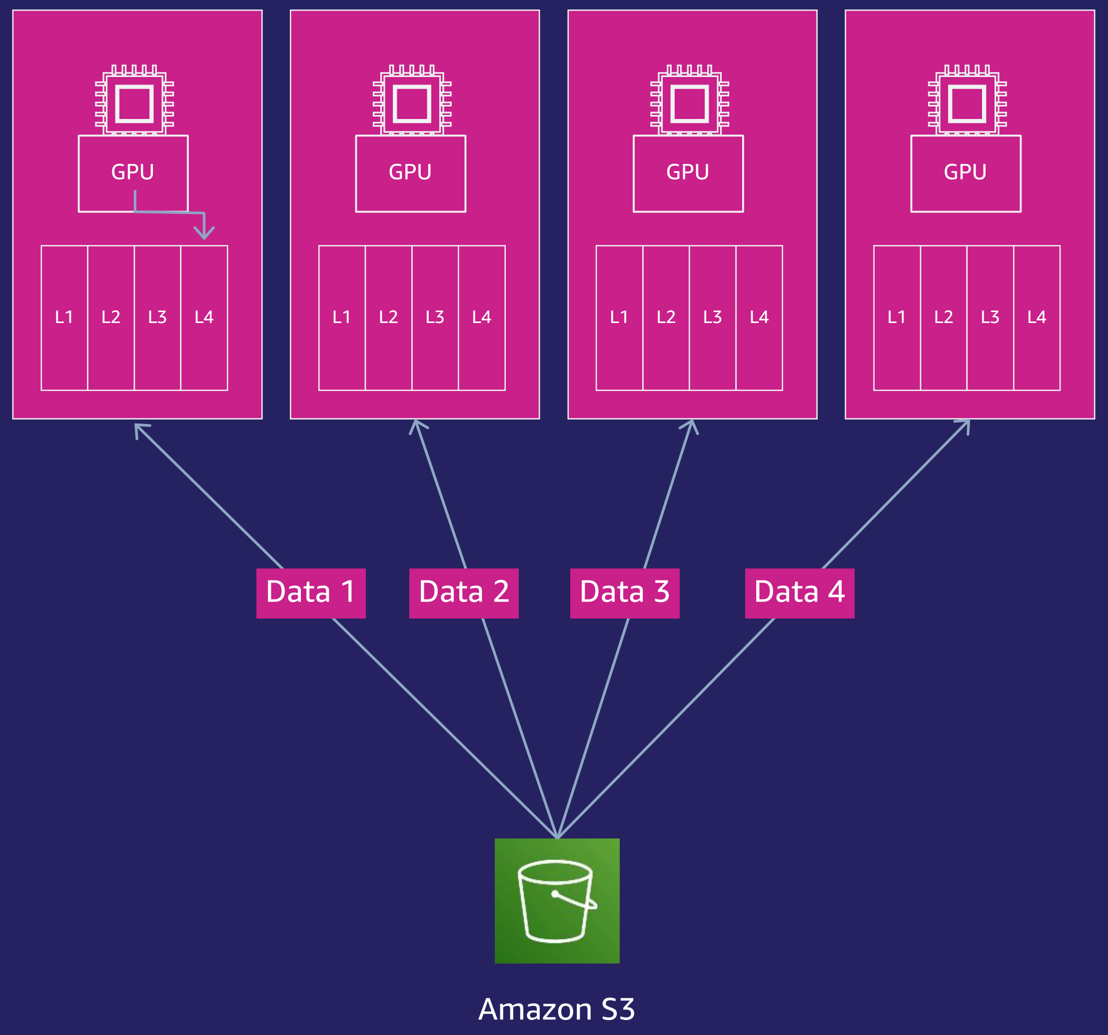
Model sizes can be limited by memory of a single GPU

- Large models cause OOM errors

Naive approaches to model parallel replicate the entire model across all GPUS

- Wasteful when model is large, gets you even more OOM errors

Hardware limitations without optimal usage can limit both research and applications



aws machine learning

Model parallelism on Amazon SageMaker (SMP)

Automated model partitioning

Interleaved pipelined training

Managed SageMaker training

Clean framework integration

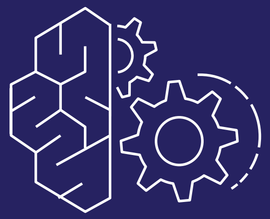
aws machine learning

1. Use SMP to automate your model partitioning

ANALYZED MODEL

- Graph structure
- Sizes of trainable weights
- Sizes of exchanged tensors
  (using SageMaker Debugger)

RUN GRAPH PARTITIONING ALGORTIHM



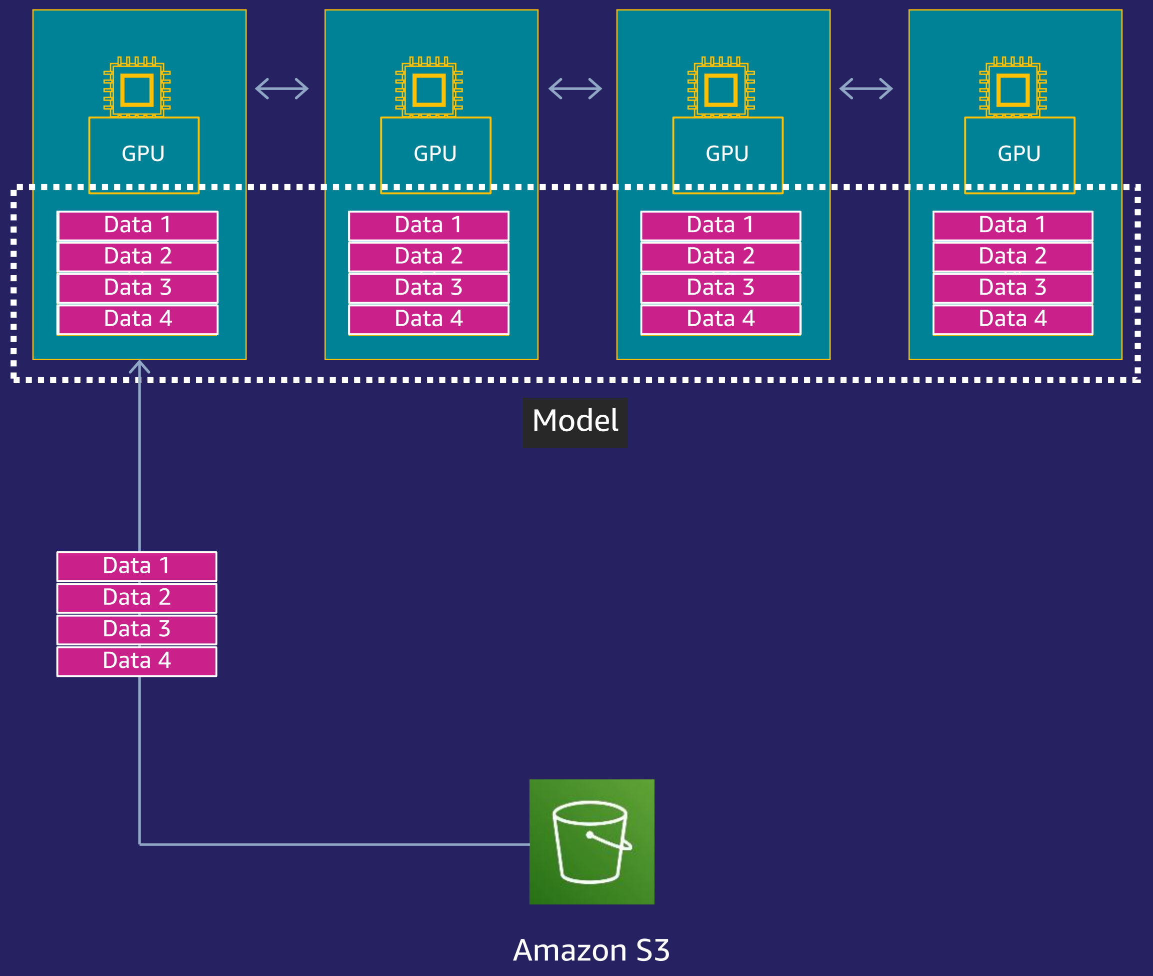
- Balance stored weights and activations
- Minimize communication

PLACE PARTITIONS ON DEVICES

- To be executed in a pipelined manner

2. Interleave pipeline execution to stabilize GPU utilization

- Split minibatches into N “microbatches”
- Feed microbatches sequentially, but process them to keep GPU utilization more even
- Minimize “idle” time on GPUs



aws machine learning

SageMaker Hosting Deep Dive

aws machine learning

Deploy ML models

Fully managed deployment for inference at scale

Wide selection of infrastructure
70+ instance types with varying levels of compute and memory to meet the needs of every use case

Single-digit millisecond overhead latency
For use cases requiring real-time responses

Asynchronous inference
Supports large models with long-running processing times

Cost-effective deployment
Multi-model/multi-container endpoints, serverless inference, and elastic scaling

Built-in integration for MLOps
ML workflows, CI/CD, lineage tracking, and catalog

Automatic deployment recommendations
Optimal instance type/count and container parameters, and fully managed load testing

aws machine learning

SageMaker inference options

NEW

Real-time inference

Low latency

Ultra high throughput

Multi-model endpoints

A/B testing

Batch transform

Process large datasets

Job-based system

Asynchronous inference

Near real-time

Large payloads (1 GB)

Long timeouts (15 mins)

aws machine learning

Serverless Inference
Fully managed offering

Managed infrastructure

Security

Monitoring

Logging

Built in availability and fault tolerance

Serverless

No need to select instance types or provision capacity

Choose memory options based on inference processing needs

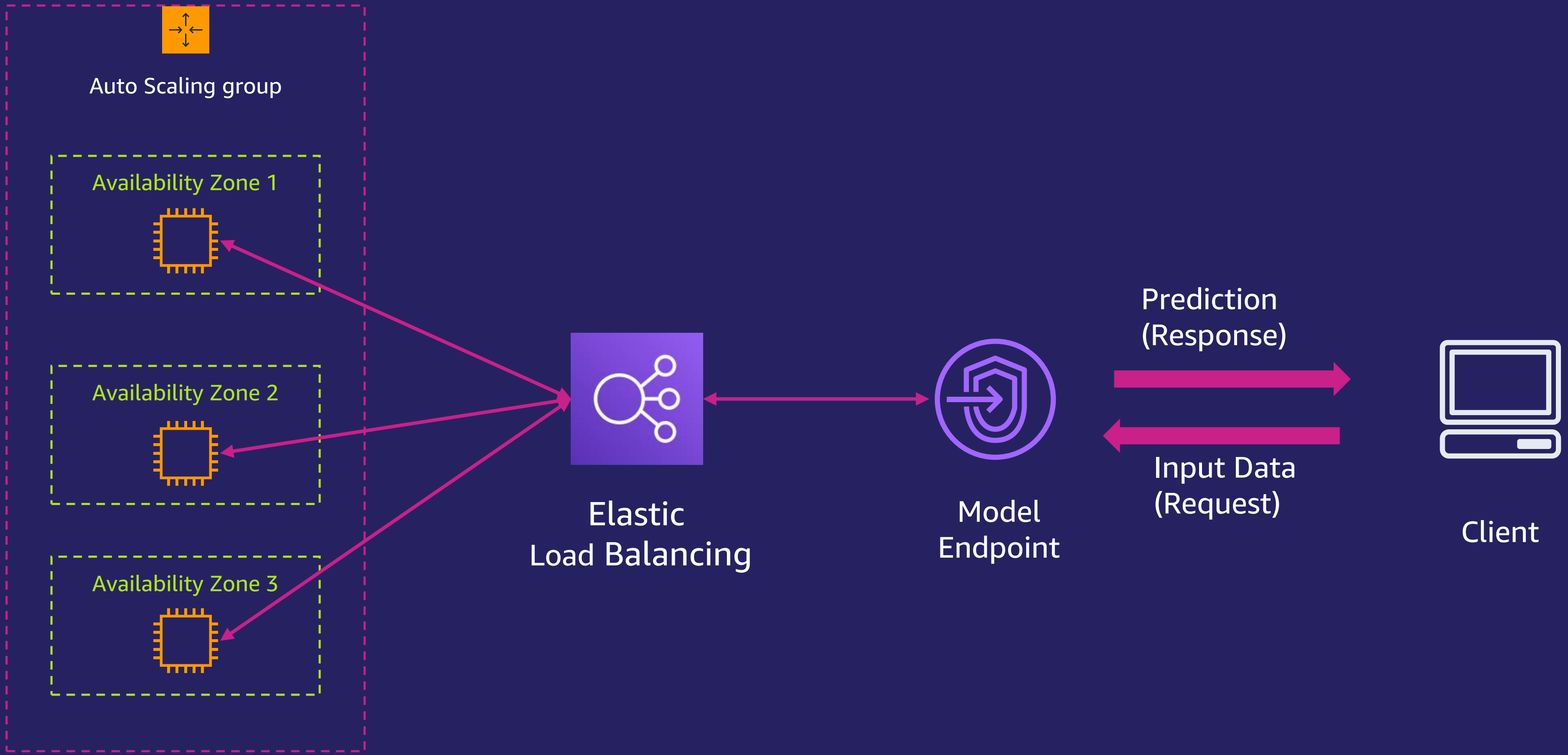
Automatically scale out, in, and down to 0

No need to set scaling policies

aws machine learning

Amazon SageMaker Deployment

SageMaker Endpoints (Private API)

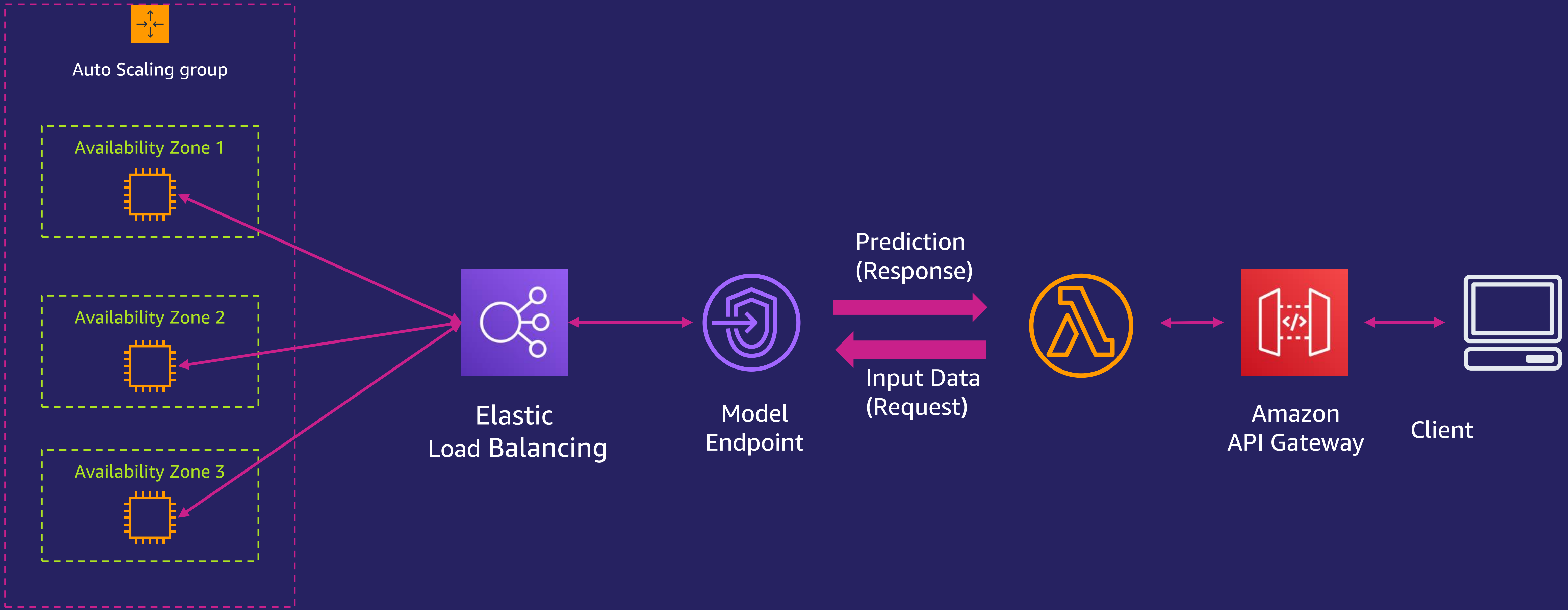


Deployment / Hosting
Amazon SageMaker ML
Compute Instances

aws machine learning

Amazon SageMaker Deployment

SageMaker Endpoints (Public API)



Deployment / Hosting
Amazon SageMaker ML
Compute Instances

aws machine learning

Real time Inference

```python
from time import gmtime, strftime
from sagemaker.tensorflow.model import TensorFlowModel

tensorflow_model = TensorFlowModel(model_data=model_path,
                                   role=role,
                                   framework_version="2.3.1")

endpoint_name = "DEMO-tf2-california-housing-model-monitor-" + strftime(
    "%Y-%m-%d-%H-%M-%S", gmtime()
)

predictor = tensorflow_model.deploy(
    initial_instance_count=1,
    instance_type="ml.m5.xlarge",
    endpoint_name=endpoint_name,
)
```

Serverless Inference

Step1 : Create Model

```
from time import gmtime, strftime

model_name = "xgboost-serverless" + strftime("%Y-%m-%d-%H-%M-%S", gmtime())
print("Model name: " + model_name)

# dummy environment variables
byo_container_env_vars = {"SAGEMAKER_CONTAINER_LOG_LEVEL": "20", "SOME_ENV_VAR": "myEnvVar"}

create_model_response = client.create_model(
    ModelName=model_name,
    Containers=[
        {
            "Image": image_uri,
            "Mode": "SingleModel",
            "ModelDataUrl": model_artifacts,
            "Environment": byo_container_env_vars,
        }
    ],
    ExecutionRoleArn=role,
)

print("Model Arn: " + create_model_response["ModelArn"])
```

Step2 : Create Endpoint Configuration

```
xgboost_epc_name = "xgboost-serverless-epc" + strftime("%Y-%m-%d-%H-%M-%S", gmtime())

endpoint_config_response = client.create_endpoint_config(
    EndpointConfigName=xgboost_epc_name,
    ProductionVariants=[
        {
            "VariantName": "byoVariant",
            "ModelName": model_name,
            "ServerlessConfig": {
                "MemorySizeInMB": 4096,
                "MaxConcurrency": 1,
            },
        },
    ],
)

print("Endpoint Configuration Arn: " + endpoint_config_response["EndpointConfigArn"])
```

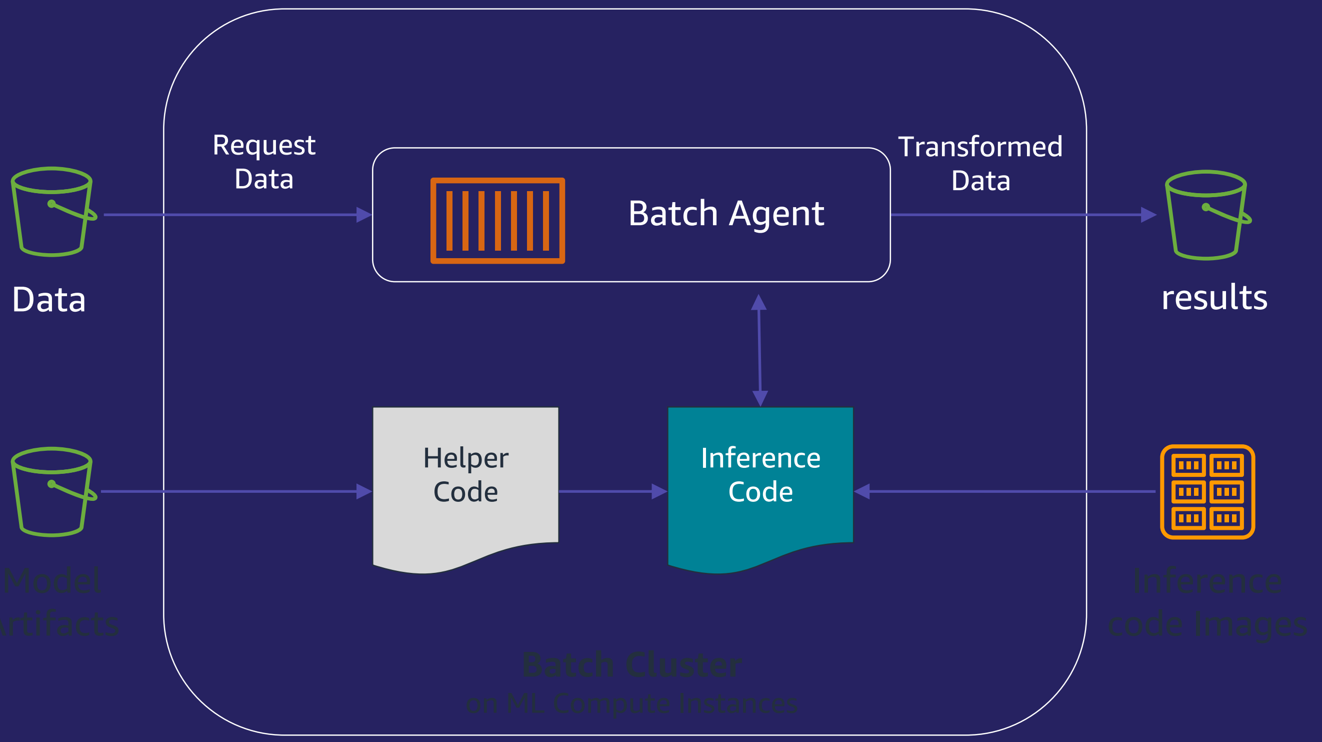
Step3 : Create Endpoint

```
endpoint_name = "xgboost-serverless-ep" + strftime("%Y-%m-%d-%H-%M-%S", gmtime())

create_endpoint_response = client.create_endpoint(
    EndpointName=endpoint_name,
    EndpointConfigName=xgboost_epc_name,
)

print("Endpoint Arn: " + create_endpoint_response["EndpointArn"])
```

Amazon SageMaker Deployment
Batch Hosting

Batch Transform

- Predictions for an entire dataset
- Transient resources (instances provisioned and terminated once job is done)
- No infrastructure to manage
- Can associate prediction results with input
- Supports Built-In/Bring-Your-Own

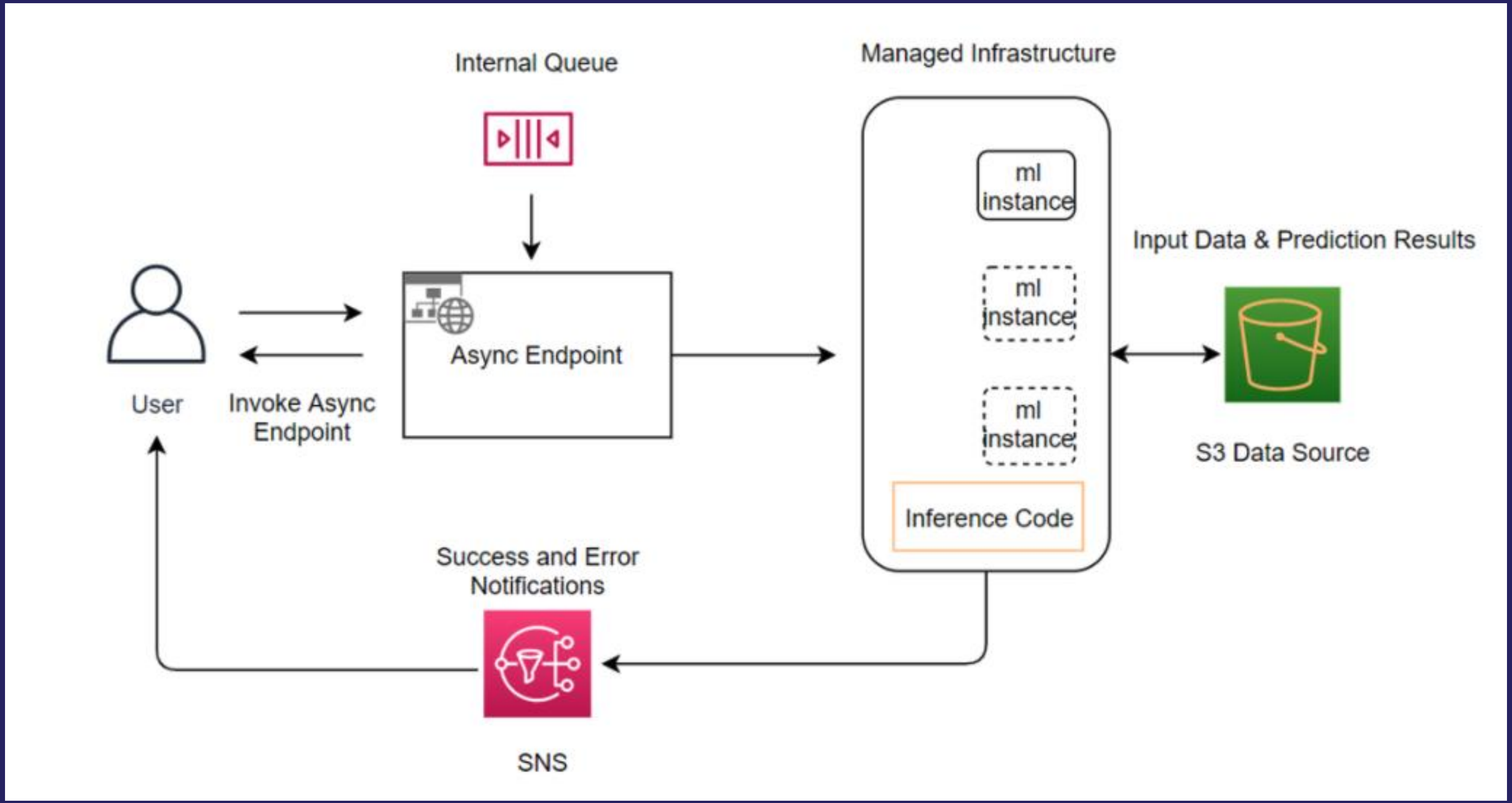


Batch Inference

```
from sagemaker.tensorflow import TensorflowModel

serving = TensorflowModel(
    model_data = 's3 location',
    role=role,
    framework_version="2.3",
    sagemaker_session=sagemaker_session,
    entry_point = 'inference.py',
    source_dir = 'code'
)

input_data_path = "s3://sagemaker-sample-data-{}/tensorflow/california_housing_data/batch.csv".format(
    sagemaker_session.boto_region_name
)
output_data_path = "s3://{}/{}/{}".format(bucket, prefix, "batch-predictions")
batch_instance_count = 2
batch_instance_type = "ml.g4dn.2xlarge"
concurrency = 32
max_payload_in_mb = 1

transformer = serving.transformer(
    instance_count=batch_instance_count,
    instance_type=batch_instance_type,
    max_concurrent_transforms=concurrency,
    max_payload=max_payload_in_mb,
    output_path=output_data_path,
)

transformer.transform(data=input_data_path, content_type="text/csv")
transformer.wait()
```

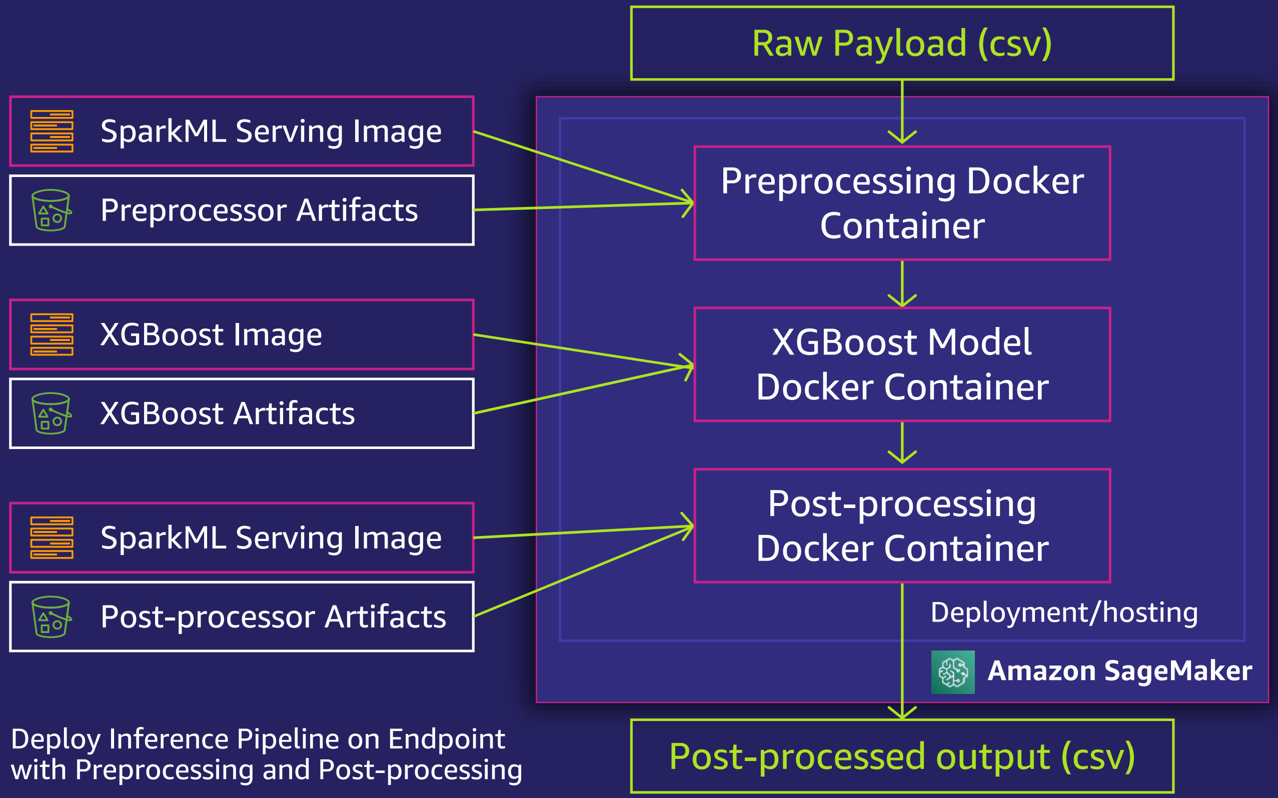
aws machine learning

Async Inference



Inference Pipelines for sequential execution of models

Execute data processing on inference requests, maintain single copy of data processing code for training and inference



Deploy Inference Pipeline on Endpoint with Preprocessing and Post-processing

Built-in containers—Scikit-Learn and Apache Spark MLLib

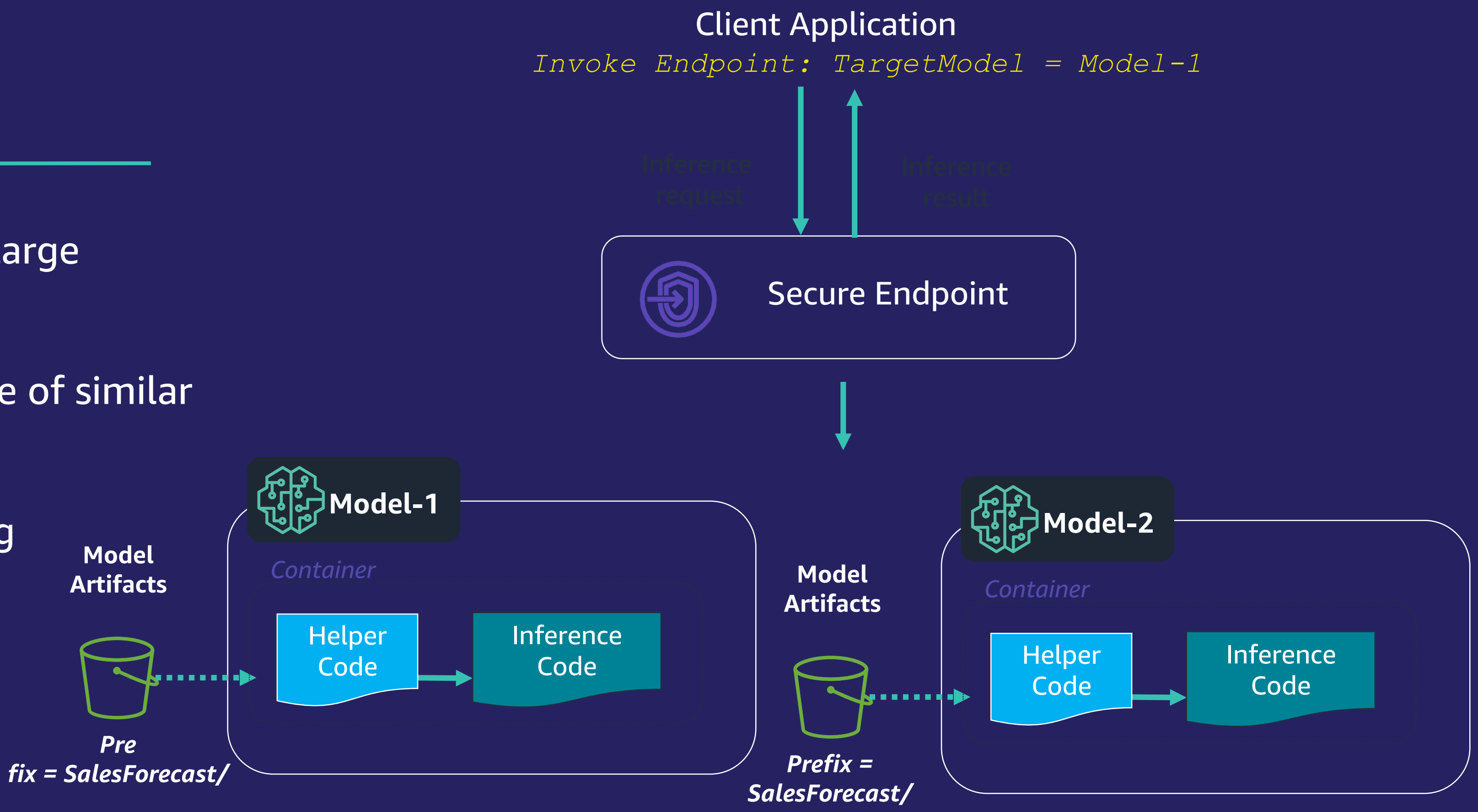
Add up to 5 containers; execute sequentially

Containers co-located on instance for low latency

Amazon SageMaker Deployment

Multi-Model Endpoints

- Scalable/Cost Effective for large number of models
- Works best when models are of similar size and latency
- Automatic memory handling

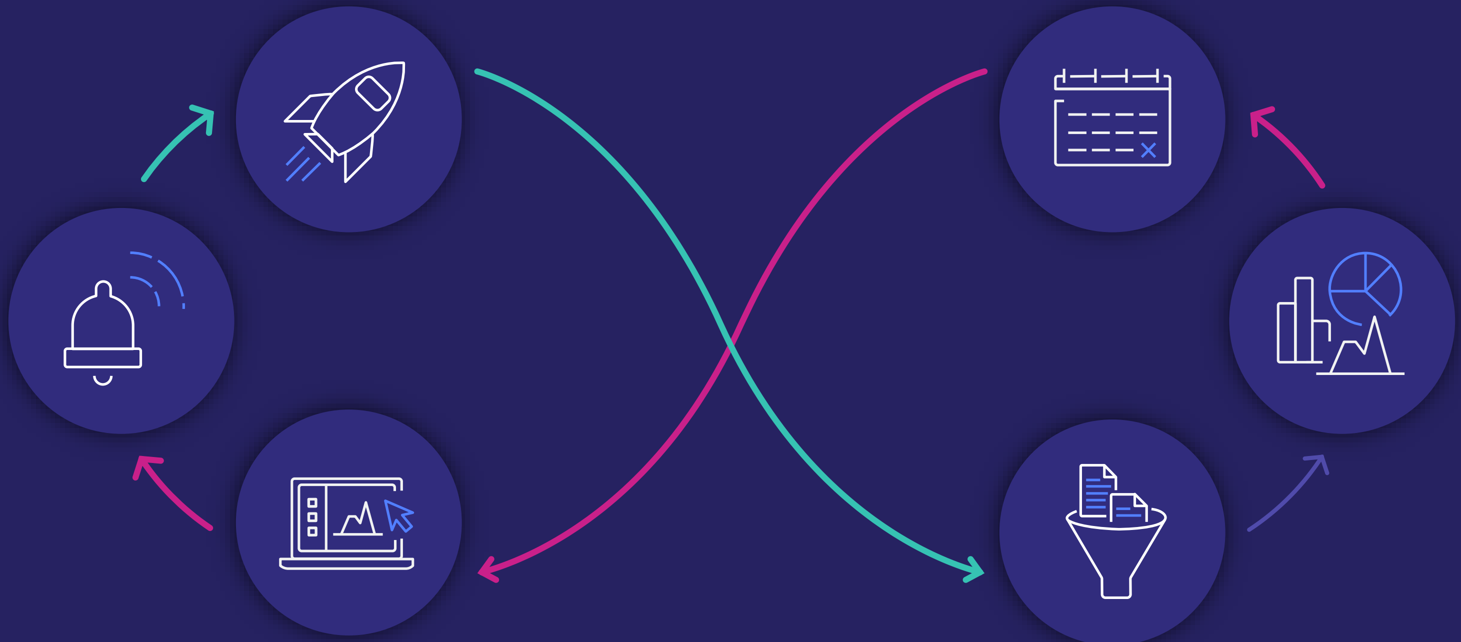


SageMaker Model Monitoring

Model Monitor: how it works

Deploy trained model
Enable data capture on SageMaker Endpoint

Schedule monitoring jobs
Detect and monitor data quality, model accuracy, bias, and explainability drift



CloudWatch alerts
Retrain models/ update data

Generate baseline
Statistics and constraints

View drift monitoring results
Reports in S3, visualize metrics in Studio, analyze in Notebooks

Collect ground truth
Merge with predictions

DEMO

Amazon SageMaker

Next Steps

Onboarding & Processing

- https://docs.aws.amazon.com/sagemaker/latest/dg/gs-studio-onboard.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/processing-job.html

Training

- https://docs.aws.amazon.com/sagemaker/latest/dg/train-model.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/distributed-training.html
- https://aws.amazon.com/sagemaker/debugger

Deployment

- https://docs.aws.amazon.com/sagemaker/latest/dg/realtime-endpoints.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/serverless-endpoints.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/async-inference.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/batch-transform.html

https://github.com/aws/amazon-sagemaker-examples

https://sagemaker.readthedocs.io/en/stable/index.html

aws machine learning

Q & A

Please Complete the session Survey

Thank you!