AWS re:Invent

AIM404-R

# Amazon SageMaker RL: Solving business problems with RL and bandits

**Girish Dilip Patil**
Senior Architect
Amazon Web Services India

**Marc Cabocel**
Senior Architect
Amazon Web Services France

**Segolene Dessertine-panhard**
Data Scientist
Amazon Web Services France

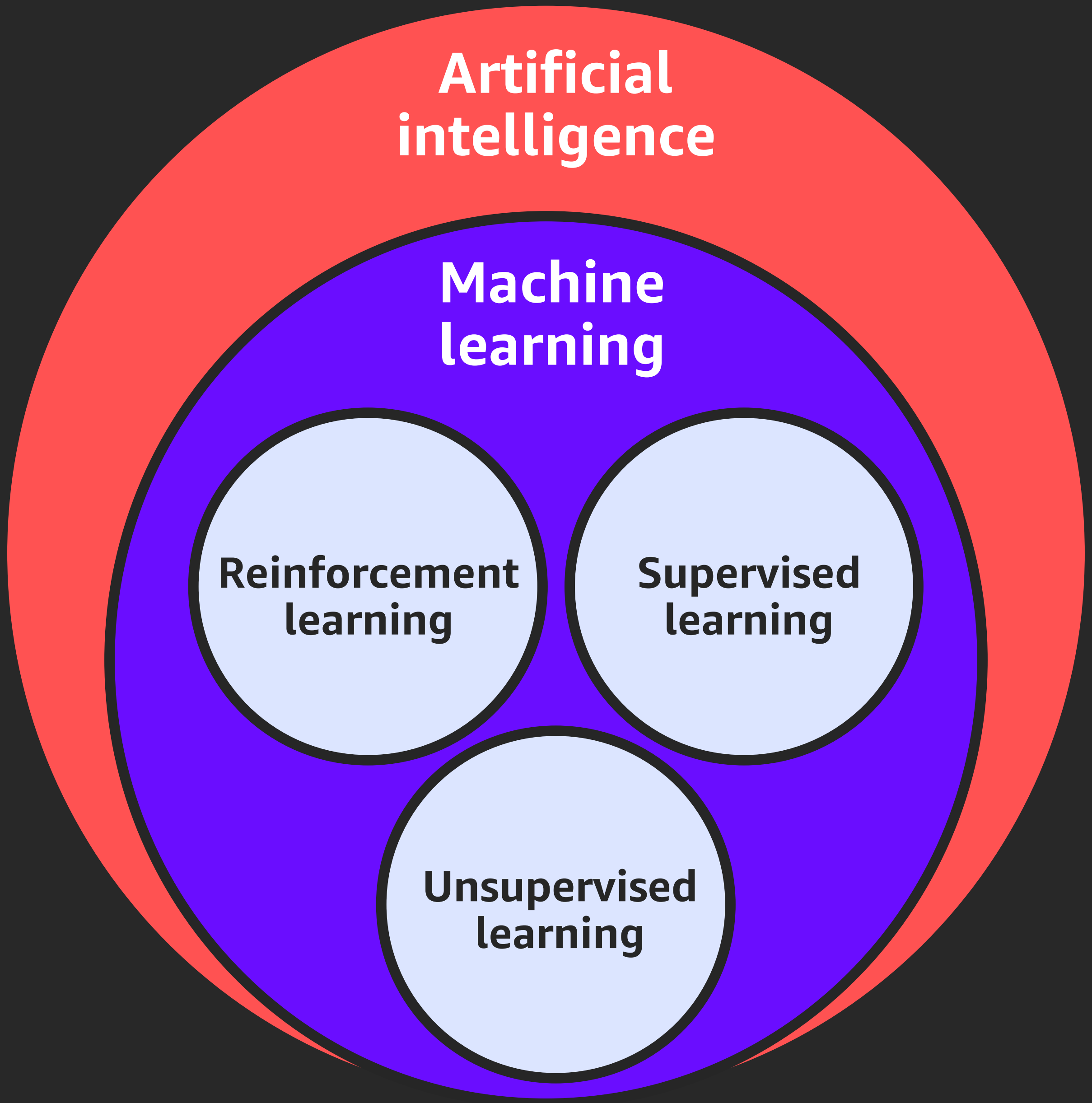
**Anna Luo**
Applied Scientist
Amazon Web Services USA

# Agenda

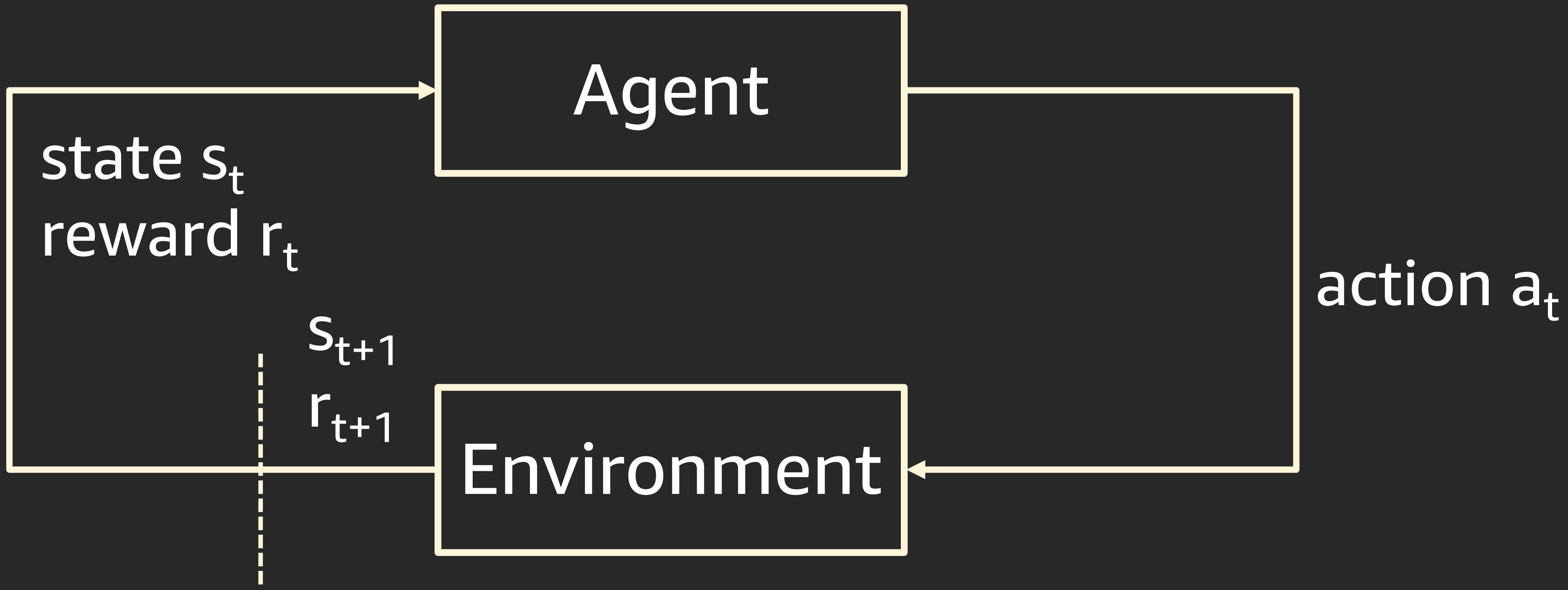
1. A quick primer on reinforcement learning (RL)
2. An important trade-off: Explore vs. exploit
3. Amazon SageMaker RL
4. Workshop #1: Training without a simulator in a real environment
5. Workshop #2: Training with a simulator
6. Conclusion

# A quick primer on reinforcement learning

# Reinforcement learning in the broader artificial intelligence context



# Reinforcement learning



Reinforcement learning is based on the reward hypothesis:

All goals can be described by the maximization of expected cumulative reward

# How RL differs from other variations of machine learning

- Reinforcement learning helps in learning a strategy to maximize a reward in a specific environment

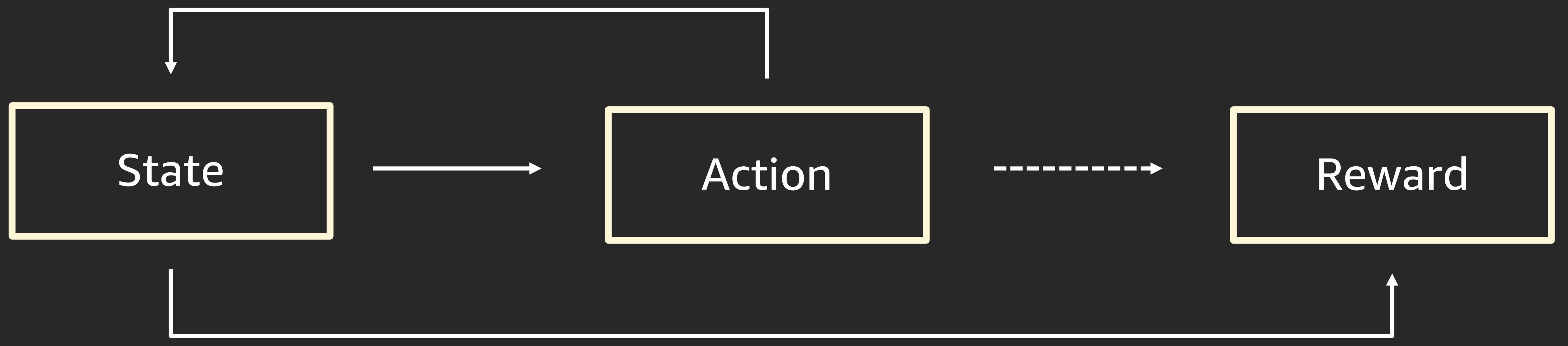
- Very useful when you don’t have supervised training data

- Agent learns by interacting with the environment (simulated or real)

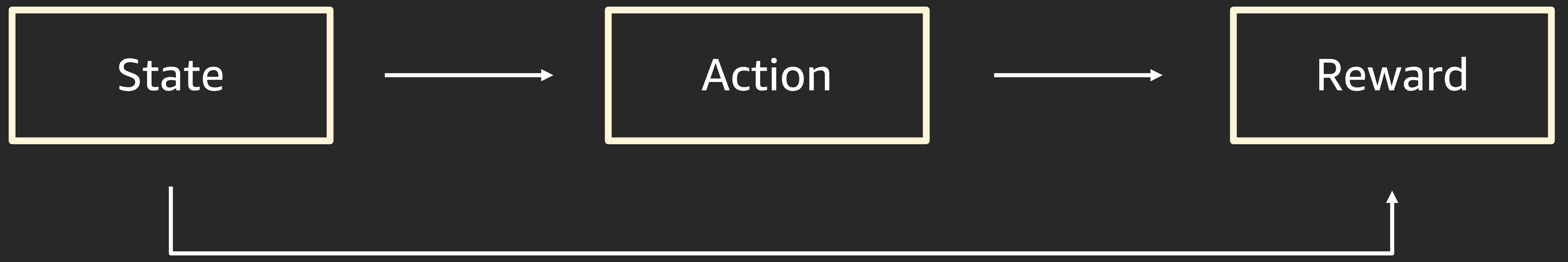
# RL is applicable in many domains

| | | | | | | |
|---|---|---|---|---|---|---|
| Simulation driven | Robotics | Industrial control | HVAC | Autonomous vehicles | Games | Virtual reality |
| Data driven | Dialog systems | Supply chain | Finance | Automatic scaling | Online content delivery | Push notifications |

# First step toward RL: Contextual multi-armed bandits



***Full RL problem***



***Contextual MAB***

# An important trade-off: Explore vs. exploit

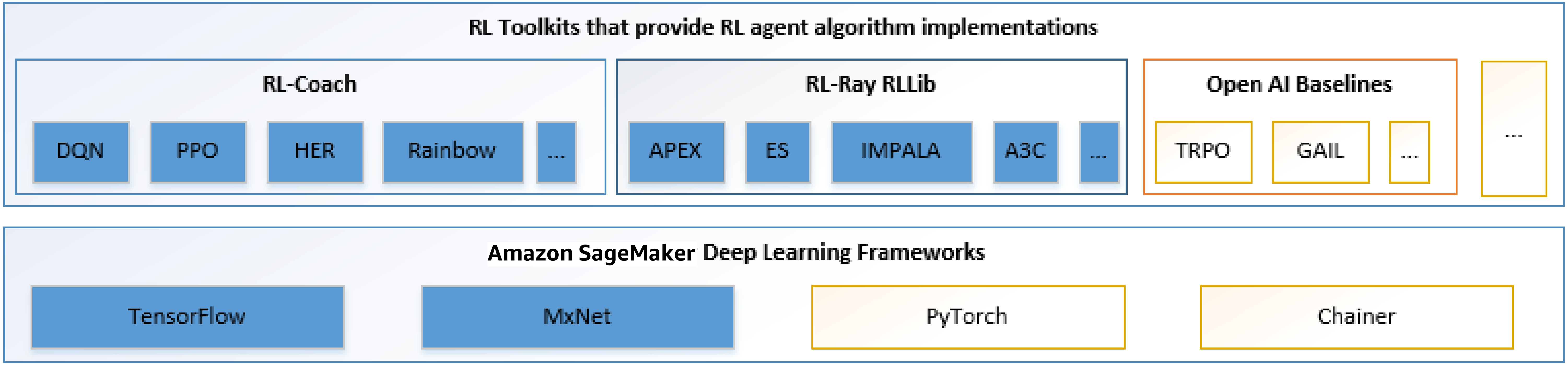
You need to have a balance between exploration and exploitation

# Amazon SageMaker RL

# Amazon SageMaker RL makes RL accessible

| | | | | |
|---|---|---|---|---|
| Difficult to get started | RL agent algorithms are complex to implement | Hard to integrate environments for training | Training is computationally expensive and time-consuming | Requires trial and error & frequent tuning of hyperparameters |
| Pre-built environments for RL; numerous examples | Support for RL agent algorithms | Easy to integrate variety of simulation environments | Single/distributed training; local/remote environment | Local mode for debugging; automatic model tuning |

# Train RL models using state-of-the-art algorithms



RL Toolkits that provide RL agent algorithm implementations

RL-Coach: DQN, PPO, HER, Rainbow, …

RL-Ray RLLib: APEX, ES, IMPALA, A3C, …

Open AI Baselines: TRPO, GAIL, …

…

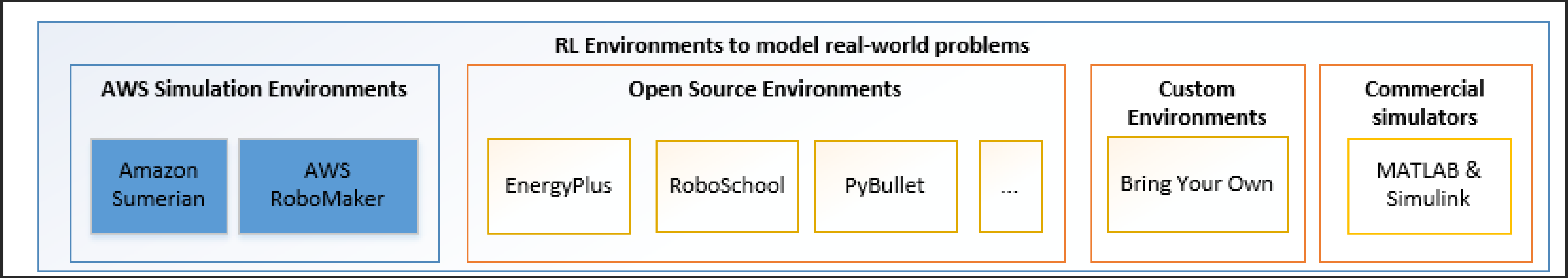
Amazon SageMaker Deep Learning Frameworks

TensorFlow, MxNet, PyTorch, Chainer

🟦 Amazon SageMaker supported □ Customer BYO

* RL Toolkits comparison

# Integrate any type of RL environment



RL Environments to model real-world problems

AWS Simulation Environments

Amazon Sumerian

AWS RoboMaker

Open Source Environments

EnergyPlus

RoboSchool

PyBullet

...

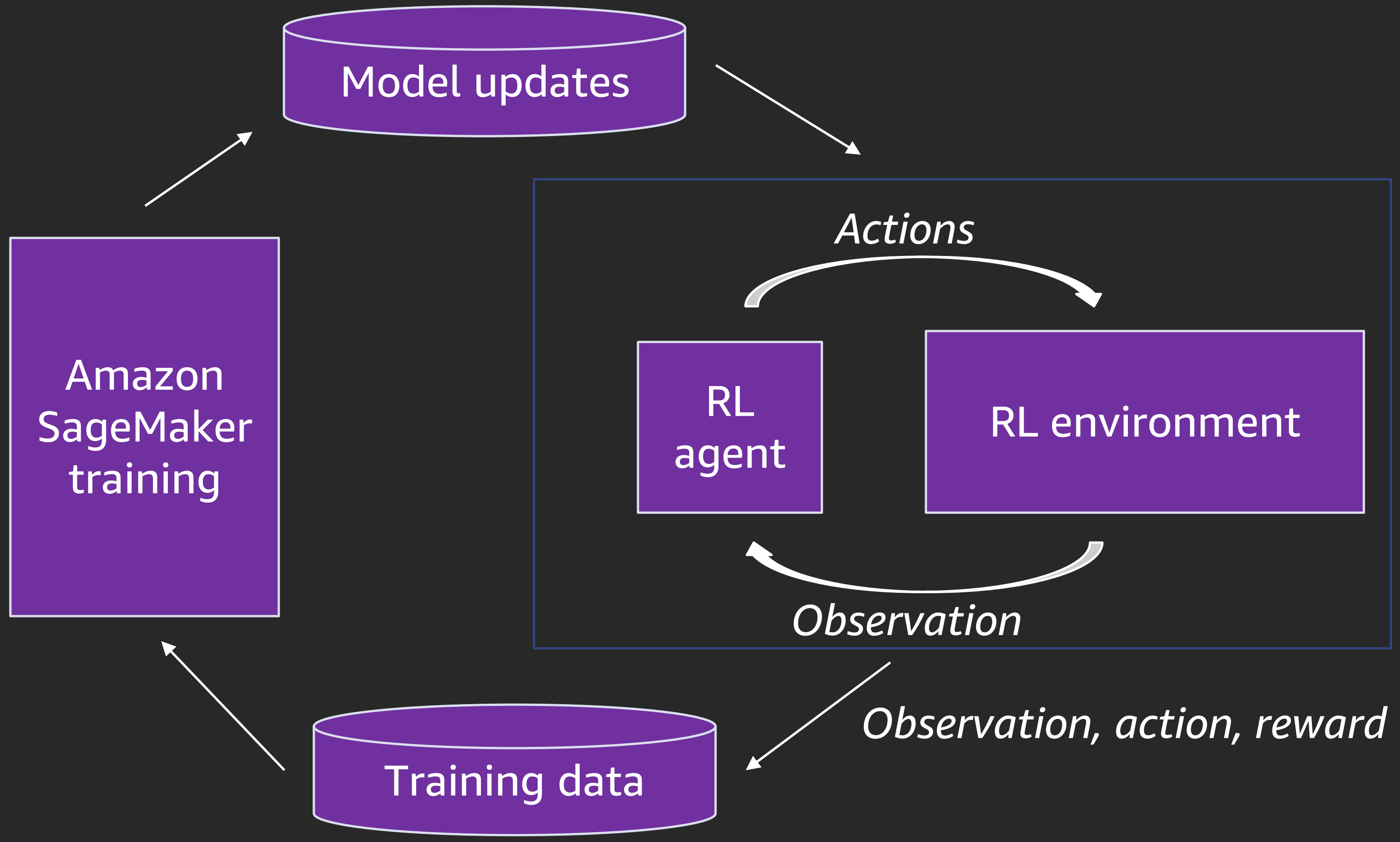
Custom Environments

Bring Your Own

Commercial simulators

MATLAB & Simulink

# Amazon SageMaker RL



# Workshop #1: Training without a simulator in a real environment

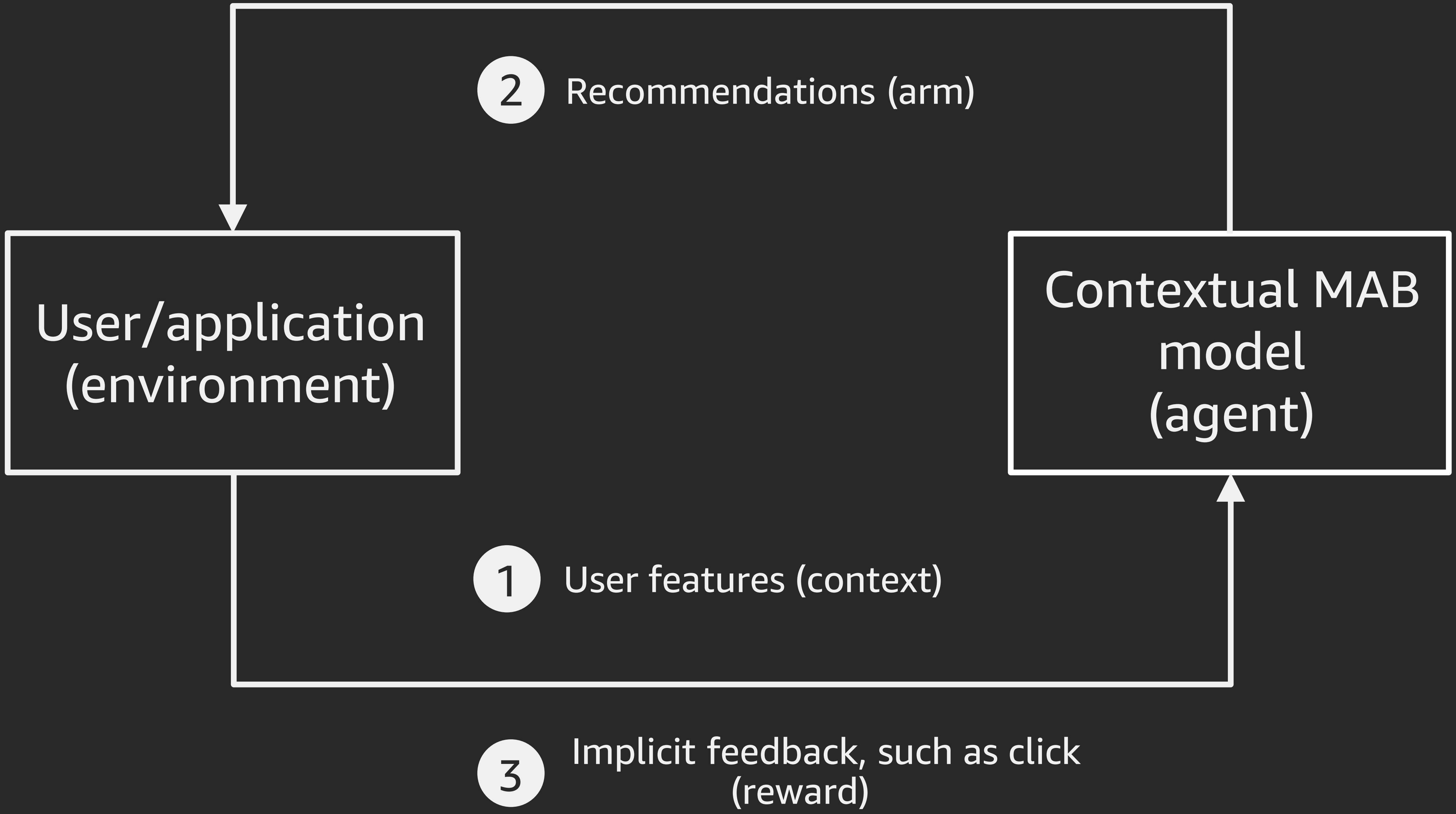
AWS re:Invent

aws

# What are the challenges

- Feedback is delayed. It needs to be joined with inputs & actions taken to prepare next training datasets.
- You have to learn fast. Unlike in a simulated environment, the agent doesn’t have the luxury to learn from millions of episodes.
- Training never stops

# Building a recommendation with contextual MAB



User/application (environment)

Contextual MAB model (agent)

2 Recommendations (arm)

1 User features (context)

3 Implicit feedback, such as click (reward)

# Bandits vs. A/B testing

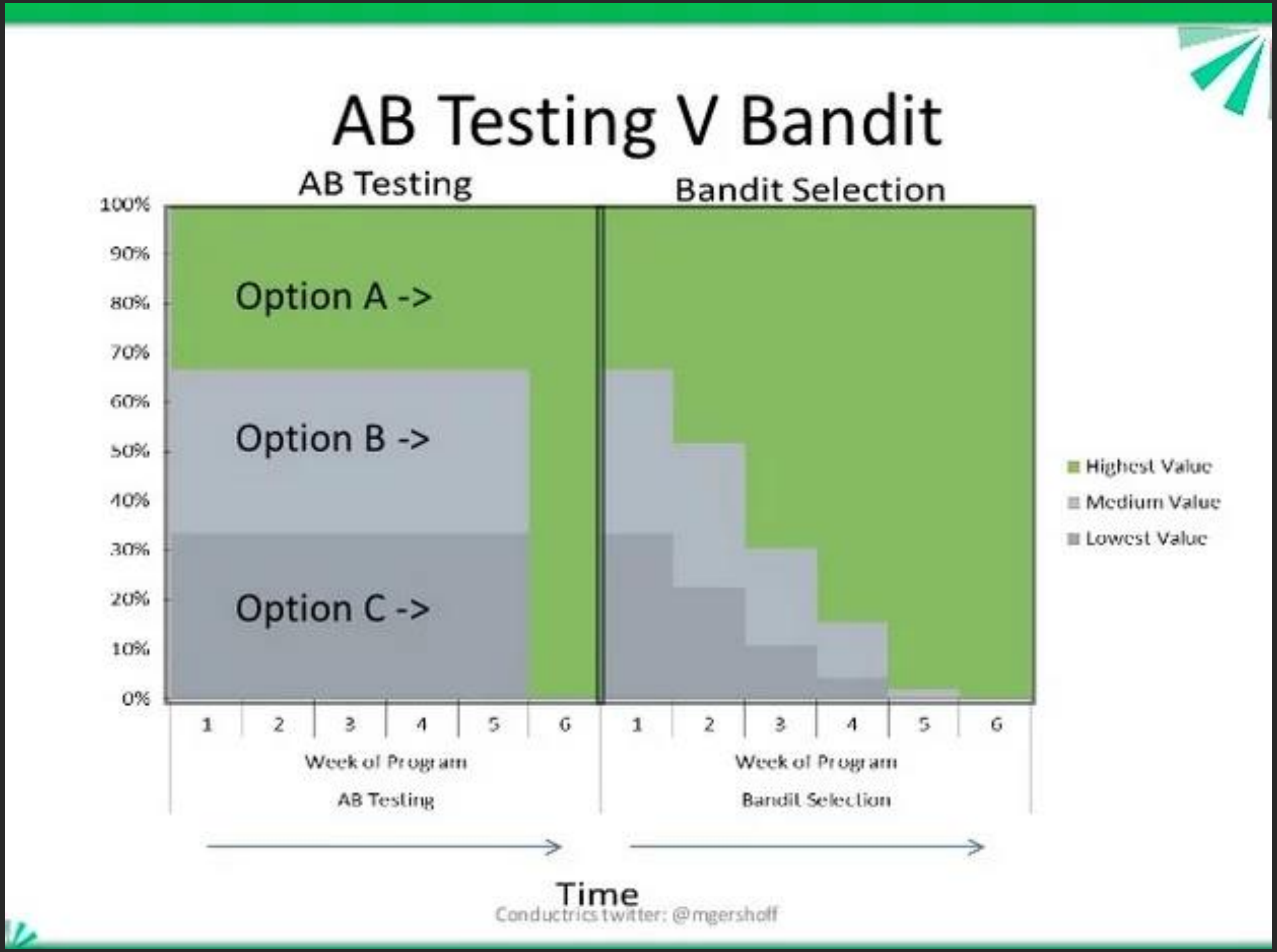
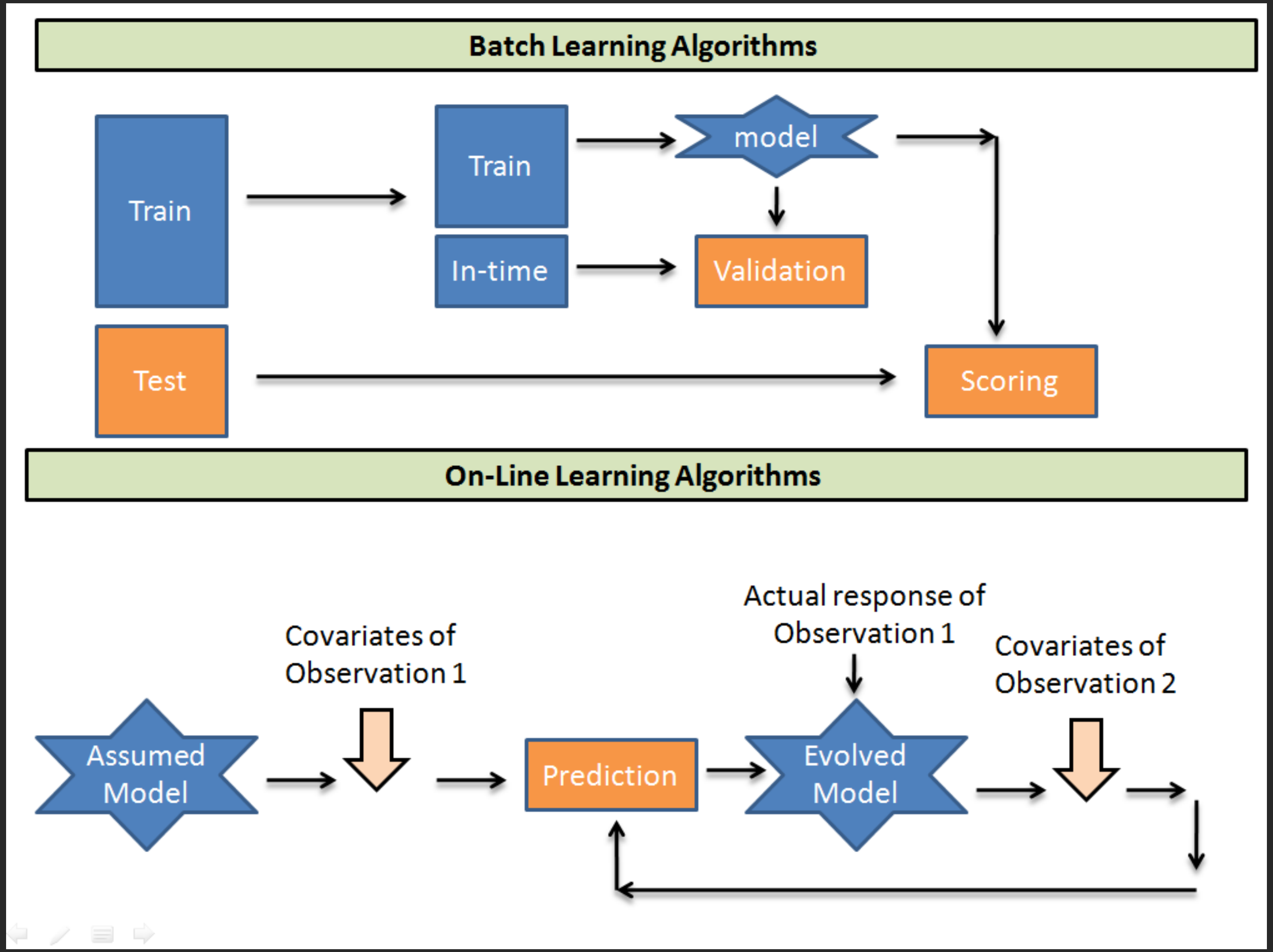


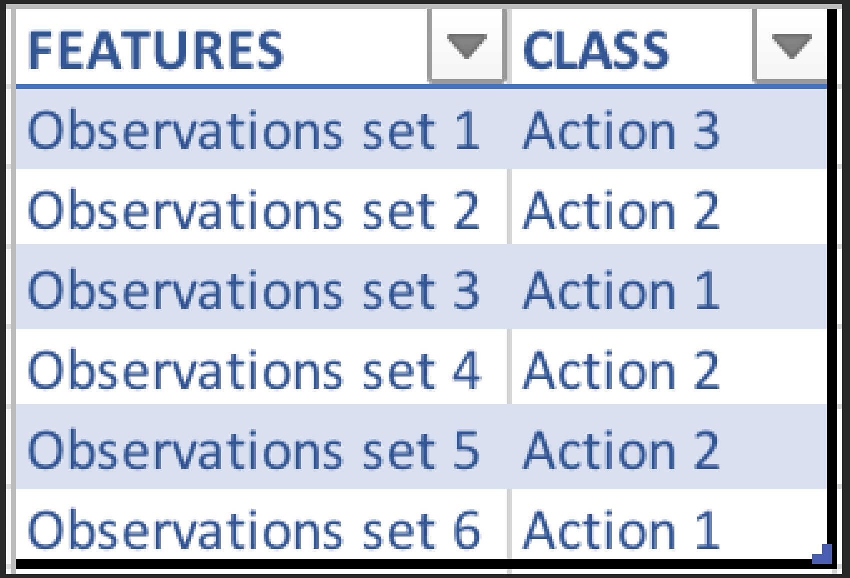
*Image source (courtesy of Matt Gershoff)*

# Online learning

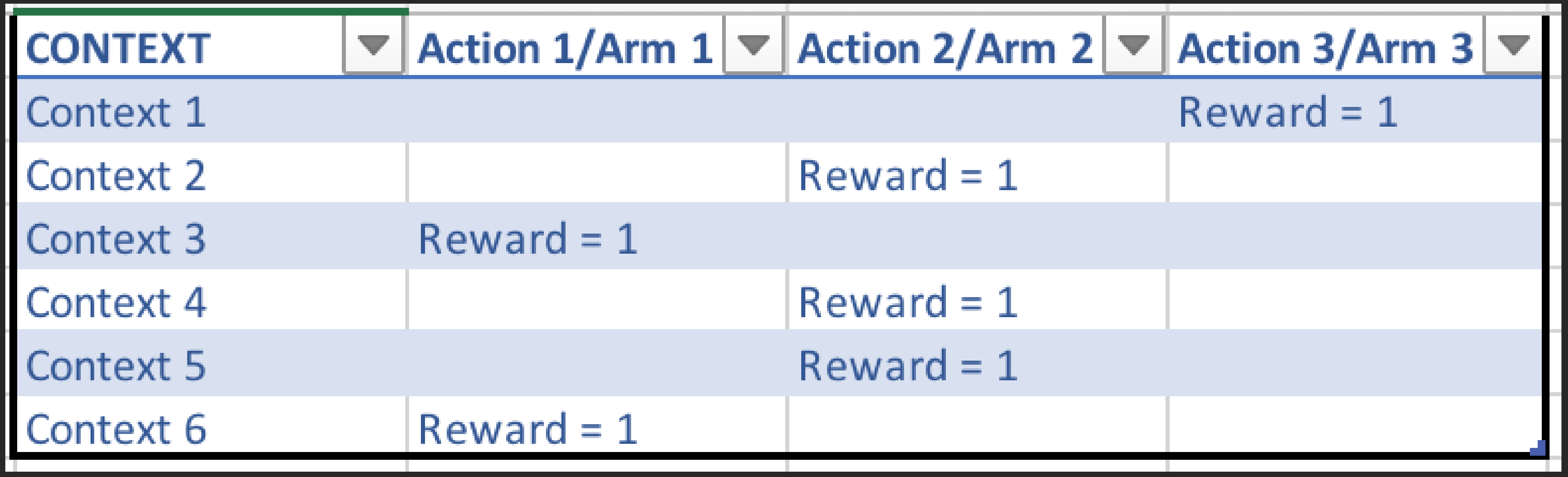

Batch Learning Algorithms
Train
Train
model
In-time
Validation
Test
Scoring
On-Line Learning Algorithms
Covariates of Observation 1
Actual response of Observation 1
Covariates of Observation 2
Assumed Model
Prediction
Evolved Model

Introduction to online machine learning simplified
(source https://analyticsvidhya.com)

# Adopting bandits into existing systems: Warm start



| FEATURES | CLASS |
|---|---|
| Observations set 1 | Action 3 |
| Observations set 2 | Action 2 |
| Observations set 3 | Action 1 |
| Observations set 4 | Action 2 |
| Observations set 5 | Action 2 |
| Observations set 6 | Action 1 |

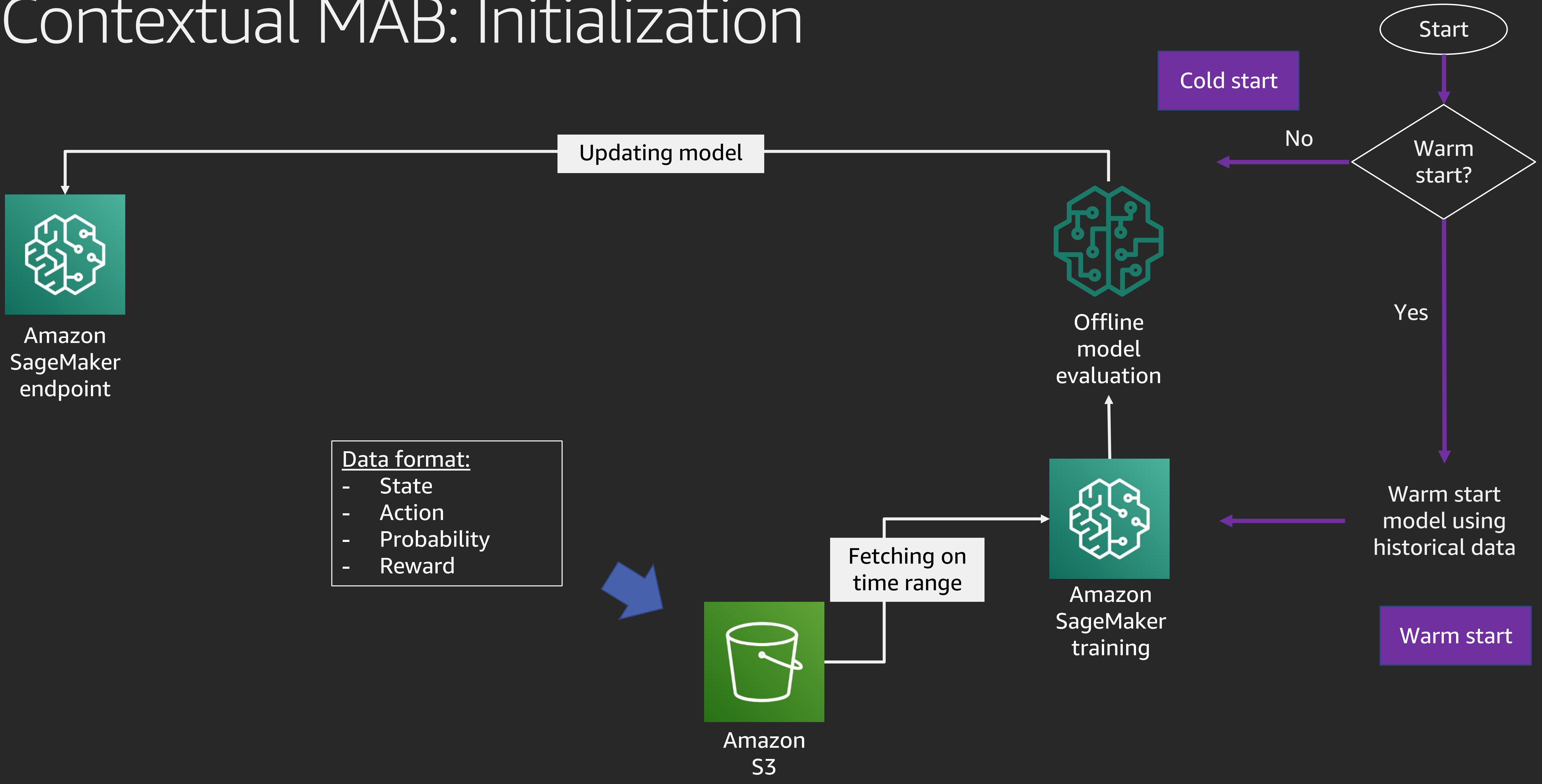


| CONTEXT | Action 1/Arm 1 | Action 2/Arm 2 | Action 3/Arm 3 |
|---|---|---|---|
| Context 1 | | | Reward = 1 |
| Context 2 | | Reward = 1 | |
| Context 3 | Reward = 1 | | |
| Context 4 | | Reward = 1 | |
| Context 5 | | Reward = 1 | |
| Context 6 | Reward = 1 | | |

# Amazon SageMaker RL bandits container

VOWPAL WABBIT

# Contextual MAB: Initialization



Start

Warm start?

No

Cold start

Yes

Warm start model using historical data

Warm start

Amazon SageMaker training

Fetching on time range

Amazon S3

Data format:

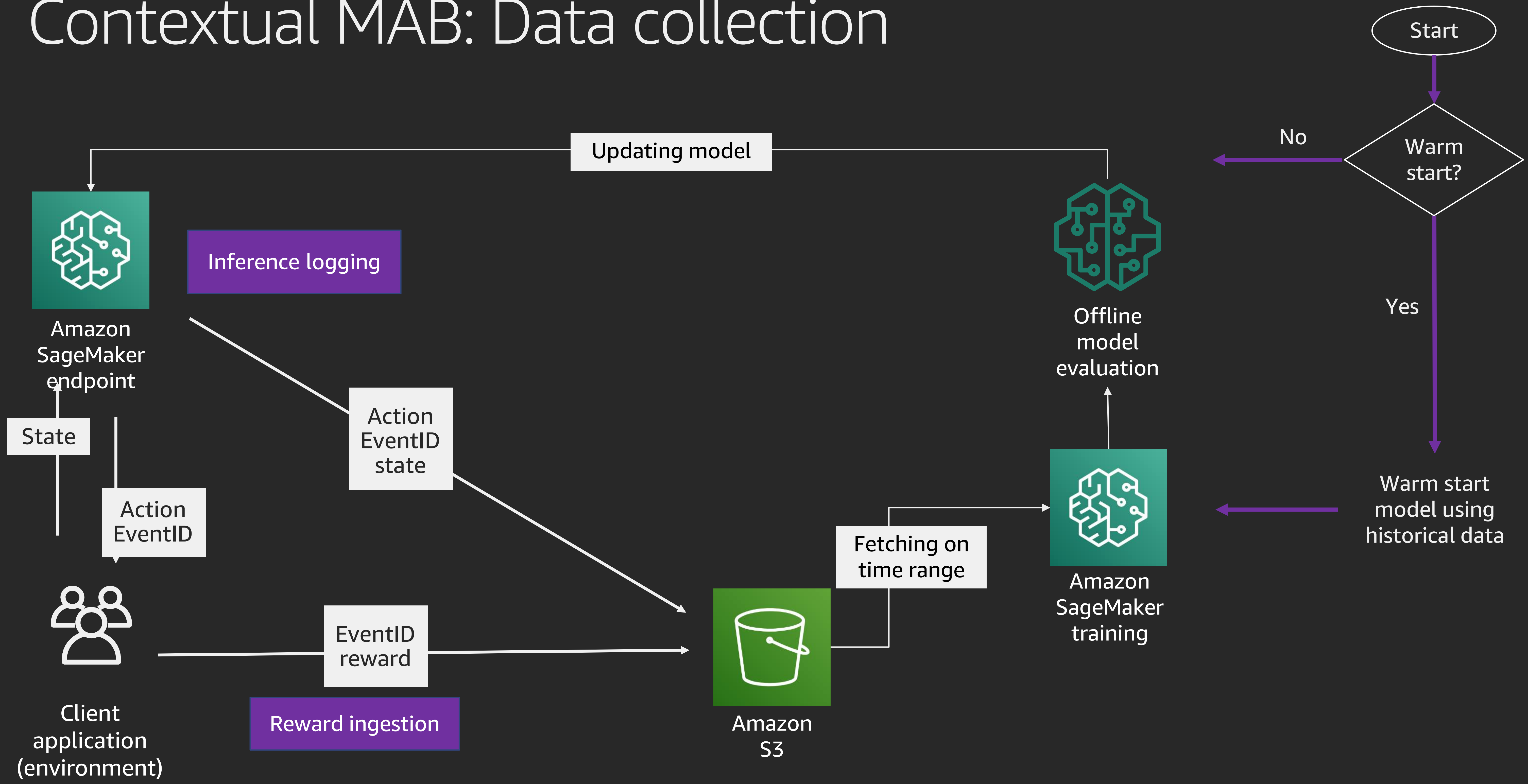
- State
- Action
- Probability
- Reward

Offline model evaluation

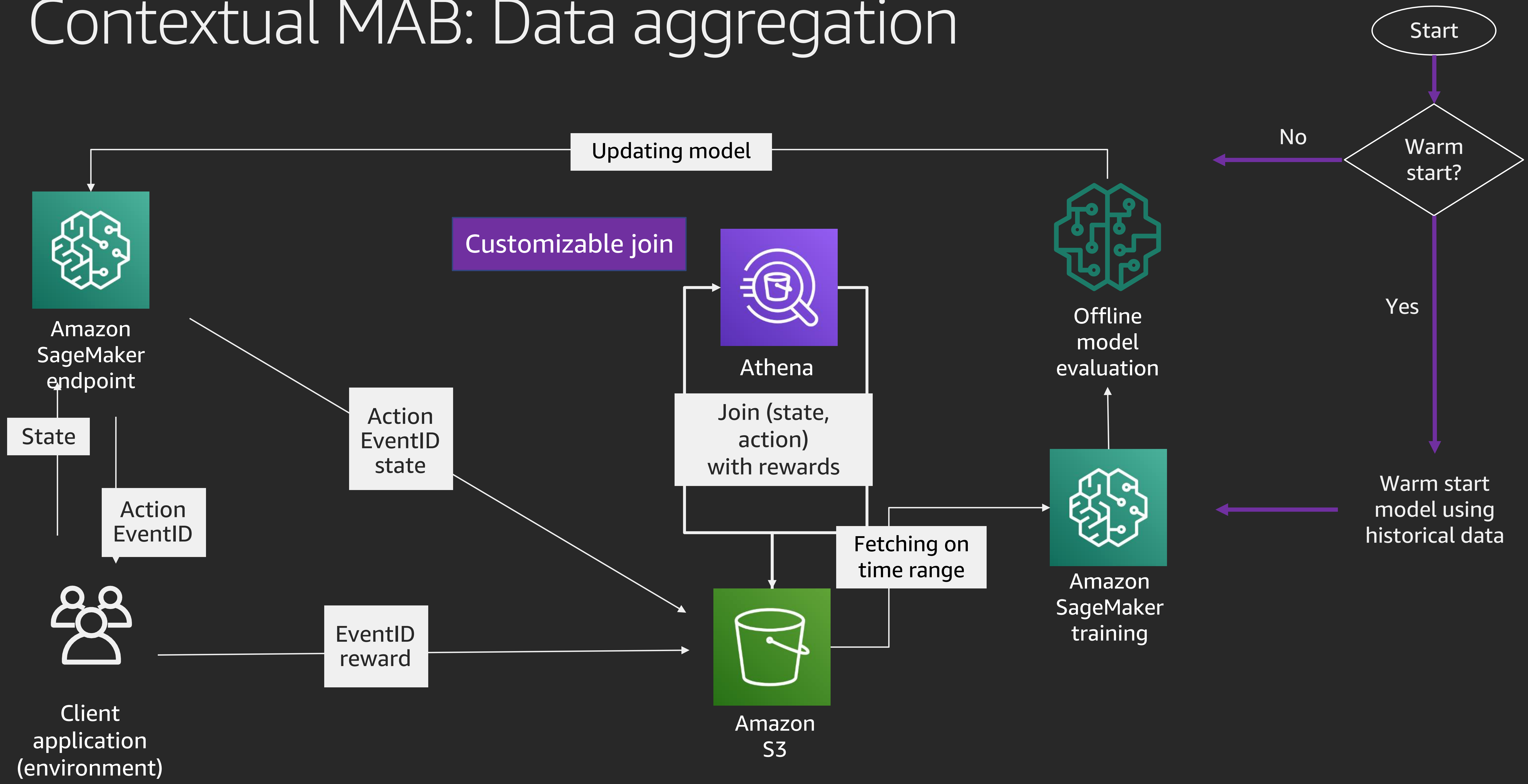
Updating model

Amazon SageMaker endpoint

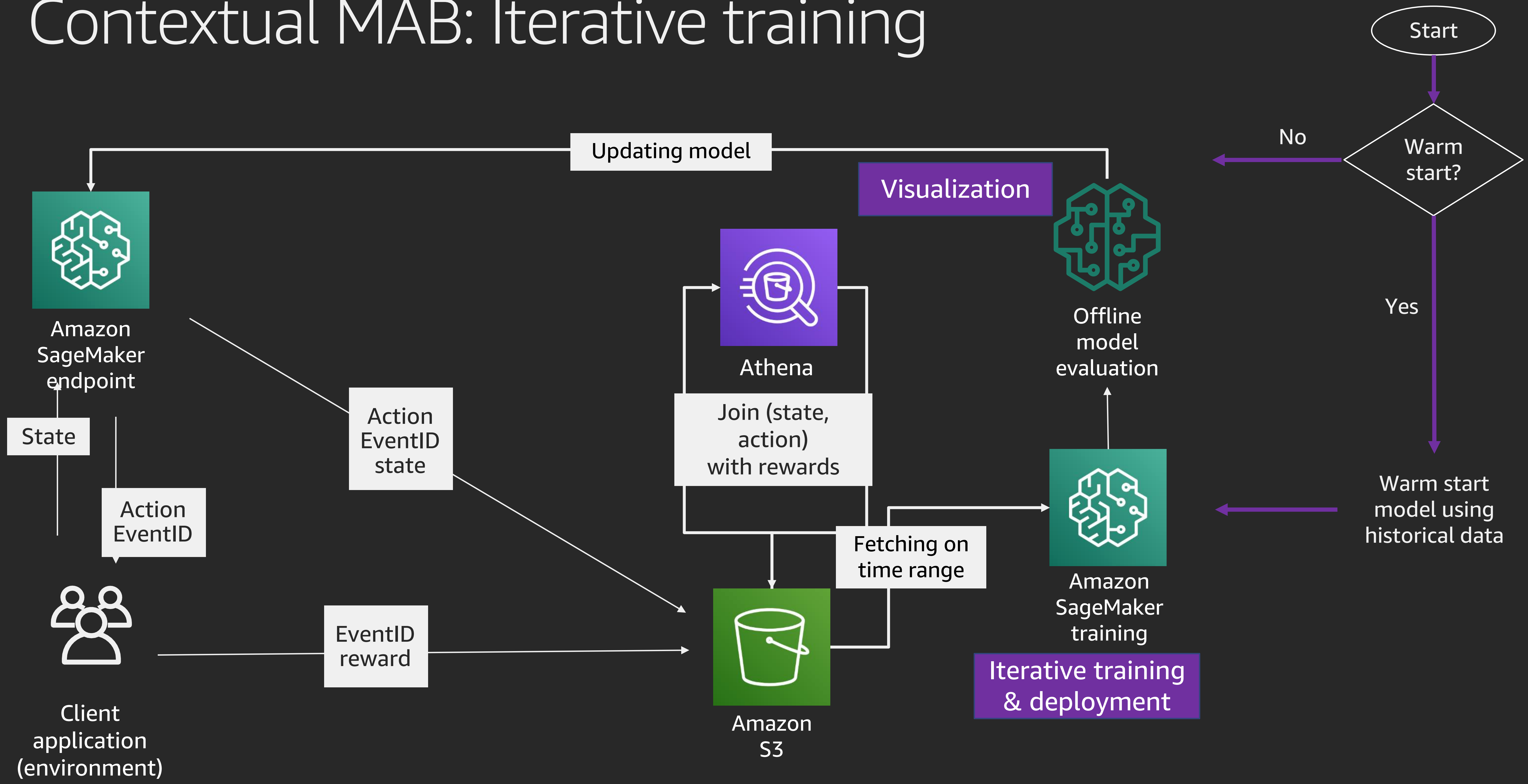
# Contextual MAB: Data collection



# Contextual MAB: Data aggregation

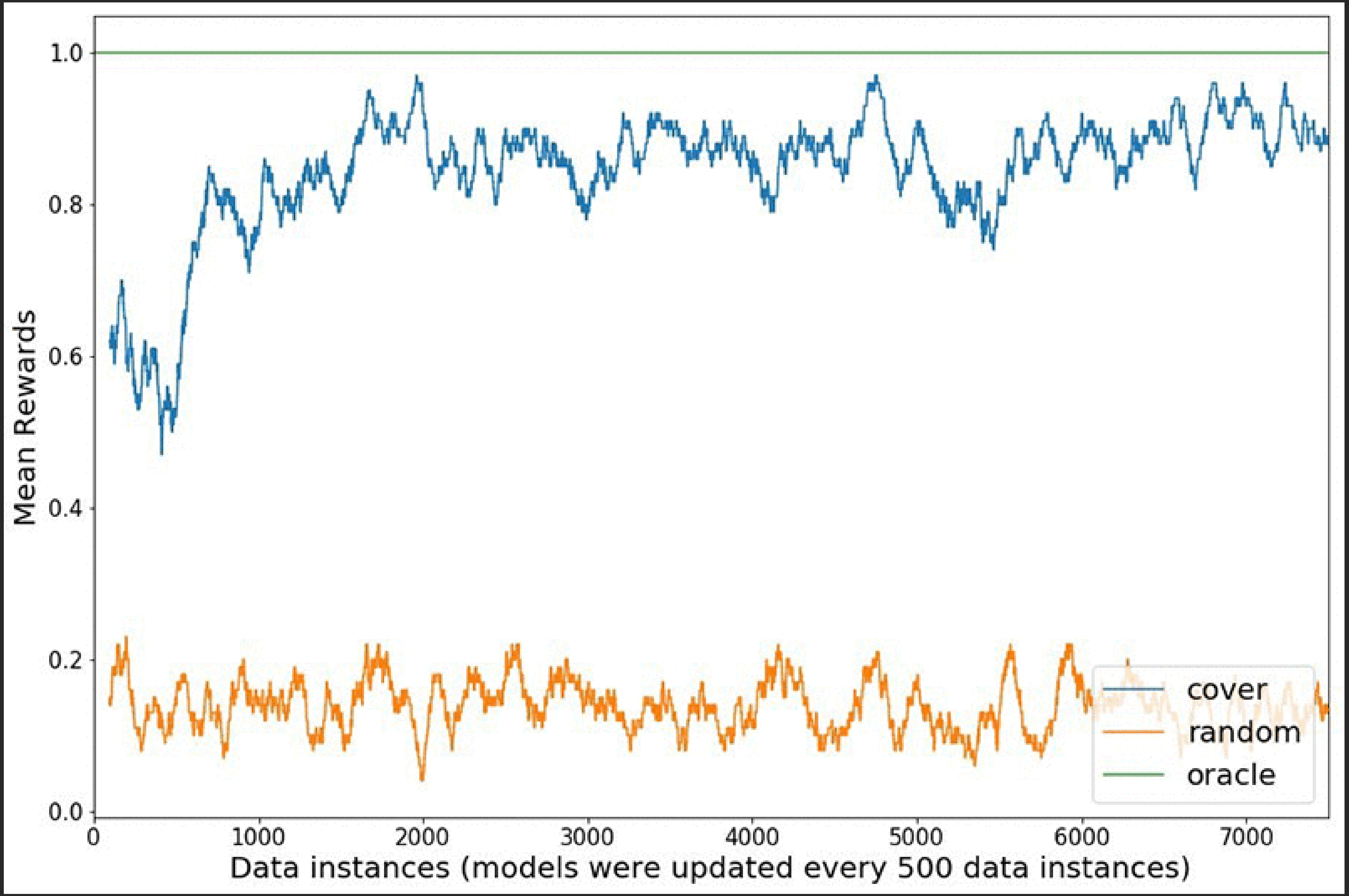


# Contextual MAB: Iterative training

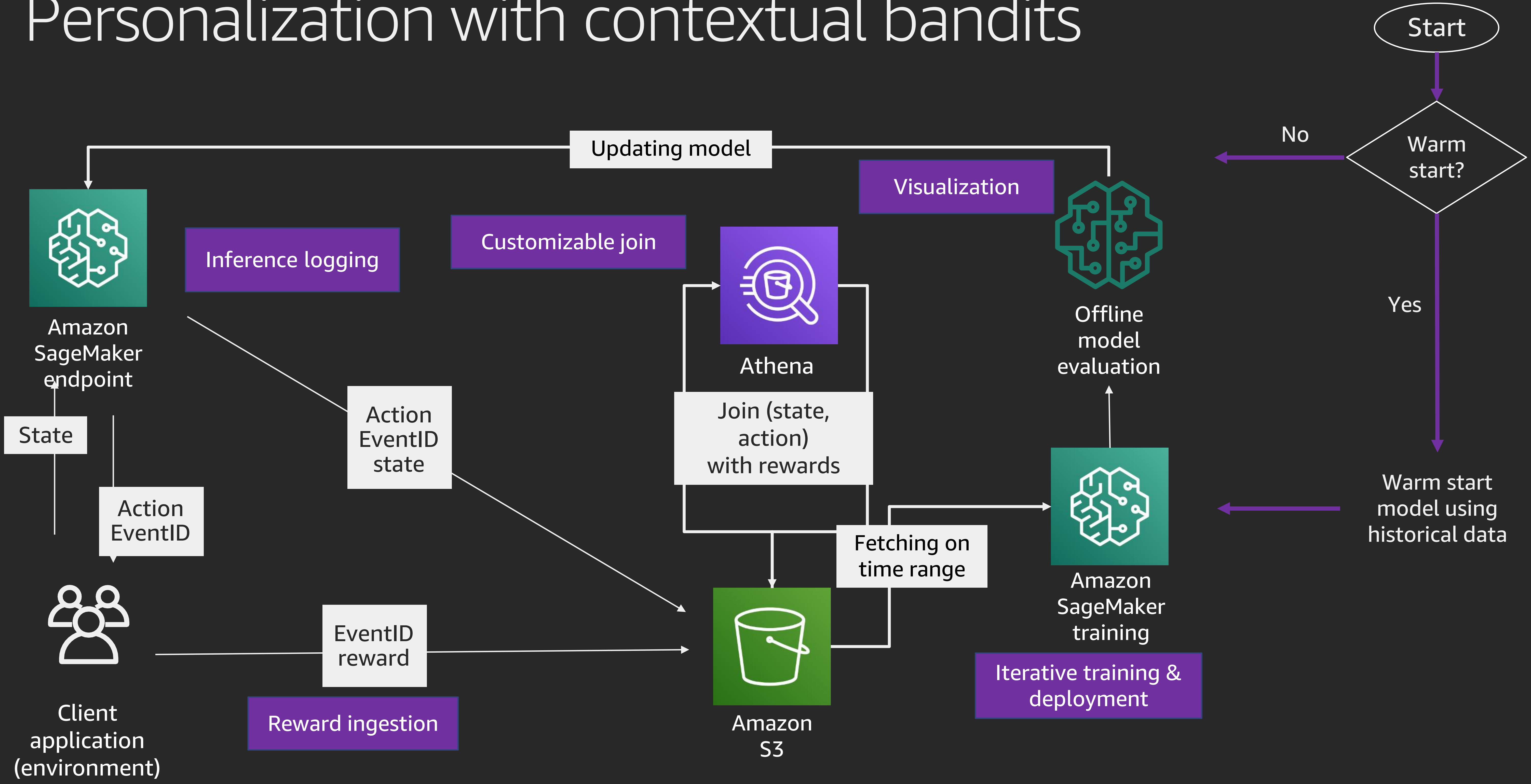


Start
Warm start?
No
Yes
Warm start model using historical data
Updating model
Visualization
Offline model evaluation
Amazon SageMaker endpoint
State
Action EventID
Action EventID state
Athena
Join (state, action) with rewards
Fetching on time range
Amazon SageMaker training
Iterative training & deployment
Client application (environment)
EventID reward
Amazon S3

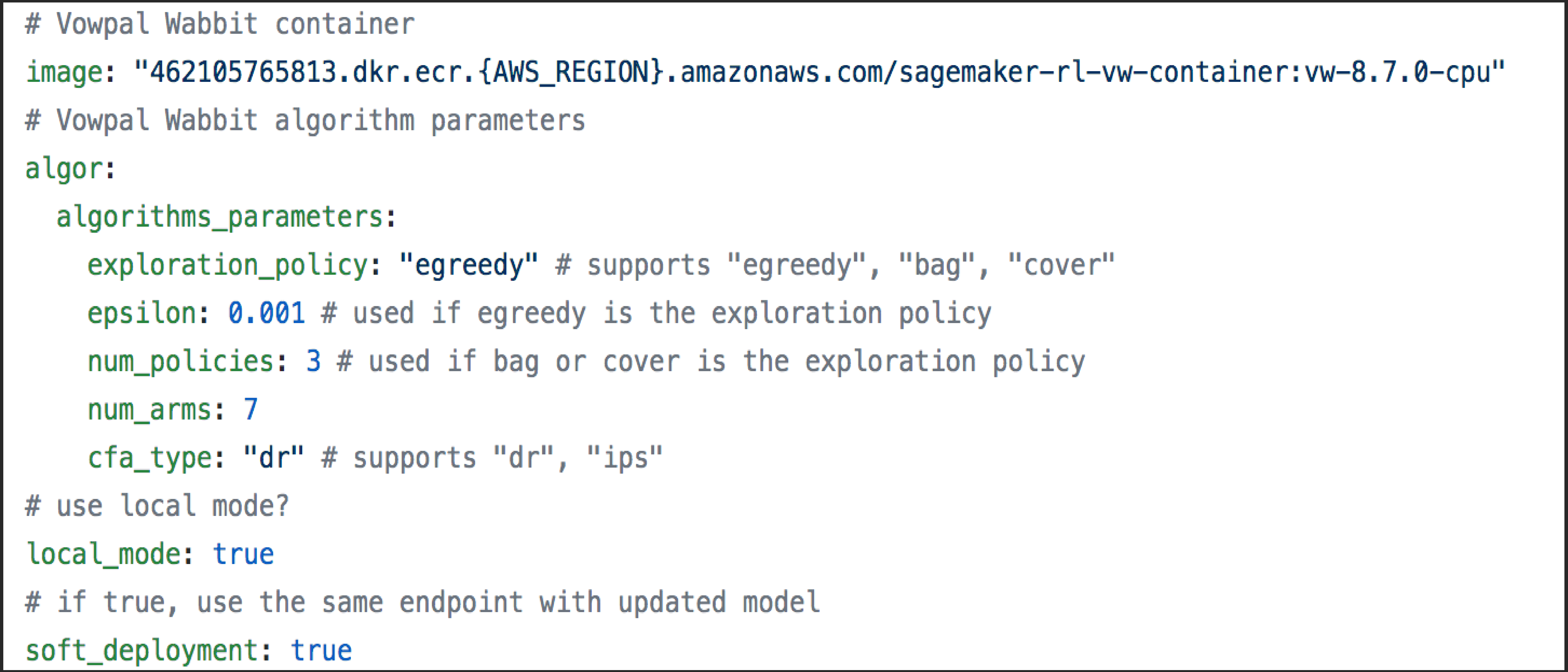
# Contextual MAB: Evaluation



# Personalization with contextual bandits



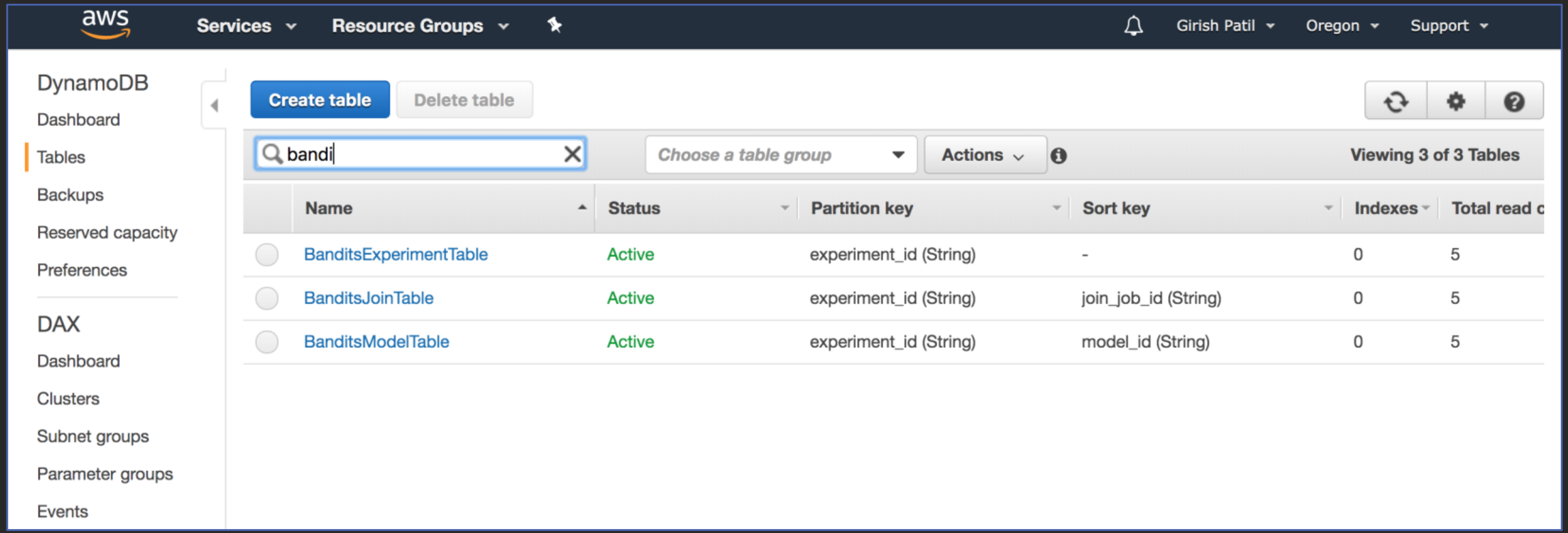
# Configurations



```
# Vowpal Wabbit container
image: "462105765813.dkr.ecr.{AWS_REGION}.amazonaws.com/sagemaker-rl-vw-container:vw-8.7.0-cpu"
# Vowpal Wabbit algorithm parameters
algor:
  algorithms_parameters:
    exploration_policy: "egreedy" # supports "egreedy", "bag", "cover"
    epsilon: 0.001 # used if egreedy is the exploration policy
    num_policies: 3 # used if bag or cover is the exploration policy
    num_arms: 7
    cfa_type: "dr" # supports "dr", "ips"
# use local mode?
local_mode: true
# if true, use the same endpoint with updated model
soft_deployment: true
```

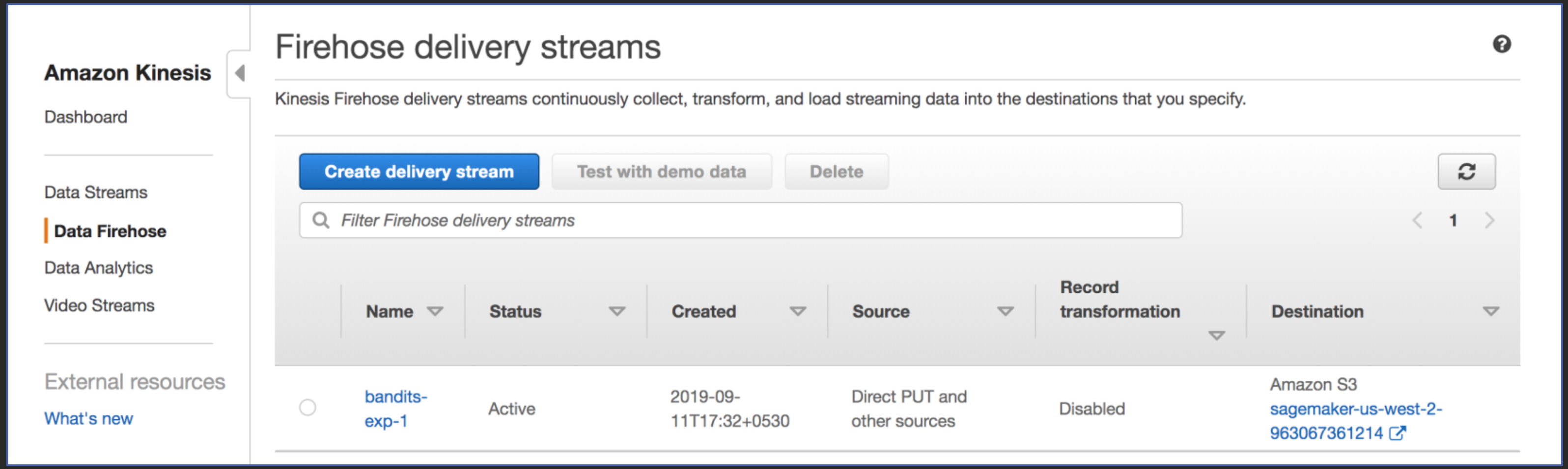
# Reviewing the setup

## Amazon DynamoDB



| Name | Status | Partition key | Sort key | Indexes | Total read c |
|---|---|---|---|---|---|
| BanditsExperimentTable | Active | experiment_id (String) | - | 0 | 5 |
| BanditsJoinTable | Active | experiment_id (String) | join_job_id (String) | 0 | 5 |
| BanditsModelTable | Active | experiment_id (String) | model_id (String) | 0 | 5 |

## Amazon Kinesis Data Firehose



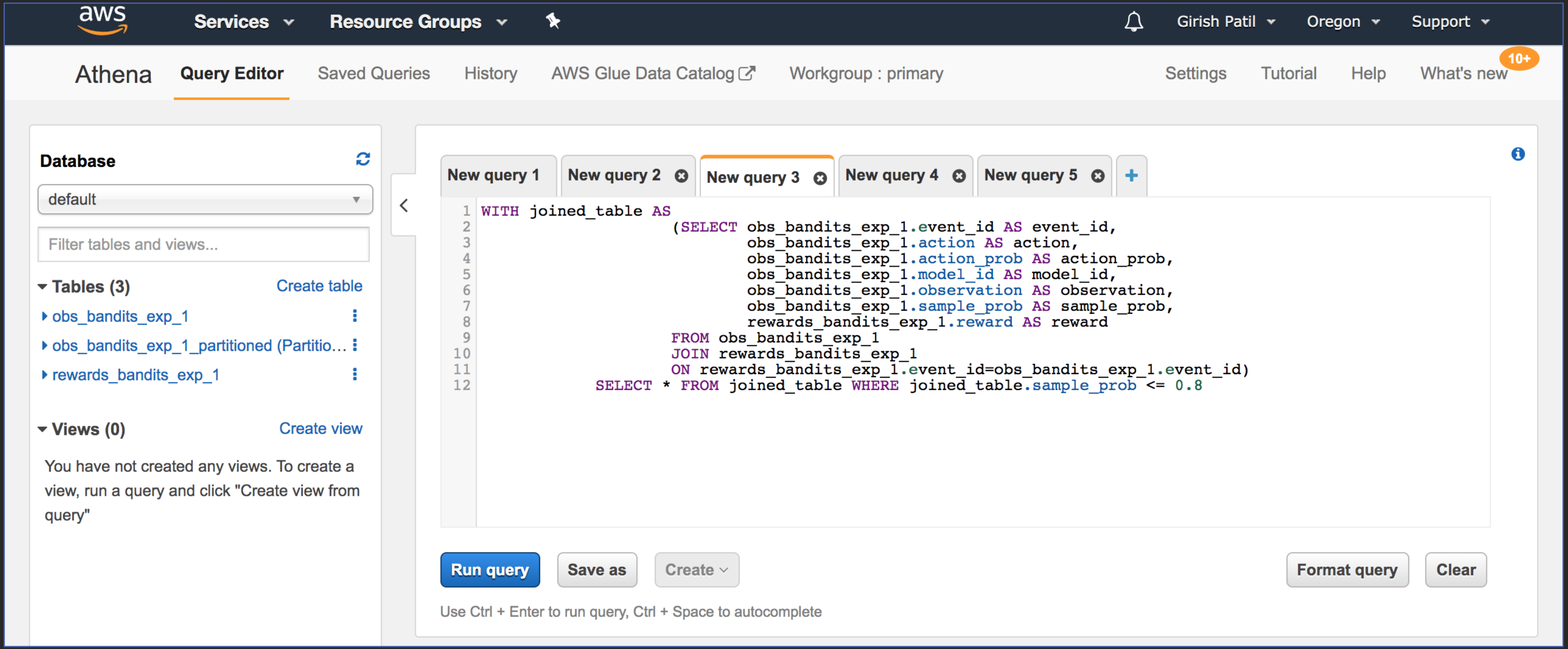
Firehose delivery streams

Kinesis Firehose delivery streams continuously collect, transform, and load streaming data into the destinations that you specify.

| Name | Status | Created | Source | Record transformation | Destination |
|---|---|---|---|---|---|
| bandits-exp-1 | Active | 2019-09-11T17:32+0530 | Direct PUT and other sources | Disabled | Amazon S3<br>sagemaker-us-west-2-963067361214 |

# Reviewing the setup, continued

## Amazon Athena



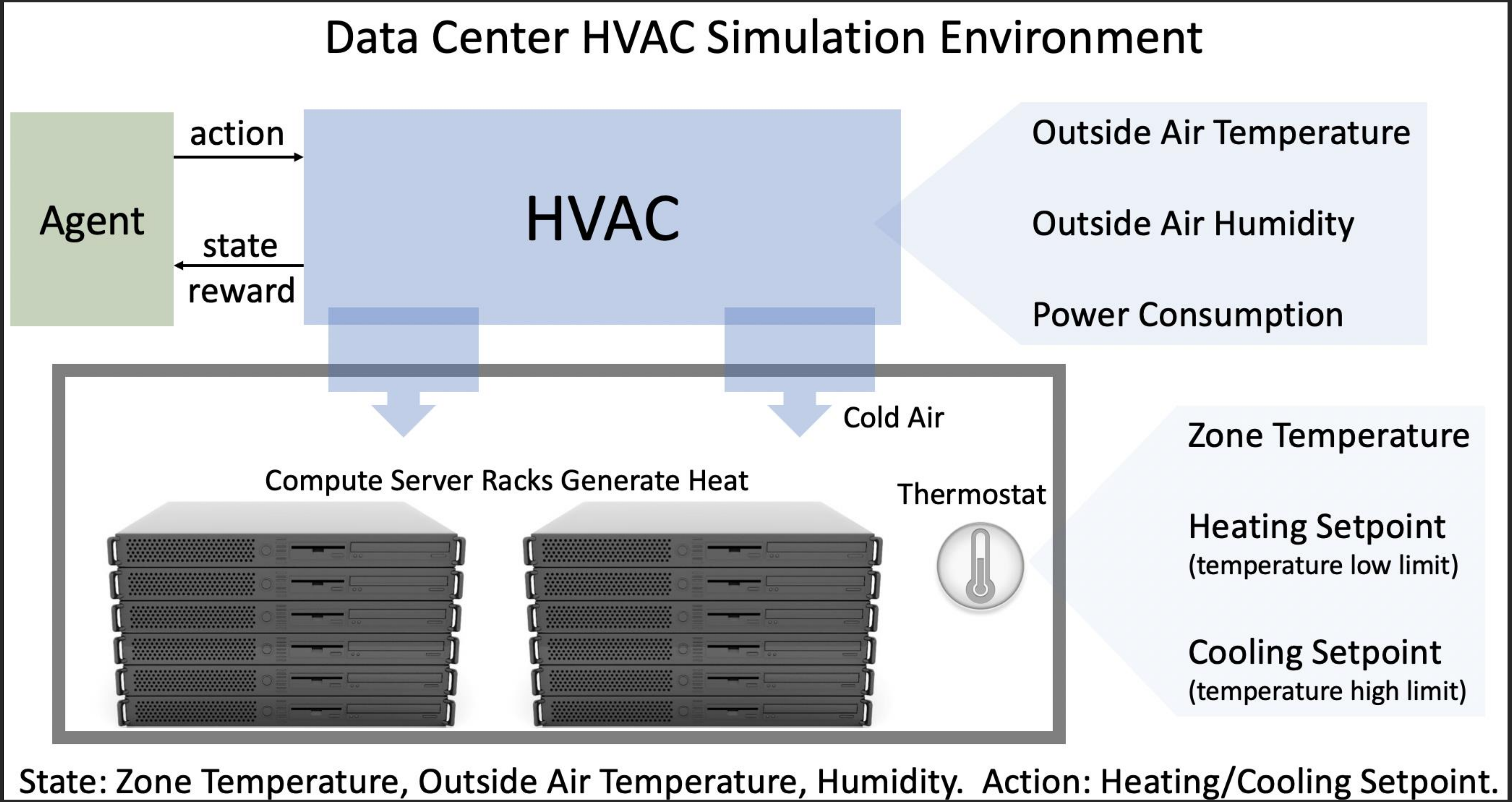
You have not created any views. To create a view, run a query and click "Create view from query"

```
WITH joined_table AS
                    (SELECT obs_bandits_exp_1.event_id AS event_id,
                            obs_bandits_exp_1.action AS action,
                            obs_bandits_exp_1.action_prob AS action_prob,
                            obs_bandits_exp_1.model_id AS model_id,
                            obs_bandits_exp_1.observation AS observation,
                            obs_bandits_exp_1.sample_prob AS sample_prob,
                            rewards_bandits_exp_1.reward AS reward
                    FROM obs_bandits_exp_1
                    JOIN rewards_bandits_exp_1
                    ON rewards_bandits_exp_1.event_id=obs_bandits_exp_1.event_id)
            SELECT * FROM joined_table WHERE joined_table.sample_prob <= 0.8
```

Use Ctrl + Enter to run query, Ctrl + Space to autocomplete

# Workshop #2: Training with a simulator

# Training with HVAC simulator



State: Zone Temperature, Outside Air Temperature, Humidity. Action: Heating/Cooling Setpoint.

LINK

# Conclusion

Amazon SageMaker (working with other AWS services) makes it equally easy to train with and without simulation environments.

Amazon SageMaker provides containers with popular RL algorithms, and you can bring your own. This includes online learning algorithms.

Contextual bandits make experimentation very effective, and they learn rapidly.

# Conquer the newest frontier of ML: Reinforcement learning with Amazon SageMaker

# Learn ML with AWS Training and Certification

The same training that our own developers use, now available on demand

Role-based ML learning paths for developers, data scientists, data platform engineers, and business decision makers

70+ free digital ML courses from AWS experts let you learn from real-world challenges tackled at AWS

Validate expertise with the **AWS Certified Machine Learning - Specialty** exam

Visit https://aws.training/machinelearning

# Thank you!

**Girish Dilip Patil**
girpatil@amazon.com

**Marc Cabocel**
cabocel@amazon.fr

**Segolene Dessertine-panhard**

**Ann Luo**

Please complete the session survey in the mobile app.

AWS re:Invent

aws