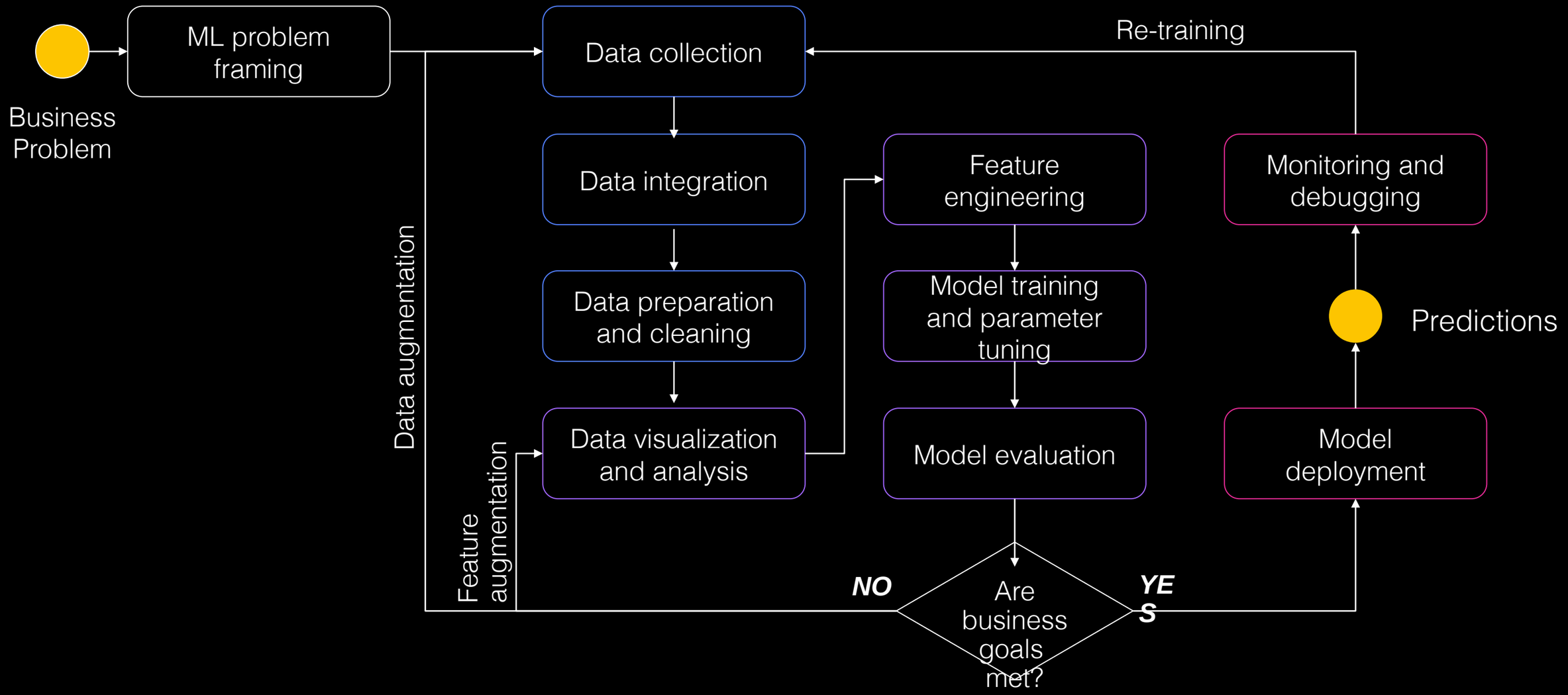
Floor 28, Tel Aviv, July 8th, 2019

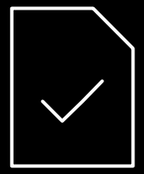
Automate your Amazon SageMaker workflows

Julien Simon
Global Evangelist, AI & Machine Learning, AWS
@julsimon

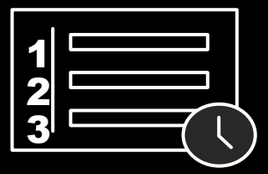
Machine learning cycle



Amazon SageMaker

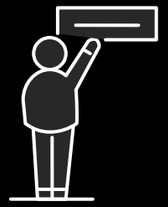


Collect and prepare training data



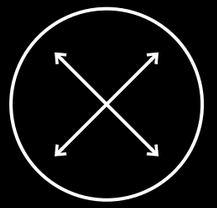
Choose and optimize your ML algorithm

Set up and manage environments for training



Train and Tune ML Models

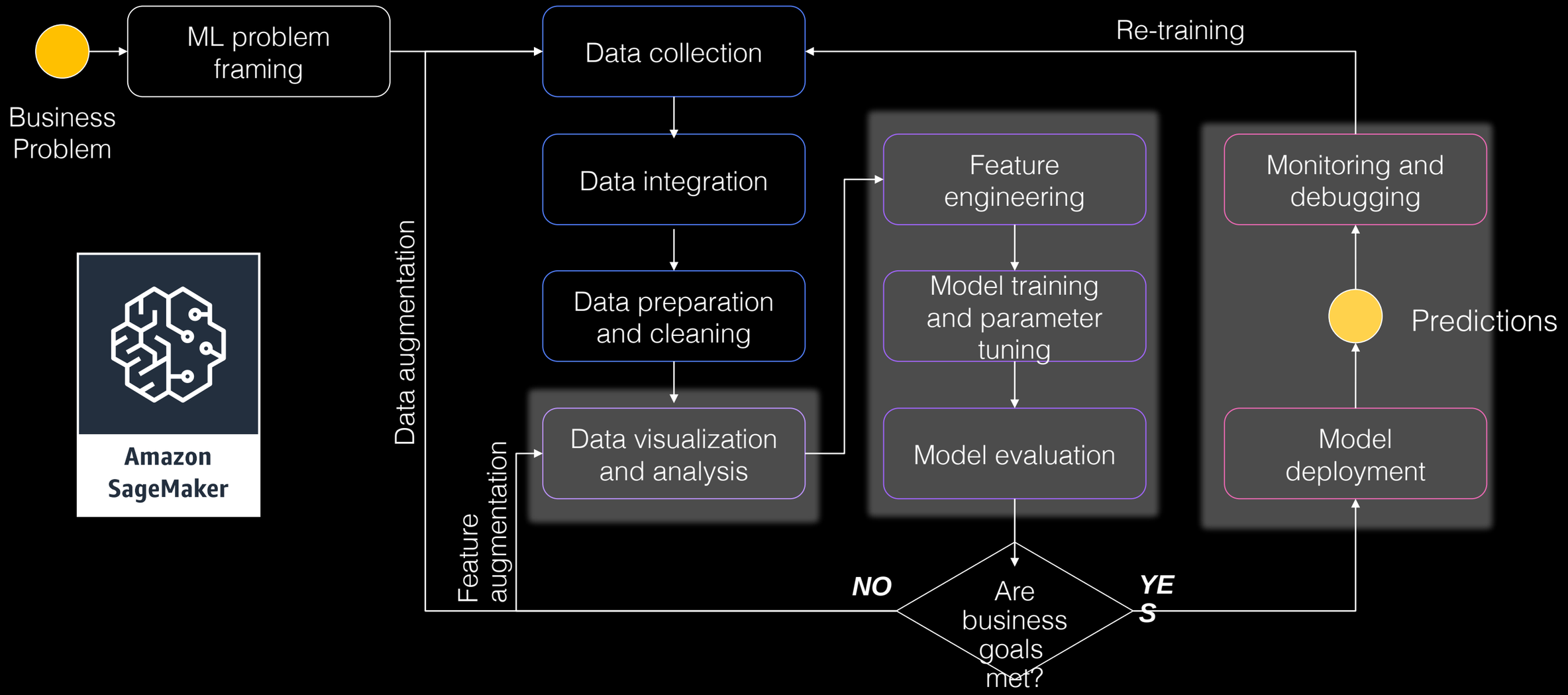
Deploy models in production



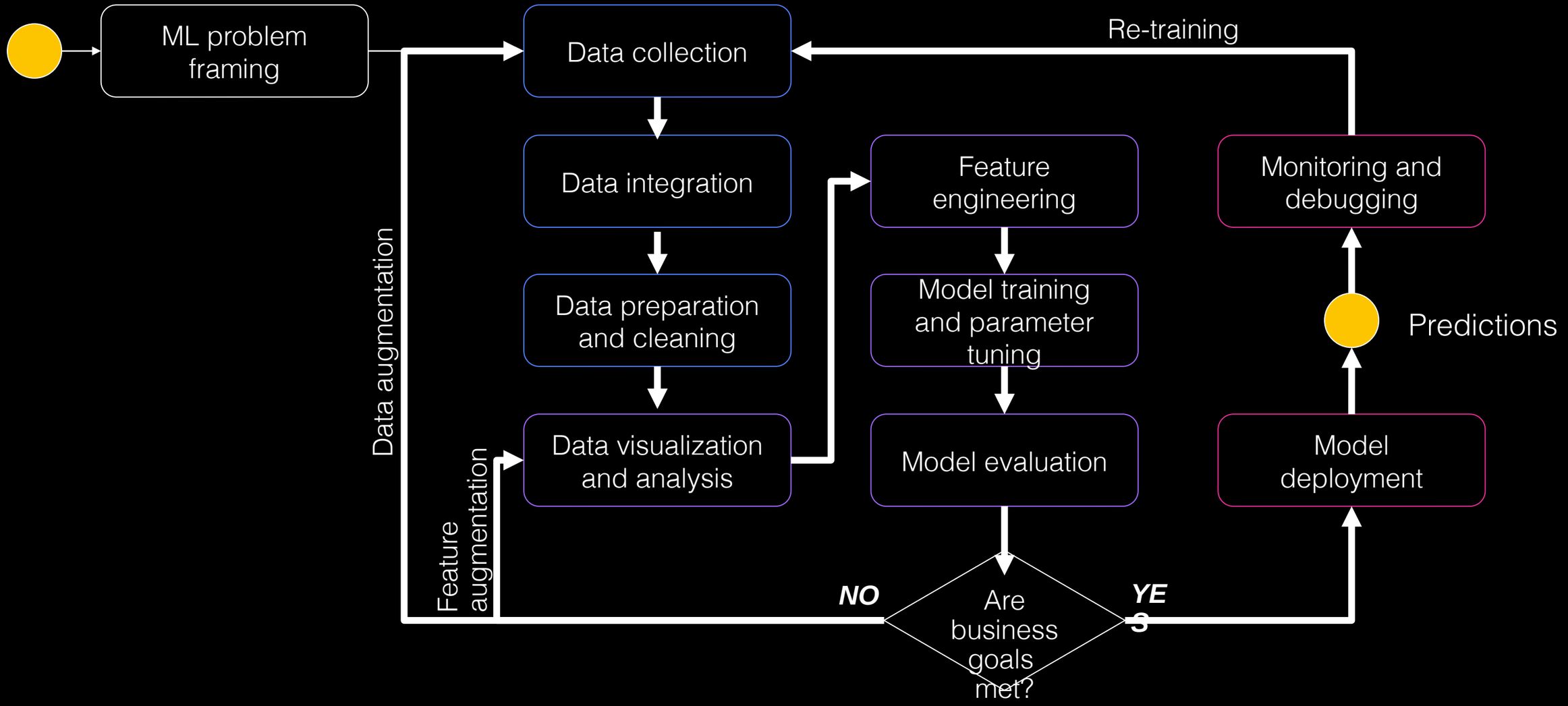
Scale and manage the production environment

Same service and APIs from experimentation to production

Build, train and deploy models using SageMaker



What about the lines between these steps?



Agenda

A recap on AWS automation

- Amazon SDKs
- AWS CloudFormation
- AWS Cloud Development Kit (CDK)

Orchestrating with AWS Step Functions

Creating notebook instances

Training / retraining models

Deploying models

- Strategies: canary deployment, blue-green deployment
- Production variants

1- AWS tools and services for automation

Automation tools and services

- Boto3
  - AWS SDK for Python, covering all AWS services
  - https://github.com/boto/boto3
  - Of course, you can use other AWS SDKs for C++, Go, Java, PHP, Ruby, JS, and .NET
- Amazon SageMaker SDK
  - High-level SDK focused on ML experimentation
  - https://github.com/aws/sagemaker-python-sdk
- Amazon CloudFormation
  - Infrastructure as Code service: templates (JSON, YAML) and stacks.
  - https://docs.aws.amazon.com/cloudformation/index.html
- AWS Cloud Development Kit (CDK)
  - The new kid on the block: write code (JS, Python, Java, C#), build a CloudFormation template
  - https://docs.aws.amazon.com/cdk/

Score card for Amazon SageMaker automation

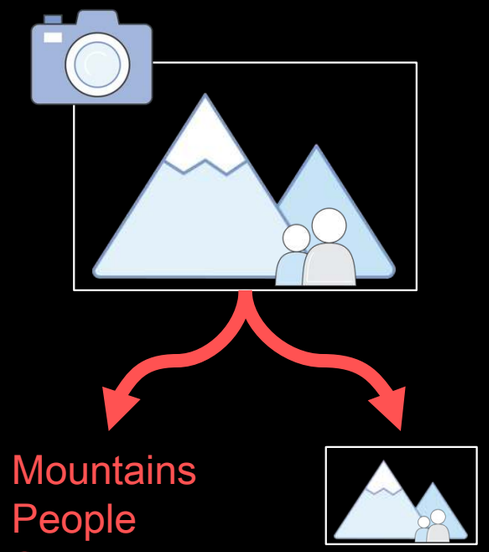
	Boto3	Amazon SageMaker SDK	AWS CloudFormation	AWS CDK
Maturity / stability	***	**	***	* (developer preview)
Documentation / examples	***	***	***	** for Javascript * for other languages
Open source	Yes	Yes	No	Yes
Service coverage	100%	100%, but high-level	Notebook instances Models Endpoints No training	Same as CF
Change management	Source control	Source control	Source control Change sets Drift detection	Same as CF
Cleanup	** (Delete APIs)	** (Delete APIs)	*** (built-in)	Same as CF
Rollback	* (difficult)	* (difficult)	*** (built-in)	Same as CF
Speed	*** (fastest)	**	*	Same as CF
Easy to learn	**	*** (easiest)	*	Not sure yet…
Personal opinion	Best coverage and fastest option. Use it with Lambda and Step Functions for robustness.	Not designed for automation, but works for most scenarios.	Rock solid. My preferred option to deploy and update endpoints	If you're allergic to CF templates or if you want to try something different. Promising, but caveat emptor ☺

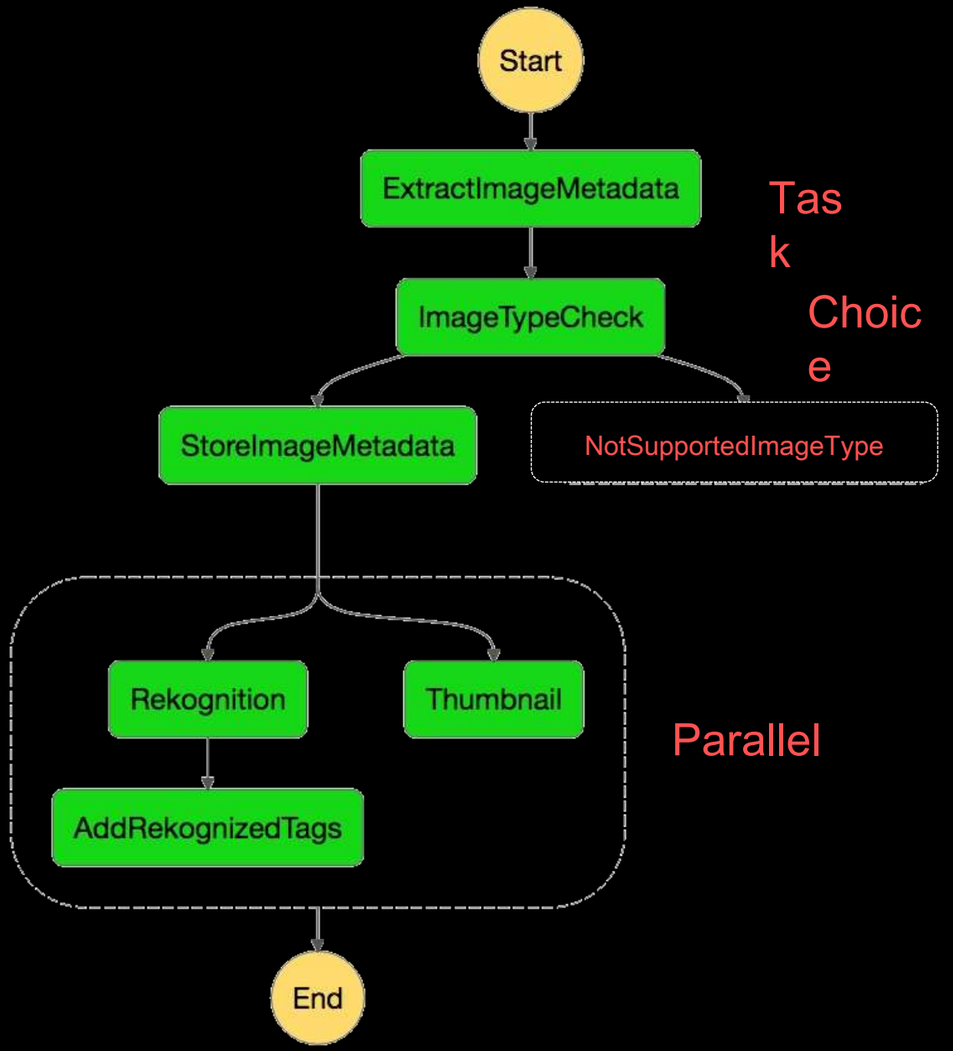
More options

- Troposphere (2013)
  - Open source project https://github.com/cloudtools/troposphere
  - Write Python code, generate CloudFormation templates (I like it a lot)
- Terraform (2014)
  - Open source tool by Hashicorp https://www.terraform.io/
  - Coverage is similar to CloudFormation
- Airflow (2015)
  - Open source project https://airflow.apache.org/
  - https://aws.amazon.com/blogs/machine-learning/build-end-to-end-machine-learning-workflows-with-amazon-sagemaker-and-apache-airflow/
- MLFlow (2018)
  - Open source project by Databricks https://mlflow.org/
  - https://www.mlflow.org/docs/latest/python_api/mlflow.sagemaker.html

2- Orchestrating with AWS Lambda and AWS Step Functions

AWS Step Functions





Step Functions uses Amazon States Language (JSON)

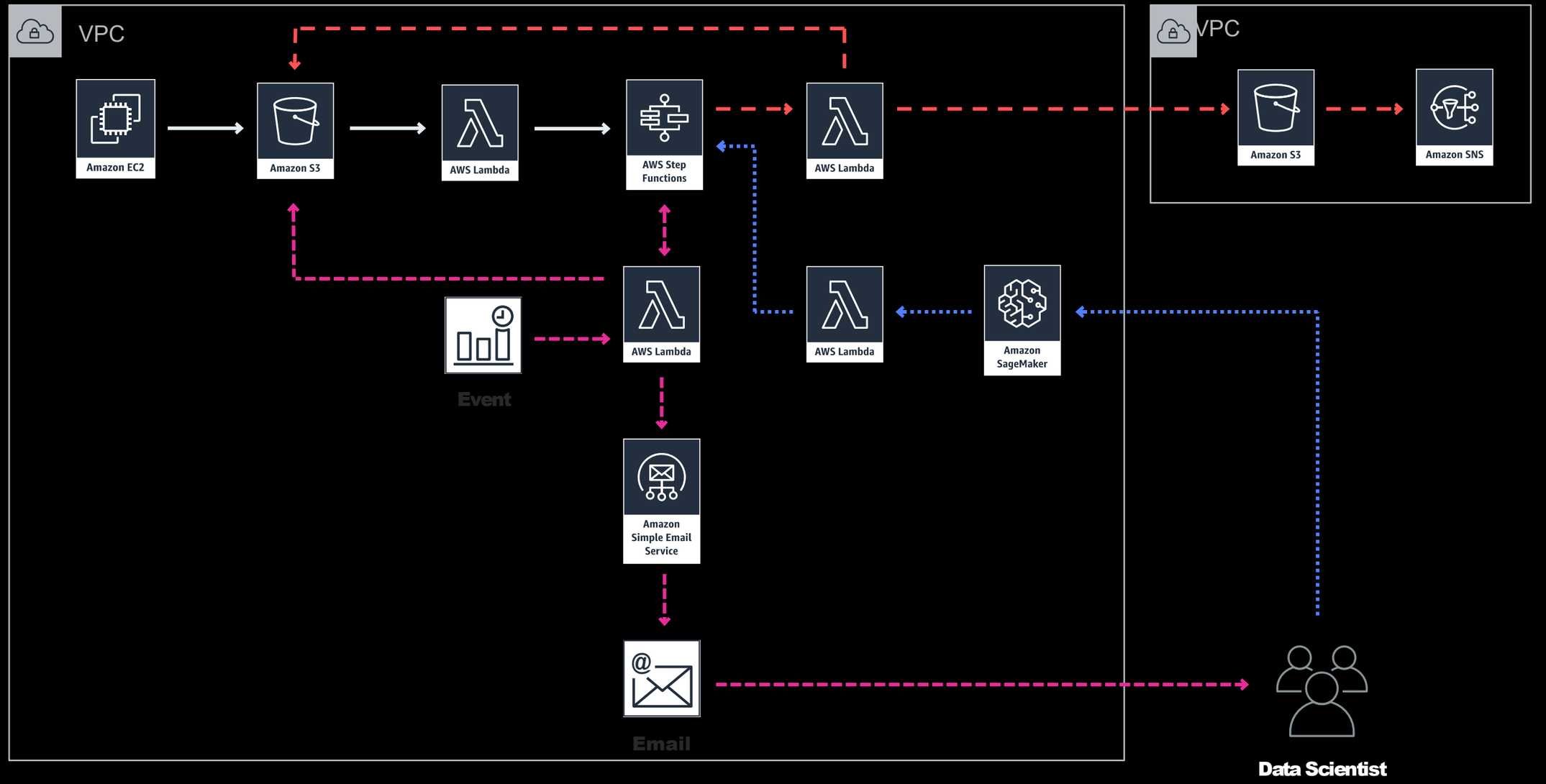
```
{
"Comment": "Image Processing workflow",
"StartAt": "ExtractImageMetadata",
"States": {
    "ExtractImageMetadata": {
    "Type": "Task",
    "Resource": "arn:aws:lambda:::function:photo-backendExtractImageMetadata-...",

    "InputPath": "$",
    "ResultPath": "$.extractedMetadata",
    "Next": "ImageTypeCheck",
    "Catch": [ {
          "ErrorEquals": [ "ImageIdentifyError"],
          "Next": "NotSupportedImageType"
          } ],
    "Retry": [ {
          "ErrorEquals": [ "States.ALL"],
          "IntervalSeconds": 1,
          "MaxAttempts": 2,
          "BackoffRate": 1.5 }, ...
```

Customer example

https://www.youtube.com/watch?v=-PWjI6Ri2aY

Customer example (cont'd)



3- Creating notebook instances

Demo

Creating a notebook instance with AWS CloudFormation

https://gitlab.com/juliensimon/sagemaker-automation/blob/master/cloudformation/notebook-instance.yml

Demo

Creating a notebook instance with AWS CDK (Python)

https://gitlab.com/juliensimon/sagemaker-automation/tree/master/cdk/notebook_instances

4 - Training models

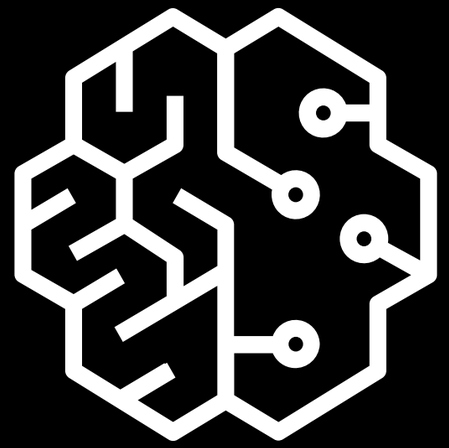
Demo

Deploying a scheduled Lambda function
to retrain a SageMaker model automatically

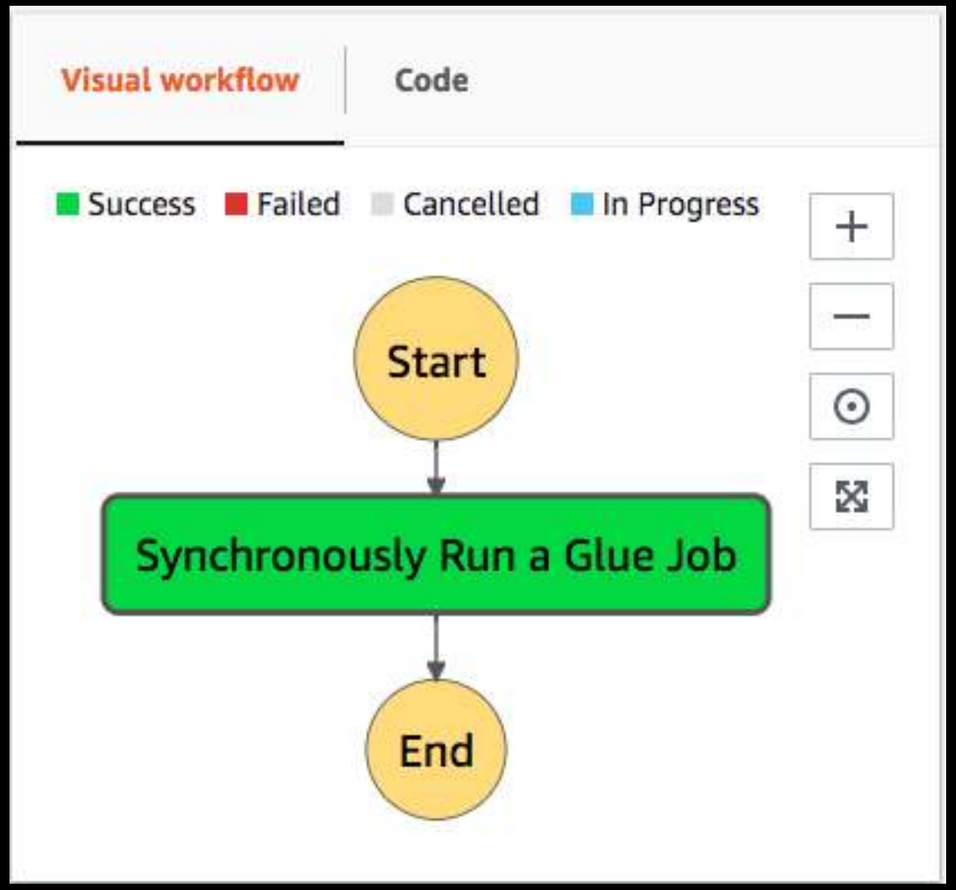
https://gitlab.com/juliensimon/aws/tree/master/lambda_frameworks/serverless/sagemakerscheduler

AWS Step Functions can “hide” asynchronous jobs

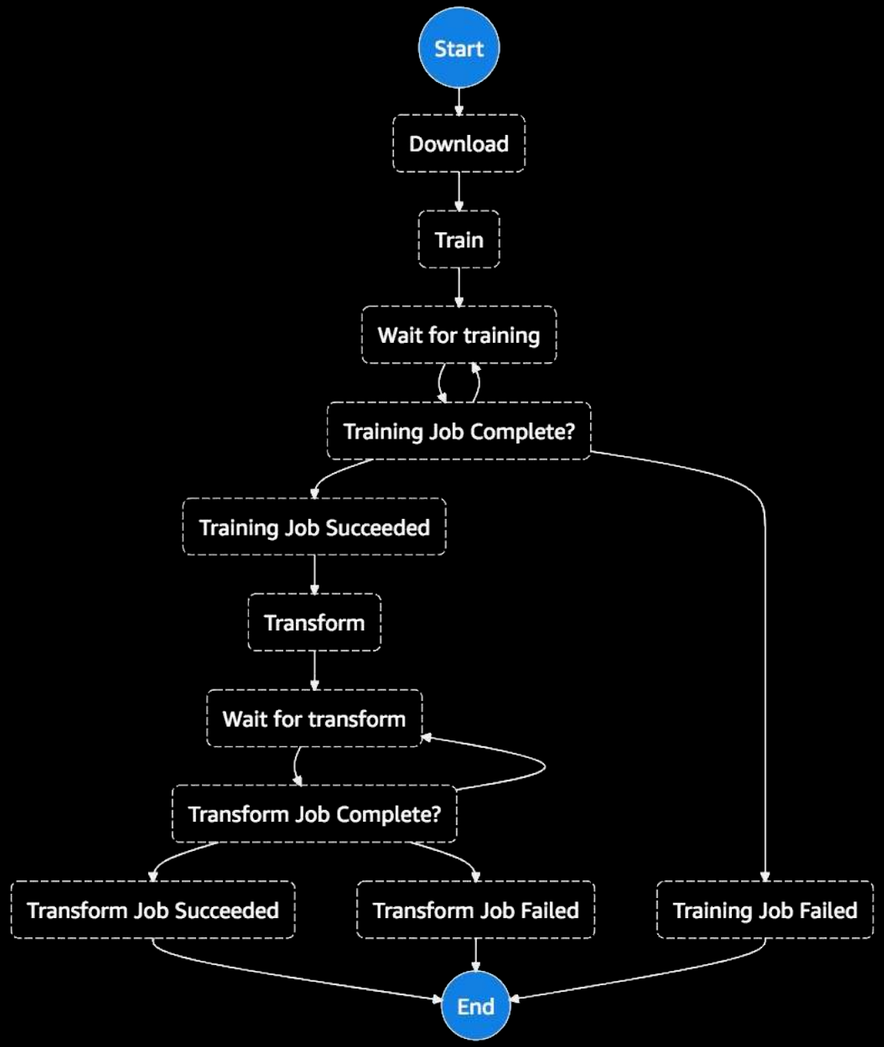
AWS
Glue

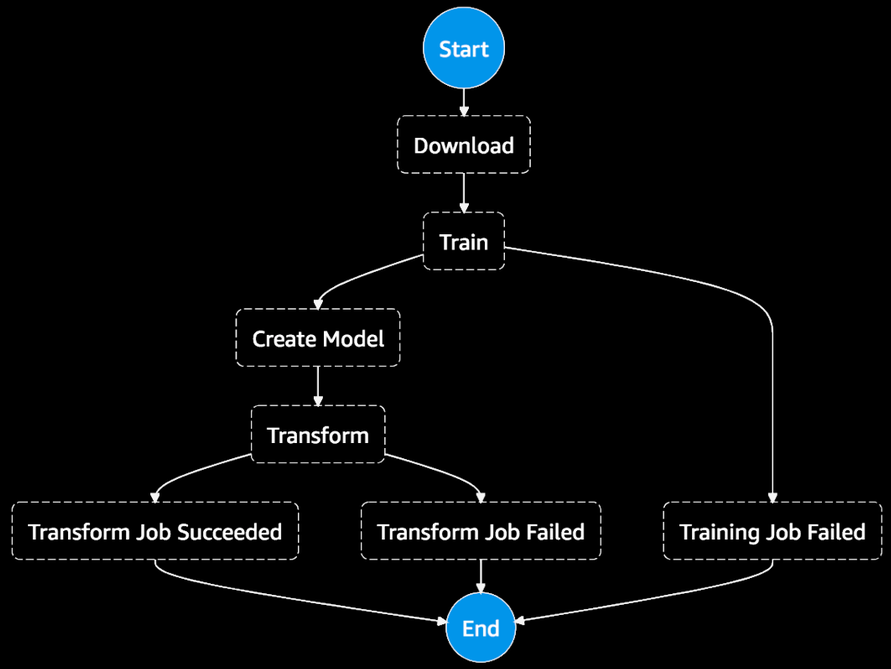


Amazon
SageMaker



Simplify machine learning workflows





Add AWS Glue ETL jobs in your workflows

```
"Synchronously Run a Glue Job": {
    "Type": "Task",
    "Resource": "arn:aws:states:::glue:startJobRun.sync",
    "Parameters":
          {
          "JobName.$": "$.myJobName”,
          “AllocatedCapacity”: 3
          },
    "Catch": [
      {"ErrorEquals": ["States.TaskFailed"],
       "ResultPath": "$.cause",
       "Next" : "Notify on Error"
       } ],
    "ResultPath": "$.jobInfo",
    "Next": "Report Success"
 }
```

Add Amazon SageMaker jobs in your workflows

```
"Synchronously Run a Training Job": {
    "Type": "Task",
    "Resource":
"arn:aws:states:::sagemaker.createTrainingJob.sync",
    "Parameters":
          {
        "AlgorithmSpecification": {...},
        "HyperParameters": {...},
        "InputDataConfig": [...],
        ...
          },
    "Catch": [
      {"ErrorEquals": ["States.TaskFailed"],
       "ResultPath": "$.cause",
       "Next" : ”Notify on Error"
       } ],
    "ResultPath": "$.jobInfo",
    "Next": "Report Success"
 }
```

```
"Synchronously Run a Transform Job": {
    "Type": "Task",
    "Resource":
"arn:aws:states:::sagemaker.createTransformJob.sync",
    "Parameters":
          {
              "TransformJobName.$": "$.transform",
              "ModelName.$": "$.model",
              "MaxConcurrentTransforms": 8,
              ...
          },
    "Catch": [
      {"ErrorEquals": ["States.TaskFailed"],
       "ResultPath": "$.cause",
       "Next" : ”Notify on Error"
       } ],
    "ResultPath": "$.jobInfo",
    "Next": "Report Success"
 }
```

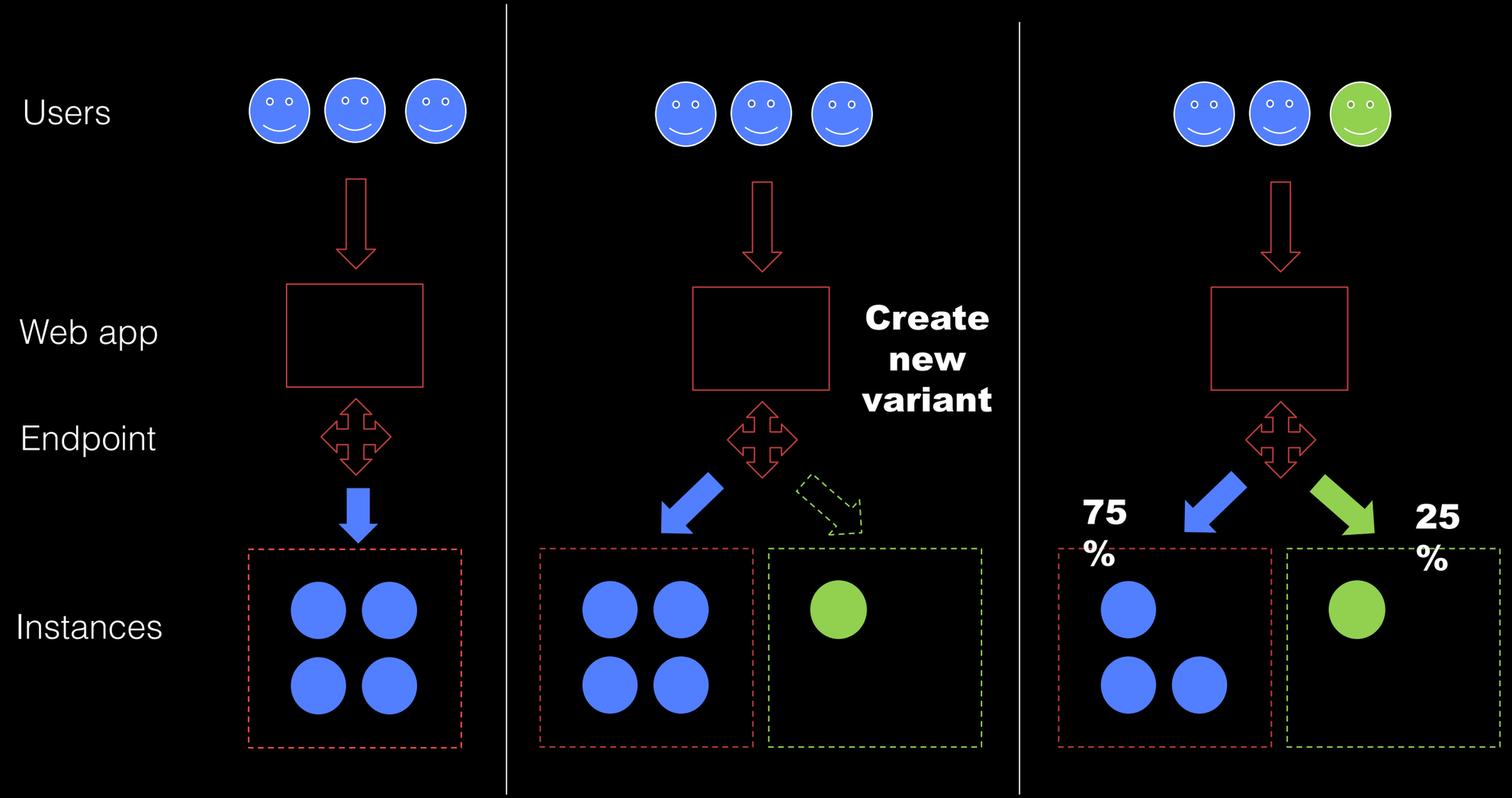

aws

Define workflows in JSON

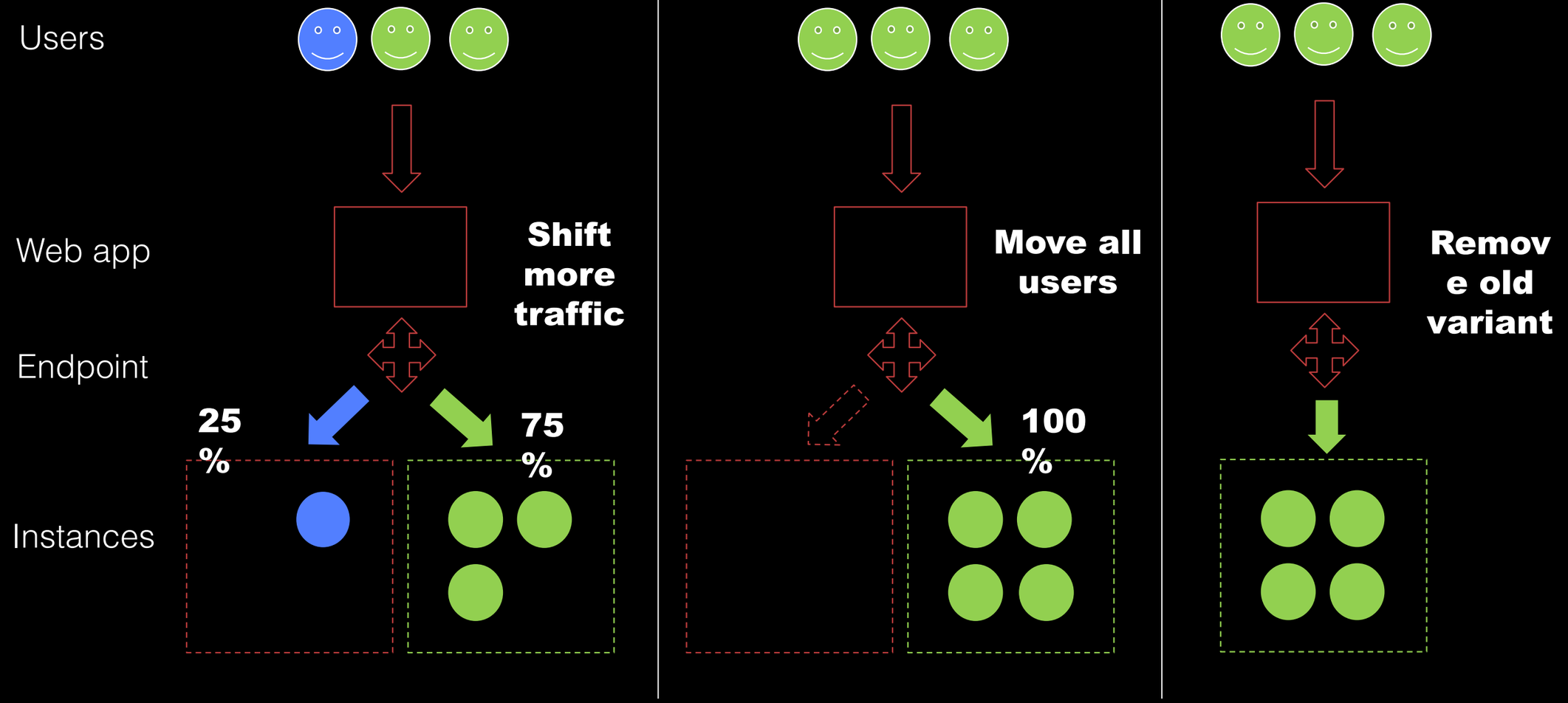
```
{
"StartAt": "Download",
"States": {
        "Download": {
                "Type": "Task",
                "Resource": "arn:aws:lambda:REGION:ACCT:function:download_data”,
                "Next": "Train"
                },
        "Train": {
                "Type": "Task",
                "Resource": "arn:aws:states:::sagemaker:createTrainingJob.sync",
                "ResultPath": "$.training_job",
                "Parameters": {
                        "AlgorithmSpecification": {
                        "TrainingImage": "811284229777.dkr.ecr.us-east-1.amazonaws.com/
                image-classification:latest",
                        "TrainingInputMode": "File"
                }…
```

5 - Deploying models

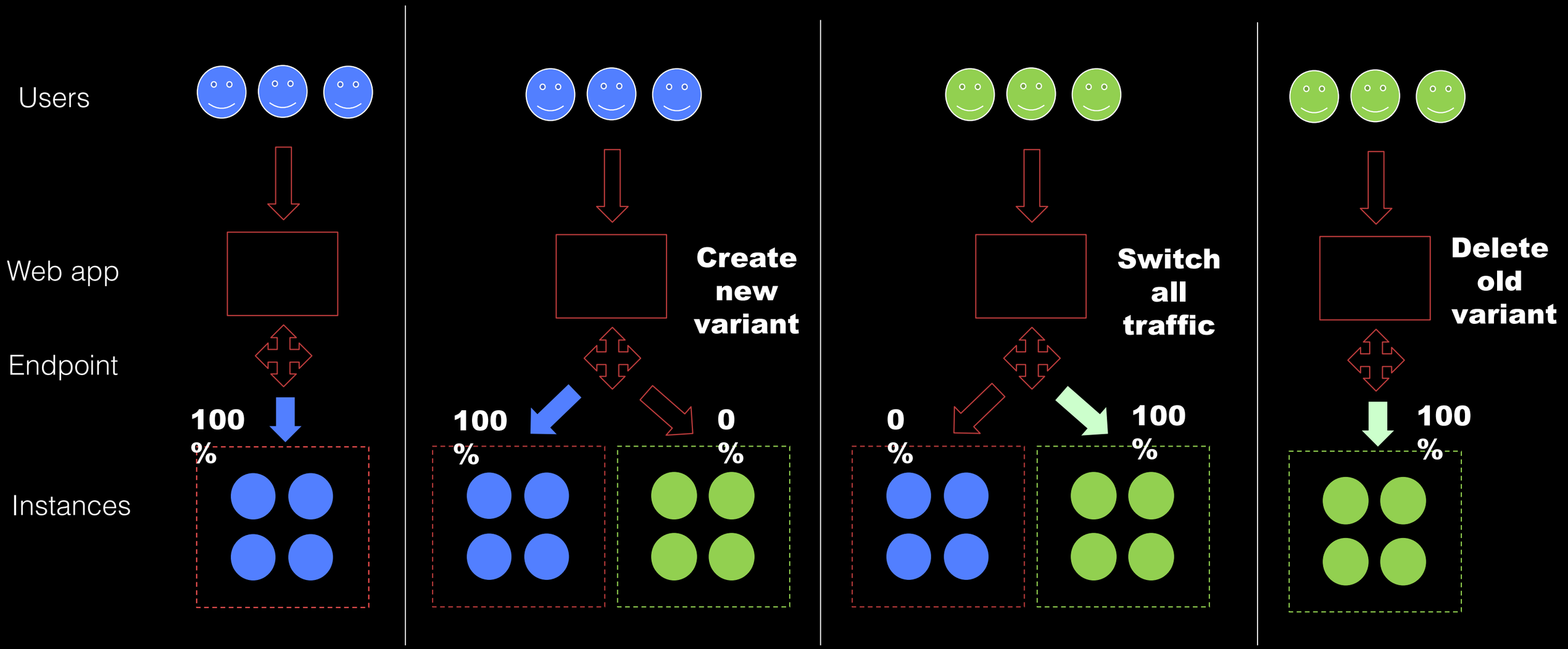
Canary deployment 1/2



Canary deployment 2/2



Blue-green deployment



Demo

Deploying Production Variants on XGBoost with boto3

https://gitlab.com/juliensimon/ent321

Demo

Deploying Production Variants on XGBoost with AWS CloudFormation

https://gitlab.com/juliensimon/sagemaker-automation/tree/master/cloudformation

Getting started

http://aws.amazon.com/free

https://aws.amazon.com/cloudformation
https://aws.amazon.com/lambda
https://aws.amazon.com/stepfunctions

https://aws.amazon.com/sagemaker
https://github.com/aws/sagemaker-python-sdk
https://github.com/awslabs/amazon-sagemaker-examples
https://github.com/aws-samples/aws-sagemaker-build

https://medium.com/@julsimon
https://gitlab.com/juliensimon/sagemaker-automation

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Merci!

Julien Simon
Global Evangelist, AI & Machine Learning, AWS
@julsimon