

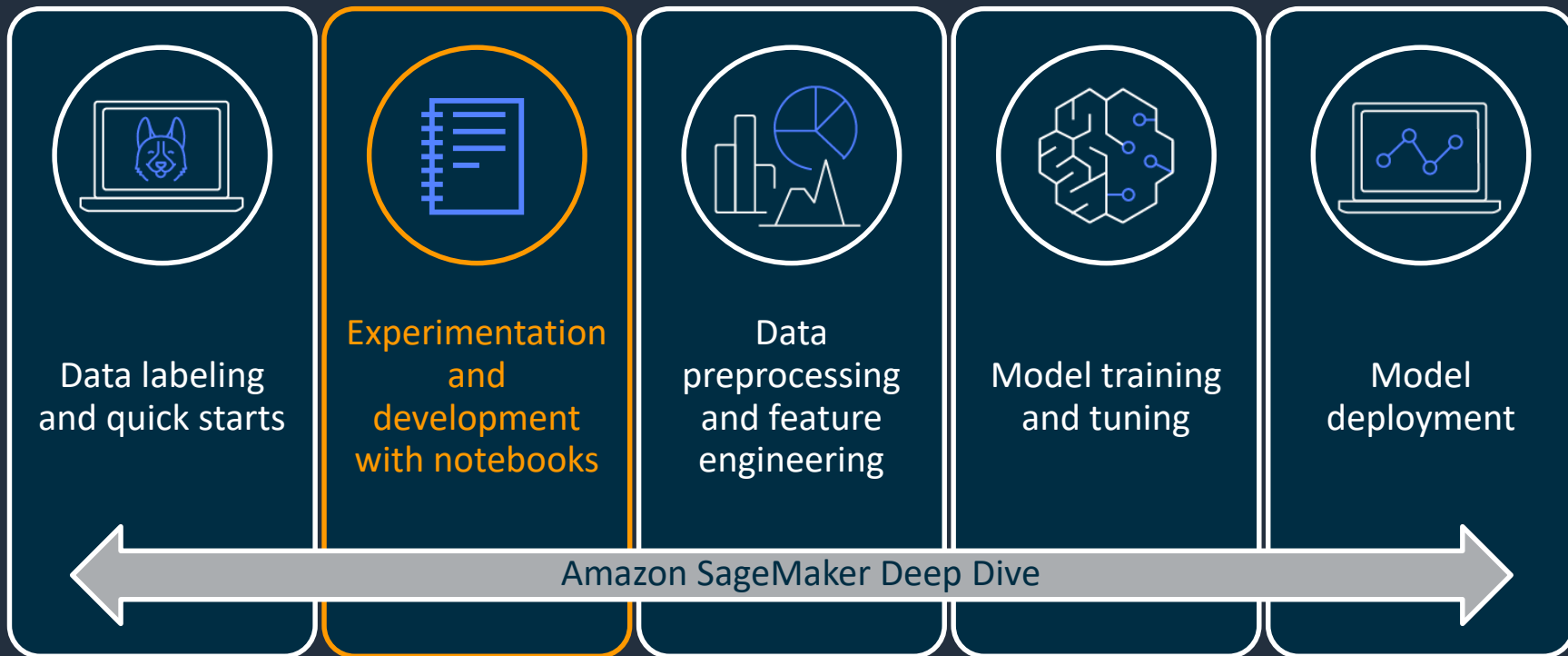


# Experimentation and development with notebooks

Amazon SageMaker Deep Dive Series



# Amazon SageMaker Deep Dive Series



# Amazon SageMaker key benefits

Most complete,  
end-to-end ML service



## Accelerate ML development

20+ tools covering the entire ML development lifecycle



## Boost data scientist productivity

The world's first integrated development environment (IDE)



## Reduce cost

Eliminate costs of writing custom integration code with integrated functionality optimized for ML

# Amazon SageMaker overview

## Amazon SageMaker

### PREPARE

#### SageMaker Ground Truth

Label training data for machine learning

#### SageMaker Data Wrangler

Aggregate and prepare data for machine learning

#### SageMaker Processing

Built-in Python, BYO R/Spark

#### SageMaker Feature Store

Store, update, retrieve, and share features

#### SageMaker Clarify

Detect bias and understand model predictions

### BUILD

#### SageMaker Studio Notebooks

Jupyter notebooks with elastic compute and sharing

#### Built-in and Bring your-own Algorithms

Dozens of optimized algorithms or bring your own

#### SageMaker Autopilot

Automatically create machine learning models with full visibility

#### SageMaker JumpStart

Pre-built solutions for common use cases

#### SageMaker Canvas

Generate accurate machine learning predictions—no code required

#### SageMaker Studio Lab

Learn and experiment with ML using a no-setup, free development environment

#### RStudio

Fully integrated development environment for machine learning

### TRAIN & TUNE

#### Managed Training

Distributed infrastructure management

#### SageMaker Experiments

Capture, organize, and compare every step

#### Automatic Model Tuning

Hyperparameter optimization

#### Distributed Training Libraries

Training for large datasets and models

#### SageMaker Debugger

Debug and profile training runs

#### Managed Spot Training

Reduce training cost by 90%

#### Managed Training Compiler

Accelerate training of deep learning models by up to 50%

### DEPLOY & MANAGE

#### Managed Deployment

Fully managed, ultra low latency, high throughput

#### Kubernetes & KubeFlow Integration

Simplify Kubernetes-based machine learning

#### Multi-Model Endpoints

Reduce cost by hosting multiple models per instance

#### SageMaker Model Monitor

Maintain accuracy of deployed models

#### SageMaker Edge Manager

Manage and monitor models on edge devices

#### SageMaker Pipelines

Workflow orchestration and automation

#### SageMaker Inference Recommender

Automate load testing and optimize model performance across ML instances

### SageMaker Studio

Integrated development environment (IDE) for ML

# Features covered in this session

## Amazon SageMaker

### PREPARE

#### SageMaker Ground Truth

Label training data for machine learning

#### SageMaker Data Wrangler

Aggregate and prepare data for machine learning

#### SageMaker Processing

Built-in Python, BYO R/Spark

#### SageMaker Feature Store

Store, update, retrieve, and share features

#### SageMaker Clarify

Detect bias and understand model predictions

### BUILD

#### SageMaker Studio Notebooks

Jupyter notebooks with elastic compute and sharing

#### Built-in and Bring your-own Algorithms

Dozens of optimized algorithms or bring your own

#### SageMaker Autopilot

Automatically create machine learning models with full visibility

#### SageMaker JumpStart

Pre-built solutions for common use cases

#### SageMaker Canvas

Generate accurate machine learning predictions—no code required

#### SageMaker Studio Lab

Learn and experiment with ML using a no-setup, free development environment

#### RStudio

Fully integrated development environment for machine learning

### TRAIN & TUNE

#### Managed Training

Distributed infrastructure management

#### SageMaker Experiments

Capture, organize, and compare every step

#### Automatic

#### Model Tuning

Hyperparameter optimization

#### Distributed Training

#### Libraries

Training for large datasets and models

#### SageMaker Debugger

Debug and profile training runs

#### Managed Spot Training

Reduce training cost by 90%

#### Managed Training Compiler

Accelerate training of deep learning models by up to 50%

### DEPLOY & MANAGE

#### Managed Deployment

Fully managed, ultra low latency, high throughput

#### Kubernetes & Kubeflow Integration

Simplify Kubernetes-based machine learning

#### Multi-Model Endpoints

Reduce cost by hosting multiple models per instance

#### SageMaker Model Monitor

Maintain accuracy of deployed models

#### SageMaker Edge Manager

Manage and monitor models on edge devices

#### SageMaker Pipelines

Workflow orchestration and automation

#### SageMaker Inference Recommender

Automate load testing and optimize model performance across ML instances

### SageMaker Studio

Integrated development environment (IDE) for ML

# Development with SageMaker IDE's

Familiar integrated development environments with managed infrastructure



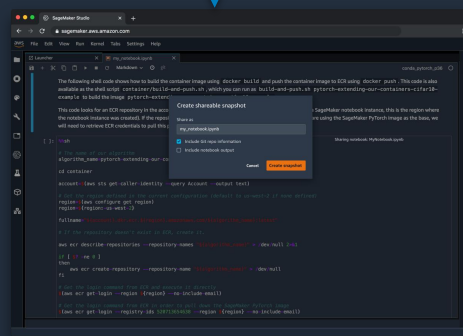
Amazon SageMaker

```
In [1]: bucket = "cyclex-s3_bucket_name_here"
previx = "sagemaker/2020-11-01-sagemaker"

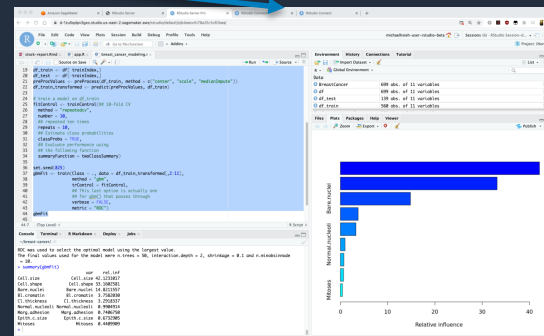
# Define IAM role
import boto3
import os
from sagemaker import get_execution_role
role = get_execution_role()

# Load the dataset
```

Classic Jupyter notebooks



Studio Jupyter notebooks



Rstudio notebooks

# ML instance types

ml.c5n.xlarge

Family

Generation

Attribute

Size

**t** general purpose

**m** memory

**c** compute

**p** GPU

**g** GPU

**3**

**4**

**5**

**6**

**n** network

**d** local storage

**m** metal

**g** graviton

small 4xlarge

medium 8xlarge

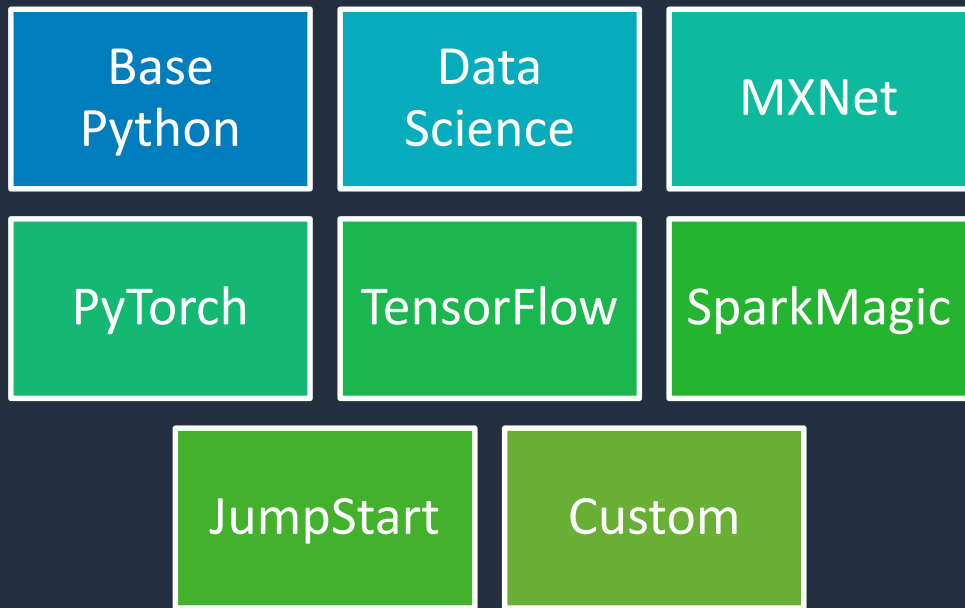
large 12xlarge

xlarge 24xlarge

2xlarge

# Kernels

Managed environments with pre-installed packages for data science.



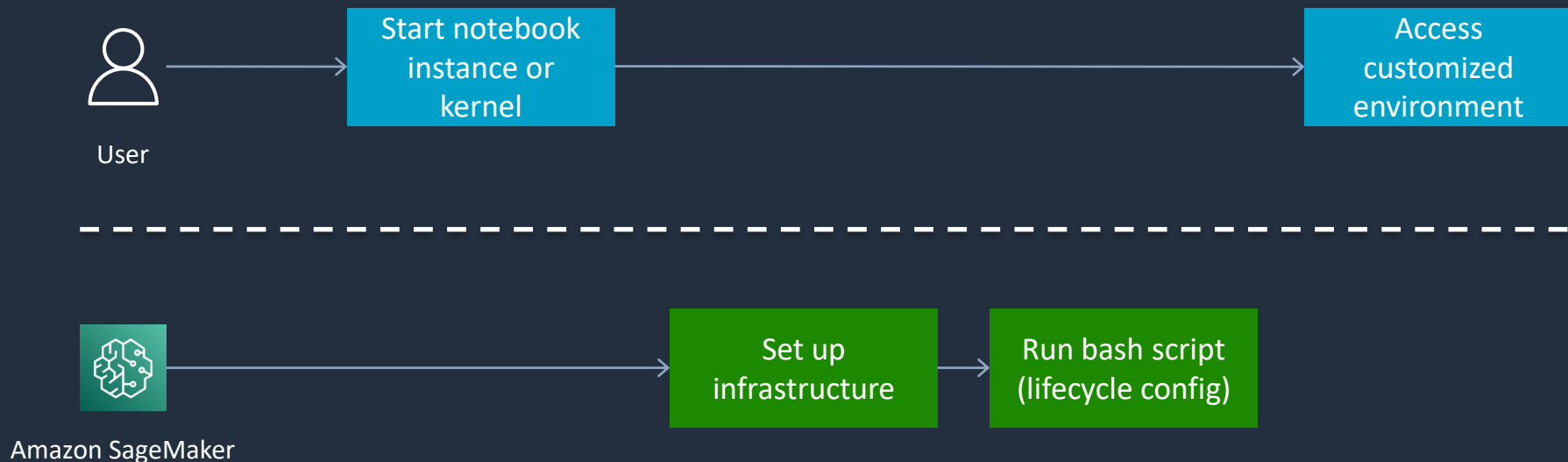
## Packages in Data Science kernel:

- bokeh
- boto3
- matplotlib
- nltk
- numpy
- pandas
- plotly
- scipy
- seaborn
- sklearn
- ...



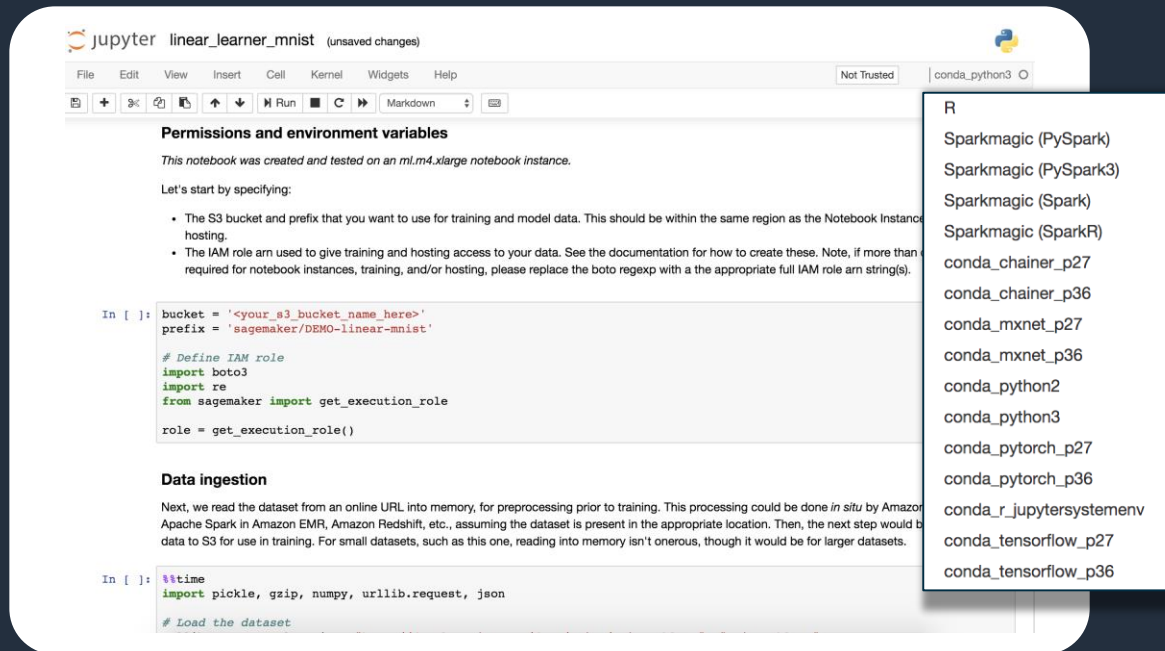
# Lifecycle configurations

Automate customization of notebook environments with bash scripts. Use to install libraries, extensions, mount file systems, connect to other services (EMR, code repository), and more...



# Classic Jupyter notebooks

- Managed Jupyter notebooks
- Supports Jupyter Lab
- Multiple built-in kernels
- Install external dependencies
- Integrate with Git
- Sample notebooks
- VPC connectivity
- Automation with Lifecycle Configuration



# Under the hood



**CONDA<sup>®</sup>**  
15+ pre-built kernels

# Demo

Classic Jupyter notebooks

# SageMaker Studio notebooks



## Easy access with Single Sign-On

Access your notebooks in seconds



## Fully managed and secure

Administrators manage access and permissions



## Fast setup

Start your notebooks without spinning up compute resources



## Easy collaboration

Share notebooks with a single click

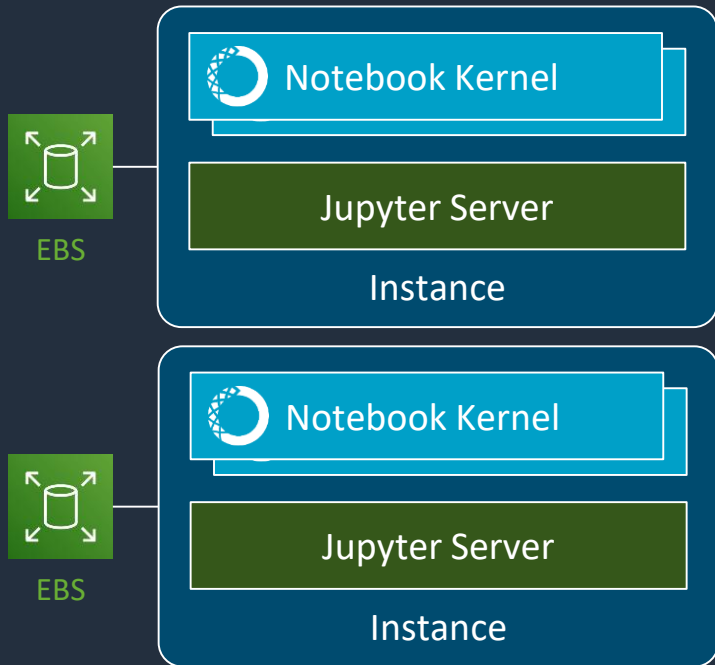


## Flexible

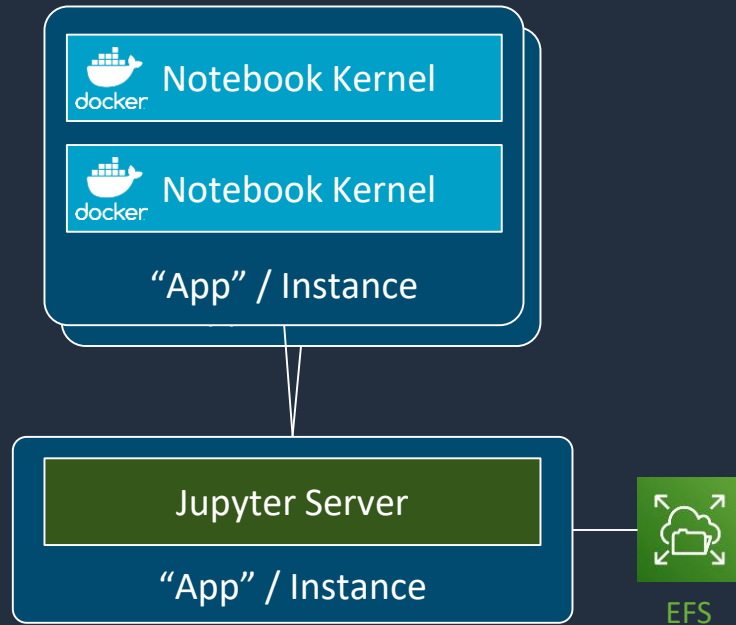
Dial up or down compute resources (coming soon)

# Differences under the hood

## Classic Notebook User



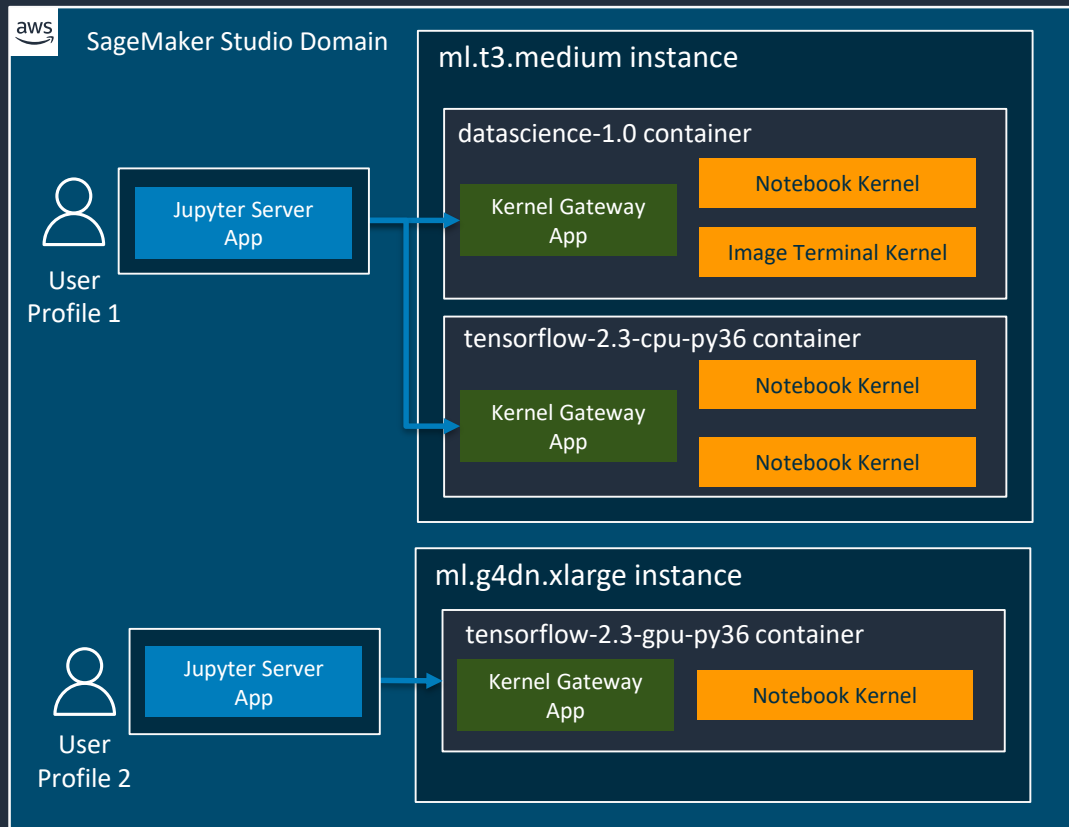
## SageMaker Studio User



# Right sizing notebook compute

SageMaker Studio is based on a Kernel Gateway architecture

- **JupyterServer** runs the Jupyter server. Enables access to EFS and notebooks without selecting a kernel.
- **Kernel Gateway** runs a SageMaker image container (an environment). Can run up to 4 Kernel Apps (if CPU/RAM/GPU requirements are met).
- **Kernel App** runs individual notebooks, terminals, or other components.



# SageMaker Studio UI

Amazon SageMaker Studio

File Edit View Run Kernel Git Tabs Settings Help

2 vCPU + 4 GiB Python 3 (Data Science) Kernel: CPU: 0.00% MEM: 2.82% Share

## Prerequisites and Data

### Initialize SageMaker

[2]:

```
from sagemaker import Session
session = Session()
bucket = session.default_bucket()
prefix = 'sagemaker/DEMO-sagemaker-clarify'
region = session.boto_region_name
# Define IAM role
from sagemaker import get_execution_role
import pandas as pd
import numpy as np
import urllib
import os

role = get_execution_role()
```

EC2 instance

Kernel

Kernel metrics

Share notebook

### Download data

Data Source: <https://archive.ics.uci.edu/ml/machine-learning-databases/adult/>

Let's download the data and save it in the local folder with the name `adult.data` and `adult.test` from UCI repository<sup>[2]</sup>.

<sup>[2]</sup> Dua Dheeru, and Efi Karra Taniskidou. "UCI Machine Learning Repository". Irvine, CA: University of California, School of Information and Computer Science (2017).

[3]:

```
adult_columns = ["Age", "Workclass", "fnlwgt", "Education", "Education-Num", "Marital Status",
                 "Occupation", "Relationship", "Ethnic group", "Sex", "Capital Gain", "Capital Loss",
                 "Hours per week", "Country", "Target"]

if not os.path.isfile('adult.data'):
    urllib.request.urlretrieve('https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data',
                              'adult.data')
    print('adult.data saved!')
else:
    print('adult.data already on disk.')

if not os.path.isfile('adult.test'):
    urllib.request.urlretrieve('https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.test',
                              'adult.test')
    print('adult.test saved!')
else:
    print('adult.test already on disk.')

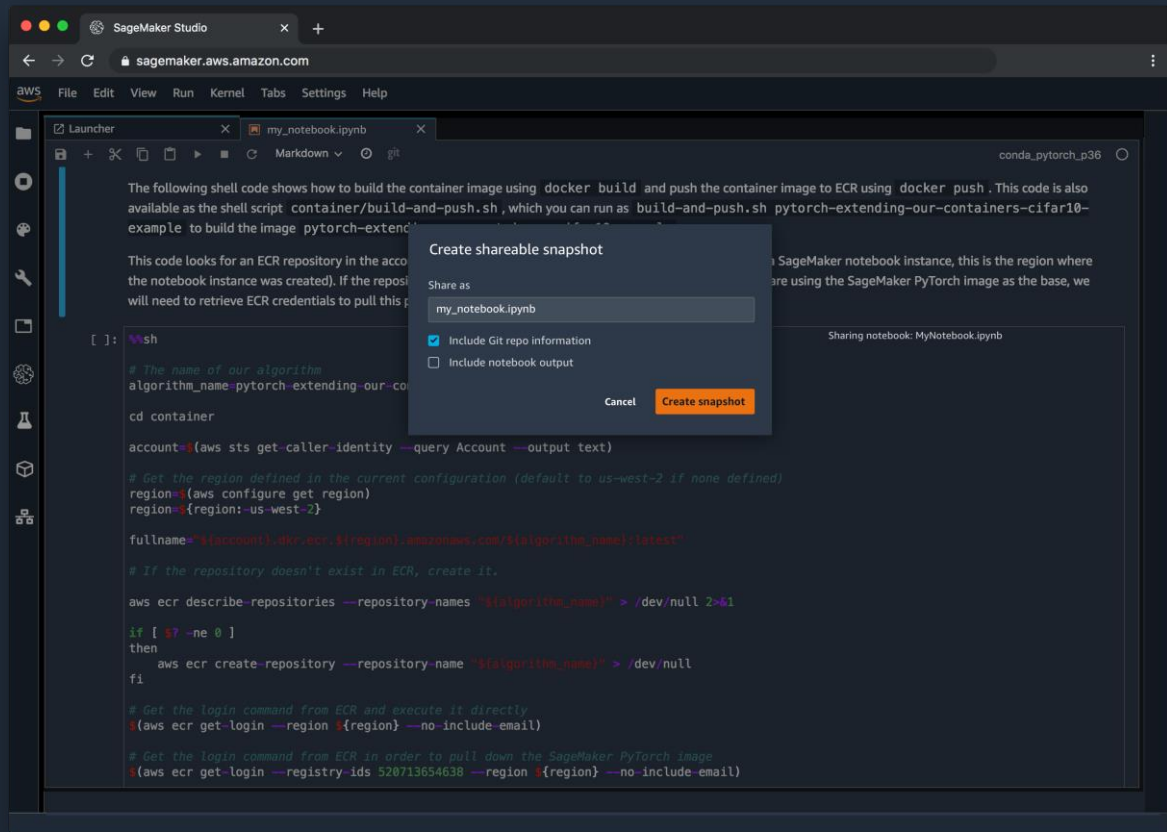
adult.data saved!
adult.test saved!
```

Resources management

0 4 Git: Idle Python 3 (Data Science) | Idle Mode: Command Ln 1, Col 1 fairness\_and\_explainability.ipynb



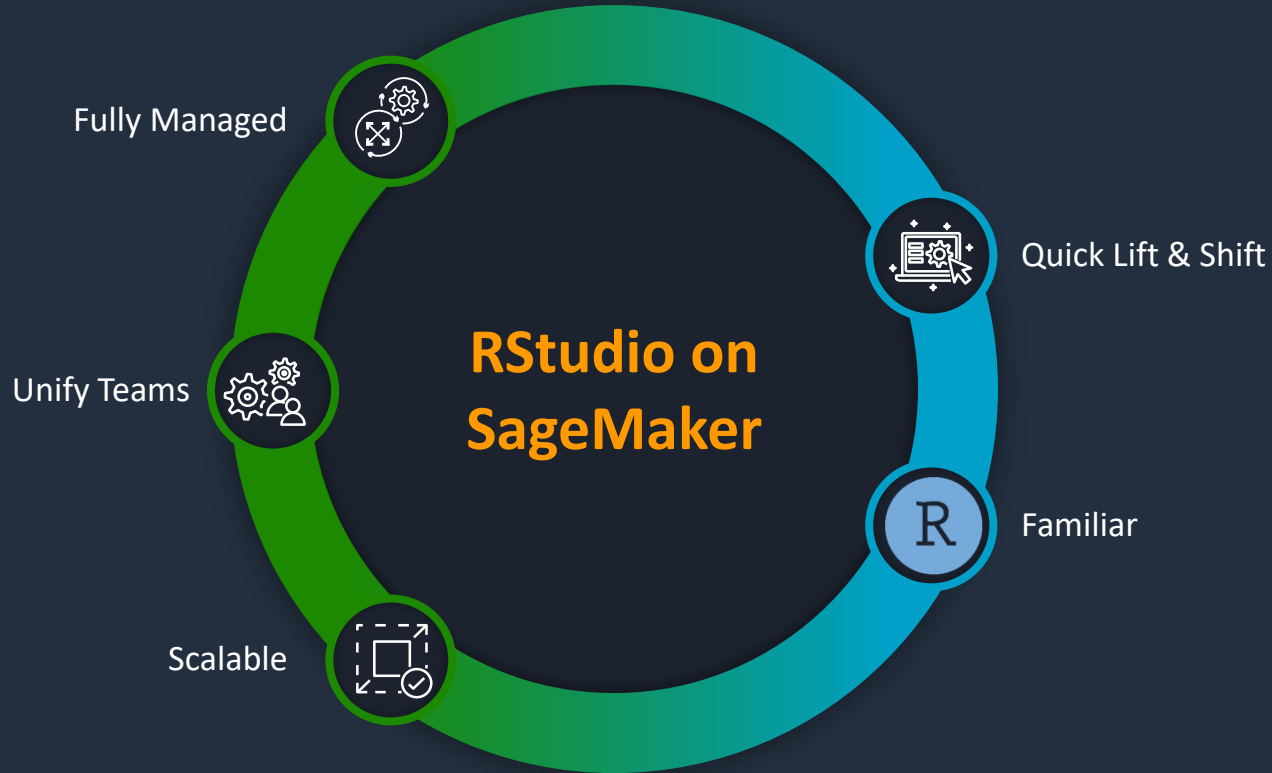
# Shareable notebooks



- Read-only copy (snapshot) of through a secure URL
- All dependences included.
- Snapshots are independent
- Changes in original notebook are not reflected in older snapshots
- Changes in snapshot, are also not reflected in original notebook

# Demo

Studio notebooks



# RStudio on Amazon SageMaker

The screenshot displays the RStudio interface within an Amazon SageMaker environment. The main editor window shows R code for training a Gradient Boosting Machine (GBM) model on breast cancer data. The code includes data loading, preprocessing, cross-validation, and model fitting. The console output shows the results of the model fit, including the ROC curve and the relative influence of various features.

```
19 df_train <- df[ trainIndex, ]
20 df_test <- df[ -trainIndex, ]
21 preProcValues <- preProcess(df_train, method = c("center", "scale", "medianImpute"))
22 df_train_transformed <- predict(preProcValues, df_train)
23
24 # train a model on df_train
25 fitControl <- trainControl(method = "repeatedcv",
26                             number = 10,
27                             repeats = 10,
28                             ## Estimate class probabilities
29                             classProbs = TRUE,
30                             ## Evaluate performance using
31                             ## the following function
32                             summaryFunction = twoClassSummary)
33
34 set.seed(825)
35 gbmFit <- train(Class ~ ., data = df_train_transformed[,2:11],
36                 method = "gbm",
37                 trControl = fitControl,
38                 ## This last option is actually one
39                 ## for gbm() that passes through
40                 verbose = FALSE,
41                 metric = "ROC")
42
43 gbmFit
```

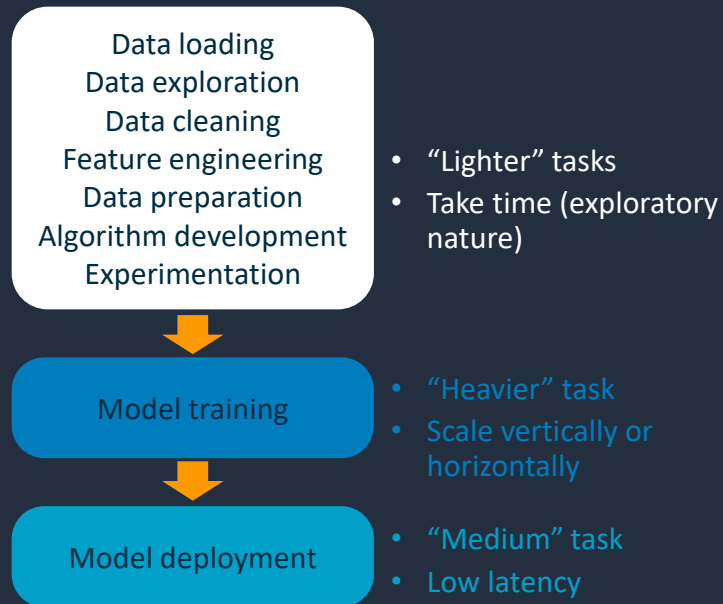
The console output shows the results of the model fit, including the ROC curve and the relative influence of various features:

```
ROC was used to select the optimal model using the largest value.
The final values used for the model were n.trees = 50, interaction.depth = 2, shrinkage = 0.1 and n.minobsinnode = 10.
> summary(gbmFit)
```

	var	rel.inf
Cell.size	Cell.size	42.1231017
Cell.shape	Cell.shape	33.1602581
Bare.nuclei	Bare.nuclei	14.8211557
Bl.cromatin	Bl.cromatin	3.7582030
Cl.thickness	Cl.thickness	3.2918337
Normal.nucleoli	Normal.nucleoli	0.9904914
Marg.adhesion	Marg.adhesion	0.7406750
Epith.c.size	Epith.c.size	0.6732905
Mitoses	Mitoses	0.4409909

The bar chart on the right side of the interface shows the relative influence of the features. The x-axis is labeled 'Relative influence' and ranges from 0 to 40. The y-axis lists the features: Bare.nuclei, Normal.nucleoli, and Mitoses. The bars are colored blue, and the values are approximately: Bare.nuclei (42.1), Normal.nucleoli (33.2), and Mitoses (3.8).

# ML stages have different resource requirements

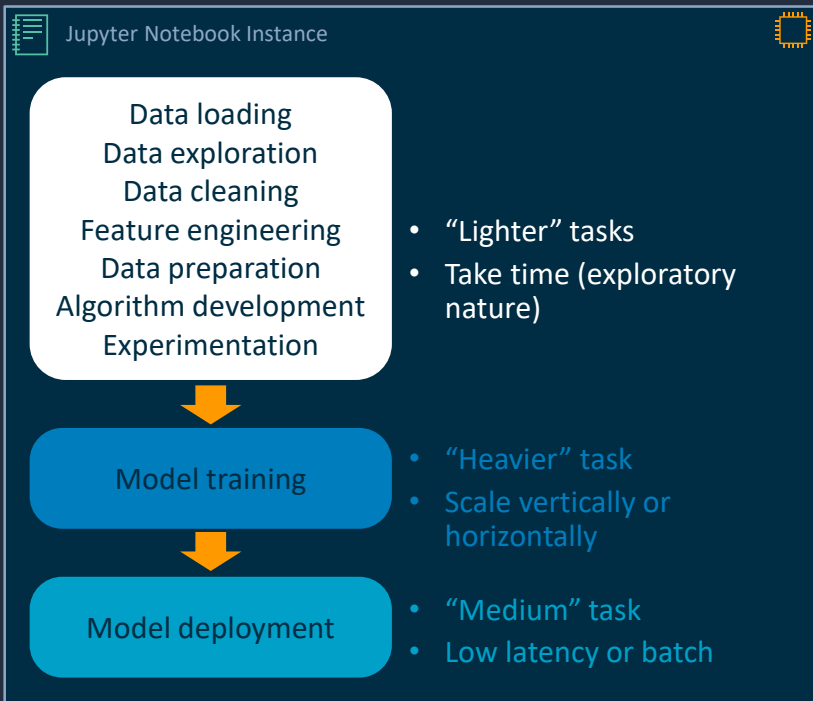


How to choose the right infrastructure?

# Option 1: Run all in one Jupyter notebook instance

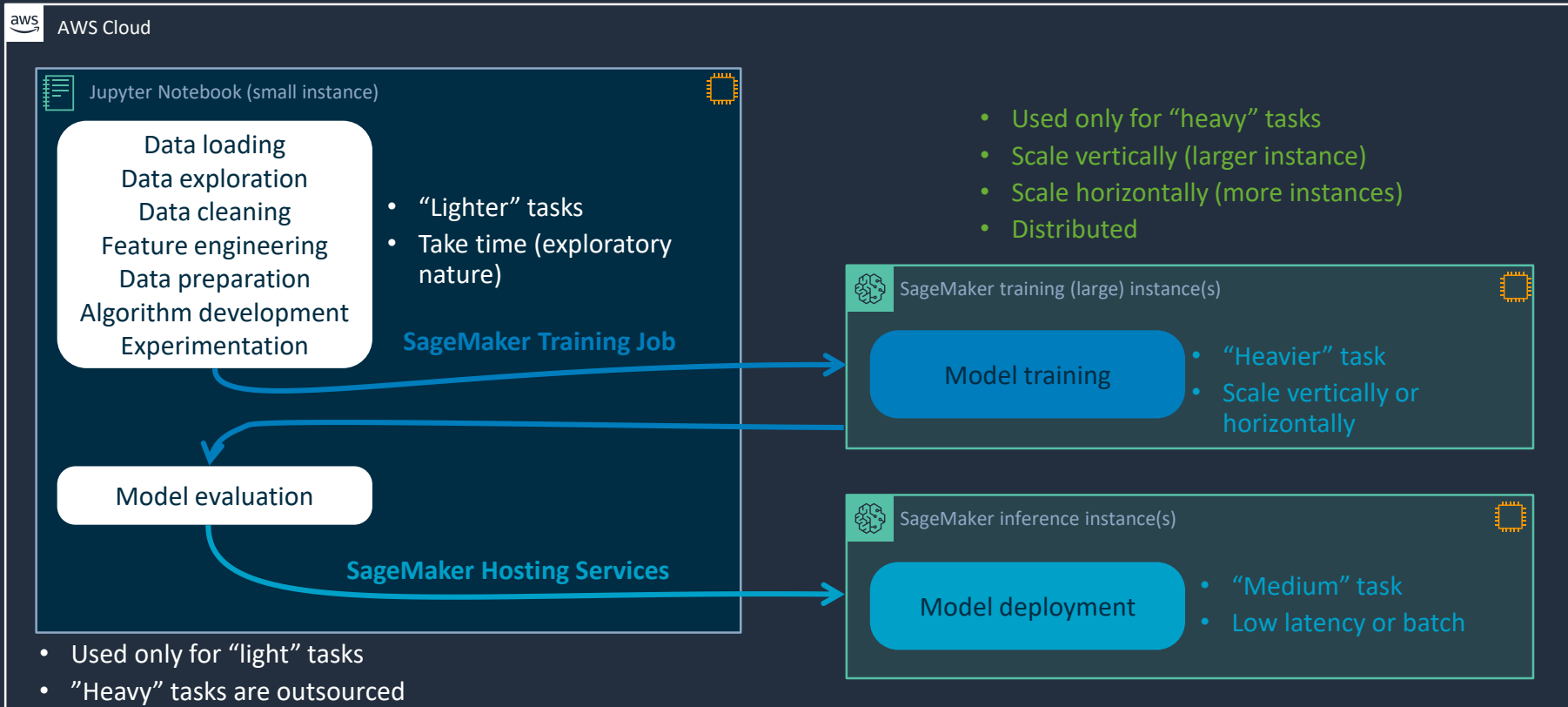


AWS Cloud

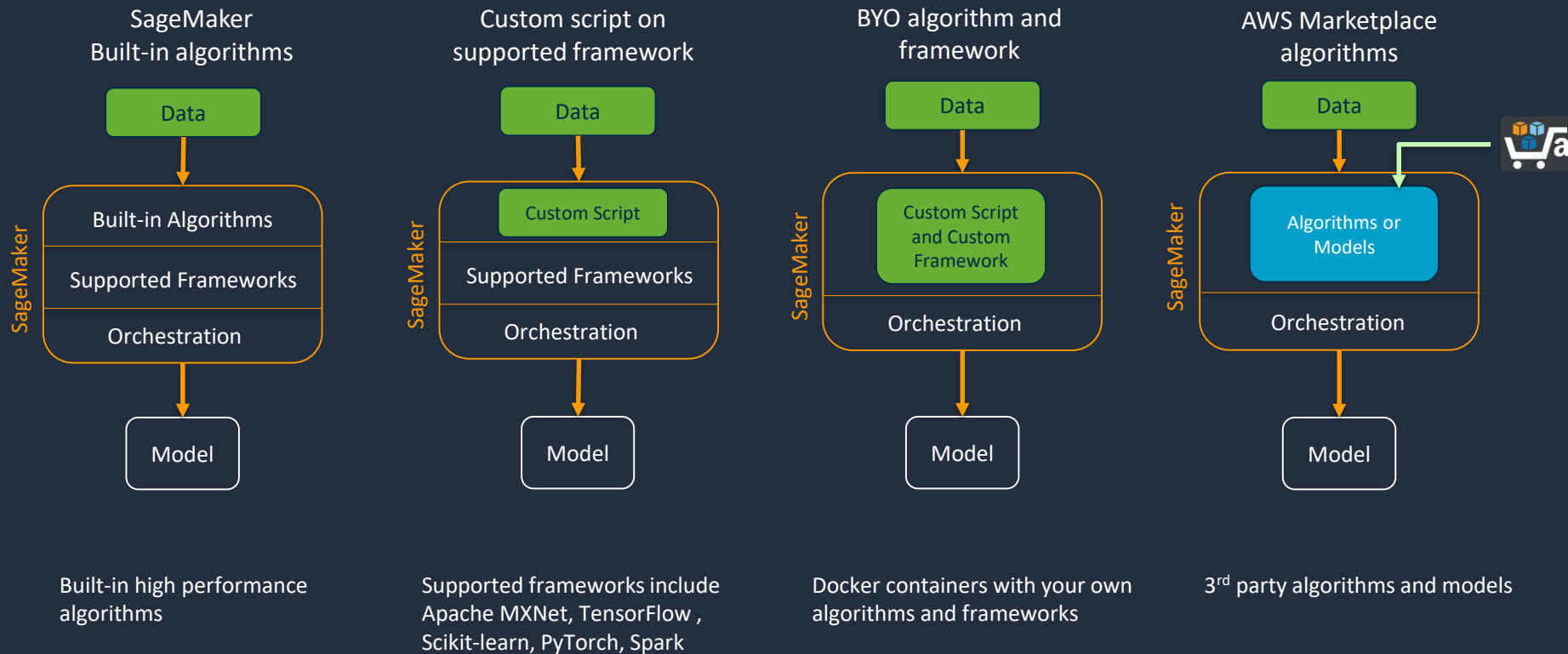


- Used both for “light” and “heavy” tasks
- Difficult to right-size, requires manual effort
- Easy to forget to downsize

# Option 2: Decouple “heavy” from “light” tasks



# Algorithms in SageMaker





# Amazon SageMaker has built-in algorithms or bring your own

## Computer vision

Image classification | Object detection |  
Semantic segmentation

## Topic modeling

LDA | NTM

## Classification and regression

Linear Learner | XGBoost | KNN

## Recommendations

Factorization machines

## Forecasting

DeepAR

## Working with text

BlazingText

## Embeddings

Object2Vec

## Clustering

KMeans

## Sequence translation

Seq2Seq

## Anomaly detection

Random cut forests | IP Insight

## Feature reduction

PCA

# Amazon SageMaker interfaces

## 1. SageMaker Console UI

## 2. AWS Command Line Interface (CLI):

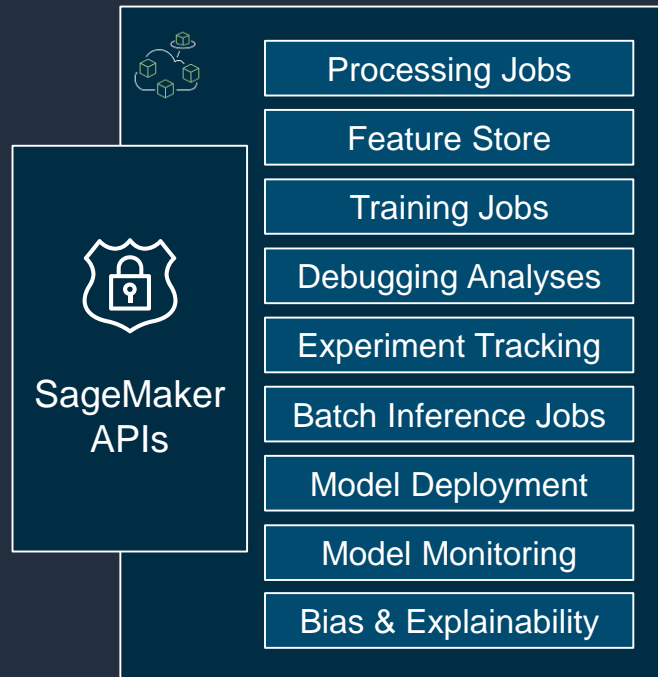
```
aws sagemaker create-endpoint  
  --endpoint-name <value>  
  --endpoint-config-name <value>
```

## 3. **sagemaker**: High-level Python SDK

```
model.deploy(initial_instance_count=1,  
             instance_type='ml.m4.xlarge')
```

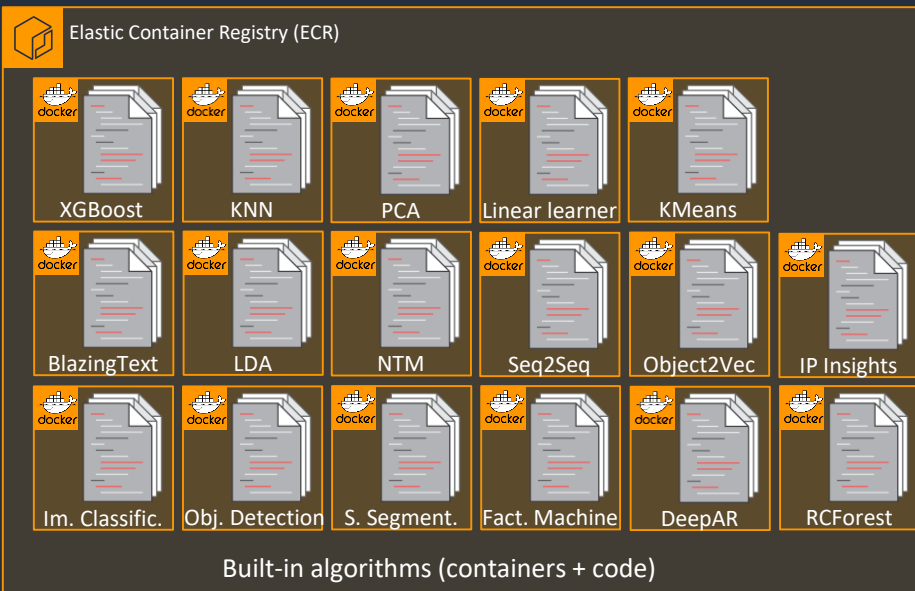
## 4. **boto3**: General AWS Python SDK

```
client.create_endpoint(EndpointName='string',  
                       EndpointConfigName='string')
```

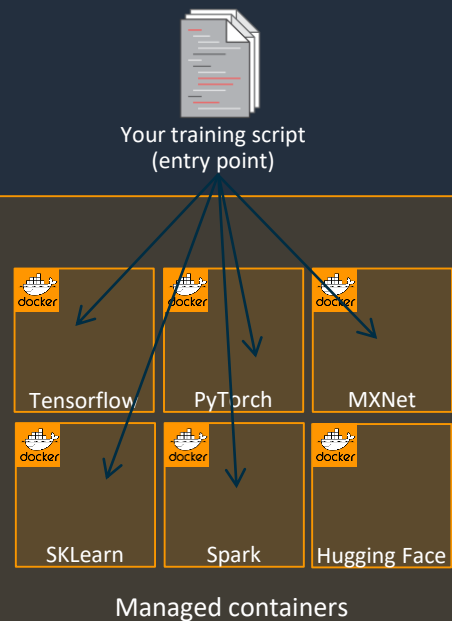


# Under the hood

Select one of the built-in containers



Add your training code to one of the supported framework managed containers



Bring your own container



Unmanaged container

# Build you own container

Build custom Docker containers for SageMaker with your preferred runtime, libraries, and security configurations.

1

OR

2

Extend existing, open-source, [SageMaker framework containers](#).

Create a container using the [SageMaker Training Toolkit](#) and [SageMaker Inference Toolkit](#).

# Features covered in this session

## Amazon SageMaker

### PREPARE

#### SageMaker Ground Truth

Label training data for machine learning

#### SageMaker Data Wrangler

Aggregate and prepare data for machine learning

#### SageMaker Processing

Built-in Python, BYO R/Spark

#### SageMaker Feature Store

Store, update, retrieve, and share features

#### SageMaker Clarify

Detect bias and understand model predictions

### BUILD

#### SageMaker Studio Notebooks

Jupyter notebooks with elastic compute and sharing

#### Built-in and Bring your-own Algorithms

Dozens of optimized algorithms or bring your own

#### SageMaker Autopilot

Automatically create machine learning models with full visibility

#### SageMaker JumpStart

Pre-built solutions for common use cases

#### SageMaker Canvas

Generate accurate machine learning predictions—no code required

#### SageMaker Studio Lab

Learn and experiment with ML using a no-setup, free development environment

#### RStudio

Fully integrated development environment for machine learning

### TRAIN & TUNE

#### Managed Training

Distributed infrastructure management

#### SageMaker Experiments

Capture, organize, and compare every step

#### Automatic

#### Model Tuning

Hyperparameter optimization

#### Distributed Training

#### Libraries

Training for large datasets and models

#### SageMaker Debugger

Debug and profile training runs

#### Managed Spot Training

Reduce training cost by 90%

#### Managed Training Compiler

Accelerate training of deep learning models by up to 50%

### DEPLOY & MANAGE

#### Managed Deployment

Fully managed, ultra low latency, high throughput

#### Kubernetes & Kubeflow Integration

Simplify Kubernetes-based machine learning

#### Multi-Model Endpoints

Reduce cost by hosting multiple models per instance

#### SageMaker Model Monitor

Maintain accuracy of deployed models

#### SageMaker Edge Manager

Manage and monitor models on edge devices

#### SageMaker Pipelines

Workflow orchestration and automation

#### SageMaker Inference Recommender

Automate load testing and optimize model performance across ML instances

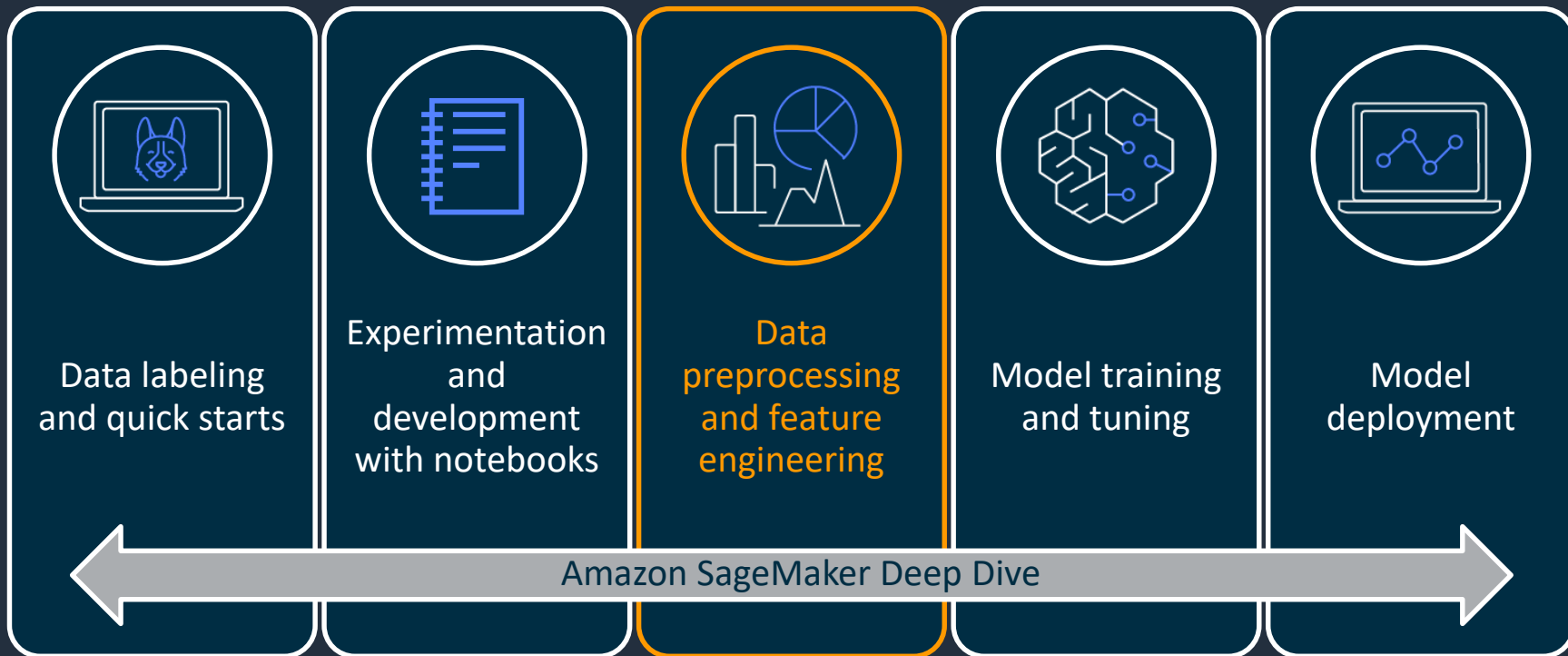
### SageMaker Studio

Integrated development environment (IDE) for ML

# Resources

- [Documentation] [Amazon SageMaker Notebooks](#)
- [Documentation] [Amazon SageMaker Studio Notebooks](#)
- [Documentation] [RStudio on Amazon SageMaker](#)
- [Documentation] [Algorithms in Amazon SageMaker](#)
- [Documentation] [Amazon SageMaker API Reference Guide](#)
- [Documentation] [Amazon SageMaker Python SDK](#)
- [Documentation] [SageMaker boto3 API](#)
- [Code Samples] [Amazon SageMaker Examples](#)
- [Workshop] [Amazon SageMaker Immersion Day](#)
- [Workshop] [Amazon SageMaker 101](#)

# Join us in the next session





# Thank you!

## Australia & New Zealand

Brad Ryan

AI/ML Partner Development Specialist

[brdryn@amazon.com](mailto:brdryn@amazon.com)

Sara van de Moosdijk

AI/ML Partner Solutions Architect

[sarmoosd@amazon.com](mailto:sarmoosd@amazon.com)

## ASEAN

Ling Chang

AI/ML Partner Development Specialist

[linglych@amazon.com](mailto:linglych@amazon.com)

Vasileios Vonikakis

AI/ML Partner Solutions Architect

[vonikakv@amazon.com](mailto:vonikakv@amazon.com)

## India

Ankit Kandoi

Data Analytics & ML Partner  
Development Specialist

[akkandoi@amazon.com](mailto:akkandoi@amazon.com)