# Onboard Foundation Models to SageMaker

SageMaker customers can now find foundation models for different modalities, tasks, and from different model providers at a single location within SageMaker, enabling them to get started with foundation models on SageMaker quickly. Customers can find foundation models based on different tasks or model providers, easily review model characteristics, and usage terms. If they want to try a foundation model, they can do so easily without leaving SageMaker using prebuilt notebooks from model providers. Since all models are hosted and deployed on AWS, customers can be rest assured that their data, whether used for evaluating or using the model at scale, is never shared with third parties.

## Pre-requisites:

- Install the AWS SDK for Python - https://aws.amazon.com/sdk-for-python/
- Install the SageMaker Python SDK - https://sagemaker.readthedocs.io/en/stable/v2.html#installation
- While not strictly necessary, install the AWS CLI tools - https://aws.amazon.com/cli/
- Send us the AWS Account ID you will be using
- Set up 30 min sync with us (junghw@amazon.com, karpmar@amazon.com, and shamanan@amazon.com)

## Overview

To onboard a Foundation Model to SageMaker, the following high-level steps are needed:

1) Update your Docker container to run on SageMaker Inference
2) Create a ModelPackage and list it on AWS Marketplace
3) Create a customer facing notebook & SDK that demonstrates how to use your model

Below we will provide resources and instructions for each of the steps:

## 1) Update your Docker Container to run on SageMaker Inference and test it

SageMaker expects your container to be built in a certain way so it can be used with the SageMaker Inference service. The main call outs are:

- SageMaker runs the container using “docker run <<image>> serve”
- SageMaker expects model resources packaged as a model.tar.gz in S3. SageMaker will un-gzip this into /opt/ml/model
- The container needs to respond to /ping (respond with 200s) and /invocations (where requests are sent to).
- Here’s a guide the provides an overview of the changes: https://towardsdatascience.com/bring-your-own-container-with-amazonsagemaker-37211d8412f4. (Author works at AWS).
- Full docs here: https://docs.aws.amazon.com/sagemaker/latest/dg/your-algorithmsinference-code.html

We recommend performing these steps locally, on an EC2 instance, on a SageMaker Notebook instance. While you're updating your container to run on SageMaker, we recommend local testing to help accelerate the process. Here are relevant links:

1. Start the container: https://github.com/aws/amazon-sagemaker-examples/blob/main/advanced_functionality/scikit_bring_your_own/container/local_test/serve_local.sh
2. Make an inference request: https://github.com/aws/amazon-sagemaker-examples/blob/main/advanced_functionality/scikit_bring_your_own/container/local_test/predict.sh
3. The notebook contains the code to deploy the model/container to a SageMaker hosted real-time endpoint: https://github.com/aws/amazon-sagemaker-examples/blob/main/advanced_functionality/scikit_bring_your_own/scikit_bring_your_own.ipynb

## 2) Create a ModelPackage and list it on AWS Marketplace

We have prepared a notebook to show how to list your model in the AWS Marketplace. If you did not get the notebook with this documentation, please let us know. Follow the instructions in the Model-Partner-Notebook.ipynb to complete this step.

Send us the Marketplace product-id and link to Marketplace product detail page once you're done so we can add it to the catalog of Foundation Models.

## 3) Create a customer facing notebook & SDK that demonstrates how to use your model

The customer facing notebook will be used by the customer to quickly get started with your model. We recommend showing at least the most basic inference request followed by more advanced requests that demonstrate the capabilities of your model. We have a template notebook that we can provide if you would like to use that as a starting point.

In addition, we recommend preparing a SDK (in Python) to improve the user experience for your customers interacting with your model. The SDK would help parse responses, structure requests, handle retry, serialization/deserialization, etc. You can incorporate this into the notebook to further demonstrate how easy it is to use your model.

## 4) Create one pager or slides that introduces model for field stakeholders

Our field team (account managers and solution architects) needs high level background of about the model. One pager should include: high level summary, tasks that model performs well, benchmark for model accuracy, latency with sample load, and example prompts for common tasks.