**Sagemaker custom model deployment- Demo**

**Objectives:**

* Use a Dockerfile to create a Docker container having a custom decision tree algorithm from the widely used scikit-learn machine learning package
* Pushed it to Amazon ECR
* Using that container to trained model with a sample dataset
* Deployed it to an HTTP endpoint

**Parts of the container:**

In the container directory are all the components needed for packaging:

|-- Dockerfile

|-- build\_and\_push.sh

`|-- decision\_trees

|-- nginx.conf

|-- predictor.py

|-- serve

|-- train

|-- wsgi.py

The usage and description of these:

* Dockerfile describes how to build the Docker container image, more details below.
* build\_and\_push.sh is a script that uses the Dockerfile to build the container images and then push it to ECR, we’ll invoke the commands directly.
* decision\_trees is the directory which contains the files that will be installed in the container.
* local\_test is a directory that shows how to test this new container on any computer that can run Docker, including an Amazon SageMaker notebook instance. Using this method, one can quickly iterate using small datasets to eliminate any structural bugs before using the container with Amazon SageMaker.

In simple words, we only install five files in the container which are:

* nginx.conf is the configuration file for the nginx front-end. Generally, one should take this file as-is.
* predictor.py is the program that actually implements the Flask web server and the decision tree predictions for this app. The actual prediction parts are customized here as per the application. All the processing is here in this file, but one may choose to have separate files for implementing the custom logic.
* serve is the program started when the container is started for hosting. It simply launches the gunicorn server which runs multiple instances of the Flask app defined in predictor.py. One take this file as-is.
* train is the program that is invoked when the container is run for training, this program is modified to implement the training algorithm.
* wsgi.py is a small wrapper used to invoke the Flask app. One should be able to take this file as-is.

In summary, the two files that will have to be changed for any other application are train and predictor.py

Below are the explanations for Sagemaker studio in AL-ML account- Mumbai region: [default-1638530055881](https://ap-south-1.console.aws.amazon.com/sagemaker/home?region=ap-south-1#/studio/d-8s1625oogm59/user/default-1638530055881)

Cell 1- Prints the contents of docker file.

Cell 2- Install sagemaker studio image build to build docker image.

Cell 3- Docker build command which will give image URI, it will take 6-7 minutes to complete.

In case of error- Update trust policy as below which will allow usage of codebuild for git repository.

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"Service": "sagemaker.amazonaws.com"

},

"Action": "sts:AssumeRole"

},

{

"Effect": "Allow",

"Principal": {

"Service": "codebuild.amazonaws.com"

},

"Action": "sts:AssumeRole"

}

]

}

Cell 4- Bucket and role defined.

Cell 5- Session connection parameters defined.

Cell 7- Starting the training job, 47 secs training time.

Cell 8- Host mode to get real time prediction from https endpoint, takes few minutes.

Cell 9-10- Then clearance of resources setup.