**DSBA 6190 | Distributed Computing Lab**

**Description:** The goal of the “containerization” portion of this lab is to learn how to build a Docker image that can train a machine learning model from data housed in your data lake and to execute this model training from a Kubernetes cluster. Then, the Databricks portion of this lab will walk you through a similar machine learning exercise, but on a distributed architecture using Apache Spark’s MLlib.

**Notes:**

* Only provision the requested resources using the defined settings. Remember, the class cloud budget is everyone’s responsibility.

## Steps:

1. Build Your Docker Image. (EACH GROUP MEMBER)
2. Push to the Azure Container Registry. (EACH GROUP MEMBER)
3. Connect to the Azure Kubernetes Service. (EACH GROUP MEMBER)
4. Run Your Container in Kubernetes and Train a Model. (EACH GROUP MEMBER)
5. Complete Distributed ML Training on Databricks with Apache Spark. (EACH GROUP MEMBER)

# Part 1: Containerization with Docker and Kubernetes

##### Example files for this part of the lab can be found [here](https://github.com/colbyford/DSBA6190-CloudComputing/tree/master/3%20-%20Distributed%20Computing/containerization/docker_example).

### Step 1: Build Your Docker Image

First, you’ll need to create a Dockerfile that can run your machine learning model training.

The idea is that you will create an image that contains the logic to read in some data (such as a CSV file) from the data lake and then train a machine learning model (e.g., a sklearn regression model) and then output the trained model back into the data lake.

A screenshot of a computer program

Description automatically generatedProvide a screenshot of your completed Dockerfile (or a link to the file in a GitHub repository, optional). Also, describe the dataset that you’re using and the machine learning task/algorithm you’ve chosen.

I’ve chosen the 'iris' dataset for this lab. The dataset contains six columns:

* 'Id': 'int64'
* 'SepalLengthCm': 'float64'
* 'SepalWidthCm': 'float64'
* 'PetalLengthCm': 'float64'
* 'PetalWidthCm': 'float64'
* 'Species': 'str'

I’m using linear regression to predict SepalLengthCm, using SepalWidth, PetalLength, and PetalWidth as descriptive variables, with SepalLengthCm as the target variable.

[Linked here in the Github Repo](https://github.com/RileyLePrell/6190_Cloud_Comput_for_DA/tree/main/Lab%202%20Distributed%20Computing)

### Step 2: Push to the Azure Container Registry

Once you’ve tested your container locally, push it to the class container registry.

Example commands:

* az acr login --name crdsba6190deveastus001
* docker tag <image name> <registry name>.azurecr.io/<image name>:<tag>
  + docker tag instructor\_sklearn crdsba6190deveastus001.azurecr.io/instructor\_sklearn:latest
* docker push <registry name>.azurecr.io/<image name>:<tag>
  + docker push crdsba6190deveastus001.azurecr.io/instructor\_sklearn:latest

Provide a screenshot of your container image in the container registry.

A screenshot of a computer

Description automatically generated

### Step 3: Connect to the Azure Kubernetes Service

Next, you’ll need to grab the credentials for the class Azure Kubernetes Service. Navigate to the service in the Azure Portal and click the Connect button.

A screenshot of a computer

Description automatically generated

Copy and paste the commands into your local terminal/command prompt.

A screenshot of a computer

Description automatically generated

### Step 4: Run Your Container in Kubernetes and Train Your Model

Lastly, create a pod specification YAML file that will apply the image and create a pod in the cluster. Then, remote into your pod and execute your machine learning task.

Example commands:

* kubectl apply -f <POD YAML FILE>
  + kubectl apply -f example\_pod.yml
* kubectl exec -it <pod\_name> -- /bin/bash
  + kubectl exec -it instructor-test-01 -- /bin/bash
  + (Once in the pod, then run “python train.py”, etc.)

Provide a screenshot of your pod running in the Kubernetes cluster.

A white background with black border

Description automatically generated

Provide a screenshot of your trained machine learning model in the class data lake.

A screenshot of a computer

Description automatically generated

# Part 2: Distributed Machine Learning on Azure Databricks

### Step 5: Follow the steps in the Azure Databricks notebook.

Turn in this Word document on Canvas for Part 1.

For Part 2, leave your notebook in your user folder in the Azure Databricks workspace.