Programming Assignment 2

A. Introduction

Goal: The purpose of this individual assignment is to learn how to develop parallel machine learning (ML) applications in Amazon AWS cloud platform. Specifically, you will learn: (1) how to use Apache Spark to train an ML model in parallel on multiple EC2 instances; (2) how to use Spark's MLlib to develop and use an ML model in the cloud; (3) How to use Docker to create a container for your ML model to simplify model deployment.

GitHub: https://github.com/RohitD007/CS643_CloudProgramming/blob/main/Wine_Prediction.py
DockerHub: https://hub.docker.com/r/dsouzarohit/cs643_rgd3_programming_assignment2

B. Parallel Training Implementation using flintrock cluster on EC2

Created instance-> EC1

Step-1: Login and access to AWS services.
Step-2: Choose AMI.
Step-3: Choose EC2 Instance Types.
Step-4: Configure Instance.
Step-5: Add Storage.
Step-6: Tag Instance.
Step-8: Review Instances Run Instance on PUTTY for Windows

Run Instance on PUTTY for Windows

Step-1:Download PuTTY.
Step-2:PuTTYGEN will be downloaded by-default with PuTTY.
Step-3:PuTTYGEN is used to convert .pem key file into .ppk file.
Step-4:Open PuTTY
Step-5:Enter the Host name as mentioned in Public key for the instances (ec1)
Step-6: Click Save.
Step-7:Set Port number as 22.
Step-8:Click Save
Step-9:Goto SSH>>Authentications and browse the .ppk key created by puttygen
Step-10:Click Open

AWS CLI Credentials setup

Step-1: On your AWS CLI ,click show
Step-2:Copy the credentials
Step-3:Paste the credentials using vi ~ /.aws/credentials in EC1
Step-4:Make sure you copy credentials each time you create a new session as they expire after session expires

Flintrock Setup

As per the instructions given in the class

pip3 install flintrock

Configure the flintrock using the below command.

flintrock configure

Open the config.yaml file in editor mode and set the keyname to the pem filename(without .pem) and path to the pem file path.

Make sure the config.yaml file installs both Spark and Hadoop on the masters and workers

Create a cluster with five m4-large instances where four of them are workers and one of them is the master.

flintrock launch <cluster_name>

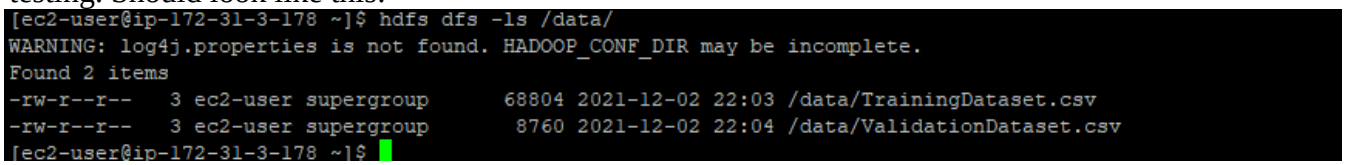
Log into spark master using:

flintrock login < cluster_name>.

inbound Rules

After Creating the Cluster add the SSH inbound rule to the master node for port 22

Add to the root of HDFS a folder called data and place in there the files to be used for training, validation, and/or testing. Should look like this:



Submit the training job with the following command:

spark-submit --master spark://ip-172-31-3-178.ec2.internal:7077 Wine_Training.py hdfs:///data/TrainingDataset.csv hdfs:///model

I have used RandoForest algorithm to train and predict the accuracy

We then want to grab our model from HDFS so we use the get command:

hdfs dfs -get /model

This is our **saved model** that we can then use to create our prediction application.

Single Machine Prediction Application

We just need to execute Spark in local master mode to only use the master node for prediction. This can be done with the following command:

spark-submit --master local[*] Wine_Prediction.py file:///home/ec2-user/ValidationDataset.csv file:///home/ec2-user/model

Where you must specify the file path to the dataset being used in `argv[1]` and the model folder in `argv[2]`.

Output would look like this after all the log messages:

```
21/12/03 22:11:14 INFO TaskSchedulerImpl: Killing all running tasks in stage 11: Stage finished
21/12/03 22:11:14 INFO DAGScheduler: Job 7 finished: collectAsMap at MulticlassMetrics.scala:61, took 0.761759 s
Accuracy = 0.55625
21/12/03 22:11:14 INFO FileSourceStrategy: Pushed Filters:
21/12/03 22:11:14 INFO FileSourceStrategy: Post-Scan Filters:
```

```
21/12/03 22:11:15 INFO DAGScheduler: Job 8 is finished. Cancelling potential speculative or zombie tasks for this job
21/12/03 22:11:15 INFO TaskSchedulerImpl: Killing all running tasks in stage 13: Stage finished
21/12/03 22:11:15 INFO DAGScheduler: Job 8 finished: collectAsMap at MulticlassMetrics.scala:61, took 0.530652 s
F1 = 0.531967
21/12/03 22:11:15 INFO SparkContext: Invoking stop() from shutdown hook
21/12/03 22:11:15 INFO SparkUI: Stopped Spark web UI at http://4a2820f51ff0:4040
21/12/03 22:11:15 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
21/12/03 22:11:15 INFO MemoryStore: MemoryStore cleared
```

Creating Docker Image for Prediction Application

1. Install most recent docker engine package: sudo amazon-linux-extras install docker or sudo yum install docker
2. Start docker : **sudo service docker start**
3. Adding ec2-user to docker group: **sudo usermod -a -G docker ec2-user**
4. Exit the flintrock cluster and login again.
5. verify ec2-user: **docker info**
6. After setting up login into your docker using : docker login. Here you have to provide you credentials for dockerhub.
7. Build the docker image using :

   docker build -t docker build -t dsouzarohit/cs643_rgd3_programming_assignment2 .

8. Run the image using:

   docker run dsouzarohit/cs643_rgd3_programming_assignment2 driver Wine_Prediction.py ValidationDataset.csv model

9. Pushing the image to Docker hub repository:

   docker push dsouzarohit/cs643_rgd3_programming_assignment2

Docker container for prediction application

1. Launch your ec2-instance and then step-up docker using the above steps.
2. Copy the **ValidationDataset.csv** into location where you will pull the repository
3. Pull the image from repositroy:

   docker pull dsouzarohit/cs643_rgd3_programming_assignment2

4. Run the image using :

   docker run dsouzarohit/cs643_rgd3_programming_assignment2 driver Wine_Prediction.py ValidationDataset.csv model