**TASK 1 : Parallel training on 4 ec2 Instances**

This project is implemented with Spark data frames api and MLib libraries, with this Native spark implementation application is automatically parallelized and distributed natively.

Wine prediction application is developed using Spark DataFrames and MLlib. Running it on AWS EMR cluster automatically parallelize and distribute job execution. Hadoop Distributed Files system is used for locating dataset files and for storing trained models.

**How to create EMR Cluster ?**

* After logging into AWS console,
* Go to EMR Service & Create Cluster
  + Enter cluster name
  + Launch Mode cluster
  + Vendor Amazon
  + Release emr-5.3.10
  + Select spark application – version 2.4.5
  + Hardware Configurations
    - select the instance type
    - number of instances to 4 (1 master 3 slaves)
  + select the ec2 key pair or generate one to access the master node.
  + click on create cluster.

**Graphical user interface, text, application, email

Description automatically generated**

**­­­­Upload files to EMR Cluster Master node**

* Once Cluster goes into waiting state, copy master node dns address and open command prompt on local machine.
* Open sftp connection to master node

sftp -i cluster-keypair.pem [hadoop@ec2-3-128-26-180.us-east-2.compute.amazonaws.com](mailto:hadoop@ec2-3-128-26-180.us-east-2.compute.amazonaws.com)



* Upload TrainingDataSet.csv, ValidationDataset.csv and winequality-1.0.jar to master node.  
  Text

  Description automatically generated

**­­­­SSH Master node :**

ssh -i ~/cluster-keypair.pem hadoop@ec2-3-128-26-180.us-east-2.compute.amazonaws.com

**Copy files to HDFS :**

* Now all files are on our master node we wanna move them to HDFS so that all slave nodes can also access them and we don’t have to manually copy them to all ec2 nodes.
* Use this command to copy files from Master node to HDFS.
  + hadoop fs -put TrainingDataset.csv /user/hadoop/TrainingDataset.csv
  + hadoop fs -put ValidationDataset.csv /user/hadoop/ValidationDataset.csv
* Use this command to verify if files are successfully copied to HDFS
  + hdfs dfs -ls -t -R

**Launch ModelTrainer application :**

* Now everything is done, we want to lauch apache-spark application on EMR cluster.
* Execute following command to run application
  + spark-submit winequality-1.0.jar

A screenshot of a computer

Description automatically generated

Fig : Console output

We can confirm job execution by going to monitor tab then spark dashboard.

Graphical user interface, application, table

Description automatically generated

Fig : Spark jobs completing confirmation

* This will create a TrainingModel folder and store trained model to it.
* Verify model is created by executing following :
  + hdfs dfs -ls -t -R

Text

Description automatically generated

* Now copy This folder back to our master node using following

hdfs dfs -copyToLocal TrainingModel /home/hadoop/wine

* Training part is done here, so we need training files to be transferred to our local environment so that we can use them to predict on single ec2 with or without docker.
* Make a tar.gz zip of folder so that we can download it
  + tar czf model.tar.gz TrainingModel

Text

Description automatically generated

* in our sftp session execute following to download mode.tar.gz on local machine
  + get wine/model.tar.gz



**---------------------------- END OF TASK 1 ----------------------------**

**TASK 2 : Predict wine quality on single ec2 instance**

At this stage we are interested in executing prediction code on single ec2 instance. For that we need TestDataset.csv, wine-quality-predict.jar, model.tar.gz (from task1)

**Ec2 instance Create:**

* After logging into AWS console,
* Go to EC2 -> launch instance, select AMI : **ami-0053f34b22df259f2**
* Select keypair and launch it.

**Ec2 instance pre configuration:**

* Do ssh to ec2 instance public dns,
  + ssh -i ec2-A.pem [ec2-user@ec2-54-158-81-112.compute-1.amazonaws.com](mailto:ec2-user@ec2-54-158-81-112.compute-1.amazonaws.com)

Text

Description automatically generated

* **Install SCALA:** 
  + wget http://downloads.typesafe.com/scala/2.11.6/scala-2.11.6.tgz
  + tar -xzvf scala-2.11.6.tgz

Text

Description automatically generated

* + Update PATH environment variable:
    - vim ~/.bashrc
    - copy following lines into file and then save it
      * export SCALA\_HOME=/home/ec2-user/scala-2.11.6
      * export PATH=$PATH:/home/ec2-user/scala-2.11.6/bin
    - source ~/.bashrc

Text

Description automatically generated

* **Install SPARK:** 
  + wget https://archive.apache.org/dist/spark/spark-2.4.5/spark-2.4.5-bin-hadoop2.7.tgz
  + sudo tar xvf spark-2.4.5-bin-hadoop2.7.tgz -C /opt
  + sudo chown -R ec2-user:ec2-user /opt/spark-2.4.5-bin-hadoop2.7
  + sudo ln -fs spark-2.4.5-bin-hadoop2.7 /opt/spark
  + Update PATH Environment
    - vim ~/.bash\_profile
    - copy following lines into file and then save it
      * export SPARK\_HOME=/opt/spark
      * PATH=$PATH:$SPARK\_HOME/bin
      * export PATH
    - source ~/.bash\_profile
* **Upload trained model and jar files :** 
  + Login to ec2 instance using sftp
  + sftp -i ec2-A.pem [ec2-user@ec2-54-158-81-112.compute-1.amazonaws.com](mailto:ec2-user@ec2-54-158-81-112.compute-1.amazonaws.com)
  + put wine-quality-predict-1.0.jar
  + put TestDataset.csv
  + put model.tar.gz

**Graphical user interface, text, application

Description automatically generated**

* **Extract model.tar.gz :** 
  + tar -xzvf model.tar.gz

Text

Description automatically generated

* **Disable unnecessary log4j :** 
  + cp $SPARK\_HOME/conf/log4j.properties.template $SPARK\_HOME/conf/log4j.properties
  + vi $SPARK\_HOME/conf/log4j.properties
  + (on line 19 of the file, change the log level from INFO to ERROR)
  + log4j.rootCategory=ERROR, console
  + Save the file and exit the text editor.
* **Run wine-predict application :**
  + spark-submit wine-quality-predict-1.0.jar

**Graphical user interface, text

Description automatically generated**

Fig : F1 Score printed on ec2

**---------------------------- END OF TASK 2 ----------------------------**

**TASK 3 : Predict wine quality using docker**

For Predicting a wine quality on TestDataset.csv using docker. We need to have a full local filepath of TestDataset.csv and provide it as an input argument while running a docker. So that TestDataset.csv can be copied local docker environment before running.

Test filename has to be **TestDataset.csv**and file has to be placed under data/ directory of container. To do this use -v parameter to map volumes.

Execute following command.

* docker pull nieldeokar/wine-prediction-mvn:1.0
* docker run -v /Users/niel/Desktop/US\_Docs/sem3/Cloud-computing/Assignments/2/nrd32-wine-prediction/data:/data nieldeokar/wine-

prediction-mvn:1.0 /data/TestDataset.csv

* In general use following format :

docker run -v [local\_testfile\_directory:/data] nieldeokar/wine-

prediction-mvn:1.0 /TestDataset.csv

A picture containing text

Description automatically generated

Fig : F1 Score printed on local machine using docker