**Shubh Patel**

**CS 643\_851**

**Programming Assignment 2**

**ReadMe**

**# Submitted Files:**

* ReadMe.docx (this document)
* **GitHub link –** <https://github.com/Shubh-A-Patel/Wine-Prediction>
* **Docker Container link -** <https://hub.docker.com/r/shubhpatel/prp2/tags>

**# Setting up the AWS cloud environment**

*To setup the resources:*

* **Creating EMR Cluster –** We need to launch 1 **EMR Cluster** with 1 Master Node and 3 Slaves configuration.
* For EMR, set the following configurations:
  + Launch Mode cluster
  + Vendor Amazon
  + Release emr-5.30.10
  + Select spark application
  + Hardware Configuration:
    - Select the instance type
    - Number of instances to 4 (1 Master Node and 3 Slaves)
  + Select the ec2 key pair for accessing the Master Node
* For EMR Cluster, I set security group inbound rules to Type SSH, Protocol TCP, and Source MYIP.
* **EC2 Instances -** We need to launch one **Amazon Linux 2 AMI** EC2 instances for prediction of out code.
* For EC2 instances, I set security group inbound rules to Type SSH, Protocol TCP, and Source MYIP.
* **S3** (optional) **-** We also need to create a **S3** using AWS console for storing some files. I named it, **prp2**

**# Creating project on Master Node**

*To setup the running environment on the Master Node:*

* We need to install **Java** and **maven** for compiling and running the code.
* To install them, use the following commands:

1) sudo wget https://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo

2) sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo

3) sudo yum install -y apache-maven

4) sudo yum update

5) To make java switch to specific version that we want to use (we are using java JDK 1.8.0)

- sudo yum install java-1.8.0-openjdk-devel

- sudo update-alternatives --config java

- now select option where java JDK is 1.8.0

* Now, check if they are installed correctly:

java -version

The above command should display the java version.

mvn --version

The above should display the maven version

*To setup the AWS SDK Keys in the Master Node:*

* We need AWS SDK Keys to grant access to the AWS resources from the code. These keys should be overwritten into **~/.aws/credentials** file in both the instances.
* The keys look like below:

[default]

aws\_access\_key\_id=**<KEY\_ID\>**

aws\_secret\_access\_key=**<SECRET\_ACCESS\_KEY\>**

aws\_session\_token=**<SESSION\_TOKEN\>**

*Now, we need to create a maven project in the Master Node:*

* To create a maven project, type the following command:

mvn archetype:generate -DgroupId=com.mycompany.app -DartifactId=my-app -DarchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4 -DinteractiveMode=false

*Now, we can type/copy the java code, xml, and datasets into the Master Node:*

* The java and xml codes are in the GitHub and link is provided above in this file
* The datasets, you can download it from S3 object that we created earlier
* TrainingDataset.csv - wget “https://prp2.s3.amazonaws.com/TrainingDataset.csv”
* ValidationDataset.csv – wget “https://prp2.s3.amazonaws.com/ValidationDataset.csv”

*Now, we need to push the datasets to slaves, so they can use it while running the code,*

* Type the following command in the same directory where you saved the dataset files
* hadoop fs -put TrainingDataset.csv /user/hadoop/TrainingDataset.csv
* hadoop fs -put ValidationDataset.csv /user/hadoop/ValidationDataset.csv

**# Running the code**

**Running code in Master Node:**

* Change directory to where the pom.xml file is located
* Create a **run.sh** shell script and copy/type the code to run the shell script
  + The codes are submitted with this file
  + You can also skip this step and run it manually
* Now, you can run the shell script using the following command

sh run.sh

* If you did not create the shell script than run the following command:

1. mvn clean compile assembly:single
2. spark-submit target/NameOfJarFile.jar
   * Here change NameOfJarFile.jar with your name of the jar file

* Now, try to run the code for Predict.java

1. For this, you need to change mainClass in pom.xml file

* Change “<mainClass>com.mycompany.app.ModelTrainer</mainClass>” to “<mainClass>com.mycompany.app.Predict</mainClass>”

1. Now, run the following command to ganarate jar file

* mvn clean compile assembly:single

**Saving files from Master Node:**

* Type to following command to get the TrainingModel from slaves
* hdfs dfs -ls -t -R
* hdfs dfs -copyToLocal TrainingModel /home/hadoop
* Now, make tar file to download the model using following command
* tar czf model.tar.gz TrainingModel
* In a new terminal login to master node using sftp
* Go to my-app/target and download jar file using get command
* Go to home/Hadoop and download model.tar.gz file using get command

**# Configure the EC2 instance**

*To setup the running environment on EC2 instance:*

* We need to install **Java** and **maven** for compiling and running the code.
* To install them, use the following commands:

1) sudo wget https://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo

2) sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo

1. sudo yum install -y apache-maven
2. wget http://downloads.typesafe.com/scala/2.11.6/scala-2.11.6.tgz
3. tar -xzvf scala-2.11.6.tgz
4. Change the following things:

* Edit the file using vim ~/.bashrc and change following things
  + export SCALA\_HOME=/home/ec2-user/scala-2.11.6
  + export PATH=$PATH:/home/ec2-user/scala-2.11.6/bin
* Now, save the file and type source ~/.bashrc

1. wget https://archive.apache.org/dist/spark/spark-2.4.5/spark-2.4.5-bin-hadoop2.7.tgz
2. sudo tar xvf spark-2.4.5-bin-hadoop2.7.tgz -C /opt
3. sudo chown -R ec2-user:ec2-user /opt/spark-2.4.5-bin-hadoop2.7
4. sudo ln -fs spark-2.4.5-bin-hadoop2.7 /opt/spark
5. Now update the PATH Environment

* Edit the file using vim ~/.bash\_profile and change following things
  + export SPARK\_HOME=/opt/spark
  + PATH=$PATH:$SPARK\_HOME/bin
  + export PATH
* Now, save the file and type source ~/.bash\_profile

4) sudo yum update

5) To make java switch to specific version that we want to use (we are using java JDK 1.8.0)

- sudo yum install java-1.8.0-openjdk-devel

- sudo update-alternatives --config java

- now select option where java JDK is 1.8.0

* Now, check if they are installed correctly:

java -version

The above command should display the java version.

mvn --version

The above should display the maven version

*To setup the AWS SDK Keys in the EC2 instance:*

* We need AWS SDK Keys to grant access to the AWS resources from the code. These keys should be overwritten into **~/.aws/credentials** file in both the instances.
* The keys look like below:

[default]

aws\_access\_key\_id=**<KEY\_ID\>**

aws\_secret\_access\_key=**<SECRET\_ACCESS\_KEY\>**

aws\_session\_token=**<SESSION\_TOKEN\>**

*Now, we can get the TrainingModel, jar file and dataset into the EC2 instance*

* In a new terminal login to ec2 instance using sftp
* Upload a model.tar.gz containing TrainingModel, a jar file and a Testing Dataset a using put command
* For Now, we can use Validation Dataset as Test Dataset
  + I renamed File to TestDataset.csv
* Now extract model.tar.gz using following command
  + tar -xzvf model.tar.gz

*Now, we need to change/disable unnecessary log4j lines*

* 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

**# Running the code**

**Running code in EC2 instance:**

* Run the code using following command
  + spark-submit NameOfJarFile.jar
    - Here change NameOfJarFile.jar with your name of the jar file

**Running code in Docker:**

* First, Type following code to setup/install docker:
  + sudo yum update -y
  + sudo amazon-linux-extras install docker
  + sudo yum install docker
* Second, start the Docker service:
  + sudo service docker start
* For Predicting using Docker, Run the code using following command:
  + docker pull shubhpatel/prp2
  + docker run -v FileTestDataSet shubhpatel/prp2
    - Here change FileTestDataSet with your name of the Test Dataset file