# CS643 Programming Assignment 2

**John Daudelin**

**12/8/2020**

**CS643-851**

**GitHub Link:** <https://github.com/johndaudelin/cs643-final-project>

**DockerHub Link:** <https://hub.docker.com/repository/docker/johndaudelin98/cs643-prediction>

# **Training Setup:**

Create an S3 Bucket by navigating in AWS to Services 🡪 S3 🡪 Create Bucket. Name the bucket “cs643johndaudelin,” un-select the “Block all public access” box (which may require also selecting the “I acknowledge…” statement that appears) and accept all the other default options and create the bucket. Click on the bucket and hit “Upload.” Then, upload the “TrainingDataset.csv” and “ValidationDataset.csv.”

# **Training:**

Navigate on AWS to Services 🡪 EMR 🡪 Create Cluster. Click on Advanced Options. Choose a release of “emr-5.32.0.” For Software Configuration, select Hadoop, JupyterEnterpriseGateway, Zeppelin, Spark, and Pig. Don’t change anything else and click Next. Under “Cluster Nodes and Instances,” type 3 for the number of core node instances (in addition to the 1 master node), delete the Task node group, and keep the default instance type for both the master and core nodes as “m5.xlarge.” Click Next. Name the cluster “CS643 Cluster.” De-select “Logging” and click Next. Under “Security Options,” select “Proceed without an EC2 key pair.” Hit “Create cluster”!

This may take a few minutes for the cluster to boot up. Meanwhile, click on “Notebooks” in the left pane, and then hit “Create Notebook.” Name the notebook “training”, and then for “Cluster,” hit “Choose” and select the cluster that you created in the previous step (“CS643 Cluster”). Keep all the other default setting and hit Create notebook. Wait till the status changes from “Starting” to “Ready,” and then click “Open in Jupyter.” Click on the notebook that you see called “training.ipynb”, and then click on Kernel 🡪 Change Kernel 🡪 PySpark on the toolbar at the top.

Copy the code from “training.py” (located [here](https://github.com/johndaudelin/cs643-final-project/blob/main/training.py) in my GitHub repository) into the first cell of the notebook. Hit Run. You should see the output of both the LogisticRegression Model and the RandomForestClassifier Model after they are trained on all four nodes.

# **Prediction Setup:**

On the AWS Console, navigate to Services 🡪 EC2 🡪 Launch Instances. Select the (presumably first) AMI, “Amazon Linux 2 AMI… ami-04d29b6f966df1537.” Select the t2.large type (useful for Docker purposes). Keep all the rest of the default options and click “Review and Launch.”

Under Security Groups, click "Edit Security Groups" and add one rule for SSH and then one Custom TCP Rule with a specified port of 8888. Specify “My IP” as the source for both rules. Hit Review and Launch.

Click Launch. On the dialog that pops up, select "Create a new key pair" and name it "cs643-final". Hit "Download key pair." Hit Launch Instances, and then hit View Instances. You will probably see a status of "Pending" for the Instance State of the EC2. While waiting for this to switch to "Running," open a terminal and move the .pem file you downloaded to your home directory. Run the following command to set the correct permissions for the .pem file:

$ chmod 400 cs643-final.pem

To connect to your EC2 instance (after it has started running), run the following command in your terminal (replacing <YOUR\_INSTANCE\_PUBLIC\_DNS> with the "Public IPv4 DNS" attribute of the EC2 instance):

$ ssh -i ~/cs643-final.pem ec2-user@<YOUR\_INSTANCE\_PUBLIC\_DNS>

Now, go back to your local terminal and navigate to the directory on your machine that holds the data files (TrainingDataset.csv and TestDataset.csv). Run the following commands to copy them over to your EC2:

$ scp -i ~/cs643-final.pem TrainingDataset.csv ec2-user@<YOUR\_INSTANCE\_PUBLIC\_DNS:~/

$ scp -i ~/cs643-final.pem TestDataset.csv ec2-user@<YOUR\_INSTANCE\_PUBLIC\_DNS:~/

SSH into your EC2 again. You should see the two data files located in the home directory. Run the following commands to move the data files in the home directory to the appropriate folder on the EC2:

$ sudo mkdir /app

$ sudo cp TrainingDataset.csv TestDataset.csv /app/

# **Prediction without Docker:**

While SSH’ed into your EC2 created in the previous step, run the following command to install and configure Java:

$ sudo yum install java-1.8.0-devel

$ export JAVA\_HOME=/usr/bin/java

Run the following commands to install Apache Spark and Python:

$ wget https://archive.apache.org/dist/spark/spark-2.4.7/spark-2.4.7-bin-hadoop2.7.tgz -P ~/server

$ cd server

$ sudo tar xvzf spark-2.4.7-bin-hadoop2.7.tgz

$ curl -O https://repo.anaconda.com/archive/Anaconda3-2020.11-Linux-x86\_64.sh

$ mv Anaconda3-2020.11-Linux-x86\_64.sh /tmp

$ cd /tmp

$ bash Anaconda3-2020.11-Linux-x86\_64.sh

Accept the license agreement and then type “yes” to begin anaconda installation. Type “yes” again when prompted and hit enter.

Now, add the following lines of code to the end of your ~/.bashrc file without modifying anything else in the file:

function snotebook ()

{

SPARK\_PATH=~/server/spark-2.4.7-bin-hadoop2.7

export PYSPARK\_DRIVER\_PYTHON="jupyter"

export PYSPARK\_DRIVER\_PYTHON\_OPTS="notebook"

export PYSPARK\_PYTHON=python3

$SPARK\_PATH/bin/pyspark --master local[2]

}

Run the following line of code to load the changes to your .bashrc file:

$ source ~/.bashrc

Close out of your SSH instance and SSH again into your EC2. Run the following code to install another required dependency, configure Jupyter with a password (choose any you like when prompted), and start up the Jupyter Notebook on the EC2.

$ pip install quinn

$ jupyter notebook password

$ jupyter notebook

Open up a new terminal tab on your local machine and run the following command to setup an SSH tunnel so that you can open the Jupyter Notebook on the EC2:

$ ssh -i ~/cs643-final.pem -N -f -L localhost:8888:localhost:8888 ec2-user@<YOUR\_INSTANCE\_PUBLIC\_DNS>

Now navigate to localhost:8888 in your browser and enter the Jupyter password that you set for your EC2 in the previous step. Once you’re in, create a new Python 3 notebook. In this notebook, paste the code from the prediction.py file (located [here](https://github.com/johndaudelin/cs643-final-project/blob/main/prediction.py) in my GitHub repository) into the first notebook cell. You will see several things printed as the code reads the data, formats it, trains our model, makes the predictions of wine quality for TestDataset, and finally outputs the F1 score.

The highest F1 score I observed (using the ValidationDataset as my TestDataset) was ~ 0.56.

# **Prediction with Docker:**

While SSH’ed into the EC2 created, run the following commands to install docker and then run my docker image on the EC2:

$ sudo yum install docker -y

$ sudo service docker start

$ sudo docker run -v /app/:/data johndaudelin98/cs643-prediction:version1

Note: While running the last command, if you receive an error message saying that you are out of disk space, you may need to run the following commands to free up space and then re-run the Docker image:

$ cd ~

$ rm -rf server

$ rm -rf anaconda3

$ sudo docker run -v /app/:/data johndaudelin98/cs643-prediction:version1

You should see the same output that you saw in the previous section. The final line shows the F1-score for the model’s predictions.