# **wine-quality-prediction - Kowsick Venkatachalapathi (31515945)**

wine-quality-prediction is a scala application which uses hadoop spark to train and predict wine quality in a distributed environment

Source code : <https://github.com/kv322/wine-quality-prediction>

Docker image : <https://hub.docker.com/repository/docker/kv322/wine-quality-prediction-0.0.1>

For more screenshots :

**To run prediction (using docker)**

docker run -v "$(pwd)"/input:/input kv322/wine-quality-prediction-0.0.1

**Note:**

1. pathToTestDataSetFile from current working directory
2. Test dataset filename should be “TestDataset.csv”

**To run prediction (without docker)**

Goto spark-2.4.0/bin directory and run below command

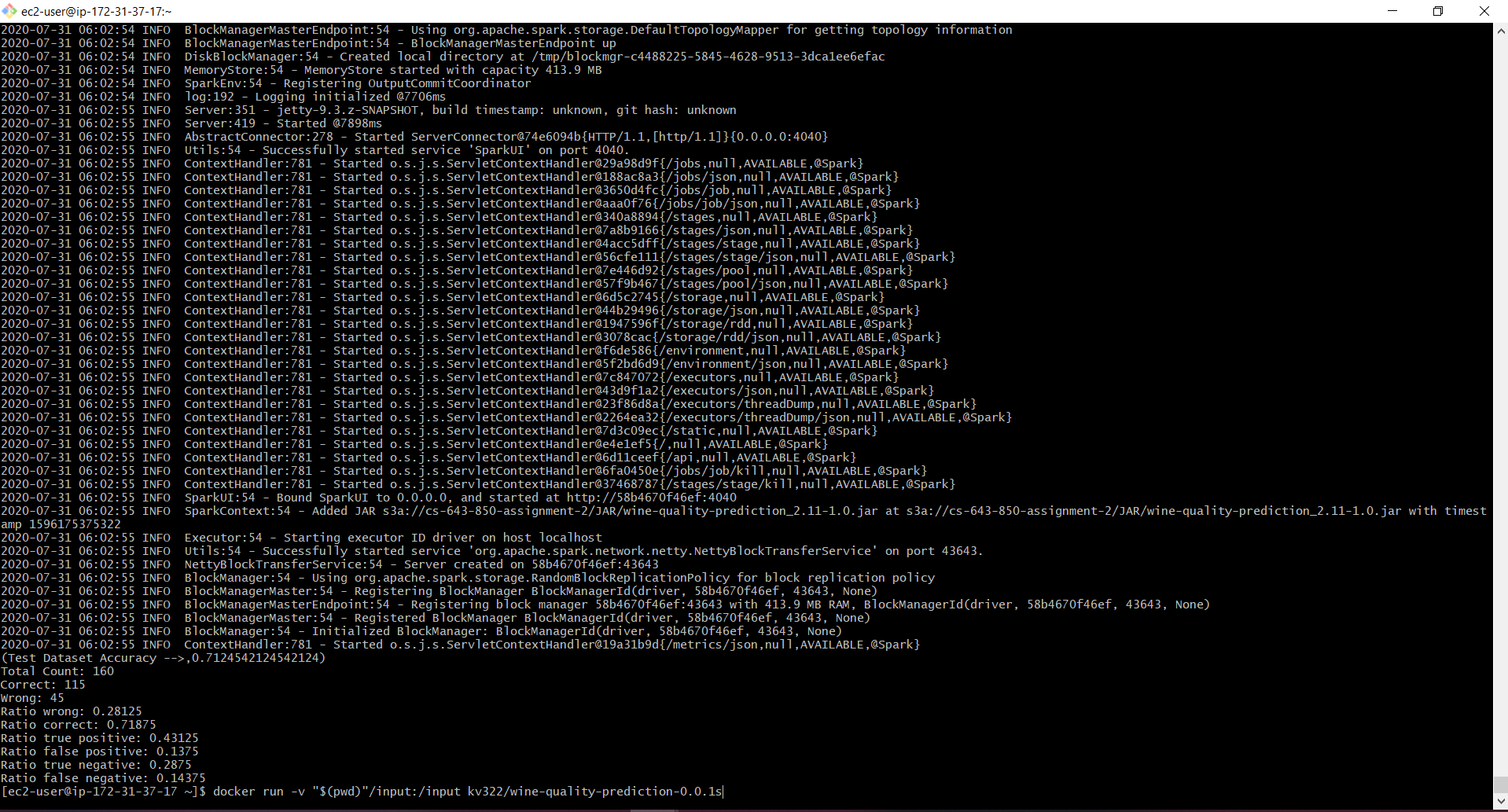
 ./spark-submit --class demo.common.WineQualityModelTrain  s3a://cs-643-850-assignment-2/JAR/wine-quality-prediction\_2.11-1.0.jar --master=local[\*] --run-type=prediction

--model-store-path=s3a://cs-643-850-assignment-2/output/model --testing-file-path=s3a:///”$(pwd)”/input/TestDataset.csv

Note:

Sample directory and path from current working directory

**Final output**



**Docker build**

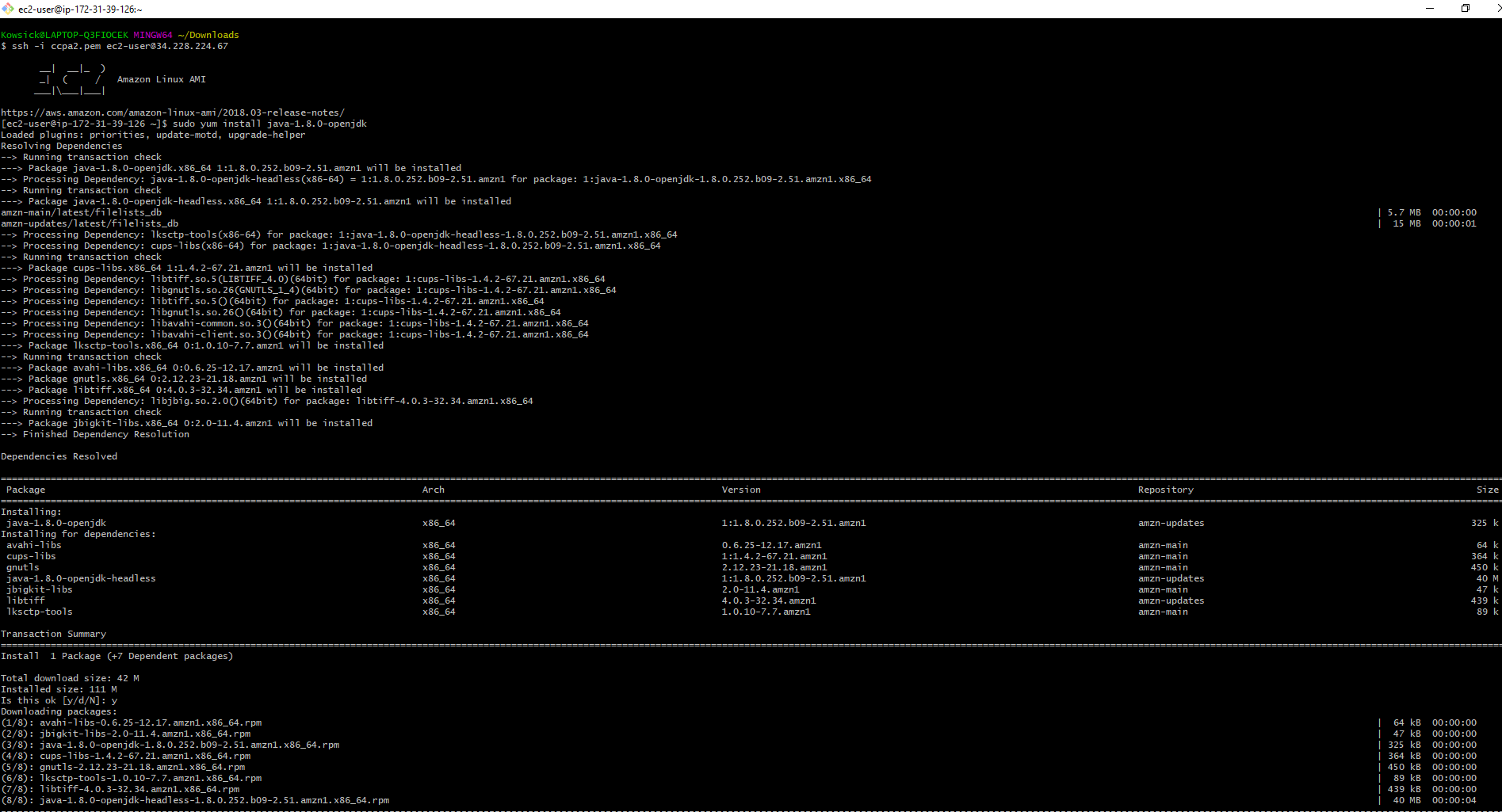
docker build --no-cache -t kv322/wine-quality-prediction-0.0.1 .

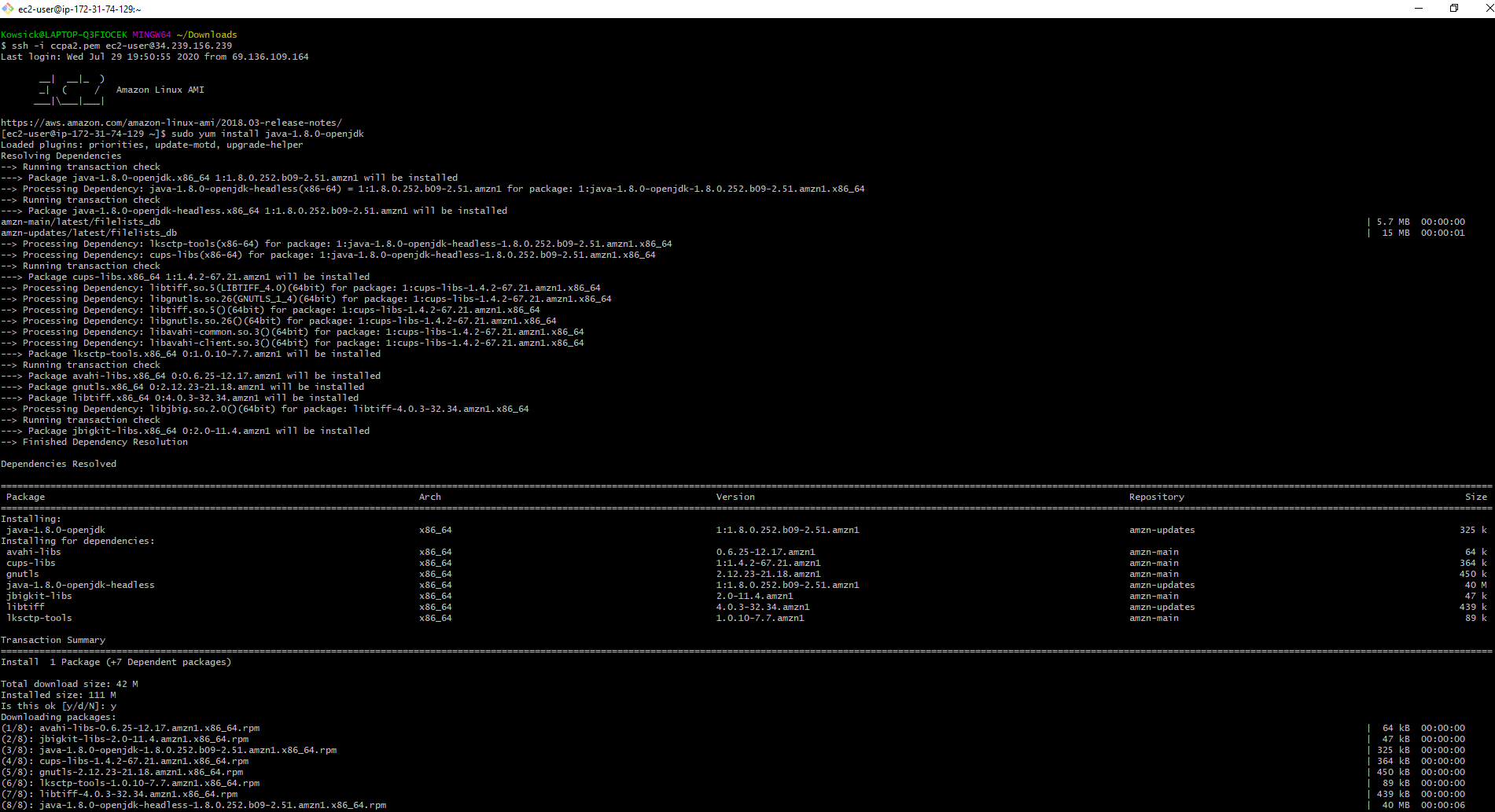
**Commands**

All installation commands should be run on all master and worker nodes

**JAVA installation**

sudo yum install java-1.8.0-openjdk  
sudo yum install java-devel  
sudo yum install java-1.8.0-openjdk-devel  
sudo yum remove java-1.7.\*





**Hadoop installation**

export HADOOP\_VERSION=2.8.4

export HADOOP\_HOME=${HOME}/hadoop-$HADOOP\_VERSION

export HADOOP\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop

export PATH=${HADOOP\_HOME}/bin:$PATH

curl -sL --retry 3 "http://archive.apache.org/dist/hadoop/common/hadoop-$HADOOP\_VERSION/hadoop-$HADOOP\_VERSION.tar.gz" | gunzip | tar -x -C /tmp/ && mv /tmp/hadoop-$HADOOP\_VERSION $HADOOP\_HOME && rm -rf $HADOOP\_HOME/share/doc

**Spark intallation**

export SPARK\_VERSION=2.4.0

export SPARK\_PACKAGE=spark-${SPARK\_VERSION}-bin-without-hadoop

export SPARK\_HOME=$HOME/spark-${SPARK\_VERSION}

export SPARK\_DIST\_CLASSPATH=$(hadoop classpath)

export PATH=${SPARK\_HOME}/bin:$PATH

Curl -sL --retry 3 "https://archive.apache.org/dist/spark/spark-${SPARK\_VERSION}/${SPARK\_PACKAGE}.tgz" | gunzip | tar x -C /tmp/ && mv /tmp/$SPARK\_PACKAGE $SPARK\_HOME

**Start master**

ssh -I ccpa2.pem ec2-user@54.82.211.128

cd spark-2.4.0-bin-without-hadoop/bin

./start-master.sh

**Start worker nodes**

**worker 1:**

ssh -I ccpa2.pem ec2-user@54.221.183.229

cd spark-2.4.0-bin-without-hadoop/bin

./start-slave.sh spark://ip-172-31-43-226.ec2.internal:7077

**worker 2:**

ssh -I ccpa2.pem ec2-user@54.242.155.83

cd spark-2.4.0-bin-without-hadoop/bin

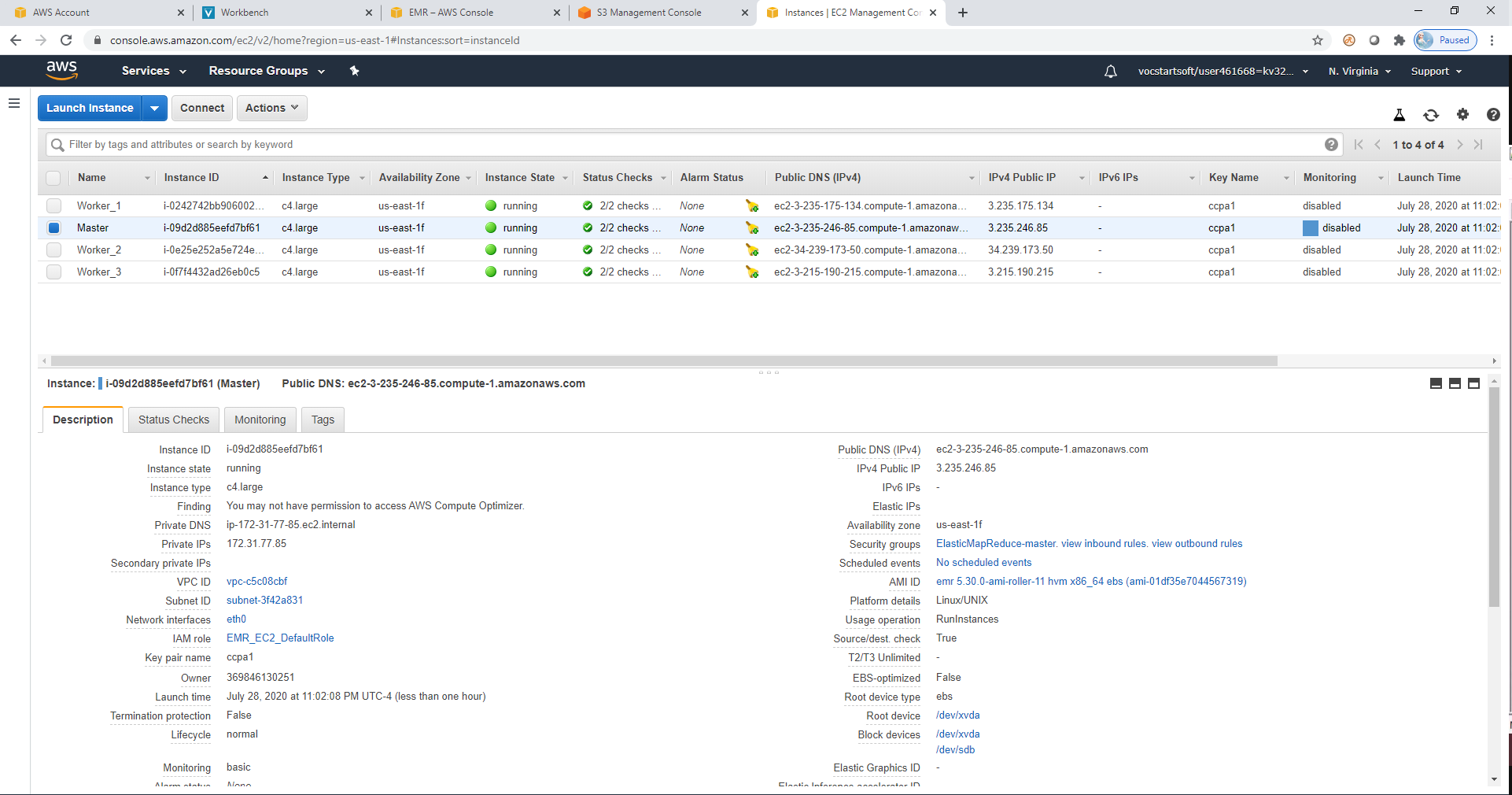
./start-slave.sh spark://ip-172-31-43-226.ec2.internal:7077

**worker 3:**

ssh -I ccpa2.pem ec2-user@34.228.224.67

cd spark-2.4.0-bin-without-hadoop/bin

./start-slave.sh spark://ip-172-31-43-226.ec2.internal:7077

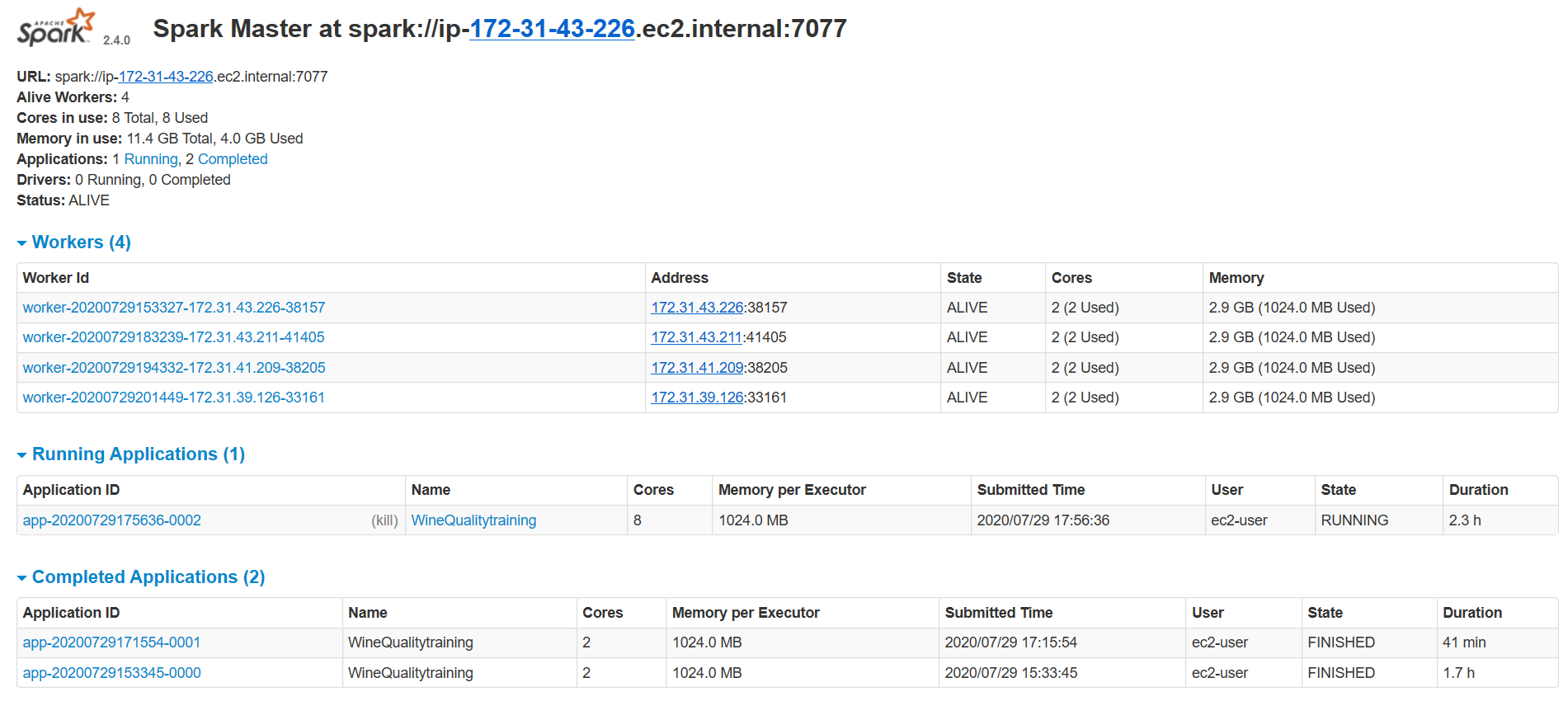


**To train**

*./spark-submit --class demo.common.WineQualityModelTrain  --conf spark.executor.extraClassPath=s3a://cs-643-850-assignment-2/JAR/aws-java-sdk-1.11.95.jar:s3a://cs-643-850-assignment-2/JAR/aws-java-sdk-core-1.11.95.jar:s3a://cs-643-850-assignment-2/JAR/aws-java-sdk-s3-1.11.95.jar:s3a://cs-643-850-assignment-2/JAR/hadoop-aws-2.8.4.jar --driver-class-path s3a://cs-643-850-assignment-2/JAR/\*.jar  --jars s3a://cs-643-850-assignment-2/JAR/\*.jar s3a://cs-643-850-assignment-2/JAR/wine-quality-prediction\_2.11-1.0.jar --master=spark://ip-172-31-43-226.ec2.internal:7077* ***--run-type=training*** *--training-file-path=s3a://cs-643-850-assignment-2/input/training/TrainingDataset.csv --validation-file-path=s3a://cs-643-850-assignment-2/input/validation/ValidationDataset.csv --model-store-path=s3a://cs-643-850-assignment-2/output/model --testing-file-path=s3a://cs-643-850-assignment-2/input/validation/ValidationDataset.csv*

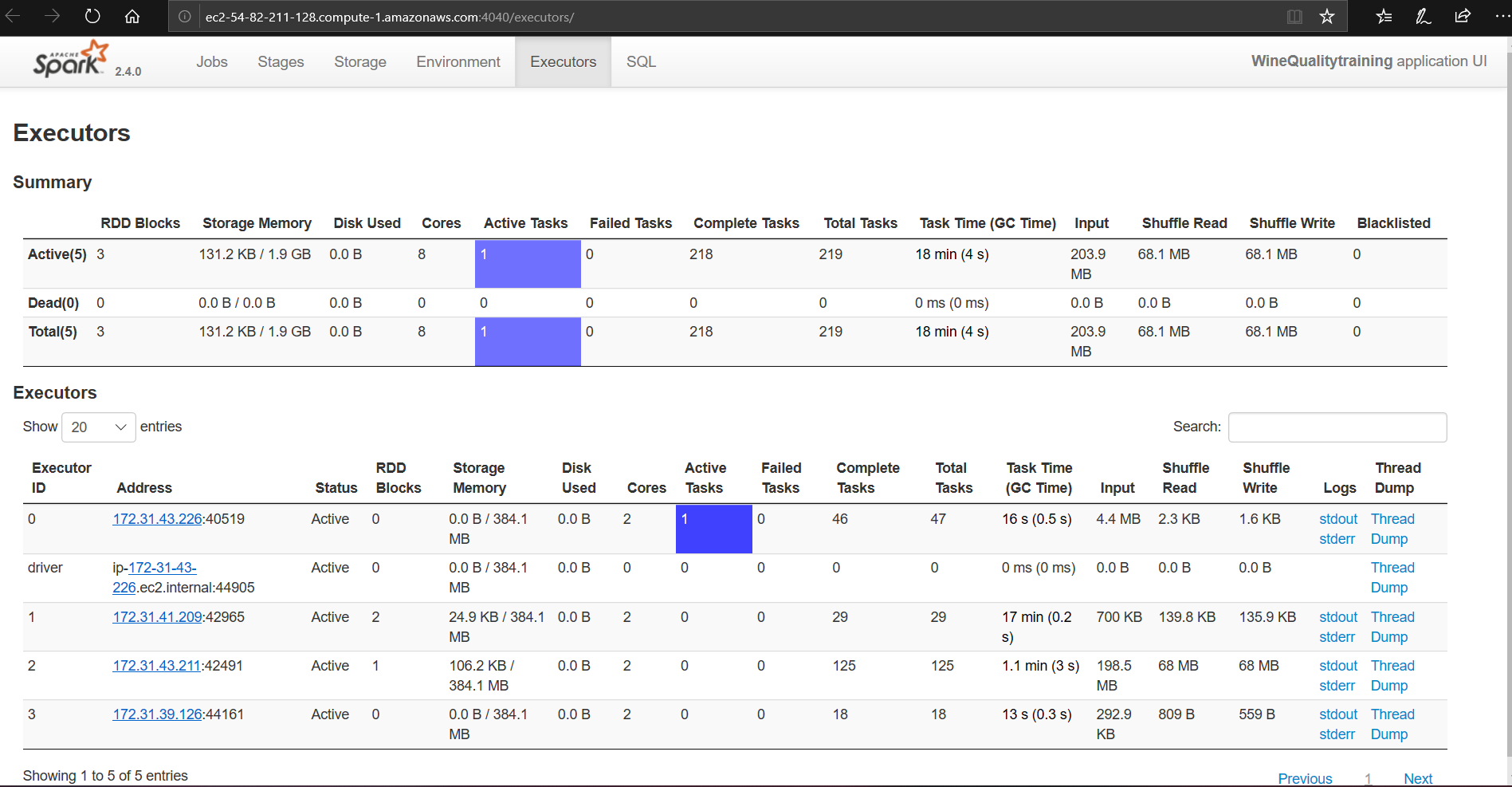
**Spark UI**

<http://ec2-54-82-211-128.compute-1.amazonaws.com:8080/>



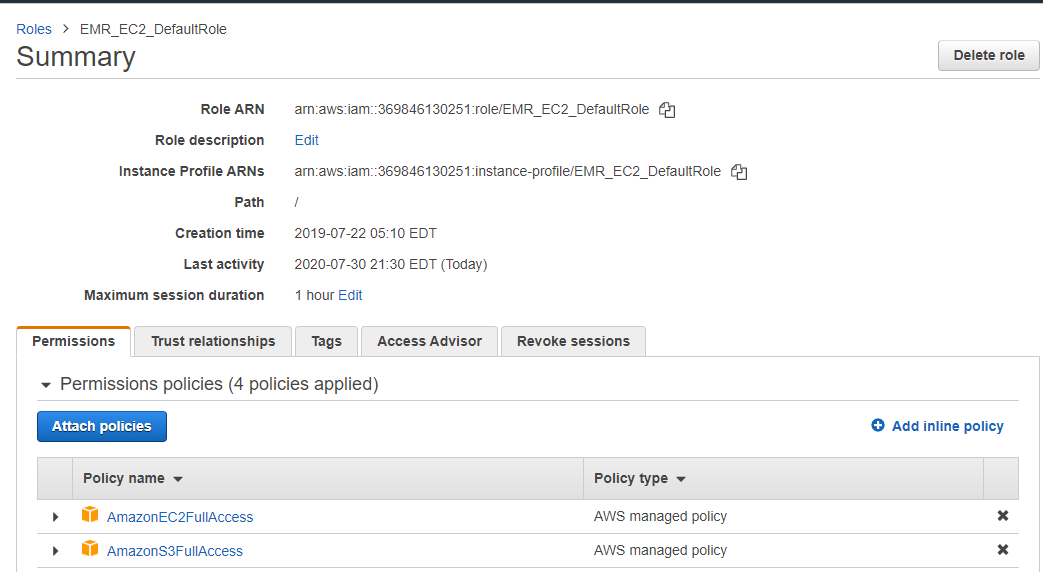
**Spark executors**

<http://ec2-54-82-211-128.compute-1.amazonaws.com:4040/executors/>

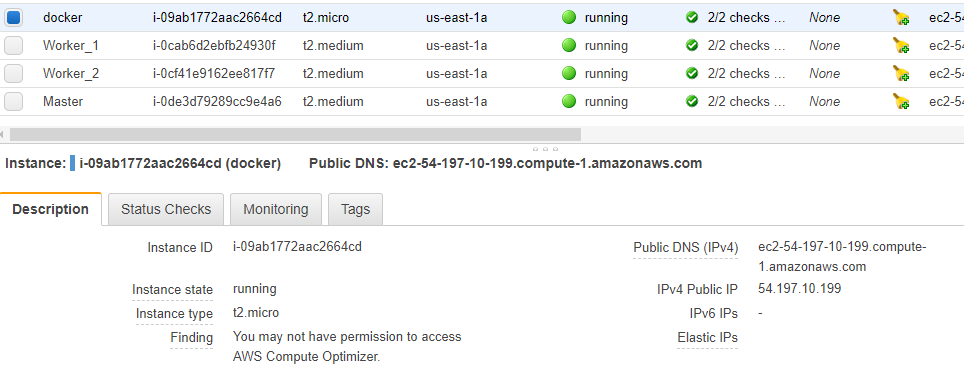


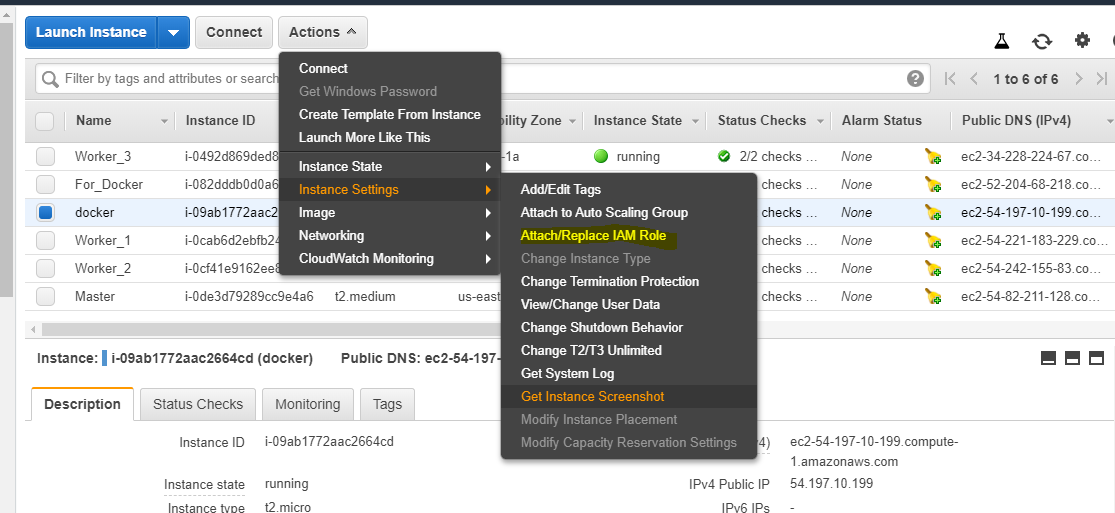
If you are running from EC2 instance, IAM role with S3 access or S3 keys are required to attach to talk to public or private s3 buckets. As a global configuration i used IAM role in my project.

1. IAM role creation with full Ec2/S3 access

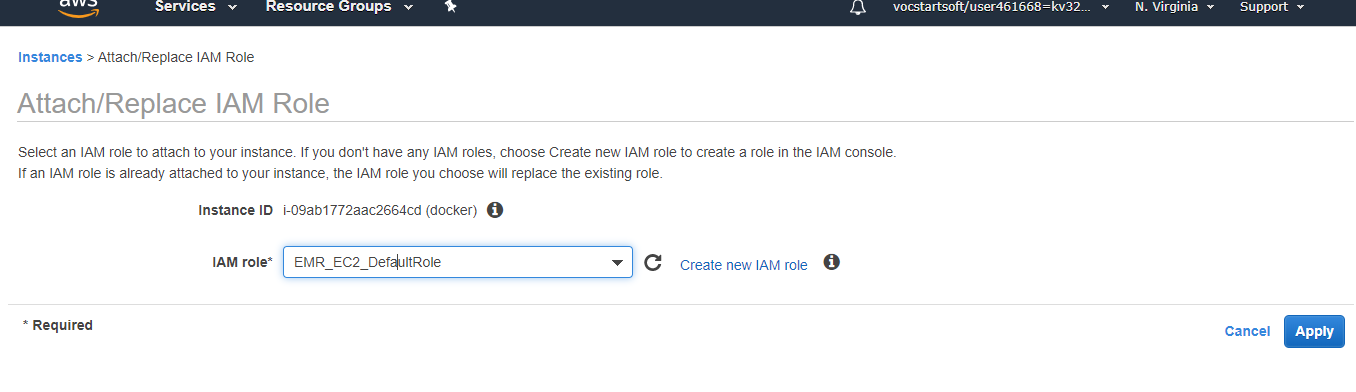


2. Attaching the IAM role to created instance to access the s3 (private or public buckets) smoothly without any credentials





Attach the created IAM role and apply



3. Add your docker run below including build & spark-submit run

wget [https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk-s3/1.11.95/aws-java-sdk-s3-1.11.95.jar](https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk-s3/1.11.95/aws-java-sdk-s3-1.11.95.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk-core/1.11.95/aws-java-sdk-core-1.11.95.jar](https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk-core/1.11.95/aws-java-sdk-core-1.11.95.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/org/apache/spark/spark-hive\_2.11/2.4.0/spark-hive\_2.11-2.4.0.jar](https://repo1.maven.org/maven2/org/apache/spark/spark-hive_2.11/2.4.0/spark-hive_2.11-2.4.0.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/com/databricks/spark-csv\_2.11/1.5.0/spark-csv\_2.11-1.5.0.jar](https://repo1.maven.org/maven2/com/databricks/spark-csv_2.11/1.5.0/spark-csv_2.11-1.5.0.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/org/apache/httpcomponents/httpclient/4.5.2/httpclient-4.5.2.jar](https://repo1.maven.org/maven2/org/apache/httpcomponents/httpclient/4.5.2/httpclient-4.5.2.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-client/2.8.4/hadoop-client-2.8.4.jar](https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-client/2.8.4/hadoop-client-2.8.4.jar" \t "_blank)

 wget [https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-common/2.8.4/hadoop-common-2.8.4.jar](https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-common/2.8.4/hadoop-common-2.8.4.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-aws/2.8.4/hadoop-aws-2.8.4.jar](https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-aws/2.8.4/hadoop-aws-2.8.4.jar" \t "_blank)

wget [https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk/1.11.95/aws-java-sdk-1.11.95.jar](https://repo1.maven.org/maven2/com/amazonaws/aws-java-sdk/1.11.95/aws-java-sdk-1.11.95.jar" \t "_blank)

Above commands should copy the jars under spark2.4.0/jars/ folder. Use the below command for docker prediction

**./spark-submit --class demo.common.WineQualityModelTrain  s3a://cs-643-850-assignment-2/JAR/wine-quality-prediction\_2.11-1.0.jar --master=local[\*] --run-type=prediction**

**--model-store-path=s3a://cs-643-850-assignment-2/output/model --testing-file-path=s3a://cs-643-850-assignment-2/input/validation/ValidationDataset.csv**