CS 643 Programming Assignment-2

Mb945@njit.edu
Mahalakshmi Balasubramanian

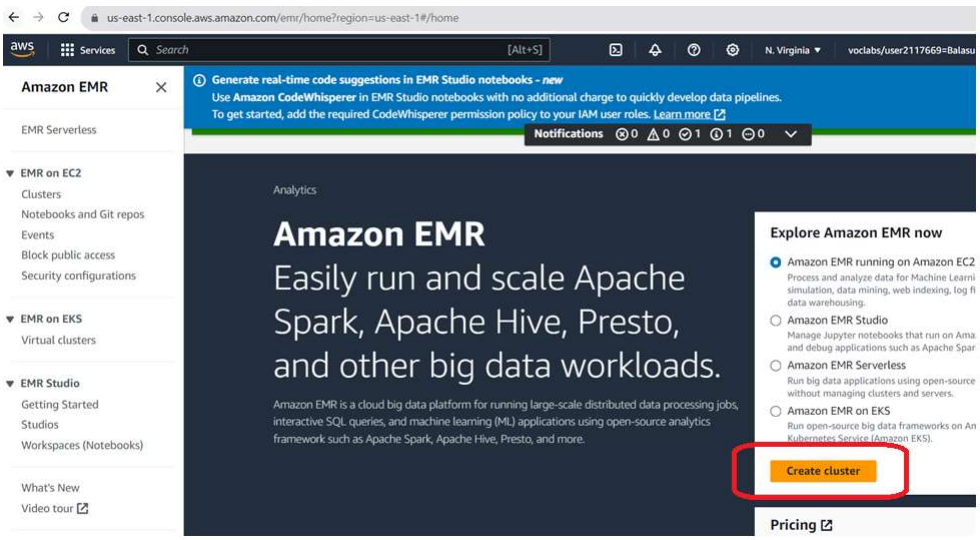
GITHUB LINK
https://github.com/mahibala-njit/CS643-WineQualityPrediction

DOCKERHUB LINK
https://hub.docker.com/layers/mahibala2007/cs643_mlprediction/version2/images/sha256-55fea35de74a75529c42406726ba30c8b2a1e951d2fd08bfdc4b9cd8d985c9e1?context=explore

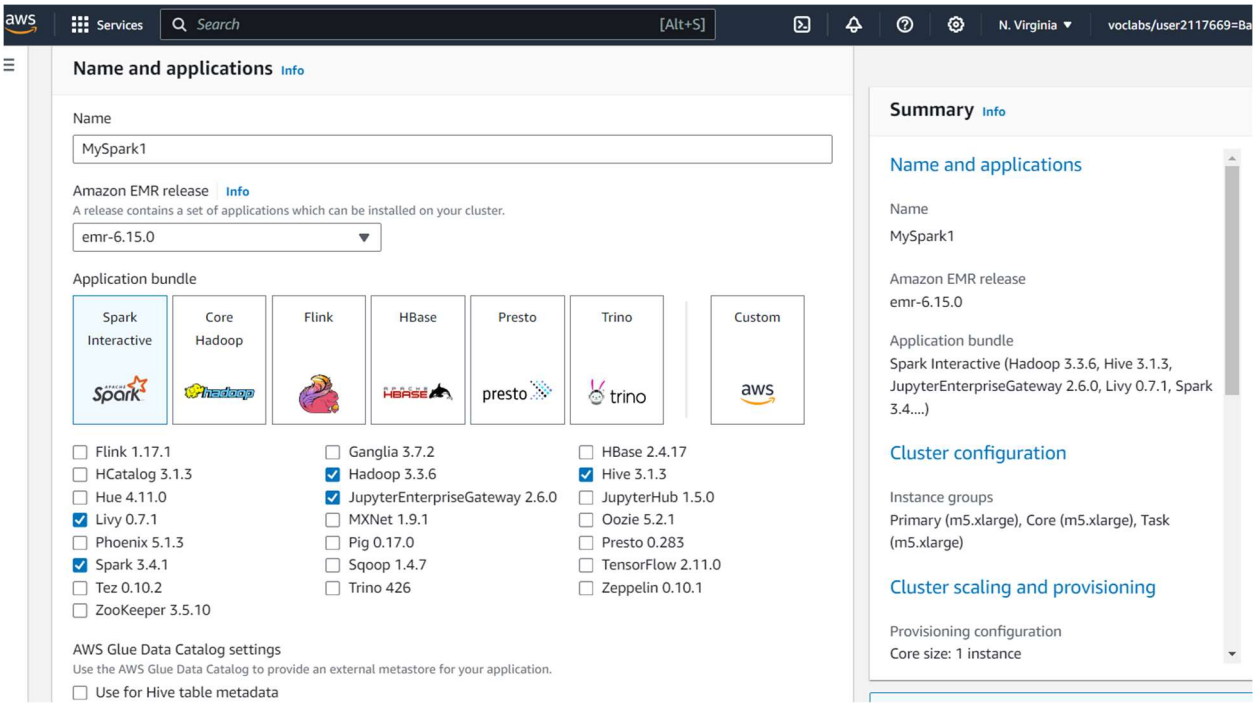
SECTION 1: AWS Cloud setup for running the training ML application - train_model.py

Step 1 : Create an EMR cluster

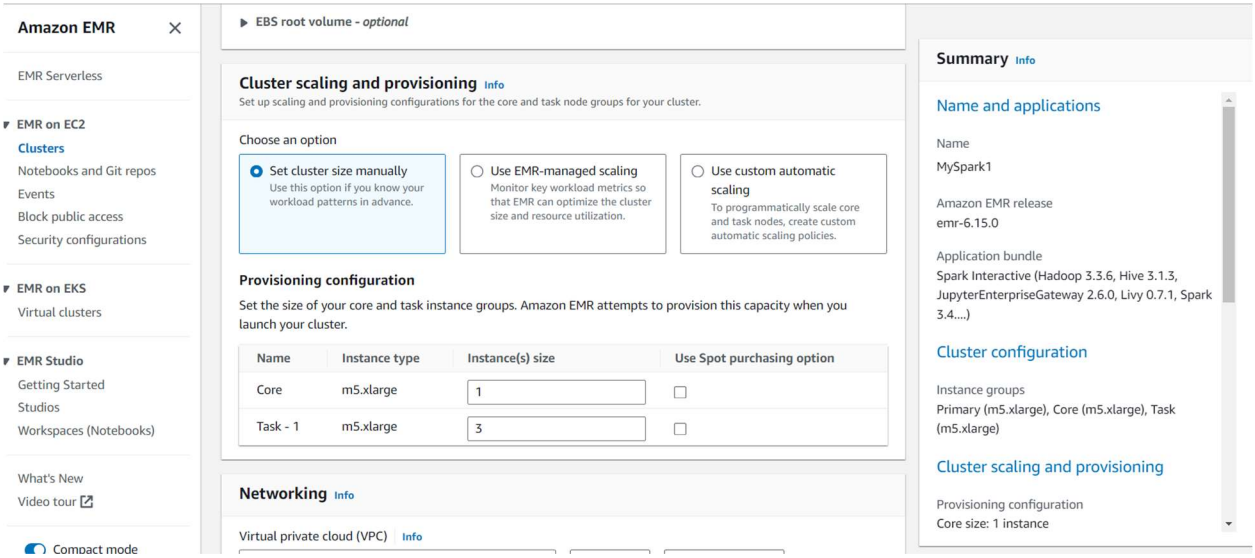
Login to AWS academy student account.
Search for “EMR”.
Click on “Create Cluster” option



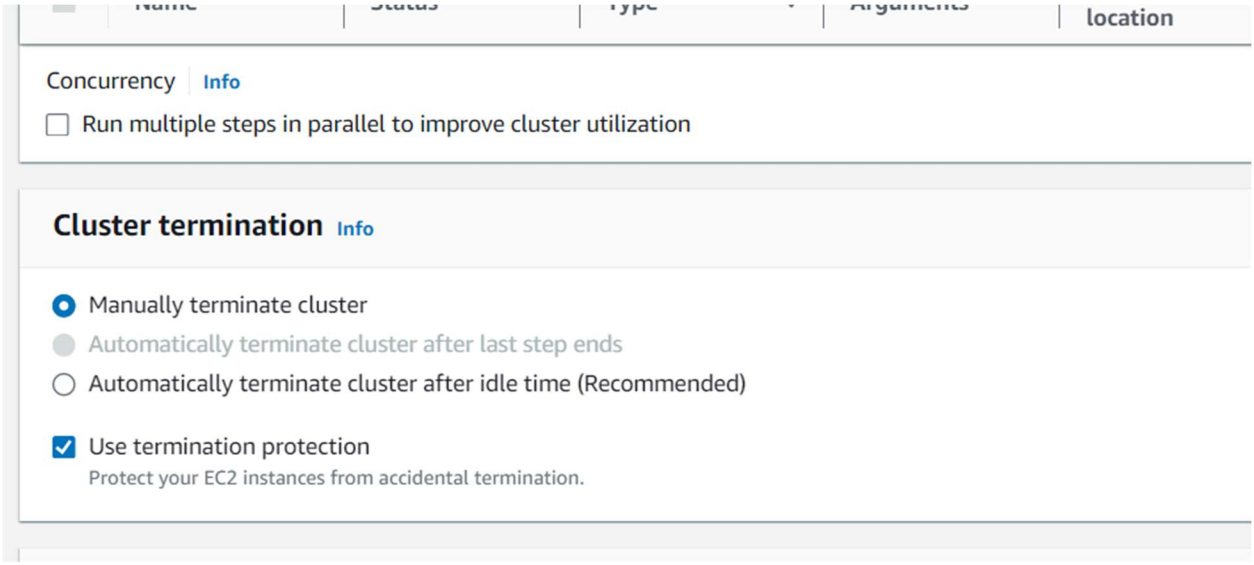
Choose “Spark Interactive” Application bundle and release as shown in the below screenshot



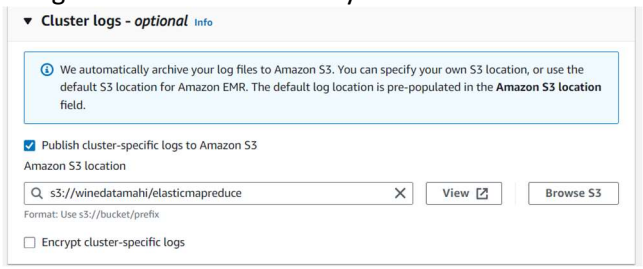
Set the Cluster size manually. For the model to be trained on 4 parallel instances. Choose Instance size as 1 for Core and 3 for “Task-1”. This will create 5 EC2 instances – One for master and 4 for slaves.



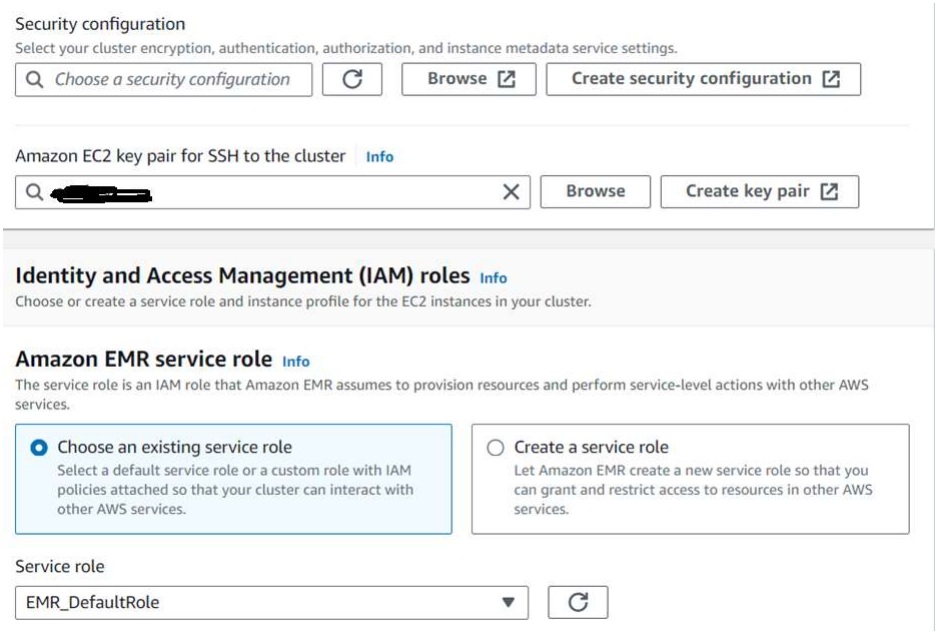
Choose the “Manually terminate cluster” option

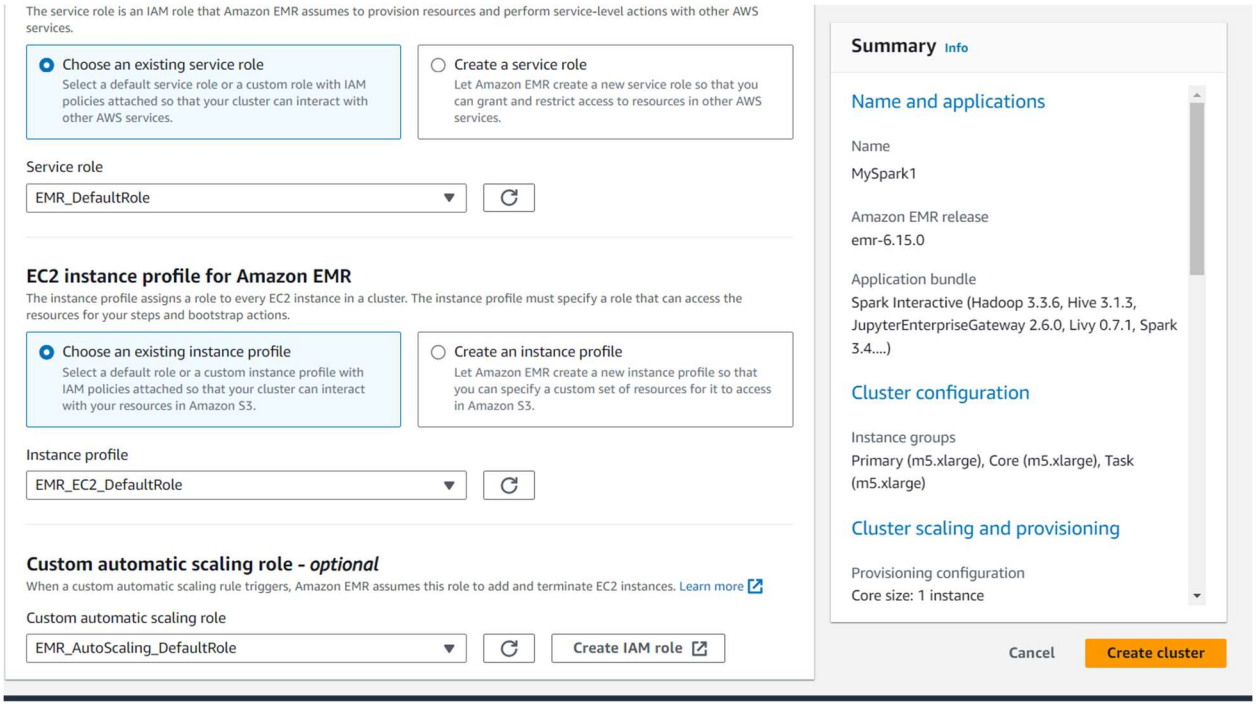


Choose an **S3 bucket for your EMR logs**. Prior to EMR cluster creation, create an S3 bucket if necessary using the same AWS academy account.



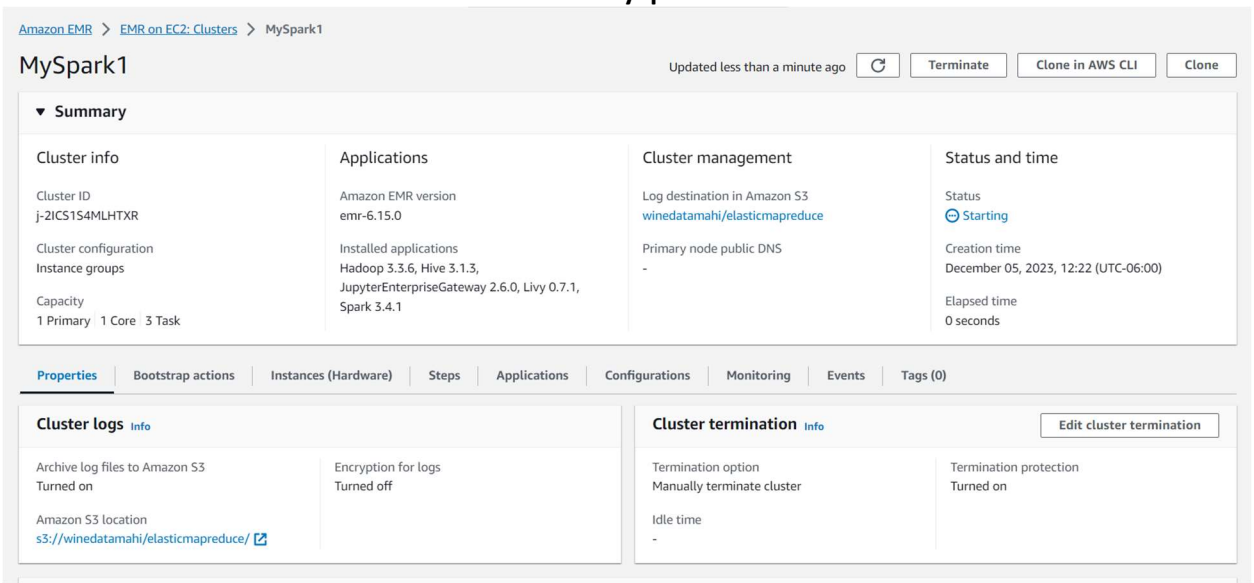
For **Security Configuration**, choose an existing keypair or create a new one. In my case, I chose an existing keypair. Make sure to select “Service role” and Instance Profile as shown below in the screenshot





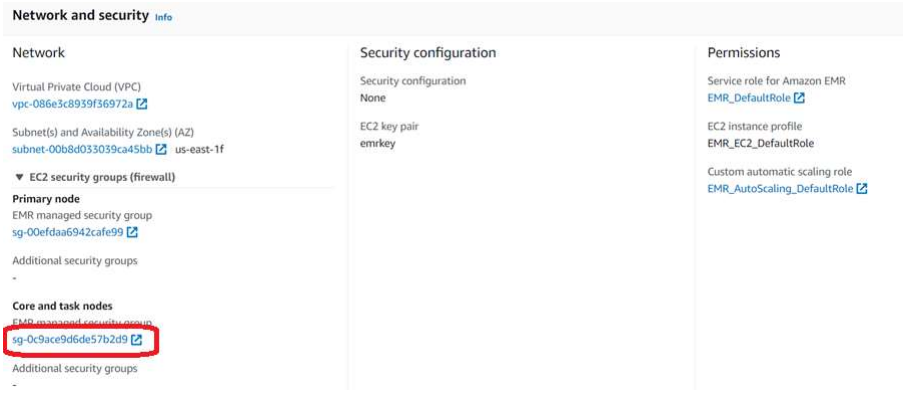
Once all the options are chosen as shown in the screenshot, Click "**Create Cluster**" and wait for few minutes for the cluster to start.

The next screenshot shows that the **EMR cluster "MySpark1" has been created.**

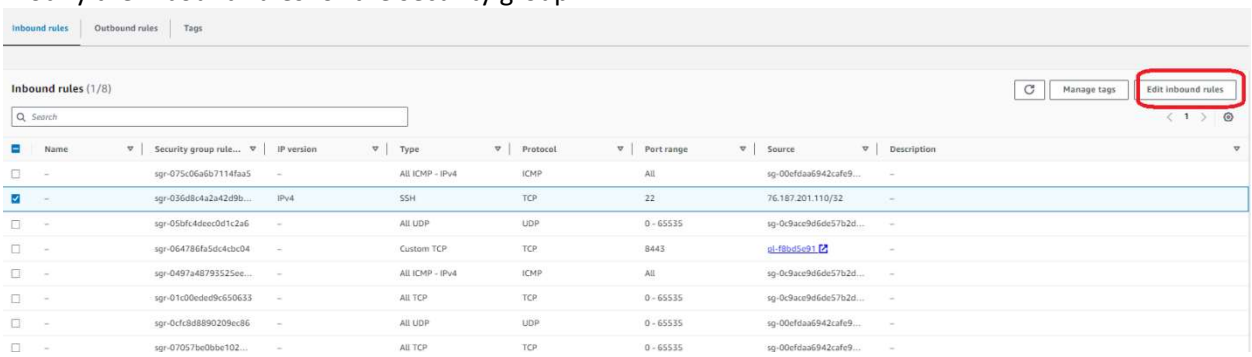


Step 2 : Modify Security group for the cluster to allow MyIP connection

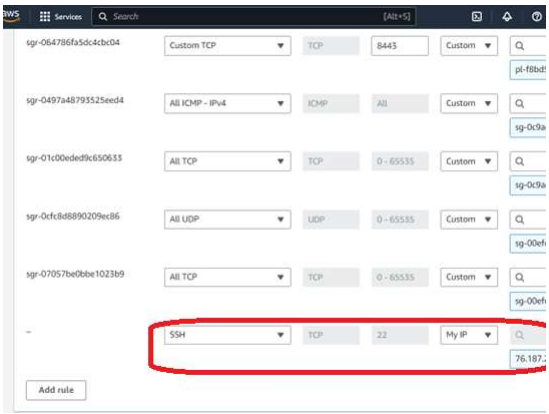
Choose the cluster which was just created with name "MySpark1". Click on the properties.
Scroll to the Nework and Security section. Select the security group that's tied to the cluster.



Modify the inbound rules for the security group



Add a rule to allow SSH from “MyIP”



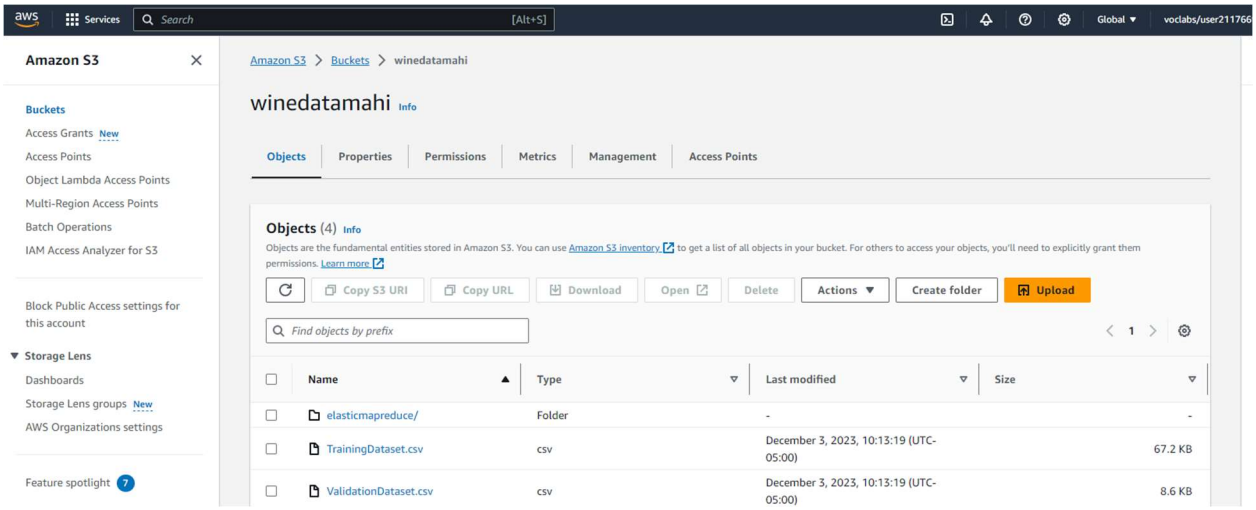
Click on “Add Rule” and then “Save Rules”

Step 3 : Create an S3 bucket and upload the input files

Input Files:

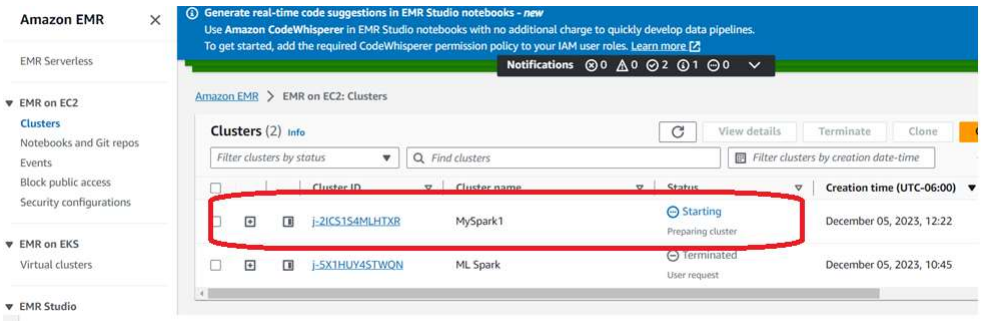
- TrainingDataset.csv
- ValidationDataset.csv

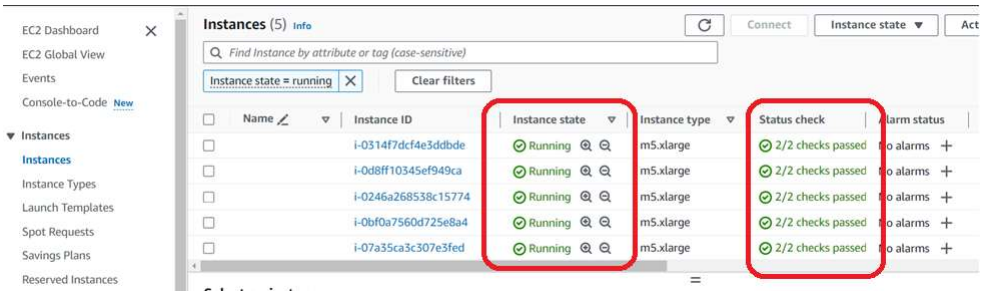
Upload these files to the S3 bucket:



Step 4: Login to Master node, transfer the necessary input files

Once Step 2 is complete, make sure the EMR cluster is up and running and all the EC2 instances are up and running and the status checks are done.





There will be 5 EC2 instances running. 1 for master and 4 for slaves.

Master EC2 instance can be identified using Security group name of the EC2 instances. It would be “ElasticMapReduce-master” for the master instance. From the properties of the EC2 instance, get the public IP or the public IPv4 DNS

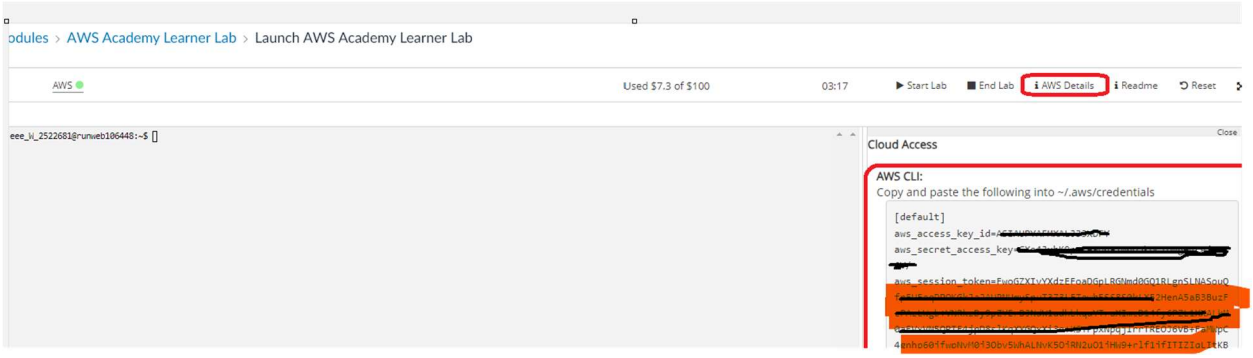
Login to EMR master node using the keypair that was created and attached to the EMR cluster during the cluster creation process.

ssh -i "C:\Users\18484\Desktop\keys\emrkey.pem" ec2-user@ec2-3-230-173-44.compute-1.amazonaws.com

```
C:\Users\18484>ssh -i "C:\Users\18484\Desktop\keys\emrkey.pem" ec2-user@ec2-3-230-173-44.compute-1.amazonaws.com
The authenticity of host 'ec2-3-230-173-44.compute-1.amazonaws.com (3.230.173.44)' can't be established.
ED25519 key fingerprint is SHA256:aLyVgF85At4gOoE5AgIknfHgL1UqtXQbuJbY/Ms1C7c.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? Yes
Warning: Permanently added 'ec2-3-230-173-44.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
       __|  __|_  )
       _|  (     /   Amazon Linux 2
      ___|\___|___|
https://aws.amazon.com/linux/amazon-linux-2
[ec2-user@ip-172-31-69-204 ~]$
```

//Create a directory to place the python files
cd ~
mkdir programming
chmod 774 programming

Get the aws credentials from the vocareum labs page as shown below



Save this information for the below use.

Create the aws credentials file by using the below commands.
cd ~
aws configure
provide the aws credential details when prompted.

vi .aws/credentials

Also try to edit the ~/.aws/credentials file and include all the contents from the AWS CLI Details screenshot shown above.

From another cmd prompt, scp the below files from your local to the master EC2 instance.

1) train_model.py
2) requirements.txt

scp -i "C:\Users\18484\Desktop\keys\emrkey.pem" "D:\Maha\PythonPlayground\WinePrediction\src\train_model.py" ec2-user@ec2-3-230-173-44.compute-1.amazonaws.com:/home/ec2-user/programming

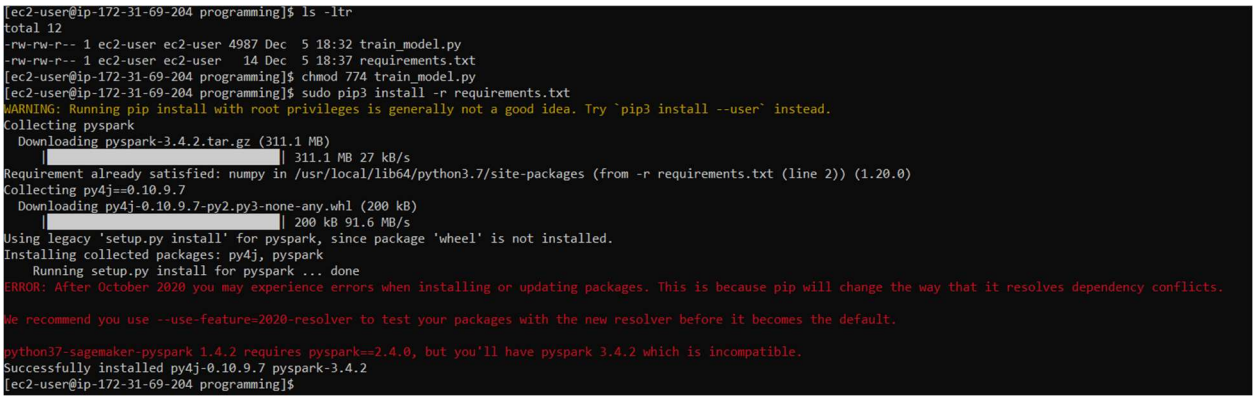
Step 5: Execute the train_model.py ML application

From the previous command prompt, run the below

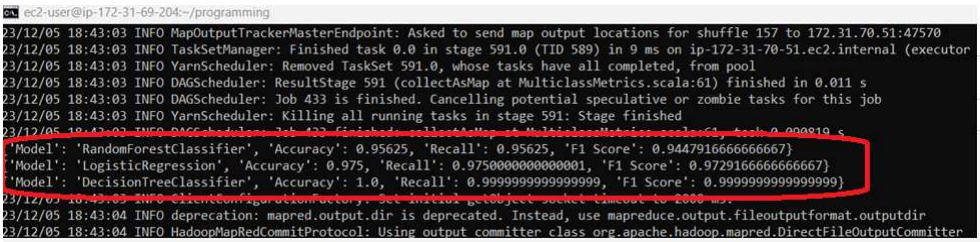
cd programming

chmod 774 train_model.py

sudo pip3 install -r requirements.txt



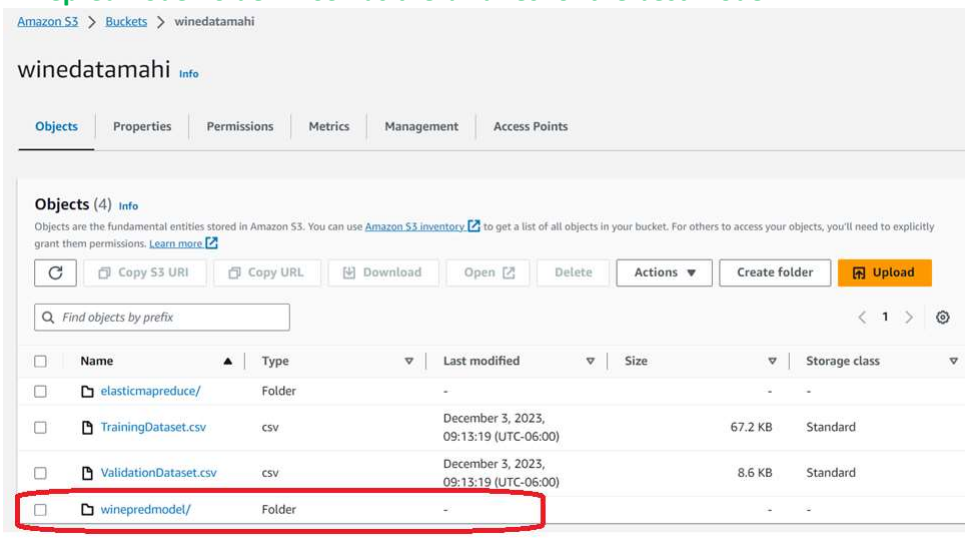
sudo spark-submit train_model.py TrainingDataset.csv ValidationDataset.csv winepredmodel

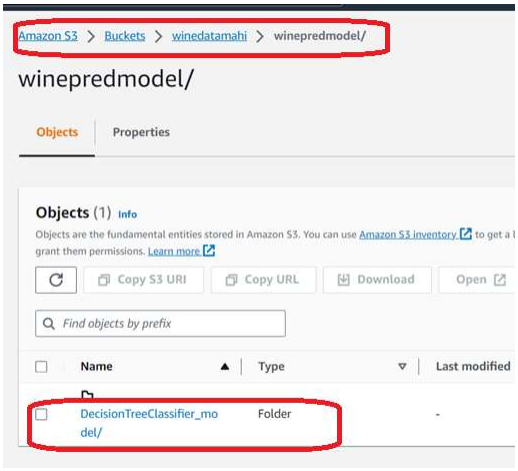


train_model.py will train an ML model using the EMR cluster and will upload the best model to S3 bucket "winedatamahi"

As you can see from the above screenshot, **"DecisionTreeClassifier" model works best for the dataset and after hyperparameter tuning using the Grid search approach , was able to achieve and F1 score of 0.99. This best model was uploaded to S3 bucket.**

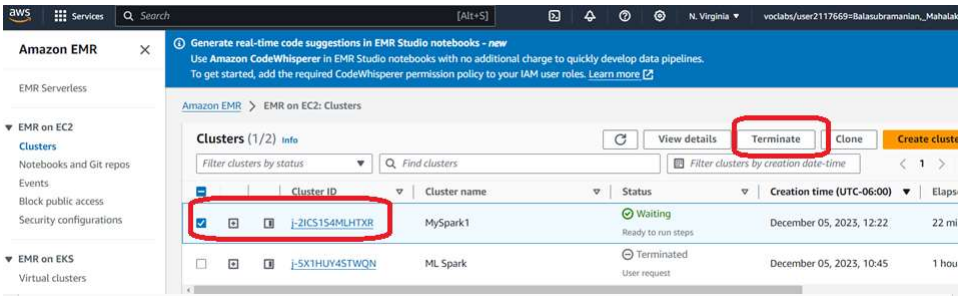
winepredmodel folder in S3 has the binaries for the best model





Step 6: Terminate the EMR cluster

Terminate the EMR Cluster once the work is complete



SECTION 2: AWS Cloud setup for running the prediction ML application - prediction.py
Setting up a standalone EC2 to execute prediction.py without Docker

Step 1: Create an EC2 instance on AWS

Create an EC2 instance with Ubuntu AMI, t2.micro instance type.
Under security, create a security group or associate an existing security group which has the inbound rule to accept ssh connection from “MyIP”.

Associate with a keypair and make sure to download the keypair which will be later used for the login from local to the EC2 instance.

Make sure to select the IAM Role for the instance as “LabInstanceProfile”

Step 2: Login to EC2 and setup the environment

Refer to previous section step 4 for instructions on login to EC2 instance, transferring the input files and setting up the aws credentials.

Login to EC2 instance:
```
ssh -i "C:\Users\18484\Desktop\keys\emrkey.pem" ubuntu@3.84.179.159
```

//Create a directory to place the python files
```
cd ~
mkdir programming
chmod 774 programming

cd ~
aws configure
vi .aws/credentials
```

Also, aws-cli may need to be installed using the below command as this is a standalone EC2 instance and not a part of EMR.

```
sudo apt update
sudo apt install -y awscli
aws configure
```

Transfer commands
Note: In this case prediction.py and requirements.txt file need to be transferred

```
scp -i "C:\Users\18484\Desktop\keys\emrkey.pem"
"D:\Maha\PythonPlayground\WinePrediction\src\prediction.py"
ubuntu@3.84.179.159:/home/ubuntu/programming
```

scp -i "C:\Users\18484\Desktop\keys\emrkey.pem"
"D:\Maha\PythonPlayground\WinePrediction\requirements.txt"
ubuntu@3.84.179.159:/home/ubuntu/programming

Execute the below commands in the given order for setting up the necessary **softwares**

sudo apt-get update && sudo apt-get install -y openjdk-8-jdk python3 python3-pip wget && sudo rm -rf /var/lib/apt/lists/*

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export SPARK_HOME=/opt/spark
export PATH=$SPARK_HOME/bin:$PATH
export PYSPARK_PYTHON=python3

sudo wget https://downloads.apache.org/spark/spark-3.5.0/spark-3.5.0-bin-hadoop3.tgz

sudo tar xvzf spark-3.5.0-bin-hadoop3.tgz -C /opt

sudo mv /opt/spark-3.5.0-bin-hadoop3 $SPARK_HOME
sudo chown -R root:root $SPARK_HOME

cd ~
sudo wget https://repo1.maven.org/maven2/org/apache/hadoop/hadoop-aws/3.3.3/hadoop-aws-3.3.3.jar
sudo mv hadoop-aws-3.3.3.jar /opt/spark/jars/
chmod 774 /opt/spark/jars/ hadoop-aws-3.3.3.jar
cd /home/ubuntu/programming
chmod 774 prediction.py

Execution of prediction.py WITHOUT Docker

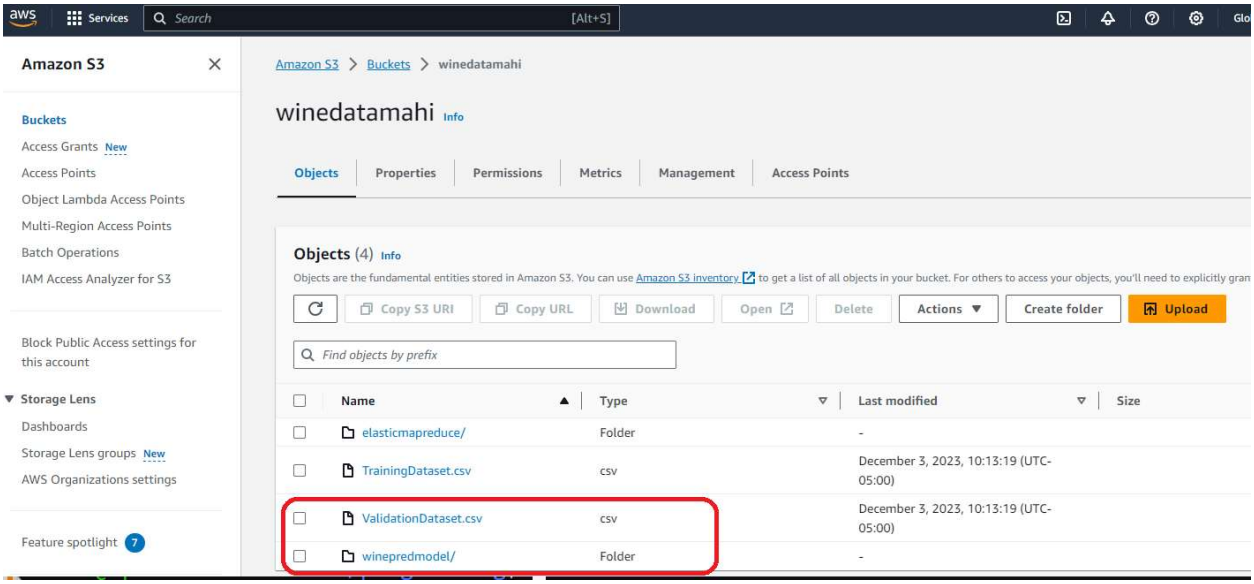
Login to EC2 standalone instance just created using the keypair which is already setup with all the required softwares.

ssh -i "D:/NewDesk/NJIT/CS 643/Programming2/emrkey.pem" ubuntu@50.19.49.5
vi ~/.aws/credentials

Add the latest aws credentials details in this file

```
Last login: Wed Dec  6 05:15:35 2023 from 76.187.201.110
ubuntu@ip-172-31-18-255:~$ vi ~/.aws/credentials
ubuntu@ip-172-31-18-255:~$ ls -ltr
total 391020
-rw-r--r-- 1 root   root   400395283 Sep  9 02:10 spark-3.5.0-bin-hadoop3.tgz
-rw-rw-r-- 1 ubuntu ubuntu         0 Dec  6 00:30 spark-3.2.0-bin-hadoop3.2.tgz
drwxrwxr-- 2 ubuntu ubuntu      4096 Dec  6 00:58 programming
ubuntu@ip-172-31-18-255:~$
```

Make sure the input file – ValidationDataset is available on the s3 bucket “winedatamahi” along with the best model stored as part of the previous training execution.



Make sure prediction.py is available on the programming folder

```
Last login: Wed Dec  6 05:15:35 2023 from 76.187.201.110
ubuntu@ip-172-31-18-255:~$ vi ~/.aws/credentials
ubuntu@ip-172-31-18-255:~$ ls -ltr
total 391020
-rw-r--r-- 1 root   root   400395283 Sep  9 02:10 spark-3.5.0-bin-hadoop3.tgz
-rw-rw-r-- 1 ubuntu ubuntu         0 Dec  6 00:30 spark-3.2.0-bin-hadoop3.2.tgz
drwxrwxr-- 2 ubuntu ubuntu      4096 Dec  6 00:58 programming
ubuntu@ip-172-31-18-255:~$ cd programming
ubuntu@ip-172-31-18-255:~/programming$ ls -ltr
total 16
-rwxrwxr-- 1 ubuntu ubuntu 4956 Dec  6 00:46 train_model.py
-rwxrwxr-- 1 ubuntu ubuntu 2947 Dec  6 00:47 prediction.py
ubuntu@ip-172-31-18-255:~/programming$ sudo spark-submit --packages org.apache.hadoop:hadoop-aws:3.3.3 prediction.py ValidationDataset.csv winepredmodel
```

Execute the prediction.py script using the command below

sudo spark-submit --packages org.apache.hadoop:hadoop-aws:3.3.3 prediction.py ValidationDataset.csv winepredmodel

```
23/12/08 02:57:54 INFO DAGScheduler: Job 11 finished: collectAsMap at MulticlassMetrics.scala:61, took 0.295124 s
Test Accuracy: 1.0
Test F1 Score: 0.9999999999999999
23/12/08 02:57:54 INFO SparkContext: SparkContext is stopping with exitCode 0.
23/12/08 02:57:54 INFO SparkUI: Stopped Spark web UI at http://ip-172-31-18-255.ec2.internal:4040
23/12/08 02:57:54 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
23/12/08 02:57:54 INFO MemoryStore: MemoryStore cleared
23/12/08 02:57:54 INFO BlockManager: BlockManager stopped
23/12/08 02:57:54 INFO BlockManagerMaster: BlockManagerMaster stopped
23/12/08 02:57:54 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
23/12/08 02:57:54 INFO SparkContext: Successfully stopped SparkContext
23/12/08 02:57:55 INFO ShutdownHookManager: Shutdown hook called
23/12/08 02:57:55 INFO ShutdownHookManager: Deleting directory /tmp/spark-254beede-965d-42ab-8ff8-cf487296d1cf
23/12/08 02:57:55 INFO ShutdownHookManager: Deleting directory /tmp/spark-39814d2e-ead7-43cb-92a4-8e3c4e980e28
23/12/08 02:57:55 INFO ShutdownHookManager: Deleting directory /tmp/spark-39814d2e-ead7-43cb-92a4-8e3c4e980e28/pyspark-
d30dae6-2ecc-4902-99d4-e576b737e7c4
23/12/08 02:57:55 INFO MetricsSystemImpl: Stopping s3a-file-system metrics system...
23/12/08 02:57:55 INFO MetricsSystemImpl: s3a-file-system metrics system stopped.
23/12/08 02:57:55 INFO MetricsSystemImpl: s3a-file-system metrics system shutdown complete.
ubuntu@ip-172-31-18-255:~/programming$
```

As shown in the above, **DecisionTree Model from S3 was used to predict the label for the ValidationDataset.csv The accuracy was 1.0 and F1 score was 0.99 for this dataset.**

SECTION 3: AWS Cloud setup for running the prediction ML application - prediction.py
Setting up a standalone EC2 to execute prediction.py WITH Docker

Step 1: Create an EC2 instance on AWS

Create an EC2 instance with Ubuntu AMI, t2.micro instance type.

Under storage, make sure to select >8GB
Under security, create a security group or associate an existing security group which has the inbound rule to accept ssh connection from “MyIP”.

Associate with a keypair and make sure to download the keypair which will be later used for the login from local to the EC2 instance.

Make sure to select the IAM Role for the instance as “LabInstanceProfile”

Step 2: Login to EC2 and setup the environment

Refer to previous section step 4 for instructions on login to EC2 instance, transferring the input files and setting up the aws credentials.

Login to EC2 instance:

```
ssh -i "C:\Users\18484\Desktop\keys\emrkey.pem" ubuntu@3.84.179.159

cd ~
aws configure
vi .aws/credentials
```

Also, aws-cli may need to be installed using the below command as this is a standalone EC2 instance and not a part of EMR.

```
sudo apt update
sudo apt install -y awscli
aws configure
```

Also, make sure docker is installed, else install using the below commands

```
sudo snap install docker
sudo apt update
sudo apt install apt-transport-https ca-certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
sudo apt update
sudo apt install docker-ce
sudo service docker start
sudo systemctl start docker
sudo service docker status
```

sudo usermod -aG docker ubuntu

Execution of prediction.py WITH Docker

Prerequisites: Please refer to the section **"SECTION 4: DOCKER CONTAINERIZATION**" on screenshots for docker image creation and pushing it to dockerhub

Docker image link:
https://hub.docker.com/layers/mahibala2007/cs643_mlprediction/version2/images/sha256-55fea35de74a75529c42406726ba30c8b2a1e951d2fd08bfdc4b9cd8d985c9e1?context=explore

Also, an EC2 instance with docker and AWS credentials setup done should be available.
Input file ValidationDataset.csv should be available on the S3 bucket - winedatamahi

Execution:
Login to EC2 instance using the keypair and use the below commands to pull the image and run the container.

sudo docker login
sudo docker pull mahibala2007/cs643_mlprediction:version2
sudo docker run mahibala2007/cs643_mlprediction:version2 spark-submit --packages org.apache.hadoop:hadoop-aws:3.3.3 prediction.py ValidationDataset.csv winepredmodel

```
ast login: Wed Dec  6 06:50:09 2023 from 76.187.201.110
buntu@ip-172-31-25-16:~$ pwd
/home/ubuntu
buntu@ip-172-31-25-16:~$ vi ~/.aws/credentials
buntu@ip-172-31-25-16:~$ sudo docker login
uthenticating with existing credentials...
ARNING! Your password will be stored unencrypted in /root/.docker/config.json.
onfigure a credential helper to remove this warning. See
ttps://docs.docker.com/engine/reference/commandline/login/#credentials-store

ogin Succeeded
buntu@ip-172-31-25-16:~$ sudo docker pull mahibala2007/cs643_mlprediction:version2
ersion2: Pulling from mahibala2007/cs643_mlprediction
e8117c0bd28: Pull complete
12e9c4237a58: Pull complete
7d8e9ec01762: Pull complete
d67dcf0736c5: Pull complete
0db9823af5f: Pull complete
296db349141: Pull complete
770b920ba9a9: Pull complete
28bfd574b87a: Pull complete
igest: sha256:55fea35de74a75529c42406726ba30c8b2a1e951d2fd08bfdc4b9cd8d985c9e1
tatus: Downloaded newer image for mahibala2007/cs643_mlprediction:version2
ocker.io/mahibala2007/cs643_mlprediction:version2
buntu@ip-172-31-25-16:~$ sudo docker run mahibala2007/cs643_mlprediction:version2 spark-submit --packages org.apache.hadoop:hadoop-aws:3.3.3 prediction.py Vali
taset.csv winepredmodel
:: loading settings :: url = jar:file:/opt/spark/jars/ivy-2.5.1.jar!/org/apache/ivy/core/settings/ivysettings.xml
Ivy Default Cache set to: /root/.ivy2/cache
The jars for the packages stored in: /root/.ivy2/jars
org.apache.hadoop#hadoop-aws added as a dependency
:: resolving dependencies :: org.apache.spark#spark-submit-parent-4ca4384d-d4fe-445e-85c4-724a15007e1f;1.0
        confs: [default]
```

```
23/12/08 03:33:54 INFO DAGScheduler: ResultStage 14 (collectAsMap at MulticlassMetrics.scala:61) finished in 0.082 s
23/12/08 03:33:54 INFO DAGScheduler: Job 11 is finished. Cancelling potential speculative or zombie tasks for this job
23/12/08 03:33:54 INFO TaskSchedulerImpl: Killing all running tasks in stage 14: Stage finished
23/12/08 03:33:54 INFO DAGScheduler: Job 11 finished: collectAsMap at MulticlassMetrics.scala:61, took 0.824460 s
Test Accuracy: 1.0
Test F1 Score: 0.9999999999999999
23/12/08 03:33:54 INFO SparkUI: Stopped Spark web UI at http://eaa6ff6146dc:4040
23/12/08 03:33:54 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
23/12/08 03:33:54 INFO MemoryStore: MemoryStore cleared
23/12/08 03:33:54 INFO BlockManager: BlockManager stopped
23/12/08 03:33:54 INFO BlockManagerMaster: BlockManagerMaster stopped
23/12/08 03:33:54 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
23/12/08 03:33:54 INFO SparkContext: Successfully stopped SparkContext
23/12/08 03:33:55 INFO ShutdownHookManager: Shutdown hook called
23/12/08 03:33:55 INFO ShutdownHookManager: Deleting directory /tmp/spark-c3ca139f-4c85-456d-8852-0460b030830c
23/12/08 03:33:55 INFO ShutdownHookManager: Deleting directory /tmp/spark-7db2251d-7b34-499e-bcb2-49bcc4de2dfa
23/12/08 03:33:55 INFO ShutdownHookManager: Deleting directory /tmp/spark-c3ca139f-4c85-456d-8852-0460b030830c/pyspark-d8d7
23/12/08 03:33:55 INFO MetricsSystemImpl: Stopping s3a-file-system metrics system...
23/12/08 03:33:55 INFO MetricsSystemImpl: s3a-file-system metrics system stopped.
23/12/08 03:33:55 INFO MetricsSystemImpl: s3a-file-system metrics system shutdown complete.
ubuntu@ip-172-31-25-16:~$
```

As shown in the above, **DecisionTree Model from S3 was used to predict the label for the ValidationDataset.csv The accuracy was 1.0 and F1 score was 0.99 for this dataset USING DOCKER.**

APPENDIX-SECTION 4: DOCKER CONTAINERIZATION:

Make sure docker is installed from the instance you are trying to create the docker image.
Also the Dockerfile need to be available as a prerequisite
Follow the commands in the below screenshot to create the image and push the image to the dockerhub

```
ubuntu@ip-172-31-25-16:~/programming$ ls -ltr
total 20
-rwxrwxr-- 1 ubuntu ubuntu 4956 Dec  6 06:53 train_model.py
-rwxrwxr-- 1 ubuntu ubuntu   22 Dec  6 06:53 requirements.txt
-rwxrwxr-- 1 ubuntu ubuntu 1195 Dec  6 06:53 Dockerfile
-rwxrwxr-- 1 ubuntu ubuntu 3141 Dec  6 07:11 prediction.py
ubuntu@ip-172-31-25-16:~/programming$ sudo docker images
REPOSITORY                     TAG        IMAGE ID       CREATED              SIZE
mahibala2007/cs643_mlprediction   version1   92306eda25b2   About an hour ago   2.41GB
ubuntu@ip-172-31-25-16:~/programming$ sudo docker build -t mahibala2007/cs643_mlprediction:version2 .
[+] Building 225.2s (14/14) FINISHED                                                                  docker:default
 => [internal] load build definition from Dockerfile                                                              0.1s
 => => transferring dockerfile: 1.23kB                                                                            0.0s
 => [internal] load .dockerignore                                                                                 0.0s
 => => transferring context: 2B                                                                                   0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest                                                  0.6s
 => [auth] library/ubuntu:pull token for registry-1.docker.io                                                     0.0s
 => [1/8] FROM docker.io/library/ubuntu:latest@sha256:8eab65df33a6de2844c9aefd19efe8ddb87b7df5e9185a4ab73af936225685bb   0.1s
 => => resolve docker.io/library/ubuntu:latest@sha256:8eab65df33a6de2844c9aefd19efe8ddb87b7df5e9185a4ab73af936225685bb   0.0s
 => => sha256:b6548eacb0639263e9d8abfee48f8ac8b327102a05335b67572f715c580a968e 2.30kB / 2.30kB                    0.0s
 => => sha256:8eab65df33a6de2844c9aefd19efe8ddb87b7df5e9185a4ab73af936225685bb 1.13kB / 1.13kB                    0.0s
 => => sha256:149d67e29f765f4db62aa52161009e99e389544e25a8f43c8c89d4a445a7ca37 424B / 424B                        0.0s
 => [internal] load build context                                                                                 0.0s
 => => transferring context: 3.25kB                                                                               0.0s
 => [2/8] RUN apt-get update && apt-get install -y     openjdk-8-jdk     python3     python3-pip     wget     && rm -rf /var/lib/apt/lists/*   70.7s
 => [3/8] RUN wget -qO- https://downloads.apache.org/spark/spark-3.3.3/spark-3.3.3-bin-hadoop3.tgz | tar xvz -C /opt     && mv /opt/spark-3.3.3-bin-hadoop3 /opt/spark     && chown -R root:root /opt/spark  21.5s
 => [4/8] WORKDIR /app                                                                                            0.1s
 => [5/8] COPY prediction.py /app/                                                                                0.1s
 => [6/8] COPY requirements.txt /app/                                                                             0.0s
 => [7/8] RUN pip install --upgrade pip && pip install -r requirements.txt                                        81.0s
 => [8/8] RUN chmod 774 *                                                                                         1.6s
 => exporting to image                                                                                            48.9s
 => => exporting layers                                                                                           48.9s
 => => writing image sha256:a466e3ce10adaf4239aa7dd0e4a7e1679fcca97d74f4012d7800ce8f406dec9e                      0.0s
 => => naming to docker.io/mahibala2007/cs643_mlprediction:version2                                               0.0s
ubuntu@ip-172-31-25-16:~/programming$
```

```
ubuntu@ip-172-31-25-16:~/programming$ sudo docker tag mahibala2007/cs643_mlprediction:version2 mahibala2007/cs643_mlprediction:version2
ubuntu@ip-172-31-25-16:~/programming$ sudo docker push mahibala2007/cs643_mlprediction:version2
The push refers to repository [docker.io/mahibala2007/cs643_mlprediction]
5fa97fa6e2e6: Pushed
4beef8c83362: Pushed
8a8e2bfd1f1f: Pushed
1ff527a87f10: Pushed
7dec2e25ecce: Pushed
d3c315077c07: Pushed
1b394a682f0f: Pushed
8ceb9643fb36: Layer already exists
version2: digest: sha256:55fea35de74a75529c42406726ba30c8b2a1e951d2fd08bfdc4b9cd8d985c9e1 size: 1998
ubuntu@ip-172-31-25-16:~/programming$
```