

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

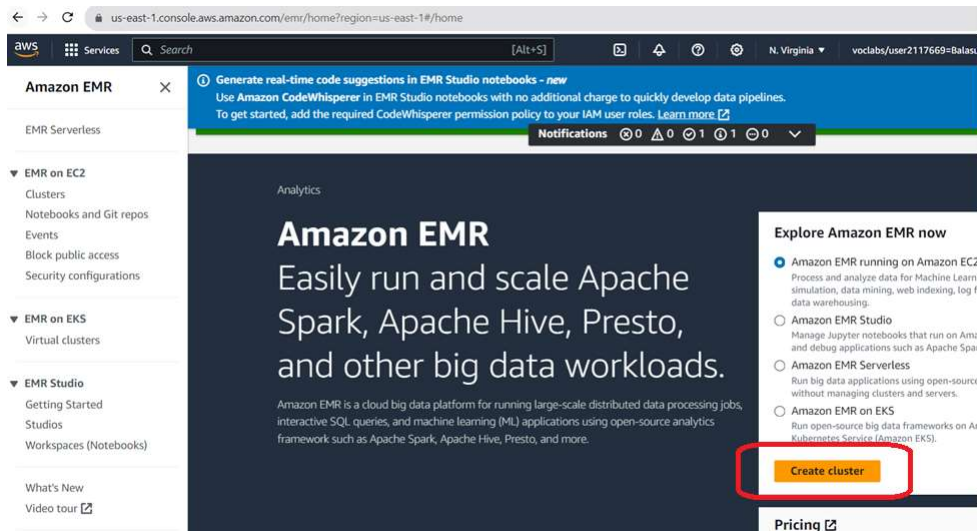
SECTION 1: AWS Cloud setup for running the training ML application - train_model.py

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

Name and applications [Info](#)

Name
MySpark1

Amazon EMR release [Info](#)
A release contains a set of applications which can be installed on your cluster.
emr-6.15.0

Application bundle

Spark Interactive

Core Hadoop

Flink

HBase

Presto

Trino

Custom

☐ Flink 1.17.1
☐ HCatalog 3.1.3
☐ Hue 4.11.0
☒ Livy 0.7.1
☐ Phoenix 5.1.3
☒ Spark 3.4.1
☐ Tez 0.10.2
☐ ZooKeeper 3.5.10

☐ Ganglia 3.7.2
☒ Hadoop 3.3.6
☒ JupyterEnterpriseGateway 2.6.0
☐ MXNet 1.9.1
☐ Pig 0.17.0
☐ Sqoop 1.4.7
☐ Trino 426

☐ HBase 2.4.17
☒ Hive 3.1.3
☐ JupyterHub 1.5.0
☐ Oozie 5.2.1
☐ Presto 0.283
☐ TensorFlow 2.11.0
☐ Zeppelin 0.10.1

AWS Glue Data Catalog settings
Use the AWS Glue Data Catalog to provide an external metastore for your application.
☐ Use for Hive table metadata

Summary [Info](#)

Name and applications

Name
MySpark1

Amazon EMR release
emr-6.15.0

Application bundle
Spark Interactive (Hadoop 3.3.6, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.7.1, Spark 3.4....)

Cluster configuration

Instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)

Cluster scaling and provisioning

Provisioning configuration
Core size: 1 instance

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.

Amazon EMR [×](#)

EMR Serverless

EMR on EC2
[Clusters](#)
Notebooks and Git repos
Events
Block public access
Security configurations

EMR on EKS
Virtual clusters

EMR Studio
Getting Started
Studios
Workspaces (Notebooks)

What's New
Video tour [📺](#)

[Compact mode](#)

Cluster scaling and provisioning [Info](#)
Set up scaling and provisioning configurations for the core and task node groups for your cluster.

Choose an option

☒ **Set cluster size manually**
Use this option if you know your workload patterns in advance.

☐ **Use EMR-managed scaling**
Monitor key workload metrics so that EMR can optimize the cluster size and resource utilization.

☐ **Use custom automatic scaling**
To programmatically scale core and task nodes, create custom automatic scaling policies.

Provisioning configuration
Set the size of your core and task instance groups. Amazon EMR attempts to provision this capacity when you launch your cluster.

Name	Instance type	Instance(s) size	Use Spot purchasing option
Core	m5.xlarge	1	<input type="checkbox"/>
Task - 1	m5.xlarge	3	<input type="checkbox"/>

Networking [Info](#)
Virtual private cloud (VPC) [Info](#)

Summary [Info](#)

Name and applications

Name
MySpark1

Amazon EMR release
emr-6.15.0

Application bundle
Spark Interactive (Hadoop 3.3.6, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.7.1, Spark 3.4....)

Cluster configuration

Instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)

Cluster scaling and provisioning

Provisioning configuration
Core size: 1 instance

Choose the "Manually terminate cluster" option

Name	Status	Type	Arguments	location
Concurrency Info <input type="checkbox"/> Run multiple steps in parallel to improve cluster utilization				
Cluster termination Info <input checked="" type="radio"/> Manually terminate cluster <input type="radio"/> Automatically terminate cluster after last step ends <input type="radio"/> Automatically terminate cluster after idle time (Recommended) <input checked="" type="checkbox"/> Use termination protection Protect your EC2 instances from accidental termination.				

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.

▼ Cluster logs - optional [Info](#)

ⓘ We automatically archive your log files to Amazon S3. You can specify your own S3 location, or use the default S3 location for Amazon EMR. The default log location is pre-populated in the **Amazon S3 location** field.

☒ Publish cluster-specific logs to Amazon S3

Amazon S3 location

Format: Use s3://bucket/prefix

☐ Encrypt cluster-specific logs

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

Security configuration
 Select your cluster encryption, authentication, authorization, and instance metadata service settings.

Amazon EC2 key pair for SSH to the cluster [Info](#)

Identity and Access Management (IAM) roles [Info](#)
 Choose or create a service role and instance profile for the EC2 instances in your cluster.

Amazon EMR service role [Info](#)
 The service role is an IAM role that Amazon EMR assumes to provision resources and perform service-level actions with other AWS services.

☒ **Choose an existing service role**
 Select a default service role or a custom role with IAM policies attached so that your cluster can interact with other AWS services.

☐ **Create a service role**
 Let Amazon EMR create a new service role so that you can grant and restrict access to resources in other AWS services.

Service role

The service role is an IAM role that Amazon EMR assumes to provision resources and perform service-level actions with other AWS services.

☒ **Choose an existing service role**
 Select a default service role or a custom role with IAM policies attached so that your cluster can interact with other AWS services.

☐ **Create a service role**
 Let Amazon EMR create a new service role so that you can grant and restrict access to resources in other AWS services.

Service role

EMR_DefaultRole

EC2 instance profile for Amazon EMR

The instance profile assigns a role to every EC2 instance in a cluster. The instance profile must specify a role that can access the resources for your steps and bootstrap actions.

☒ **Choose an existing instance profile**
 Select a default role or a custom instance profile with IAM policies attached so that your cluster can interact with your resources in Amazon S3.

☐ **Create an instance profile**
 Let Amazon EMR create a new instance profile so that you can specify a custom set of resources for it to access in Amazon S3.

Instance profile

EMR_EC2_DefaultRole

Custom automatic scaling role - optional

When a custom automatic scaling rule triggers, Amazon EMR assumes this role to add and terminate EC2 instances. [Learn more](#)

Custom automatic scaling role

EMR_AutoScaling_DefaultRole

Create IAM role

Summary Info

Name and applications

Name
MySpark1

Amazon EMR release
emr-6.15.0

Application bundle
Spark Interactive (Hadoop 3.3.6, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.7.1, Spark 3.4....)

Cluster configuration

Instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)

Cluster scaling and provisioning

Provisioning configuration
Core size: 1 instance

Cancel **Create cluster**

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.

Amazon EMR > EMR on EC2: Clusters > MySpark1

MySpark1 Updated less than a minute ago Terminate Clone in AWS CLI Clone

▼ Summary

Cluster info Cluster ID j-2ICS154MLHTXR Cluster configuration Instance groups Capacity 1 Primary 1 Core 3 Task	Applications Amazon EMR version emr-6.15.0 Installed applications Hadoop 3.3.6, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.7.1, Spark 3.4.1	Cluster management Log destination in Amazon S3 winedatamahi/elasticmapreduce Primary node public DNS -	Status and time Status Starting Creation time December 05, 2023, 12:22 (UTC-06:00) Elapsed time 0 seconds
---	--	--	---

Properties Bootstrap actions Instances (Hardware) Steps Applications Configurations Monitoring Events Tags (0)

Cluster logs Info

Archive log files to Amazon S3
Turned on
Amazon S3 location
[s3://winedatamahi/elasticmapreduce/](#)

Encryption for logs
Turned off

Cluster termination Info Edit cluster termination

Termination option
Manually terminate cluster
Idle time
-

Termination protection
Turned on

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 Network and Security section. Select the security group that’s tied to the cluster.

Network and security Info

Network

Virtual Private Cloud (VPC)
vpc-086e3c8939f36972a

Subnet(s) and Availability Zone(s) (AZ)
subnet-00b8d033039ca45bb us-east-1f

▼ EC2 security groups (firewall)

Primary node
EMR managed security group
sg-00efdaa6942cafe99

Additional security groups

Core and task nodes
EMR managed security group
sg-0c9ace9d6de57b2d9

Additional security groups

Security configuration

Security configuration
None

EC2 key pair
emrkey

Permissions

Service role for Amazon EMR
EMR_DefaultRole

EC2 instance profile
EMR_EC2_DefaultRole

Custom automatic scaling role
EMR_AutoScaling_DefaultRole

Modify the inbound rules for the security group

Inbound rules | Outbound rules | Tags

Inbound rules (1/8)

Search

Manage tags Edit inbound rules

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sg-075c06a6b7114fa5	-	All ICMP - IPv4	ICMP	All	sg-00efdaa6942cafe9...	-
✓	sg-036d8c4a2a42d9b...	IPv4	SSH	TCP	22	76.187.201.110/32	-
-	sg-053fc4deec0d1c2a6	-	All UDP	UDP	0 - 65535	sg-0c9ace9d6de57b2d...	-
-	sg-064786fa5d4c4bc04	-	Custom TCP	TCP	8443	sg-0c9ace9d6de57b2d...	-
-	sg-0497a48793525ee...	-	All ICMP - IPv4	ICMP	All	sg-0c9ace9d6de57b2d...	-
-	sg-01c00dede9c650633	-	All TCP	TCP	0 - 65535	sg-0c9ace9d6de57b2d...	-
-	sg-0cfc8d8890209ec86	-	All UDP	UDP	0 - 65535	sg-00efdaa6942cafe9...	-
-	sg-07057be0bbe102369	-	All TCP	TCP	0 - 65535	sg-00efdaa6942cafe9...	-

Add a rule to allow SSH from “MyIP”

Services Search [Alt+S]

sg-064786fa5d4c4bc04 Custom TCP TCP 8443 Custom

sg-0497a48793525ee04 All ICMP - IPv4 ICMP All Custom

sg-01c00dede9c650633 All TCP TCP 0 - 65535 Custom

sg-0cfc8d8890209ec86 All UDP UDP 0 - 65535 Custom

sg-07057be0bbe102369 All TCP TCP 0 - 65535 Custom

SSH TCP 22 My IP

Add rule

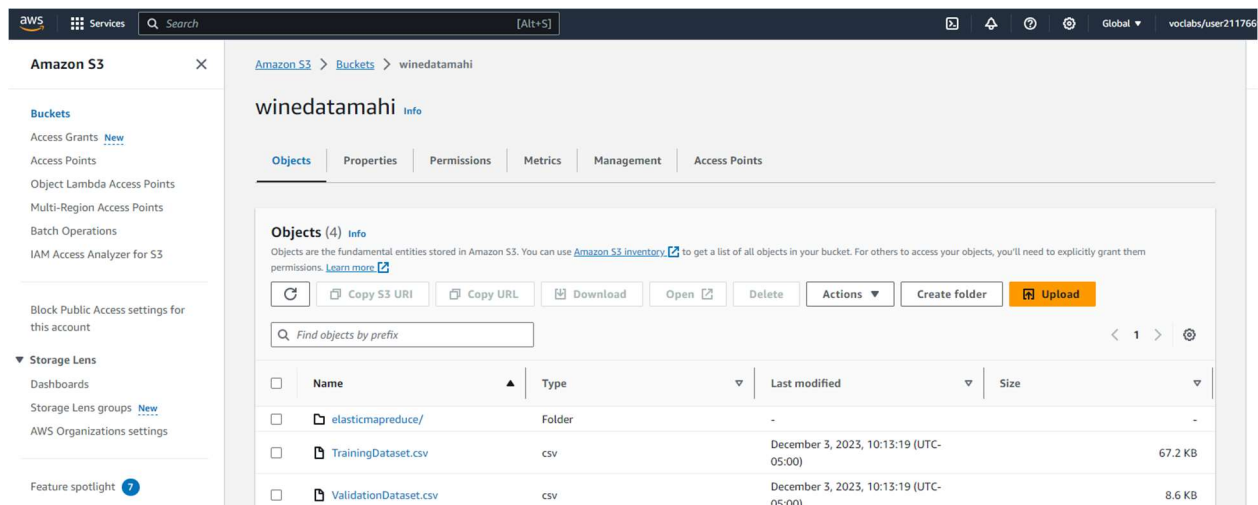
Click on “Add Rule” and then “Save Rules”

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

Input Files:

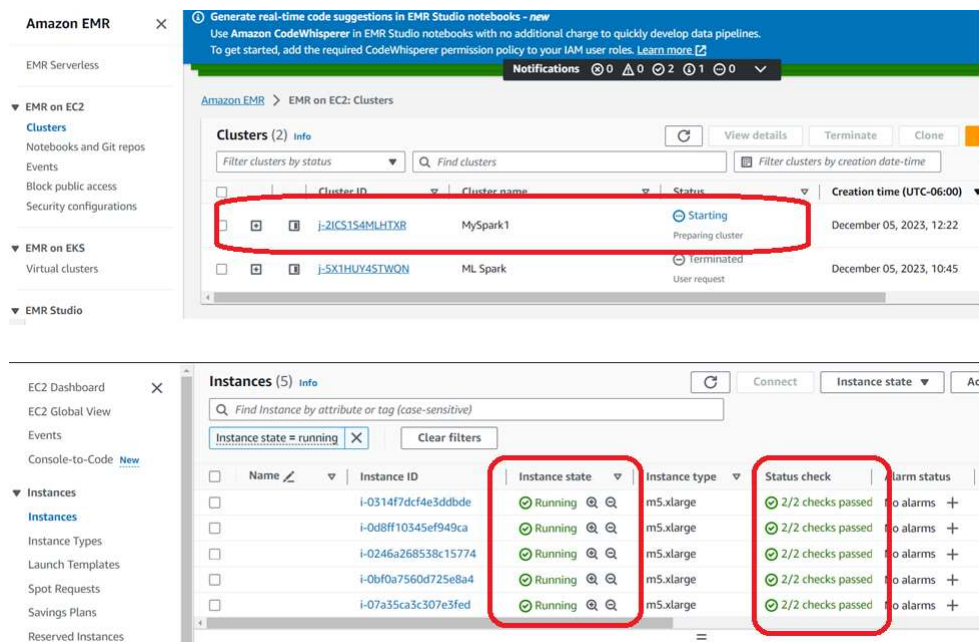
- TrainingDataset.csv
- ValidationDataset.csv

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.


```
C:\Users\18484ssh -i "C:\Users\18484\Desktop\keys\ec2-user.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:aLYVgF85At4g0oE5Ag1knfhgLUqtQbuJbY/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
~\#####
~\###| AL2 End of Life is 2025-06-30.
~\#|
~\V~^~^~^ A newer version of Amazon Linux is available!
~\~\~\~\
~\~\~\~\ Amazon Linux 2023, GA and supported until 2028-03-15.
~\m/'~\~\~\~\ https://aws.amazon.com/linux/amazon-linux-2023/

26 package(s) needed for security, out of 35 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMM MMMMMMM RRRRRRRRRRRRRRR
E:::E:::E:::E:::E::: M:::M::: M:::M::: R:::R:::R:::R:::
EE:::E:::E:::E:::E::: M:::M::: M:::M::: R:::RRRRRR:::R
E:::E::: EEEEE M:::M::: M:::M::: RR:::R R:::R
E:::E::: M:::M::: M:::M::: M:::M::: R:::R R:::R
E:::E:::EEEEEEEE M:::M M:::M M:::M M:::M R:::RRRRRR:::R
E:::E:::E:::E::: M:::M M:::M M:::M M:::M R:::RRRR:::R
E:::E:::EEEEEEEE M:::M M:::M M:::M R:::RRRRRR:::R
E:::E::: M:::M M:::M M:::M R:::R R:::R
[ec2-user@ip-172-31-69-204 ~]$
EE:::EEEEEEEE::: E:::M::: M:::M::: R:::R R:::R
M:::M::: RR:::R R:::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM MMMMMMM RRRRRRR RRRRRR

[ec2-user@ip-172-31-69-204 ~]$
```

chmod 774 programming

Modules > AWS Academy Learner Lab > Launch AWS Academy Learner Lab

AWS ● Used \$7.3 of \$100 03:17 ▶ Start Lab ■ End Lab **AWS Details** ⓘ Readme ↺ Reset

eee_h1_2522681g@unweb106448:~\$

Cloud Access

AWS CLI:
Copy and paste the following into ~/.aws/credentials

```
[default]
aws_access_key_id=
aws_secret_access_key=
aws_session_token=
```

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
```

```
C:\Users\18484>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
train_model.py 100% 4987 77.9KB/s 00:00
C:\Users\18484>
C:\Users\18484>scp -i "C:\Users\18484\Desktop\keys\emrkey.pem" "D:\Maha\PythonPlayground\WinePrediction\requirements.txt" ec2-user@ec2-3-230-173-44.compute-1.amazonaws.com:/home/ec2-user/programming
requirements.txt 100% 14 0.2KB/s 00:00
```

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
```

```
[ec2-user@ip-172-31-69-204 programming]$ ls -ltr
total 12
-rw-rw-r-- 1 ec2-user ec2-user 4987 Dec  5 18:32 train_model.py
-rw-rw-r-- 1 ec2-user ec2-user  14 Dec  5 18:37 requirements.txt
[ec2-user@ip-172-31-69-204 programming]$ chmod 774 train_model.py
[ec2-user@ip-172-31-69-204 programming]$ sudo pip3 install -r requirements.txt
WARNING: Running pip install with root privileges is generally not a good idea. Try `pip3 install --user` instead.
Collecting pyspark
  Downloading pyspark-3.4.2.tar.gz (311.1 MB)
    | 311.1 MB 27 kB/s
Requirement already satisfied: numpy in /usr/local/lib64/python3.7/site-packages (from -r requirements.txt (line 2)) (1.20.0)
Collecting py4j==0.10.9.7
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
    | 200 kB 91.6 MB/s
Using legacy 'setup.py install' for pyspark, since package 'wheel' is not installed.
Installing collected packages: py4j, pyspark
  Running setup.py install for pyspark ... done
ERROR: After October 2020 you may experience errors when installing or updating packages. This is because pip will change the way that it resolves dependency conflicts.
We recommend you use --use-feature=2020-resolver to test your packages with the new resolver before it becomes the default.

python37-sagemaker-pyspark 1.4.2 requires pyspark==2.4.0, but you'll have pyspark 3.4.2 which is incompatible.
Successfully installed py4j-0.10.9.7 pyspark-3.4.2
[ec2-user@ip-172-31-69-204 programming]$
```

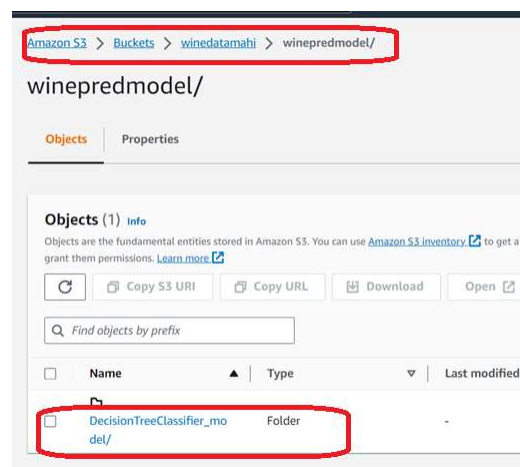
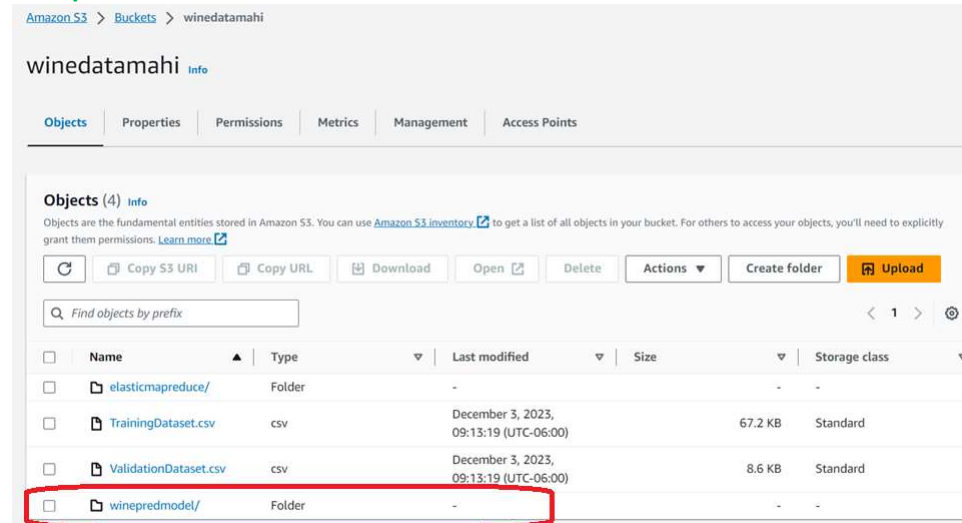
sudo spark-submit train_model.py TrainingDataset.csv ValidationDataset.csv winepredmodel

```
ec2-user@ip-172-31-69-204:~/programming
23/12/05 18:43:03 INFO MapOutputTrackerMasterEndpoint: Asked to send map output locations for shuffle 157 to 172.31.70.51:47570
23/12/05 18:43:03 INFO TaskSetManager: Finished task 0.0 in stage 591.0 (TID 589) in 9 ms on ip-172-31-70-51.ec2.internal (executor
23/12/05 18:43:03 INFO YarnScheduler: Removed TaskSet 591.0, whose tasks have all completed, from pool
23/12/05 18:43:03 INFO DAGScheduler: ResultStage 591 (collectAsMap at MulticlassMetrics.scala:61) finished in 0.011 s
23/12/05 18:43:03 INFO DAGScheduler: Job 433 is finished. Cancelling potential speculative or zombie tasks for this job
23/12/05 18:43:03 INFO YarnScheduler: Killing all running tasks in stage 591: Stage finished
23/12/05 18:43:03 INFO DAGScheduler: Job 433 finished: collectAsMap at MulticlassMetrics.scala:61, Task 0.000819 s
{"Model": "RandomForestClassifier", "Accuracy": 0.95625, "Recall": 0.95625, "F1 Score": 0.9447916666666667}
{"Model": "LogisticRegression", "Accuracy": 0.975, "Recall": 0.9750000000000001, "F1 Score": 0.9729166666666667}
{"Model": "DecisionTreeClassifier", "Accuracy": 1.0, "Recall": 0.9999999999999999, "F1 Score": 0.9999999999999999}
23/12/05 18:43:03 INFO ClientRMRegistrationFactory: Set initial getobject socket timeout to 2000 ms.
23/12/05 18:43:04 INFO deprecation: mapred.output.dir is deprecated. Instead, use mapreduce.output.fileoutputformat.outputdir
23/12/05 18:43:04 INFO HadoopMapRedCommitProtocol: Using output committer class org.apache.hadoop.mapred.DirectFileOutputCommitter
```

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

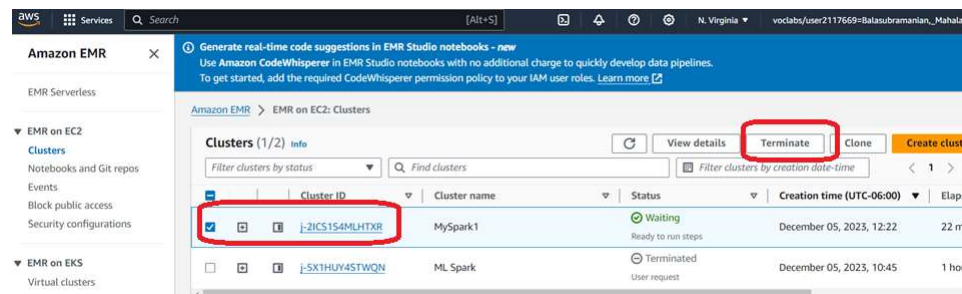
As you can see from the above screenshot, “DecisionTreeClassifier” model works best for the dataset and after hyperparameter tuning using the Grid search approach , was able to achieve and F1 score of 0.99. This best model was uploaded to S3 bucket.

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 PYSARK_PYTHON=python3
```

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

```
sudo tar xvf 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
```

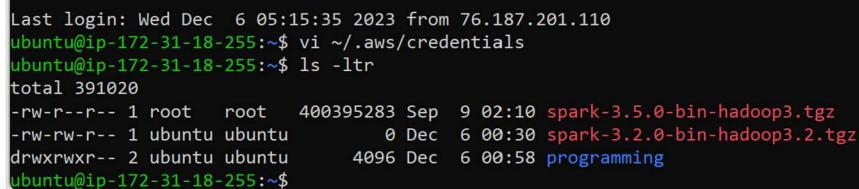
```
sudo mv hadoop-aws-3.3.3.jar /opt/spark/jars/  
chmod 774 /opt/spark/jars/ hadoop-aws-3.3.3.jar  
cd /home/ubuntu/programming  
chmod 774 prediction.py
```

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.

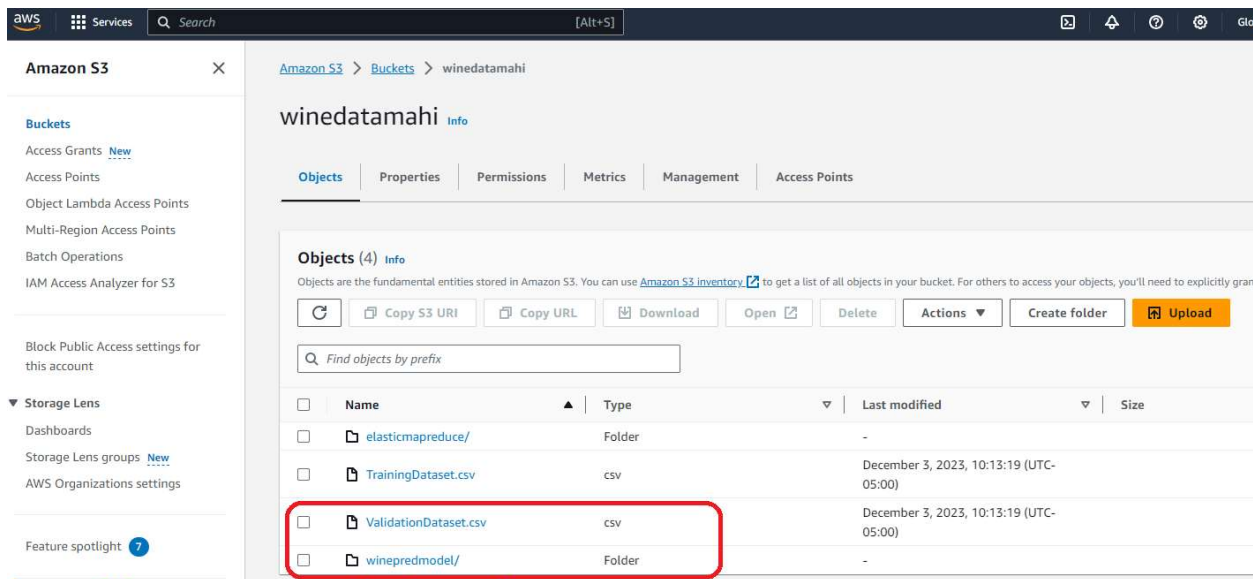
```
ssh -i "D:/NewDesk/NJIT/CS 643/Programming2/emrkey.pem" ubuntu@50.19.49.5  
vi ~/.aws/credentials
```

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 TaskSchedulerImpl: Killing all running tasks in stage 14: Stage finished
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

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

Login Succeeded
ubuntu@ip-172-31-25-16:~$ sudo docker pull mahibala2007/cs643_mlprediction:version2
version2: Pulling from mahibala2007/cs643_mlprediction
5e8117c0bd28: Pull complete
12e9c4237a58: Pull complete
7d8e9ec01762: Pull complete
467dcf0736c5: Pull complete
b0db9823af5f: Pull complete
c296db349141: Pull complete
770b920ba9a9: Pull complete
28bfd574b87a: Pull complete
Digest: sha256:55fea35de74a75529c42406726ba30c8b2a1e951d2fd08bfdc4b9cd8d985c9e1
Status: Downloaded newer image for mahibala2007/cs643_mlprediction:version2
docker.io/mahibala2007/cs643_mlprediction:version2
ubuntu@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 Val
tasent.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          92386ed25b2        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
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 1.23kB
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [auth] library/ubuntu:pull token for registry-1.docker.io
=> [1/8] FROM docker.io/library/ubuntu:latest@sha256:8eab65df33a0de204dc9ae6d19efebdd87b7df5e9185a4ab73af936225685bb
=> => resolve docker.io/library/ubuntu:latest@sha256:8eab65df33a0de204dc9ae6d19efebdd87b7df5e9185a4ab73af936225685bb
=> => sha256:8eab65df33a0de204dc9ae6d19efebdd87b7df5e9185a4ab73af936225685bb 2.38kB / 2.38kB
=> => sha256:8eab65df33a0de204dc9ae6d19efebdd87b7df5e9185a4ab73af936225685bb 1.13kB / 1.13kB
=> => sha256:149db7e29f763f4db62aa52101809e99e389544e75a8f438c89da4a5a7ca37 424B / 424B
=> [internal] load build context
=> => transferring context: 3.25kB
=> [2/8] RUN apt-get update && apt-get install -y openjdk-8-jdk python3 python3-pip wget && rm -rf /var/lib/apt/lists/*
=> [3/8] RUN wget -O- 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
=> [4/8] WORKDIR /app
=> [5/8] COPY prediction.py /app/
=> [6/8] COPY requirements.txt /app/
=> [7/8] RUN pip install --upgrade pip && pip install -r requirements.txt
=> [8/8] RUN choco 774 *
=> exporting to image
=> writing image sha256:a66e1ce10ada1423baa7d8b04a7e1679fcca97d74f4012d7800ce0ff400edcbe
=> naming to docker.io/mahibala2007/cs643_mlprediction:version2
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
8a8e2bdf1f1f: Pushed
1ff527a87f10: Pushed
7dec2e25ecce: Pushed
d3c315077c07: Pushed
1b394a682f0f: Pushed
8ceb9643fb36: Layer already exists
version2: digest: sha256:55fea35de74a75529c42406726ba30c8b2a1e951d2fd08bfdc4b9cd8985c9e1 size: 1998
ubuntu@ip-172-31-25-16:~/programming$

```