AWS Spark Wine Quality Prediction Application

pySpark AWS Wine Prediction App Is an application that was developed with Python's pySpark interface and installed on the AWR EMR cluster.

The purpose of this project is to leverage publicly available data to train a machine learning model in parallel on EC2 instances, with the goal of predicting wine quality. In order to make deployments easier, the project also uses Docker to produce a container image for a trained machine learning model.

This project contains 2 main python source files:

wine_prediction.py reads training dataset from S3 and trains model in parallel on EMR spark cluster. Once model is trained, it can be run on provided test data provided via S3. This program stores trained model in S3 bucket (Public URL for bucket - S3://pa2winequ)

wine_prediction.py program loads trained model and executes that model on given test data file. This will then print F1 score as metrics for accuracy of the trained model.

Docker file: Docker file to create docker image and run container for simplified deployment.

Link to GitHub code

https://github.com/vijayasindhudandeboina/AWS-Spark-wine-Quality

Link to docker container image

https://hub.docker.com/repository/docker/vijaya430/wine-prediction-app/general

Source files:

`wine_prediction.py` reads Training dataset from S3 and trains model in parallel on EMR spark cluster. Once model is trained, it can be run on provided test data provided via S3. This program stores trained model in S3 bucket (Public URL for bucket - S3://wine-data-12)

`wine_test_data_prediction.py` program loads trained model and executes that model on given test data file. This will then print F1 score as metrics for accuracy of the trained model.

`Docker file` creates docker image and run container for simplified deployment.

Instruction to use:

Phase 1: Pre-requisites

Prepare the PEM file, credentials, and AWS account (same as last assignment).

Phase 2: Copy data to S3.

We will copy all the datasets given and the python scripts to s3 in our bucket.

Phase 3: Establish an EMR Cluster

1. Create Spark cluster in AWS.

Using the EMR console that AWS provides, users can establish spark clusters.

Kindly follow the instructions to build one using four EC2 instances (you can use more, depending on your load).

1. Use the "EC2-> Network & Security -> Key-pairs" navigation to create a key pair for the EMR cluster.

Utilize the format pem. The {name of key pair}>.pem file will be downloaded as a result. To perform an SSH to an EC2 instance, you will need to keep it secure.

2. Go to the console for Amazon EMR. Next, select Clusters -> Create Cluster from the menu.

3. Now complete the corresponding sections:

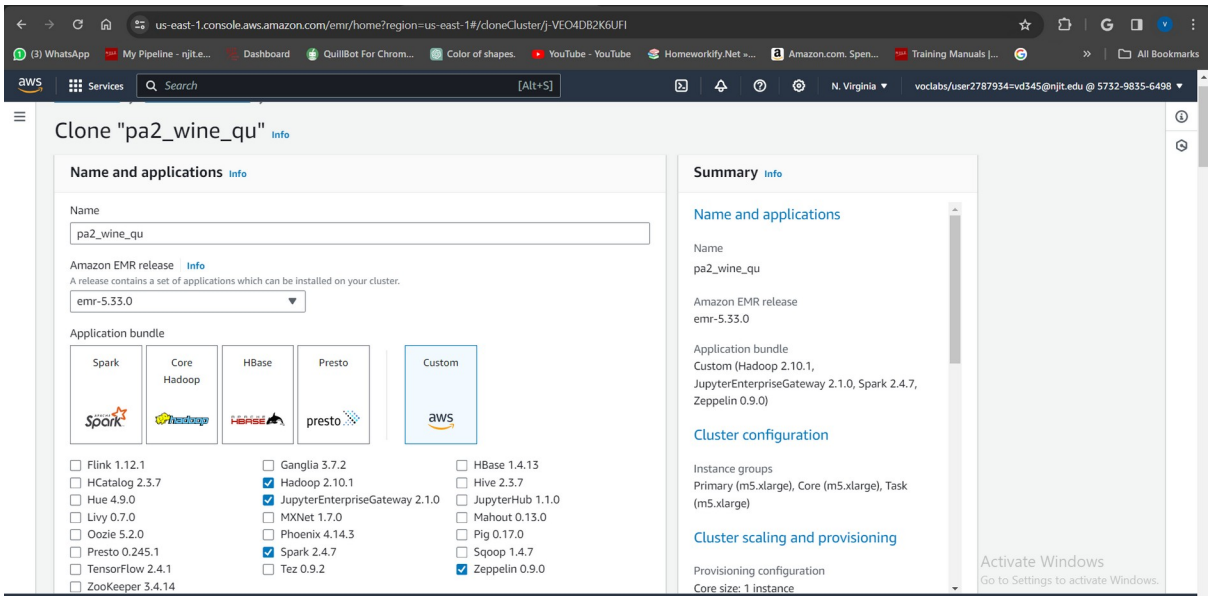
Proceed to the General Configuration -> Cluster Name -> Software Configuration -> EMR 5.33 menu and choose 'Spark: Spark 2.4.7 on Hadoop 2.10.1 YARN and Zeppelin 0.9.0'.

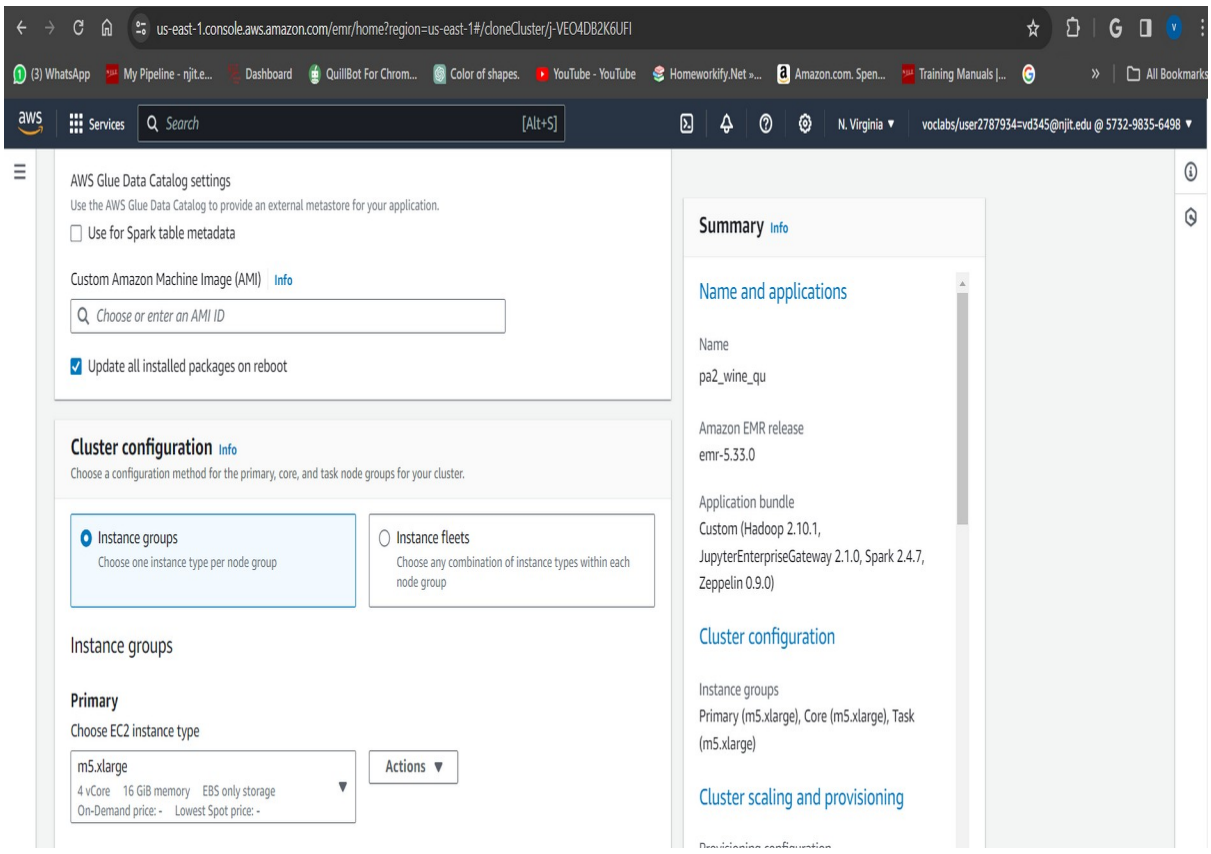
Hardware Configuration -> Set the number of instances to 4.

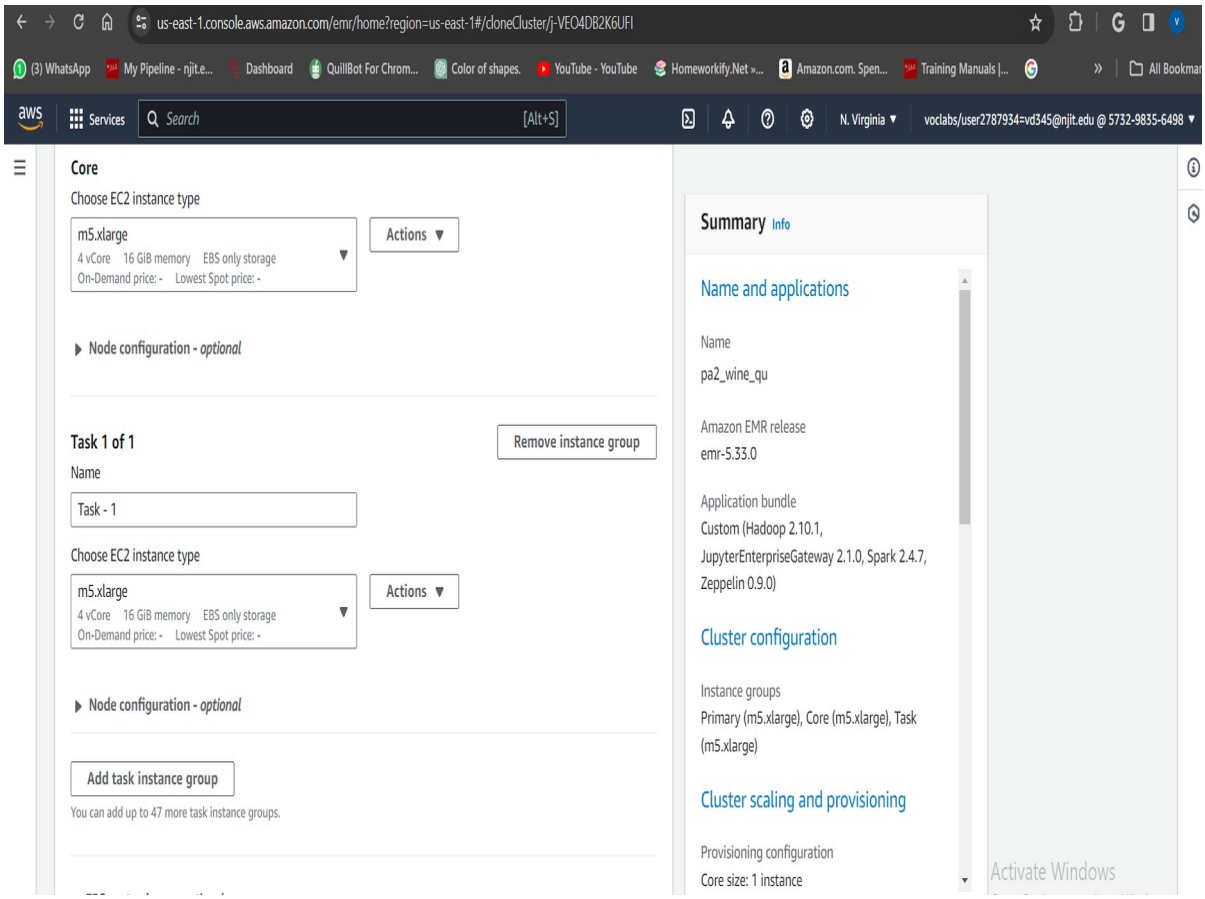
Pem key generated in the previous step under Security Access -> Provide.

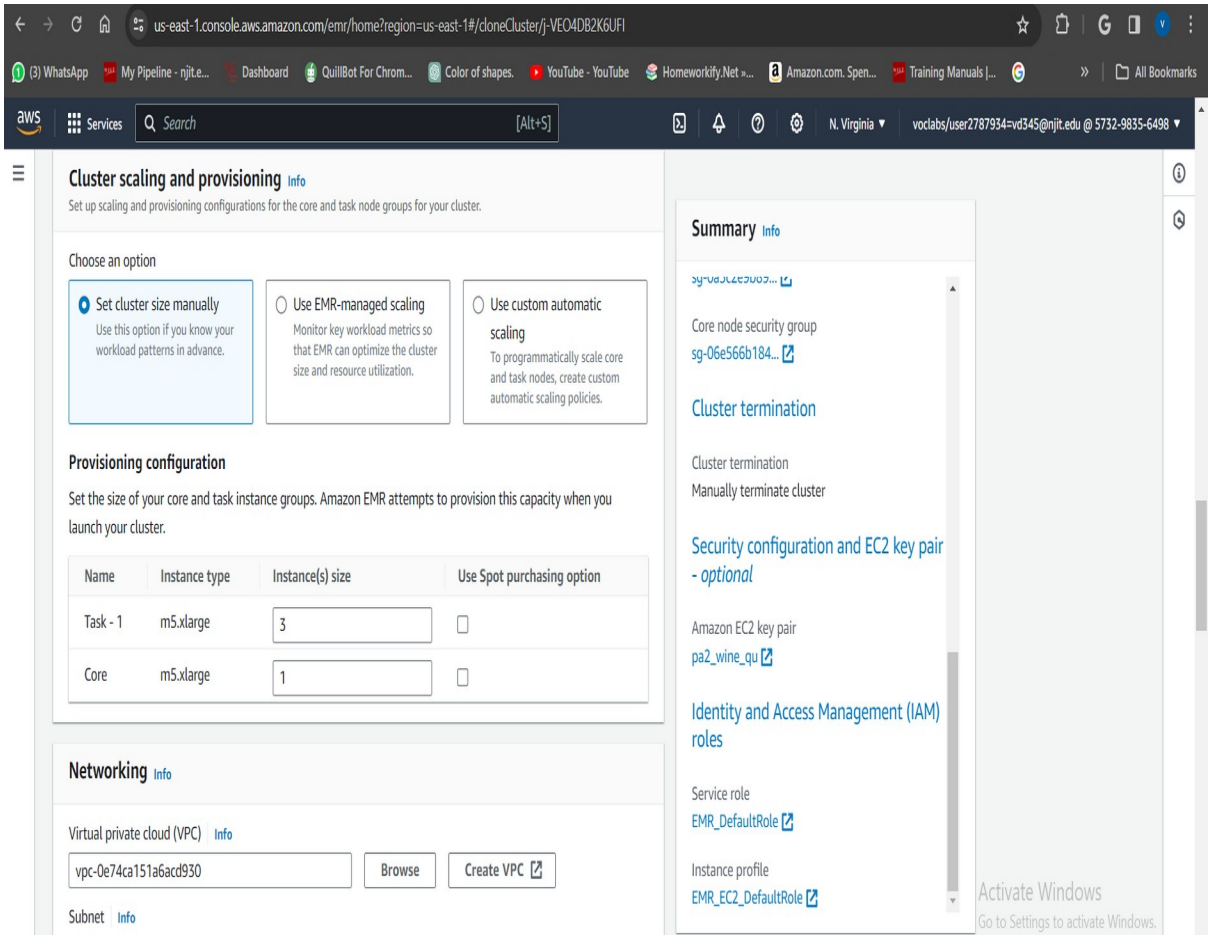
The remaining settings can be left as-is.

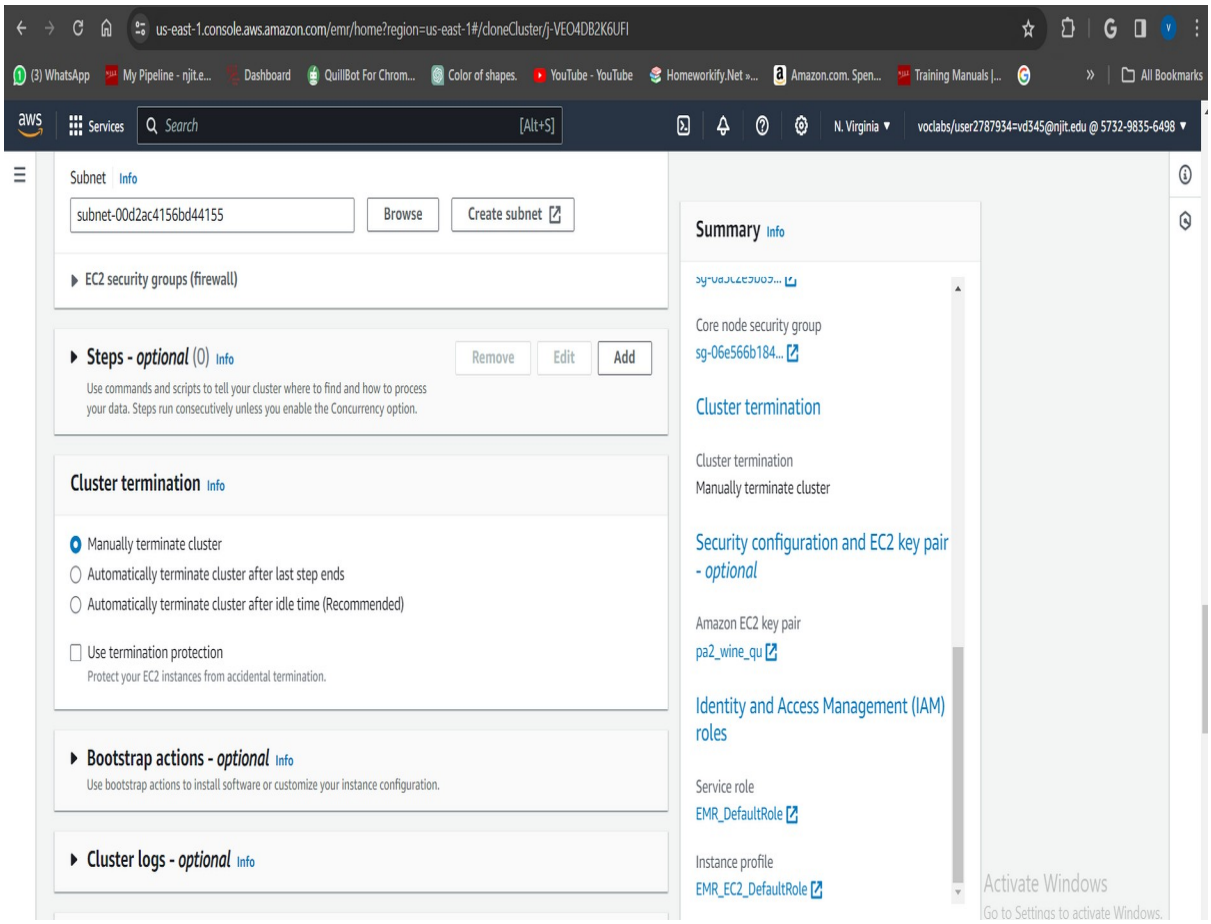
4. Following a successful cluster creation, the cluster status should be "Waiting."

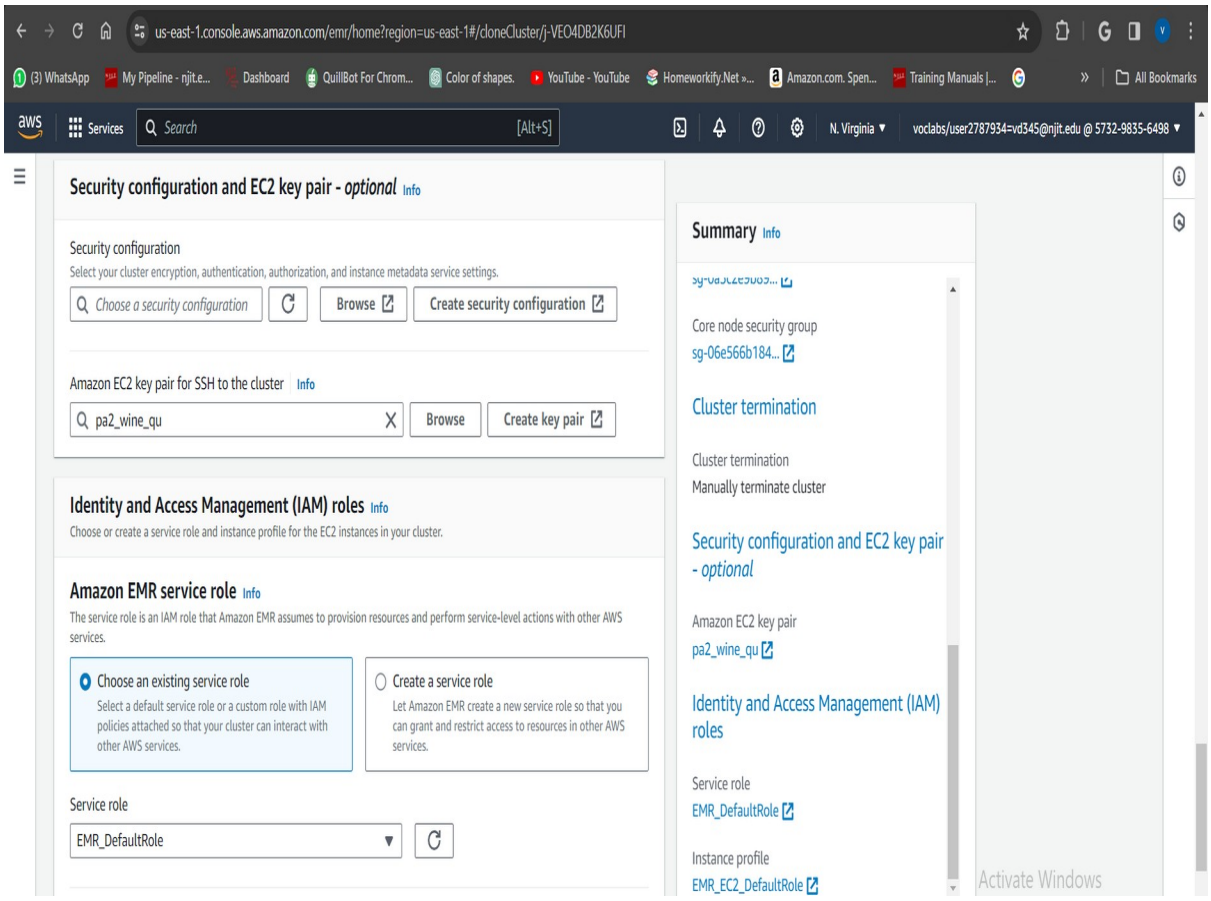

us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/cloneCluster/j-VEO4DB2K6UFI
Clone "pa2_wine_qu" Info
Name and applications Info
Name
pa2_wine_qu
Amazon EMR release Info
A release contains a set of applications which can be installed on your cluster.
emr-5.33.0
Application bundle
Spark
Core Hadoop
HBase
Presto
Custom
Flink 1.12.1
HCatalog 2.3.7
Hue 4.9.0
Livy 0.7.0
Oozie 5.2.0
Presto 0.245.1
TensorFlow 2.4.1
ZooKeeper 3.4.14
Ganglia 3.7.2
Hadoop 2.10.1
JupyterEnterpriseGateway 2.1.0
MXNet 1.7.0
Phoenix 4.14.3
Spark 2.4.7
Tez 0.9.2
HBase 1.4.13
Hive 2.3.7
JupyterHub 1.1.0
Mahout 0.13.0
Pig 0.17.0
Sqoop 1.4.7
Zeppelin 0.9.0
Summary Info
Name and applications
Name
pa2_wine_qu
Amazon EMR release
emr-5.33.0
Application bundle
Custom (Hadoop 2.10.1, JupyterEnterpriseGateway 2.1.0, Spark 2.4.7, Zeppelin 0.9.0)
Cluster configuration
Instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)
Cluster scaling and provisioning
Provisioning configuration
Core size: 1 instance
Activate Windows
Go to Settings to activate Windows.

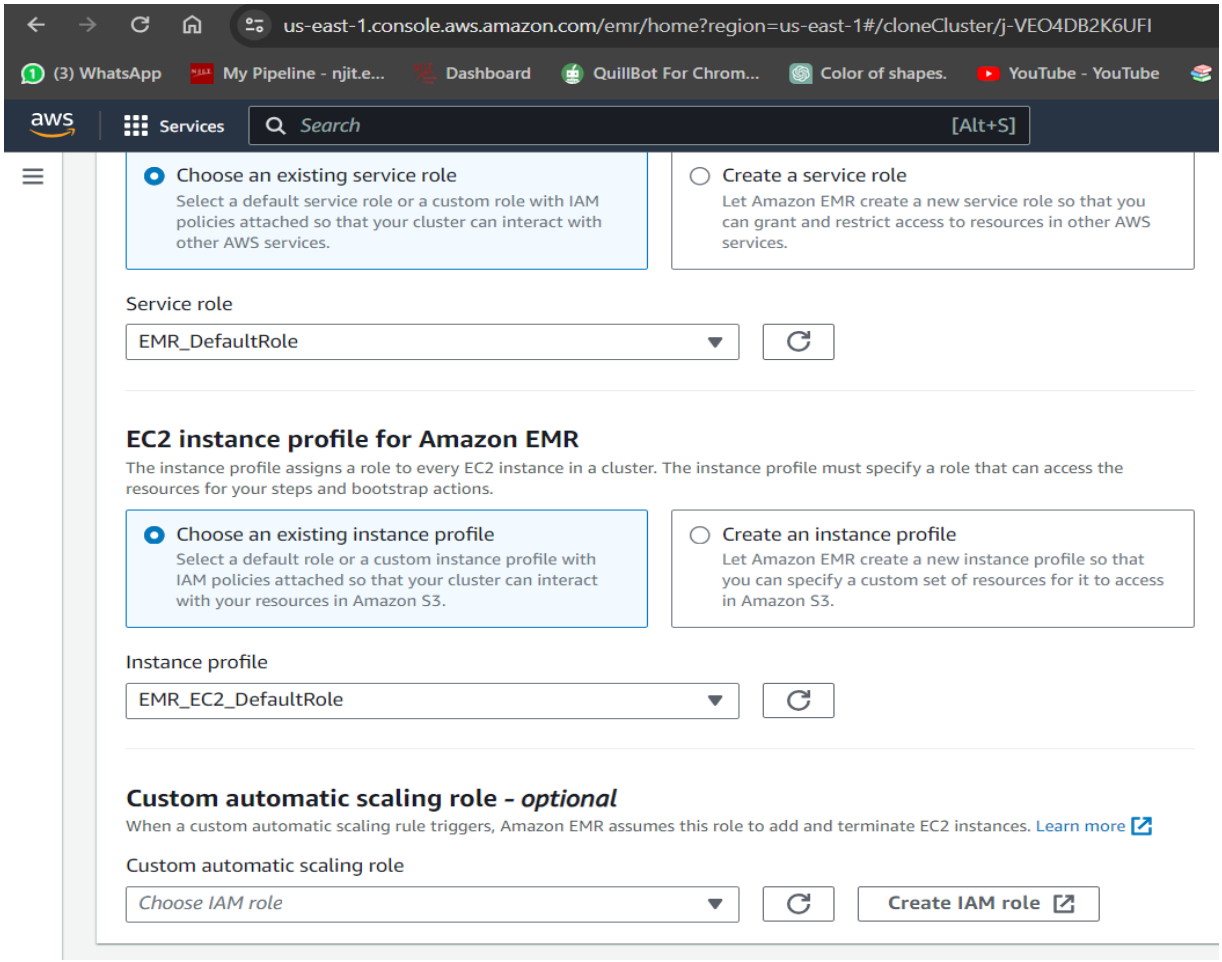

us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/cloneCluster/j-VEO4DB2K6UFI
AWS Glue Data Catalog settings
Use the AWS Glue Data Catalog to provide an external metastore for your application.
Use for Spark table metadata
Custom Amazon Machine Image (AMI) Info
Choose or enter an AMI ID
Update all installed packages on reboot
Cluster configuration Info
Choose a configuration method for the primary, core, and task node groups for your cluster.
Instance groups
Choose one instance type per node group
Instance fleets
Choose any combination of instance types within each node group
Instance groups
Primary
Choose EC2 instance type
m5.xlarge
4 vCore 16 GiB memory EBS only storage
On-Demand price: - Lowest Spot price: -
Actions
Summary Info
Name and applications
Name
pa2_wine_qu
Amazon EMR release
emr-5.33.0
Application bundle
Custom (Hadoop 2.10.1, JupyterEnterpriseGateway 2.1.0, Spark 2.4.7, Zeppelin 0.9.0)
Cluster configuration
Instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)
Cluster scaling and provisioning

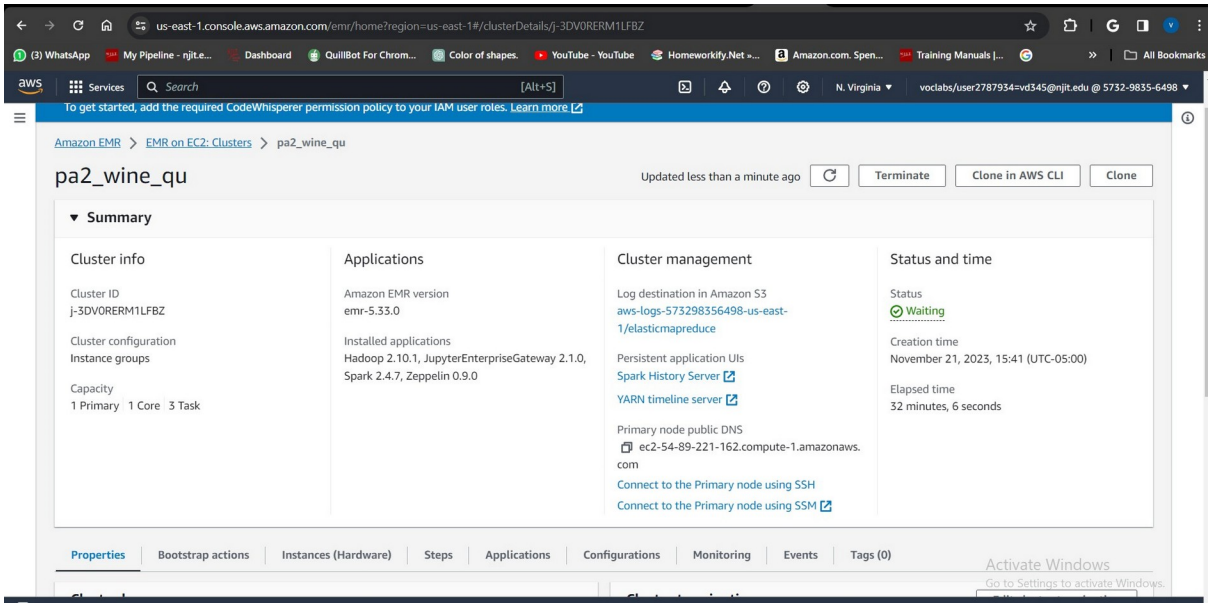

us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/cloneCluster/j-VEO4DB2K6UFI
Services
Search
[Alt+S]
N. Virginia
voclabs/user2787934=vd345@njit.edu @ 5732-9835-6498
Core
Choose EC2 instance type
m5.xlarge
4 vCore 16 GiB memory EBS only storage
On-Demand price: - Lowest Spot price: -
Actions
Node configuration - optional
Task 1 of 1
Remove instance group
Name
Task - 1
Choose EC2 instance type
m5.xlarge
4 vCore 16 GiB memory EBS only storage
On-Demand price: - Lowest Spot price: -
Actions
Node configuration - optional
Add task instance group
You can add up to 47 more task instance groups.
Summary Info
Name and applications
Name
pa2_wine_qu
Amazon EMR release
emr-5.33.0
Application bundle
Custom (Hadoop 2.10.1, JupyterEnterpriseGateway 2.1.0, Spark 2.4.7, Zeppelin 0.9.0)
Cluster configuration
Instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)
Cluster scaling and provisioning
Provisioning configuration
Core size: 1 instance
Activate Windows


us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/cloneCluster/j-VEO4DB2K6UFI
Services
Search
[Alt+S]
N. Virginia
voclabs/user2787934=vd345@njit.edu @ 5732-9835-6498
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
Task - 1
m5.xlarge
3
Core
m5.xlarge
1
Networking Info
Virtual private cloud (VPC) Info
vpc-0e74ca151a6acd930
Browse
Create VPC
Subnet Info
Summary Info
Core node security group
sg-06e566b184...
Cluster termination
Cluster termination
Manually terminate cluster
Security configuration and EC2 key pair - optional
Amazon EC2 key pair
pa2_wine_qu
Identity and Access Management (IAM) roles
Service role
EMR_DefaultRole
Instance profile
EMR_EC2_DefaultRole
Activate Windows
Go to Settings to activate Windows.


us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/cloneCluster/j-VEO4DB2K6UFI
Services
Search
[Alt+S]
N. Virginia
voclabs/user2787934=vd345@njit.edu @ 5732-9835-6498
Subnet Info
subnet-00d2ac4156bd44155
Browse
Create subnet
EC2 security groups (firewall)
Steps - optional (0) Info
Remove
Edit
Add
Use commands and scripts to tell your cluster where to find and how to process your data. Steps run consecutively unless you enable the Concurrency option.
Cluster termination Info
Manually terminate cluster
Automatically terminate cluster after last step ends
Automatically terminate cluster after idle time (Recommended)
Use termination protection
Protect your EC2 instances from accidental termination.
Bootstrap actions - optional Info
Use bootstrap actions to install software or customize your instance configuration.
Cluster logs - optional Info
Summary Info
Core node security group
sg-06e566b184...
Cluster termination
Cluster termination
Manually terminate cluster
Security configuration and EC2 key pair - optional
Amazon EC2 key pair
pa2_wine_qu
Identity and Access Management (IAM) roles
Service role
EMR_DefaultRole
Instance profile
EMR_EC2_DefaultRole
Activate Windows
Go to Settings to activate Windows.


us-east-1.console.aws.amazon.com/emr/home?region=us-east-1#/cloneCluster/j-VEO4DB2K6UFI
Services
Search
[Alt+S]
N. Virginia
voclabs/user2787934=vd345@njit.edu @ 5732-9835-6498
Security configuration and EC2 key pair - optional Info
Security configuration
Select your cluster encryption, authentication, authorization, and instance metadata service settings.
Choose a security configuration
Browse
Create security configuration
Amazon EC2 key pair for SSH to the cluster Info
pa2_wine_qu
Browse
Create key pair
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
EMR_DefaultRole
Summary Info
Core node security group
sg-06e566b184...
Cluster termination
Cluster termination
Manually terminate cluster
Security configuration and EC2 key pair - optional
Amazon EC2 key pair
pa2_wine_qu
Identity and Access Management (IAM) roles
Service role
EMR_DefaultRole
Instance profile
EMR_EC2_DefaultRole
Activate Windows





phase 4. How to train a machine learning model in a Spark cluster with four EC2 instances running concurrently.

1. When the cluster is ready to accept jobs, you can either use the step button to add steps or submit manually.

To submit manually, SSH to the cluster's Master using the following command:

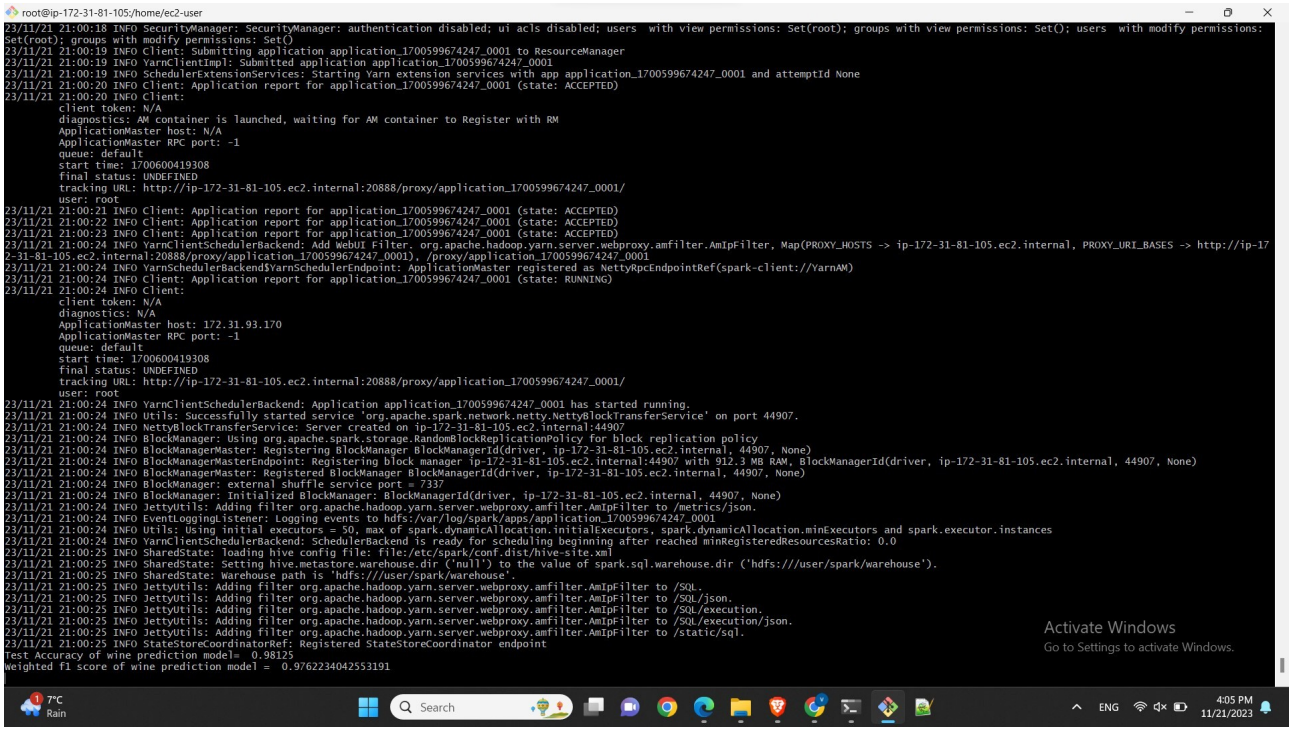
ssh -i "ec2key.pem" User>>@Public IPv4 DNS>>

2. On successful login to master , change to root user by running command:

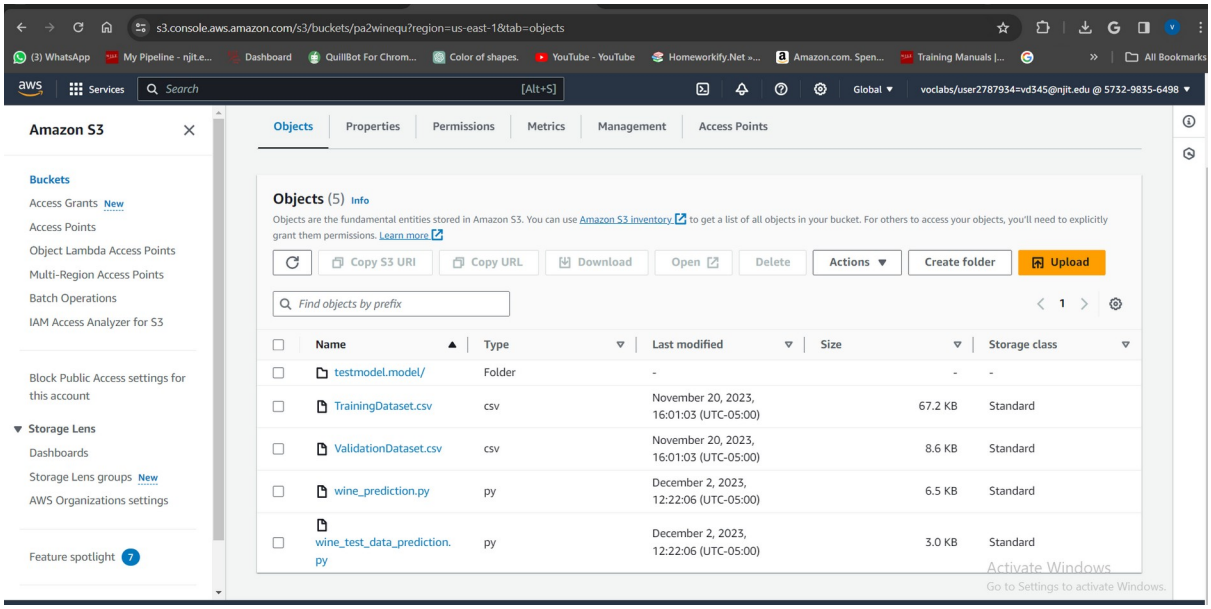
“ sudo su ”

3. Use the following command to submit the job:

spark-submit s3://pa2winequ/wine_prediction.py



4. You can trace status of this job in EMR UI application logs. Once status is succeded a test.model will be created in s3 bucket-s3://pa2winequ.



phase 5. Running the Machine learning model without using docker:

- Clone the Repository: Start by cloning this repository onto your local machine.

- Set Up Local Spark Environment: Ensure that you have a local Spark environment ready for running this application. If you haven't set up Spark yet, you can refer to the [official Spark documentation] (https://spark.apache.org/docs/latest) for instructions.

- Navigate to the Python File Folder: Go to the 'python file' directory within the cloned repository. Prepare Training Dataset and Place Training Dataset in the 'C:\pa2-cc\data\csv' folder.

phase 6. Run ML model using Docker.

1. Install docker where you want to run this container
2. A public image has been created and posted on DockerHub using cmd –

   docker build -t wine_prediction_app .

```
PS C:\Users\vijay\pySparkAWSWinePredictionApp-main> docker build -t wine-prediction-app .
[+] Building 124.9s (28/28) FINISHED                                                                    docker:default
 => [internal] load build definition from Dockerfile                                                             0.1s
 => => transferring dockerfile: 1.27kB                                                                           0.0s
 => [internal] load .dockerignore                                                                                0.1s
 => => transferring context: 2B                                                                                  0.0s
 => [internal] load metadata for docker.io/library/centos:7                                                      1.0s
 => [auth] library/centos:pull token for registry-1.docker.io                                                    0.0s
 => [ 1/22] FROM docker.io/library/centos:7@sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d32916d6487418e  0.2s
 => => resolve docker.io/library/centos:7@sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d32916d6487418ea4  0.1s
 => => sha256:eeb6ee3f44bd0b5103bb561b4c16bcb82328cfe5809ab675bb17ab3a16c517c9 2.75kB / 2.75kB                   0.0s
 => => sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d32916d6487418ea4 1.20kB / 1.20kB                   0.0s
 => => sha256:dead07b4d8ed7e29e98de0f4504d87e8880d4347859d839686a31da35a3b532f 529B / 529B                       0.0s
 => [internal] load build context                                                                                0.1s
 => => transferring context: 688.19kB                                                                            0.1s
 => [ 2/22] RUN yum -y update && yum -y install python3 python3-dev python3-pip python3-virtualenv  java-1.8.0-o  79.7s
 => [ 3/22] RUN python -V                                                                                        0.5s
 => [ 4/22] RUN python3 -V                                                                                       0.5s
 => [ 5/22] RUN pip3 install --upgrade pip                                                                       2.5s
 => [ 6/22] RUN pip3 install numpy panda                                                                         5.1s
 => [ 7/22] RUN pip3 install pandas                                                                              6.0s
 => [ 8/22] RUN wget --no-verbose -O apache-spark.tgz "https://archive.apache.org/dist/spark/spark-3.1.2/spark-3  16.7s
 => [ 9/22] RUN ln -s /opt/spark-3.1.2-bin-hadoop2.7 /opt/spark                                                  0.4s
 => [10/22] RUN (echo 'export SPARK_HOME=/opt/spark' >> ~/.bashrc && echo 'export PATH=$SPARK_HOME/bin:$PATH' >>   0.6s
 => [11/22] RUN mkdir /code                                                                                      0.5s
 => [12/22] RUN mkdir /code/data                                                                                 0.5s
 => [13/22] RUN mkdir /code/data/csv                                                                             0.5s
 => [14/22] RUN mkdir /code/data/model                                                                           0.6s
 => [15/22] RUN mkdir /code/src                                                                                  0.5s
 => [16/22] RUN mkdir /code/data/testdata.model/                                                                 0.5s
 => [17/22] COPY src/wine_test_data_prediction.py /code/src                                                      0.0s
 => [18/22] COPY data/model/testdata.model/ /code/data/model/testdata.model                                      0.1s
 => [19/22] RUN rm /bin/sh && ln -s /bin/bash /bin/sh                                                            0.4s
 => [20/22] RUN /bin/bash -c "source ~/.bashrc"                                                                  0.5s
 => [21/22] RUN /bin/sh -c "source ~/.bashrc"                                                                    0.5s
 => [22/22] WORKDIR /code/                                                                                       0.0s
 => exporting to image                                                                                           6.7s
 => => exporting layers                                                                                          6.7s
 => => writing image sha256:333701a6bbceaa0c5ebe4409b3fa06683013fb8cfc0685fadbaec84a80769abf                     0.0s
 => => naming to docker.io/library/wine-prediction-app                                                           0.0s
```

1. You can see the image using cmd
   docker image ls

2. You can push this in docker hub repository
   docker push vijaya340/wine_prediction_app
3. A public image has been created and posted on Docker Hub. Use the command
   Docker pull vijaya340/wine_prediction_app
   to get the image on your machine.
4. Place your training dataset file in a folder (lets call it directory dirA) , which you will Count with docker container and run it using below cmd
   docker run -v
   C:/Users/vijay/pySparkAWSWinePredictionApp-main/src:/code/data/csv
   winepred TrainingDataset.csv

```
What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\Users\vijay\pySparkAWSWinePredictionApp-main> docker images
REPOSITORY                  TAG       IMAGE ID       CREATED         SIZE
wine-prediction-app         latest    333701a6bbce   59 seconds ago   1.21GB
ubuntu                      latest    e4c58958181a   6 weeks ago      77.8MB
dfordeepika/awssparkwineapp   latest    1d641e9d364e   2 years ago      1.07GB
PS C:\Users\vijay\pySparkAWSWinePredictionApp-main> docker run -v C:/Users/vijay/Desktop/pa2:/code/data/csv wine-prediction-app TrainingDataset.csv
23/11/21 03:40:25 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
23/11/21 03:40:29 INFO SparkContext: Running Spark version 3.1.2
23/11/21 03:40:29 INFO ResourceUtils: ==============================================================
23/11/21 03:40:29 INFO ResourceUtils: No custom resources configured for spark.driver.
23/11/21 03:40:29 INFO ResourceUtils: ==============================================================
23/11/21 03:40:29 INFO SparkContext: Submitted application: tanvi_cs643_wine_prediction
23/11/21 03:40:29 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores -> name: cores, amount: 1, script: , vendor: , memory
 -> name: memory, amount: 1024, script: , vendor: , offHeap -> name: offHeap, amount: 0, script: , vendor: ), task resources: Map(cpus -> name: cpus, amount
: 1.0)
23/11/21 03:40:29 INFO ResourceProfile: Limiting resource is cpu
23/11/21 03:40:29 INFO ResourceProfileManager: Added ResourceProfile id: 0
23/11/21 03:40:30 INFO SecurityManager: Changing view acls to: root
23/11/21 03:40:30 INFO SecurityManager: Changing modify acls to: root
23/11/21 03:40:30 INFO SecurityManager: Changing view acls groups to:
23/11/21 03:40:30 INFO SecurityManager: Changing modify acls groups to:
23/11/21 03:40:30 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(root); groups with vie
w permissions: Set(); users  with modify permissions: Set(root); groups with modify permissions: Set()
23/11/21 03:40:30 INFO Utils: Successfully started service 'sparkDriver' on port 36373.
23/11/21 03:40:30 INFO SparkEnv: Registering MapOutputTracker
23/11/21 03:40:30 INFO SparkEnv: Registering BlockManagerMaster
23/11/21 03:40:30 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
23/11/21 03:40:30 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
23/11/21 03:40:30 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
23/11/21 03:40:30 INFO DiskBlockManager: Created local directory at /tmp/blockmgr-a622e406-16d8-47ad-9c67-56b322f7f45f
23/11/21 03:40:30 INFO MemoryStore: MemoryStore started with capacity 366.3 MiB
23/11/21 03:40:30 INFO SparkEnv: Registering OutputCommitCoordinator
23/11/21 03:40:31 INFO Utils: Successfully started service 'SparkUI' on port 4040.
23/11/21 03:40:31 INFO SparkUI: Bound SparkUI to 0.0.0.0, and started at http://0369ac72f56c:4040
23/11/21 03:40:32 INFO Executor: Starting executor ID driver on host 0369ac72f56c
23/11/21 03:40:32 INFO Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 32979.
23/11/21 03:40:32 INFO NettyBlockTransferService: Server created on 0369ac72f56c:32979
```

phase 7. Conclusion

After running the models with docker the quality of wine predicted to be

	With Docker
Accuracy	98.358
F1 Score	97.766

From the above table we concluded with docker the accuracy of the model is yielding accuracy is 98.358