Wine Quality Prediction System

Distributed Machine Learning with Apache Spark and Docker

Project Overview

This project focuses on building a **distributed machine learning application** that predicts wine quality based on its physical and chemical properties. By leveraging the power of **Apache Spark**, **Amazon AWS**, and **Docker**, this project demonstrates an end-to-end solution for training, validating, and deploying machine learning models in a scalable, automated, and portable manner.

Key Features:

Distributed Model Training on AWS:
Training the wine quality prediction model using Apache Spark in a distributed environment on AWS EMR clusters, enabling parallel processing for faster computations.

Prediction Application:
A standalone application that uses trained models to predict wine quality.
Accepts a test file as input and outputs key metrics like the F1 score to evaluate model performance.

Containerized Deployment with Docker:
The prediction application is packaged into a Docker container, ensuring smooth and consistent deployment across different platforms.

CI/CD Integration:
Automated workflows using Jenkins and Kubernetes for efficient model deployment and management.

Technologies Used:

- **Apache Spark**: For distributed data processing and machine learning.
- **Amazon AWS (EMR, S3):** To train models in a distributed cluster environment.
- **Docker:** To package the prediction application into a portable container.
- **Jenkins and Kubernetes:** To automate deployment and manage scalability.

Some Important Links:

Category	Details
GitHub Repository	https://github.com/mewawalaabdeali/Wine_Prediction_Distributed_System_Apache_Spark.git
DockerHub Repository	https://hub.docker.com/r/aliabde24/wine_xgb
Training File	wine_quality_training_pa2_submission.py
Prediction file	wine_prediction_pa2_submission.py
Training Dataset	s3://winepredictionabdeali/TrainingDataset.csv
Validation Dataset	s3://winepredictionabdeali/ValidationDataset.csv
Different Models	s3://winepredictionabdeali/Testing_models/
Stable Image	docker pull aliabde24/wine_xgb
Stable Model	RandomForestModel_20241210032258

Overview:

Step No.	Section	Sub-Section	Description
1	Environment Setup	Development Setup	Initial setup using Databricks and VScode for modularization.
2		Running Setup (EMR Cluster)	Instructions to set up and run the project on an EMR cluster, including a bootstrap script.
3		Standalone Setup	Instructions for running the project on a standalone machine.
4		Dockerizing	Steps to build and push Docker containers for the project, including setting up Dockerfile and dependencies.
5		Jenkins Setup(Optional)	Steps to configure Jenkins for CI/CD pipeline, including parameter changes for different environments.
6		Kubernetes Setup(Optioal)	Configuring Kubernetes with Jenkins pipeline integration.
7	Running the Project	Training	Training models with setup on EMR clusters and standalone machines.
8		Prediction	Running predictions on standalone machines, Docker, and Kubernetes.
9		Model Optimization	Applying regularization, tuning, and optimization techniques for better model performance.
10		Jenkins Pipeline Automation	Steps for automating the entire process using Jenkins, from model training to deployment.
11		Kubernetes Integration	Explanation of how Kubernetes fetches parameters from Jenkins pipeline and integrates into the setup.
12	Docker Run	Running Docker Container	How to run the docker Container

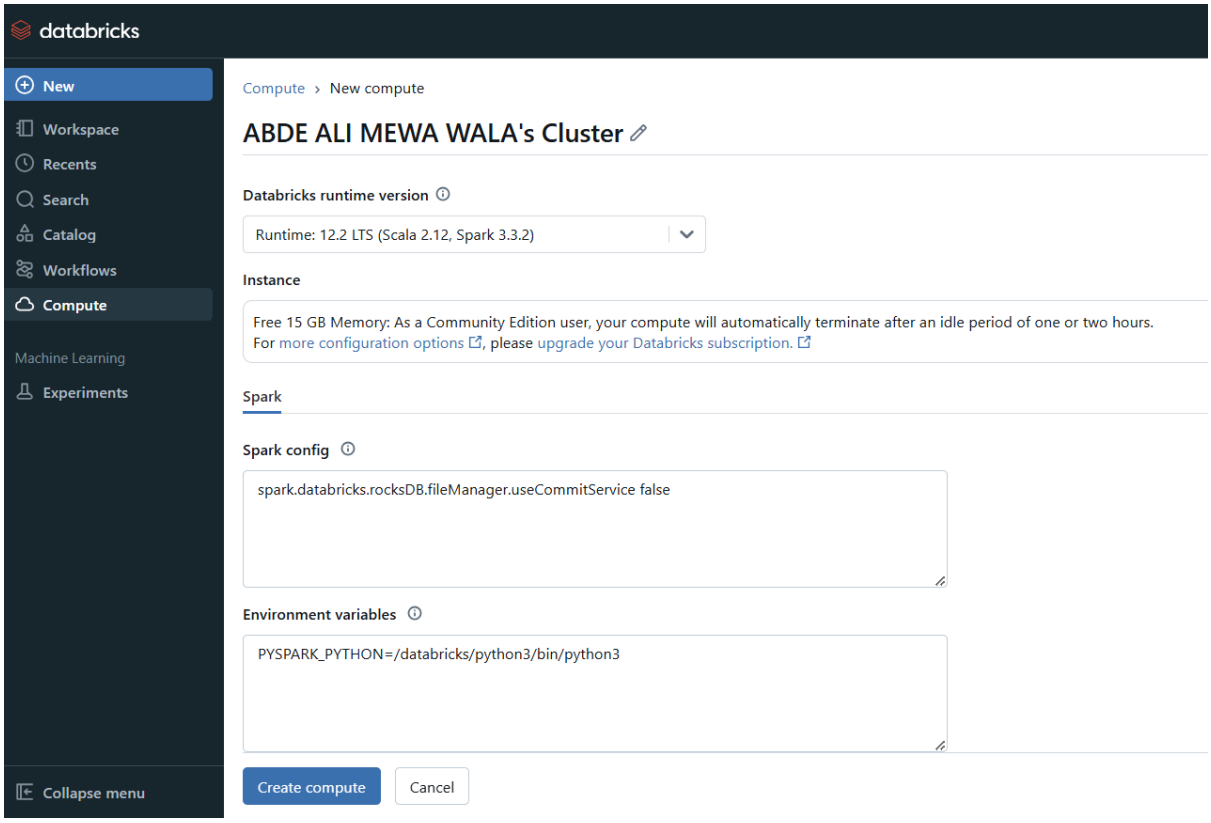
Development Setup:

Databricks Project Setup

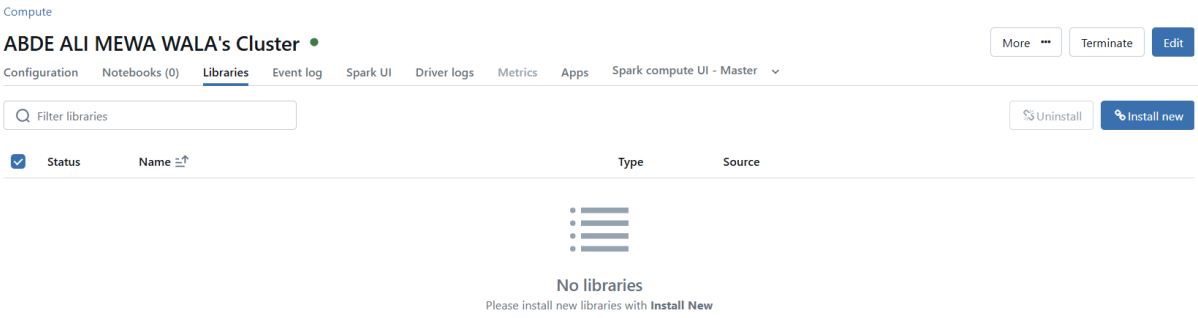
Initial Models created with Databricks:

WineProductionModel
WineProductionModelDatabricks

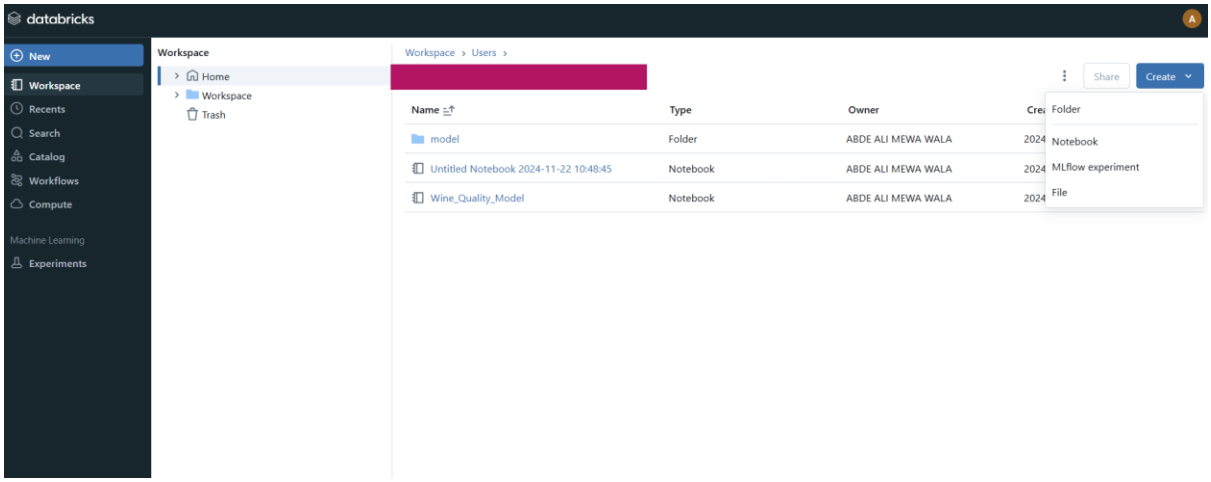
1. **Create a Databricks Account**
   - Go to Databricks and sign up for an account or log in if you already have one.
2. **Create a Databricks Workspace**
   - After logging in, you'll be directed to the Databricks workspace. If you're setting up a new workspace, click **Create Workspace**.
   - Select a region, and you can proceed with the default settings.
3. **Set Up a Cluster**
   - In the Databricks workspace, click on **Compute** from the left-hand side menu.
   - Click on **Create Compute**.
   - Select the **Databricks Runtime version**.

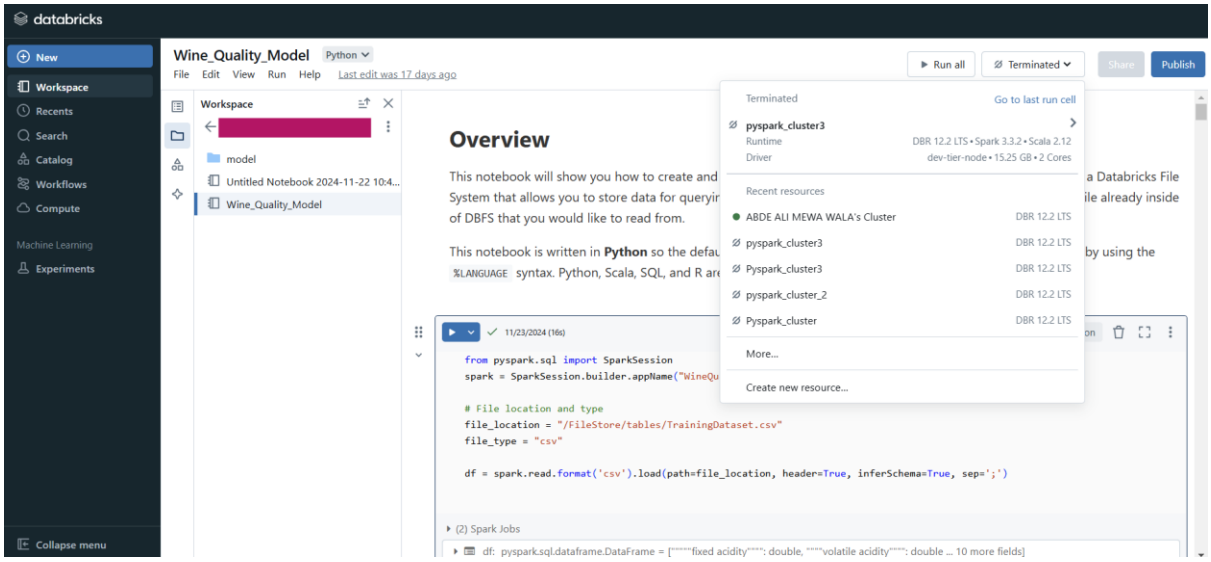


4. **Install Required Libraries in Databricks**
   - Once the cluster is created, click on the **libraries** tab.
   - Install the required libraries such as pandas, numpy, scikit-learn, findspark, pyarrow, boto3, etc., using the **PyPI** tab.
   - For example: Install pandas via pip install pandas.



5. **Setting up the Notebooks**
   - Click on **Workspace** > **Create** > **Notebook** to create a new notebook.
   - Use the Python language for your notebook.



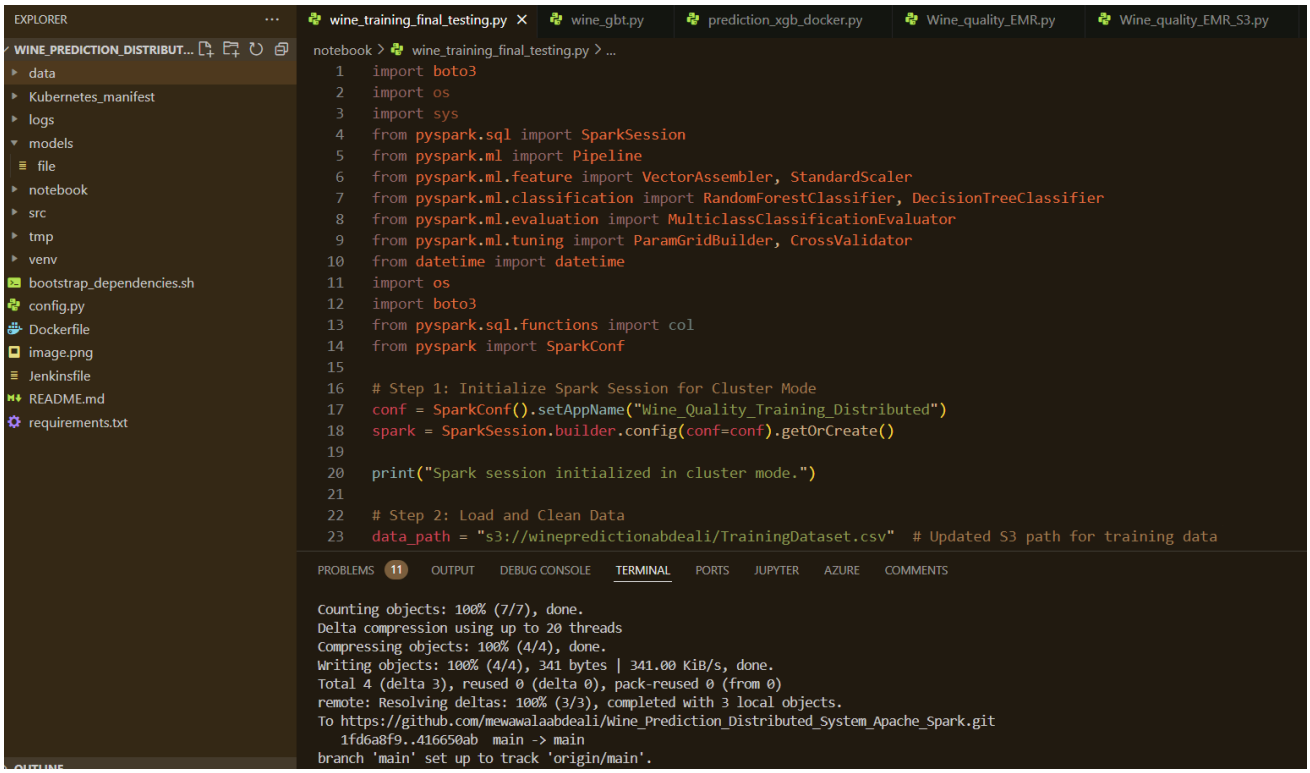


• You can save this model as ipynb or sourcefile (.py).

VSCode Project Setup
Github : [WinePredictionProject]

Step	Description	Commands/Details
1	Install VSCode	Download and install VSCode from VSCode Official Website.
2	Install Required Extensions	Use Extensions Marketplace to install: Python, Pylance, GitLens, and Docker (optional).
3	Install Python and Set Up Virtual Environment	Install Python if not installed. Create and activate a virtual environment:
		conda -p create venv/ conda activate venv/
4	Clone the Git Repository	Clone the project repository to your local machine:
		git clone https://github.com/mewawalaabdeali/Wine_Prediction_Distributed_System_Apache_Spark.git

5	Open the Project in VSCode	Navigate to the project folder: cd Wine_Prediction_Distributed_System_Apache_Spark. Open VSCode:



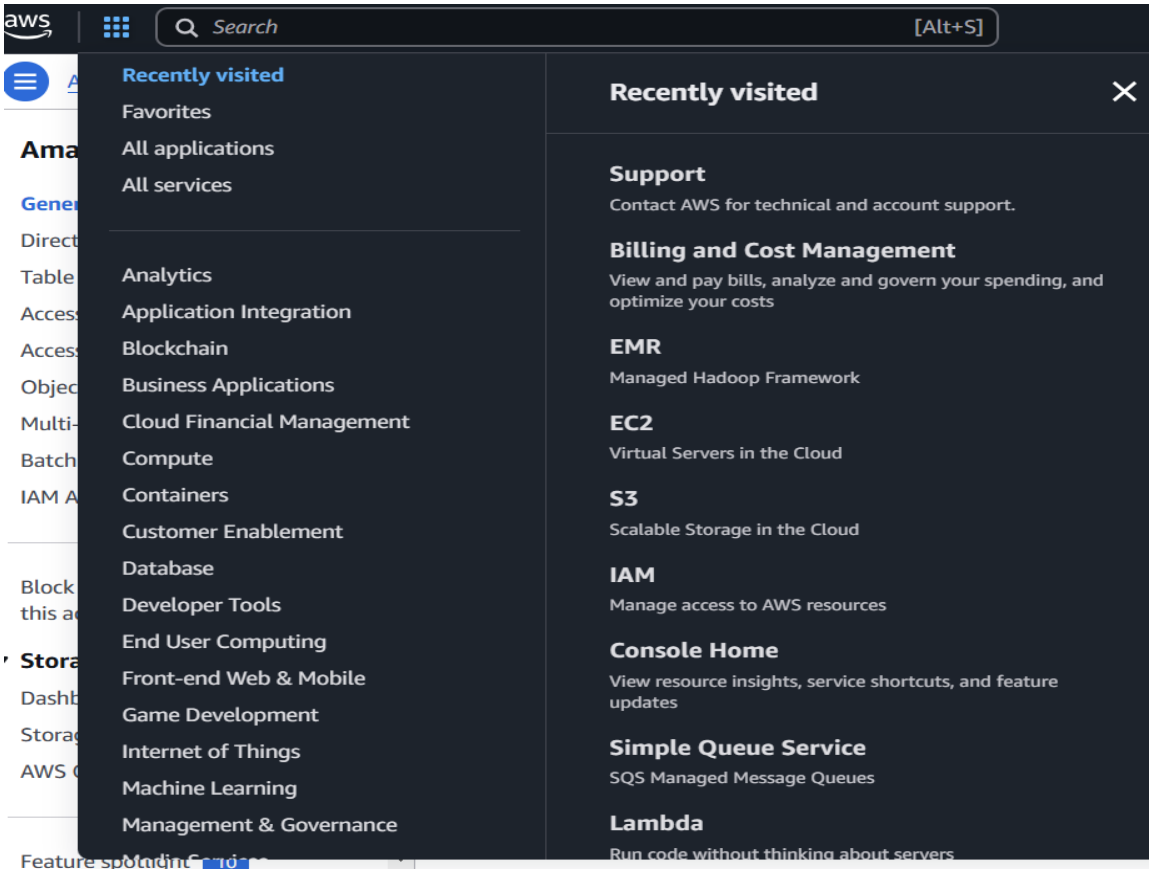
Prerequisites Before Setting Up an EMR Cluster

1. AWS Account

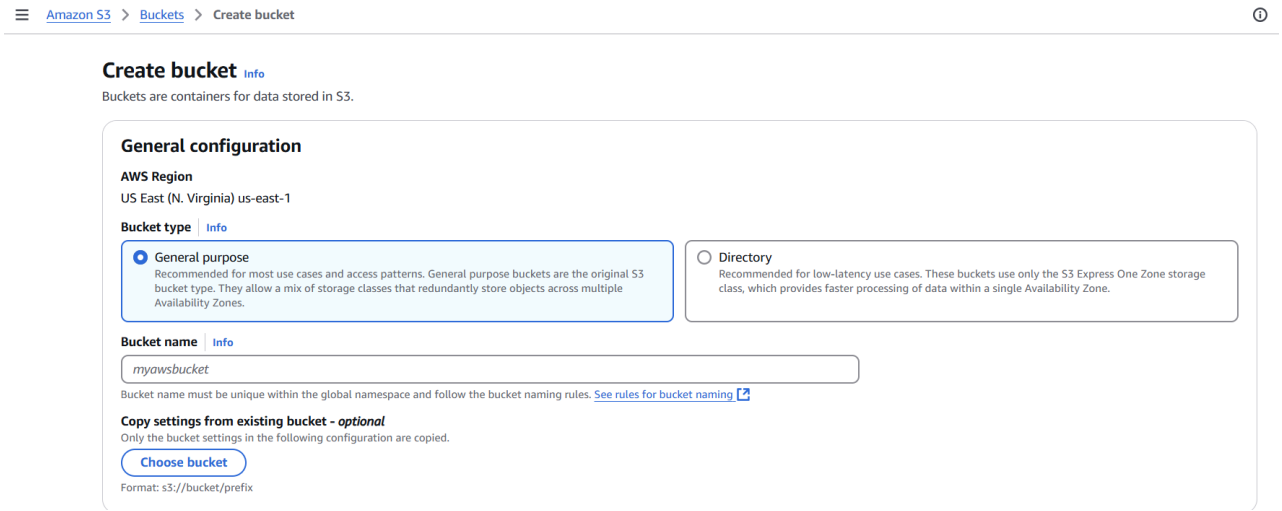
- Ensure you have an active AWS account with administrative access.

2. Set Up an S3 Bucket

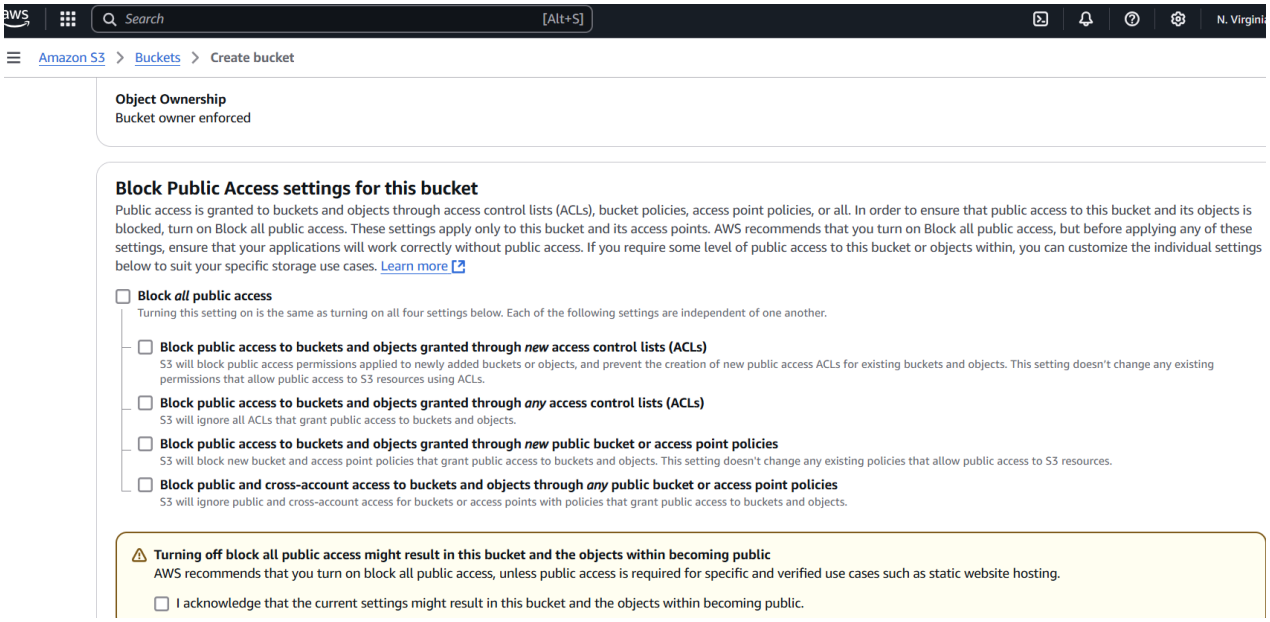
1. Navigate to the **S3** service in the AWS Management Console.



2. Click **"Create bucket"** and configure:
    - **Bucket name**: Choose a unique name (e.g., wine-prediction-emr-bucket).
    - **Region**: Select the same region where the EMR cluster will be created.



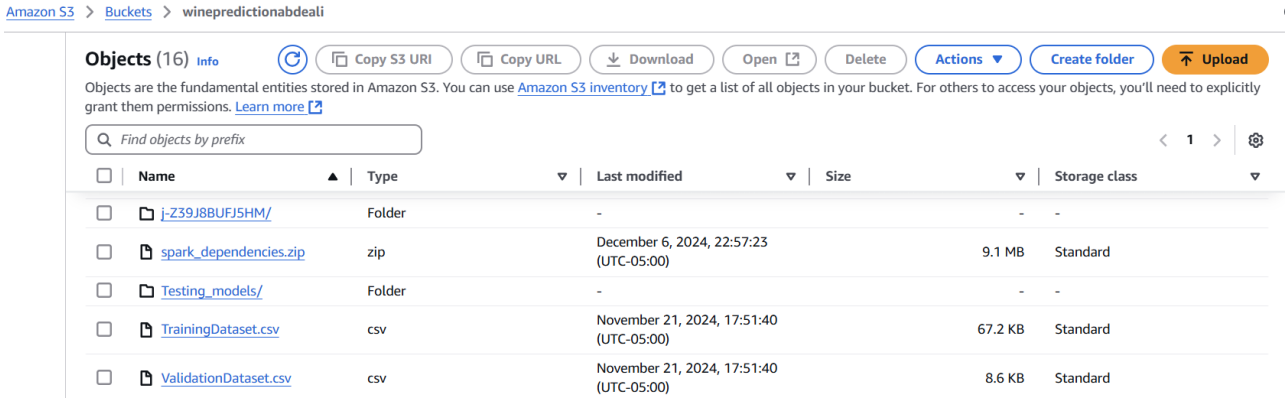
3. Configure **Bucket Options**:
    - Leave default settings or enable versioning, encryption, and public access restrictions as needed.
    - Untick “Block all public access” to make it public.



4. Click **"Create bucket"**.

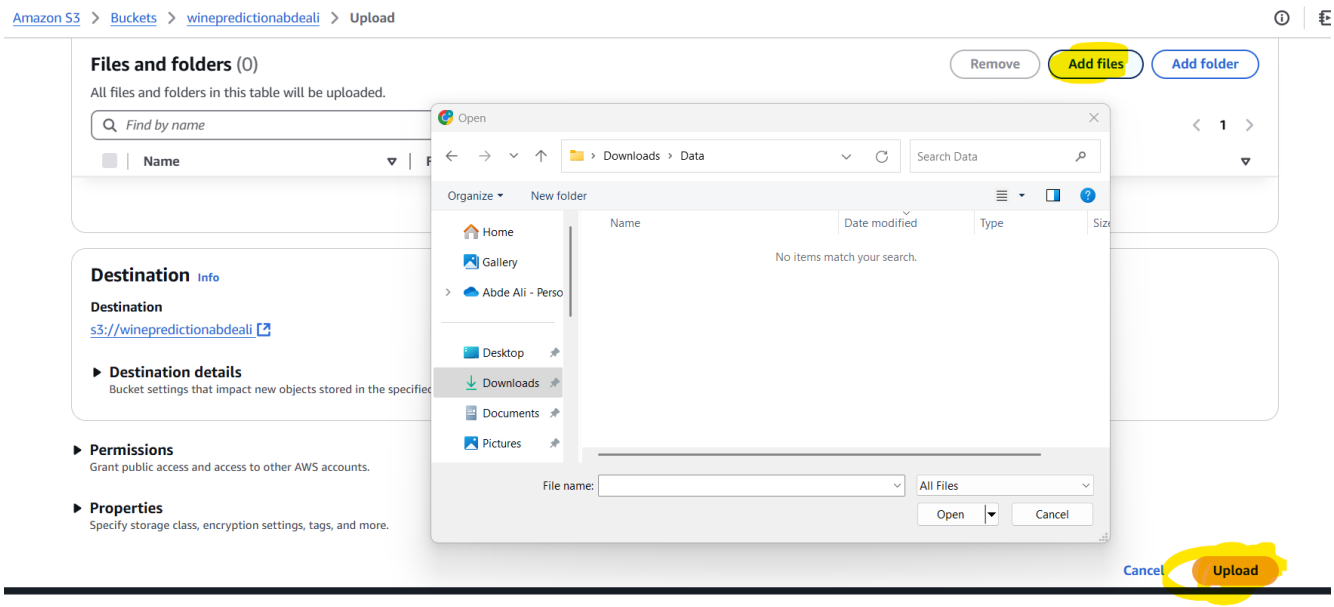
3. Upload Required Files to S3

- Upload the following files to the S3 bucket:
    1. **Bootstrap script**: bootstrap_dependencies.sh – Upload from project repository
    2. **Training dataset**. Upload from project repository
    3. **Validation datasets**. Upload from project repository



Steps to Upload:

1. Select the bucket and click **"Upload"**.
2. Drag and drop the files or browse to select them.
3. Confirm upload by clicking **"Upload"**.



4. Ensure Region Compatibility

- **S3 bucket and EMR cluster** must reside in the same AWS region to avoid latency and ensure seamless communication.

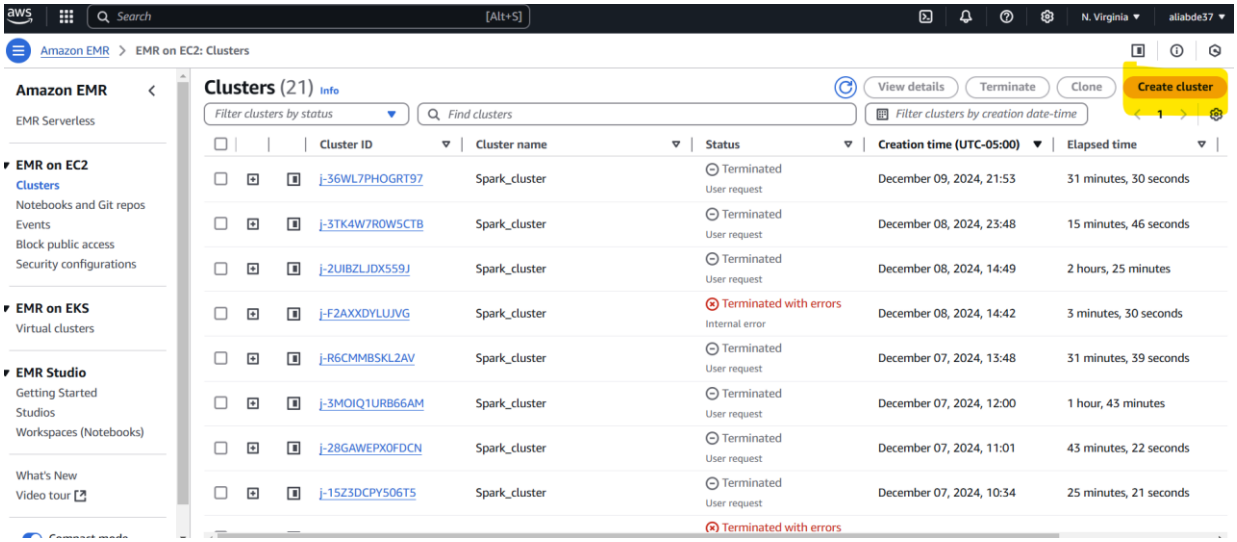
5. Configure IAM Roles

1. Go to the **IAM** service.
2. Ensure the following roles exist:
    - **EMR_DefaultRole**: Grants EMR the necessary permissions to access S3 and other AWS services.
    - **EMR_EC2_DefaultRole**: Allows the EC2 instances in the cluster to communicate with other AWS services.
3. If not, create the roles by navigating to **Roles** > **Create Role**.
4. In Permissions – Add S3FullPermission to give full read and write permission to the S3 bucket and the models would be saved to the S3.

Steps to Set Up an EMR Cluster

1. Navigate to the EMR Service

1. Log in to the **AWS Management Console**.
2. Search for **"EMR"** in the search bar and select the **Amazon EMR** service.

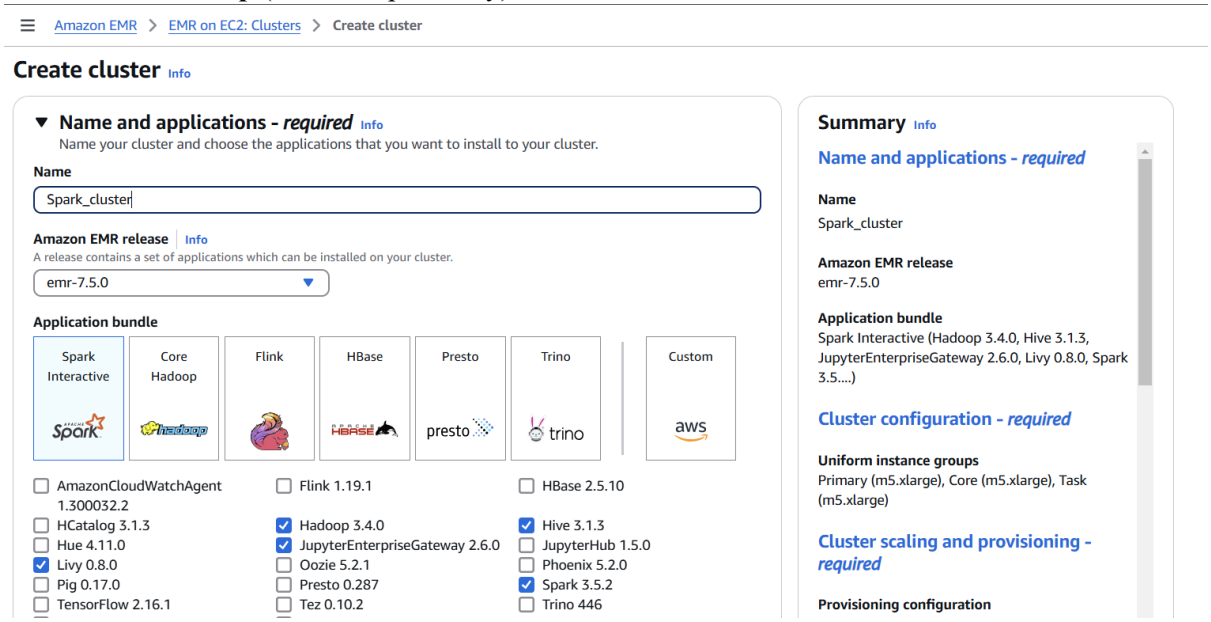


2. Create an EMR Cluster

1. Click on **"Create cluster"**.
2. Choose the **"Go to advanced options"** for detailed customization.

3. Select Software and Steps

1. **Software Configuration**:
   - **Release version**: Choose the latest stable release or the version compatible with your project (e.g., EMR 6.x).
   - **Applications**: Select the required applications:
     - **Spark** (required for PySpark jobs).
     - **Hadoop** (default dependency).



4. Configure Hardware

1. **Instance Type**:
   - Master node: Select a suitable instance type (e.g., m5.xlarge for medium workloads).
   - Core nodes: Add worker nodes (e.g., m5.xlarge, number depends on workload).
     - Use **Spot Instances** for cost savings if acceptable for your workload.

Cluster configuration - *required* Info

Choose a configuration method for the primary, core, and task node groups for your cluster.

- (●) **Uniform instance groups** — Choose the same EC2 instance type and purchasing option (On-Demand or Spot) for all nodes in your node group. Learn more
- ( ) **Flexible instance fleets** — Choose from the widest variety of provisioning options for the EC2 instances in your cluster. Diversify instance types and purchasing options, and use an allocation strategy. Learn more

Uniform instance groups

Primary

Choose EC2 instance type

m5.xlarge
4 vCore 16 GiB memory
EBS only storage
On-Demand price: $0.192 per instance/...
Lowest Spot price: $0.068 (us-east-1f)

Actions

[ ] Use high availability
Launch highly available, more resilient cluster with three primary nodes on On-Demand Instances. This configuration applies for the lifetime of your cluster. Learn more

▸ **Node configuration - *optional***

Core

Choose EC2 instance type

m5.xlarge
4 vCore 16 GiB memory
EBS only storage
On-Demand price: $0.192 per instance/...
Lowest Spot price: $0.068 (us-east-1f)

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: $0.192 per instance/...
Lowest Spot price: $0.068 (us-east-1f)

Actions

▸ **Node configuration - *optional***

Summary Info

Name and applications - *required*

Name
Spark_cluster

Amazon EMR release
emr-7.5.0

Application bundle
Spark Interactive (Hadoop 3.4.0, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.8.0, Spark 3.5....)

Cluster configuration - *required*

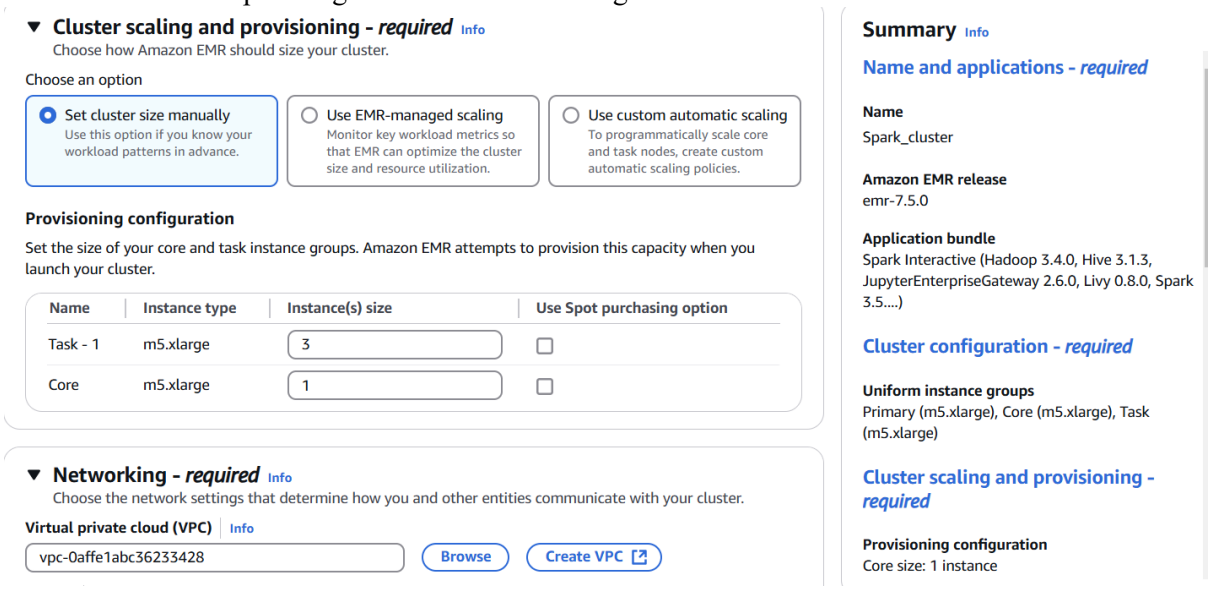
Uniform instance groups
Primary (m5.xlarge), Core (m5.xlarge), Task (m5.xlarge)

Cluster scaling and provisioning - *required*

Provisioning configuration
Core size: 1 instance

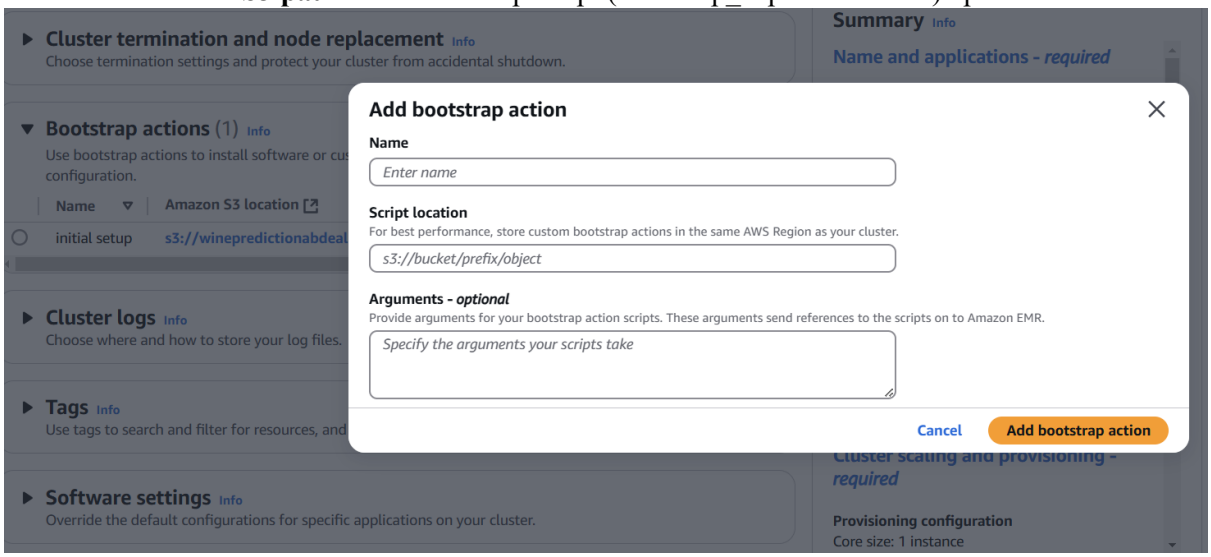
2. **Cluster Scaling**:

- Keep Configurations as shown in figure.



5. Bootstrap Actions

1. Click **"Add bootstrap action"**.
2. Provide the **S3 path** to the bootstrap script (bootstrap_dependencies.sh) uploaded earlier:

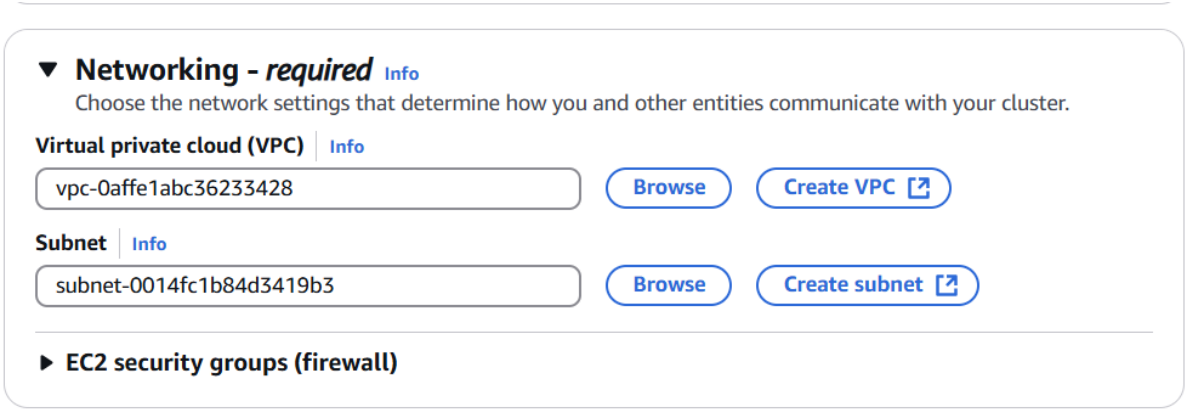


6. Specify Cluster Name

- Provide a descriptive name for the cluster (e.g., WinePredictionEMRCluster).

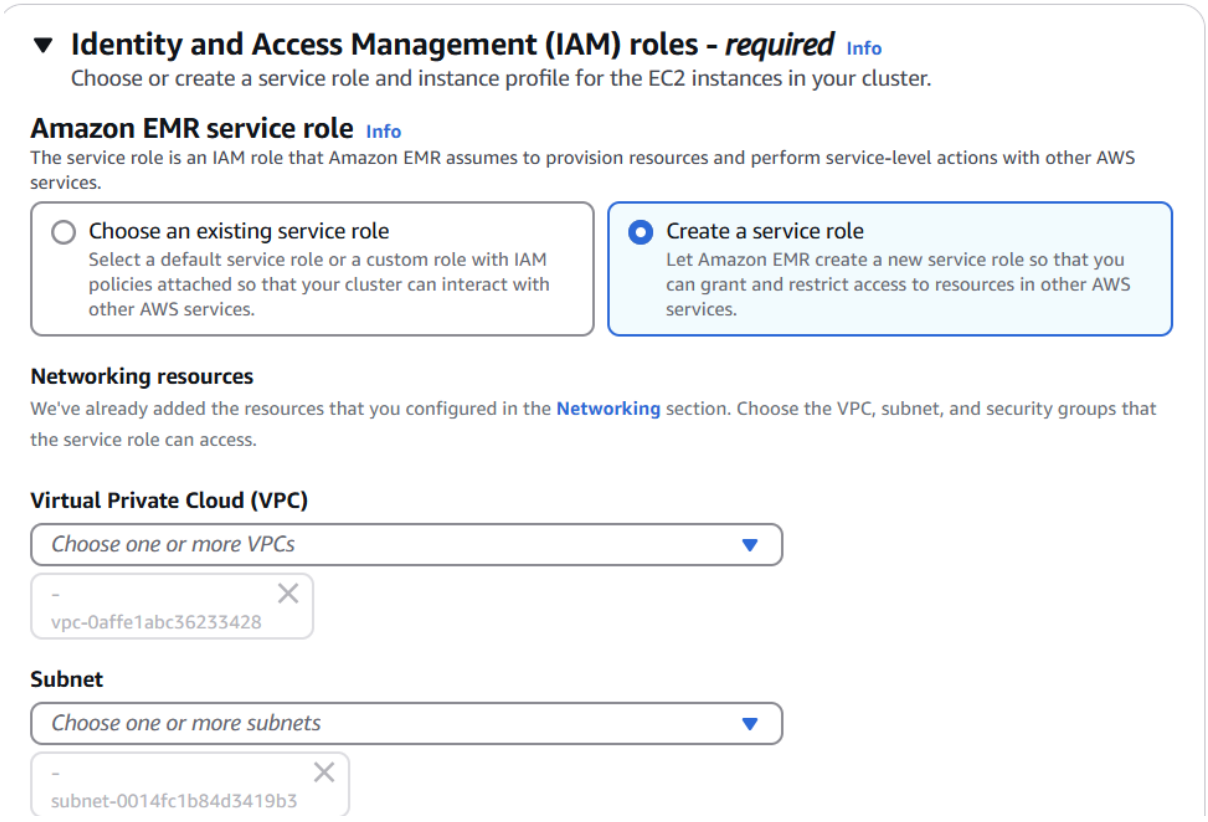
7. Networking

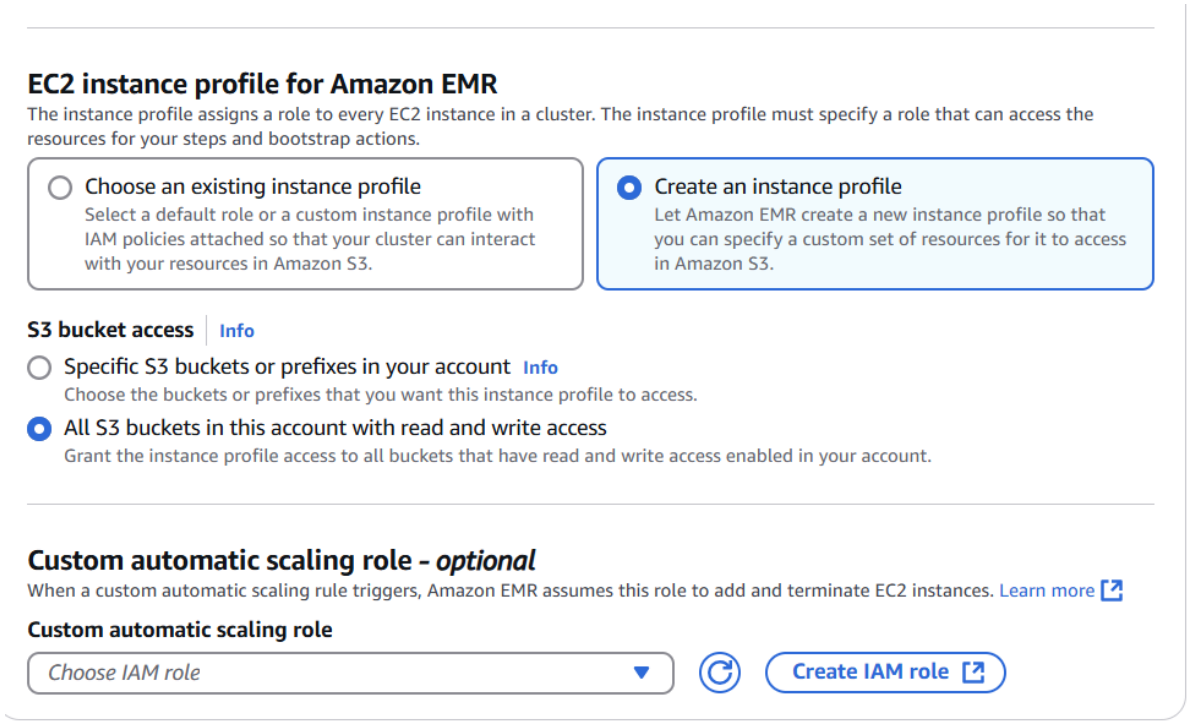
1. Ensure the **VPC** and **Subnet** are correctly configured:
   - Typically, select the default VPC and subnet.
2. Enable **Cluster Logging**:
   - Provide an S3 bucket path for logs (e.g., s3://<your-bucket-name>/logs).
3. Select the Keypair or create keypair, to login into the master machine



8. Security and Access

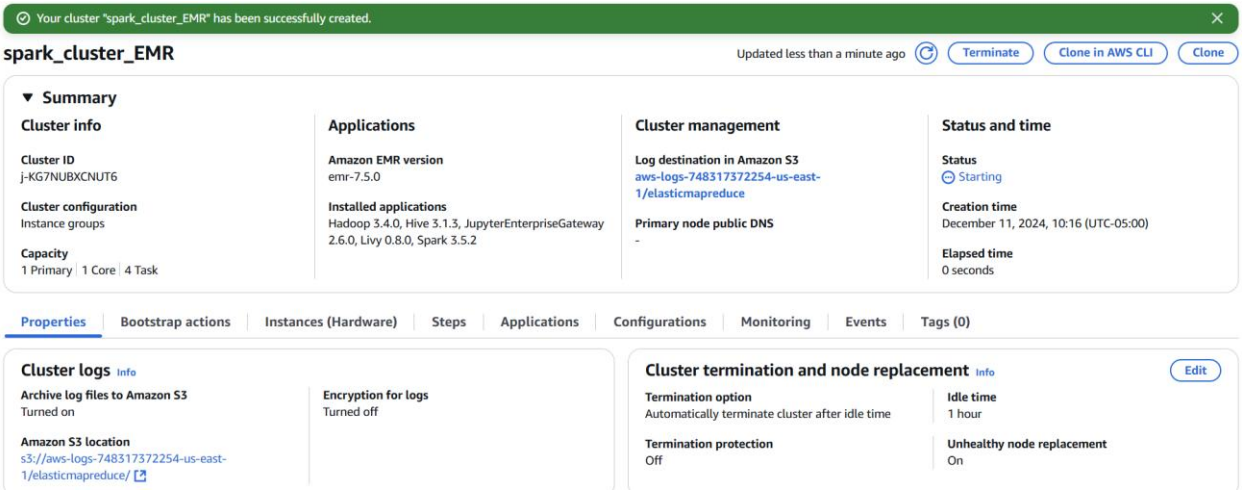
1. Select the default **EMR_DefaultRole** and **EMR_EC2_DefaultRole** IAM roles. It also has options to create or you can choose the one created in the above steps.
2. Optionally, configure EC2 Key Pair to access cluster nodes via SSH.

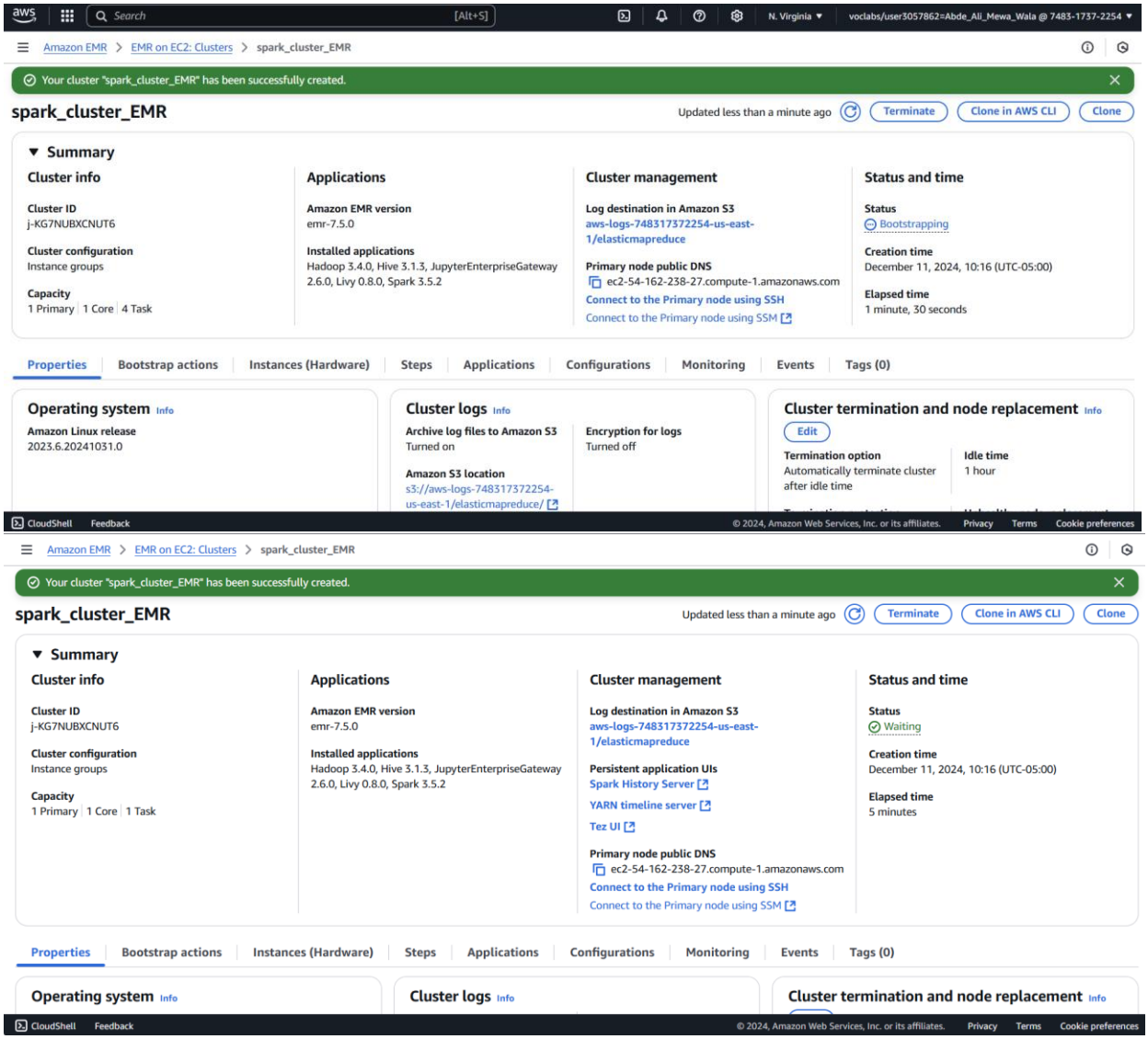




9. Launch the Cluster

1. Review the configuration and click **"Create cluster"**.
2. Wait for the status to change to **"Waiting"** (indicating the cluster is ready).





Standalone Machine Setup:

Machine OS: Ubuntu

Machine type: t3large

Task	Steps	Commands
Standalone Setup	Install Java Development Kit (JDK) and python-pip:	sudo apt update && sudo apt upgrade -y
		sudo apt install openjdk-11-jdk -y sudo apt install python3-pip -y pip3 install --upgrade pip
	Download and Install Apache Spark:	wget https://dlcdn.apache.org/spark/spark-3.5.0/spark-3.5.0-bin-hadoop3.tgz
		tar -xvf spark-3.5.0-bin-hadoop3.tgz
		sudo mv spark-3.5.0-bin-hadoop3 /opt/spark
	Set Up Environment Variables:	echo "export SPARK_HOME=/opt/spark" >> ~/.bashr
		echo "export PATH=\$PATH:\$SPARK_HOME/bin:\$SPARK_HOME/sbin" >> ~/.bashrc
		source ~/.bashrc
	Clone the Project Repository:	git clone https://github.com/mewawalaabdeali/Wine_Prediction_Distributed_System_Apache_Spark.git

		cd Wine_Prediction_Distributed_System_Apache_Spark
Dockerizing	Install Docker:	sudo apt update
		sudo apt install docker.io -y
		sudo systemctl start docker
		sudo systemctl enable docker
	Build the Docker Image:	docker build -t wine_prediction .
	Push the Docker Image to a Registry:	docker tag wine_prediction <your-dockerhub-username>/wine_prediction
		docker push <your-dockerhub-username>/wine_prediction
Jenkins Setup (Optional)	Install Jenkins and Start Service:	sudo apt update
		sudo apt install openjdk-11-jdk -y
		`wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key
	Access Jenkins Web Interface:	Open a browser and go to http://<your-server-ip>:8080
	Configure Jenkins Pipeline:	Use the Jenkinsfile from your project. Update parameters for S3 buckets and cluster details as needed.
Kubernetes Setup	Update the System and Install Docker:	sudo apt update
		sudo apt install -y apt-transport-https curl
		sudo apt install -y docker.io
		sudo systemctl start docker
		sudo systemctl enable docker
	Add Kubernetes Repository:	`curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg
	Install Kubernetes Components:	sudo apt update
		sudo apt install -y kubelet kubeadm kubectl
	Initialize Kubernetes Cluster:	sudo kubeadm init --pod-network-cidr=10.244.0.0/16
	Set Up kubectl for the Current User:	mkdir -p $HOME/.kube
		sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
		sudo chown $(id -u):$(id -g) $HOME/.kube/config
	Install a Pod Network Add-On (Flannel):	kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
	Verify Kubernetes Setup:	kubectl get nodes
		kubectl get pods -A
	Create Deployment YAML File for Model Deployment:	Create a file named wine-prediction-deployment.yaml with the following content:
		yaml apiVersion: apps/v1 kind: Deployment metadata: name: wine-prediction spec: replicas: 1 ...
	Deploy Model to Kubernetes:	kubectl apply -f wine-prediction-deployment.yaml

Task	Steps	Commands
Training on EMR Cluster	Run the training script using spark-submit in cluster mode:	spark-submit --deploy-mode client Wine_Quality_Cluster_mode.py
	Ensure S3 bucket paths and variables are correctly configured in the script.	Modify data_path and output_path in the script to point to your S3 bucket: s3://your-bucket-name/TrainingDataset.csv.
Training on Standalone	Run the training script on standalone mode with Spark:	spark-submit --master local[*] Wine_Quality_Standalone.py

	Verify dependencies are installed and environment is configured.	Use the bootstrap script provided during the standalone setup.
Verify Training Results	Inspect the saved model in the specified S3 path or local directory.	aws s3 ls s3://your-bucket-name/models/ (for EMR)
		ls models/ (for local runs)

Task	Steps	Commands
Prediction on Standalone Machine	Run the prediction script with validation dataset and model path as arguments:	spark-submit prediction.py --validation_file_path "s3://your-bucket-name/ValidationDataset.csv" --model_folder_name "RandomForestModel_20241210093000"
	Ensure the validation dataset is accessible (S3 or local).	Use aws s3 cp if the validation file is stored in S3:
		aws s3 cp s3://your-bucket-name/ValidationDataset.csv ./
	Inspect predictions uploaded to S3 or stored locally.	For S3: aws s3 ls s3://your-bucket-name/predictions/
		For local: ls predictions/

Task	Steps	Commands
Build Docker Image	Create the Docker image for the project:	docker build -t wine-prediction:latest .
	Verify the Docker image is built successfully.	docker images
Run Docker Container	Run the container for prediction or training:	docker run -it --rm -v $(pwd):/app -e S3_BUCKET_NAME="your-bucket-name" \ wine-prediction:latest
	Provide necessary environment variables for S3 bucket or paths in the container run.	Include -e flags for any required variables:
		-e MODEL_PATH="s3://your-bucket-name/models"
Push Docker Image	Tag and push the Docker image to a repository (e.g., DockerHub):	docker tag wine-prediction:latest your-dockerhub-username/wine-prediction:latest
		docker push your-dockerhub-username/wine-prediction:latest
Verify Container Logs	Check logs to debug or verify the results:	docker logs <container_id

Running the Project:

1. Training the Model

- Initialize the environment (either EMR cluster or standalone machine) as shown in setup instructions above.
- SSH into the machine.

```
24/12/09 04:42:52 INFO MetricsSystemImpl: s3a-file-system metrics system shutdown complete.
[hadoop@ip-172-31-32-108 ~]$ client_loop: send disconnect: Connection reset

C:\Users\asus\Downloads>ssh -i ~/Downloads/canvas_EMR.pem hadoop@ec2-54-162-238-27.compute-1.amazonaws.com
The authenticity of host 'ec2-54-162-238-27.compute-1.amazonaws.com (54.162.238.27)' can't be established.
ED25519 key fingerprint is SHA256:Nzqh3BsyVORgwwr30WkA4cGF0O5G3CXRs+CzHSu/3/w.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-162-238-27.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

A newer release of "Amazon Linux" is available.
  Version 2023.6.20241111:
  Version 2023.6.20241121:
  Version 2023.6.20241209:
Run "/usr/bin/dnf check-release-update" for full release and version update info
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'

EEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::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:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R::::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R::::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R::::RRRRRR:::::R
  E::::E             M:::::M    M:::M    M:::::M   R::::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R::::R      R::::R
EE:::::EEEEEEEE:::E M:::::M             M:::::M   R::::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-33-127 ~]$
```

- Ensure the training dataset is uploaded to the appropriate path (S3 for EMR, local for standalone).
- Run the below commands sequentially to setup the remaining configs on the cluster.
- sudo yum update -y

```
[hadoop@ip-172-31-33-127 ~]$ sudo yum update -y
Last metadata expiration check: 0:07:38 ago on Wed Dec 11 15:17:31 2024.
================================================================================================
WARNING:
  A newer release of "Amazon Linux" is available.

  Available Versions:

  Version 2023.6.20241111:
    Run the following command to upgrade to 2023.6.20241111:

      dnf upgrade --releasever=2023.6.20241111

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.6.20241111.html

  Version 2023.6.20241121:
    Run the following command to upgrade to 2023.6.20241121:

      dnf upgrade --releasever=2023.6.20241121

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.6.20241121.html

  Version 2023.6.20241209:
    Run the following command to upgrade to 2023.6.20241209:

      dnf upgrade --releasever=2023.6.20241209

    Release notes:
     https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.6.20241209.html

================================================================================================
Dependencies resolved.
```

- pip install pyspark pandas numpy findspark pyarrow matplotlib scikit-learn boto3

```
[hadoop@ip-172-31-33-127 ~]$ pip install pyspark pandas numpy findspark pyarrow matplotlib scikit-learn boto3
Defaulting to user installation because normal site-packages is not writeable
Collecting pyspark
  Downloading pyspark-3.5.3.tar.gz (317.3 MB)
     |████████████████████████████████| 317.3 MB 38 kB/s
  Preparing metadata (setup.py) ... done
Collecting pandas
  Downloading pandas-2.2.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.1 MB)
     |████████████████████████████████| 13.1 MB 55.3 MB/s
Requirement already satisfied: numpy in /usr/lib64/python3.9/site-packages (1.21.1)
Collecting findspark
  Downloading findspark-2.0.1-py2.py3-none-any.whl (4.4 kB)
Collecting pyarrow
  Downloading pyarrow-18.1.0-cp39-cp39-manylinux_2_28_x86_64.whl (40.1 MB)
     |████████████████████████████████| 40.1 MB 107.4 MB/s
Collecting matplotlib
  Downloading matplotlib-3.9.3-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (8.3 MB)
     |████████████████████████████████| 8.3 MB 102.5 MB/s
Collecting scikit-learn
  Downloading scikit_learn-1.6.0-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (13.5 MB)
     |████████████████████████████████| 13.5 MB 94.5 MB/s
Collecting boto3
  Downloading boto3-1.35.78-py3-none-any.whl (139 kB)
     |████████████████████████████████| 139 kB 92.0 MB/s
Collecting py4j==0.10.9.7
  Downloading py4j-0.10.9.7-py2.py3-none-any.whl (200 kB)
     |████████████████████████████████| 200 kB 105.8 MB/s
Collecting python-dateutil>=2.8.2
  Downloading python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
```

- pip install python-dateutil==2.8.2

```
[hadoop@ip-172-31-33-127 ~]$ pip install python-dateutil==2.8.2
Defaulting to user installation because normal site-packages is not writeable
Collecting python-dateutil==2.8.2
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     |████████████████████████████████| 247 kB 5.0 MB/s
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.9/site-packages
Installing collected packages: python-dateutil
  Attempting uninstall: python-dateutil
    Found existing installation: python-dateutil 2.9.0.post0
    Uninstalling python-dateutil-2.9.0.post0:
      Successfully uninstalled python-dateutil-2.9.0.post0
Successfully installed python-dateutil-2.8.2
[hadoop@ip-172-31-33-127 ~]$
```

- pip install --upgrade --force-reinstall awscli

```
[hadoop@ip-172-31-33-127 ~]$ pip install --upgrade --force-reinstall awscli
Defaulting to user installation because normal site-packages is not writeable
Collecting awscli
  Downloading awscli-1.36.19-py3-none-any.whl (4.5 MB)
     |████████████████████████████████| 4.5 MB 6.8 MB/s
Collecting rsa<4.8,>=3.1.2
  Downloading rsa-4.7.2-py3-none-any.whl (34 kB)
Collecting docutils<0.17,>=0.10
  Downloading docutils-0.16-py2.py3-none-any.whl (548 kB)
     |████████████████████████████████| 548 kB 112.8 MB/s
Collecting s3transfer<0.11.0,>=0.10.0
  Using cached s3transfer-0.10.4-py3-none-any.whl (83 kB)
Collecting PyYAML<6.1,>=3.10
  Downloading PyYAML-6.0.2-cp39-cp39-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (737 kB)
     |████████████████████████████████| 737 kB 99.5 MB/s
Collecting colorama<0.4.7,>=0.2.5
  Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Collecting botocore==1.35.78
  Using cached botocore-1.35.78-py3-none-any.whl (13.2 MB)
Collecting urllib3<1.27,>=1.25.4
  Downloading urllib3-1.26.20-py2.py3-none-any.whl (144 kB)
     |████████████████████████████████| 144 kB 109.0 MB/s
Collecting python-dateutil<3.0.0,>=2.1
  Using cached python_dateutil-2.9.0.post0-py2.py3-none-any.whl (229 kB)
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting pyasn1>=0.1.3
  Downloading pyasn1-0.6.1-py3-none-any.whl (83 kB)
     |████████████████████████████████| 83 kB 4.6 MB/s
Collecting six>=1.5
  Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: six, urllib3, python-dateutil, jmespath, pyasn1, botocore, s3transfer,
  Attempting uninstall: python-dateutil
    Found existing installation: python-dateutil 2.8.2
    Uninstalling python-dateutil-2.8.2:
      Successfully uninstalled python-dateutil-2.8.2
```

- nano ~/.bashrc
- export PATH=$PATH:/home/hadoop/.local/bin

```
# Source global definitions
if [ -f /etc/bashrc ]; then
  . /etc/bashrc
fi

_set_aws_region() {
  local curl_opts="--retry 5 -f --silent --connect-timeout 2"
  local token=$(curl ${curl_opts} -X PUT "http://169.254.169.254/latest/api/token
  export AWS_DEFAULT_REGION=$(curl ${curl_opts} -H "X-aws-ec2-metadata-token: ${t

  unset -f $FUNCNAME
}

# set the default region for the AWS CLI
_set_aws_region
export JAVA_HOME=/etc/alternatives/jre
export HOME=/home/hadoop
export PATH=$PATH:/home/hadoop/.local/bin
```

- source ~/.bashrc
- sudo yum install git -y

```
[hadoop@ip-172-31-33-127 ~]$ nano ~/.bashrc
[hadoop@ip-172-31-33-127 ~]$ source ~/.bashrc
[hadoop@ip-172-31-33-127 ~]$ sudo yum install git -y
Last metadata expiration check: 0:15:08 ago on Wed Dec 11 15:17:31 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture        Version                  Reposi
================================================================================
Installing:
 git                     x86_64              2.40.1-1.amzn2023.0.3    amazor
Installing dependencies:
 git-core                x86_64              2.40.1-1.amzn2023.0.3    amazor
 git-core-doc            noarch              2.40.1-1.amzn2023.0.3    amazor
 perl-Error              noarch              1:0.17029-5.amzn2023.0.2 amazor
 perl-Git                noarch              2.40.1-1.amzn2023.0.3    amazor
 perl-TermReadKey        x86_64              2.38-9.amzn2023.0.2      amazor
 perl-lib                x86_64              0.65-477.amzn2023.0.6    amazor

Transaction Summary
================================================================================
Install  7 Packages

Total download size: 7.1 M
Installed size: 34 M
Downloading Packages:
(1/7): git-2.40.1-1.amzn2023.0.3.x86_64.rpm
(2/7): perl-Error-0.17029-5.amzn2023.0.2.noarch.rpm
(3/7): git-core-doc-2.40.1-1.amzn2023.0.3.noarch.rpm
(4/7): perl-Git-2.40.1-1.amzn2023.0.3.noarch.rpm
(5/7): git-core-2.40.1-1.amzn2023.0.3.x86_64.rpm
(6/7): perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64.rpm
(7/7): perl-lib-0.65-477.amzn2023.0.6.x86_64.rpm
--------------------------------------------------------------------------------
Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
```

- git clone

```
[hadoop@ip-172-31-33-127 ~]$ git clone https://github.com/mewawalaabdeali/Wine_Prediction_Distributed_System_Apache_Spark.git
Cloning into 'Wine_Prediction_Distributed_System_Apache_Spark'...
remote: Enumerating objects: 50884, done.
remote: Counting objects: 100% (555/555), done.
remote: Compressing objects: 100% (210/210), done.
remote: Total 50884 (delta 395), reused 500 (delta 340), pack-reused 50329 (from 1)
Receiving objects: 100% (50884/50884), 771.21 MiB | 32.12 MiB/s, done.
Resolving deltas: 100% (11249/11249), done.
Updating files: 100% (46160/46160), done.
[hadoop@ip-172-31-33-127 ~]$ LS
-bash: LS: command not found
[hadoop@ip-172-31-33-127 ~]$ ls
Wine_Prediction_Distributed_System_Apache_Spark
[hadoop@ip-172-31-33-127 ~]$ cd Wine_Prediction_Distributed_System_Apache_Spark/
[hadoop@ip-172-31-33-127 Wine_Prediction_Distributed_System_Apache_Spark]$ ls
Dockerfile   Kubernetes_manifest  bootstrap_dependencies.sh  data       logs    notebook          src  venv
Jenkinsfile  README.md            config.py                  image.png  models  requirements.txt  tmp
[hadoop@ip-172-31-33-127 Wine_Prediction_Distributed_System_Apache_Spark]$
```

- Use spark-submit to execute the training script. The stable file/script name is provided in the first page of the document.

```
[hadoop@ip-172-31-33-127 ~]$ spark-submit Wine_Prediction_Distributed_System_Apache_Spark/notebook/wine_training_cluster_canvas.py --deploy-mode client
```

- Parameters to consider.

Specification	Requirement	Commands and Details
Training Dataset Path	An S3 path where the training dataset (TrainingDataset.csv) is stored.	s3://<bucket-name>/TrainingDataset.csv
Validation Dataset Path	Path required for prediction scripts (not used in this training file).	s3://<bucket-name>/ValidationDataset.csv
Model Save Path	An S3 folder path where trained models will be saved. Dynamically appends subfolders.	s3://<bucket-name>/Testing_models/
Bucket Name Variable	Update the bucket_name variable in the script to match your bucket.	Replace winepredictionabdealicanvas with your bucket name.
S3 Bucket Region	Ensure S3 bucket and EMR cluster are in the same AWS region.	us-east-1
Required Dependencies	Install dependencies on the cluster or machine.	pip install pyspark pandas numpy findspark pyarrow matplotlib boto3
Spark Submit Command	Use appropriate parameters to pass training and save paths.	spark-submit Wine_Prediction_Distributed_System_Apache_Spark/notebook/<stable_script_name> --deploy-mode client

- Once the training is complete, verify the model files are saved in the expected output location (S3 bucket or local directory).
  When the Script runs successfully, it outputs the following things, in some cases, it doesn't print the model save path (debugging yet to done on that), in which case, kindly check your bucket to for the model.

```
24/12/11 15:38:42 INFO FileFormatWriter: Write Job f3005dc5-e227-4aee-8a07-52ccb999bd70 committed. El
24/12/11 15:38:42 INFO FileFormatWriter: Finished processing stats for write job f3005dc5-e227-4aee-8
24/12/11 15:38:42 INFO SQLExecution: Generating and posting SparkListenerSQLExecutionObfuscatedInfo.
24/12/11 15:38:42 INFO SQLExecution: Posted SparkListenerSQLExecutionObfuscatedInfo in 0 ms
Best model saved to: s3a://winepredictionabdealicanvas/winemodels/RandomForestModel_20241211153832
MODEL_NAME=RandomForestModel_20241211153832
24/12/11 15:38:42 INFO FileSourceStrategy: Pushed Filters:
24/12/11 15:38:42 INFO FileSourceStrategy: Post-Scan Filters:
24/12/11 15:38:42 INFO CodeGenerator: Code generated in 26.587191 ms
24/12/11 15:38:42 INFO MemoryStore: Block broadcast_782 stored as values in memory (estimated size 29
24/12/11 15:38:42 INFO MemoryStore: Block broadcast_782_piece0 stored as bytes in memory (estimated s
24/12/11 15:38:42 INFO BlockManagerInfo: Added broadcast_782_piece0 in memory on ip-172-31-33-127.ec2
24/12/11 15:38:42 INFO SparkContext: Created broadcast 782 from rdd at MulticlassClassificationEvalua
24/12/11 15:38:42 INFO FileSourceScanExec: Planning scan with bin packing, max size: 4194304 bytes, o
r of split files: 1, prefetch: false
24/12/11 15:38:42 INFO FileSourceScanExec: relation: None, fileSplitsInPartitionHistogram: Vector((1
24/12/11 15:38:42 INFO SparkContext: Starting job: collectAsMap at MulticlassMetrics.scala:61
24/12/11 15:38:42 INFO DAGScheduler: Registering RDD 1791 (map at MulticlassMetrics.scala:52) as inpu
24/12/11 15:38:42 INFO DAGScheduler: Got job 348 (collectAsMap at MulticlassMetrics.scala:61) with 1
24/12/11 15:38:42 INFO DAGScheduler: Final stage: ResultStage 598 (collectAsMap at MulticlassMetrics.
24/12/11 15:38:42 INFO DAGScheduler: Parents of final stage: List(ShuffleMapStage 597)
24/12/11 15:38:42 INFO DAGScheduler: Missing parents: List(ShuffleMapStage 597)
24/12/11 15:38:42 INFO DAGScheduler: Submitting ShuffleMapStage 597 (MapPartitionsRDD[1791] at map at
nts
```

```
**********************************************************************
++++++++++++++++++++++ Model Evaluation Metrics ++++++++++++++++++++++
**********************************************************************
[Train] F1 Score      : 0.9704
[Train] Accuracy      : 0.9707
----------------------------------------------------------------------
[Test]  F1 Score      : 0.7301
[Test]  Accuracy      : 0.7432
**********************************************************************
24/12/11 15:38:43 INFO SparkContext: SparkContext is stopping with exitCode 0.
24/12/11 15:38:43 INFO SparkUI: Stopped Spark web UI at http://ip-172-31-33-127.ec2.internal:4040
24/12/11 15:38:43 INFO YarnClientSchedulerBackend: Interrupting monitor thread
24/12/11 15:38:43 INFO YarnClientSchedulerBackend: Shutting down all executors
24/12/11 15:38:43 INFO YarnSchedulerBackend$YarnDriverEndpoint: Asking each executor to shut down
24/12/11 15:38:43 INFO YarnClientSchedulerBackend: YARN client scheduler backend Stopped
24/12/11 15:38:44 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/12/11 15:38:44 INFO MemoryStore: MemoryStore cleared
24/12/11 15:38:44 INFO BlockManager: BlockManager stopped
24/12/11 15:38:44 INFO BlockManagerMaster: BlockManagerMaster stopped
```

2. Running Predictions

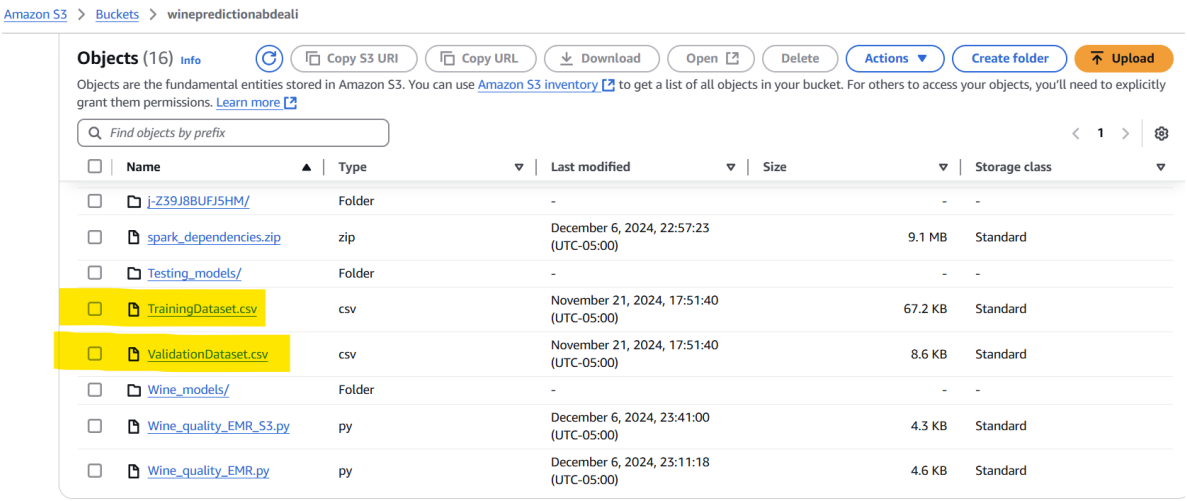
- Parameters to be taken into account

Specification	Requirement	Commands and Details
Validation Dataset Path	Path for the validation dataset (can be local or S3).	s3://<bucket-name>/ValidationDataset.csv or /local/path/to/ValidationDataset.csv
Model Folder Name	Name of the folder where the trained model is stored.	PipelineModel_Grid_YYYYMMDDHHMMSS
S3 Bucket for Models	S3 bucket containing the saved models.	Update bucket_name variable in the script to match the desired bucket.
Local Model Directory	Directory path to save/download models locally if not already present.	/home/hadoop/Wine_Prediction_Distributed_System_Apache_Spark/models/<model_folder_name>
Local Validation Path	Location to download the validation file if the provided path is an S3 path.	/home/hadoop/Wine_Prediction_Distributed_System_Apache_Spark/data/ValidationDataset.csv
Predictions S3 Key	Path where prediction results will be uploaded to S3.	Wine_models/<model_folder_name>_Predictions.csv
Dependencies	Required Python libraries installed on the machine running the script.	pyspark, boto3, pandas, numpy, etc.
S3 Bucket Region	Ensure the bucket is in the same AWS region as the machine or service accessing it.	us-east-1 (example region)
Command-Line Arguments	Input arguments to run the script.	<validation_file_path_or_s3> and <model_folder_name>

- Setup the Machine as given instructions above
- Login into the Machine via SSH

```
C:\Users\asus>ssh -i Downloads\MLKeys\ML_testing.pem ubuntu@3.234.224.0
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1019-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Wed Dec 11 03:58:16 UTC 2024

  System load:  0.0                Temperature:           -273.1 C
  Usage of /:   36.6% of 28.02GB   Processes:             110
  Memory usage: 3%                 Users logged in:       0
  Swap usage:   0%                 IPv4 address for ens5: 172.31.14.243

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed Dec 11 03:40:29 2024 from 96.234.17.150
ubuntu@ip-172-31-14-243:~$
```

- Ensure the validation dataset is available (S3 or local based on environment).



- Clone the git repo and check the content. Use spark-submit to execute the prediction script with the validation dataset and the model path as inputs.

```
ubuntu@ip-172-31-14-243:~$ ls
Wine_Prediction_Distributed_System_Apache_Spark  outputstandalone2.log  outputstandalone7.log        outputstandalonelatest3.log  spark_output_draft2.log
outputnew.log                                    outputstandalone3.log  outputstandalone8.log        outputstandalonelatest4.log
outputstandalone1.log                            outputstandalone4.log  outputstandalone9.log        outputstandalonelatest5.log
outputstandalone10.log                           outputstandalone5.log  outputstandalonelatest1.log  spark-3.4.1-bin-hadoop3.tgz
outputstandalone11.log                           outputstandalone6.log  outputstandalonelatest2.log  spark_output.log
ubuntu@ip-172-31-14-243:~$
```

- Run the Spark Command:

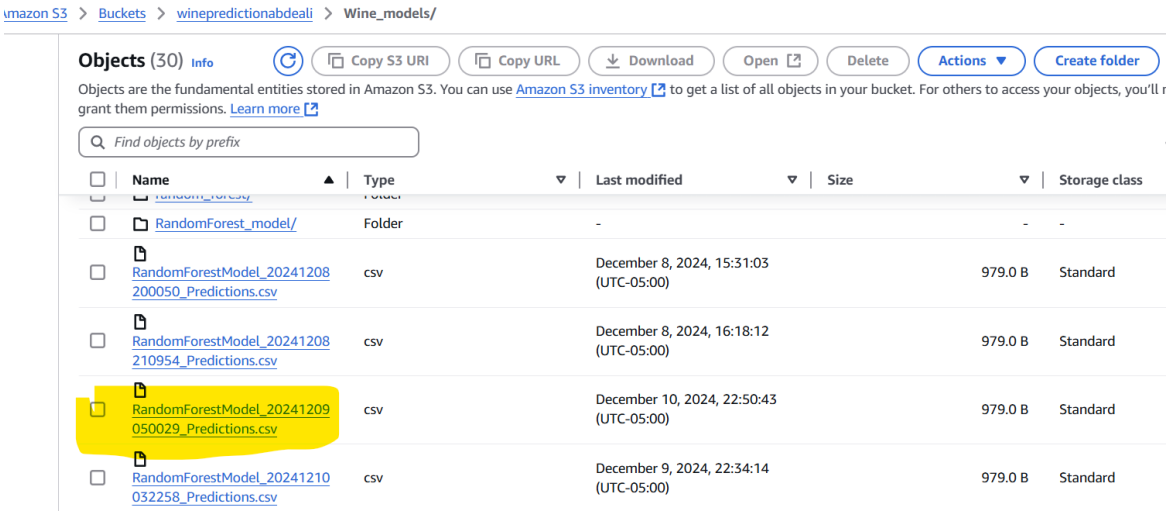
```
ubuntu@ip-172-31-14-243:~$ spark-submit --master local[*] --deploy-mode client Wine_Prediction_Distributed_System_Apache_Spark/notebook/prediction_xgb.py s3
://winepredictionabdeali/ValidationDataset.csv RandomForestModel_20241209050029
```

```
24/12/11 03:50:11 INFO ResourceUtils: ==============================================================
24/12/11 03:50:11 INFO ResourceUtils: No custom resources configured for spark.driver.
24/12/11 03:50:11 INFO ResourceUtils: ==============================================================
24/12/11 03:50:11 INFO SparkContext: Submitted application: Wine_Quality_Prediction
24/12/11 03:50:11 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores ->
 -> name: memory, amount: 1024, script: , vendor: , offHeap -> name: offHeap, amount: 0, script: , vendor
: 1.0)
24/12/11 03:50:11 INFO ResourceProfile: Limiting resource is cpu
24/12/11 03:50:11 INFO ResourceProfileManager: Added ResourceProfile id: 0
24/12/11 03:50:12 INFO SecurityManager: Changing view acls to: ubuntu
24/12/11 03:50:12 INFO SecurityManager: Changing modify acls to: ubuntu
24/12/11 03:50:12 INFO SecurityManager: Changing view acls groups to:
24/12/11 03:50:12 INFO SecurityManager: Changing modify acls groups to:
24/12/11 03:50:12 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users
rmissions: EMPTY; users with modify permissions: ubuntu; groups with modify permissions: EMPTY
24/12/11 03:50:12 INFO Utils: Successfully started service 'sparkDriver' on port 39779.
24/12/11 03:50:12 INFO SparkEnv: Registering MapOutputTracker
24/12/11 03:50:12 INFO SparkEnv: Registering BlockManagerMaster
24/12/11 03:50:12 INFO BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper f
24/12/11 03:50:12 INFO BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
24/12/11 03:50:12 INFO SparkEnv: Registering BlockManagerMasterHeartbeat
24/12/11 03:50:12 INFO DiskBlockManager: Created local directory at /tmp/blockmgr-fe6af566-d77e-40e8-bfa0
24/12/11 03:50:12 INFO MemoryStore: MemoryStore started with capacity 434.4 MiB
24/12/11 03:50:12 INFO SparkEnv: Registering OutputCommitCoordinator
24/12/11 03:50:13 INFO JettyUtils: Start Jetty 0.0.0.0:4040 for SparkUI
24/12/11 03:50:13 INFO Utils: Successfully started service 'SparkUI' on port 4040.
24/12/11 03:50:13 INFO Executor: Starting executor ID driver on host ip-172-31-14-243.ec2.internal
24/12/11 03:50:13 INFO Executor: Starting executor with user classpath (userClassPathFirst = false): ''
```

```
Prediction Evaluation Metrics:
Accuracy: 0.5250
Precision: 0.5059
Recall: 0.5250
F1 Score: 0.5135
24/12/11 03:50:41 INFO FileSourceStrategy: Pushed Filters:
24/12/11 03:50:41 INFO FileSourceStrategy: Post-Scan Filters:
24/12/11 03:50:41 INFO CodeGenerator: Code generated in 85.302296 ms
24/12/11 03:50:41 INFO MemoryStore: Block broadcast_58 stored as values in memory
24/12/11 03:50:41 INFO MemoryStore: Block broadcast_58_piece0 stored as bytes in m
24/12/11 03:50:41 INFO BlockManagerInfo: Added broadcast_58_piece0 in memory on ip
24/12/11 03:50:41 INFO SparkContext: Created broadcast 58 from toPandas at /home/u
xgb.py:102
24/12/11 03:50:41 INFO FileSourceScanExec: Planning scan with bin packing, max siz
24/12/11 03:50:41 INFO BlockManagerInfo: Removed broadcast_55_piece0 on ip-172-31-
24/12/11 03:50:41 INFO BlockManagerInfo: Removed broadcast_57_piece0 on ip-172-31-
24/12/11 03:50:41 INFO BlockManagerInfo: Removed broadcast_56_piece0 on ip-172-31-
24/12/11 03:50:42 INFO SparkContext: Starting job: toPandas at /home/ubuntu/Wine_P
24/12/11 03:50:42 INFO DAGScheduler: Got job 27 (toPandas at /home/ubuntu/Wine_Pre
th 1 output partitions
24/12/11 03:50:42 INFO DAGScheduler: Final stage: ResultStage 34 (toPandas at /hom
on_xgb.py:102)
24/12/11 03:50:42 INFO DAGScheduler: Parents of final stage: List()
24/12/11 03:50:42 INFO DAGScheduler: Missing parents: List()
```

- Validate the prediction results, which will be saved to the specified location (S3 or local).

```
Predictions uploaded to S3: s3://winepredictionabdeali/Wine_models/RandomForestModel_20241209050029_Predictions.csv
24/12/11 03:50:42 INFO SparkContext: SparkContext is stopping with exitCode 0.
24/12/11 03:50:42 INFO SparkUI: Stopped Spark web UI at http://ip-172-31-14-243.ec2.internal:4040
24/12/11 03:50:42 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/12/11 03:50:42 INFO MemoryStore: MemoryStore cleared
24/12/11 03:50:42 INFO BlockManager: BlockManager stopped
24/12/11 03:50:42 INFO BlockManagerMaster: BlockManagerMaster stopped
24/12/11 03:50:42 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
24/12/11 03:50:42 INFO SparkContext: Successfully stopped SparkContext
Spark session stopped.
24/12/11 03:50:43 INFO ShutdownHookManager: Shutdown hook called
24/12/11 03:50:43 INFO ShutdownHookManager: Deleting directory /tmp/spark-2a4bc93e-927f-4199-a7dc-d35d744d09ab
24/12/11 03:50:43 INFO ShutdownHookManager: Deleting directory /tmp/spark-2b6abf05-1408-42f1-9bac-42ce94c4512b/pyspark-16
24/12/11 03:50:43 INFO ShutdownHookManager: Deleting directory /tmp/spark-2b6abf05-1408-42f1-9bac-42ce94c4512b
ubuntu@ip-172-31-14-243:~$
```



3. Dockerizing the Application

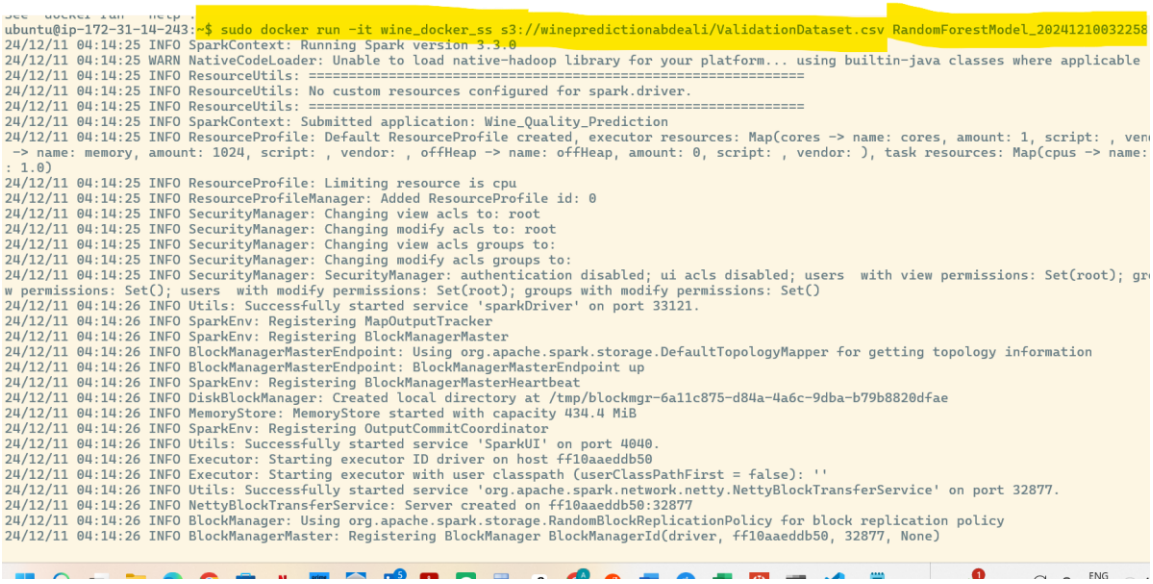
- Build a Docker image for the application using the Dockerfile in the repository.

```
ubuntu@ip-172-31-14-243:~/Wine_Prediction_Distributed_System_Apache_Spark$ ls
Dockerfile          bootstrap_dependencies.sh  notebook                                                    requirements.txt
Jenkinsfile         config.py                  outputstandalonelatestjenkins.log                           spark_output.log
Kubernetes_manifest data                       outputstandalonelatestjenkins_new.log                       src
README.md           image.png                  part-00000                                                  tmp
WinePredictions.csv logs                       part-00000-5fa96e09-2f76-44b6-9228-ba100cfaac17-c000.snappy.parquet  venv
_SUCCESS            models                     part-00000-b9992a39-bda7-4cc8-a74f-aec722a5a974-c000.snappy.parquet
ubuntu@ip-172-31-14-243:~/Wine_Prediction_Distributed_System_Apache_Spark$ sudo docker build -t wine_docker_ss .
[+] Building 439.2s (16/16) FINISHED                                                                                                docker:default
 => [internal] load build definition from Dockerfile                                                                                          0.0s
 => => transferring dockerfile: 1.91kB                                                                                                        0.0s
 => [internal] load metadata for docker.io/library/python:3.8-slim                                                                            0.3s
 => [auth] library/python:pull token for registry-1.docker.io                                                                                 0.0s
 => [internal] load .dockerignore                                                                                                             0.0s
 => => transferring context: 2B                                                                                                               0.0s
 => [ 1/10] FROM docker.io/library/python:3.8-slim@sha256:1d52838af602b4b5a831beb13a0e4d073280665ea7be7f69ce2382f29c5a613f                    0.3s
 => => resolve docker.io/library/python:3.8-slim@sha256:1d52838af602b4b5a831beb13a0e4d073280665ea7be7f69ce2382f29c5a613f                      0.0s
 => => sha256:b5f62925bd0f63f48cc8acd5e87d0c3a07e2f229cd2fb0a9586e68ed17f45ee3 5.25kB / 5.25kB                                                0.0s
 => => sha256:1d52838af602b4b5a831beb13a0e4d073280665ea7be7f69ce2382f29c5a613f 10.41kB / 10.41kB                                              0.0s
 => => sha256:314bc2fb0714b7807bf5699c98f0c73817e579799f2d91567ab7e9510f5601a5 1.75kB / 1.75kB                                                0.0s
 => [internal] load build context                                                                                                             0.7s
 => => transferring context: 8.28MB                                                                                                           0.6s
 => [ 2/10] RUN apt-get update && apt-get install -y     default-jdk wget curl     && apt-get clean                                       81.9s
 => [ 3/10] RUN export JAVA_HOME=$(dirname $(dirname $(readlink -f $(which java)))) &&     echo "JAVA_HOME=$JAVA_HOME" >> /etc/environment &&     ech  0.2s
 => [ 4/10] RUN wget https://archive.apache.org/dist/spark/spark-3.3.0/spark-3.3.0-bin-hadoop3.tgz     && tar -xvzf spark-3.3.0-bin-hadoop3.tgz -C   283.3s
 => [ 5/10] WORKDIR /Wine_Prediction_Distributed_System_Apache_Spark                                                                          0.0s
 => [ 6/10] RUN mkdir -p /Wine_Prediction_Distributed_System_Apache_Spark/notebook             /Wine_Prediction_Distributed_System_Apache_Spark/mode  0.3s
 => [ 7/10] COPY requirements.txt /Wine_Prediction_Distributed_System_Apache_Spark/                                                           0.1s
 => [ 8/10] RUN pip install --no-cache-dir -r requirements.txt                                                                               63.9s
 => [ 9/10] COPY notebook/prediction_xgb_docker.py /Wine_Prediction_Distributed_System_Apache_Spark/notebook/                                 0.1s
 => [10/10] COPY models /Wine_Prediction_Distributed_System_Apache_Spark/models/                                                              0.2s
 => exporting to image                                                                                                                        8.4s
 => => exporting layers                                                                                                                       8.4s
 => => writing image sha256:1a943bc58c6904d1cc9ed0a7e3afed400a55e3e5d73bcc185525b4cf55177c3d                                                  0.0s
 => => naming to docker.io/library/wine_docker_ss                                                                                             0.0s
ubuntu@ip-172-31-14-243:~/Wine_Prediction_Distributed_System_Apache_Spark$
```

- Tag and push the image to Dockerhub

```
 => => naming to docker.io/library/wine_docker_ss
ubuntu@ip-172-31-14-243:~/Wine_Prediction_Distributed_System_Apache_Spark$ cd
ubuntu@ip-172-31-14-243:~$ sudo docker tag wine_docker_ss aliabde24/wine_ss:1012
ubuntu@ip-172-31-14-243:~$ sudo docker push aliabde24/wine_ss:1012
The push refers to repository [docker.io/aliabde24/wine_ss]
f80cf81edccb: Pushed
cdfdd427bdd6: Pushed
eee00cb6b890: Pushed
4c8b3446adc2: Pushed
12feed858f01: Pushed
f5073cf72af9: Pushed
f7be63e4c9b4: Pushed
801617a33e68: Pushed
1f552b1ee8bb: Pushed
d2a2207b52a4: Mounted from aliabde24/wine_xgb
5d2d143f3d7f: Mounted from aliabde24/wine_xgb
c3772b569c3a: Mounted from aliabde24/wine_xgb
8d853c8add5d: Mounted from aliabde24/wine_xgb
1012: digest: sha256:1e3452681bb2e22454ecc619fc6648525c6f2a53e3b38f2745f817eb829ce427 size: 3045
ubuntu@ip-172-31-14-243:~$
```

- Run the Docker container to execute predictions using the validation dataset and the pre-trained model.



- Verify the outputs generated by the Docker container.

```
Prediction Evaluation Metrics:
Accuracy: 0.5250
Precision: 0.5059
Recall: 0.5250
F1 Score: 0.5135
24/12/11 04:14:49 INFO FileSourceStrategy: Pushed Filters:
24/12/11 04:14:49 INFO FileSourceStrategy: Post-Scan Filters:
24/12/11 04:14:49 INFO FileSourceStrategy: Output Data Schema: struct
 double, """residual sugar""": double, """chlorides""": double ..
24/12/11 04:14:49 INFO CodeGenerator: Code generated in 42.105601 ms
24/12/11 04:14:49 INFO MemoryStore: Block broadcast_58 stored as valu
24/12/11 04:14:49 INFO BlockManagerInfo: Removed broadcast_53_piece0
24/12/11 04:14:49 INFO BlockManagerInfo: Removed broadcast_56_piece0
24/12/11 04:14:49 INFO MemoryStore: Block broadcast_58_piece0 stored
24/12/11 04:14:49 INFO BlockManagerInfo: Added broadcast_58_piece0 ir
24/12/11 04:14:49 INFO SparkContext: Created broadcast 58 from toPanc
```

```
24/12/11 04:14:49 INFO DAGScheduler: Job 27 is finished. Cancelling potential speculative or zombie tasks for this j
24/12/11 04:14:49 INFO TaskSchedulerImpl: Killing all running tasks in stage 34: Stage finished
24/12/11 04:14:49 INFO DAGScheduler: Job 27 finished: toPandas at /Wine_Prediction_Distributed_System_Apache_Spark/n
ok 0.184921 s
Predictions uploaded to S3: s3://winepredictionabdeali/Wine_models/RandomForestModel_20241210032258_Predictions.csv
24/12/11 04:14:49 INFO SparkUI: Stopped Spark web UI at http://ff10aaeddb50:4040
24/12/11 04:14:49 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/12/11 04:14:49 INFO MemoryStore: MemoryStore cleared
24/12/11 04:14:49 INFO BlockManager: BlockManager stopped
24/12/11 04:14:49 INFO BlockManagerMaster: BlockManagerMaster stopped
24/12/11 04:14:49 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
24/12/11 04:14:49 INFO SparkContext: Successfully stopped SparkContext
Spark session stopped.
24/12/11 04:14:50 INFO ShutdownHookManager: Shutdown hook called
24/12/11 04:14:50 INFO ShutdownHookManager: Deleting directory /tmp/spark-e42a9d8f-aa01-4b01-ab2b-9eea004cbd09
24/12/11 04:14:50 INFO ShutdownHookManager: Deleting directory /tmp/spark-d63e2a9d-8bae-412d-aab6-bca4c5455687/pyspa
24/12/11 04:14:50 INFO ShutdownHookManager: Deleting directory /tmp/spark-d63e2a9d-8bae-412d-aab6-bca4c5455687
ubuntu@ip-172-31-14-243:~$
```

AI ChatGPT/Copilot Help:

Issue/Problem	Details	Resolution Provided by AI
Dependency Conflicts	Python dependencies like `awscli` and `numpy` caused versioning conflicts on the EMR cluster.	Upgraded and reinstalled specific versions using `pip install --upgrade --force-reinstall`. Resolved conflicts with `python-dateutil==2.8.2`.
Data Type Issues	Spark threw errors for unsupported string data types during transformations.	Ensured all columns were cast to appropriate data types using `.cast("double")` for numerical columns and `.cast("integer")` for labels.
Model Save Issues	Models were not being saved to S3 properly or overwriting existing models.	Implemented unique naming logic using a timestamp and parameter hash. Used `.overwrite()` for explicit overwriting of directories.
Cluster Mode Setup Issues	Spark jobs failed due to missing dependencies or cluster misconfigurations.	Installed required dependencies via `yum` and `pip`. Configured Spark sessions with necessary libraries and S3 settings yet the Manual cluster setup never worked
S3 Access Denied	EMR instances failed to write models to the S3 bucket due to IAM permission issues.	Updated bucket policy and IAM role to allow write access explicitly for the EMR cluster's account ID.

File Already Exists Error	Spark jobs failed due to attempts to overwrite existing HDFS directories during save operations.	Used `model.write().overwrite()` logic and ensured unique paths for each run using timestamps.
Data Not Distributed Properly	Data was not distributed as expected across the cluster for training.	Verified `spark-submit` configuration and ensured proper cluster utilization via executor logs.
Prediction File Issues	Errors in prediction logic due to improper model loading or validation file handling.	Corrected file handling for both local and S3 datasets. Ensured models downloaded and loaded correctly.
Metrics Not Displaying	Evaluation metrics like F1 score and accuracy were not printed in logs.	Enhanced output to display metrics clearly for training and testing data Yet in some runs, the metrics is not printed miraculously. Need to debug.
Model Directory Structure Issues	Model directories were inconsistently structured during save and upload to S3.	Adjusted logic to save models in a uniform structure and ensured all files were uploaded recursively.
Hadoop FS Error in Manual Cluster Mode	Hadoop errors occurred when saving metadata to HDFS in cluster mode.	Verified HDFS permissions, ensured unique output directories, and avoided conflicts by using .overwrite() or unique folder names Yet Manual cluster never worked.
Standalone vs. Cluster Mode Differences	Jobs succeeded on standalone but failed in cluster mode.	Cluster Mode need more debugging
Bucket Policy Inconsistencies	S3 operations worked on standalone but failed on the cluster due to policy differences.	Updated bucket policies and ensured proper IAM role attachment for both standalone and cluster environments.
Data Cleaning Issues	Errors occurred due to inconsistent column names in the training dataset.	Standardized column names by removing unnecessary quotes and spaces. Applied schema inference and strict type casting during data ingestion.
Validation Dataset Path Issues	Validation dataset not loading due to wrong file path or directory issues.	Ensured directory structures matched expected paths. Implemented dynamic handling for validation file paths from S3.
Spark Session Initialization	Spark session failed to initialize correctly in cluster mode.	Adjusted Spark session configuration for distributed environments.
Oversampling Logic Implementation	Model underperformed due to class imbalance in the training dataset.	Implemented oversampling to balance classes and improve model generalization.

Experience with AI:

- Mostly useful in error debugging.
- Also helped in building modular code from the notebook code.

To be added Later:

4. Automating with Jenkins Pipeline

- **Jenkinsfile, Kubernetes deployment.yml** can be found in the project directory.
- Configure a Jenkins job to automate training and prediction workflows.
- Add pipeline parameters like validation file path, model folder name, and other necessary inputs.

- Trigger the Jenkins pipeline and monitor its execution.
- Verify the outputs, including logs and artifacts generated by the pipeline.

5. Deploying with Kubernetes (Optional)

- Configure Kubernetes deployment using a deployment.yml file.
- Apply the configuration to deploy the application.
- Expose the deployment as a service for external access.
- Use the service to run predictions and validate outputs.
- Monitor the deployment and ensure scalability for distributed execution.