

# Wine Quality Prediction on AWS

## Introduction

The goal of this project is to develop a machine learning (ML) model for predicting wine quality using Apache Spark on AWS EMR. The assignment involves parallel training on a Spark cluster managed by EMR, single-node predictions, Docker containerization for deployment, and sharing the code repository on GitHub for collaboration and evaluation.

Link to GitHub: [Wine\\_Quality\\_Prediction\\_inAWS](#)

Link to Docker: [wine-quality-prediction](#)

## Tools Used

1. AWS EMR: Managed Hadoop framework to set up and run Apache Spark clusters for distributed computing.
2. Apache Spark: Framework for distributed data processing and machine learning with MLlib. Used for model training and prediction tasks.
3. Python as Programming language for implementing training and prediction code
4. Sparks MLlib: Machine learning library for classification, regression, and evaluation metrics.
5. Library for implementing machine learning models like RandomForestClassifier.
6. Docker: Containerization platform to package and deploy the prediction application.
7. AWS S3: Cloud storage for datasets and the trained model pipeline.
8. GitHub: Repository for storing and sharing project code, scripts, and documentation.

## Infrastructure Setup

### 1. Create s3 Bucket

Create an S3 bucket in our aws cloud to store training/predicting script, dataset and our model after getting trained.

- S3 bucket as      aws: pa2winequalitybucket

### 2. Create Key-pair for our EMR cluster

- EC2 > Network > Key-pairs
- Create key as: - pa2\_pruthvidholkia\_WineQuality.pem

### 3. Now let's create an EMR cluster as per project requirement for parallel job

- Name: **my\_cluster\_wineQuality\_predict\_18**
- Create Cluster > Application bundle custom (Hadoop, spark) > OS (Amazon Linux)
- Cluster Config > Core and Task(m5.xlarge) > Cluster scaling and provisioning (set task to 3)

- Networking > subnet > browse and select different region if cluster gets terminated because of instance type (m5).
- Security Config > Amazon EC2 keypair for ssh > pa2\_pruthvidholkia\_WineQuality
- IAM Role > EMR\_DefaultRole
- EC2 Instance > EMR\_DefaultRole

**Name and applications - required** [Info](#)  
Name your cluster and choose the applications that you want to install to your cluster.

Name

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

Application bundle

Spark Interactive	Core Hadoop	Flink	HBase	Presto	Trino	Custom
-------------------	-------------	-------	-------	--------	-------	--------

☐ AmazonCloudWatchAgent 1.300032.2  
☐ HCatalog 3.1.3  
☐ Hue 4.11.0  
☒ Livy 0.8.0  
☐ Pig 0.17.0  
☐ TensorFlow 2.16.1  
☒ Zeppelin 0.11.1

☐ Flink 1.19.1  
☒ Hadoop 3.4.0  
☐ JupyterEnterpriseGateway 2.6.0  
☐ Oozie 5.2.1  
☐ Presto 0.287  
☐ Tez 0.10.2  
☐ ZooKeeper 3.9.2

☐ HBase 2.5.10  
☐ Hive 3.1.3  
☐ JupyterHub 1.5.0  
☐ Phoenix 5.2.0  
☒ Spark 3.5.2  
☐ Trino 446

Figure 1

**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: -  
Lowest Spot price: - [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: -  
Lowest Spot price: - [Actions](#)

**Node configuration - optional**

**Task 1 of 1** [Remove instance group](#)

Name

Choose EC2 instance type  
m5.xlarge  
4 vCore 16 GiB memory  
EBS only storage On-Demand price: -  
Lowest Spot price: - [Actions](#)

Figure 2

**Cluster scaling and provisioning - required** [Info](#)  
Choose how Amazon EMR should size 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	<input type="text" value="3"/>	<input checked="" type="checkbox"/>
Core	m5.xlarge	<input type="text" value="1"/>	<input checked="" type="checkbox"/>

**Networking - required** [Info](#)  
Choose the network settings that determine how you and other entities communicate with your cluster.

**Virtual private cloud (VPC)** [Info](#)  
 [Browse](#) [Create VPC](#)

**Subnet** [Info](#)  
 [Browse](#) [Create subnet](#)

► **EC2 security groups (firewall)**

Figure 3

**Security configuration and EC2 key pair** [Info](#)  
Choose a security configuration or create a new one that you can reuse with other clusters.

**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](#)  
 [Browse](#) [Create key pair](#)

**Identity and Access Management (IAM) roles - required** [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**  
 [Create](#)

**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**  
 [Create](#)

Figure 4

#### 4. Parallel Model Training on EMR and Predicting

After creating cluster upload your training.py and predicting.py and both the datasets to your s3 because now we will run our cluster and train our model parallelly in 4 instances.

- ssh your EMR cluster with PuTTY:

pass your pa2\_pruthvidholkia\_WineQuality.pem key to PuTTY

pass your public DNS in Hostname or IP

[hadoop@ec2-18-205-20-68.compute-1.amazonaws.com](mailto:hadoop@ec2-18-205-20-68.compute-1.amazonaws.com)



## Cluster management

Log destination in Amazon S3  
aws-logs-210071813569-us-east-1/elasticmapreduce

## Persistent application UIs

[Spark History Server](#) 

[YARN timeline server](#) 

### Primary node public DNS

ec2-18-205-20-68.compute-1.amazonaws.com

### Connect to the Primary node using SSH

Figure 5

- Install numpy, imbalanced-learn and pandas in your EMR

```
pip install numpy
pip install imbalanced-learn
pip install pandas
```
- Run this cmd to execute train\_wineQuality.py which performs data preprocessing, training model and saving it in same s3 location:

```
spark-submit s3://pa2winequalitybucket/train_wineQuality.py
```

- After a while your model will get trained and get saved to your s3 bucket which you will pass in train file

```
=====
Model successfully saved to s3a://pa2winequalitybucket/optimized_wine_model_rf.model
Training Weighted Recall: 0.9832869080779945
=====
24/12/07 23:25:16 INFO SparkContext: SparkContext is stopping with exitCode 0.
```

Figure 7

- Pass this cmd in your Putty terminal as you did for training part:  
spark-submit s3://pa2winequalitybucket/predict\_wineQuality.py

Once the predict\_wineQuality runs successfully in your EMR you could see **f1 score** as below:

```
=====
Validation Accuracy: 0.775
Validation F1 Score: 0.7994607420189821
Validation Weighted Recall: 0.775
=====
24/12/07 23:32:25 INFO SparkContext: SparkContext is stopping with exitCode 0.
```

Figure 8

- This prediction will run on single ec2 instance the master node or master ec2 as per project requirement and how we achieve this in predict file you should see this:

```
# Initialize Spark session
spark = SparkSession.builder \
    .appName('PredictWineQuality') \
    .master('local[*]') \
    .getOrCreate()
```

Figure 9

- Since the script explicitly specifies local[\*], even if the EC2 instance is part of a cluster, this job won't utilize other nodes in the cluster.

## 6. Docker Containerization (Running application with Docker)

Objective: Package the prediction application for deployment.

Steps:

- Load all files and train model folder in you EMR.  
aws s3 cp s3://pa2winequalitybucket/train\_wineQuality.py ./train\_wineQuality.py  
aws s3 cp s3://pa2winequalitybucket/predict\_wineQuality.py ./predict\_wineQuality.py  
aws s3 cp s3://pa2winequalitybucket/requirements.txt ./requirements.txt  
aws s3 cp s3://pa2winequalitybucket/TrainingDataset.csv ./TrainingDataset.csv  
aws s3 cp s3://pa2winequalitybucket/ValidationDataset.csv ./ValidationDataset.csv  
aws s3 cp s3://pa2winequalitybucket/optimized\_wine\_model\_rf.model  
./optimized\_wine\_model\_rf.model -recursive
- Created a Dockerfile with the following components:
  - Spark setup.
  - Python dependencies.
  - Train and prediction script (train\_wineQuality.py, predict\_wineQuality.py).
- Create the image called wine-quality-prediction  
sudo docker build -t wine-quality-prediction .
- Login to docker with sudo login docker
- sudo docker tag wine-quality-prediction pruthvidholkia/wine-quality-prediction:latest
- Push: sudo docker push pruthvidholkia/wine-quality-prediction:latest

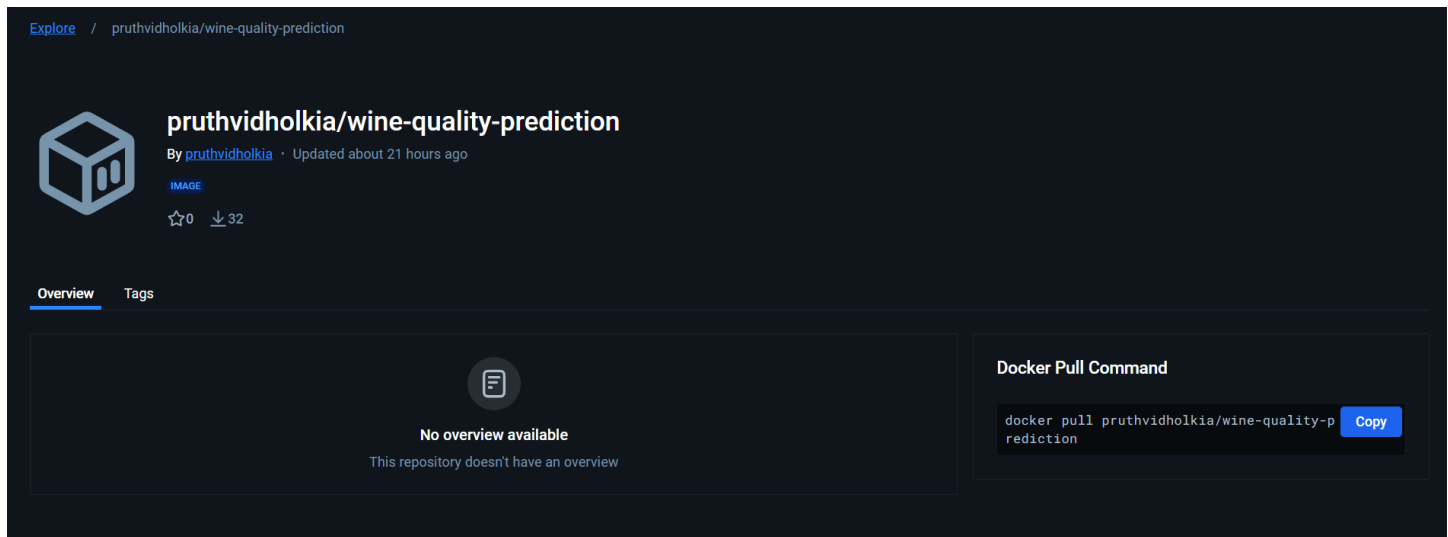


Figure 10