**READ-ME-PROGRAM2**

**Table of Contents:**

1. Setting up AWS . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .1
2. Creating EC2 Instances . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
3. Creating S3 Buckets . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
4. Locally SSH into EC2 Virtual Instances . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5
5. Creating Key Pairs to Locally SSH Into EMR Cluster . . . . . . . . . . . . . . . . . . . . . . . . . . . . .6
6. Creating EMR Clusters . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .7
7. Creating Security Groups for EC2 Instances . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .8
8. Running the Python Wine Quality Prediction Application (Locally on MAC) . . . . . . . 10
9. Running the Python Wine Quality Prediction Application (JupyterHub Browser). . . .10
10. Results and Outputs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .12
11. Code Explanation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .12
12. External Github and Dockerhub Code Links . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .13

**Setting Up AWS Academy**

1. First step in setting up the cloud environment and run the application is to set up and activate your aws account user. In our case the instructions and associated links were provided by the professor. If you are not student, follow the instructions here: <https://docs.aws.amazon.com/accounts/latest/reference/manage-acct-creating.html>
2. Once our AWS learner Lab account is set up we should be able to access the learner page in canvas as shown below.

A screenshot of a website

Description automatically generated

1. To actually access AWS, click the modules tab, then “learner lab”. You see the screen below. If you click start, that will start a session, and the AWS link on the top left will turn green from red. Clicking end lab will end it. The “AWS details” tab shows detail on start and end times and allows use to download key that we need. Click “Download PEM” for Mac.

A screenshot of a computer

Description automatically generated

1. Press start lab, then click the AWS link was it turns green. It take you to that looks like the image below.

A screenshot of a computer

Description automatically generated

**Creating E2 Instances**

1. Once we are on the screen above, we can go ahead and create instances for each app process A and B. Click “Launch a virtual machine” under the “Build a solution” tab. Should look like below.

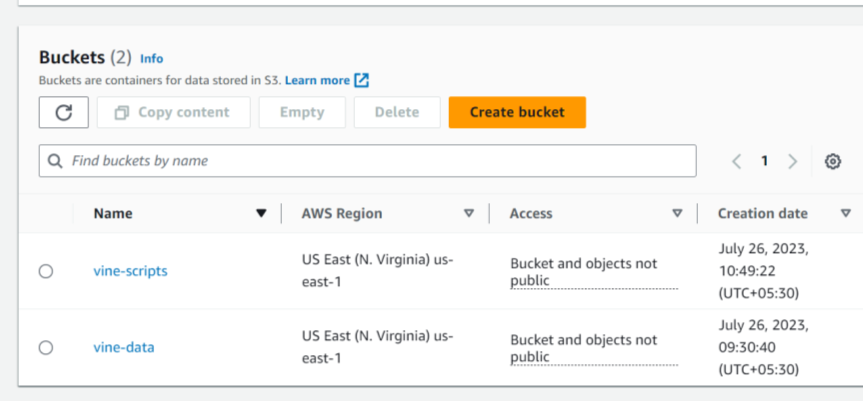
A screenshot of a computer

Description automatically generated

1. Create a name, can be anything. Select Amazon Linux for AMI. Keep Instance type setting the same. Select vockey for key pair. Or you can also create your own key pair, if you do that just make sure you use that same key pair for both instances. For Network Settings, under firewall, check all the rule for allowing traffic to access our server. Also, in the drop down next to “allow SSH traffic from” select “My IP”. Configure storage settings remains the same. Once everything looks good, click the orange “launch instance” button.

**Creating S3 Buckets**

1. Create two S3 buckets.



1. Create VPC called vine-prediction-vpc

A screenshot of a computer

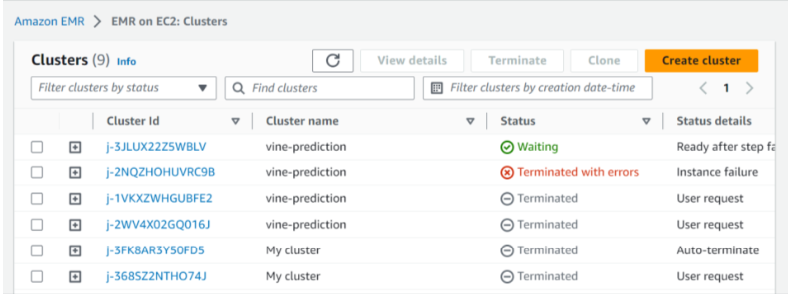
Description automatically generated

1. Create a EC2 key pair called vine\_pred

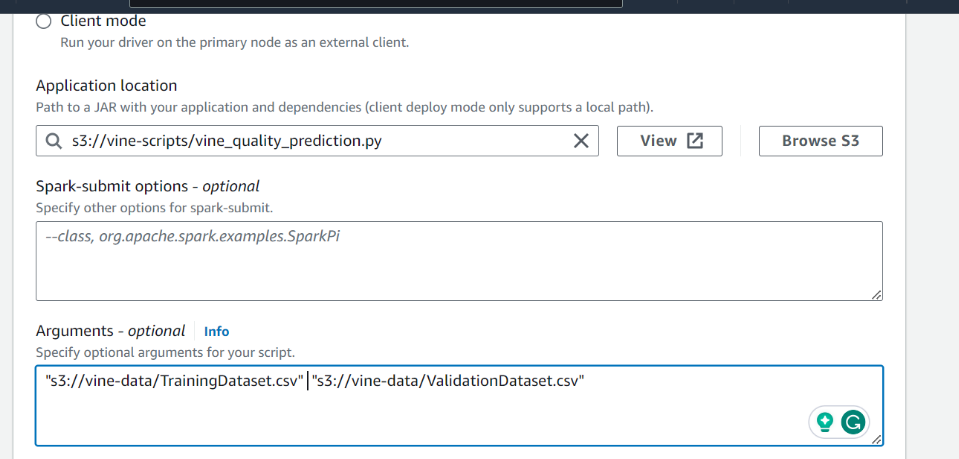
A screenshot of a computer

Description automatically generated

1. Create a EMR cluster called **vine-prediction (**1 Primary,1 Core,2 Task)  
   This cluster will be terminated automatically after some inactive time. If you need to recreate the cluster you can do it by simply cloning the terminated cluster.



1. Submit spark job. Goto the cluster and submit a step. You can pass arguments like below.



**Locally SSH into E2 Virtual Instances Via MAC Terminal**

1. On the left menu, click instances under instances header. Click the blue Instance id of the instance you just created. Should see the instance summary details as shown below.

A screenshot of a computer

Description automatically generated

1. Make note of the Public IPv4 address shown in the middle column at the top and Open terminal on mac. Change the directory to the location where the .pem certificate key was downloaded. In my case, it is the downloads folder.



1. Change the permission on the key to be read only by the running the command “chmond 40 labsuser.pem”.



1. In the terminal window, run the following command “ssh -i <filename>.(pem or cer) ec2-user@<public-ip>”. You will be asked if you are sure you want to connect? Type “yes” and press enter. A computer screen with white text

   Description automatically generated
2. You know you did it right when you see the cool bird image in the terminal. Also your username will change to ec2-user@<ip>.

A computer screen with white text

Description automatically generated

**Creating Key Pairs to Locally SSH Into EMR Cluster**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, under **Network & Security**, choose **Key Pairs**.
3. Choose **Create key pair**.
4. For **Name**, enter a descriptive name for the key pair. Amazon EC2 associates the public key with the name that you specify as the key name. A key name can include up to 255 ASCII characters. It can’t include leading or trailing spaces.
5. For **Key pair type**, choose either **RSA** or **ED25519**.
6. For **Private key file format**, choose the format in which to save the private key. To save the private key in a format that can be used with OpenSSH, choose **pem**. To save the private key in a format that can be used with PuTTY, choose **ppk**.
7. To add a tag to the public key, choose **Add tag**, and enter the key and value for the tag. Repeat for each tag.
8. Choose **Create key pair**.
9. The private key file is automatically downloaded by your browser. The base file name is the name that you specified as the name of your key pair, and the file name extension is determined by the file format that you chose. Save the private key file in a safe place.

**Creating EMR Clusters**

1. Search for EMR is AWS’s main search bar. Click “EMR” , then the “Create Cluster” blue button to create the cluster.

A screenshot of a computer

Description automatically generated

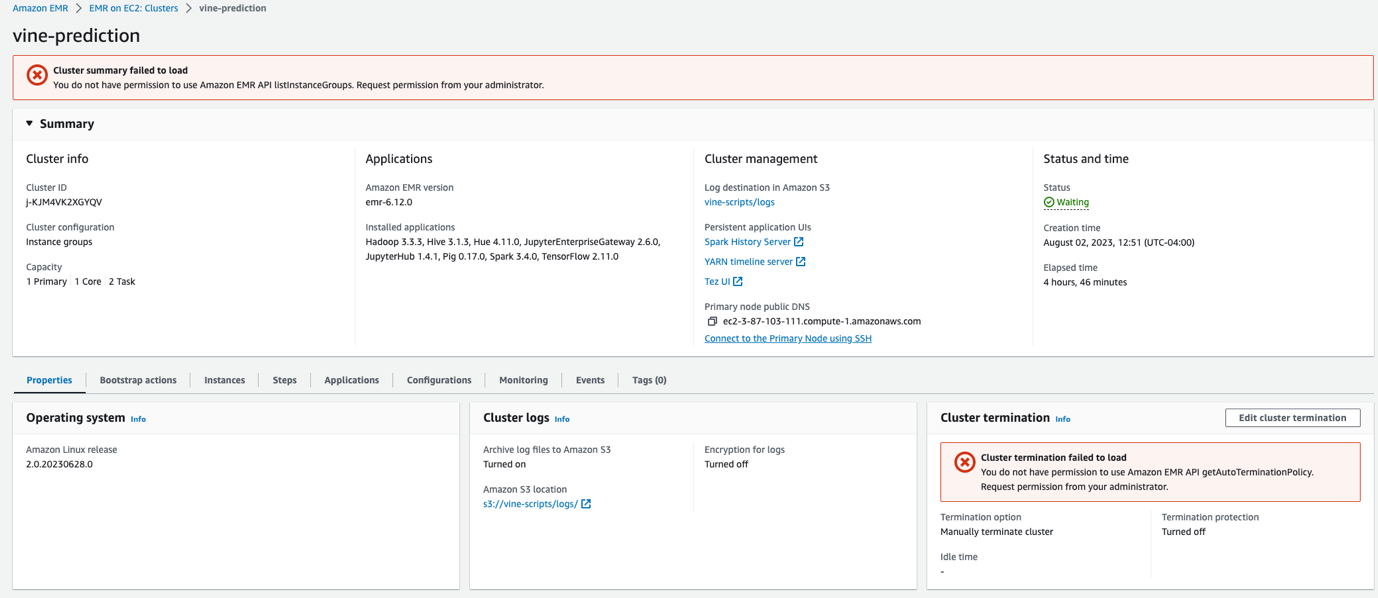
1. Make a name for the cluster and remember it. Configure the software and hardware settings as follows:

A screenshot of a computer

Description automatically generated

1. Then under “Security and Access” Select “Mixture” for **EC2 key pair**. Click “Create Cluster”. It will take 10 to 15 minutes to start, so go make some coffee.

**Creating Security Groups for EC2 Instances**

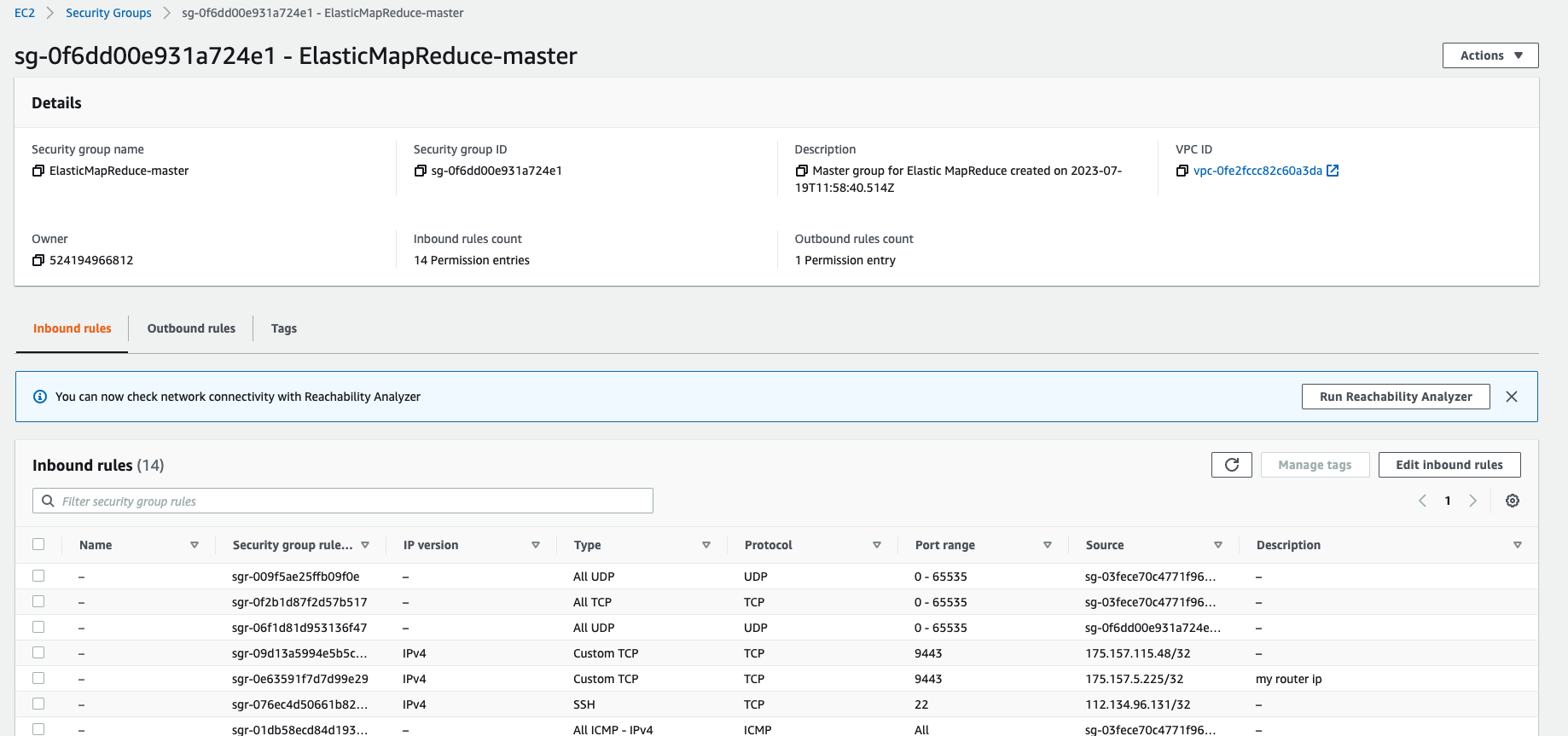
1. Once the EMR cluster Status is running/waiting. Go ahead and open it up in AWS. You will see a summary of the cluster and properties. Copy the “Primary node public DNS” under the “cluster management” column. 
2. After that, open terminal and ping the address you just copied using the ping command.A screenshot of a computer

   Description automatically generated
3. Copy the IP address in the parathesis. A bunch of timed out requested should continuously fill the screen. Open any browser type https://<copied IP>:<Port #>. Replace <copied id> and <Port #> with appropriate values.
4. To get the Port # we will need to create security group. On EMR on EC2 Cluster page, under properties tab, under “Network and Security” click “EC2 security groups (firewall)”

A screenshot of a computer

Description automatically generated

1. In this menu, click the blue link under primary nodeA screenshot of a computer

   Description automatically generated
2. In the menu that opens up, click “edit inbound rules” button, and then click “Add Rule” at the bottom of the list. 
3. You will add a rule with similar properties as the one shown below. 
4. The column next to TCP shows Port Range, that’s our Port # to put in the browser link from Step #3. When everything is running right the following will be seen in the browser.

**Running The Python Wine Quality Prediction Application (Locally on MAC)**

1. Open a terminal window. On Mac OS X, choose Applications > Utilities > Terminal. On other Linux distributions, terminal is typically found at Applications > Accessories > Terminal.
2. To establish a connection to the primary node, enter the following command. Replace **~/vine.pem** with the location and filename of the private key file (.pem) that you used to launch the cluster.

ssh -i ~/vine.pem [hadoop@ec2-3-87-103-111.compute-1.amazonaws.com](mailto:hadoop@ec2-3-87-103-111.compute-1.amazonaws.com)

1. Enter yes to dismiss the security warning.
2. After connecting to the cluster you can submit the job using the following command: spark-submit --deploy-mode cluster s3://vine-scripts/vine\_quality\_prediction.py s3://vine-data/TrainingDataset.csv s3://vine-data/ValidationDataset.csv

**Running The Python Wine Quality Prediction Application (JupyterHub Browser)**

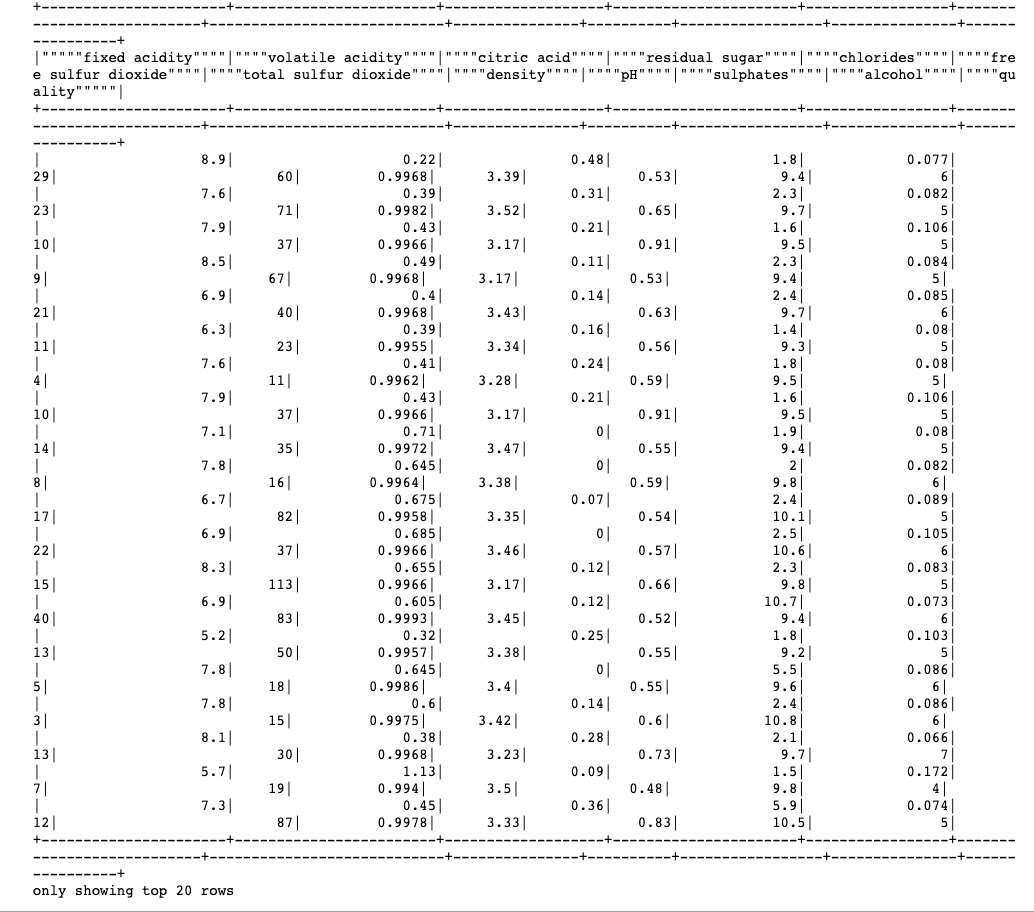
1. Type the link in any browser using info from “Creating Security Groups for EC2 Instances”steps #3 and #8 . When everything is running right, you will see a login screen for our JupyterHub server. Log in using **Username:** jovyan and **Password:** jupyter.
2. Using the “upload” button at the top right, upload the “vine\_quality\_prediction.ipynb” file which is part of the folder this read-me file was in. A screenshot of a computer

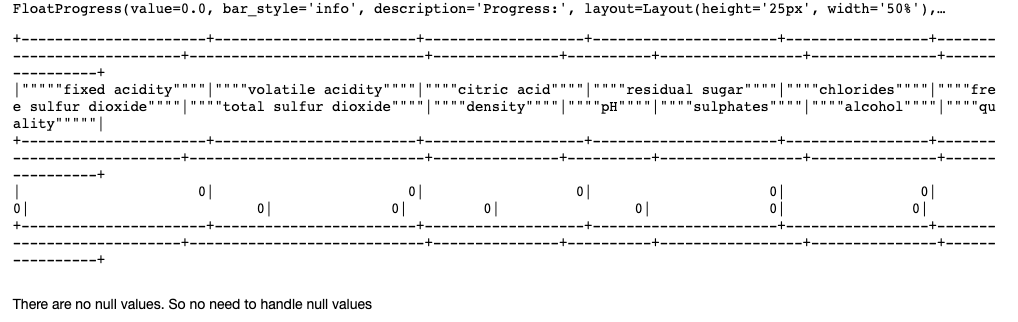
   Description automatically generated
3. Once its uploaded, click it to open, select each of the code blocks and press “run” at the top of the screenA screenshot of a computer

   Description automatically generated

**Results**

1. The figures below shows the nutritional facts and predicted quality

****

****

**Code Explanation**

1. **Data Type Conversion:**

The first part of the code converts all the columns in df\_train and df\_val to the DoubleType() data type. This is done to ensure that all the columns have numerical values, which is a requirement for machine learning algorithms in Spark.

1. **Define Target Variable:**

The code defines the target variable's name as '""""quality"""""'. This target variable is the variable we want to predict using machine learning models.

1. **Training and Testing Sets:**

The code then defines the training and testing datasets. It assigns the df\_train to training\_data and df\_val to testing\_data. These two DataFrames will be used for training and evaluating the machine learning models, respectively.

1. **Feature Columns:**

The code creates a list feature\_columns, which includes all the columns in df\_train except for the last one (last column is the target variable). These are the columns that will be used as features for the machine learning models.

1. **Vector Assembler:**

The VectorAssembler is used to combine the feature columns into a single vector column named "features". This is a required step in Spark MLlib as many algorithms expect the input features to be in a single vector column. The "features" column will be used as the input for the machine learning models.

**External Github and Dockerhub Code Links**