

SAI KARTHIK KASUMURTHY (sk3374@njit.edu)

## Programming Assignment 2

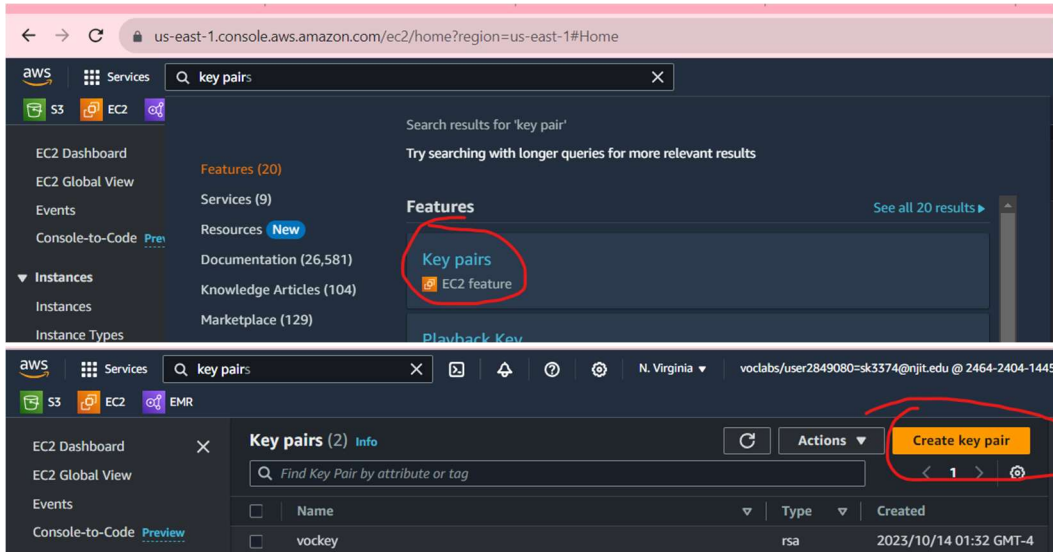
GitHub Link: <https://github.com/karthik984/WinePredictionAnalysis>

Docker Image Link: <https://hub.docker.com/r/karthikkk999/wine-prediction>

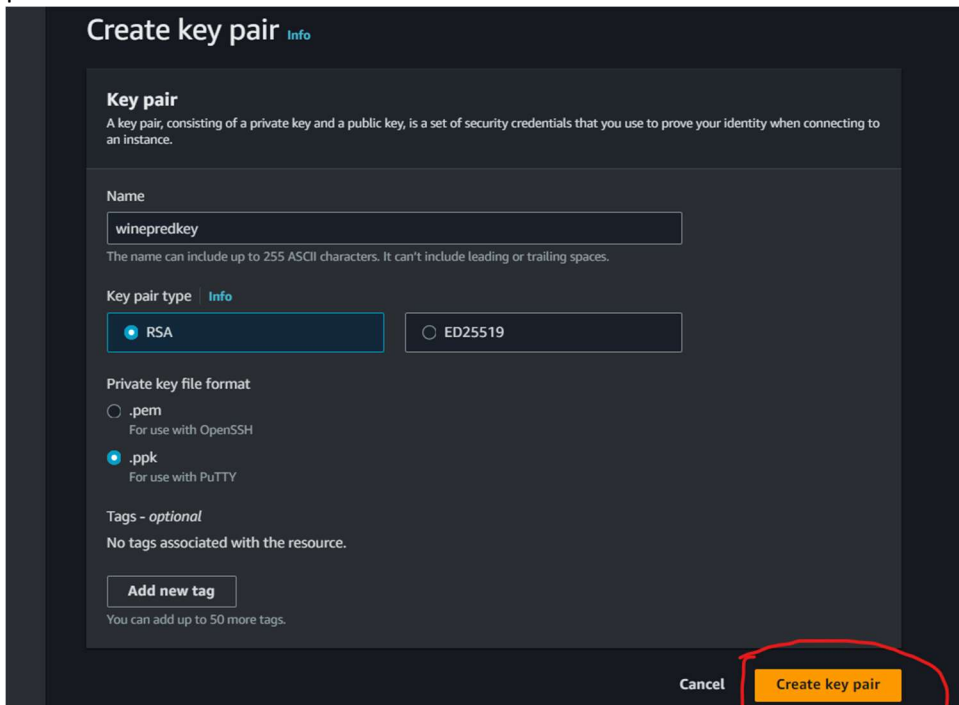
### Step-by-step process on how to set-up the cloud environment, run the model training and the application prediction:

#### 1. Create a Key Pair:

- a. Go to EC2 feature > “Key Pairs” and click on “Create key pair” as



- b. Give it a name and select “.pem” if you have MAC OS or “.ppk” if you have Windows and click on “Create Key pair”



- c. This will also download the “.ppk” or “.pem” file to your local machine. Make sure you save it. You will require it later.
- d. Once Created it should appear in your key pairs list:

Key pairs (2) <a href="#">Info</a>			
Find Key Pair by attribute or tag			
<input type="checkbox"/>	Name	Type	Created
<input type="checkbox"/>	vockey	rsa	2023/10/14 01:32 GMT-4
<input type="checkbox"/>	winepredkey	rsa	2023/12/06 12:51 GMT-5

## 2. Create EMR Cluster:

- a. Go to “Amazon EMR > EMR on EC2: Clusters > Create cluster” and give it a name and select the latest emr version. Also check the Hadoop, Spark , Zeppelin which are required by our code to train the model

[Amazon EMR](#) > [EMR on EC2: Clusters](#) > [Create cluster](#)


### Create cluster [Info](#)


**Name and applications** [Info](#)


Name  
WinePredCluster1


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


Application bundle


Spark  
Interactive  



Core  
Hadoop  


Flink  


HBase  


Presto  


Trino  


Custom  


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

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

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

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

**Operating system options** [Info](#)

☒ Amazon Linux release  
☐ Custom Amazon Machine Image (AMI)

☒ Automatically apply latest Amazon Linux updates

- b. Keep Cluster Configuration as it is and scroll until “Cluster scaling and provisioning”.

- c. In “Cluster scaling and provisioning” select 1 Core Instance and 3 Task – 1 instances. (Total 4)

### Cluster scaling and provisioning Info

Set up scaling and provisioning configurations for the core and task node groups for your cluster.

Choose an option

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

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

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

### Provisioning configuration

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

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

- d. Select the Cluster termination as per your convenience. I had selected 3 hours

### Cluster termination Info

☐ Manually terminate cluster

☐ Automatically terminate cluster after last step ends

☒ **Automatically terminate cluster after idle time (Recommended)**

**Idle time**  
Enter the time until your cluster terminates.

Choose a time that is greater than 1 minute (00:01:00) and less than 7 days. The time is in hh:mm:ss (24-hour) format.

☒ **Use termination protection**  
Protect your EC2 instances from accidental termination.

- e. Keep rest as it is and scroll down to “Security configuration and “EC2 key pair”. Select the key that you have created in Step 1.

### Security configuration and EC2 key pair - optional Info

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

**Amazon EC2 key pair for SSH to the cluster Info**

vockey  
key-0c20c9d7bbb2eba75

winepredkey  
key-0892e38bbe1b540d9

want to enable SSH or use Hue SQL assistant

- f. Select the EMR\_DefaultRole as the service role and EMR\_EC2\_DefaultRole as the instance profile and the rest as it is.

The screenshot shows the 'Identity and Access Management (IAM) roles' page in the AWS console, specifically for configuring roles for Amazon EMR. The page has a dark theme. At the top, it says 'Choose or create a service role and instance profile for the EC2 instances in your cluster.' Below this, there are two main sections: 'Amazon EMR service role' and 'EC2 instance profile for Amazon EMR'. Each section has two options: 'Choose an existing [role/profile]' and 'Create a [role/profile]'. In the 'Service role' section, 'EMR\_DefaultRole' is selected. In the 'Instance profile' section, 'EMR\_EC2\_DefaultRole' is selected. At the bottom, there is a section for 'Custom automatic scaling role - optional' with a dropdown set to 'Choose IAM role' and a 'Create IAM role' button.

**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

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

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

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

Instance profile  
EMR\_EC2\_DefaultRole

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

Custom automatic scaling role  
Choose IAM role

Create IAM role

- g. Verify the summary and create the cluster

The screenshot shows a summary bar at the bottom of the AWS console with two buttons: 'Cancel' and 'Create cluster'.

Cancel Create cluster

### 3. Creating S3 Bucket:

- a. Go Amazon S3 > Buckets and create an S3 bucket with a name as shown below

The screenshot shows the 'Buckets' page in the Amazon S3 console. It has a dark theme. At the top, there's a navigation bar with 'S3', 'EC2', and 'EMR' icons. Below the navigation bar, there's a section for 'Account snapshot' with a 'View Storage Lens dashboard' button. The main content area is titled 'General purpose buckets' and 'Directory buckets'. Under 'General purpose buckets', there's a list of buckets (currently empty) and a 'Create bucket' button, which is circled in red.

Amazon S3 > Buckets

Account snapshot  
Storage lens provides visibility into storage usage and activity trends. [Learn more](#)

View Storage Lens dashboard

General purpose buckets Directory buckets

General purpose buckets (2) [Info](#) Copy ARN Empty Delete Create bucket

Buckets are containers for data stored in S3. [Learn more](#)

- b. Select the AWS region, give a name to the S3 bucket, keep the rest settings as it is

Amazon S3 > Buckets > Create bucket

### Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

#### General configuration

AWS Region  
US East (N. Virginia) us-east-1

Bucket type [Info](#)

☒ **General purpose**  
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory - New**  
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)  
sk3374-winepred

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

#### Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

[Cancel](#) [Create bucket](#)

- c. Once created it should appear in the list of s3 buckets

### General purpose buckets (2) [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

Find buckets by name

	Name	Location	Access
<input type="radio"/>	aws-logs-246424041445-us-east-1	US East (N. Virginia)	Public
<input type="radio"/>	sk3374-winepred	US East (N. Virginia)	Public

- d. Upload the TrainingDataset.csv and ValidationDataset.csv and the src/wine\_prediction\_train.py train and

### Upload [Info](#)

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose [Add files](#) or [Add folder](#).

#### Files and folders (3 Total, 79.3 KB)

[Remove](#) [Add files](#) [Add folder](#)

All files and folders in this table will be uploaded.

Find by name

<input type="checkbox"/>	Name	Folder	Type	Size
<input type="checkbox"/>	wine_prediction_train.py	src/	text/x-python	3.6 KB
<input type="checkbox"/>	ValidationDataset.csv	-	text/csv	8.6 KB
<input type="checkbox"/>	TrainingDataset.csv	-	text/csv	67.2 KB

#### Destination [Info](#)

Destination  
s3://sk3374-winepred

[Destination details](#)  
Bucket settings that impact new objects stored in the specified destination.

[Permissions](#)  
Grant public access and access to other AWS accounts.

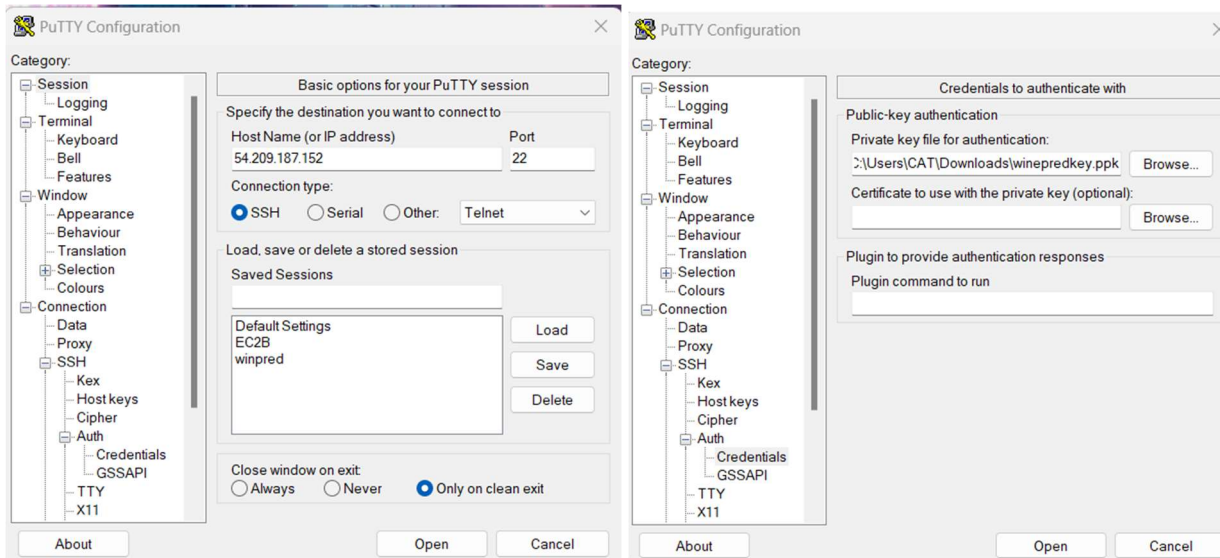
[Properties](#)  
Specify storage class, encryption settings, tags, and more.

[Cancel](#) [Upload](#)

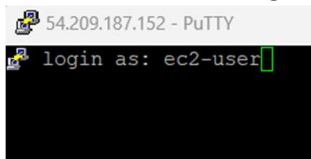
- e. The folder "src/" would have the code that trains and creates the model.

#### 4. Parallely Training the Model in the EMR Cluster

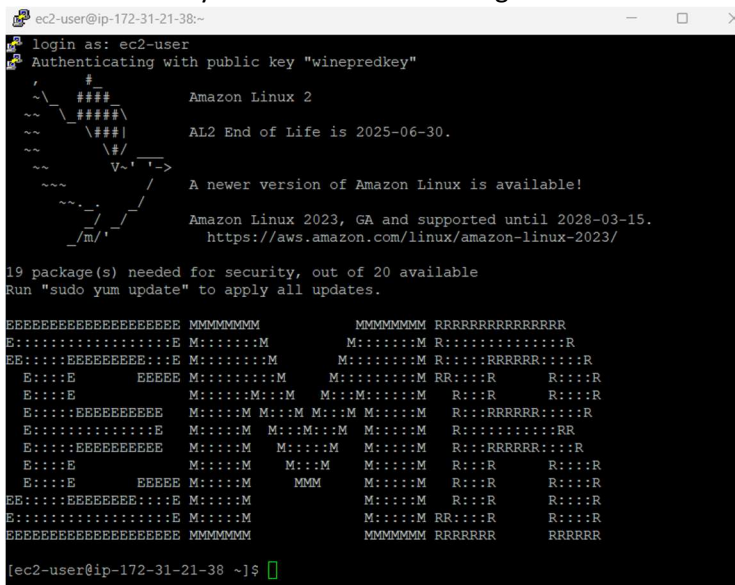
- a. Connect to the EMR EC2 Master node through putty using the “winepred.ppk” file from your local as shown below:



- b. Use ec2-user as the login user



- c. After successful you should see something like this





- d. Now run the command “`sudo spark-submit s3://sk3374-winepred/src/wine_prediction_train.py`” to start and train the model

```
E:::EEEEEEEE M:::M M:::M M:::M R:::RRRRR:::R
E:::EEEEEEEE M:::M M:::M M:::M R:::RRRRR:::R
E:::EEEEEEEE M:::M M:::M M:::M R:::RRRRR:::R
E:::E M:::M M:::M M:::M R:::R R:::R
E:::E EEEEE M:::M M:::M M:::M R:::R R:::R
EE:::EEEEEEEE M:::M M:::M M:::M R:::R R:::R
E:::EEEEEEEE M:::M M:::M M:::M R:::R R:::R
EEEEEEEEEEEEEEEE M:::M M:::M M:::M R:::R R:::R

[ec2-user@ip-172-31-21-38 ~]$ sudo spark-submit s3://sk3374-winepred/src/wine_prediction_train.py
23/12/08 00:40:42 INFO SparkContext: Running Spark version 3.4.1-amzn-2
23/12/08 00:40:42 INFO ResourceUtils: =====
```

- e. You will see something like this it the code is running and the model is getting trained:

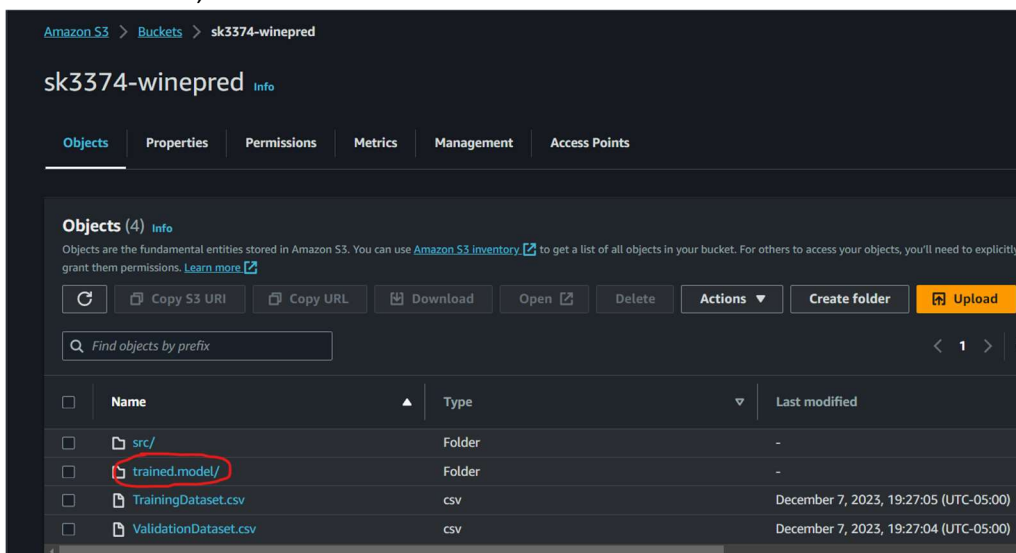
```
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /environment/json: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /executors: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /executors/json: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /executors/threadDump: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /executors/threadDump/json: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /static: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /api: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /jobs/job/kill: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /stages/stage/kill: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /metrics/json: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO YarnClientSchedulerBackend: SchedulerBackend is ready for scheduling beginning after reached minRegisteredResourcesRatio: 0.0
Initial Model Test Accuracy: 0.99375
/usr/lib/spark/python/lib/pyspark.zip/pyspark/sql/context.py:159: FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead.
Initial Model Weighted f1 score: 0.9933730158730157

23/12/08 00:45:02 INFO ServerInfo: Adding filter to /api: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /jobs/job/kill: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO ServerInfo: Adding filter to /stages/stage/kill: org.apache.hadoop.yarn.server.webproxy.amfilter.AmIpFilter
23/12/08 00:45:02 INFO YarnClientSchedulerBackend: SchedulerBackend is ready for scheduling beginning after reached minRegisteredResourcesRatio: 0.0
Initial Model Test Accuracy: 0.99375
/usr/lib/spark/python/lib/pyspark.zip/pyspark/sql/context.py:159: FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead.
Initial Model Weighted f1 score: 0.9933730158730157
Best Model: PipelineModel_5452385c59fe
Best Model Test Accuracy: 0.96875
Best Model Weighted f1 score: 0.9547916666666667
[ec2-user@ip-172-31-21-38 ~]$
```

- f. Once the code completed executed it should create a model and store in in the AWS S3 Bucket that you had created “`s3://sk3374-winepred`” earlier as per the login in the code:

```
18 input_path_train = "s3://sk3374-winepred/TrainingDataset.csv"
19 input_path_valid = "s3://sk3374-winepred/ValidationDataset.csv"
20 output_path = "s3://sk3374-winepred/trained.model"
21
22 # Reading and cleaning data
23
79 # Save the best model
80 best_model.write().overwrite().save(output_path)
81
```

- g. As shown below, the model would be created in the s3 bucket in the folder “`trained.model/`”



## 5. Testing the Trained Model on EC2 Instance and Locally

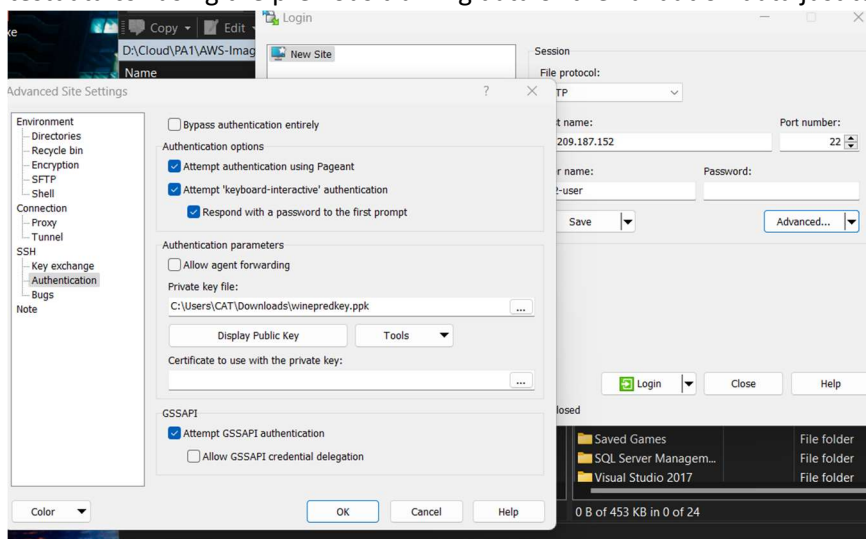
- a. Now can you can get the model from your s3 bucket to your EC2 instance node using the below command  
“aws s3 sync s3://sk3374-winepred/trained.model ./trained.model/”

```
ec2-user@ip-172-31-21-38:~$ aws s3 sync s3://sk3374-winepred/trained.model ./trained.model/
```

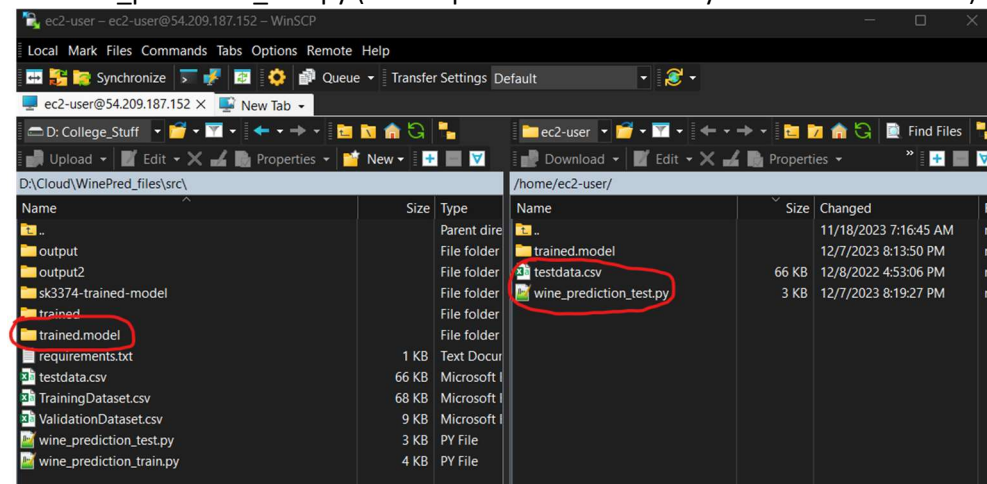
Now when you do ls you can see the downloaded model into your instance

```
[ec2-user@ip-172-31-21-38 ~]$ ls
trained.model
[ec2-user@ip-172-31-21-38 ~]$ cd trained.model
[ec2-user@ip-172-31-21-38 trained.model]$ ls
metadata stages
[ec2-user@ip-172-31-21-38 trained.model]$
```

- b. Now using WinSCP connect to the EC2 instances to test the trained model using a script. Create a testdata.csv using the previous training data or the validation data just to test the model



- c. Now download the trained.model folder from the EC2 instance to your local and transfer the testdata.csv and wine\_prediction\_test.py (the script to test the accuracy of the trained model) to the EC2 instance





- d. One done make sure all the files needed for testing the trained model are on your EC2 instance.

```
ec2-user@ip-172-31-21-38:~  
[ec2-user@ip-172-31-21-38 ~]$ ls  
testdata.csv  trained.model  wine_prediction_test.py  
[ec2-user@ip-172-31-21-38 ~]$
```

- e. Now go ahead and run the wine\_prediction\_test.py using the below command  
python wine\_prediction\_test.py

```
root@ip-172-31-21-38:/home/ec2-user  
[root@ip-172-31-21-38 ec2-user]# python wine_prediction_test.py  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
23/12/08 02:06:54 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using  
/usr/local/lib/python3.7/site-packages/pyspark/context.py:317: FutureWarning: Python 3.7 support is deprecated  
warnings.warn("Python 3.7 support is deprecated in Spark 3.4.", FutureWarning)  
Current directory:  
/home/ec2-user  
Sample predictions:  
+-----+-----+-----+-----+-----+-----+-----+-----+  
|fixed acidity|volatile acidity|citric acid|residual sugar|chlorides|free sulfur dioxide|total sulfur dioxide|  
+-----+-----+-----+-----+-----+-----+-----+-----+  
| 8.9| 0.22| 0.48| 1.8| 0.077| 29.0|  
| 7.6| 0.39| 0.31| 2.3| 0.082| 23.0|  
| 7.9| 0.43| 0.21| 1.6| 0.106| 10.0|  
| 8.5| 0.49| 0.11| 2.3| 0.084| 9.0|  
| 6.9| 0.4| 0.14| 2.4| 0.085| 21.0|  
+-----+-----+-----+-----+-----+-----+-----+-----+  
only showing top 5 rows  
  
None  
Test Accuracy of the wine prediction model: 0.9843627834245504  
/usr/local/lib/python3.7/site-packages/pyspark/sql/context.py:159: FutureWarning: Deprecated in 3.0.0.  
FutureWarning,  
Weighted F1 score of the wine prediction model: 0.9779339666913879  
[root@ip-172-31-21-38 ec2-user]#
```

- f. Also you can run the wine\_prediction\_test.py in your local once you get the trained.model from the S3 as shown below

```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS D:\Cloud\WinePred\src> python wine_prediction_test.py  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).  
Current directory:  
D:\Cloud\WinePred\src  
Sample predictions:  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
|fixed acidity|volatile acidity|citric acid|residual sugar|chlorides|free sulfur dioxide|total sulfur dioxide|density| pH|sulphates|  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
| 8.9| 0.22| 0.48| 1.8| 0.077| 29.0| 60.0| 0.9968| 3.39| 0.53|  
| 7.6| 0.39| 0.31| 2.3| 0.082| 23.0| 71.0| 0.9982| 3.52| 0.65|  
| 7.9| 0.43| 0.21| 1.6| 0.106| 10.0| 37.0| 0.9966| 3.17| 0.91|  
| 8.5| 0.49| 0.11| 2.3| 0.084| 9.0| 67.0| 0.9968| 3.17| 0.53|  
| 6.9| 0.4| 0.14| 2.4| 0.085| 21.0| 40.0| 0.9968| 3.43| 0.63|  
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+  
only showing top 5 rows  
  
None  
Test Accuracy of the wine prediction model: 0.9843627834245504  
C:\Python311\Lib\site-packages\pyspark\sql\context.py:158: FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate()  
warnings.warn(  
Weighted F1 score of the wine prediction model: 0.9779339666913879  
0.9843627834245504, 0.9779339666913879, 0.9843627834245504, 0.9779339666913879, 0.9843627834245504, 0.9779339666913879, 0.9843627834245504, 0.9779339666913879, 0.9843627834245504, 0.9779339666913879
```

## 6. Create a Docker Image and Push it to DockerHub

- Now create a Dockerfile with all the required configurations to run the wine\_prediction\_test.py in your repo
- Build an image using Dockerfile. Below is the command that is used create and build a docker image.

Command: "docker build -t winepredimage:version1 ."

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\Cloud\WinePredictionAnalysis> docker build -t winepredimage:version1 .
[+] Building 1.0s (18/18) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 1.34kB
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/centos:7
=> [auth] library/centos:pull token for registry-1.docker.io
=> [ 1/12] FROM docker.io/library/centos:7@sha256:be65f488b7764ad3638f236b7b515b3678369a5124c47b8d
=> [internal] load build context
=> => transferring context: 744.71kB
=> CACHED [ 2/12] RUN yum -y update && yum -y install python3 python3-dev python3-pip python3-virt
=> CACHED [ 3/12] RUN python -V && python3 -V
=> CACHED [ 4/12] RUN pip3 install --upgrade pip && pip3 install numpy pandas pyspark
=> CACHED [ 5/12] RUN wget --no-verbose -O apache-spark.tgz "https://archive.apache.org/dist/spark
=> CACHED [ 6/12] RUN ln -s /opt/spark-3.1.2-bin-hadoop2.7 /opt/spark
=> CACHED [ 7/12] RUN echo 'export SPARK_HOME=/opt/spark' >> ~/.bashrc && echo 'export PATH=$S
=> CACHED [ 8/12] RUN mkdir -p /winepred/src /winepred/src/trained.model
=> CACHED [ 9/12] COPY src/wine_prediction_test.py /winepred/src/
=> CACHED [10/12] COPY src/testdata.csv /winepred/src/
=> CACHED [11/12] COPY src/trained.model /winepred/src/trained.model/
=> CACHED [12/12] WORKDIR /winepred/src/
=> exporting to image
=> => exporting layers
=> => writing image sha256:e3117fd38f8a04ff89d245f899f4dfd9cd138bd44cc0b7cc0af76b5b45009029
=> => naming to docker.io/library/winepredimage:version1
PS D:\Cloud\WinePredictionAnalysis>
```

- Once built, test it on your local using the docker run command as shown below

Command: "docker run winepredimage:version1" (This command is only for running on local)

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

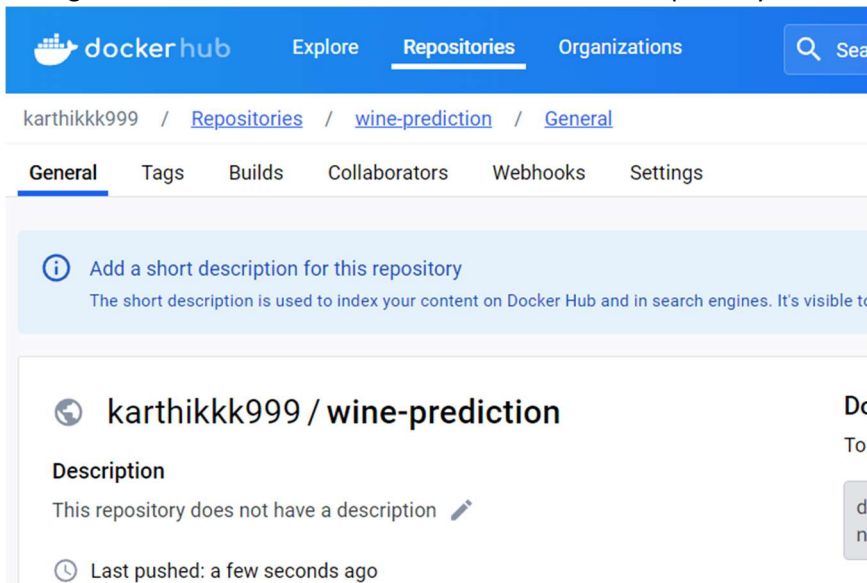
PS D:\Cloud\WinePredictionAnalysis> docker run winepredimage:version1
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/12/08 03:54:23 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-
/usr/local/lib/python3.6/site-packages/pyspark/context.py:238: FutureWarning: Python 3.6 support is deprecated in
FutureWarning
/usr/local/lib/python3.6/site-packages/pyspark/sql/context.py:127: FutureWarning: Deprecated in 3.0.0. Use Sparks
FutureWarning
Current directory:
/winepred/src
Sample predictions:
+-----+-----+-----+-----+-----+-----+-----+-----+
|fixed acidity|volatile acidity|citric acid|residual sugar|chlorides|free sulfur dioxide|total sulfur dioxide|der
probability|prediction|
+-----+-----+-----+-----+-----+-----+-----+-----+
|      8.9|      0.22|      0.48|      1.8|      0.077|      29.0|      60.0| 0.
90372428107...|      1.0|      0.39|      0.31|      2.3|      0.082|      23.0|      71.0| 0.
06291204975...|      0.0|      0.43|      0.21|      1.6|      0.106|      10.0|      37.0| 0.
77161820213...|      0.0|      0.49|      0.11|      2.3|      0.084|      9.0|      67.0| 0.
32641896582...|      0.0|      0.4|      0.14|      2.4|      0.085|      21.0|      40.0| 0.
58382336961...|      1.0|
+-----+-----+-----+-----+-----+-----+-----+-----+
only showing top 5 rows

None
Test Accuracy of the wine prediction model: 0.9843627834245504
Weighted F1 score of the wine prediction model: 0.9779339666913879
PS D:\Cloud\WinePredictionAnalysis>
```

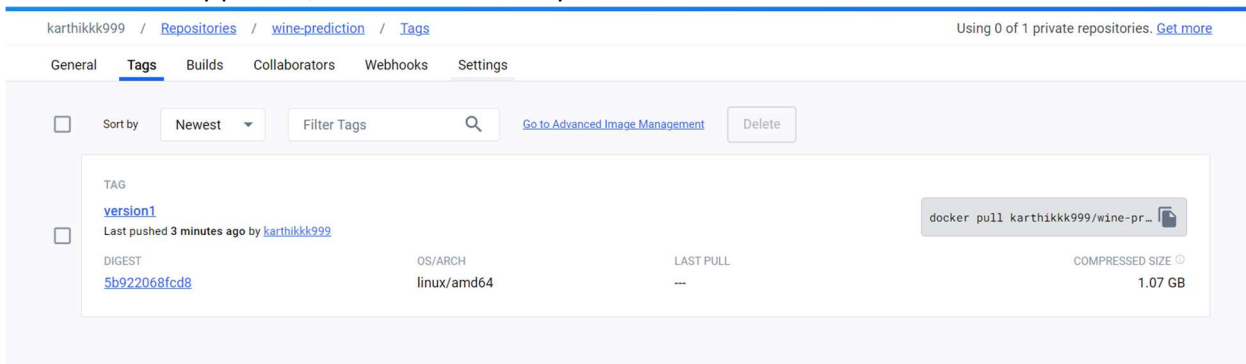
- d. Now use docker login to login to your DockerHub account. (I have already logged in and that's why it said Login Succeeded)

```
PS D:\Cloud\WinePredictionAnalysis> docker login
Authenticating with existing credentials...
Login Succeeded
```

- e. Now go to DockerHub on a web browser and create a repository called wine-prediction



- f. Now tag the local image to your repository that you have created using the below command  
“docker tag winepredimage:version1 karthikkk999/wine-prediction:version1”
- g. And push the docker image to the DockerHub using the command  
“docker push karthikkk999/wine-prediction:version1”
- h. Once successfully pushed, You can see this on your DockerHub account.



## 7. Pulling the Docker Image and Running it on an EC2 Instance

- a. SSH into any of the EC2 instances that you want to run the docker image and install docker

“sudo yum install docker”

```
[ec2-user@ip-172-31-25-239 ~]$ sudo yum install docker
Last metadata expiration check: 0:01:55 ago on Fri Dec 8 04:24:49 2023.
Dependencies resolved.
=====
Package                               Arch      Version                               Repository      Size
=====
Installing:
docker                                x86_64    24.0.5-1.amzn2023.0.2                amazonlinux     42 M
Installing dependencies:
containerd                            x86_64    1.7.2-1.amzn2023.0.4                amazonlinux     34 M
iptables-libs                         x86_64    1.8.8-3.amzn2023.0.2                amazonlinux    401 k
iptables-nft                         x86_64    1.8.8-3.amzn2023.0.2                amazonlinux    183 k
libcgroup                            x86_64    3.0-1.amzn2023.0.1                  amazonlinux     75 k
libnetfilter_conntrack               x86_64    1.0.8-2.amzn2023.0.2                amazonlinux     58 k
libnftnl                             x86_64    1.0.1-19.amzn2023.0.2               amazonlinux     30 k
libnftnl                             x86_64    1.2.2-2.amzn2023.0.2                amazonlinux     84 k
pigz                                  x86_64    2.5-1.amzn2023.0.3                  amazonlinux     83 k
runc                                  x86_64    1.1.7-1.amzn2023.0.2               amazonlinux     2.0 M
Installed:
 containerd-1.7.2-1.amzn2023.0.4.x86_64  docker-24.0.5-1.amzn2023.0.2.x86_64  iptables-
 libnftnl-1.0.1-19.amzn2023.0.2.x86_64  libnftnl-1.2.2-2.amzn2023.0.2.x86_64  pigz-2.5-
Complete!
[ec2-user@ip-172-31-25-239 ~]$
```

- b. Start docker and pull the wine-prediction image that you have built

“sudo service docker start”

“sudo docker pull karthikkk999/wine-prediction:version1”

```
[ec2-user@ip-172-31-25-239 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-25-239 ~]$ sudo docker pull karthikkk999/wine-prediction:version1
version1: Pulling from karthikkk999/wine-prediction
2d473b07cdd5: Extracting 6.685MB/76.1MB
2d6546646a1e: Download complete
e162e3ae95db: Download complete
2754057e8779: Downloading 290MB/626.4MB
f86de3a212d7: Downloading 166.7MB/229.3MB
83873b103326: Download complete
b54bd823ef8f: Download complete
e41f60adab8a: Download complete
d642780fcde7: Download complete
9117622a57c5: Download complete
33b95396b6c5: Download complete
4f4fb700ef54: Waiting
█
```



- c. Once the docker image is pulled run it using the docker run command  
"sudo docker run karthikkk999/wine-prediction:version1"

```
ec2-user@ip-172-31-25-239:~  
[ec2-user@ip-172-31-25-239 ~]$ sudo docker run karthikkk999/wine-prediction:ver  
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties  
Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel  
23/12/08 04:31:56 WARN NativeCodeLoader: Unable to load native-hadoop library f  
/usr/local/lib/python3.6/site-packages/pyspark/context.py:238: FutureWarning: P  
FutureWarning  
/usr/local/lib/python3.6/site-packages/pyspark/sql/context.py:127: FutureWarnin  
.  
FutureWarning  
Current directory:  
/winepred/src  
Sample predictions:  
+-----+-----+-----+-----+-----+-----+-----+-----+  
+-----+-----+-----+-----+-----+-----+-----+-----+  
|fixed acidity|volatile acidity|citric acid|residual sugar|chlorides|free sulfu  
features|label|rawPrediction|probability|prediction|  
+-----+-----+-----+-----+-----+-----+-----+-----+  
+-----+-----+-----+-----+-----+-----+-----+-----+  
|8.9|0.22|0.48|1.8|0.077|1.0|  
[8.9,0.22,0.48,1....|1.0|[19.4518621405359...|[0.03890372428107...|1.0|  
|7.6|0.39|0.31|2.3|0.082|0.0|  
[7.6,0.39,0.31,2....|0.0|[479.531456024878...|[0.95906291204975...|0.0|  
|7.9|0.43|0.21|1.6|0.106|0.0|  
[7.9,0.43,0.21,1....|0.0|[484.885809101067...|[0.96977161820213...|0.0|  
|8.5|0.49|0.11|2.3|0.084|0.0|  
[8.5,0.49,0.11,2....|0.0|[491.163209482913...|[0.98232641896582...|0.0|  
|6.9|0.4|0.14|2.4|0.085|1.0|  
[6.9,0.4,0.14,2.4...|1.0|[3.29191168480642...|[0.00658382336961...|1.0|  
+-----+-----+-----+-----+-----+-----+-----+-----+  
+-----+-----+-----+-----+-----+-----+-----+-----+  
only showing top 5 rows  
  
None  
Test Accuracy of the wine prediction model: 0.9843627834245504  
Weighted F1 score of the wine prediction model: 0.9779339666913879  
[ec2-user@ip-172-31-25-239 ~]$
```

DONE!