

Configuration Manual

MSc Research Project
MSc Cloud Computing

Shreya Modak
Student ID: 21208671

School of Computing
National College of Ireland

Supervisor: Shreyas Setlur Arun

National College of Ireland
Project Submission Sheet
School of Computing



| | |
|-----------------------------|----------------------|
| Student Name: | Shreya Modak |
| Student ID: | 21208671 |
| Programme: | MSc Cloud Computing |
| Year: | 2023 |
| Module: | MSc Research Project |
| Supervisor: | Shreyas Setlur Arun |
| Submission Due Date: | 14/12/2023 |
| Project Title: | Configuration Manual |
| Word Count: | 311 |
| Page Count: | 10 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

| | |
|-------------------|--------------------|
| Signature: | Shreya Modak |
| Date: | 14th December 2023 |

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

| | |
|--|--------------------------|
| Attach a completed copy of this sheet to each project (including multiple copies). | <input type="checkbox"/> |
| Attach a Moodle submission receipt of the online project submission , to each project (including multiple copies). | <input type="checkbox"/> |
| You must ensure that you retain a HARD COPY of the project , both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | <input type="checkbox"/> |

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

| Office Use Only | |
|----------------------------------|--|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

Configuration Manual

Shreya Modak
21208671

1 Introduction

The steps to configure the code files (.ipynb files) are outlined in the configuration manual.

2 Requirements

HARDWARE REQUIREMENT






- Operating System: Compatible with multiple operating systems, Windows, Mac and Linux.

SOFTWARE REQUIREMENT

- AWS Account
- AWS Sagemaker Access




3 Code Configuration

- There are four .ipynb files and as their name suggest, there are three separate and one combine or comparative analysis done on the code.
- A total of six models were trained under a distributed and parallel processing environment to be run on the Amazon Sazemaker.

| | | | |
|--|------------------|---------------------|----------|
|  Dataset csv files | 14-12-2023 00:13 | File folder | |
|  Comparision of the three at one instance | 13-12-2023 23:43 | Jupyter Source File | 1,440 KB |
|  distributed_keras | 13-12-2023 23:52 | Jupyter Source File | 8 KB |
|  distributed_randomforest_instance | 13-12-2023 23:53 | Jupyter Source File | 4 KB |
|  distributed_xGBoost_instance | 13-12-2023 23:53 | Jupyter Source File | 12 KB |

4 Dataset Configuration

- The dataset folder contains the csv data files required to run the code.

| | | | | |
|---|---|------------------|----------------------|-----------|
|  | diabetes_012_health_indicators_BRFSS201... | 13-12-2023 23:55 | Microsoft Excel C... | 22,206 KB |
|  | diabetes_binary_5050split_health_indicat... | 13-12-2023 23:56 | Microsoft Excel C... | 6,199 KB |
|  | diabetes_binary_health_indicators_BRFSS... | 14-12-2023 00:01 | Microsoft Excel C... | 22,206 KB |

5 Implementation

- Login to AWS MANAGEMENT CONSOLE using root user login and password.



Sign in

☒ **Root user**

Account owner that performs tasks requiring unrestricted access. [Learn more](#)

☐ **IAM user**

User within an account that performs daily tasks. [Learn more](#)

Root user email address

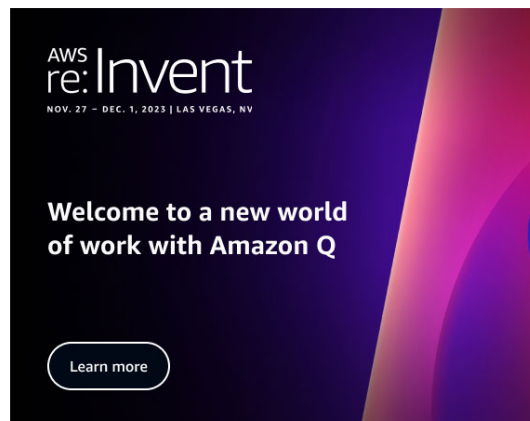
pbcdmatlab@gmail.com

Next

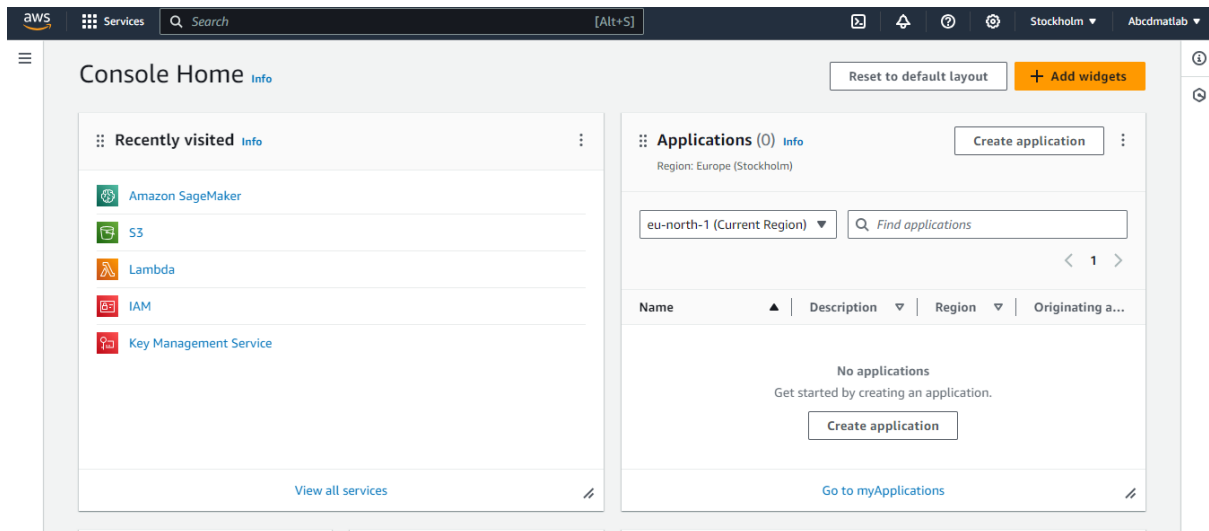
By continuing, you agree to the [AWS Customer Agreement](#) or other agreement for AWS services, and the [Privacy Notice](#). This site uses essential cookies. See our [Cookie Notice](#) for more information.

New to AWS?

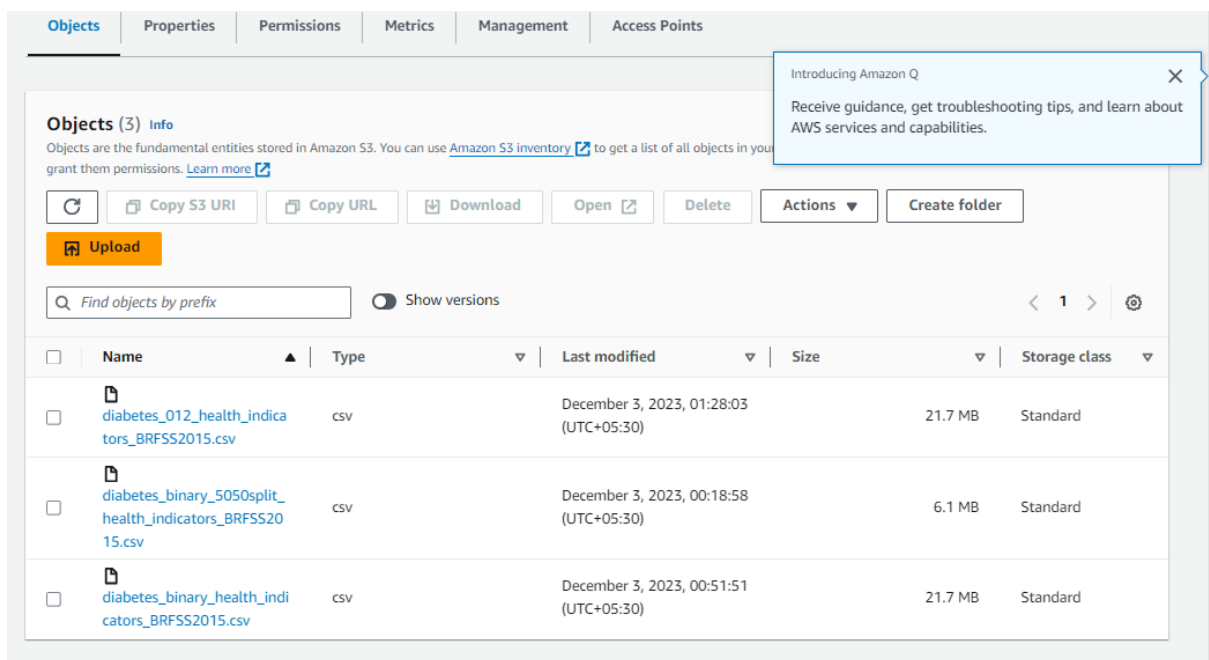
Create a new AWS account



- Open S3 bucket



- Upload all three csv data to a s3 bucket, which have all the permissions for s3 functions.




Amazon S3 > Buckets > mybucketabcdmat

mybucketabcdmat Info Publicly accessible

Introducing Amazon Q
Receive guidance, get troubleshooting tips, and learn about AWS services and capabilities.


Objects | Properties | **Permissions** | Metrics | Management | Access Points

Permissions overview

Access
 **Public**

Block public access (bucket settings) Edit

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
 **Off**

► Individual Block Public Access settings for this bucket

- Set the bucket policy as shown

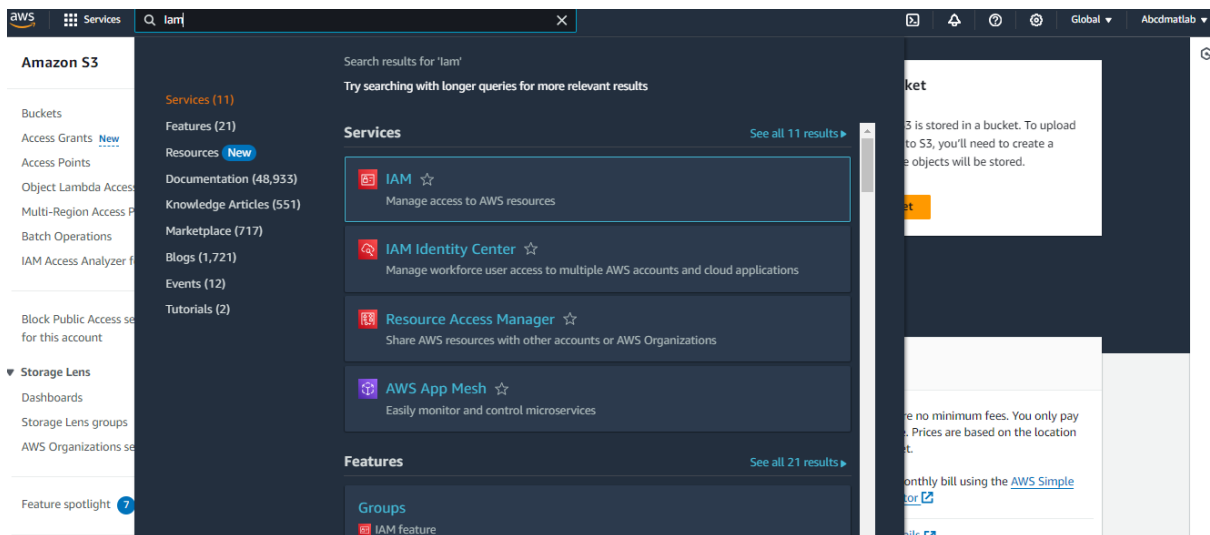
Bucket policy Edit Delete

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

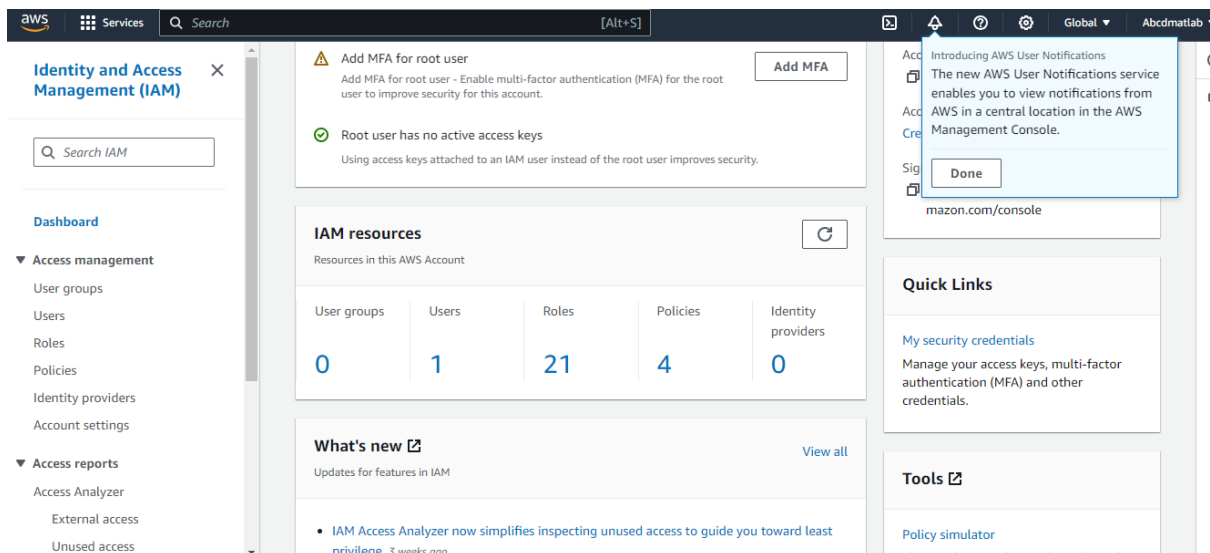
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::mybucketabcdmat/*"
    }
  ]
}
```

Copy

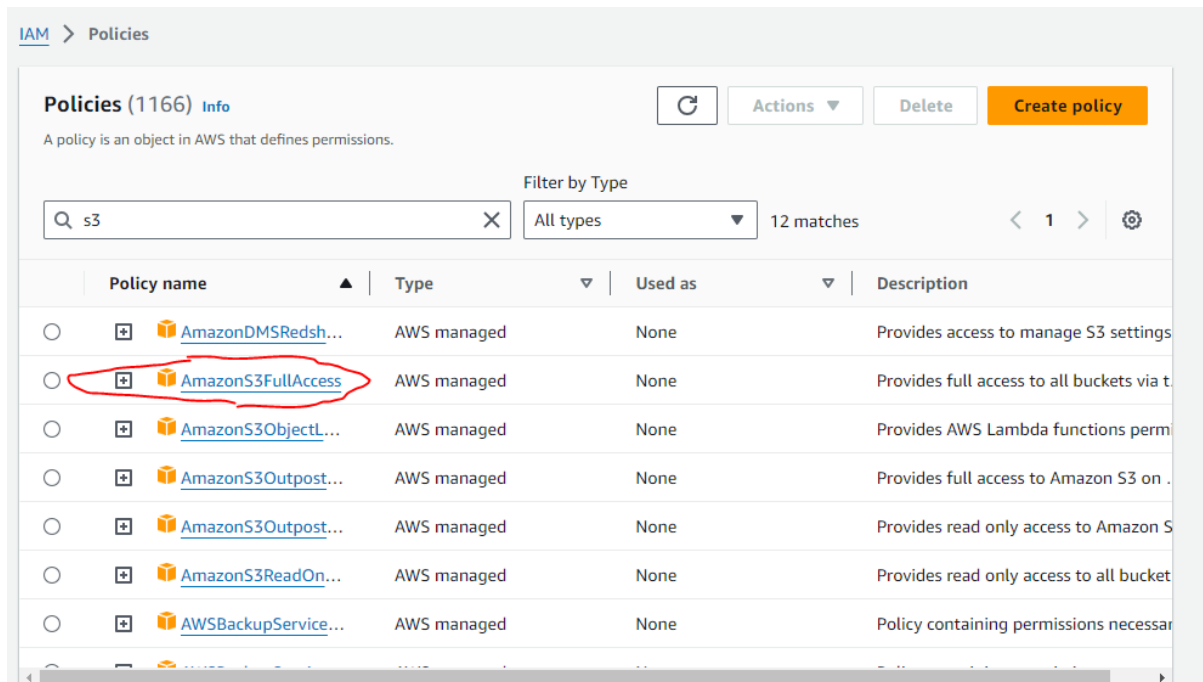
- Create IAM Roles in a separate tab for accessing the buckets



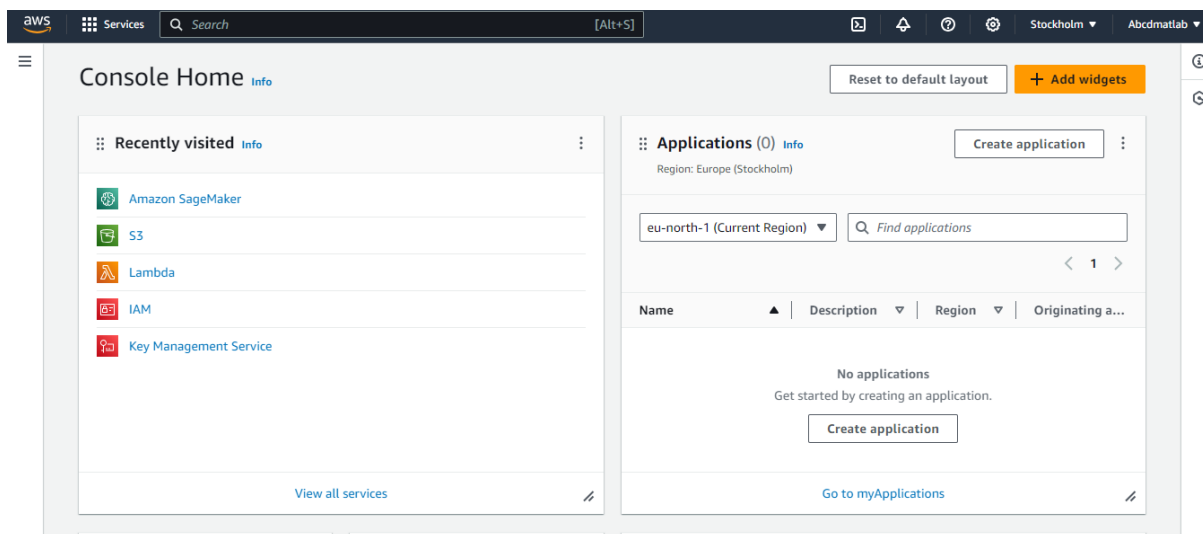
- Create a user and then their role



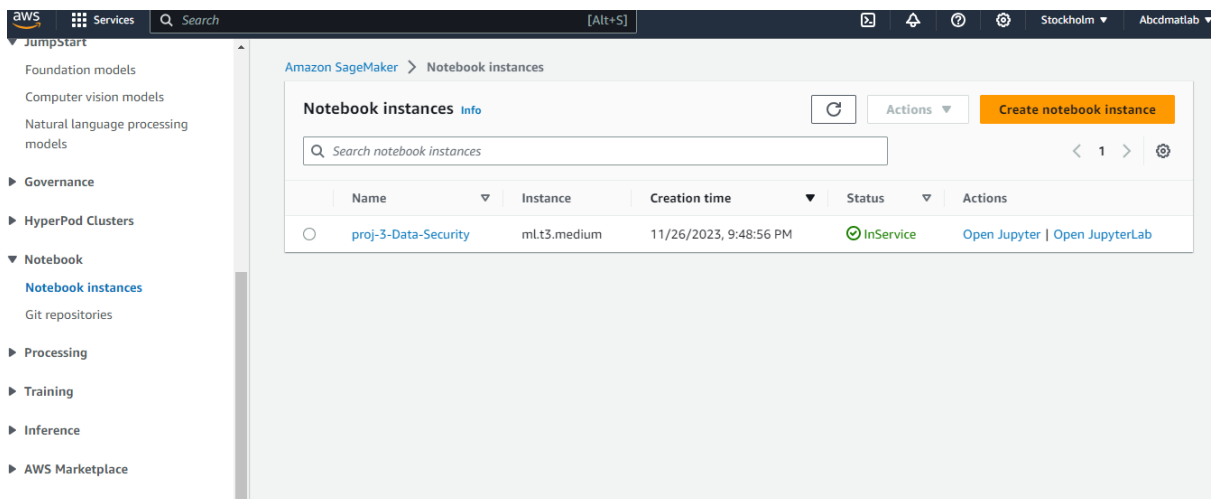
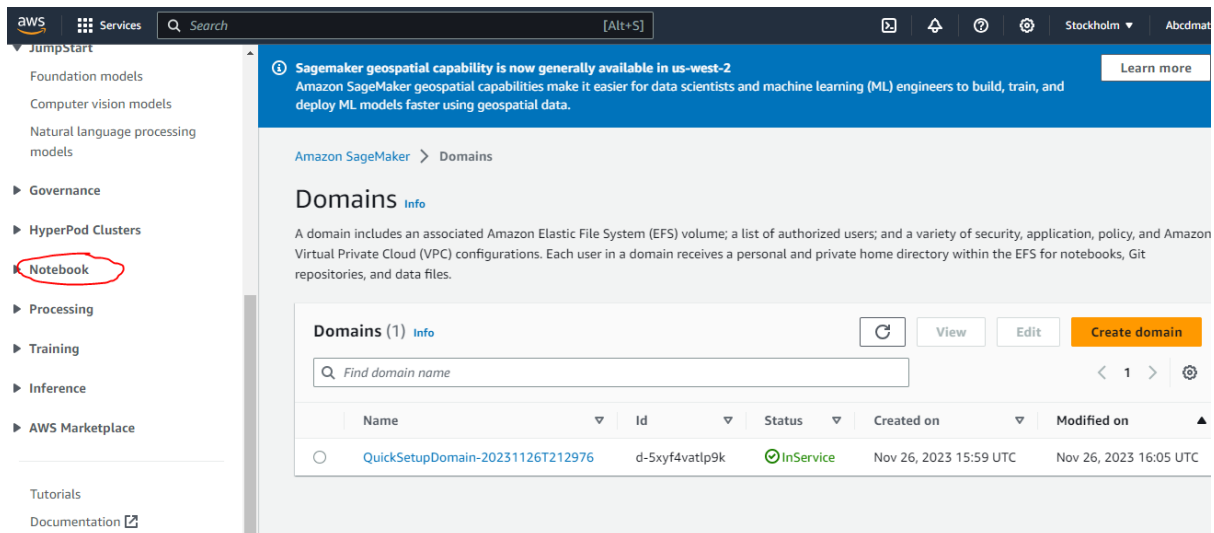
- Under Policies, allow for s3 Full access



- Open Amazon Sagemaker, by going back to managemnt console and opening AWS Sagemaker



- Search for the notebook instances in the left pane



- Create a notebook instance if you don't have open. After a notebook instance has been created and ready, you can further open it in jupyter and upload code files as shown below

| | Name | Last Modified | File size |
|--------------------------|---|-----------------------|-----------|
| <input type="checkbox"/> | Comparison of the three at one instance.ipynb | Running 2 hours ago | 1.47 MB |
| <input type="checkbox"/> | distributed_keras.ipynb | Running 5 minutes ago | 7.04 kB |
| <input type="checkbox"/> | distributed_randomforest_instance.ipynb | Running 11 days ago | 3.76 kB |
| <input type="checkbox"/> | distributed_xGBoost_instance.ipynb | Running 11 days ago | 12 kB |

- Open the distributed_keras.ipynb and do the required pip installs.

pip install pandas numpy seaborn boto3 matplotlib scikit-learn

- Run the model for keras and observe the accuracy and classification score

```

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from sklearn.metrics import accuracy_score, classification_report

# Create a simple neural network model
model = Sequential()
model.add(Dense(64, input_dim=X_train.shape[1], activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(3, activation='softmax')) # Adjust the number of units based on your target classes

# Compile the model
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))

# Evaluate the model
nn_predictions = model.predict(X_test)
nn_predictions_classes = np.argmax(nn_predictions, axis=1) # Get the index of the maximum value along axis 1

nn_accuracy = accuracy_score(y_test, nn_predictions_classes)
nn_classification_report = classification_report(y_test, nn_predictions_classes)

# Print the results
print(f'Neural Network Accuracy: {nn_accuracy}')
print('Neural Network Classification Report:')
print(nn_classification_report)

```

Epoch 1/10

2023-12-02 22:29:49.652841: W external/local_tsl/tsl/framework/cpu_allocator_impl.cc:83] Allocation of 34094592 exceeds 10% of free system memory.

6342/6342 [=====] - 16s 2ms/step - loss: 0.4239 - accuracy: 0.8432 - val_loss: 0.4028 - val_accuracy: 0.8481

Epoch 2/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.4055 - accuracy: 0.8464 - val_loss: 0.3969 - val_accuracy: 0.8492

Epoch 3/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.4019 - accuracy: 0.8474 - val_loss: 0.3947 - val_accuracy: 0.8496

Epoch 4/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.3999 - accuracy: 0.8475 - val_loss: 0.3943 - val_accuracy: 0.8476

Epoch 5/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.3985 - accuracy: 0.8478 - val_loss: 0.3942 - val_accuracy: 0.8484

Epoch 6/10
6342/6342 [=====] - 13s 2ms/step - loss: 0.3976 - accuracy: 0.8480 - val_loss: 0.3933 - val_accuracy: 0.8499

Epoch 7/10
6342/6342 [=====] - 13s 2ms/step - loss: 0.3975 - accuracy: 0.8485 - val_loss: 0.3955 - val_accuracy: 0.8480

Epoch 8/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.3971 - accuracy: 0.8480 - val_loss: 0.3914 - val_accuracy: 0.8501

Epoch 9/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.3967 - accuracy: 0.8481 - val_loss: 0.3922 - val_accuracy: 0.8499

Epoch 10/10
6342/6342 [=====] - 12s 2ms/step - loss: 0.3964 - accuracy: 0.8486 - val_loss: 0.3925 - val_accuracy: 0.8486

1586/1586 [=====] - 2s 1ms/step
Neural Network Accuracy: 0.8486281929990539

Neural Network Classification Report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0.0 | 0.87 | 0.97 | 0.92 | 42795 |
| 1.0 | 0.00 | 0.00 | 0.00 | 944 |
| 2.0 | 0.53 | 0.22 | 0.31 | 6997 |
| accuracy | | | 0.85 | 50736 |
| macro avg | 0.47 | 0.40 | 0.41 | 50736 |
| weighted avg | 0.81 | 0.85 | 0.82 | 50736 |

- Similarly, run the random forest model

```
In [17]: from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Example: Random Forest Classifier
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestClassifier()
model.fit(X_train, y_train)
predictions = model.predict(X_test)

# Evaluate the model
print("Accuracy:", accuracy_score(y_test, predictions))
print("Classification Report:\n", classification_report(y_test, predictions))
```

Accuracy: 0.8417888678650268

Classification Report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0.0 | 0.86 | 0.97 | 0.91 | 42795 |
| 1.0 | 0.00 | 0.00 | 0.00 | 944 |
| 2.0 | 0.47 | 0.20 | 0.28 | 6997 |
| accuracy | | | 0.84 | 50736 |
| macro avg | 0.45 | 0.39 | 0.40 | 50736 |
| weighted avg | 0.79 | 0.84 | 0.81 | 50736 |

- Similarly, run the xgboost model and observe the results

```
In [61]: from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report

# Assuming X_train, X_test, y_train, and y_test are defined

# Create and train the XGBoost model
xgb_model = XGBClassifier()
xgb_model.fit(X_train, y_train)

# Make predictions
xgb_predictions = xgb_model.predict(X_test)

# Evaluate the model
xgb_accuracy = accuracy_score(y_test, xgb_predictions)
xgb_classification_report = classification_report(y_test, xgb_predictions)

# Print the results
print(f'XGBoost Accuracy: {xgb_accuracy}')
print('XGBoost Classification Report:')
print(xgb_classification_report)
```

XGBoost Accuracy: 0.8504415011037527

XGBoost Classification Report:

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0.0 | 0.87 | 0.98 | 0.92 | 42795 |
| 1.0 | 0.00 | 0.00 | 0.00 | 944 |
| 2.0 | 0.55 | 0.20 | 0.29 | 6997 |
| accuracy | | | 0.85 | 50736 |
| macro avg | 0.47 | 0.39 | 0.40 | 50736 |
| weighted avg | 0.81 | 0.85 | 0.81 | 50736 |

- Upload the csv files to the jupyter also, so that it will be easier to compare for large processing such as comparisons of three models in a single notebook instance

jupyter

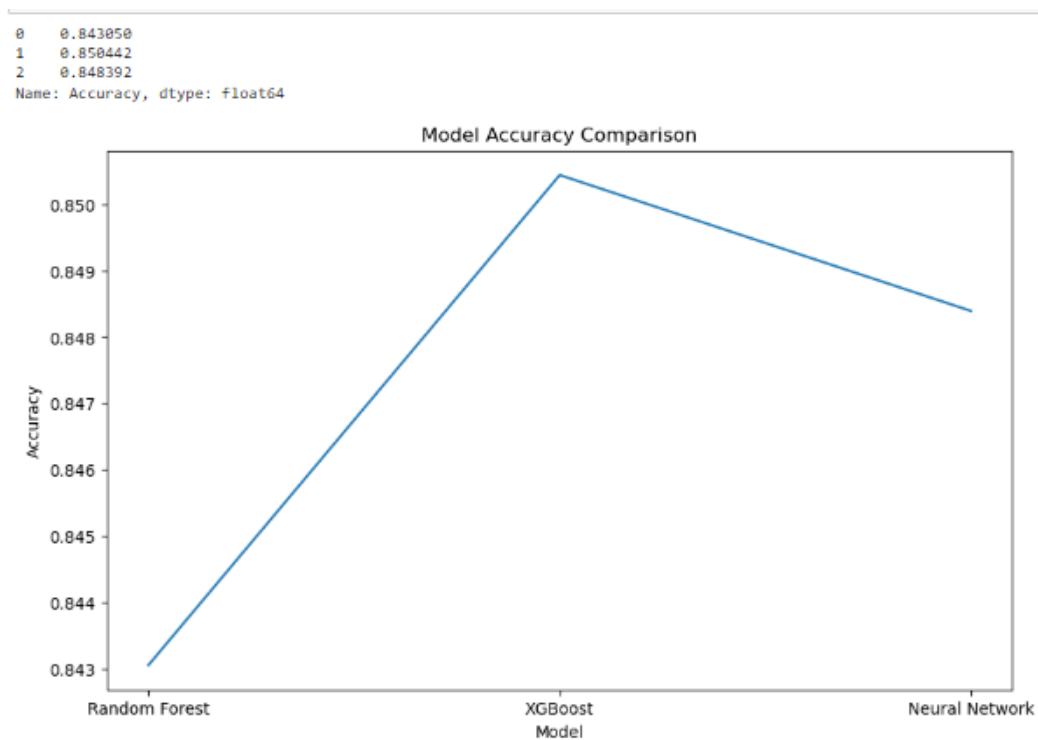
Open JupyterLab Quit Logout

Files Running Clusters SageMaker Examples Conda

Select items to perform actions on them. Upload New ↺

| | Name | Last Modified | File size |
|--------------------------|---|---------------------|-----------|
| <input type="checkbox"/> | / | | |
| <input type="checkbox"/> | Comparison of the three at one instance.ipynb | Running an hour ago | 1.47 MB |
| <input type="checkbox"/> | distributed_keras.ipynb | Running 11 days ago | 7.94 kB |
| <input type="checkbox"/> | distributed_randomforest_instance.ipynb | Running 11 days ago | 3.76 kB |
| <input type="checkbox"/> | distributed_xGBoost_instance.ipynb | Running 11 days ago | 12 kB |
| <input type="checkbox"/> | diabetes_012_health_indicators_BRFSS2015.csv | 17 days ago | 22.7 MB |
| <input type="checkbox"/> | diabetes_binary_5050split_health_indicators_BRFSS2015.csv | 17 days ago | 6.35 MB |
| <input type="checkbox"/> | diabetes_binary_health_indicators_BRFSS2015.csv | 17 days ago | 22.7 MB |
| <input type="checkbox"/> | Train_data.csv | 17 days ago | 2.88 MB |

- Now run the comparative instance, containing the comparison of the three model at once, and observe the results



6 Video Demonstration

Link : <https://youtu.be/FdAEwxGc0XA>