Configuration Manual

MSc Research Project
MSc in Cloud Computing

Amit Goswami
x21152594

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Amit Goswami

Student ID: x21152594

Programme: Cloud Computing **Year:** 2022-23

Module: MSc Research Project

Lecturer: Vikas Sahni

Submission Due Date: 01/Feb/2023

Project Title: Configuration Manual

Page Count: 12

Word Count: 1263

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: ……
…

Date: 31/Jan/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□

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.	□

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

Amit Goswami
x21152594

1. Introduction

The configuration manual describes how to implement dynamic load balancing algorithm and resource management using machine learning. It also includes the general setup used for installing the project's required tools. This configuration manual will help academic students and other researchers gain a better understanding of the method used to carry out thisresearch project.

2. Prerequisites

The following prerequisites which are used for this project:

- **AWS EC2 Instance** - EC2 is one of the wide range of services offered by Amazon which is also considered as the basic computing component available in the technology stack. EC2 provides a secured and highly scalable computing capacity which assists developers. Moreover, it can be easily combined with a lot of services, significantly helps in scaling up and down, and has paid service as well.

- **Python** - Artificial intelligence facilitates computers to learn and understand in the absence of explicit and certain programming. The main aim of machine learning is to develop computer programs that can better adapt to new data. Moreover, this article talks about the fundamentals of machine learning and the use of python language in algorithms of machine learning.

- **Jupyter notebook** - It is a user interface built in Python language that enables users to execute web server tasks and submit code solutions within an organised input/ output cells list.

- **Anaconda** - In data sciences, Anaconda is referred to as an open-source Python and R distribution system which facilitates users in order to easily manage and deploy the packages. Anaconda utilises a package management system known as Conda, it supervises the package versions. Moreover, it thoroughly inspects the environment before beginning an installation in order to avoid clash with any other packages or frameworks.

3. Libraries/Packages Used

- **Pandas** - It is an open-source Python based package applicable in machine learning tasks and data analysis or data science. It is constructed on NumPy that assists multidimensional arrays.

- **NumPy** - A Python programming language library that allows more data storage with less memory. NumPy, which includes a multidimensional array and other resources, enables Python programmers to store numbers efficiently.

- **Seaborn** - A matplotlib based open source library that works in Python programming language which can be utilised for data visualisation and data analysis. Seaborn is simple to use with dataframes and the Pandas library. The generated graphs can also be easily customised. Here are a few advantages of data visualisation.

- **hvPlot** - hvPlot is a HoloViews-based high-level plotting API that offers a mainstream and steady API essential for data plotting in all the formats available.

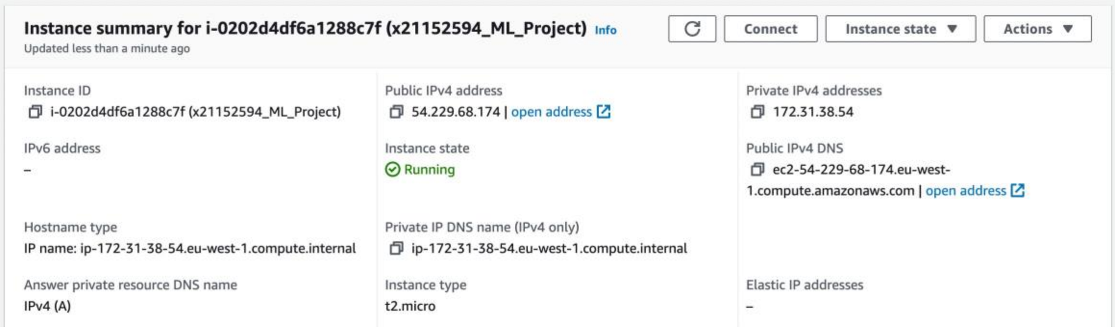
- **Sklearn** - It is a Python library that contains many unsupervised and supervised learning algorithms. It is based on technology that you may be familiar with, such as NumPy, pandas, and Matplotlib!

- **Tensorflow** - A Python based open source library that facilitates numerical computations which helps in making neural networks and learning machine algorithms simpler and faster.

4. Configuration Steps

Step 1: Creating EC2 instance AWS

Link : https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html



Instance summary for i-0202d4df6a1288c7f (x21152594_ML_Project)

Updated less than a minute ago

Instance ID: i-0202d4df6a1288c7f (x21152594_ML_Project)

Public IPv4 address: 54.229.68.174 | open address

Private IPv4 addresses: 172.31.38.54

IPv6 address: –

Instance state: Running

Public IPv4 DNS: ec2-54-229-68-174.eu-west-1.compute.amazonaws.com | open address

Hostname type: IP name: ip-172-31-38-54.eu-west-1.compute.internal

Private IP DNS name (IPv4 only): ip-172-31-38-54.eu-west-1.compute.internal

Answer private resource DNS name: IPv4 (A)

Instance type: t2.micro

Elastic IP addresses: –

Step 2: Installation Jupyter Notebook

It is necessary to install Jupyter after forming connections with the EC2. One of the easiest ways to conduct this is to just simply download an Anaconda distribution that correlates with the Operation System (In case the user is operating on Ubuntu, mentioned below is one of the Linux versions).

It is necessary to copy and run the installer link in order to download the Anaconda distribution to EC2.
wget https://repo.anaconda.com/archive/Anaconda3-2019.03-Linux-x86_64.sh

(Above mentioned link is categorised as the recent versions available) In order to install Jupyter Notebook, it is important to run bash on the downloaded file: bash Anaconda3- 2019.03-Linux-x86_64.sh

Step 3: Configuring the path of Jupyter Notebook

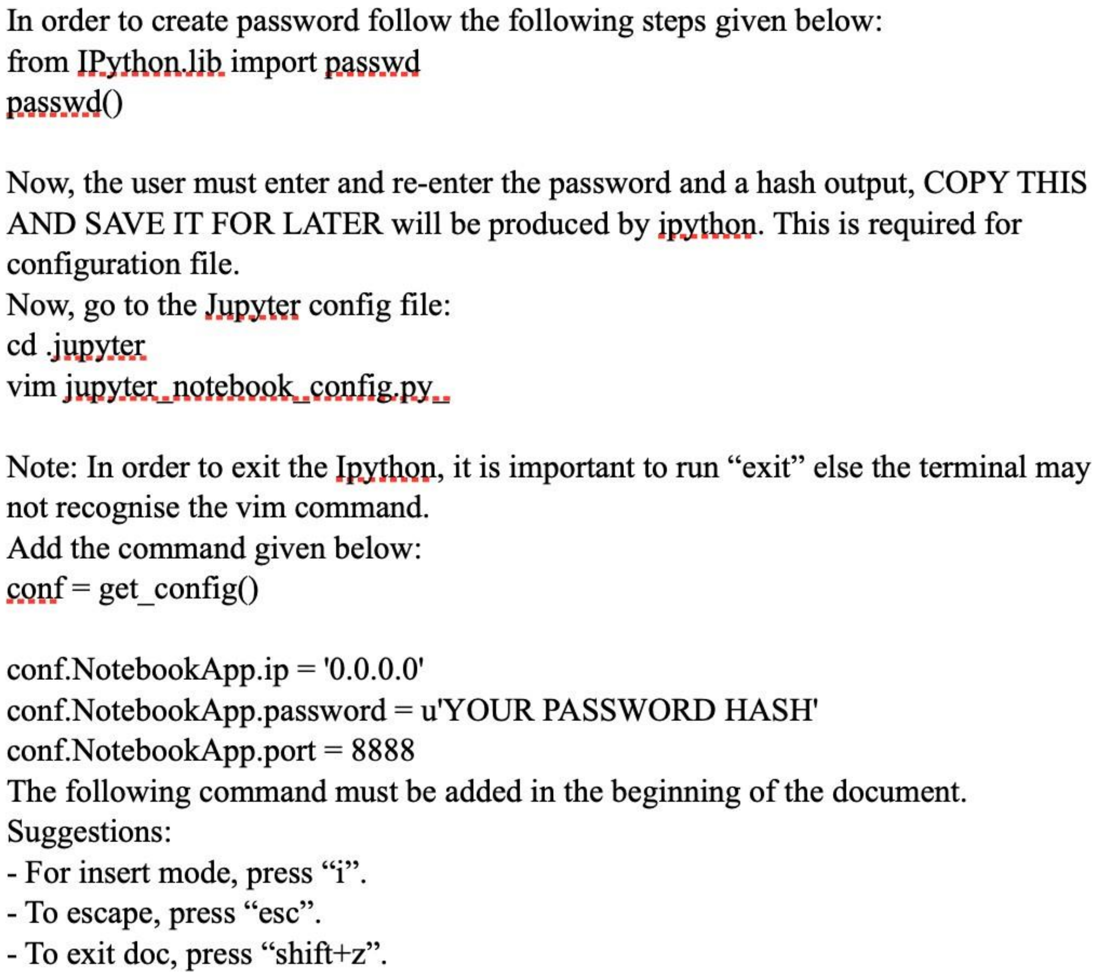
Initially, It is mandatory to integrate Jupyter to the path of the system. Moreover, it can be ascertained if it exists on the path already by running: which python, It will be necessary to add the path, if no path is returned. Now, attach the following code to your .bashrc file in order to attach Jupyter functionality to the terminal:

export PATH=/home/ubuntu/anaconda3/bin:$PATH
Now, you need to run the following code after successfully saving the change:
source .bashrc

Finally, run which python as it will return the path in the folder called Python in Anaconda itself

Step 4: Configuration of Jupyter Notebook settings

In order to construct Jupyter configuration file, it is necessary to run the following code:



In order to create password follow the following steps given below:

```
from IPython.lib import passwd
passwd()
```

Now, the user must enter and re-enter the password and a hash output, COPY THIS AND SAVE IT FOR LATER will be produced by ipython. This is required for configuration file.

Now, go to the Jupyter config file:

```
cd .jupyter
vim jupyter_notebook_config.py_
```

Note: In order to exit the Ipython, it is important to run “exit” else the terminal may not recognise the vim command.

Add the command given below:

```
conf = get_config()

conf.NotebookApp.ip = '0.0.0.0'
conf.NotebookApp.password = u'YOUR PASSWORD HASH'
conf.NotebookApp.port = 8888
```

The following command must be added in the beginning of the document.

Suggestions:

- For insert mode, press “i”.
- To escape, press “esc”.
- To exit doc, press “shift+z”.

Step 5: Construct a directory for notebooks

In this step, run the following code given below so as to create a folder to keep all the Jupyter Notebooks.

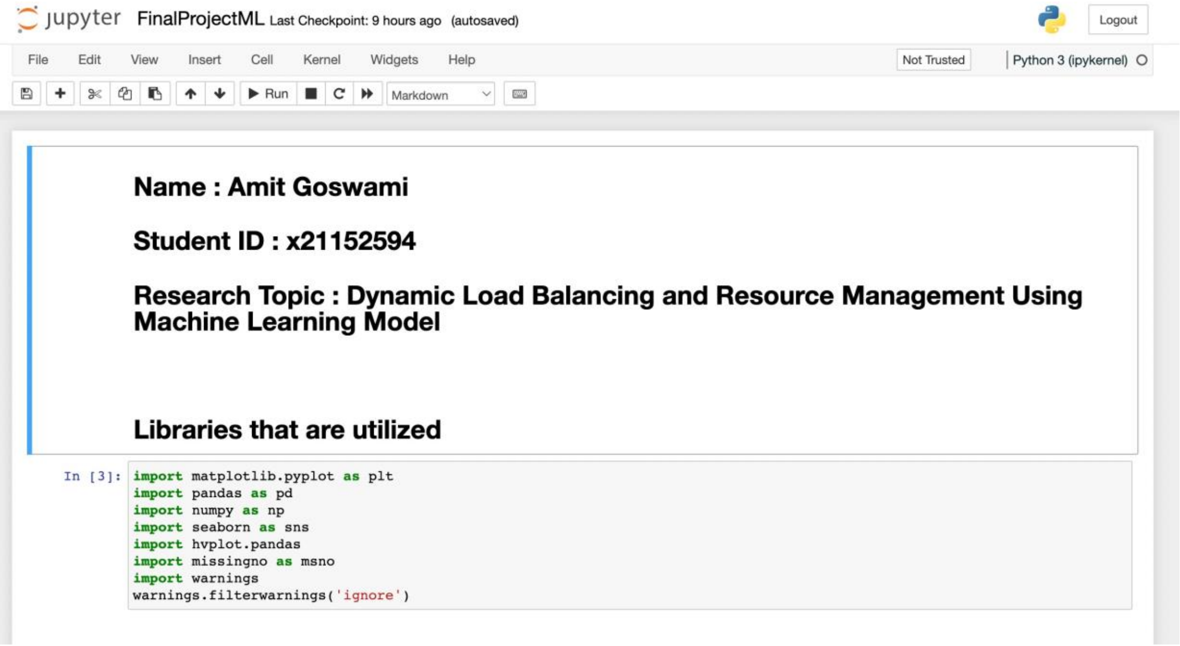
```
mkdir MyNotebooks
```

This is one of the simplest steps and this folder can be named anything like “MyNotebooks”.

Step 6: Connecting to EC2 Jupyter server

Now, it is easy to run the Jupyter notebook and simultaneously gain access to the EC2 server. Now, run the following command so as to run the notebook:

https://(your AWS public dns):8888/



Name : Amit Goswami

Student ID : x21152594

Research Topic : Dynamic Load Balancing and Resource Management Using Machine Learning Model

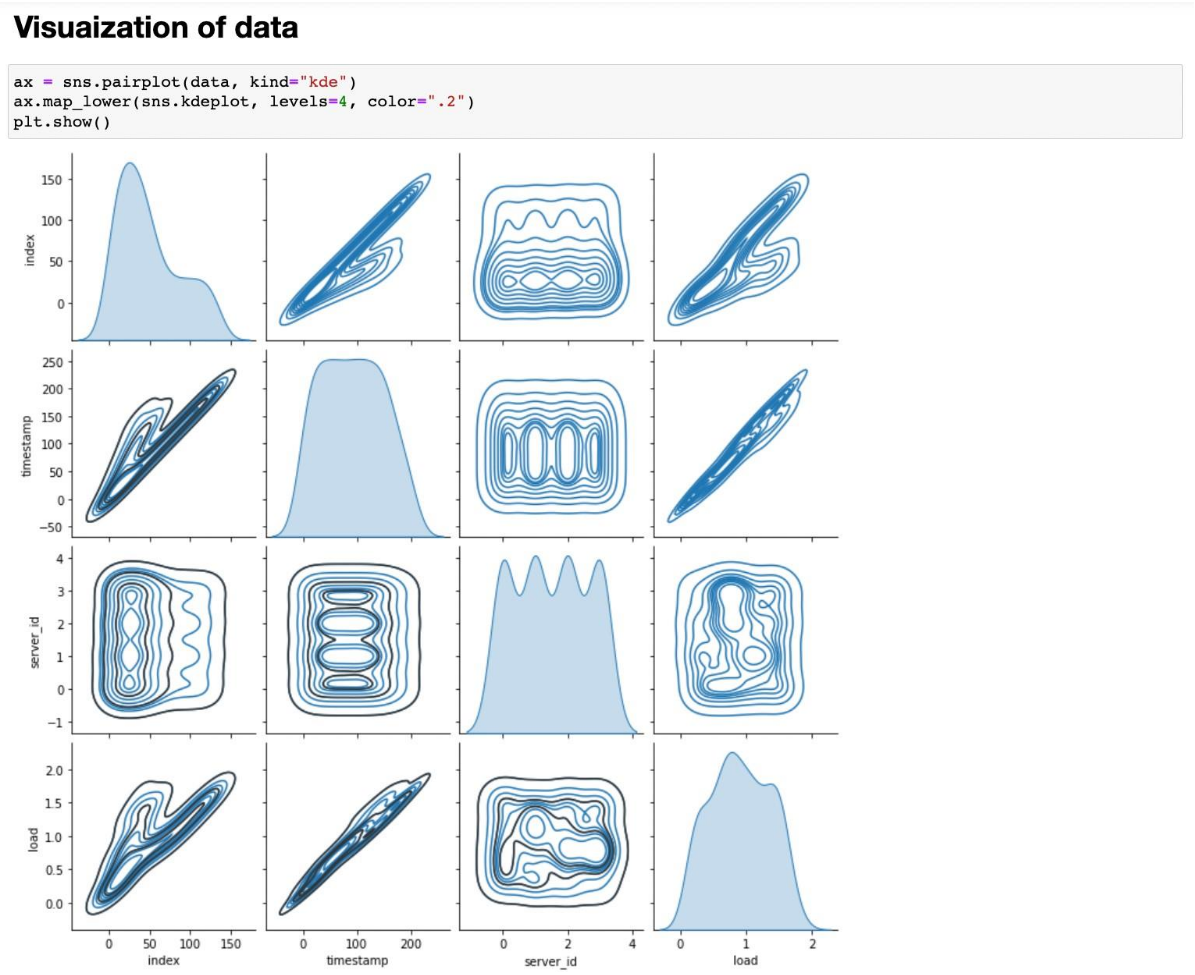
Libraries that are utilized

In [3]:
```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
import hvplot.pandas
import missingno as msno
import warnings
warnings.filterwarnings('ignore')
```

Step 7: Installing TensorFlow from the source for Anaconda

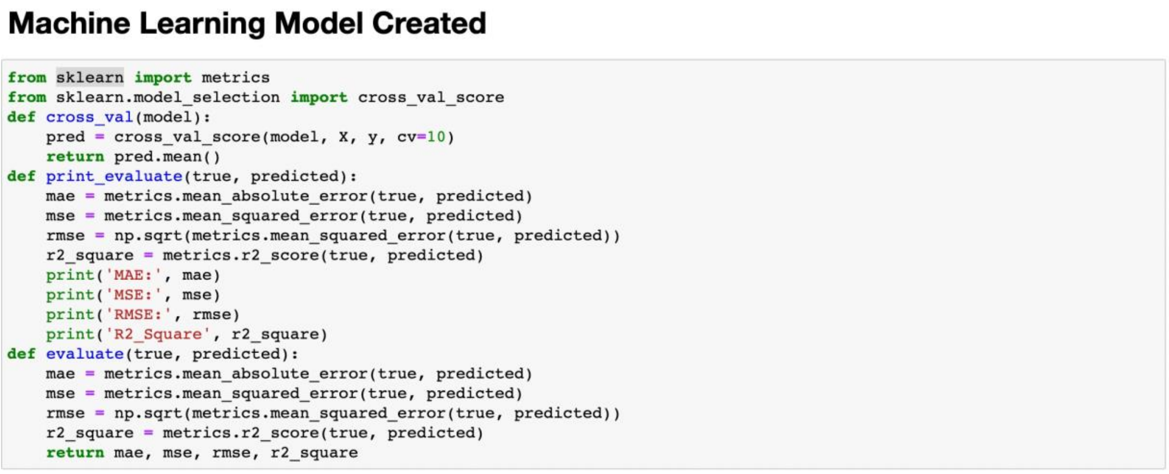
- Open Anaconda prompt.
- Now, construct a conda environment and it could be named anything like tensorflow using the command:conda create -n tensorflow
- Initiate the conda environment.activate tensorflow
- Now, Install tensorflow using the following command.conda install -c anaconda tensorflow-gpu

5. Implementation Steps



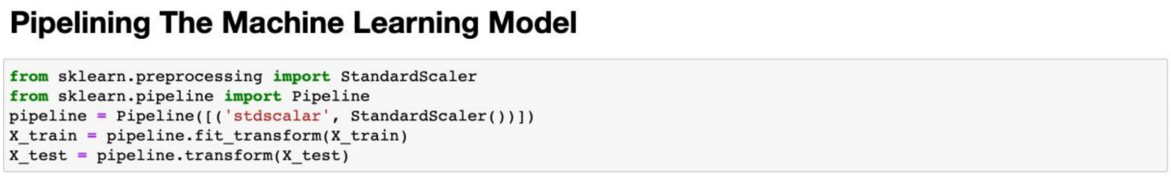
Visuaization of data

```
ax = sns.pairplot(data, kind="kde")
ax.map_lower(sns.kdeplot, levels=4, color=".2")
plt.show()
```



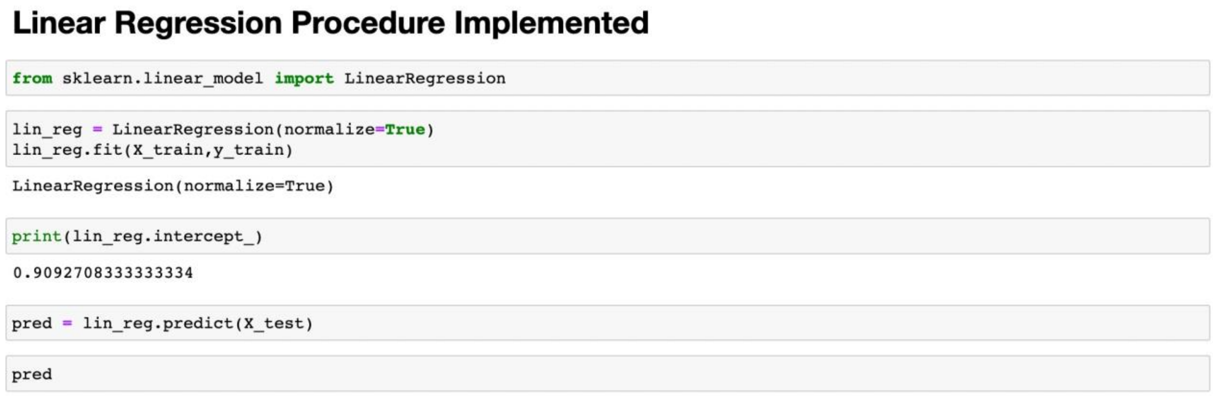
Machine Learning Model Created

```
from sklearn import metrics
from sklearn.model_selection import cross_val_score
def cross_val(model):
    pred = cross_val_score(model, X, y, cv=10)
    return pred.mean()
def print_evaluate(true, predicted):
    mae = metrics.mean_absolute_error(true, predicted)
    mse = metrics.mean_squared_error(true, predicted)
    rmse = np.sqrt(metrics.mean_squared_error(true, predicted))
    r2_square = metrics.r2_score(true, predicted)
    print('MAE:', mae)
    print('MSE:', mse)
    print('RMSE:', rmse)
    print('R2_Square', r2_square)
def evaluate(true, predicted):
    mae = metrics.mean_absolute_error(true, predicted)
    mse = metrics.mean_squared_error(true, predicted)
    rmse = np.sqrt(metrics.mean_squared_error(true, predicted))
    r2_square = metrics.r2_score(true, predicted)
    return mae, mse, rmse, r2_square
```



Pipelining The Machine Learning Model

```
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
pipeline = Pipeline([('stdscalar', StandardScaler())])
X_train = pipeline.fit_transform(X_train)
X_test = pipeline.transform(X_test)
```



Linear Regression Procedure Implemented

```
from sklearn.linear_model import LinearRegression
```

```
lin_reg = LinearRegression(normalize=True)
lin_reg.fit(X_train,y_train)
```

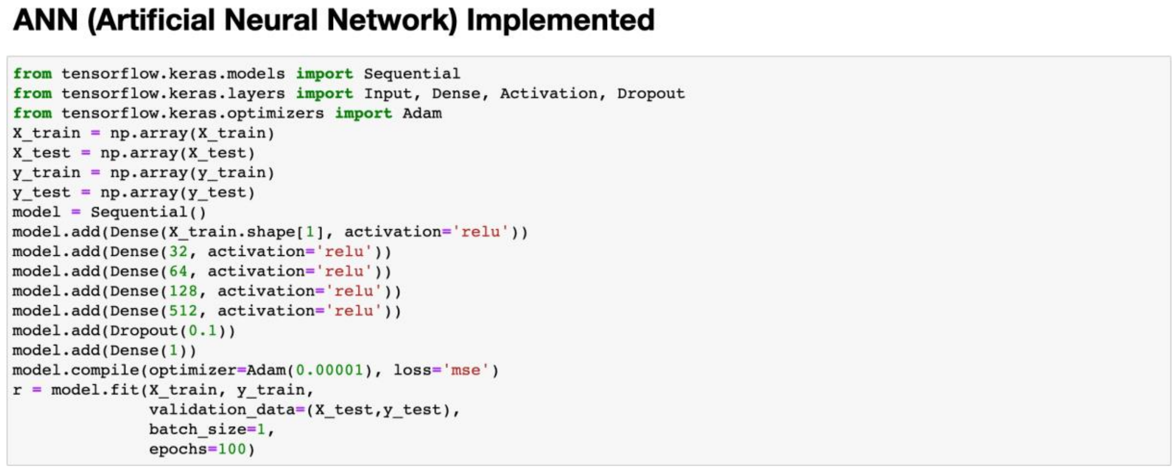
```
LinearRegression(normalize=True)
```

```
print(lin_reg.intercept_)
```

```
0.9092708333333334
```

```
pred = lin_reg.predict(X_test)
```

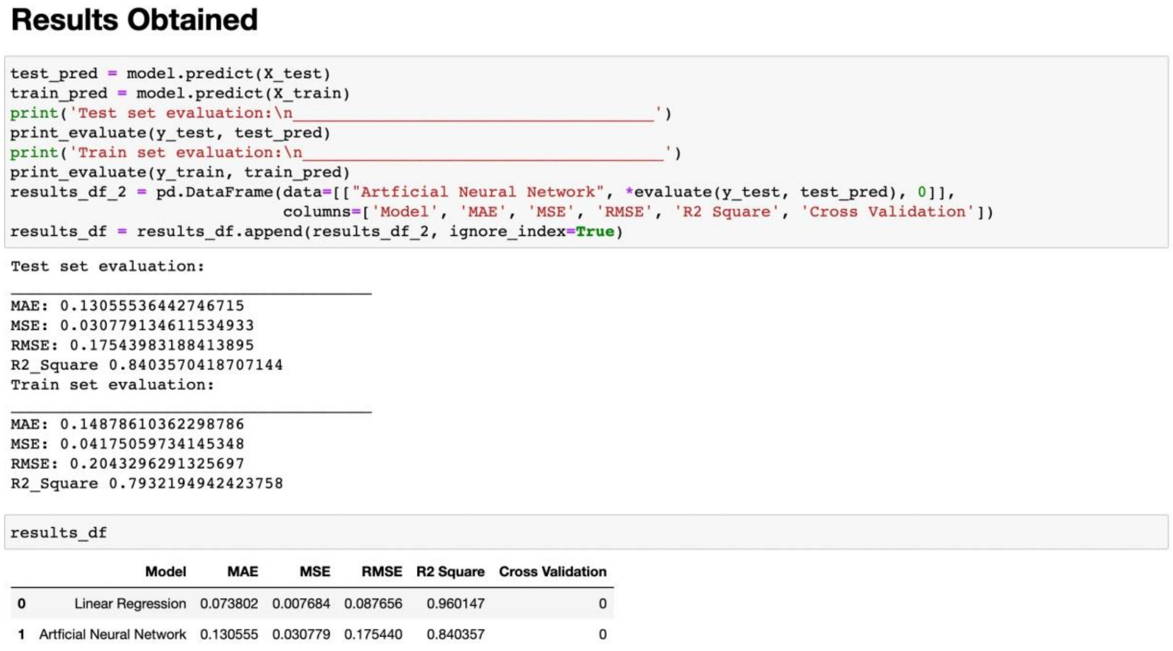
```
pred
```



ANN (Artificial Neural Network) Implemented

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Input, Dense, Activation, Dropout
from tensorflow.keras.optimizers import Adam
X_train = np.array(X_train)
X_test = np.array(X_test)
y_train = np.array(y_train)
y_test = np.array(y_test)
model = Sequential()
model.add(Dense(X_train.shape[1], activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(128, activation='relu'))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(1))
model.compile(optimizer=Adam(0.00001), loss='mse')
r = model.fit(X_train, y_train,
              validation_data=(X_test,y_test),
              batch_size=1,
              epochs=100)
```

6. Compared Results



Results Obtained

```
test_pred = model.predict(X_test)
train_pred = model.predict(X_train)
print('Test set evaluation:\n_____________________________________')
print_evaluate(y_test, test_pred)
print('Train set evaluation:\n_____________________________________')
print_evaluate(y_train, train_pred)
results_df_2 = pd.DataFrame(data=[["Artficial Neural Network", *evaluate(y_test, test_pred), 0]],
                            columns=['Model', 'MAE', 'MSE', 'RMSE', 'R2 Square', 'Cross Validation'])
results_df = results_df.append(results_df_2, ignore_index=True)
```

```
Test set evaluation:
_____________________________________
MAE: 0.13055536442746715
MSE: 0.030779134611534933
RMSE: 0.17543983188413895
R2_Square 0.8403570418707144
Train set evaluation:
_____________________________________
MAE: 0.14878610362298786
MSE: 0.04175059734145348
RMSE: 0.2043296291325697
R2_Square 0.7932194942423758
```

```
results_df
```

	Model	MAE	MSE	RMSE	R2 Square	Cross Validation
0	Linear Regression	0.073802	0.007684	0.087656	0.960147	0
1	Artficial Neural Network	0.130555	0.030779	0.175440	0.840357	0

7. Conclusion

The R2 Square value of the Artificial Neural Network algorithm is better than that of the linear regression algorithm for predicting the load, and this configuration manual provides instructions for recreating the same environment that was used in the research that found that the Artificial Neural Network predicts better than the linear regression algorithm when passed through the machine learning model.

References

Vichi, M. (2015). *AWS console:* Amazon. Retrieved December 14, 2022, from https://aws.amazon.com/console/

Project jupyter. Project Jupyter. (n.d.). Retrieved December 14, 2022, from https://jupyter.org/

Python Setup. Python.org. (n.d.). Retrieved December 14, 2022, from https://www.python.org/downloads/

Create production-grade machine learning models with tensorflow. TensorFlow. (n.d.). Retrieved December 14, 2022, from https://www.tensorflow.org/?gclid=CjwKCAiAheacBhB8EiwAItVO26SlzQng1wd33k7v_GcdC_SaQIehwdqksXoL1DPTGd07rW7RSS55yRoCJXUQAvD_BwE

NumPy. (n.d.). Retrieved December 14, 2022, from https://numpy.org/

Pandas. pandas. (n.d.). Retrieved December 14, 2022, from https://pandas.pydata.org/