**ASSIGNMENT 3**

**Info 7390 Advanced Data Science and architecture**

****

**Documentation**

*Presented by:*

*Dhanisha Damodar Phadate*

*001859234*

[*phadate.d@husky.neu.edu*](file:///C:\Users\palak\Downloads\phadate.d@husky.neu.edu)

*Palak Sharma*

*001828562*

[*sharma.pala@husky.neu.edu*](file:///C:\Users\palak\Downloads\sharma.pala@husky.neu.edu)

*Dharani Thirumalaisamy*

*001887922*

*thirumalaisamy.d@husky.neu.edu*

**CONTENTS**

1. Part 1: Pipelining ........................ 3- 18

a) Generation of error metrics with ranks ......................... 3-12

b) Creating pickle files for each model ........................13-17

c) Creating docker image ............................18

2. Part 2:Model Deployment .........................19-

a) Firing up an EC2 instance on AWS. .........................20-

b) Create Flask application using REST API .........................21-

c) Uploading the inputs taken from the user to S3 .........................22-

d) Using the pickle file of the pipeline present on S3 to train the data given by the user.

e) Returning Back an error metrics for each Model used.

f) Also providing the user with an option to enter a row of value and give the predicted value for 2 decay coefficient.

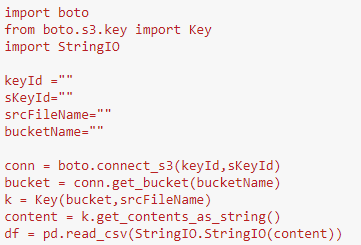
g) Authentication for the user is included as well.

**PART 1 : PIPELING**

**GENERATION OF ERROR METRICS WITH RANKS**

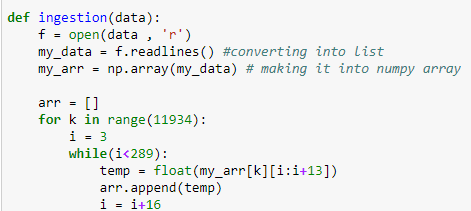
**STEP 1**: Import all the necessary packages from the python libraries

**STEP 2**: Import the data from Amazon S3



These commands will retrieve the data from the Amazon web service.

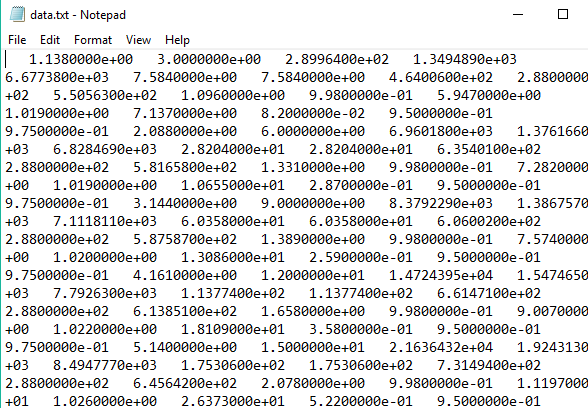
**STEP 3**: Ingestion of data



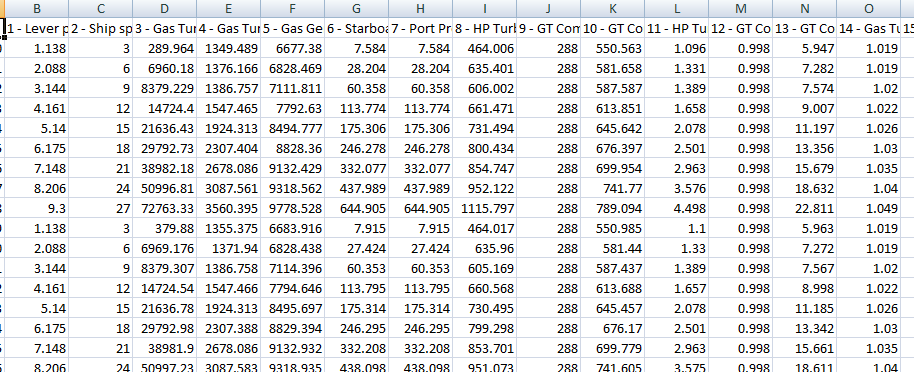
The data is in .txt format which is converted to proper format so that it could be read as a csv file.

The final dataset is stored in final\_dataset.csv file.

Given dataset :



Final\_dataset.csv :

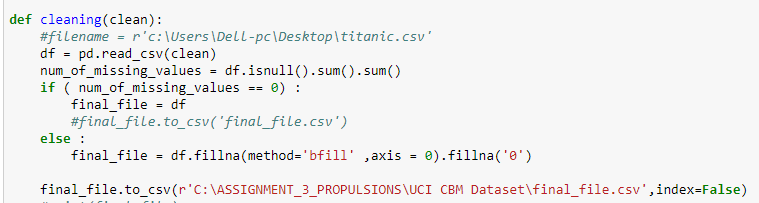


This is the final dataset with which all the other preprocessing techniques are carried out.

The dataset had 18 columns and 11,934 rows.

**STEP 4**: Data Cleaning

It is important to check for the null values and replace it with other values or '0's'.

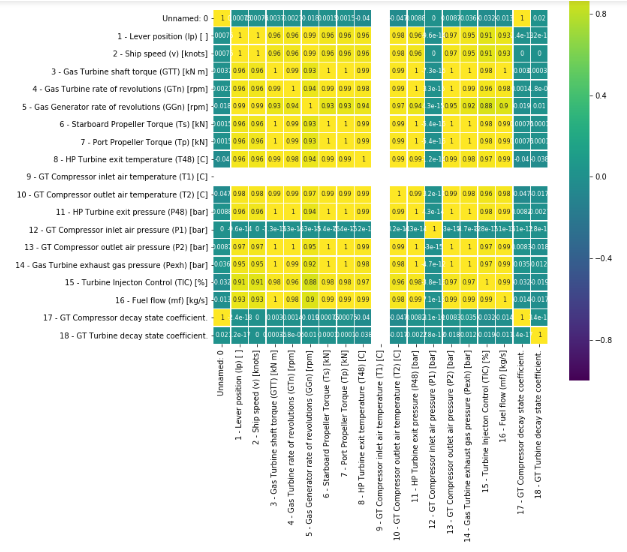
This piece of code will check for null values in the dataset and if it finds any it will replace those values with the one after it.

As our dataset doesn't have any null values , the dataset remains the same.

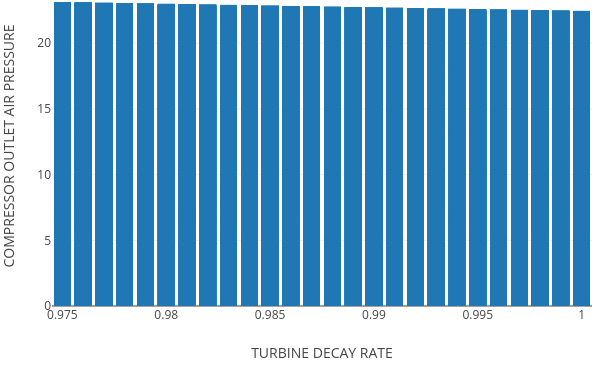
**STEP 5**: Exploratory Data Analysis

Seaborn and Plotly are used to visualize the dataset but no important information can be drawn for them as all the graphs are similar.

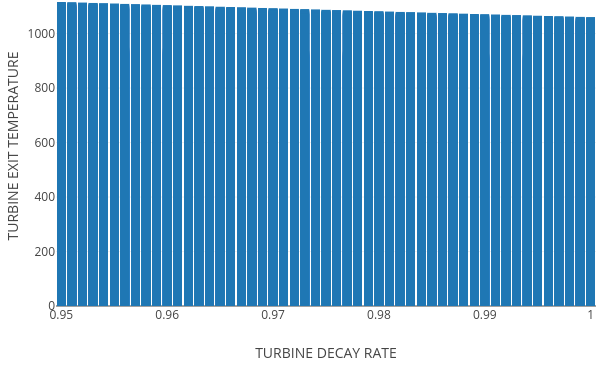
Heatmap :



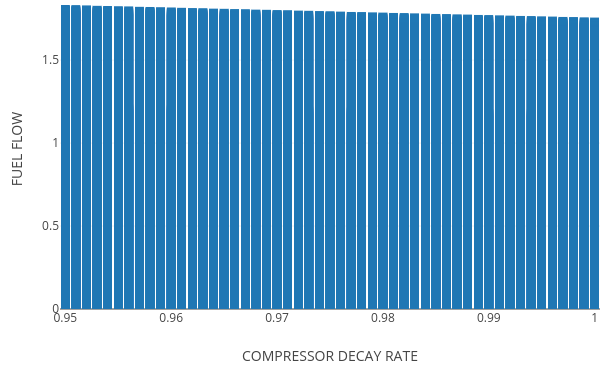
Bar charts :



Here we can see that , as Turbine decay rate increases the compressor outlet air pressure decreases which shows a inverse relation.



Similar is the case with Turbine external temperature too.



The above graph shows that the compressor decay rate is inversely proportional to fuel flow.

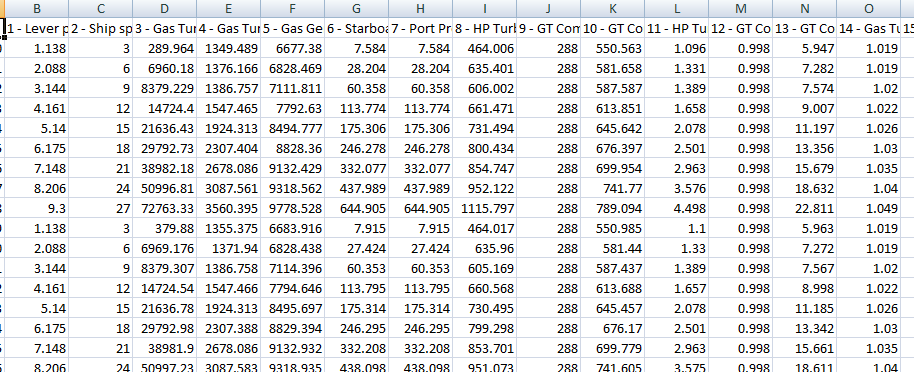
As most of the graphs are like this , no concrete conclusion can be drawn from it. One possible reason could be the outliers.

All these images are saved to the local system as trace0-23.png's.

**STEP 6**: Since the dataset that we have, has outliers. It is necessary to scale them.

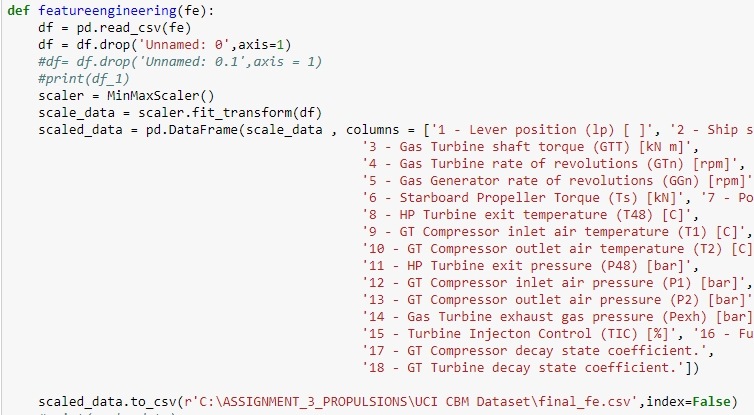
Here we are using min-max scaler and store it in final\_fe.csv.

Final\_dataset.csv :

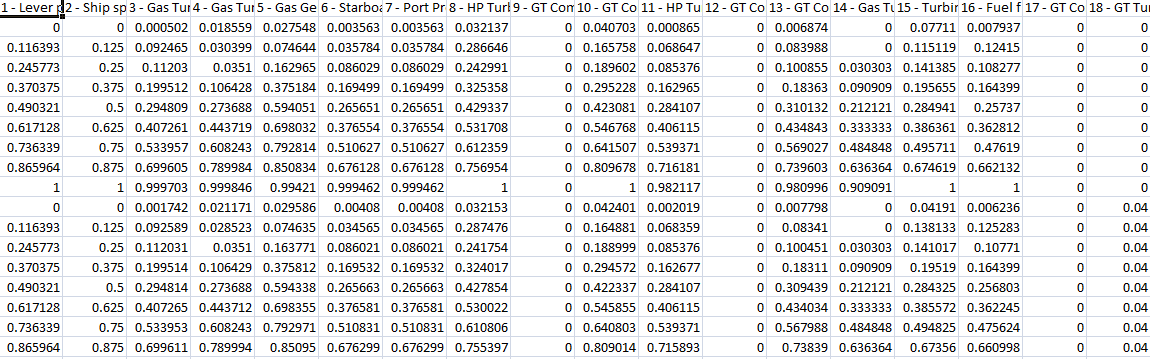


Final\_fe.csv :

code snippet -

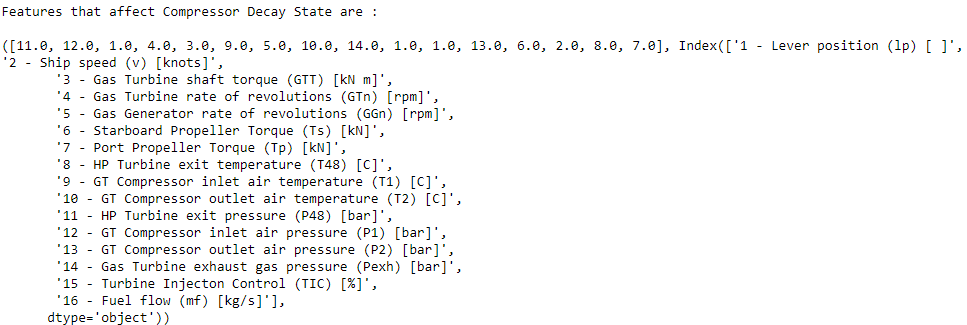


**output -**

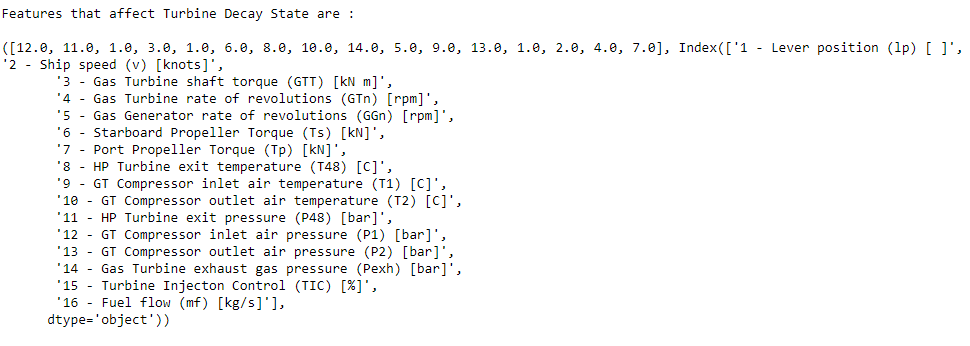


**STEP 7**: Feature Selection

As it is a regression problem , we can go with recursive feature elimination since other methods didn't provide effective results.

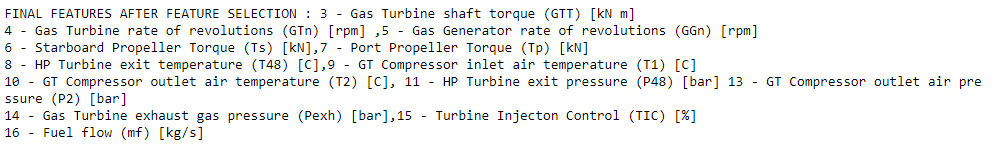


Features with their corresponding ranks. This is for Compressor Decay state.



This is for Turbine decay state.

So combining these two :



**STEP 8** : Modeling

Three types of algorithms are used here :

1. Linear Regression

2. Random Forest Regressor

3. K-nearest neighbor Regressor

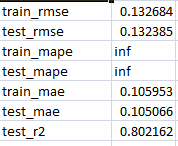
These three algorithms were chosen as it is a regression problem and the error metrics can be compared to find out the best algorithm for this problem.

The error metrics used are : RMSE , MAPE, MAE , R2.

The individual results of each algorithm is saved in separate csv files for more clarification in the local system along with their corresponding pickle file.

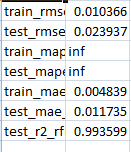
Linear Regression :

Output :



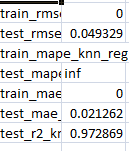
Random Forest Regression :

Output :



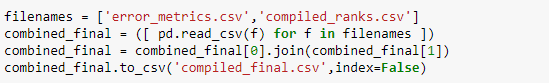
K-Nearest Neighbor :

Output :

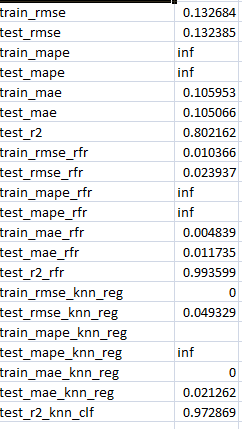


These are the error metrics for each model.

**STEP 9**: Compilation of all the error metrics file and pickle file

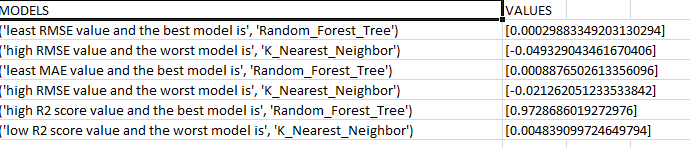


compiled error metrics file :

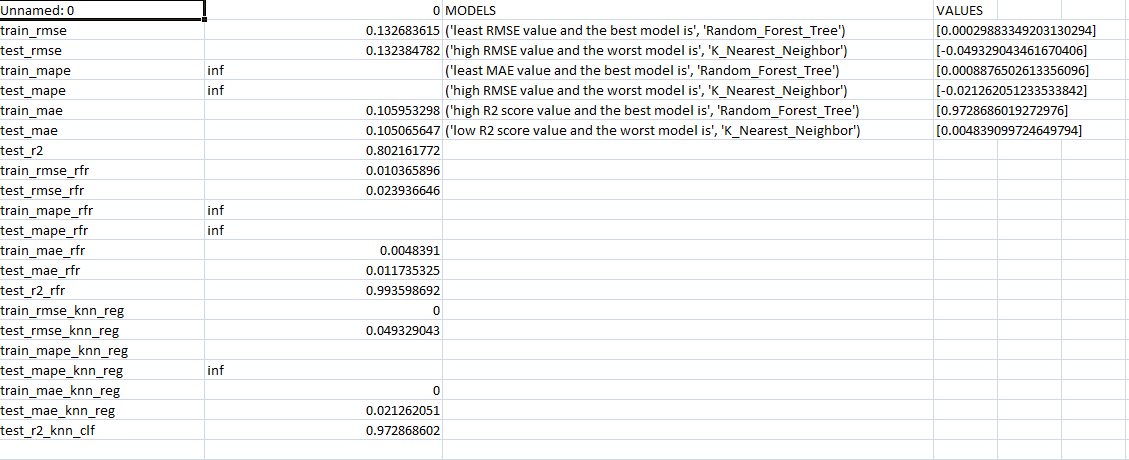


The models are ranked based on their error metrics and are saved in a csv file which is complied with the error metrics csv file to produce the final output.

Ranks :



Final csv after combining with the error metrics csv file (which is compiled\_final.csv):



**STEP 10**: Bucket Upload



These commands , with the amazon access\_key , secret\_access\_key , location, bucket name will push the compiled.csv and the zipped pickle file to S3 platform.

**CREATING PICKLE FILES FOR EACH MODELS**

**STEP 1** : Import all the necessary packages.

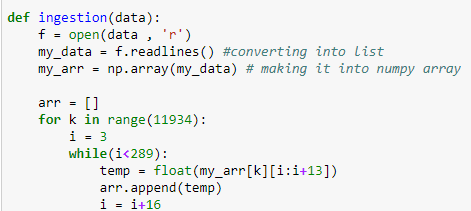
**STEP 2** : Making directories to store the dataset



**STEP 3** : Download the dataset and Extract all the files



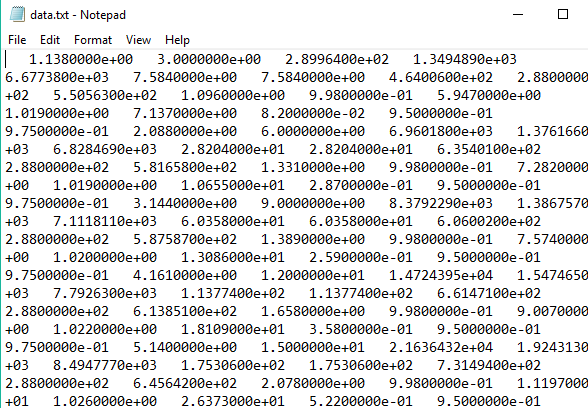
**STEP 4** : Data ingestion



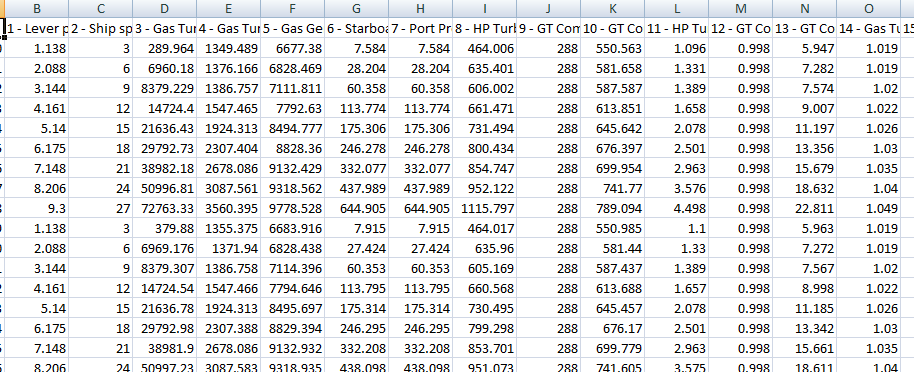
The data is in .txt format which is converted to proper format so that it could be read as a csv file.

The final dataset is stored in final\_dataset.csv file.

Given dataset :



Final\_dataset.csv :

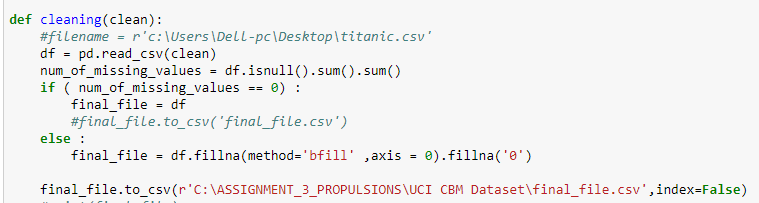


This is the final dataset with which all the other preprocessing techniques are carried out.

The dataset had 18 columns and 11,934 rows.

**STEP 5**: Data Cleaning

It is important to check for the null values and replace it with other values or '0's'.

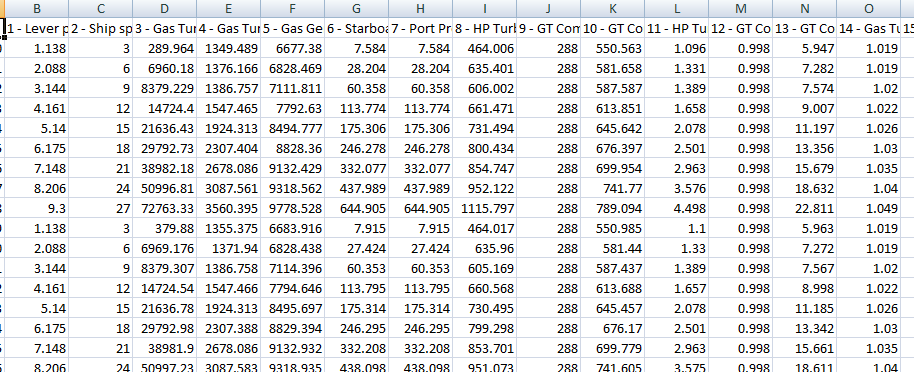
This piece of code will check for null values in the dataset and if it finds any it will replace those values with the one after it.

As our dataset doesn't have any null values , the dataset remains the same.

**STEP 6**: Since the dataset that we have, has outliers. It is necessary to scale them.

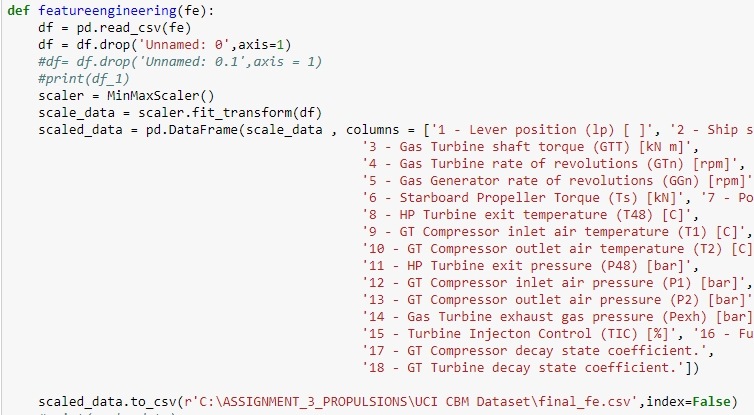
Here we are using min-max scaler and store it in final\_fe.csv.

Final\_dataset.csv :

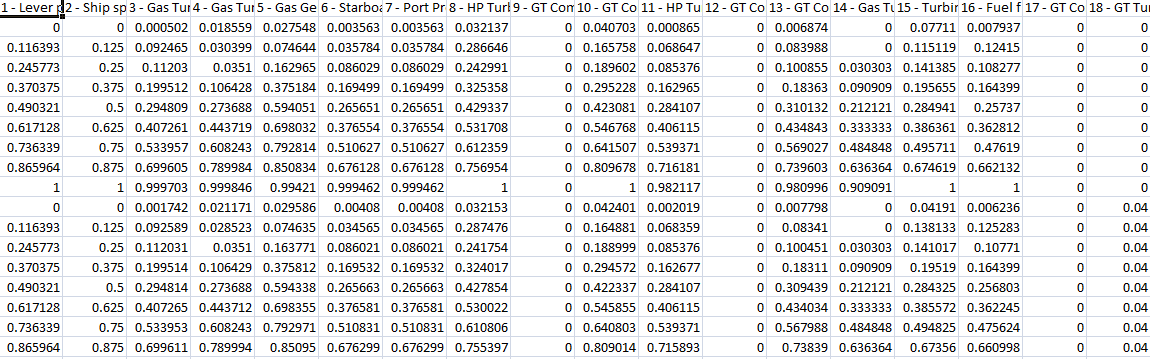


Final\_fe.csv :

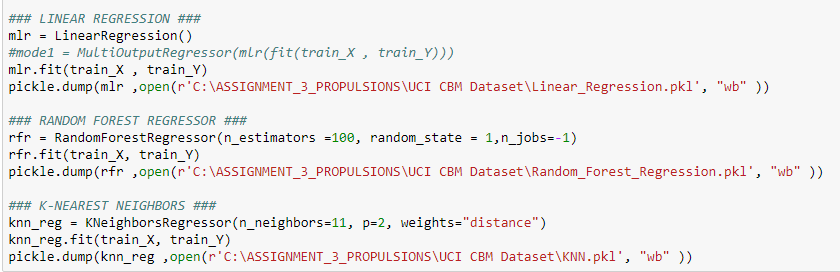
code snippet -



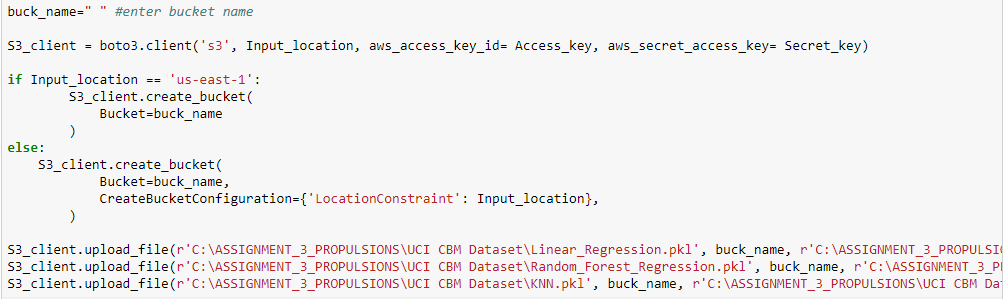
output -



**STEP 7** : Pickling all the models



**STEP 8** : Uploading it to S3



**CREATING DOCKER IMAGE**

**TO BUILD AND RUN DOCKER IMAGE LOCALLY** :

|  |
| --- |
|  |
| Vim dockerfile | | |
|  |
|  |

**TO BUILD THE IMAGE**:

docker build -f dockerfile -t assign3

**TO RUN THE IMAGE TO CREATE CONTAINER**:

docker run -e Access\_key=Access\_key -e Secret\_key=Secret\_key -ti assign3

**TO TAG THE IMAGE:**

docker tag <image id> dhanisha/assign3

**TO PUSH THE DOCKER IMAGE TO DOCKER HUB:**

docker push dhanisha/assign3

**TO PULL THE DOCKER IMAGE FROM THE DOCKER HUB:**

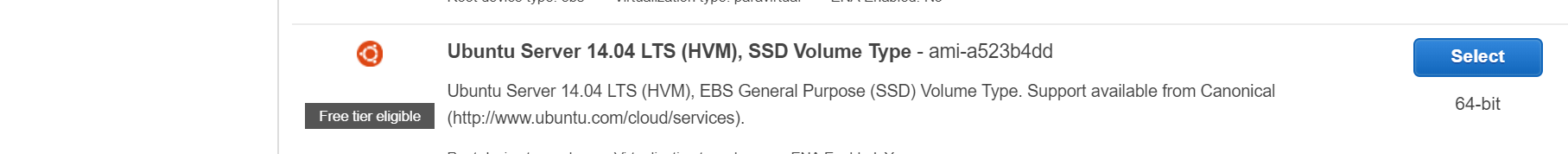
docker pull dhanisha/assign3

**TO RUN THE IMAGE PULLED FROM DOCKER HUB:**

docker run -e Access\_key=Access\_key -e Secret\_key=Secret\_key -ti assign3

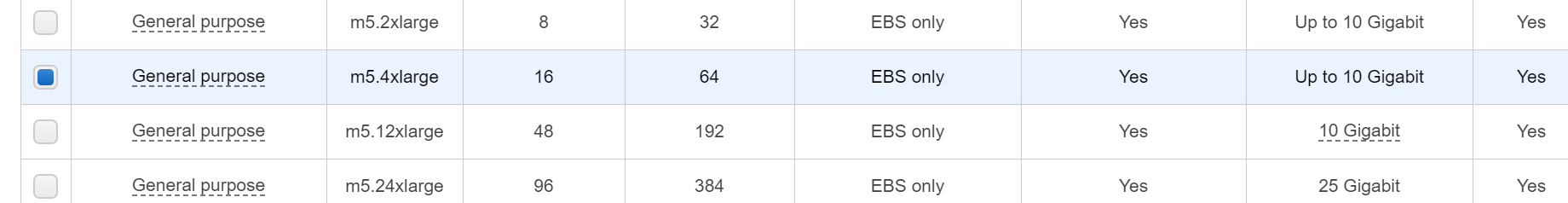
**PART 2 :MODEL DEPLOYMENT**

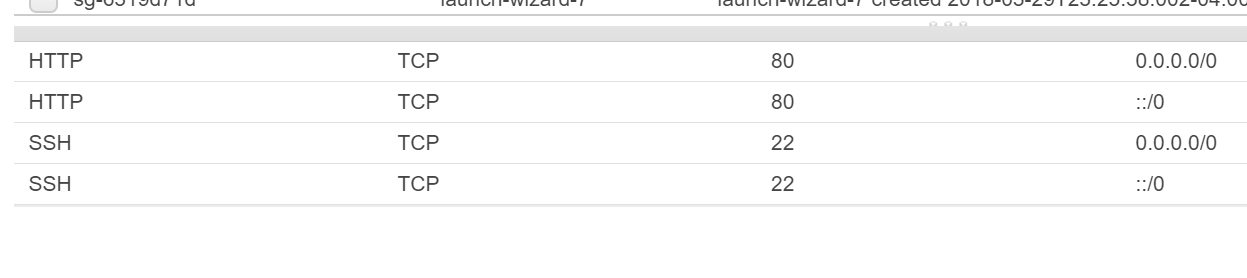
**FIRING INSTANCE ON EC2**



Selecting a 14.04 ubuntu machine to start the server on the ec2 instance.

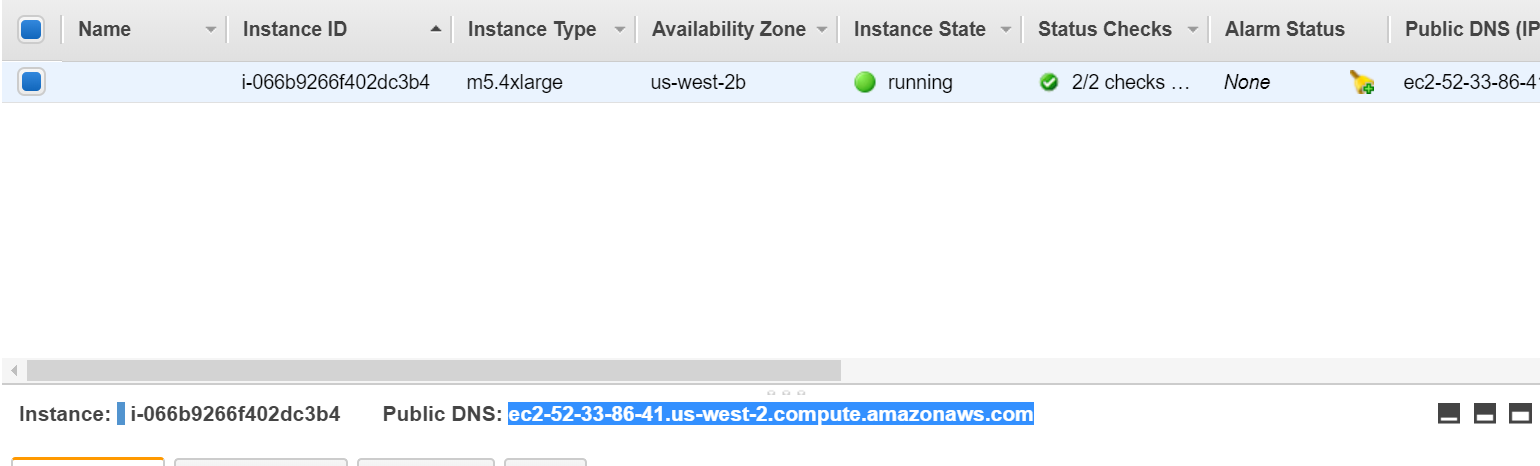
Launching the instance with the following configurations and visibility’





The visibility is set to “ANYWHERE”

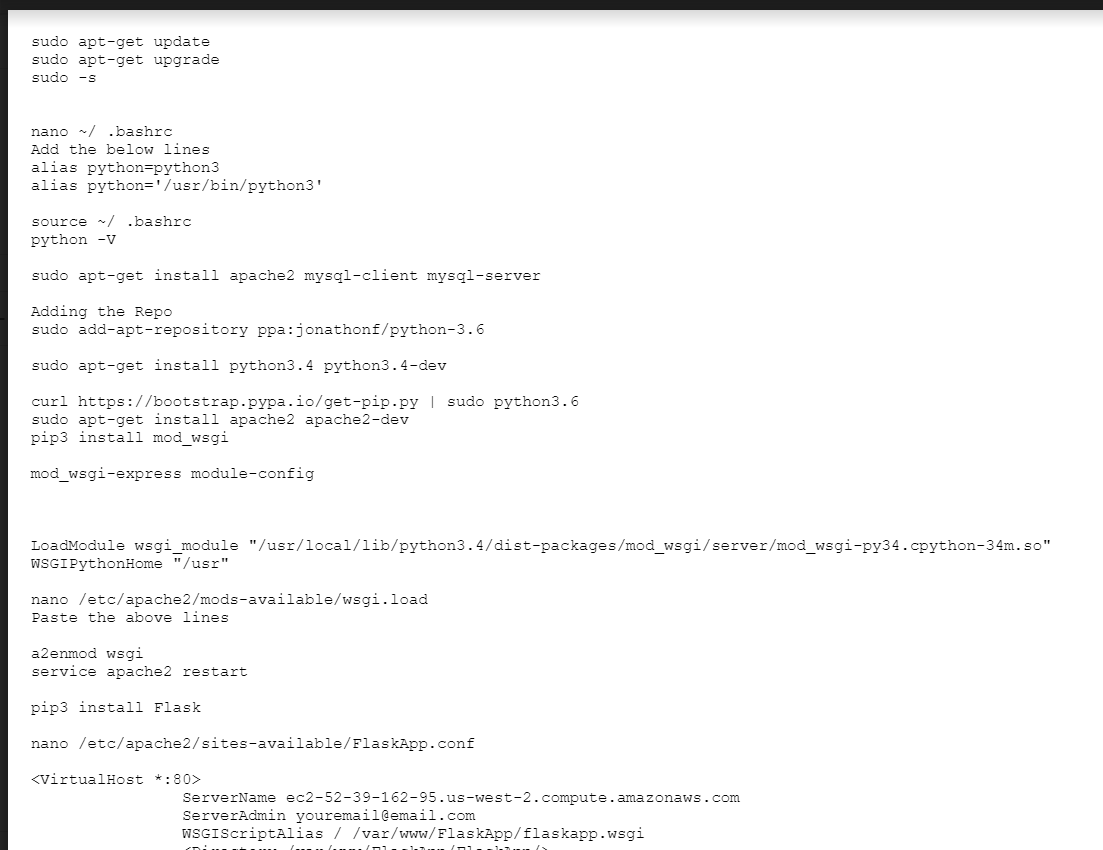
And the instance will be launched with a public DNS and IP address

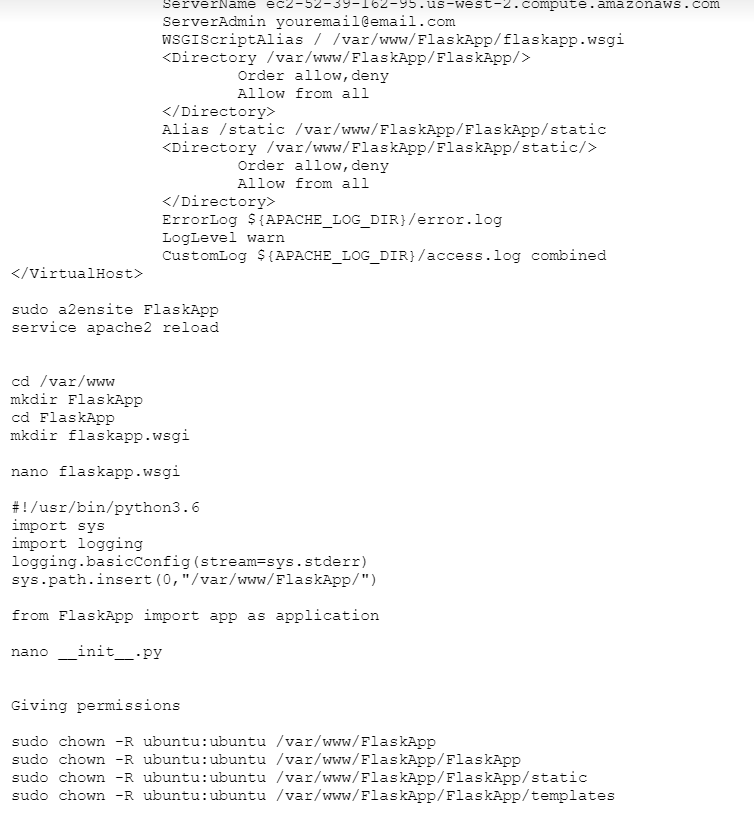


After initiating the instance, putty key is generated and pem and ppk files are generated and given into the authentication of SSH in putty.

ADD THE SCREEN SHOT for PUTTY KEY AND NORMAL PUTTY.

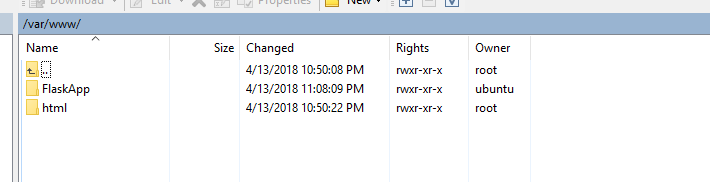
The following set of commands are used for initialization of putty environment.

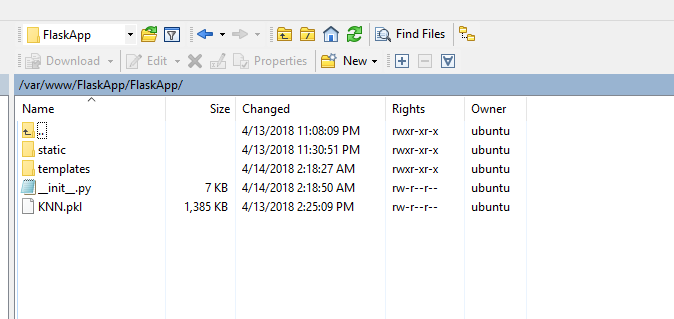


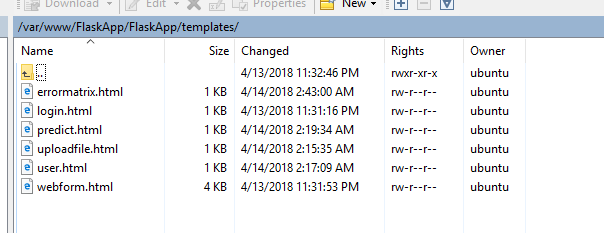


After the completion of flask code, WINSCP is used to access the folders and directories created on the EC2 instance.

The following set of files are created in each folders:







DEPLOYED APP ON EC2 INSTANCE

